Processing

Reference Manual

Part-No: H9776A3

10.04.2002

Bruker software support is available via phone, fax, e-mail, Internet, or ISDN. Please contact your local office, or directly:

Address:	Bruker Analytik GmbH Software Department Silberstreifen D-76287 Rheinstetten Germany
Phone:	+49 (721) 5161 440
Fax:	+49 (721) 5161 480
E-mail:	nmr-software-support@bruker.de
FTP:	ftp.bruker.de / ftp.bruker.com
WWW:	www.bruker.de / www.bruker.com
ISDN:	on request

Contents

Chapter 1

Introduction

1.1 About this manual

This manual is a reference to XWIN-NMR processing commands and parameters. Every command is described on a separate page with its syntax and function as well and its main input/output files and input/output parameters. Most of them are processing commands in the sense that they manipulate the data. The manual, however, also includes several commands that analyse data or send information to the screen or printer.

1.2 Conventions

Font conventions

`abs` - commands to be entered on the command line are in courier bold italic

utilities - commands to be clicked are in times bold italic

`fid` - filenames are in courier

name - any name which is not a filename is in times italic

File/directory conventions

<xwhome> - the XWIN-NMR home directory (default C\:Bruker or /u)

Header conventions

SYNTAX - only included if the command described requires arguments.

USED IN AU PROGRAMS - only included if an AU macro exist for the command described

1.3 About dimensions

XWIN-NMR can process 1, 2 and 3 dimensional data. The dimensions of a dataset are indicated with the terms F3, F2 and F1 which are used as follows:

1D data

F1 - first and only dimension

2D data

F2 - first dimension (acquisition or direct dimension)
F1 - second dimension (indirect dimension)

Commands like ***xf2*** and ***abs2*** work in the F2 dimension. ***xf1***, ***abs1*** etc. work in F1. ***xfb***, ***xtrf*** etc. work in both F2 and F1.

3D data

F3 - first dimension (acquisition or direct dimension)
F2 - second dimension (indirect dimension)
F1 - third dimension (indirect dimension)

Commands like ***tf3*** and ***tabs3*** work in F3. ***tf2***, ***tabs2*** etc. work in F2. ***tf1***, ***tabs1*** etc. work in F1.

1.4 About time and frequency domain data

The result of an acquisition is a representation of intensity values versus acquisition time (seconds); the data are in the time domain. The result of a Fourier transform is a representation of intensity values versus frequency (Hz or ppm); the data are in the frequency domain.

Examples of time domain data are:

- raw data (1D, 2D, and 3D)
- 1D data processed with ***bc***, ***em*** or ***gm***
- 2D data processed with ***xf2*** (time domain in F1)
- 3D data processed with ***tf3*** (time domain in F2 and F1)

Examples of frequency domain data are:

- 1D data processed with ***ft***, ***ef***, ***gf***, ***efp***, ***gfp***, ***trf****
- 2D data processed with ***xfb***, ***xf2***, ***xf1***, ***xtrf****
- 3D data processed ***tf3***, ***tf2***, ***tf1***

Be aware: the commands ***trf**** and ***xtrf**** only perform a Fourier transform if the processing parameter FT_mod (type ***edp***) is set (see ***trf***).

Time and frequency domain data can usually be distinguished by the data type (FID versus spectrum) and axis labelling (Hz or ppm versus sec). The only unequivocal way to distinguish them, however, is the processing status parameter FT_mod (type ***dpp***):

- FT_mod = no : no FT was done and the data are still in the time domain
- FT_mod = f* : FT was done and the data are in the frequency domain
- FT_mod = i* : FT and IFT was done and the data are again in the time domain

1.5 About raw and processed data

The result of an acquisition are raw data. Raw data are data which have not been processed in any way. They are stored in:

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - 1D raw data
`ser` - 2D or 3D raw data

The result of processing are processed data. They are stored in:

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - 1D processed data
`2rr`, `2ir`, `2ri`, `2ii` - 2D processed data

`3rrr`, `3irr`, `3rir`, `3rri` - 3D processed data

Concerning their input data, processing commands can be divided into:

- commands which only work on raw data
- commands which only work on processed data
- commands which work on raw or processed data

1.5.1 Commands which only work on raw data

The following commands only work on raw data. If no raw data exist, they stop with an error message.

- 1D commands ***bc***, ***trf***, ***addfid***, ***convdta***
- 2D commands ***xtrf***, ***xtrf2***, ***convdta***
- 3D commands ***tf3***, ***convdta***

1.5.2 Commands which work on raw data or processed data

The following processing commands work on raw or processed 1D data:

em, ***gm***, ***sinm***, ***qsin***, ***sinc***, ***qsinc***, ***uwm***, ***tm***, ***traf***, ***trafs***, ***ft***, ***ef***, ***gf***, ***efp***, ***gfp***

They work on raw data if one of the following is true:

- no processed data exist (file `1r` and/or `1i` do not exist)
- processed data exist but they are already Fourier transformed

They work on processed data if the following is true:

- processed data exist but they are not Fourier transformed

add, ***addc***, ***and***, ***div***, ***filt***, ***ls***, ***mul***, ***mulc***, ***or***, ***rs***, ***rv***, ***xor***, ***zf***, ***zp***

They work on raw data if the parameter DATMOD = raw

They work on processed data if the parameter DATMOD = processed

The following processing commands work on raw or processed 2D data:

xfb, ***xf2***, ***xf1***

They work on raw data if one of the following is true:

- the option *raw* is added, e.g. ***xfb raw***

- no processed data (i.e. the file `2rr`) exist
- the processing status parameter files `procs` or `proc2s` do not exist or are not readable
- the last processing command on this dataset was interrupted (in this case the entry LOC in the file `dsp.hdr` is negative)
- for ***xf2***: data are already Fourier transformed in F2
- for ***xf1***: data are already Fourier transformed in F1
- for ***xfb***: data are already Fourier transformed in both F2 and F1
- the processing status parameter PH_mod is set to *ps* (power spectrum) or *mc* (magnitude spectrum) in F2 and/or F1

They work on processed data if one of the following is true:

- the option *processed* is used, e.g. ***xfb processed***
- none of the conditions for using raw data is fulfilled

1.5.3 Commands which always work on processed data

Several processing commands can, by definition, only work on processed data. If no processed data exist, they stop with an error message.

On 1D data:

abs, ***absf***, ***absd***, ***apk***, ***apk0***, ***apk1***, ***apks***, ***bcm***, ***sab***, ***trfp***, ***ift***, ***ht***, ***genfid***, ***filt***

On 2D data:

abs2, ***abs1***, ***abst2***, ***abst1***, ***sub2***, ***sub1***, ***sub1d2***, ***sub1d1***, ***bcm2***, ***bcm1***, ***xf2p***, ***xf1p***, ***xfbp***, ***xf2m***, ***xf1m***, ***xfbm***, ***xf2ps***, ***xf1ps***, ***xfbps***, ***sym***, ***syma***, ***symj***, ***tilt***, ***ptilt***, ***ptilt1***, ***rev2***, ***rev1***, ***xif2***, ***xif1***, ***xht2***, ***xht1***, ***xtrfp***, ***xtrfp2***, ***xtrfp1***, ***add2d***, ***genser***

On 3D data:

tf2, ***tf1***, ***tht3***, ***tht2***, ***tht1***,***tf3p***, ***tf2p***, ***tf1p***,***tabs3***, ***tabs2***, ***tabs1***

1.6 About digitally filtered Avance data

The first points of the raw data measured on an Avance spectrometer are called

group delay. These points represent the delay caused by the digital filter and do not contain spectral information. The first points of the group delay are always zero. The group delay only exists if digital filtering is actually used, i.e. if the acquisition parameter DIGMOD is set to digital.

1.7 Usage of processing commands in AU programs

Many processing commands described in this manual can also be used in AU programs. The description of these commands contains an entry USAGE IN AU PROGRAMS. This means an AU macro is available which is usually the name of the command in capitalized letters. If the entry USAGE IN AU PROGRAMS is missing, no AU macro is available. Usually, such a command requires user interaction and it would not make sense to put it in an AU program. However, if you still want to use such a command in AU, you can always use the XCMD macro which takes an XWIN-NMR command as argument. Examples are:

```
XCMD("edp")
XCMD("setdef ackn no")
```

AU programs can be set up with the command ***edau***.

Most XWIN-NMR commands can also be used in an XWIN-NMR macro. These are scripts created with ***edmac*** containing a sequence of XWIN-NMR commands. The syntax of each line is simply an XWIN-NMR command as it would be entered on the command line in lowercase letters.

1.8 Clicking commands from the XWIN-NMR menu

This manuals describes all processing commands as they can be entered on the command line. However, they can also be clicked from the XWIN-NMR popup menus. Most commands can be found under the ***Process***, ***Analysis***, ***Output*** or ***Windows*** menu. The corresponding command line commands are specified in square brackets.

Chapter 2

XWIN-NMR parameters

2.1 About XWIN-NMR parameters

XWIN-NMR parameters are divided in acquisition, processing, plot and output parameters. In this manual, we will mainly concern ourselves with processing parameters.

The following terms are used:

processing parameters

Parameters which must be set, for example with ***edp***, and are then interpreted by processing commands.

acquisition status parameters

Parameters which are set by acquisition commands like ***zg***. They represent the acquisition status of a dataset and can be viewed, for example, with ***dpa***. Some acquisition status parameters are used as input by processing commands.

processing status parameters

Parameters which are set by processing commands. They represent the processing status of a dataset and can be viewed, for example, with ***dpp***. Most processing status parameters get the value of the corresponding processing parameter as

it was set by the user (***edp***). Some parameters, however, are explicitly set or modified by the processing command.

input parameters

Parameters which are interpreted by processing commands. These can be:

- processing parameters (set by the user). Most input parameters are processing parameters.
- acquisition status parameters (set by an acquisition command). An example is parameter AQ_mod.
- processing status parameters (set by the previous processing command). An example is the parameter SI set by ***ft*** and then interpreted by ***abs***. This means you cannot change the size between ***ft*** and ***abs***.

output parameters

Parameters which are set or modified by processing commands. These can be:

- processing status parameters. Examples are FT_mod and YMAX_p, set by ***ft***. Most output parameters are processing status parameters.
- processing parameters. Examples are PHC0 and PHC1, set by ***apk*** and SR and OFFSET, set by ***sref***.

Processing parameters can be set with the parameter editor ***edp*** and processing status parameters can be viewed with ***dpp***. Alternatively, each parameter can be set or viewed by entering its name in lowercase letters on the command line. Here are some examples of how you can set or view the parameter SI:

On a 1D dataset:

- ***si*** - set the parameter SI
- ***1s si*** - view the status parameter SI

On a 2D dataset:

- ***si*** - set the parameter SI in the F2 dimension (= acquisition dimension)
- ***2 si*** - set the parameter SI in the F2 dimension (same as ***si***)
- ***1 si*** - set the parameter SI in the F1 dimension
- ***2s si*** - view the status parameter SI in the F2 dimension
- ***1s si*** - view the status parameter SI in the F1 dimension

On a 3D dataset:

- ***si*** - set the parameter SI in the F3 dimension (= acquisition dimension)
- ***3 si*** - set the parameter SI in the F3 dimension (same as ***si***)
- ***2 si*** - set the parameter SI in the F2 dimension
- ***1 si*** - set the parameter SI in the F1 dimension
- ***3s si*** - view the status parameter SI in the F3 dimension
- ***2s si*** - view the status parameter SI in the F2 dimension
- ***1s si*** - view the status parameter SI in the F1 dimension

Although status parameters are normally not changed by the user, a command like ***1s si*** allows you to do that. This, however, could make the dataset inconsistent. If a processing status parameter is changed with the ***1s*** command, this is reported in the audit trail file `auditp.txt` which resides under the dataset *procno*. Likewise, if an acquisition status parameter is changed with ***1s***, this is reported in the file `audita.txt` which resides under the *expno*.

Before any processing has been done, the processing status parameters of a dataset do not contain significant values. After the first processing command, they represent the current processing status of the data. Any further processing command will update the processing status parameters.

After processing, the relevant processing status parameters are usually set to the same values as the corresponding processing parameters. In other words, the command has done what you told it to do. There are, however, some exceptions:

- when a processing command was interrupted, the processing status parameters might not have been updated yet.
- some processing parameters are modified by the processing command, e.g. STSI is rounded to the next higher multiple of 16 by ***xfb***. The rounded value is stored as the processing status parameter.
- the values of some parameters are a result of processing. They cannot be set by the user (they do not appear as processing parameters) but they are stored as processing status parameters. Examples are NC_proc, S_DEV and TILT.

2.2 Parameter values

With respect to the type of values they take, parameters can be divided into three

groups:

- parameters taking integer values, e.g. SI, TDeff, ABSG, NSP
- parameters taking float or double values, e.g. LB, PHC0, ABSF1
- parameters using a predefined list of values, e.g. BC_mod, WDW, PSCAL

You can easily see to which group a parameter belongs from the parameter table opened with the command ***edp***. Note that the values of parameters which use a predefined list are actually stored as integers. The first value of the list is always stored as 0, the second value as 1 etc. Table 2.1 shows the values of the parameter PH_mod as an example:

Parameter value	Integer stored in the proc(s) file
no	0
pk	1
mc	2
ps	3

Table 2.1

2.3 Parameter files

XWIN-NMR parameters are stored in various files in the dataset directory tree.

In a 1D dataset:

<du>/data/<user>/nmr/<name>/<expno>/

`acqu` - acquisition parameters
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - processing parameters
`procs` - processing status parameters
`outd` - output device parameters
`meta` - plot parameters for ***plot*** and ***view***
`meta.ext` - extended plot parameters

In a 2D dataset:

<du>/data/<user>/nmr/<name>/<expno>/

`acqu` - F2 acquisition parameters
`acqu2` - F1 acquisition parameters
`acqus` - F2 acquisition status parameters
`acqu2s` - F1 acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - F2 processing parameters
`proc2` - F1 processing parameters
`procs` - F2 processing status parameters
`proc2s` - F1 processing status parameters
`outd` - output device parameters
`meta` - plot parameters for ***plot*** and ***view***
`meta.ext` - extended plot parameters

In a 3D dataset:

<du>/data/<user>/nmr/<name>/<expno>/

`acqu` - F3 acquisition parameters
`acqu2` - F2 acquisition parameters
`acqu3` - F1 acquisition parameters
`acqus` - F3 acquisition status parameters
`acqu2s` - F2 acquisition status parameters
`acqu3s` - F1 acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - F3 processing parameters
`proc2` - F2 processing parameters
`proc3` - F1 processing parameters
`procs` - F3 processing status parameters
`proc2s` - F2 processing status parameters
`proc3s` - F1 processing status parameters
`outd` - output device parameters

2.4 List of processing parameters

This paragraph contains a list of all processing parameters with a description of their function and the commands they are interpreted by. Please note that compos-

ite processing commands like ***efp*** (which combines ***em***, ***ft*** and ***pk***) are not mentioned here. Nevertheless, they interpret all parameters which are interpreted by the single commands they combine. Processing parameters can be set with ***edp*** or by typing their names in lowercase letters on the command line.

ABSF1 - low field limit of the region which is baseline corrected

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value (ppm) and must be greater than ABSF2
- interpreted by ***absf***, ***apkf***, ***abs1***, ***abs2***, ***abst****, ***absot****, ***zert****, ***tabs****
- The 1D commands ***abs*** and ***absd*** do not interpret ABSF1 because they work on the entire spectrum. The command ***apkf***, for automatic phase correction, uses ABSF1 as the left limit of the region on which it calculates the phase values.

ABSF2 - high field limit of the region which is baseline corrected

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value (ppm), must be smaller than ABSF1
- interpreted by ***absf***, ***apkf***, ***abs2***, ***abs1***, ***abst****, ***absot****, ***zert****, ***tabs****
- The 1D commands ***abs*** and ***absd*** do not interpret ABSF2 because they work on the entire spectrum. The command ***apkf***, for automatic phase correction, uses ABSF2 as the right limit of the region on which it calculates the phase values.

ABSG - degree of the polynomial which is subtracted in baseline correction

- used in 1D, 2D and 3D datasets in all dimensions
- takes an integer value between 0 and 5 (default is 5)
- interpreted by ***abs***, ***absd***, ***absf***, ***abs2***, ***abs1***, ***abst****, ***absot****, ***tabs****
- A polynomial of degree ABSG is calculated by the baseline correction commands and then subtracted from the spectrum.

ABSL - integral sensitivity factor with reference to the noise

- used in 1D datasets
- takes a float value between 0 and 100 (default is 3)
- interpreted by ***abs***, ***absd***, ***absf***

- Data points greater than ABSL*(standard deviation) are considered spectral information, all other points are considered noise.

ALPHA - correction factor

- used in 2D datasets in F2 and F1
- takes a float value
- interpreted by ***ptilt***, ***ptilt1*** and ***add2d***
- For ***ptilt***, F2 ALPHA is the tilt factor. For ***ptilt1***, F1 ALPHA is the tilt factor. They must have a value between -2.0 and 2.0. For ***add2d***, F2 ALPHA is the multiplication factor for the current dataset (see also parameter GAMMA).

ASSFAC - assign the highest or second highest peak as reference for scaling

- used in 1D datasets
- takes a float value (default is 0.0)
- interpreted by ***plot****, ***view****, ***pp****, ***lipp****
- This parameter is interpreted as follows:

  If ASSFAC > 1, the second highest peak is used as reference for scaling, if the following is true: *h2 < hmax/ASSFAC*, where *h2* is the intensity of the second highest peak and *hmax* the intensity of the highest peak. If this condition is false, the highest peak is used as reference.

  If ASSFAC < -1, two plots are made on two sheets of paper if the following condition is true: *h2 < hmax = abs(ASSFAC)*. For the first plot, the second highest peak is used as reference, for the second plot the highest peak. However, the second plot is omitted, if the plot was created by the command sequence ***plots*** - ***flplot***.

  Other values of ASSFAC have no effect on the plot scaling.

ASSFACI - assigns the highest or second highest integral trail for scaling

- used in 1D datasets
- takes a positive float value
- interpreted by ***plot****, ***view****
- ASSFACI is interpreted as follows:

  If *h2 < hmax/ASSFACI*, the second largest integral trail is used for scaling,

where *h2* is the value of the second largest and *hmax* that of the largest integral. Otherwise, the largest integral is taken for scaling. The integral values are not affected by ASSFACI. Note that ASSFACI values greater than 1 have an effect.

ASSFACX - as ASSFAC but for automatic expansion of plots

- used in 1D datasets
- takes a float value
- interpreted by ***plotx***, ***viewx***

  If ASSFACX > 1, the second highest peak of the expanded regions is used as reference for scaling, if the following condition is true: *h2* < *hmax/ASSFACX*, where *h2* is the intensity of the second highest peak and *hmax* that of the highest peak. If this condition is false, the highest peak of the expanded regions is used as reference.

  If ASSFACX < -1 and *h2* < *-hmax/ASSFACX*, every expanded region is plotted twice, once with the largest and once with the second largest signal as a reference.

ASSWID - region excluded from second highest peak search

- used in 1D datasets
- takes a float value (Hz, default is 0)
- interpreted by ***plot****, ***view****, ***pp****, ***lipp****
- ASSWID is interpreted as follows:

  If abs(ASSFAC) > 1, a region of width ASSWID around the highest peak is excluded from the search for the second highest peak

AUNMP - processing AU program name

- used in 1D, 2D and 3D datasets in the first dimension
- takes a character string value
- interpreted by ***xaup***
- In all Bruker standard parameter sets, the parameter AUNMP is set to a suitable processing AU program.

AZFE - integral extension factor

- used in 1D datasets

- takes a float value (ppm, default 0.1)
- interpreted by ***abs***
- Integral regions are extended at both sides by AZFE ppm. If this extension causes adjacent regions to overlap, the centre of the overlap is used as the limit of the two regions.

AZFW - minimum distance between peaks for independent integration

- used in 1D datasets
- takes a float value (ppm)
- interpreted by ***abs***, ***ldcon***, ***gdcon***, ***mdcon***
- If peaks are more than AZFW apart, they are treated independently. If peaks are less than AZFW ppm apart, they are considered to be overlapping.

BCFW - filter width for FID baseline correction.

- used in 1D datasets
- takes a float value (ppm)
- interpreted by ***bc*** when BC_mod = sfil or qfil
- sfil/qfil is used to suppress signals in the center of the spectrum. BCFW determines the width of the region, around the center of the spectrum, which is affected by ***bc***.

BC_mod - FID baseline correction mode

- used for 1D, 2D, and 3D dataset in all dimensions
  (only useful in the acquisition dimension)
- takes one of the values *no*, *single*, *quad*, *spol*, *qpol*, *sfil*, *qfil*
- interpreted by ***bc***, ***em***, ***gm***, ***ft***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf****
- The values of BC_mod and the corresponding functions are shown in table 2.2. Most commands evaluate BC_mod for the function to be subtracted but not for the detection mode. The latter is then evaluated from the acquisition status parameter AQ_mod. This means, for example, it does not matter if you set BC_mod to *single* or *quad*. Only ***trf*** and ***xtrf**** evaluate the detection mode from BC_mod and distinguish between BC_mod = single and BC_mod = quad. The same counts for the values *spol/qpol* and *sfile/qfile*.

COROFFS - correction offset for FID baseline correction

BC_mod	Function subtracted from the FID	Detection mode
no	no function	
single	average intensity of the last quarter of the FID	single channel
quad	average intensity of the last quarter of the FID	quadrature
spol	polynomial of degree 5 (least square fit)	single channel
qpol	polynomial of degree 5 (least square fit)	quadrature
sfil	Gaussian function of width BCFW [a]	single channel
qfil	Gaussian function of width BCFW [a]	quadrature

Table 2.2

a. Marion, Ikura, Bax, J. Magn. Res. 84, 425-420 (1989)

- used in 1D, 2D and 3D datasets in all dimensions
- takes a double value (Hz, default is 0.0)
- interpreted by ***bc***, ***em***, ***gm***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf3***, ***tf2***, ***tf1***
- COROFFS is only interpreted for BC_mod = qpol or qfil. The center of the
- baseline correction is shifted by COROFFS Hz.

DATMOD - data mode: work on ’raw’ or ’proc’essed data

- used in 1D datasets
- takes the value *raw* or *proc*
- interpreted by ***add***, ***addc***, ***and***, ***div***, ***filt***, ***mul***, ***mulc***, ***ls***, ***or***, ***rs***, ***rv***, ***xor***, ***zf***, ***zp***

DC - multiplication factor or addition constant

- used in 1D datasets
- takes a float value
- interpreted by ***add***, ***addc***, ***addfid*** and ***mulc***
- For ***addc***, DC is an addition constant. For ***add***, ***addfid*** and ***mulc***, DC is a multiplication factor.

DFILT - Digital filter filename

- used in 1D datasets

- takes a character string value
- interpreted by ***filt***
- The file specified by DFILT must reside in the directory:
  `<xwhome>/exp/stan/nmr/filt/1d`
  and must be set up from a command shell. One standard file called `three-point` is delivered with XWIN-NMR.

FCOR - first (FID) data point multiplication factor

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value between 0.0 and 2.0
- interpreted by ***ft***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf***, ***xtrfp***, ***tf3***, ***tf2***, ***tf1***

For 1D digitally filtered Avance data (DIGMOD = digital), FCOR does not play a role because the first raw data point is always zero. FCOR, however, allows you to control the DC offset of the spectrum in the following cases:

- on A*X data
- on Avance data measured in analog mode (DIGMOD = analog)
- on 2D/3D Avance data in the second/second+third dimension

FT_mod - Fourier transform mode

- used in 1D, 2D and 3D in all dimensions
- takes one of the values *no*, *fsr*, *fqr*, *fsc*, *fqc*, *isr*, *iqr*, iqc, *isc*
- interpreted by ***trf***, ***xtrf****, ***xtrfp****
- the Fourier transform commands ***ft*** (1D), ***xfb***, ***xf2***, ***xf1*** (2D) and ***tf**** (3D) do not interpret FT_mod because they evaluate the Fourier transform mode from the acquisition status parameter AQ_mod. They do, however, set the processing status parameter FT_mod.
- The values of FT_mod have the following meaning:

GAMMA - multiplication factor

- used in 2D datasets in F2
- takes a float value
- interpreted by ***add2d***
- GAMMA is the multiplication factor for the second dataset (see also parameter ALPHA).

FT_mod	Fourier transform mode
no	no Fourier transform
fsr	forward, single channel, real
fqr	forward, quadrature, real
fsc	forward, single channel, complex
fqc	forward, quadrature, complex
isr	inverse, single channel, real
iqr	inverse, quadrature, real
isc	inverse, single channel, complex
iqc	inverse, quadrature, complex

Table 2.3

GB - Gaussian broadening factor for Gaussian window multiplication

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value between 0.0 and 1.0
- interpreted by ***gm***
- interpreted by ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf**** if WDW = EM or GM

INTBC - automatic baseline correction of integrals created by ***abs***

- used in 1D datasets
- takes the value *yes* or *no*
- interpreted by ***plot***, ***view***, ***li***, ***lipp***, ***lippf***
- INTBC has no effect on integrals which were created interactively from the integration menu.

INTSCL - scale 1D integrals relative to a reference dataset

- used in 1D datasets
- takes an integer value
- interpreted by ***plot***, ***view***, ***li***, ***lipp***, ***lippf***
- INTSCL is used as follows:

  For INTSCL > 0, the integral values are scaled individually for each spectrum.

For INTSCL = 0, the integrals on the plot will obtain the same numeric values as defined interactively from the integrate menu.

For INTSCL = -1, scaling is performed relatively to the last spectrum plotted.

ISEN - integral sensitivity factor with reference to the largest integral

- used in 1D datasets
- takes a positive float value (default 128)
- interpreted by ***abs***, ***absd***, ***absf***
- Only the regions of integrals which are larger (area) than the largest integral divided by ISEN are stored.

LB - Lorentzian broadening factor for exponential window multiplication

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value
- interpreted by ***em***, ***gm***
- interpreted by ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf**** if WDW = EM or GM
- LB must be positive for an exponential and negative for Gaussian window multiplication.

LEV0 - lowest 2D contour level multiplication factor

- used in 2D datasets in F2
- takes a positive float value (default is 35)
- interpreted by ***levcalc***
- ***levcalc*** sets the lowest contour level to LEV0*S_DEV, where S_DEV (standard deviation) is a processing status parameter.

LPBIN - number of points for linear prediction

- used in 1D, 2D and 3D datasets in all dimensions
- takes a positive integer value
- interpreted by ***ft***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf****

For backward prediction, LPBIN represents the number of input points with a maximum of TD - abs(TDoff). The default value of LPBIN is zero, which means all data points are used as input. The status parameter LPBIN (***dpp***) shows how many input points were actually used. For forward prediction, LPBIN can be used to reduce the number of prediction output points as specified in table 2.4.

Note LPBIN only has an effect in the last two cases. If LPBIN is smaller than TD or greater than 2*SI this has the same effect as LPBIN = 0.

parameter values	normal points	predicted points	zeroes
LPBIN = 0, 2*SI < TD	2*SI	-	-
LPBIN = 0, TD < 2*SI < 2*TD	TD	2*SI - TD	-
LPBIN = 0, 2*TD < 2*SI	TD	TD	2*SI - 2*TD
TD < LPBIN < 2*SI< 2*TD	TD	LPBIN - TD	2*SI - LPBIN
TD < LPBIN < 2*TD < 2*SI	TD	LPBIN - TD	2*SI - LPBIN

Table 2.4 Linear forward prediction

MAXI - maximum relative intensity for peak picking

- used in 1D datasets
- takes a float value (cm)
- interpreted by ***pp****, ***plot****, ***view****, ***li***, ***lipp****
- only peaks with an intensity smaller than MAXI will appear in the peak list. MAXI can also be set interactively from the ***Utilities*** menu.

MC2 - Fourier transform mode of the second (and third) dimension

the processing parameter MC2 is only interpreted if the acquisition status parameter FnMODE (***dpa***) does not exist or has the value *undefined*. FnMODE has been introduced with XWIN-NMR 3.0 and must be set (with ***eda***) according to the experiment type before the acquisition is started. As MC2, FnMODE only exists in the second (and third dimension). On datasets acquired with XWIN-NMR 2.6 or earlier, MC2 is interpreted and must be set before the data are processed. The parameter MC2:

- is used in 2D datasets in the second dimension (F1)
- is used in 3D datasets in the second and third dimension (F2 and F1)
- takes one of the values *QF*, *QSEQ*, *TPPI*, *States*, *States-TPPI*, *echo-antiecho*
- is interpreted by ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf****

ME_mod - FID linear prediction mode

- used in 1D, 2D and 3D datasets in all dimensions

- takes one of the values *no*, *LPfr*, *LPfc*, *LPbr*, *LPbc*
- interpreted by ***ft***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf****
- The values of ME_mod have the following meaning:

LPfr	forward LP on real data
LPfc	forward LP on complex data
LPbr	backward LP on real data
LPbc	backward LP on complex data

Table 2.5

Linear prediction is only performed for NCOEF > 0. Furthermore, LPBIN and, for backward prediction, TDoff play a role. The commands ***ft***, ***xfb***, ***xf2*** and ***xf1*** evaluate ME_mod but do not distinguish between LPfr and LPfc nor do they distinguish between LPbr and LPbc. The reason is that the detection mode (real or complex) is evaluated from the acquisition status parameter AQ_mod. However, ***trf***, ***xtrf*** and ***xtrf2*** evaluate the detection mode from ME_mod. in 1D, a combination of forward and backward prediction can be done by running ***trf*** with ME_mod = LPfc and ***trfp*** (or ***ft***) with ME_mod = LPbc. In 2D, this would be the sequence ***xtrf*** - ***xtrfp*** (or ***xfb***)

MI - minimum relative intensity for peak picking

- used in 1D datasets
- takes a float value (cm)
- interpreted by ***pp****, ***plot****, ***view****, ***li***, ***lipp****
- only peaks with an intensity greater than MI will appear in the peak list. MI can also be set interactively from the ***Utilities*** menu.

NCOEF - number of linear prediction coefficients

- used in on 1D, 2D and 3D datasets in all dimensions
- takes a positive integer value (default is 0)
- interpreted by ***ft***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf****
- NCOEF is typically set to 2-3 times the number of expected peaks. For NCOEF = 0, no prediction is done. Linear prediction also depends on the parameters ME_mod, LPBIN and TDoff.

NLEV - number of positive contour levels in a 2D spectrum

- used in 2D datasets in the F2 dimension
- takes positive integer value (default 6)
- interpreted by ***levcalc***
- The total number of levels (positive and negative) calculated by ***levcalc*** is 2*NLEV

NOISF1 - low field (left) limit of the noise region

- used in 1D datasets
- takes a float value (ppm)
- interpreted by ***sino***
- The noise in the region between NOISF1 and NOISF2 is calculated according to the algorithm described for the command ***sino***.

NOISF2 - high field (right) limit of the noise region

- used in 1D datasets
- takes a float value (ppm)
- interpreted by ***sino***
- The noise in the region between NOISF1 and NOISF2 is calculated according to the algorithm described for the command ***sino***.

NSP - number of data points shifted during right shift or left shift

- used in 1D datasets
- takes a positive integer value (default is 1)
- interpreted by ***ls*** and ***rs***
- NSP points are discarded from one end and NSP zeroes are added to the other end of the spectrum.

NZP - number of data points set to zero intensity

- used in 1D datasets
- takes a positive integer value (default is 0)
- interpreted by ***zp***
- ***zp*** sets the intensity of the first NZP points of the dataset to zero.

OFFSET - the ppm value of the first data point of the spectrum

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value (ppm)
- set by ***sref*** or interactive calibration
- The value is calculated according to the relation:

  OFFSET = (SFO1/SF-1) * 1.0e6 + 0.5 * SW * SFO1/SF

  where SW and SFO1 are acquisition status parameters. In fact, the relation for OFFSET depends on the acquisition mode. When the acquisition status parameter AQ_mod is *qsim*, *qseq* or *DQD*, which is usually the case, the above relation count. When AQ_mod is *qf*, the equation:

  OFFSET = (SFO1/SF-1) * 1.0e6

  is used.

PC - peak picking sensitivity

- used in 1D datasets
- takes a float value
- interpreted by ***pp****, ***plot****, ***view****, ***li***, ***lipp****
- a spectral point is only a considered peak if it is a maximum which is greater than the previous minimum plus 4*PC*noise. In addition to MI, PC provides an extra way of controlling the peak picking sensitivity. It allows you, for instance, to detect a shoulder on a large peak.

PHC0 - zero order phase correction value (frequency independent)

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value (degrees)
- set by ***apk***, ***apks***, ***apkf***, ***apk0*** on 1D datasets
- set interactively from the phase menu on 1D and 2D datasets
- interpreted by ***pk***, ***xfbp***, ***xf2p***, ***xf1p***, ***tf*p***
- interpreted by ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf3***, ***tf2***, ***tf1*** when PH_mod = pk
- PHC0 is one of the few examples where a processing parameter is set by a processing command. For example, ***apk*** sets both the processing and processing status parameter PHC0. ***pk*** reads the processing parameter and updates the processing status parameter. For multiple phase corrections, the

total zero order phase value is stored as the processing status parameter PHC0.

PHC1 - first order phase correction value (frequency dependent)

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value (degrees)
- set by ***apk***, ***apks***, ***apkf***, ***apk1*** on 1D datasets
- set interactively from the phase menu on 1D and 2D datasets
- interpreted by ***pk***, ***xfbp***, ***xf2p***, ***xf1p***, ***tf*p***
- interpreted by ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf3***, ***tf2***, ***tf1*** when PH_mod = pk
- PHC1 is one of the few examples where a processing parameter is set by a processing command. For example, ***apk*** sets both the processing and processing status parameter PHC1. ***pk*** reads the processing parameter and updates the processing status parameter. For multiple phase corrections the total first order phase value is stored as the processing status parameter PHC1.

PH_mod - phase correction mode

- used in 1D, 2D and 3D datasets in all dimensions
- takes one of the value *no*, *pk*, *mc*, *ps*
- interpreted by ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf****
- The values of PH_mod are described in table 2.6.

PH_mod	mode
no	no phase correction
pk	phase correction according to PHC0 and PHC1
mc	magnitude calculation
ps	power spectrum

Table 2.6

- The value PH_mod = pk is only useful if the phase values are known and the parameters PHC0 and PHC1 have been set accordingly. In 1D, they can be

determined with ***apk*** or ***apks***, or, interactively, from the ***phase*** menu. In 2D and 3D, they can only be determined interactively.

PKNL - group delay handling (Avance) or filter correction (A*X)

- used in 1D, 2D and 3D datasets in the first dimension
- takes the value *true* or *false*
- interpreted by ***ft***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf****
- On A*X spectrometers, PKNL = true causes a non linear 5th order phase correction of the raw data. This corrects possible errors caused by non linear behaviour of the analog filters. On Avance spectrometers, PKNL must always be set to TRUE. For digitally filtered data, it causes ***ft*** to handle the group delay of the FID. For analog data it has no effect.

PSCAL - determines the region with reference peak for vertical scaling

- used in 1D datasets
- takes one of the values *global*, *preg*, *ireg*, *pireg*, *sreg*, *psreg*, *noise*
- interpreted by ***pp****, ***plot****, ***view****, ***li***, ***lipp****
- the values of PSCAL have the following meaning:.

PSCAL	Peak used as reference for vertical scaling
global	The highest peak of the entire spectrum.
preg	The highest peak within the plot region.
ireg	The highest peak within the regions specified in the `reg` file. If the `reg` file does not exist, *global* is used.
pireg	as *ireg*, but the peak must also lie within the plot region.
sreg	The highest peak in the regions specified in scaling region file. This file is specified by the parameter SREGLST. If SREGLST is not set or specifies a file which does not exist, *global* is used.
psreg	as *sreg* but the peak must also lie within the plot region.
noise	The intensity of the noise.

Table 2.7

- For PSCAL = ireg or pireg, the `reg` file is interpreted. The `reg` file can be created from the ***integrate*** menu and can be viewed or edited with the command ***edmisc reg***.
- For PSCAL = sreg or psreg, the scaling region file is interpreted. This feature is used to exclude the region in which the solvent peak is expected. The name of a scaling region file is typically of the form NUCLEUS.SOLVENT, e.g. 1H.CDCl3. For all common nucleus/solvent combinations, a scaling region file is delivered with XWIN-NMR. These can be viewed or edited with the command ***edlist scl***. In several 1D standard parameter sets which are used during automation, PSCAL is set to *sreg* and SREGLIST to NUCLEUS.SOLVENT as defined by the parameters NUCLEUS and SOLVENT.

PSIGN - peak sign for peak picking

- used in 1D datasets
- takes the value *pos*, *neg* or *both* (default is *pos*)
- interpreted by ***pp****, ***plot****, ***view****, ***lipp****
- in most 1D standard parameter sets PSIGN is set to *pos* which means only positive peaks are picked

REVERSE - flag indicating to reverse the spectrum during Fourier transform

- used in 1D, 2D and 3D datasets in all dimensions
- takes the value *true* or *false* (default is *false*)
- interpreted by ***ft***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf****
- Reversing the spectrum can also be done after Fourier transform with the commands ***rv*** (1D) or ***rev2***, ***rev1*** (2D).

SF - spectral reference frequency

- used in 1D, 2D and 3D datasets in the first dimension
- takes a positive float value
- set by ***sref*** or interactive calibration
- ***sref*** calculates SF according to the relation:

  SF=BF1/(1.0+RShift * 1e-6)

  where *RShift* is taken from the ***edlock*** table and BF1 is an acquisition status parameter. SF is interpreted by display and plot routines for generating the

axis (scale) calibration.

SI - size of the processed data

- used in 1D, 2D and 3D datasets in all dimensions
- takes an integer value
- interpreted by processing commands which work on the raw data (commands working on processed interpret the processing status parameter SI)
- The total size of the processed data (real+imaginary) is 2*SI. In Bruker standard parameter sets (see ***rpar***), SI is set to TD/2, where TD is an acquisition status parameter specifying the number of raw data points.

SIGF1 - low field (left) limit of the signal region

- used in 1D and 2D datasets
- takes a float value (ppm), must be greater than SIGF2
- interpreted by ***sino***
- If SIGF1 = SIGF2, the signal region is defined by the entire spectrum minus the first 16th part or, if the scaling region file exists, by the regions in this file. The name of the scaling region file is NUC1.SOLVENT where NUC1 and SOLVENT are acquisition status parameters.
- SIGF1 is also used in 2D datasets as the low field limit for 2D baseline correction by ***abst2***, ***abst1***, ***absot2***, ***absot1***, ***zert1***, and ***zert2***.

SIGF2 - high field (right) limit of the signal region

- used in 1D and 2D datasets
- takes a float value (ppm), must be smaller than SIGF1
- interpreted by ***sino***
- If SIGF1 = SIGF2, the signal region is defined by the entire spectrum minus the first 16th part or, if the scaling region file exists, by the regions in this file. The scaling region file is defined as NUC1.SOLVENT where NUC1 and SOLVENT are acquisition status parameters.
- SIGF2 is also used in 2D datasets as the high field limit for 2D baseline correction by ***abst2***, ***abst1***, ***absot2***, ***absot1***, ***zert1***, and ***zert2***.

SINO - signal to noise ratio

- used in 1D datasets

- takes a float value
- used in AU as an acquisition criterion (not used by processing commands)
- the processing parameter SINO (set with ***edp***) can be used in an AU program to specify a signal/noise ratio which must be reached in an acquisition. The acquisition runs until the value of SINO is reached and then it stops. An example of such an AU program is ***au_zgsino***. SINO can be set with ***edp*** but not from the command line. The reason is that entering ***sino*** on the command line would execute the command ***sino***. Note that the processing parameter SINO (***edp***) has a different purpose than the processing status parameter SINO (***dpp***). The latter represents the signal to noise ratio calculated by the processing command ***sino***.

SREGLST - name of the scaling region file

- used in 1D datasets
- takes a character string value
- interpreted by ***pp****, ***plot****, ***view****, ***li***, ***lipp**** if PSCAL = sreg or psreg
- interpreted by ***sino***
- scaling region files contain the regions in which the reference peak is searched. They are used to exclude the region in which the solvent peak is expected. Because this region is nucleus and solvent specific the name of a scaling region file is of the form NUCLEUS.SOLVENT, e.g. 1H.CDCl3. For all common nucleus/solvent combinations, a scaling region file is delivered with XWIN-NMR. They can be viewed or edited with ***edlist scl***.

SSB - sine bell shift

- used in 1D, 2D and 3D datasets in all dimensions
- takes a positive float value
- interpreted by ***sinm***, ***qsin***, ***sinc***, ***qsinc***
- interpreted by ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf**** if WDW = sine, qsine, sinc or qsinc

SR - spectral reference

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value (Hz)
- set by ***sref*** or interactive calibration

- The spectral reference is calculated according to the relation:

  SR = SF - BF1

STSI - strip size: number of output points of strip transform

- used in 1D, 2D and 3D datasets in all dimensions
- takes an integer value between 0 and SI (default 0)
- interpreted ***ft***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf***, ***xtrf2***, ***tf3***, ***tf2***, ***tf1***
- During strip transform, only the region determined by STSI and STSR is stored. For STSI = 0, a normal (full) transform is done. STSI is always rounded; in 1D to the next lower multiple of 4, in 2D and 3D to the next higher multiple of 16. Furthermore, when the 2D (3D) data are stored in submatrix (subcube) format, STSI is rounded to the next multiple of the submatrix (subcube) size.

STSR - strip start: first output point of a strip transform

- used in 1D, 2D and 3D datasets in all dimensions
- takes an integer value between 0 and SI (default 0)
- interpreted ***ft***, ***xfb***, ***xf2***, ***xf1***, ***xtrf***, ***xtrf2***, ***tf3***, ***tf2***, ***tf1***
- During strip transform, only the region determined by STSI and STSR is stored.

TDeff - number of raw data points to be used for processing

- used in 1D, 2D and 3D datasets in all dimensions
- takes an integer value between 0 and TD (default is 0 which means all)
- interpreted by processing commands which work on the raw data
- The first TDeff raw data points are used for processing. For TDeff = 0, all points are used, with a maximum of 2*SI.

TDoff - number of raw data points ignored or predicted

- used in 1D, 2D and 3D datasets in all dimensions
- integer value between 0 and TD (default is 0)
- interpreted by 2D and 3D processing commands which work on raw data The first raw data point that contributes to processing is shifted by TDoff points. For 0 < TDoff < TD the first TDoff raw data points are cut off at the

beginning and TDoff zeroes are appended at the end (corresponds to left shift). For TDoff < 0 -TDoff zeroes are prepended at the beginning and:

- for SI < (TD-TDoff)/2 raw data are cut off at the end
- for DIGMOD=digital, the zeroes would be prepended to the group delay which does not make sense. You can avoid that by converting the raw data with ***convdta*** before you process them.

- also interpreted by 1D, 2D and 3D processing commands which do linear backward prediction, i.e. ***ft***, ***xfb*** of ***tf3*** when ME_mod is *lpbr* or *lpbc*. For TDoff > 0, the first TDoff points are replaced by predicted points. For TDoff < 0, abs(TDoff) predicted points are added to the beginning and cut off at the end of the raw data. If zero filling occurs (2*SI > TD), then only zeroes are cut off at the end as long as abs(TDoff) < 2*SI - TD. Note that digitally filtered Avance data start with a group delay. This means that a backward prediction does not make sense unless the data are first converted AMX format with ***convdta***.

TM1 - the end of the rising edge of a trapezoidal window

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value between 0.0 and 1.0
- interpreted by ***tm***
- TM1 represents a fraction of the acquisition time and must be smaller than TM2

TM2 - the start of the falling edge of a trapezoidal window

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value between 0.0 and 1.0
- interpreted by ***tm***
- TM2 represents a fraction of the acquisition time and must be greater than TM1.

TOPLEV - highest 2D contour level

- used in 2D datasets in the F2 dimension
- takes a float value between 0 and 100 (default is 100%)
- interpreted by ***levcalc***

- TOPLEV is a percentage of the maximum intensity in the spectrum as expressed by the processing status parameter YMAX_p. For TOPLEV = 0, the highest level is set to 85% of the maximum intensity.

WDW - FID window multiplication mode

- used in 1D, 2D and 3D datasets in all dimensions
- takes one of the values *no*, *em*, *gm*, *sine*, *qsine*, *trap*, *user*, *sinc*, *qsinc*, *traf*, *trafs*
- interpreted by ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf****
- On 1D data, window multiplication is usually done with commands like ***em***, ***gm***, ***sinm*** etc. which do not interpret WDW. These command are already specific for one type of window multiplication. The values of WDW have the following meaning:

WDW value	Function	Dependent parameters	Specific 1D command
em	Exponential	LB	***em***
gm	Gaussian	GB, LB	***gm***
sine	Sine	SSB	***sinm***
qsine	Sine squared	SSB	***qsin***
trap	Trapezium	TM2, TM1	***tm***
user	User defined		***uwm***
sinc	Sine	SSB, GB	***sinc***
qsinc	Sine squared	SSB, GB	***qsinc***
traf	Traficante (JMR, **71**, 1987, 237)		***traf***
trafs	Traficante (JMR, **71**, 1987, 237)		***trafs***

Table 2.8

2.5 Processing status parameters

After processing, most processing status parameters have been set to the same value as the corresponding processing parameter. For some processing status parameters, however, this is different. The reason can be that:

- the corresponding processing parameter does not exist, e.g. NC_proc
- the corresponding processing parameter is not interpreted, e.g. FT_mod
- the value of the corresponding processing parameter is adjusted, e.g. STSI

These type of processing status parameters are listed below and described as output parameters for each processing command. They can be viewed with ***dpp*** (see also chapter 2.1).

BYTORDP - byte order of the processed data

- used in 1D, 2D and 3D datasets in the first dimension
- takes the value *little* or *big*
- set by the first processing command
- interpreted by various processing commands
- Big endian and little endian are terms that describe the order in which a sequence of bytes are stored in a 4-byte integer. Big endian means the most significant byte is stored first, i.e. at the lowest storage address. Little-endian means the least significant byte is stored first. XWIN-NMR runs on computers with different byte order, for example SGI workstations are big endian and Intel PC's are little endian. The byte order of the raw data is determined by the computer which controls the spectrometer and is stored in the acquisition status parameter BYTORDA (type ***1s bytorda***). This allows raw data to be processed on computers of the same or different storage types. The first processing command interprets BYTORDA, stores the processed data in the byte order of the computer on which it runs and sets the processing status parameter BYTORDP accordingly (type ***1s bytordp***). All further processing commands interpret this status parameter and store the data accordingly. As such, the byte order of the computer is handled automatically and is user transparent. 2D and 3D processing commands, however, allow you to store the processed data with a byte order different from the computer on which they run. For example, the command ***xfb little*** and ***tf3 little*** on an SGI stores the data in little endian although the computer is big endian. The processing status parameter BYTORDP is set accordingly.

FT_mod - Fourier transform mode

- used in 1D, 2D and 3D datasets in all dimensions
- takes one of the values *no*, *fsr*, *fqr*, *fsc*, *fqc*, *isr*, *iqr*, *iqc*, *isc*

- set by all Fourier transform commands, e.g. ***ft***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***trf****, ***xtrf****, ***tf3***, ***tf2***, ***tf1***
- interpreted by ***trf*** and ***xtrf****.
- also exists as processing (***edp***) parameter (interpreted by ***trf*** and ***xtrf****)
- The values of FT_mod are described in chapter 2.4.

MC2 - Fourier transform mode of the second (and third) dimension

- is used in 2D datasets in the second dimension (F1)
- is used in 3D datasets in the second and third dimension (F2 and F1)
- takes one of the values *QF*, *QSEQ*, *TPPI*, *States*, *States-TPPI*, *echo-antiecho*
- is set by ***xfb***, ***xf2***, ***xf1***, ***xtrf****, ***tf****
- is interpreted by ***xf1***, ***xtrf1***, ***tf2***, ***tf1***
- The processing status parameter MC2 is set according to the acquisition status parameter FnMODE. If, however, FnMODE = undefined, the processing status parameter MC2 is set according to the processing parameter MC2. Furthermore, status MC2 is interpreted during 2D processing in F1, on processed data, for example by ***xf1*** on data which have already been processed with ***xf2***.

NC_proc - intensity scaling factor

- used in 1D, 2D and 3D datasets in the first dimension
- takes an integer value
- set by all processing commands
- only exists as processing status parameter
- Processing in XWIN-NMR performs calculations in double precision floating point but stores the result in 32-bit integer values. During double to integer conversion, the data are scaled up or down such that the highest intensity of the spectrum lies between 2^{28} and 2^{29}. This means the 32 bit resolution is not entirely used. This allows for the highest intensity to be increased, for example during phase correction, without causing data overflow. NC_proc shows the amount of scaling that was done, for example:

  NC_proc = -3 : data were scaled up (multiplied by 2) three times
  NC_proc = 4 : the data were scaled down (divided by 2) four times

- Although NC_proc is normally calculated by processing commands, 2D processing also allows you to predefine the scaling factor with the argument ***nc_proc***, for example:

  xfb nc_proc 2

  scales down the data twice. However, you can only scale the data more down (or less up) than the command would have done without the argument ***nc_proc***. The latter is shown by the processing status parameter NC_proc (type ***dpp***). Smaller (more negative) values of ***nc_proc*** are ignored to avoid data overflow. The command:

  xfb nc_proc last

  takes the current value of the processing status parameter NC_proc (type ***dpp***) as input value.

PHC0 - zero order phase correction value (frequency independent)

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value (degrees)
- set by ***apk***, ***apks***, ***apkf***, ***apk0*** in 1D datasets
- set interactively from the phase menu in 1D and 2D datasets
- also exists as processing parameter (***edp***)
- PHC0 is one of the few examples where a processing parameter is set by a processing command. For example, ***apk*** sets both the processing and processing status parameter PHC0. ***pk*** reads the processing parameter and updates the processing status parameter. After multiple phase corrections, the processing status parameter PHC0 shows the total zero order phase correction.

PHC1 - first order phase correction value (frequency dependent)

- used in 1D, 2D and 3D datasets in all dimensions
- takes a float value (degrees)
- set by ***apk***, ***apks***, ***apkf***, ***apk1*** in 1D datasets
- set interactively from the phase menu in 1D and 2D datasets
- also exists as processing parameter (***edp***)
- PHC1 is one of the few examples where a processing parameter is set by a processing command. For example, ***apk*** sets both the processing and

processing status parameter PHC1. ***pk*** reads the processing parameter and updates the processing status parameter. For multiple phase corrections, the processing status parameter PHC1 shows the total first order phase correction.

S_DEV - standard deviation of the processed data

- used in 2D and 3D datasets in the first dimension
- takes a float value
- set by all processing commands, e.g. ***xfb***, ***xfbp***, ***abs2***, ***tf****, ***tabs****
- interpreted by ***levcalc***
- only exists as processing status parameter (***dpp***)

SINO - signal to noise ratio

- used in 1D datasets
- takes a float value
- set by ***sino***
- also exists as processing parameter
- The signal is determined in the region between SIGF2 and SIGF1. The noise is determined in the region between NOISF2 and NOISF1. Note that SINO also exists as a processing parameter (***edp***) which has a different purpose (see chapter 2.4)

SW_p - spectral width of the processed data

- used in 1D, 2D and 3D datasets in all dimensions
- takes a double value
- set by all processing commands
- only exists as processing status parameter
- Normally, SW_p will be the same as the acquisition status parameter SW. However, in case of stripped data, the processing spectral width differs from the acquired spectral width.

SYMM - 2D symmetrization type done

- used in 2D datasets in the F2 dimension
- takes the value *no*, *sym*, *syma* or *symj*
- set by ***sym***, ***syma*** and ***symj***

- only exists as processing status parameter (***dpp***)
- SYMM shows the (last) kind of symmetrization that was done.

STSI - strip size; the number of output points of a strip transform

- used in 1D, 2D and 3D datasets in all dimensions
- takes an integer value between 0 and SI (default 0)
- also exists as processing parameter (***edp***)
- rounded by ***ft***, ***trf***, ***xfb***, ***xf2***, ***xf1***, ***xtrf***, ***xtrf2***, ***tf3***, ***tf2***, ***tf1***
- During strip transform, only the region determined by STSI and STSR is stored. Processing commands round the value of the processing parameter STSI; in 1D to the next lower multiple of 4, in 2D and 3D to the next higher multiple of 16. Furthermore, when the 2D (3D) data are stored in submatrix (subcube) format, STSI is rounded to the next multiple of the submatrix (subcube) size. The rounded value is stored as the processing status parameter STSI. If no strip transform is done (STSI = 0), the status STSI is set to the value of SI.

TDeff - number of raw data points that were used for processing

- used in 1D, 2D and 3D datasets in all dimensions
- set by ***ft***, ***xfb***, ***xf2***, ***xf1***, ***trf****, ***xtrf****
- also exists as processing parameter (***edp***)
- Normally, all raw data points are used as input. However, the number of input points can be decreased with the processing parameter TDeff or increased by doing linear forward or backward prediction with TDoff < 0. The number of raw data points that were actually used is stored in the processing status parameter TDeff.

TILT - flag indicating whether a tilt command has been performed

- used in 2D datasets in the F2 dimension
- takes the value TRUE or FALSE
- set by ***ptilt***, ***ptilt1*** or ***tilt***
- only exists as processing status parameter (***dpp***)

XDIM - submatrix or subcube size

- used in 2D and 3D datasets in all dimensions

- takes an integer value
- set by ***xfb***, ***xf2***, ***xf1***, ***xtrf***, ***xtrf2***, ***tf3***
- also exists as processing parameter
- Although XDIM is normally calculated by processing commands, 2D and 3D processing also allow you to predefine the submatrix sizes. On a 2D dataset, the command:

  xfb xdim

  interprets the processing parameter XDIM in both F2 and F1. Note that the submatrix sizes cannot be set with ***edp*** but only with ***2 xdim*** and ***1 xdim***. On a 3D dataset, the command:

  tf3 c

  prompts you for the XDIM values in F3, F2 and F1. It does not interpret the processing parameter XDIM.

FTSIZE - Fourier transform size

- used in 1D, 2D and 3D datasets in all dimensions
- takes an integer value
- set by all processing command that perform Fourier transform
- Normally, the status parameter FSIZE has the same value as the status parameter SI. Only in case of strip transform (STSR > 0 and/or STSI > 0), they are different. FTSIZE then represents the size with which the raw data were Fourier transformed whereas SI represents the size with which the processed data are stored.

YMAX_p - maximum intensity of the processed data

- used in 1D, 2D and 3D datasets in the first dimension
- takes a float value
- set by all processing commands
- only exists as processing status parameter (***dpp***)

YMIN_p - minimum intensity of the processed data

- used in 1D, 2D and 3D datasets in the first dimension
- takes a float value
- set by all processing commands

- only exists as processing status parameter (***dpp***)

2.6 Relaxation parameters

Relaxation parameters can be set with the command ***edt1*** which can be entered from the Relaxation menu.

COMPNO - number of components contributing to the relaxation curve

- used in pseudo 2D relaxation datasets
- takes an integer value (default is 1)
- interpreted by ***simfit***
- Peak positions are determined on a row which is specified by the parameter START (usually the first row). These positions are then used by ***pd*** for each row of the 2D data. However, peak positions sometimes drifts in the course of the experiment, i.e. they might shift one or more points in successive rows. Therefore, ***pd*** searches for the maximum intensity at the predefined peak position plus or minus DRIFT. The command ***pd0*** ignores the value of DRIFT and takes the intensity exactly the predefined peak position. If there is no drift, ***pd*** and ***pd0*** have the same result.

CURSOR - the peak whose relaxation value is calculated

- used in pseudo 2D relaxation datasets
- takes an integer value (default is 1)
- interpreted by ***ct1***, ***ct2***, ***simfit***
- automatically set by ***nxtp***, ***pd***, ***dat1***, ***dat2***, ***simfit all***, ***simfit asc***
- CURSOR can be set with ***edt1*** to a specific peak. You can then fit the corresponding curve with ***ct1*** or ***simfit***. Normally, however, CURSOR is automatically set by one of the relaxation commands. In the following examples the number between brackets refers to the value of CURSOR after the specified command :

  pd (1) → ***ct1*** → ***nxtp*** (2) → ***ct1*** → ***nxtp*** (3) → ***ct1***

  simfit asc (1) → ***nxtp*** (2) → ***simfit*** → ***nxtp*** (3) → ***simfit***

  dat1 (last)

  simfit all (last)

where *last* refers to the last peak.

DRIFT - drift of the peak positions in the course of the experiment

- used in pseudo 2D relaxation datasets
- takes an integer value (must be 1 or greater, default is 5)
- interpreted by ***pd***
- Relaxation analysis is usually done with a series of relaxation curves, one for each peak in the spectrum. One curve shows the intensity distribution of one peak over a series of experiments, i.e. a series of rows in a pseudo 2D dataset. First the peak positions are determined on one row, for example with ***ppt1***. Then the command ***pd*** determines the intensity at these positions in each row. However, peak positions sometimes drifts in the course of the experiment, i.e. they can be slightly different in different rows. Therefore, ***pd*** searches for the maximum intensity in a range around a each peak position. This range is determined by the parameter DRIFT. The command ***pd0*** ignores the value of DRIFT and takes the intensity exactly the given peak position. If there is no drift, ***pd*** and ***pd0*** have the same result.

EDGUESS - table of initial values and step rates of the function variables

- used in pseudo 2D relaxation datasets
- interpreted by ***simfit***
- The EDGUESS table shows all variables of the function specified by FCT-TYPE. For each variable, the initial guess (G) and step rate (S) can be set for each component (C). Table 7.6 shows the EDGUESS table for an inversion recovery experiment, with 2 components.

GC1I0	0.5	SC1I0	0.05
GC1A	1.0	SC1A	0.1
GC1T1	2.0	SC1T1	0.2
GC2I0	0.5	SC2I0	0.05
GC2A	1.0	SC2A	0.1
GC2T1	2.0	SC2T1	0.2

Table 2.9

The initial guess for I[0] must be such that the total value of all components

does not exceed 1. If there is only one component, I[0] is usually set to 1. The step rate is usually set to about one tenth or the initial guess. If the step rate of a variable is set to zero, then this variable is not changed during the iterations. Note that the commands ***ct1***, ***ct2***, ***dat1*** or ***dat2*** do not use the EDGUESS table. They calculate the initial values and step rates of the T1/T2 function variables I[0], P and T1.

FCTTYPE - function type used for fitting the relaxation curve

- used in pseudo 2D relaxation datasets
- takes one of the values listed in table 2.10
- interpreted by ***simfit***
- Table 2.10 shows the experiment types which ***simfit*** can handle and the corresponding fit functions.

Exp. type	Comp	Fit function
uxnmrt1t2	1	I[t] = I[0]+P*exp(t/T1)
invrec	1 - 4	I[t] = I[0]*(1-2A*exp(-t/T1))
satrec	1 - 6	I[t] = I[0]*(1-exp(-t/T1))
cpt1rho	1 - 4	I[t] = I[0]/(1-TIS/T1rho)*(exp(-t/T1rho)-exp(t/TIS))
expdec	1 - 6	I[t] = I[0]*exp(-t/T)
gaussdec	1 - 6	I[t] = I[0]*exp(-SQR(t/T))
lorgauss	1 - 3	I[t] = IL*exp(-t/TL)+IG*exp(-SQR(t/TG))
linear	1 - 6	I[t] = A+B*t
varbigdel	1 - 6	I = I[0]*exp(-D*SQR(2*PI*gamma*G*LD)*(BD-LD/3)*1e4)
varlitdel	1 - 6	I = I[0]*exp(-D*SQR(2*PI*gamma*G*LD)*(BD-LD/3)*1e4)
vargrad	1 - 6	I = I[0]*exp(-D*SQR(2*PI*gamma*G*LD)*(BD-LD/3)*1e4)
raddamp	1 - 6	MZ[t]=A0+MZ[0]*tanh((t-T0)/TRD)

Table 2.10

- Note that ***ct1***, ***ct2***, ***dat1*** and ***dat2*** do not evaluate FCTTYPE because they can only handle T1/T2 experiments. They do, however, set FTCTYPE to the value *t1/t2*.

FITTYPE - relaxation fit type

- used in pseudo 2D relaxation datasets
- takes the value *area* or *intensity* (default is *intensity*)
- interpreted by ***pd***, ***pd0***, ***ct1***, ***dat1*** and ***simfit***
- Before you run ***pd***, both the integral ranges and peak positions should be determined (see ***rspc*** and ***ppt1***). ***pd*** then picks the points storing both their integrals and intensities but it only displays one curve; the one defined by FITTYP. ***ct1*** or ***simfit*** then calculate the relaxation value for one peak according to FITTYPE. You can change FITTYP and recalculate the relaxation value without running ***pd*** again. The same counts for the commands ***dat1*** and ***simfit all*** which fit all peaks.

INC - point (1D) or row (2D) increment

- used in 1D and pseudo 2D relaxation datasets
- takes an integer value (default is 1)
- interpreted by ***pft2*** (1D data)
- interpreted by ***pd*** and ***pd0*** (pseudo 2D data)
- Starting with START, every INC point (1D) or row (pseudo 2D) is used for relaxation analysis.

LISTINC - time increment between data points

- used in 1D and pseudo 2D relaxation datasets
- takes an float value (default is 0.001 sec)
- interpreted by ***pft2*** (1D data) when LISTTYPE = auto
- interpreted by ***pd*** and ***pd0*** (pseudo 2D data) when LISTTYPE = auto
- LISTINC represents the time increment between the data points. The time increment between points which are actually used for relaxation analysis is INC*LISTINC. Note that LISTINC does not determine which points are used for the calculation. This is determined by START, INC and NUMPNTS.

LISTTYP - type of lists with the experimental delays (tau values)

- used in 1D and pseudo 2D datasets
- takes the value *dw*, *auto*, *vdlist*, *vplist*, *vclist*, etc.
- interpreted by ***pft2*** (1D)
- interpreted by ***pd*** and ***pd0*** (pseudo 2D)

- On 1D data, LISTTYP is usually set to *dw*. In that case, the dwell time, as expressed by the acquisition status parameter DW is used as the time increment between the data points. If, however, LISTTYP = auto, the time of the first data point is determined by X_START and the time increment by LIST-INC. On pseudo 2D data, LISTTYP is usually set to *vdlist*. In that case, the command ***pd*** interprets the file `vdlist` in the acquisition data directory (under the expno). If this file does not exist, ***pd*** evaluates the acquisition status parameter VDLIST (which can be viewed ***dpa*** or ***2s vdlist***) and interprets the specified file. The same procedure counts when LISTTYPE is *vp*, *vc* etc.

NUMPNTS - number of data points used for relaxation analysis

- used in 1D and pseudo 2D relaxation datasets
- takes an integer value (default is TD)
- interpreted by ***pft2*** (1D)
- interpreted by ***pd*** and ***pd0*** (pseudo 2D)
- The default value of NUMPNTS is the number of available points, i.e. TD (1D) or F1 TD (pseudo 2D). TD is the acquisition status parameter which can be viewed with ***dpa*** or ***1s td***. Note that if you increase INC, you must reduce NUMPNTS such that INC*NUMPNTS does not exceed TD.

START - first point (1D) or row (2D) used for relaxation analysis

- used in 1D and pseudo 2D relaxation datasets
- takes an integer value (default is 1)
- interpreted by ***pft2*** (1D data)
- interpreted by ***pd*** and ***pd0*** (pseudo 2D data)
- Note that the default value (1) is not the first but the second point of a 1D dataset. It is, however, the first row of a pseudo 2D dataset. The nth point or row used is START + n*INC.

X_START - x-axis start; time of the first point used for relaxation analysis

- used in 1D and pseudo 2D relaxation datasets
- takes an float value (default is 0.0 sec)
- interpreted by ***pft2*** (1D data) when LISTTYPE = auto
- interpreted by ***pd*** and ***pd0*** (pseudo 2D data) when LISTTYPE = auto

- X_START represents the time (tau) of the first data point (START) used for relaxation analysis. Note that X_START does not determine which points are used for the calculation. This is determined by START, INC and NUMPNTS.

2.7 Output device parameters

CURPLOT - plotter or printer to be used for plotting

This plotter or printer will be used to plot the current dataset with any of the ***plot**** commands. The value of CURPLOT is overruled by a plotter specified in ***setres***.

PFORMAT - format file for the parameter object on the plot (default *normpl*)

used by all ***view**** and ***plot**** commands

DFORMAT - format file for parameter listing on the screen (default *normdp*)

used by ***dp***, ***dpa***, ***dpp***, ***dpg***, ***dpgx***, ***dpc*** and ***dpo***

LFORMAT - format file for parameter listing on the printer (default *normpl*)

used by ***lp***, ***lpa***, ***lpp***, ***lpg***, ***lpgx***, ***lpc*** and ***lpo***

CURPRIN - printer to be used for printing from XWIN-NMR

used by ***lp***, ***lpa***, ***lpp***, ***lpg***, ***lpgx***, ***lpc***, ***lpo*** and all print buttons

LAYOUT - XWIN-PLOT layout file

used by ***xwinplot*** and ***autoplot***

Chapter 3

1D Processing commands

This chapter describes all XWIN-NMR 1D processing commands. Several of them can also be used to process one row of 2D or 3D data. They store their output in processed data files and do not change the raw data.

For each command, the relevant input and output parameters are mentioned. Furthermore, the relevant input and output files and their location are mentioned. Although file handling is completely transparent, it is sometimes useful to know which files are involved and where they reside. For example, if you have permission problems or if you want to process or interpret your data with third party software.

abs, absf, absd

NAME

abs - automatic baseline correction
absf - automatic baseline correction of the plot region
absd - automatic baseline correction with a different algorithm

DESCRIPTION

The command ***abs*** performs an automatic baseline correction of the spectrum, by subtracting a polynomial. The degree of the polynomial is determined by the parameter ABSG which has a value between 0 and 5, with a default of 5. ***abs*** first determines which parts of the spectrum contain spectral information and stores the result in the file `intrng` (integral regions). The remaining part of the spectrum is considered baseline and used to fit the polynomial function.

abs interprets the parameter ABSG to determine to the degree of the polynomial to be subtracted.

abs interprets the parameters ABSL, AZFW, AZFE and ISEN to determine the integral regions. Data points greater than ABSL*(standard deviation) are considered spectral information, all other points are considered noise. If two peaks are more than AZFW apart, they are treated independently. If they are less than AZFW ppm apart, they are considered to be overlapping. Integral regions are extended at both sides by AZFE ppm. If this extension causes adjacent regions to overlap, the centre of the overlap is used as the limit of the two regions. Only regions whose integrals are larger (area) than the largest integral divided by ISEN are considered.

abs n does not store the integral ranges. It is, for example, used in the command sequence ***ef***, ***mc***, ***abs***, ***efp***, ***abs n*** to store the integral regions of both positive and negative peaks. The command ***abs*** only stores the regions of positive peaks.

absd works like ***abs***, except that it uses a different algorithm[1]. It is, for example, used when a small peak lies on the foot of a large peak. In that case, ***absd*** allows you to correct the baseline around the small peak which can then be integrated. Usually ***absd*** is followed by ***abs***.

absf works like ***abs*** except that it only corrects the spectral region which is

1. It uses the same algorithm as the command ***abs*** in DISNMR

determined by the parameters ABSF1 and ABSF2.

If automatic baseline correction does not give satisfactory results, you can apply an interactively determined polynomial, exponential, sine or spline baseline correction. See the commands ***bcm*** and ***sab*** for details.

The integral regions determined by ***abs*** can be viewed/edited with the command ***edmisc intrng***. You can view the integral regions and the corresponding integrals by entering the ***integrate*** menu and reading the `intrng` file from the ***File*** menu. The integral regions are also used by various commands which calculate spectral integrals like ***li***, ***lipp*** and ***plot***.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***absg***, ***absf1*** etc.:

ABSG - degree of the polynomial to be subtracted (0 to 5, default of 5)
ABSF1 - low field (left) limit of the region corrected by ***absf***
ABSF2 - high field (right) limit of the region corrected by ***absf***
ABSL - integral sensitivity factor with reference to the noise
AZFW - minimum distance between peaks for independent integration
AZFE - integral extension factor
ISEN - integral sensitivity factor with reference to the largest integral

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`procs` - processing status parameters
`intrng` - integral regions
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

ABS

ABSD

ABSF

SEE ALSO

bcm, sab

add, addfid

NAME

add - add two datasets multiplying one of them with DC
addfid - add two FIDs multiplying one of them with DC

DESCRIPTION

The command ***add*** adds two datasets multiplying one of them with a user defined constant. The input data are specified with the command ***edc2*** as the so-called second and third dataset. The third dataset is multiplied with the value of DC. The result is stored in the current dataset. ***add*** works on raw or on processed data, depending on the value of DATMOD. For DATMOD = raw, ***add*** adds the raw data of the second and third dataset but stores the result as processed data in the current dataset. As such, the raw data of the current dataset are not overwritten.

The command ***addfid*** adds two raw datasets multiplying one of them with the factor DC. The input data are specified with the command ***edc2*** as the so-called second and third dataset. The third dataset is multiplied with the value of DC. The result is stored in the current dataset. It works like ***add*** with DATMOD = raw, except that it overwrites the raw data.

For both ***add*** and ***addfid***, the second or third dataset can be the same as the current dataset. This is, for example, useful if you repeatedly want to add an spectrum (or FID) to the current spectrum (or FID).

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***dc***, ***datmod*** etc.:

DC - multiplication factor
DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - processing parameters

<du2>/data/<user2>/nmr/<name2>/<expno2>/

`fid` - ’second’ raw data (input of ***addfid*** or ***add*** if DATMOD = raw)

<du2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

`1r`, `1i` - 'second' processed data (input of ***add*** if DATMOD = proc)

<du3>/data/<user3>/nmr/<name3>/<expno3>/

`fid` - 'third' raw data (input of ***addfid*** or ***add*** if DATMOD = raw)

<du3>/data/<user3>/nmr/<name3>/<expno3>/pdata/<procno3>/

`1r`, `1i` - 'third' processed data (input of ***add*** if DATMOD = proc)

Note that *du*, *user* and *name* of the current, second and third dataset are often the same and only the *expno* and/or *procno* are different

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw 1D data of the current dataset (output of ***addfid***)
`audita.txt` - acquisition audit trail (output of ***addfid***)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data of the current dataset (output of ***add***)
`procs` - processing status parameters
`auditp.txt` - processing audit trail (output of ***add***)

USAGE IN AU PROGRAMS

ADD

ADDFID

SEE ALSO

addc, mul, mulc, div, and, or, xor, edc2

addc

NAME

addc - add a constant DC to a spectrum or FID

DESCRIPTION

The command ***addc*** adds the value of DC to the current dataset. It can work on raw or processed data, depending on the value of DATMOD. The result is stored as processed data in the current dataset.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***dc***, ***datmod*** etc.:

DC - addition constant
DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if DATMOD = raw)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`procs` - processing status parameters

USAGE IN AU PROGRAMS

ADDC

SEE ALSO

add, addfid, mul, mulc, div, and, or, xor

and

NAME

and - combine two datasets according to a logical ’and’

DESCRIPTION

The command ***and*** combines two datasets according to a logical ’and’ (boolean operation). The input data must be specified as the second and third dataset with the command ***edc2***. The result is stored in the current dataset.

Depending on the value of DATMOD, ***and*** works on raw or processed data. For DATMOD = raw, ***and*** combines the raw data of the second and third dataset but stores the result as processed data in the current dataset.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***datmod*** :

DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - processing parameters

<du2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

`1r`, `1i` - ’second’ processed data (input if DATMOD = proc)

<du3>/data/<user3>/nmr/<name3>/<expno3>/pdata/<procno3>/

`1r`, `1i` - ’third’ processed data (input if DATMOD = proc)

Note that *du*, *user* and *name* of the current, second and third dataset are often the same and only the *expno* and/or *procno* are different. For DATMOD = raw, the files `fid` under *expno2* and *expno3* are input.

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters

USAGE IN AU PROGRAMS

AND

SEE ALSO

or, xor, add, addc, addfid, mul, mulc, div, edc2

apk, apk0, apk1, apks, apkf

NAME

apk - automatic phase correction
apkf - automatic phase correction based on a certain spectral region
apks - automatic phase correction with a different algorithm
apk0 - zero order automatic phase correction
apk1 - first order automatic phase correction

DESCRIPTION

The command ***apk*** calculates the zero and first order phase values and then corrects the spectrum according to these values. The phase values are stored in the parameters PHC0 and PHC1, respectively. Note that ***apk*** stores the calculated phase values both as input (processing) parameters (***edp***) and as output (processing status) parameters (***dpp***).

apkf works like ***apk*** except that it uses only a certain region of the spectrum for the calculation of the phase values. This region is determined by the parameters ABSF1 and ABSF2. The calculated phase values are then applied to the entire spectrum. Note that the parameters ABSF1 and ABSF2 are also used by the command ***absf.***

apks works like ***apk*** except that it uses a different algorithm which gives better results on certain spectra.

apk0 works like ***apk*** except that it only performs the zero order phase correction.

apk1 works like ***apk*** except that it only performs the first order phase correction.

The command ***apk*** or ***apks*** give satisfactory results on most spectra but not on all. Sometimes interactive phase correction is required which can be done from the ***phase*** menu.

The command ***pk*** also performs a phase correction but it simply applies the current values of PHC0 and PHC1 (see ***pk***).

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***absf1***, ***absf2*** etc.:

ABSF1 - low field (left) limit of the region used by ***apkf***
ABSF2 - high field (right) limit of the region used by ***apkf***

OUTPUT PARAMETERS

can be viewed with ***edp***, ***dpp*** or by typing ***phc0***, ***1s phc0*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

Note that this is one of the rare cases where the output parameters of a command are stored as processing (***edp***) and as processing status parameters (***dpp***).

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`proc` - processing parameters
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

APK

APKF

APKS

APK0

APK1

SEE ALSO

pk, mc, ps, fp, efp, fmc

bc

NAME

bc - baseline correction of the FID

DESCRIPTION

The command ***bc*** performs a baseline correction of raw 1D data. The type of correction is determined by the processing parameter BC_mod as shown in table 3.1.

BC_mod	Function subtracted from the FID	Detection mode
no	no function	
single	average intensity of the last quarter of the FID	single channel
quad	average intensity of the last quarter of the FID	quadrature
spol	polynomial of degree 5 (least square fit)	single channel
qpol	polynomial of degree 5 (least square fit)	quadrature
sfil	Gaussian function of width BCFW [a]	single channel
qfil	Gaussian function of width BCFW	quadrature

Table 3.1

a. Marion, Ikura, Bax, J. Magn. Res. 84, 425-420 (1989)

spol/qpol and *sfil/qfil* are especially used to subtract strong signals, e.g. a water signal at the centre of the spectrum. Note that *sfile/qfile* perform a better reduction at the risk of losing valuable signal. For reducing off-centre signal, you can set the parameter COROFFS to the offset frequency.

In this table, *s(ingle)* stands for single detection mode and *q(uad)* for quadrature detection mode. ***bc*** evaluate BC_mod for the function to be subtracted but not for the detection mode. The latter is evaluated from the acquisition status parameter AQ_mod. This means, for example, it does not matter if you set BC_mod to *single* or *quad*. The same counts for the values *spol*/*qpol* and *sfile/qfile*. Note that the commands ***trf*** and ***xtrf**** do evaluate the detection mode from BC_mod.

The command ***bc*** is automatically executed as a part of the commands ***em***, ***gm***, ***ft***, or any of the composite Fourier transform commands.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***bc_mod***, ***bcfw*** etc.:

BC_mod - FID baseline correction mode
BCFW - filter width for BC_mod = sfil or qfil
COROFFS - correction offset in Hz, for BC_mod = spol or qpol and sfil/qfil

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (time domain)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (time domain)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

BC

SEE ALSO

em, gm, ft

bcm

NAME

bcm - user defined spectrum baseline correction

DESCRIPTION

The command ***bcm*** performs a spectrum baseline correction by subtracting a polynomial, sine or exponential function.

This involves the following steps:

1. Enter ***basl*** to change to the baseline menu.
2. Click ***polynom***, ***sine*** or ***expon*** to select the baseline correction function
3. Fit the baseline of the spectrum with the function you selected in step 2 (initially represented by a straight horizontal line). Start by pressing button ***A*** and moving the mouse to determine the zero order correction. Continue with the buttons ***B***, ***C*** for higher order corrections until the line matches the baseline of the spectrum.
4. Click ***return*** → ***Save & return*** to change the main menu. A ***bcm*** is automatically done.

The interactively determined baseline function can be stored and applied to similar spectra as follows:

1. ***wmisc base_info <name>***
2. Read a similar dataset
3. ***rmisc base_info <name>***
4. ***bcm***

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`proc` - processing parameters
`base_info` - baseline correction coefficients

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

BCM

SEE ALSO

sab, abs, rmisc, wmisc, edmisc

div

NAME

div - divide two datasets

DESCRIPTION

The command ***div*** divides two datasets. The input data must be specified as the second and third dataset with the command ***edc2***. The result is stored in the current dataset.

Depending on the value of DATMOD, ***div*** works on raw or processed data. For DATMOD = raw, ***div*** divides the raw data of the second and third dataset but stores the result as processed data in the current dataset.

With XWIN-NMR 4.0 and newer, ***div*** performs a complex division on complex spectra. This requires for both input datasets that:

- the status parameter FT_mod = fqc or fsc
- real (file `1r`) and imaginary (file `1i`) data exist

This is the case for most Avance data. In XWIN-NMR 3.1 or older, or if the above requirements are not fulfilled, real and imaginary data are divided pointwise. When complex division has been performed, this is reported in the audit trail output file.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***datmod*** :

DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - processing parameters

<du2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)

<du3>/data/<user3>/nmr/<name3>/<expno3>/pdata/<procno3>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)

Note that *du*, *user* and *name* of the current, second and third dataset are often the same and only the *expno* and/or *procno* are different. For DATMOD = raw, the files `fid` under *expno2* and *expno3* are input.

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

DIV

SEE ALSO

add, addc, addfid, mul, mulc, and, or, xor, edc2

dt

NAME

dt - calculate the first derivative

DESCRIPTION

The command ***dt*** calculates the first derivative of the current dataset. Depending on the value of DATMOD, ***dt*** works on the raw or on the processed data.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***datmod*** :

DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if DATMOD = raw)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

DT

duadd

NAME

duadd - add two datasets according to their chemical shift values

DESCRIPTION

The command ***duadd*** adds two datasets according to their chemical shift values. Each ppm value of one dataset is added to the same ppm value of a second dataset. Note the difference with the command ***add*** which performs a point to point addition and is independent of the spectrum calibration.

duadd is useful when the two input spectra are:

- of different size
- referenced differently
- acquired with different frequencies (i.e. on different spectrometers)

For data with equal size, reference and spectrometer frequency, ***add*** and ***duadd*** give the same result. Furthermore, ***duadd*** allows you to add data with a user defined offset.

The input data of ***duadd*** are the current dataset and the second dataset. The result is stored in the third dataset. The second and third dataset must be specified with the command ***edc2***.

The command ***duadd*** takes 4 arguments and can be used as follows:

duadd
add the second dataset to the current dataset

duadd <offset>
add the second dataset, shifted by *offset* ppm

duadd <offset> <fact>
add the second dataset, shifted by *offset* ppm and multiplied by *fact*

duadd ppm
add corresponding ppm values

duadd hz
add corresponding ***Hz*** values

duadd y
overwrite possibly existing output data

duadd <offset> <fact> ppm|hz y
combination of the above

The default values of *offset* and *fact* are 0.0 and 1.0, respectively. Note that the argument *ppm* or *hz* is only required if the input data were acquired with different basic frequencies, i.e. when they come from different spectrometers.

duadd only works on processed data, independent of the value of DATMOD.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - current processed data

<du2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

`1r`, `1i` - 'second' processed data

Note that *du*, *user*, *name* and *expno* of the current and second dataset are often the same and only the *procno* is different

OUTPUT FILES

<du3>/data/<user3>/nmr/<name3>/<expno3>/pdata/<procno3>/

`1r`, `1i` - 'third' processed data
`procs` - processing status parameters
`auditp.txt` - processing audit trail

Note that *du*, *user*, *name* and *expno* of the current and third dataset are often the same and only the *procno* is different

SEE ALSO

edc2, add, addfid, addc

ef, efp

NAME

ef - exponential window multiplication + Fourier transform
efp - exponential window multiplication + Fourier transform + phase correction

DESCRIPTION

The composite processing command ***ef*** is a combination of ***em*** and ***ft***, i.e. it performs an exponential window multiplication and a Fourier transform.

efp is a combination of ***em***, ***ft*** and ***pk***, i.e. it does the same as ***ef*** but, in addition, performs a phase correction.

ef and ***efp*** automatically perform an FID baseline correction according to BC_mod.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

EF

EFP

SEE ALSO

gf, gfp, fp, fmc, em, gm, ft, pk, mc

em

NAME

em - exponential window multiplication of the FID

DESCRIPTION

The command ***em*** performs an exponential window multiplication of the FID. It is the most used window function for NMR spectra. ***em*** multiplies each data point *i* with the factor:

$$\exp\left(-\frac{(i-1)\cdot LB\cdot\pi}{2\cdot SWH}\right)$$

where LB (the line broadening factor) is a processing parameter and SWH (the spectral width) an acquisition status parameter.

The value for LB can be set with ***edp*** or determined interactively from the window function menu (***winfunc***).

em automatically performs an FID baseline correction according to BC_mod.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***lb***, ***bc_mod*** etc.:

LB - Lorentzian broadening factor
BC_mod - FID baseline correction mode

set by the acquisition, can be viewed with ***dpa*** :

SWH - spectral width

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)

`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

EM

SEE ALSO

gm, sinm, qsin, sinc, qsinc, tm, traf, trafs, uwm, ef, efp, bc

fmc

NAME

fmc - Fourier transform + magnitude calculation

DESCRIPTION

The composite processing command ***fmc*** is a combination of ***ft*** and ***mc***, i.e. it performs a Fourier transform and a magnitude calculation.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

FMC

SEE ALSO

mc, ps, fp, ef, efp, gf, gfp

filt

NAME

filt - digital filtering

DESCRIPTION

The command ***filt*** smoothes the data by replacing each point with a weighted average of its surrounding points. By default, ***filt*** uses the weighting coefficients 1-2-1 which means that the intensity *p(i)* of data point *i* is replaced by:

$$1 \cdot p(i-1) + 2 \cdot p(i) + 1 \cdot p(i+1).$$

Different weighting algorithms can be set up by creating a new file in the directory:

```
<xwhome>/exp/stan/nmr/filt/1d
```

Just copy the default file `threepoint` to a different name and modify it with a text editor. The file must look like:

3,1,2,1

or

5,1,2,3,2,1

where the first number represents the number of points used for smoothing and must be odd. The other numbers are the weighting coefficients for the data points. The processing parameter DFILT determines which file is used by ***filt***.

This is one of the few cases where file handling cannot be done from XWIN-NMR and needs to done on Windows or UNIX level.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***dfilt***, ***datmod*** etc. :

DFILT - digital filter filename
DATMOD - data mode: work on ’raw’ or ’proc’essed data.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`proc` - processing parameters

<xwhome>/exp/stan/nmr/filt/1d/*

digital filtering file(s)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

FILT

fp

NAME

fp - Fourier transform +phase correction

DESCRIPTION

The composite processing command ***fp*** is a combination of ***ft*** and ***pk***, i.e. it performs a Fourier transform and a phase correction.

fp automatically performs an FID baseline correction according to BC_mod.

INPUT AND OUTPUT PARAMETERS

see ***ft*** and ***pk***

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

FP

SEE ALSO

ef, efp, gf, gfp, fmc

ft

NAME

ft - 1D Fourier transform

DESCRIPTION

The command ***ft*** Fourier transforms a 1D dataset. Fourier transform is the main step in processing NMR data. The time domain data (FID) which are created by acquisition are transformed into frequency domain data (spectrum) which can be interpreted. Usually, Fourier transform is preceded by other processing steps like FID baseline correction (***bc***) and window multiplication (***em***, ***gm***, etc.) and followed by steps like phase correction (***apk***) and spectrum baseline correction (***abs***).

The size of the resulting spectrum is determined by the parameter SI. An FID of TD time domain points is transformed to a spectrum of SI real and SI imaginary data points. A typical value for SI is TD/2. In that case, all points of the FID are used by the Fourier transform and no zero filling is done.

The size of the spectrum and the number of FID points which are used can be determined in the following ways:

- SI > TD/2: the FID is zero filled
- SI < TD/2: only the first 2*SI points of the FID are used
- 0 < TDeff < TD: only the first TDeff points of the FID are used

In the latter two cases, the spectrum will contain less information then the FID. Note that the parameter TDoff only plays a role for linear prediction and in 2D and 3D Fourier transform.

You can also perform a so-called strip transform which means that only a certain region of the spectrum is stored. This can be done by setting the parameters STSR and STSI which represent the strip start and strip size, respectively. They can take values between 0 and SI. The processing status parameters STSI and SI are both set to this value. You can check this with ***dpp***.

The Fourier transform mode depends on the acquisition mode; *single*, *sequential* or *simultaneous*. For this purpose, ***ft*** evaluates the acquisition status parameter AQ_mod as shown in table 3.2. ***ft*** does not evaluate the processing parameter

AQ_mod	FT_mod	Fourier transform mode
qf	fsr	forward, single channel, real
qsim	fqc	forward, quadrature, complex
qseq	fqr	forward, quadrature, real
DQD	fqc	forward, quadrature, complex

Table 3.2

FT_mod but it does store the Fourier transform mode, as evaluated from the acquisition mode, in the processing status parameter FT_mod. Note that, the command ***trf*** determines the Fourier transform mode from the processing parameter FT_mod and not from the acquisition mode (see ***trf***).

ft evaluates the parameter FCOR. The first point of the FID is multiplied with FCOR which is a value between 0.0 and 2.0. However, on Avance spectrometers, the FID of digitally filtered data starts with a group delay of which the first points are zero so that the value of FCOR is irrelevant. On A*X data, FCOR allows you to control the DC offset of the spectrum.

ft evaluates the parameter PKNL. On A*X spectrometers, PKNL = true causes a non linear 5th order phase correction of the raw data. This corrects possible errors caused by non linear behaviour of the analog filters. On Avance spectrometers, PKNL must always be set to TRUE. For digitally filtered data, it causes ***ft*** to handle the group delay of the FID. For analog data it has no effect.

ft evaluates the parameter REVERSE. If REVERSE = TRUE, the spectrum will be reversed, i.e. the first output data point become the last and the last point become the first. The same effect is attained by using the command ***rv*** after ***ft***.

ft automatically performs an FID baseline correction according to BC_mod.

ft also performs linear prediction according to ME_mod. This parameter can take the following values:

no : no linear prediction
LPfr : forward LP on real data
LPfc : forward LP on complex data
LPbr : backward LP on real data
LPbc : backward LP on complex data

Forward prediction can, for example, be used to extend truncated FIDs. Back-

ward prediction can be used to improve the initial data points of the FID. ***ft*** determines the detection mode (real or complex) from the acquisition status parameter AQ_mod, not from ME_mod. As such, ***ft*** does not distinguish between ME_mod = LPfr and ME_mod = LPfc. The same counts for backward prediction. ***trf***, however, does determine the detection mode from ME_mod. Linear prediction is only performed for NCOEF > 0. Furthermore, LPBIN and, for backward prediction, TDoff play a role (see these parameters in chapter 2.4). By default, ME_mod is set to *no* which means no linear prediction is done.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***si***, ***stsr*** etc.:

SI - size of the processed data
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
TDeff - number of raw data points to be used for processing
FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)
REVERSE - flag indicating to reverse the spectrum
PKNL - group delay handling (Avance) or filter correction (A*X)
ME_mod - FID linear prediction mode
NCOEF - number of linear prediction coefficients
LPBIN - number of points for linear prediction
TDoff - number of raw data points predicted for ME_mod = LPb*

set by the acquisition, can be viewed with ***dpa*** or by typing ***1s aq_mod*** etc.:

AQ_mod - acquisition mode (determines the Fourier transform mode)
TD - time domain; number of raw data points

OUTPUT PARAMETERS

can be viewed with ***dpp*** or by typing ***1s ft_mod***, ***1s tdeff*** etc.:

FT_mod - Fourier transform mode
TDeff - number of raw data points that were used for processing
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
NC_proc - intensity scaling factor
YMAX_p - maximum intensity of the processed data
YMIN_p - minimum intensity of the processed data

can only be viewed by typing ***1s bytordp***:

BYTORDP - data storage order

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

FT

SEE ALSO

ef, efp, gf, gfp, fp, fmc, trf, trfp, ift, ht, bc, em, gm, apk, sref

gdcon

NAME

gdcon - Gaussian deconvolution

DESCRIPTION

The command ***gdcon*** deconvolves the spectrum by fitting a Gaussian function to the peaks. It is typically used for overlapping peaks with a Gaussian lineshape to determine the ratio of each individual peak. ***gdcon*** only works on the plot region, as determined by the parameters F1P and F2P. Furthermore, it selects peaks according to the peak picking parameters MI, MAXI and PC (see ***pp***).

gdcon evaluates the parameter AZFW which determines the minimum distance between two peaks for them to be fitted independently. Peaks which are less than AZFW ppm apart, are considered to be overlapping. As a rule of the thumb, you can set AZFW to ten times the width at half height of the signal.

The result of ***gdcon***, the fitted lineshape, is stored as the so-called second data-set (specified with ***edc2***). This is typically the next processed data number (procno). As such, the deconvolved spectrum can be compared with the original one in ***dual*** display mode.

Another result of ***gdcon*** is a list of peaks within the plot region, and for each peak its frequency, width, intensity and area. This list is displayed on the screen, sent to the printer or stored into a file, depending on the value of CURPRIN (type ***edo***). Under Windows, you can also set CURPRIN to Clipboard or Enhanced Metafile.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***azfw***, ***f1*** etc.:

AZFW - minimum distance in ppm for peaks to be fitted independently
F1 (F1P) - low field (left) limit of the deconvolution region (= plot region)
F2 (F2P) - high field (right) limit of the deconvolution region (= plot region)
MI, MAXI, PC - peak picking parameters (see ***pp***)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`procs` - processing status parameters

USAGE IN AU PROGRAMS

GDCON

SEE ALSO

ldcon, mdcon, edc2

genfid

NAME

genfid - generate pseudo-raw 1D data

SYNTAX

genfid [<expno> [<name> [<user> [<du>]]]] [y] [n]

DESCRIPTION

The command ***genfid*** generates pseudo-raw data from processed data. This command is normally used in combination with the command ***ift*** which performs an inverse Fourier transform, converting a spectrum into an FID. In fact, ***ift*** transforms processed frequency domain data into processed time domain data. ***genfid*** converts these processed time domain data into pseudo-raw time domain data and stores them under a new name or experiment number (expno).

Note that ***genfid*** does not modify the data, but only stores them in a different format. The number of data points of the pseudo-raw data, is twice the size (SI) of the processed data they are created from. The acquisition status parameter TD (type ***dpa***) is set accordingly; TD = 2*SI.

genfid takes six arguments and an be used as follows:

1. ***genfid***
   You will be prompted for the *expno* under which the FID must be stored
2. ***genfid <expno>***
   The FID will be stored under the specified *expno*.
3. ***genfid <expno> <name> y***
   The FID will be stored under the specified *name* and *expno*. The last argument (*y*) causes ***genfid*** to overwrite possibly existing data.
4. ***genfid <expno> <name> <user> <du> y n***
   The output will be stored under the specified *expno*, *name*, *user* and *disk unit*. The second last argument (*y*) causes ***genfid*** to overwrite possibly existing data. The last argument (*n*) causes XWIN-NMR to keep the display on the input dataset rather than change it to the output dataset.

You can use any other combination of arguments as long they are entered in the correct order. Note that the last argument in example 3 and the last two argu-

ments in example 5, can only take the values *y* and *n*, respectively. The processed data number (*procno*) of the output dataset is always set to 1.

genfid can be used if you want to reprocess a 1D spectrum, for example with different processing parameters, but the raw data do not exist any more. An example of such a procedure is:

ift (if the data are Fourier transformed)

genfid (to create the pseudo-raw data)

edp (to set the processing parameters)

ef (to process the pseudo-raw data)

If the input data are processed but not Fourier transformed, you can skip the first step.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed time domain data (real, imaginary)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - pseudo-raw data
`audita.txt` - acquisition audit trail

USAGE IN AU PROGRAMS

GENFID(expno)
overwrites possibly existing raw data in the specified expno

SEE ALSO

ift, genser

gf, gfp

NAME

gf - Gaussian window multiplication + Fourier transform
gfp - Gaussian window multiplication + Fourier transform + phase correction

DESCRIPTION

The composite processing command ***gf*** is a combination of ***gm*** and ***ft***, i.e. it performs a Gaussian window multiplication and a Fourier transform.

gfp is a combination of ***gm***, ***ft*** and ***pk***, i.e. it does the same as ***gf*** but, in addition, performs a phase correction.

gf and ***gfp*** automatically perform an FID baseline correction according to BC_mod.

INPUT AND OUTPUT PARAMETERS

see ***gm***, ***ft*** and ***pk***

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

GF

GFP

SEE ALSO

ef, efp, fp, fmc, em, gm, ft, pk, mc

gm

NAME

gm - Gaussian window multiplication of the FID

DESCRIPTION

The command ***gm*** performs an Gaussian window multiplication of the FID. The result is a more Gaussian lineshape (with sharper edges) after Fourier transform. ***gm*** multiplies the FID with the function:

$$\exp(((-at)) - (-bt^2))$$

where t is the acquisition time in seconds and a and b are defined by:

$$a = \pi \cdot LB \quad \text{and} \quad b = -\frac{a}{2GB \cdot AQ}$$

In these equation, LB and GB are processing parameters which represent the exponential broadening factor and the Gaussian broadening factor, respectively. AQ is an acquisition status parameter which represents the acquisition time.

gm allows you to separate overlapping peaks. The quality of the separation depends on the choice of the parameters LB and GB. Suitable values can be determined interactively in the window function menu (command ***winfunc***). The value of LB must be negative, typically the half line width of the spectral peaks. Note that for exponential window multiplication (***em***), LB must be positive. The value of GB must lie between 0 and 1. It determines the position of the top of the Gaussian function. For example, for GB = 0.5 the top lies in the middle of the FID. Note that for large values of GB (close to 1), peaks can become negative at the edges which can impair quantitative analysis of the spectrum.

The command ***gm*** implicitly performs a baseline correction of the FID, according to the processing parameter BC_mod.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***lb***, ***gb*** etc.:

LB - Lorentzian broadening factor
GB - Gaussian broadening factor
BC_mod - FID baseline correction mode

set by the acquisition, can be viewed with ***dpa*** or by typing ***ls aq_mod***:

AQ - acquisition time

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

GM

SEE ALSO

em, gf, gfp, bc

ht

NAME

ht - 1D Hilbert transform

DESCRIPTION

The command ***ht*** performs a Hilbert transform which means the imaginary part of a spectrum is calculated from the real part. This is only useful when the real data have been created from zero filled raw data. Only then, will they contain the entire spectral information.

Imaginary data are required for phase correction. They are normally created together with the real data by Fourier transform. Directly after the Fourier transform, real and imaginary data are consistent and can be used for phase correction. If, however, the real data are manipulated, e.g. by ***abs***, they are no longer consistent with the imaginary data. In that case, or when the imaginary data have been deleted, ***ht*** can be used to create new imaginary data.

Hilbert transform is based on the so called dispersion relations or Kramers-Kronig relations (see, for example, R. R. Ernst, G. Bodenhausen and A. Wokaun, Principles of nuclear magnetic resonance in one and two dimensions, Clarendon Press, Oxford, 1987).

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1i` - imaginary processed data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

HT

SEE ALSO

ft, ift, trf, trfp

ift

NAME

ift - inverse Fourier transform

DESCRIPTION

The command ***ift*** performs an inverse Fourier transform of a 1D spectrum, thus creating an artificial FID. Normally, ***ift*** is done when the raw data do not exist any more. If, however, raw data do exist, they are not overwritten. ***ift*** stores the resulting FID as processed data, i.e. it overwrites the current spectrum.

After ***ift***, you can create pseudo-raw data with the command ***genfid*** which creates a new dataset. Note that the number of data points of the pseudo-raw data, is twice the size of the processed data they are created from. The acquisition status parameter TD (***dpa***) is set accordingly; TD = 2*SI.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (frequency domain)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (time domain)
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

IFT

SEE ALSO

genfid, ft, trf, trfp

ldcon

NAME

ldcon - Lorentzian deconvolution

DESCRIPTION

The command ***ldcon*** deconvolves the spectrum fitting a Lorentzian function to the peaks. It is typically used for overlapping peaks with a Lorentzian lineshape to determine the ratio of each individual peak. ***ldcon*** only works on the plot region, as determined by the parameters F1P and F2P. Furthermore, it selects peaks according to the peak picking parameters MI, MAXI and PC (see ***pp***).

ldcon evaluates the parameter AZFW. For AZFW = 0, all peaks are fitted independently. For AZFW > 0, peaks which are further apart than AZFW are fitted independently and peaks which are less than AZFW apart are considered to be overlapping. The default value of AZFW, as defined in the Bruker standard parameter sets, is 0.1.

The result of a ***ldcon***, the fitted lineshape, is stored as the so-called second dataset (specified with ***edc2***). This is typically the next processed data number (procno). As such, the deconvolved spectrum can be compared with the original one in ***dual*** display mode.

Another result of ***ldcon*** is a list of peaks within the plot region, and for each peak its frequency, width, intensity and area. This list is displayed on the screen, sent to the printer or stored into a file, depending on the value of CURPRIN (type ***edo***). Under Windows, you can also set CURPRIN to Clipboard or Enhanced Metafile.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***azfw***, ***f1*** etc.

AZFW - minimum distance in ppm for peaks to be fitted independently
F1 (F1P) - low field (left) limit of the deconvolution region (= plot region)
F2 (F2P) - high field (right) limit of the deconvolution region (= plot region)
MI, MAXI, PC - peak picking parameters (see ***pp***)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`procs` - processing status parameters

USAGE IN AU PROGRAMS

LDCON

SEE ALSO

gdcon, mdcon, edc2

ls

NAME

ls - left shift data NSP points

DESCRIPTION

The command ***ls*** shifts the data to the left. The number of points shifted is determined by the parameter NSP. The right end of the data is filled with NSP zeroes. Depending on the parameter DATMOD, ***ls*** works on raw or processed data.

The value of NSP is the number of the real plus imaginary data points that are shifted. As such, the real data are shifted NSP/2 points and the imaginary data are shifted NSP/2 points. For odd values of NSP the real and imaginary data points are interchanged. As such the displayed spectrum is not only shifted to the left but also changes from real (absorption) to imaginary (dispersion) or vice versa. Note that his only plays a role for DATMOD = proc.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***nsp***, ***datmod*** etc.:

NSP - number of points to be shifted
DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if DATMOD = raw)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters

`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

LS

SEE ALSO

rs, pk

mc

NAME

mc - magnitude calculation

DESCRIPTION

The command ***mc*** calculates the magnitude spectrum. The intensity of each point *i* is replaced by its absolute value according to the formula:

$$ABS(i) = \sqrt{(R(i)^2 + I(i)^2)}$$

where R and I are the real and imaginary part of the spectrum, respectively. If no processed input data exist, ***mc*** works on the raw data.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw 1D data (input if `1r`, `1i` do not exist)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (input if they exist)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

MC

SEE ALSO

fmc, pk, ps

mdcon

NAME

mdcon - mixed Gaussian/Lorentzian deconvolution

DESCRIPTION

The command ***mdcon*** deconvolves the spectrum by fitting a mixed Lorentzian/Gaussian function to the peaks. It is typically used to deconvolve spectra which cannot be approximated by a pure Lorentzian or a pure Gaussian lineshape. ***mdcon*** only works on the plot region, as determined by the parameters F1P and F2P. Furthermore, it selects peaks according to the peak picking parameters MI, MAXI and PC (see ***pp***).

Before you can use ***mdcon***, you must first run ***ppp*** to create a special peak list. Then you must edit this list with ***edmisc peaklist***, specify the percentage of Gaussian lineshape for each peak, store the list and finally run ***mdcon***.

mdcon evaluates the parameter AZFW which determines the minimum distance between two peaks for them to be fitted independently. Peaks which are less than AZFW ppm apart, are considered to be overlapping. As a rule of the thumb, you can set AZFW to ten times the width at half height of the signal.

The result of a deconvolution command, the fitted lineshape, is stored as the so-called second dataset (specified with ***edc2***). This is typically the next processed data number (procno). As such, the deconvolved spectrum can be compared with the original one in ***dual*** display mode.

Another result of ***mdcon*** is a list of peaks within the plot region, and for each peak its frequency, width, intensity, area and the percentage of Lorentzian line shape used. This list is displayed on the screen, sent to the printer or stored into a file, depending on the value of CURPRIN (type ***edo***). Under Windows, you can also set CURPRIN to Clipboard or Enhanced Metafile.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***azfw***, ***f1*** etc.:

AZFW - minimum distance in ppm for peaks to be fitted independently
F1 (F1P) - left limit of the deconvolution region (= plot region)
F2 (F2P) - right limit of the deconvolution region (= plot region)

MI, MAXI, PC - peak picking parameters (see ***pp***)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`proc` - processing parameters
`peaklist` - peak list created with ***ppp***

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`procs` - processing status parameters

USAGE IN AU PROGRAMS

MDCON

SEE ALSO

ldcon, gdcon, edc2, ppp

mul

NAME

mul - multiply two datasets

DESCRIPTION

The command ***mul*** multiplies two datasets. The input data must be specified as the second and third dataset with the command ***edc2***. The result is stored in the current dataset.

Depending on the value of DATMOD, ***mul*** works on raw or processed data. For DATMOD = raw, ***mul*** divides the raw data of the second and third dataset but stores the result as processed data in the current dataset.

With XWIN-NMR 4.0 and newer, ***mul*** performs a complex multiplication on complex spectra. This requires for both input datasets that:

- the status parameter FT_mod = fqc or fsc
- real (file `1r`) and imaginary (file `1i`) data exist

This is the case for most Avance data. In XWIN-NMR 3.1 or older, or if the above requirements are not fulfilled, real and imaginary data are multiplied pointwise. When complex multiplication has been performed, this is reported in the audit trail output file.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***datmod*** :

DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - processing parameters

<du2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)

<du3>/data/<user3>/nmr/<name3>/<expno3>/pdata/<procno3>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)

Note that *du*, *user* and *name* of the current, second and third dataset are often the same and only the *expno* and/or *procno* are different. For DATMOD = raw, the files `fid` under *expno2* and *expno3* are input.

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

MUL

SEE ALSO

add, addc, addfid, mulc, div, and, or, xor, edc2

mulc

NAME

mulc - multiply a dataset with a constant

DESCRIPTION

The command ***mulc*** multiplies the current dataset with a constant. The value of the constant is determined by the processing parameter DC.

mulc works on raw or processed data, depending on the parameter DATMOD.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***dc***, ***datmod*** etc.:

DC - multiplication factor
DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if DATMOD = raw)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - input data (input if DATMOD = proc)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

MULC

SEE ALSO

add, addc, addfid, mul, div, and, or, xor, edc2

nm

NAME

nm - negate data

DESCRIPTION

The command ***nm*** negates the current data which means all data points are multiplied by -1.

nm works on raw or processed data, depending on the parameter DATMOD.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***datmod*** :

DATMOD - data mode: work on 'raw' or 'proc'essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if DATMOD = raw)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - input data (input if DATMOD = proc)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

NM

SEE ALSO

mulc, zp

or

NAME

or - combine two datasets according to a logical ’or’

DESCRIPTION

The command ***or*** combines two datasets according to a logical ’or’ (boolean operation). The input data must be specified as the second and third dataset with the command ***edc2***. The result is stored in the current dataset.

Depending on the value of DATMOD, ***or*** works on raw or processed data. For DATMOD = raw, ***or*** combines the raw data of the second and third dataset but stores the result as processed data in the current dataset.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***datmod*** :

DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - processing parameters

<du2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)

<du3>/data/<user3>/nmr/<name3>/<expno3>/pdata/<procno3>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)

Note that *du*, *user* and *name* of the current, second and third dataset are often the same and only the *expno* and/or *procno* are different. For DATMOD = raw, the files `fid` under *expno2* and *expno3* are input.

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters

`auditp.txt` - processing audit trail

SEE ALSO

add, addc, addfid, mul, mulc, div, and, xor, edc2

pk

NAME

pk - phase correction according to user defined phase values

DESCRIPTION

The command ***pk*** performs a zero and first order phase correction according to user defined phase values. These phase values are read from the processing parameters PHC0 and PHC1.

The data, consisting of real points R(i) and imaginary points I(i) is phase corrected according to the formula:

$$R0(i) = R(i)\cos a(i) - I(i)\sin a(i)$$
$$I0(i) = I(i)\cos a(i) + R(i)\sin a(i)$$

where:

$$a(i) = PHC0 + (i-1)PHC1$$

where i > 0, R0 and I0 represent the corrected values and PHC0 and PHC1 are processing parameters.

pk does not calculate the phase values but uses the preset values. Therefore, ***pk*** is only useful when these are known. The phase values can be determined interactively from the ***phase*** menu or automatically with ***apk*** or ***apks***.

pk is typically used in a series of experiments where the first spectrum is corrected with ***apk*** and each successive spectrum with ***pk***, using the same values (see AU programs ***mulanal*** and ***proc_noe***).

pk applies but does not change the processing parameters PHC0 and PHC1 (***edp***). It does, however, change the corresponding processing status parameters (***dpp***), by adding the applied phase values.

pk is a part of the composite processing commands ***efp***, ***fp*** and ***gfp***.

pk can also be used to perform a phase correction on an FID rather than a spectrum. This is automatically done if you enter ***pk*** on a dataset which does not con-

tain processed data. Phase correction on an FID is used prior to Fourier transform to induce a shift in the resulting spectrum. The spectrum is shifted according to the value of PHC1; one real data point to the left for each 360°. A negative value of PHC1 causes a right shift. The points which are cut off on one side of the spectrum are appended on the other side. Note the difference with performing a left shift (***ls***) or right shift (***rs***) which appends zeroes at the opposite side. If processed data do exist and you still want to do a phase correction on the FID, you can do this with the command ***trf***.

INPUT PARAMETER

set by the user with ***edp*** or by typing ***phc0***, ***phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if no processed data exist)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

PK

SEE ALSO

ft, trf, trfp, mc, ps, apk, apks

ps

NAME

ps - calculate the power spectrum

DESCRIPTION

The command ***ps*** calculates the power spectrum of the current dataset, replacing the intensity of each data point *i* according to the formula:

$$PS(i) = R(i)^2 + I(i)^2$$

where R and I are the real and imaginary part of the spectrum, respectively. It can also work on the FID. The result is always stored as the real data.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if no processed data exist)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

PS

SEE ALSO

mc, pk

qsin

NAME

qsin - sine squared window multiplication

DESCRIPTION

The command ***qsin*** performs a sine squared window multiplication, according to the function:

$$QSIN(t) = (\sin(\pi - PHI) \cdot (t/AQ))^2$$

where

$$0 < t < AQ \text{ and } PHI = \pi/SBB$$

and SSB is a processing parameter.

Typically values are SSB = 1 for a pure sine function and SSB = 2 for a pure cosine function. Values greater than 2 give mixed sine/cosine functions. Note that all values smaller than 2 have the same effect as SSB = 1, namely a pure sine function.

qsin automatically performs an FID baseline correction according to BC_mod.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***ssb*** :

SSB - sine bell shift

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

QSIN

SEE ALSO

sinm, sinc, qsinc, em, gm, tm, traf, trafs, uwm

qsinc

NAME

qsinc - sinc squared window multiplication

DESCRIPTION

The command ***qsinc*** performs a sinc squared window multiplication, according to the function:

$$QSINC(t) = \left(\frac{\sin t}{t}\right)^2$$

where

$$-2\pi \cdot SBB \cdot GB < t < 2\pi \cdot SSB \cdot (1 - GB)$$

and SSB and GB are processing parameters.

qsinc automatically performs an FID baseline correction according to BC_mod.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***ssb***, ***gb*** etc.:

SSB - sine bell shift
GB - Gaussian broadening factor

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

SEE ALSO

sinc, sinm, qsin, em, gm, tm, traf, trafs, uwm

rs

NAME

rs - right shift the data NSP points

DESCRIPTION

The command ***rs*** shifts the data to the right. The number of points shifted is determined by the parameter NSP. The left end of the data is filled with NSP zeroes. Depending on the parameter DATMOD, ***rs*** works on raw or processed data.

The value of NSP is the number of the real plus imaginary data points that are shifted. As such, the real data are shifted NSP/2 points and the imaginary data are shifted NSP/2 points. For odd values of NSP the real and imaginary data points are interchanged. As such the displayed spectrum is not only shifted to the right but also changes from real (absorption) to imaginary (dispersion) or vice versa. Note that his only plays a role for DATMOD = proc.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***nsp***, ***datmod*** etc.:

NSP - number of points to be shifted
DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if DATMOD = raw)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters

`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

RS

SEE ALSO

ls, pk

rv

NAME

rv - reverse a 1D spectrum or FID

DESCRIPTION

The command ***rv*** reverses the data with respect to the middle data point, i.e. the leftmost data point becomes the rightmost point and vice versa. The real and imaginary part of the spectrum are thereby interchanged. Depending on the value of DATMOD, ***rv*** works on the raw or on the processed data. The result is always store as processed data.

A spectrum can also be reversed as a part of the Fourier transform by setting the processing parameter REVERSE to TRUE.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***datmod*** :

DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if DATMOD = raw)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

RV

sab

NAME

sab - spline baseline correction

DESCRIPTION

The command ***sab*** performs a spline baseline correction. This is based on a predefined set of data points which are considered to be a part of the baseline. The regions between these points are individually fitted.

Spline baseline correction involves the following steps:

1. Enter ***basl*** to change to the baseline menu.
2. Click ***def-pts*** to attach the cursor to the spectrum.
   (if the baseline points have been defined before, you are first prompted to append to (a) or overwrite (o) the existing list of points)
3. Move the cursor over the spectrum and click the middle mouse button at several positions which are part of the baseline.
4. Click the left mouse button to release the cursor from the spectrum.
5. Click ***return*** → ***Return*** to change to the main menu.
6. Enter ***sab*** to do baseline correction according to the points just defined.

The set of baseline points can be stored for general usage and applied to similar spectra as follows:

1. ***wmisc baslpnts <name>***
2. Read a similar dataset
3. ***rmisc baslpnts <name>***
4. ***sab***

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`baslpnts` - baseline points (points and ppm values)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

SAB

SEE ALSO

abs, absf, absd, bcm

sinc

NAME

sinc - sinc window multiplication

DESCRIPTION

The command ***sinc*** performs a sinc window multiplication, according to the function:

$$SINC(t) = \frac{\sin t}{t}$$

where

$$-2\pi \cdot SBB \cdot GB < t < 2\pi \cdot SSB \cdot (1 - GB)$$

and SSB and GB are processing parameters.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***ssb***, ***gb*** etc. :

SSB - sine bell shift
GB - Gaussian broadening factor

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

SEE ALSO

qsinc, sinm, qsin, em, gm, tm, traf, trafs, uwm

sinm

NAME

sinm - sine window multiplication

DESCRIPTION

The command ***sinm*** performs a sine window multiplication, according to the function:

$$\sin((\pi - PHI) \cdot (t/AQ) + PHI)$$

where

$$0 < t < AQ \text{ and } PHI = \pi/SBB$$

and SSB is a processing parameter.

Typically values are SSB = 1 for a pure sine function and SSB = 2 for a pure cosine function. Values greater than 2 give a mixed sine/cosine function. Note that all values smaller than 2, for example 0, have the same effect as SSB = 1, namely a pure sine function.

sinm automatically performs an FID baseline correction according to BC_mod.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***ssb*** :

SSB - sine bell shift

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

SINM

SEE ALSO

qsin, sinc, qsinc, em, gm, tm, traf, trafs, uwm

tm

NAME

tm - trapezoidal window multiplication of the FID

DESCRIPTION

The command ***tm*** performs a trapezoidal window multiplication of the FID. The rising and falling edge of this function are defined by the parameters TM1 and TM2. These represent a fraction of the acquisition time as displayed in figure 3.1.

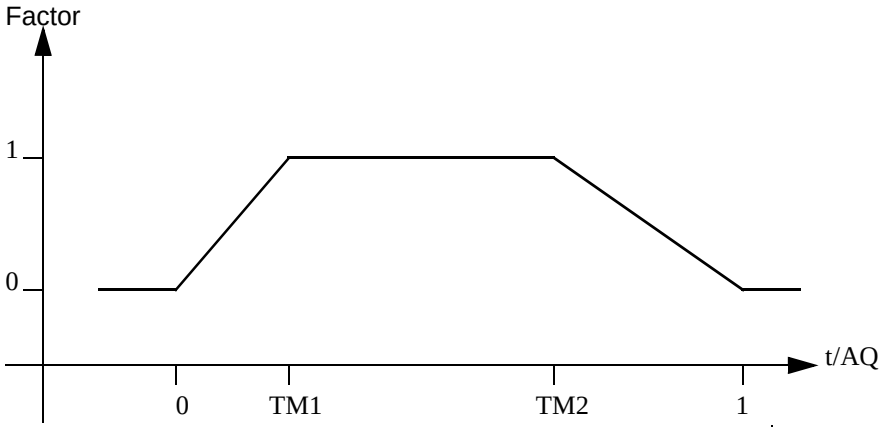

Figure 3.1

tm automatically performs an FID baseline correction according to BC_mod.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***tm1***, ***tm2*** etc.:

TM1 - the end of the rising edge of a trapeziodal window
TM2 - the start of the falling edge of a trapezoidal window

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if 1r, 1i do not exist or are Fourier transformed)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TM

SEE ALSO

em, gm, sinm, qsin, sinc, qsinc, uwm, traf, trafs

trf, trfp

NAME

trf - user defined processing of raw data
trfp - user defined processing of processed data

DESCRIPTION

The command ***trf*** processes the raw data performing the following steps:

- baseline correction according to BC_mod
- linear prediction according to ME_mod
- window multiplication according to WDW
- Fourier transform according to FT_mod
- phase correction according to PH_mod

trf offers the following features:

- when all parameters mentioned above are set to *no*, the raw data (file `fid`) are simply stored as processed data (files `1r`, `1i`). The even points are stored as real data (file `1r`) and the odd points as imaginary data (file `1i`). The size of these processed data and the number of input FID points are determined by the parameters SI and TDeff, as described for the command ***ft***. For example, if 0 < TDeff < TD, the processed data are truncated. This allows you to create an FID with a smaller size than the original one (see also the command ***genfid***).
- ***trf*** evaluates BC_mod for the baseline correction mode (e.g. quad, qpol or qfil) and detection mode (e.g. single or quad, spol or qpol, sfil or qfile). Note that the command ***bc*** evaluates the acquisition status parameter AQ_mod for the detection mode and ignores the BC_mod detection mode (see parameter BC_mod).
- ***trf*** evaluates WDW for the window multiplication mode (*em*, *gm*, **sine**, *qsine*, *trap*, *user*, *sinc*, *qsinc*, *traf* or *trafs*). This allows you to vary the window multiplication by varying the value of WDW rather than the window multiplication command. This can be useful in AU programs.
- the Fourier transform is performed according to FT_mod. Normally, the Fourier transform is done with the command ***ft*** which determines the Fourier transform mode from acquisition status parameter AQ_mod.

However, for some datasets, no value of AQ_mod translates to a correct Fourier transform mode. An example of this is when you read a column (with ***rsc***) from a 2D dataset which was measured with FnMODE (or MC2) = States-TPPI and Fourier transformed in the F2 dimension only. The resulting FID can only be Fourier transformed correctly with ***trf***. The parameter FT_mod is automatically set to the correct value by the ***rsc*** command. ***trf*** can also be used manipulate the acquisition mode of raw data by Fourier transforming the data with one FT_mod and inverse Fourier transforming them with a different FT_mod. From the resulting data you could create pseudo-raw data (using ***genfid***) with a different acquisition mode than the original raw data. Finally, ***trf*** allows you to process the data without Fourier transform (FT_mod = no). Table 3.3 shows a list of FT_mod values:

FT_mod	Fourier transform mode
no	no Fourier transform
fsr	forward, single channel, real
fqr	forward, quadrature, real
fsc	forward, single channel, complex
fqc	forward, quadrature, complex
isr	inverse, single channel, real
iqr	inverse, quadrature, real
isc	inverse, single channel, complex
iqc	inverse, quadrature, complex

Table 3.3

The command ***trfp*** works like ***trf*** except that it always works on processed data. If no processed data exist, ***trfp*** stops with an error message.

trfp can be used to perform multiple additive baseline corrections, to remove multiple frequency baseline distortions. This cannot be done with ***bc*** or ***trf*** because these commands always work on the raw data, i.e. they are not additive. Note that the window multiplication commands (e.g. ***em***, ***gm***, ***sine*** etc.) are additive. The same counts for linear prediction (part of ***ft***) and phase correction (***pk***).

trf can be used to do a combination of forward and backward prediction. Just run ***trf*** with ME_mod = LPfc and then ***trfp*** (or ***ft***) with ME_mod = LPbc.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***si***, ***stsr*** etc.:

SI - size of the processed data
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
TDeff - number of raw data points to be used for processing
FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)

set by the acquisition, can be viewed with ***dpa*** or by typing ***1s aq_mod*** etc.:

AQ_mod - acquisition mode (determines the Fourier transform mode)
TD - time domain; number of raw data points

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***si***, ***stsr*** etc.:

SI - size of the processed data
TDeff - number of raw data points to be used for processing
FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)
BC_mod - FID baseline correction mode
  BCFW - filter width for BC_mod = sfil or qfil
  COROFFS - correction offset for BC_mod = spol/qpol or sfil/qfil
ME_mod - FID linear prediction mode
  NCOEF - number of linear prediction coefficients
  LPBIN - number of points for linear prediction
  TDoff - number of raw data points predicted for ME_mod = LPb*
WDW - FID window multiplication mode
  LB - Lorentzian broadening factor for WDW = em or gm
  GB - Gaussian broadening factor for WDW = gm, sinc or qsinc
  SSB - Sine bell shift for WDW = sine, qsine, sinc or qsinc
  TM1, TM2 - limits of the trapezoidal window
FT_mod - Fourier transform mode
  REVERSE - flag indicating to reverse the spectrum
  PKNL - group delay handling (Avance) or filter correction (A*X)
  STSR - strip start: first output point of strip transform
  STSI - strip size: number of output points of strip transform

PH_mod - phase correction mode
PHC0 - zero order phase correction value for PH_mod = pk
PHC1 - first order phase correction value for PH_mod = pk

set by the acquisition, can be viewed with ***dpa*** or by typing ***ls td*** :

TD - time domain; number of raw data points

OUTPUT PARAMETERS

can be viewed with ***dpp*** or by typing ***ls ft_mod***, ***ls tdeff*** etc.:

TDeff - number of raw data points that were used for processing
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
NC_proc - intensity scaling factor
YMAX_p - maximum intensity of the processed data
YMIN_p - minimum intensity of the processed data

can only be viewed by typing ***ls bytordp***:

BYTORDP - data storage order

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input of ***trf)***

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (input of ***trfp***)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TRF

TRFP

SEE ALSO

ft, ef, efp, gf, gfp, fmc, ift, bc, em, pk

traf, trafs

NAME

traf - Traficante window multiplication of the FID
trafs - Traficante window multiplication of the FID

DESCRIPTION

The commands ***traf*** and ***trafs*** perform window multiplication of the FID. The algorithms used by ***traf*** and ***trafs*** are described by D. D. Traficante and G. A. Nemeth in J. Magn. Res., **71** (1987) 237).

traf and ***trafs*** automatically perform an FID baseline correction according to BC_mod.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if `1r`, `1i` do not exist or are Fourier transformed)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed data (input if they exist but are not Fourier transformed)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`procs` - processing status parameters
`auditp.txt` - processing audit trail

SEE ALSO

em, gm, sinm, qsin, sinc, qsinc, tm, uwm

uwm

NAME

uwm - user defined window multiplication

DESCRIPTION

The command ***uwm*** performs a user defined window multiplication. The window function must be stored as FID under a separate data name or experiment number. This dataset must then be defined as the second dataset with the command ***edc2***. The command ***uwm*** multiplies the FID of the current dataset with the FID of the second dataset, and stores the result as processed data of the current dataset. The number of points of the windows function must be equal to or greater than the number of points of the current FID. In the latter case, the window is truncated before multiplication is applied. Type ***ls td*** on the current and on the second dataset to check this.

Before you can use the command ***uwm***, you must first create the window function. One way of doing that is the following:

1. Type ***new*** and define a new dataset
2. Type ***edau calfun*** to edit the Bruker AU program ***calfun***. Modify it according to the window function you want to create. The header of ***calfun*** describes how to do that. Leave the editor, saving and compiling the AU program. Then enter ***calfun*** to execute the AU program and actually create the window function.

INPUT PARAMETERS

set by the acquisition, can be viewed with ***dpa*** or by typing ***ls td*** :

TD - time domain; number of raw data points (size of the FID)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data of the current dataset

<du2>/data/<user2>/nmr/<name2>/<expno2>/

`fid` - window function as defined in the second dataset

`acqus` - acquisition status parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (time domain)
`procs` - processing status parameters

USAGE IN AU PROGRAMS

UWM

SEE ALSO

edc, new, edc2, em, gm, sinm, qsin, sinc, qsinc, tm, traf, trafs

xor

NAME

xor - combine two datasets according to a logical ’xor’

DESCRIPTION

The command ***xor*** combines two datasets according to a logical ’xor’ (boolean operation) which is an exclusive ’or’. The input data must be specified as the second and third dataset with the command ***edc2***. The result is stored in the current dataset.

Depending on the value of DATMOD, ***xor*** works on raw or processed data. For DATMOD = raw, ***xor*** combines the raw data of the second and third dataset but stores the result as processed data in the current dataset.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***datmod*** :

DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - processing parameters

<du2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)

<du3>/data/<user3>/nmr/<name3>/<expno3>/pdata/<procno3>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)

Note that *du*, *user* and *name* of the current, second and third dataset are often the same and only the *expno* and/or *procno* are different. For DATMOD = raw, the files `fid` under *expno2* and *expno3* are input.

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)

`procs` - processing status parameters
`auditp.txt` - processing audit trail

SEE ALSO

add, addc, addfid, mul, mulc, div, and, or, edc2

zf

NAME

zf - zero data

DESCRIPTION

The command ***zf*** sets the intensity of all data points to zero. Depending on the value of DATMOD, ***zf*** works on raw or processed data. The result is always stored as processed data, the raw data are never overwritten.

The output of ***zf*** is usually the same for DATMOD = raw or processed, namely SI processed data point with zero intensity. However, for DATMOD = proc, the existing processed data are set to zero whereas for DATMOD = raw, new processed data are created according to the current processing parameters. The result is different when the data have been Fourier transformed with STSI < SI. ***zf*** with DATMOD = proc creates STSI zeroes whereas ***zf*** with DATMOD = raw creates SI zeroes. The reason is that ***zf*** with DATMOD = raw reprocesses the raw data but does not interpret STSI since no Fourier transform is done.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***datmod***, ***si*** etc.:

DATMOD - data mode: work on ’raw’ or ’proc’essed data
SI - size of the processed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if DATMOD = raw)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc and output)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data

`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

ZF

SEE ALSO

zp

zp

NAME

zp - zero the first NZP points of a dataset

DESCRIPTION

The command ***zp*** sets the intensity of the first NZP points of the dataset to zero. It works on raw or processed data depending on the value of DATMOD. The parameter NZP can take a value between 0 and the size of the FID or spectrum.

The value of NZP is the number of the real plus imaginary data points that are zeroed. As such, the first (NZP+1)/2 real points and the first NSP/2 imaginary data points are zeroed.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***nzp***, ***datmod*** etc.:

NZP - number of data points set to zero intensity
DATMOD - data mode: work on ’raw’ or ’proc’essed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - raw data (input if DATMOD = raw)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (input if DATMOD = proc)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data (real, imaginary)
`procs` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

ZP

SEE ALSO

zf

Chapter 4

2D processing commands

This chapter describes all XWIN-NMR 2D processing commands. Most of them only work on 2D data but some, e.g. ***xfb***, can also be used to process a plane of 3D data. They store their output in processed data files and do not change the raw data.

We will often refer to the two dimensions of a 2D dataset as the F2 and F1 dimension. F2 is the acquisition dimension which is displayed horizontally and F1 the orthogonal dimension which is displayed vertically. The names of most 2D processing commands express the dimension in which they work, e.g. ***xf2*** works in F2, ***xf1*** in F1 and ***xfb*** in both dimensions. F2 traces are usually referred to as rows, F1 traces as columns. Some commands express this terminology, e.g. ***rsr*** reads and stores rows and ***rsc*** reads and stores columns of a 2D spectrum.

For each command, the relevant input and output parameters are mentioned. Furthermore, the relevant input and output files and their location are mentioned. Although file handling is completely transparent, it is sometimes useful to know which files are involved and where they reside. For example, if you have permission problems or if you want to process or interpret your data with third party software.

abs1, absd1

NAME

abs1 - automatic baseline correction in the F1 dimension (columns)
absd1 - automatic baseline correction in F1 with a different algorithm

DESCRIPTION

The command ***abs1*** performs an automatic baseline correction in the F1 dimension. This means it subtracts a polynomial from the columns of the processed 2D data. The degree of the polynomial is determined by the parameter ABSG which has a value between 0 and 5, with a default of 5. It works like ***absf*** in 1D which means it only corrects the spectral region between ABSF1 and ABSF2.

absd1 works like ***abs1***, except that it uses a different algorithm[1]. It is, for example, used when a small peak lies on the foot of a large peak. In that case, ***absd1*** allows you to correct the baseline around the small peak which can then be integrated. Usually ***absd1*** is followed by ***abs1***.

INPUT PARAMETERS

F1 parameters

set by the user with ***edp*** or by typing ***1 absg***, ***1 absf1*** etc.:

ABSG - degree of the polynomial to be subtracted (0 to 5, default is 5)
ABSF1 - low field limit of the region which is baseline corrected
ABSF2 - high field limit of the region which is baseline corrected

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`proc2` - F1 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

1. It uses the same algorithm as the command ***abs*** in DISNMR

`proc2s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

ABS1

ABSD1

SEE ALSO

abs2, absd2, abst1, abst2, absot1, absot2, bcm1, bcm2

abs2, absd2

NAME

abs2 - automatic baseline correction in the F2 dimension (rows)
absd2 - automatic baseline correction in F2 with a different algorithm

DESCRIPTION

The command ***abs2*** performs an automatic baseline correction in the F2 dimension. This means it subtracts a polynomial from the rows of the processed 2D data. The degree of the polynomial is determined by the parameter ABSG which has a value between 0 and 5, with a default of 5. It works like ***absf*** in 1D which means it only corrects the spectral region between ABSF1 and ABSF2.

absd2 works like ***abs2***, except that it uses a different algorithm[1]. It is, for example, used when a small peak lies on the foot of a large peak. In that case, ***absd2*** allows you to correct the baseline around the small peak which can then be integrated. Usually ***absd2*** is followed by ***abs2***.

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***absf1***, ***absf2*** etc.:

ABSG - degree of the polynomial to be subtracted (0 to 5, default is 5)
ABSF1 - low field limit of the region which is baseline corrected
ABSF2 - high field limit of the region which is baseline corrected

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`proc` - F2 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

1. It uses the same algorithm as the command ***abs*** in DISNMR

`procs` - F2 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

ABS2

ABSD2

SEE ALSO

abs1, absd1, abst1, abst2, absot1, absot2, bcm1, bcm2

abst1, absot1

NAME

abst1 - automatic selective baseline correction in the F1 dimension (columns)
absot1 - automatic selective baseline correction in F1 with a different algorithm

DESCRIPTION

The command ***abst1*** performs an automatic selective baseline correction in the F1 dimension. This means it corrects the columns of the processed 2D data. It works like ***abs1*** except for the following:

- only the columns between F2-ABSF2 and F2-ABSF1 are corrected
- the part (region) of each column which is corrected shifts from column to column. The first column is corrected between F1-ABSF2 and F1-ABSF1. The last column is corrected between F1-SIGF2 and F1-SIGF1. For intermediate columns, the low field limit is an interpolation of F1-ABSF2 and F1-SIGF2 and the high field limit is an interpolation of F1-ABSF1 and F1-SIGF1.

absot1 works like ***abst1***, except that it has a different algorithm which applies a larger correction.

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***absf1***, ***absf2*** etc.:

ABSF1 - low field limit that defines the first column to be corrected
ABSF2 - high field limit that defines the last column to be corrected

F1 parameters

set by the user with ***edp*** or by typing ***1 absf1***, ***1 absf2*** etc.:

ABSF1 - low field limit of the correction region in the first column
ABSF2 - high field limit of the correction region in the first column
SIGF1 - low field limit of the correction region in the last column
SIGF2 - high field limit of the correction region in the last column

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`proc2` - F1 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`proc2s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

ABST1

ABSOT1

SEE ALSO

abst2, absot2, abs2, abs1, absd2, absd1, bcm2, bcm1

abst2, absot2

NAME

abst2 - automatic selective baseline correction in the F2 dimension (rows)
absot2 - automatic selective baseline correction in F2 with a different algorithm

DESCRIPTION

The command ***abst2*** performs an automatic selective baseline correction in the F2 dimension. This means it correct the rows of the processed 2D data. It works like ***abs2*** except for the following:

- only the rows between F1-ABSF2 and F1-ABSF1 are corrected
- the part (region) of each row which is corrected shifts from row to row. The first row is corrected between F2-ABSF2 and F2-ABSF1. The last row is corrected between F2-SIGF2 and F2-SIGF1. For intermediate rows, the low field limit is an interpolation of F2-ABSF2 and F2-SIGF2 and the high field limit is an interpolation of F2-ABSF1 and F2-SIGF1.

absot2 works like ***abst2***, except that it has a different algorithm which applies a larger correction.

INPUT PARAMETERS

F1 parameters

set by the user with ***edp*** or by typing ***1 absf1***, ***1 absf2*** etc.:

ABSF1 - low field limit that defines the first row to be corrected
ABSF2 - high field limit that defines the last row to be corrected

F2 parameters

set by the user with ***edp*** or by typing ***absf1***, ***absf2*** etc.:

ABSF1 - low field limit of the correction region in the first row
ABSF2 - high field limit of the correction region in the first row
SIGF1 - low field limit of the correction region in the last row
SIGF2 - high field limit of the correction region in the last row

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`proc` - F2 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`procs` - F2 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

ABST2

ABSOT2

SEE ALSO

abst1, absot1, abs1, abs2, absd1, absd2, bcm1, bcm2

add2d

NAME

add2d - add or subtract two 2D datasets

DESCRIPTION

The command ***add2d*** adds or subtracts two 2D spectra. The input data are the current dataset and the second dataset. The latter one must be defined with the ***edc2*** command. ***add2d*** performs the following operation:

current = ALPHA*current + GAMMA*second

where ALPHA and GAMMA are processing parameters. Both real and imaginary data are added. The result is stored in the current dataset, i.e. the current processed data are overwritten.

Caution: the two 2D datasets to be added must have equal sizes.

For APLHA = 1 and GAMMA = -1, the spectra are subtracted.

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***alpha***, ***gamma*** etc.:

ALPHA - multiplication factor of the current spectrum
GAMMA - multiplication factor of the second spectrum

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed data of the current dataset
`proc` - F2 processing parameters

<du2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

`2rr`, `2ir`, `2ri`, `2ii` - processed data of the second dataset

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed data
`procs` - F2 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

ADD2D

SEE ALSO

add, addfid

bcm2, bcm1

NAME

bcm2 - user defined baseline correction in the F2 dimension (rows)
bcm1 - user defined baseline correction in the F1 dimension (columns)

DESCRIPTION

The command ***bcm2*** performs a baseline correction in the F2 dimension by subtracting a polynomial, sine or exponential function. Before you can use ***bcm2***, you must first do the following:

1. Read a row with ***rsr*** (XWIN-NMR automatically changes to the 1D menu)
2. Enter ***basl*** to change to the baseline menu.
3. Click ***polynom***, ***sine*** or ***expon*** to select the baseline correction function
4. Fit the baseline of the spectrum with the function you selected in step 2 (initially represented by a straight horizontal line). Start by pressing button ***A*** and moving the mouse to determine the zero order correction. Continue with the buttons ***B***, ***C*** for higher order corrections until the line matches the baseline of the spectrum.
5. Click ***return*** → ***Save & return*** to change the main menu.
6. Click the ***2D*** button to return to the 2D menu

Then you can enter ***bcm2*** to perform the baseline correction.

bcm2 only works on the real data. This means the imaginary data no longer match the real data after ***bcm2*** and cannot be used for phase correction.

bcm1 works like ***bcm2***, except that it performs a baseline correction in the F1 dimension (columns). Before you can use ***bcm1***, you must read a column with ***rsc*** and define the baseline on it.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`base_info` - baseline correction coefficients

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

BCM2

BCM1

SEE ALSO

abs2, abs1, absd2, absd1, abst2, abst1, absot2, absot1

dosy2d

NAME

dosy2d - process a 2D DOSY dataset

DESCRIPTION

The command ***dosy2d*** processes a 2D DOSY dataset.

DOSY is a special representation of diffusion measurements. Instead of generating just numbers using the T1/T2 fitting package (i.e. diffusion coefficients and error values), the DOSY processing gives pseudo 2D data where the F1 axis displays diffusion constants rather than NMR frequencies.

At the time of this writing, only exponential fitting (up to 3 exponentials) is supported.

For more information on ***dosy2d***, refer to the manual "DOSY and Diffusion" under ***Help → Other topics***.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`difflist` - list of gradient amplitudes in Gauss/cm

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - 2D data processed in F2 only
`dosy` - DOSY processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - 2D processed data
`auditp.txt` - processing audit trail

SEE ALSO

eddosy, dosy3d

f2disco, f1disco

NAME

f2disco - calculate disco projection of the F2 dimension
f1disco - calculate disco projection of the F1 dimension

SYNTAX

f2disco [<firstrow> [<lastrow> [<refcol> [<procno> [y]]]]]
f1disco [<firstcol> [<lastcol> [<refrow> [<procno> [y]]]]]

DESCRIPTION

The command ***f2disco*** calculates the disco projection of the F2 dimension and writes it to a 1D dataset. It takes five arguments:

firstrow - the first row to be added or subtracted
lastrow - the last row to be added or subtracted
refcol - the reference column determining addition or subtraction
procno - the processed data number of the resulting 1D dataset
chdata - change the display to the output 1D dataset or not (*y* or *n*, default *n*)

Here are some examples of the usage of ***f2disco***:

f2disco
prompts for *firstrow*, *lastrow* and *refrow* and stores the disco projection under data name ~TEMP

f2disco <firstrow>
prompts for *lastrow* and *refrow* and stores the disco projection under data name ~TEMP

f2disco <firstrow> <lastrow> <refrow>
stores the specified disco projection under data name ~TEMP

f2disco <firstrow> <lastrow> <refrow> <procno>
stores the specified disco projection under the specified procno of the current data name

f2disco <firstrow> <lastrow> <refcol> <procno> y
stores the specified disco projection under the specified procno of the current data name and changes the display to this procno

Like ***f2sum***, ***f2disco*** calculates the sum of all rows between *firstrow* and *lastrow*. However, for each row, the intensity at the intersection with the reference column is determined. If this intensity is positive, the row is added to the total. If it is negative, the row is subtracted from the total.

f1disco works like ***f2disco***, except that it calculates the sum of the specified columns considering the intensities at the intersections with a reference row.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<*procno*>/

`1r`, `1i`- 1D spectrum containing the F1 disco projection
`auditp.txt` - processing audit trail

If the commands are used with less than four arguments, the files are stored in:

<du>/data/<user>/nmr/~*TEMP*/1/pdata/1/

USAGE IN AU PROGRAMS

F2DISCO(firstrow, lastrow, refcol, procno)

F1DISCO(firstcol, lastcol, refrow, procno)

for procno = -1, the disco projection is written to the dataset ~TEMP

SEE ALSO

f2projn, f2projp, f1projn, f1projp, f2sum, f1sum, proj, rhpp, rhnp, rvpp, rvnp

f2projn, f2projp, f1projn, f1projp

NAME

f2projn - calculate negative partial projection of the F2 dimension
f2projp - calculate positive partial projection of the F2 dimension
f1projn - calculate negative partial projection of the F1 dimension
f1projp - calculate positive partial projection of the F1 dimension

SYNTAX

f2projn [<firstrow> [<lastrow> [<procno> [y]]]]

f1projn [<firstcol> [<lastcol> [<procno> [y]]]]

The syntax of f2projp, f1projp are the same as for f2projn, f1projn, respectively.

DESCRIPTION

The commands ***f2proj**** and ***f1proj**** calculate partial 1D projections of the 2D dataset in the F2 and F1 dimension, respectively. A projection is a 1D trace where every point has the highest intensity of all points of the corresponding orthogonal trace in the 2D spectrum. Partial means that only a specified range of rows (or columns) is are evaluated, i.e. only a part of the orthogonal trace is scanned for the highest intensity. Negative projections contain only negative intensities, positive projections contain only positive intensities.

The commands ***f2proj**** takes four arguments:

firstrow - the first row to be evaluated
lastrow - the last row to be evaluated
procno - the processed data number of the resulting 1D dataset
chdata - change the display to the output 1D dataset or not (*y* or *n*, default *n*)

Here are some examples of the usage of ***f2projn*** :

f2projn
prompts for *firstrow* and *lastrow* and stores the projection under data name ~TEMP

f2projn <firstrow>
prompts for *lastrow* and stores the projection under data name ~TEMP

f2projn <firstrow> <lastrow>

stores the specified projection under data name ~TEMP

f2projn <firstrow> <lastrow> <procno>
stores the specified projection under the specified procno of the current data name

f2projn <firstrow> <lastrow> <procno> y
stores the specified projection under the specified procno of the current data name and changes the display to this procno

f2projp works like ***f2projn***, except that it calculates the positive partial projection in the F2 dimension.

f1projn works like ***f2projn***, except that it calculates the negative partial projection in the F1 dimension.

f1projp works like ***f2projn***, except that it calculates the positive partial projections in the F1 dimension.

The positive partial projections can also be calculated from the 2D utilities menu.

A special case is the command ***f1projp*** or ***f1projn*** on a hypercomplex 2D dataset (MC2 ≠ QF) that has been processed in F2 only. Suppose you would perform the following command sequence:

xf2 - to process the data in F2 only

1s si - to check the F1 size of the 2D data (hit **Enter** to accept it)

1s mc2 - to check status MC2 (≠ QF) → click ***Cancel***

f1projp - to store the F1 projection in ~TEMP and change to that dataset

1s si - to check the size of the resulting 1D dataset (hit **Enter** to accept it)

You will see that the size of the 1D data is only half the F1 size of the 2D data. The reason is that ***f1projp*** unshuffles the input data (file `2rr`). As such, ***f1projp*** behaves like the command ***rsc***. If you want to prevent the unshuffling of the input data (file `2rr`), you can use the following trick. Set the status parameter MC2 to QF before you run ***f1projp*** :

1s mc2 → click ***QF***

Then, the size of the 1D data equals the F1 size of the 2D data. In XWIN-NMR 3.1 and earlier, this trick is not needed because in these versions ***f1projp*** does not

unshuffle the input data for MC2 ≠ QF.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - processed data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`f2projn` - ascii file specifying the range of rows and the 1D data path
`f2projp` - ascii file specifying the range of rows and the 1D data path
`f1projn` - ascii file specifying the range of columns and the 1D data path
`f1projp` - ascii file specifying the range of columns and the 1D data path

<du>/data/<user>/nmr/<name>/<expno>/pdata/<*procno*>/

`1r` - 1D spectrum containing the projection
`auditp.txt` - processing audit trail

If the commands are used with less than three arguments, the files are stored in:

<du>/data/<user>/nmr/~*TEMP*/1/pdata/1/

USAGE IN AU PROGRAMS

F2PROJN(firstrow, lastrow, procno)

F2PROJP(firstrow, lastrow, procno)

F1PROJN(firstcol, lastcol, procno)

F1PROJP(firstcol, lastcol, procno)

For all these macros counts that if procno = -1, the projection is written to the dataset ~TEMP

SEE ALSO

f2disco, f1disco, f2sum, f1sum, proj, rhpp, rhnp, rvpp, rvnp

f2sum, f1sum

NAME

f2sum - calculates the sum of a range of rows (F2)
f1sum - calculates the sum of a range of columns (F1)

SYNTAX

f2sum [<firstrow> [<lastrow> [<procno> [y]]]]
f1sum [<firstcol> [<lastcol> [<procno> [y]]]]

DESCRIPTION

The command ***f2sum*** calculates the sum of all rows within a specified region. It takes four arguments:

firstrow - the first row of the region
lastrow - the last row of the region
procno - the processed data number of the resulting 1D dataset
chdata - change the display to the output 1D dataset or not (*y* or *n*, default *n*)

All rows between *firstrow* and *lastrow* are added to the total.

Here are some examples of the usage of ***f2sum*** :

f2sum
prompts for *firstrow* and *lastrow* and stores the sum under data name ~TEMP

f2sum <firstrow>
prompts for *lastrow* and stores the sum under data name ~TEMP

f2sum <firstrow> <lastrow>
stores the specified sum under data name ~TEMP

f2sum <firstrow> <lastrow> <procno>
stores the specified sum under the specified procno of the current data name

f2sum <firstrow> <lastrow> <procno> y
stores the specified sum under the specified procno of the current data name and changes the display to this procno

f1sum works like ***f2sum***, except that it calculates the sum of a range of columns instead of rows.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<*procno*>/

`1r`, `1i`- 1D spectrum containing the sum
`auditp.txt` - processing audit trail

If the commands are used with less than three arguments, the files are stored in:

<du>/data/<user>/nmr/~*TEMP*/1/pdata/1/

USAGE IN AU PROGRAMS

F2SUM(firstrow, lastrow, procno)

F1SUM(firstcol, lastcol, procno)

For both macros counts that if procno = -1, the sum is written to the dataset ~TEMP

SEE ALSO

f2disco, f1disco, f2projn, f2projp, f1projn, f1projp, proj, rhpp, rhnp, rvpp, rvnp

genser

NAME

genser - generate pseudo-raw 2D data

SYNTAX

genser [<expno> [<name> [<user> [<du>]]]] [y] [n]

DESCRIPTION

The command ***genser*** generates pseudo-raw data from processed 2D data. It is normally used in combination with ***xif2*** and ***xif1***. These commands perform an inverse Fourier transform, converting processed frequency domain data into processed time domain data. ***genser*** converts these processed time domain data into pseudo-raw time domain data and stores them under a new name or experiment number (expno).

Note that ***genser*** does not modify the data, but only stores them in a different format. The number of data points of the pseudo-raw data, is twice the size (SI) of the processed data they are created from. The acquisition status parameter TD (type ***dpa***) is set accordingly; TD = 2*SI. This count for both the F2 and F1 dimension.

genser takes six arguments and can be used as follows:

1. ***genser***
   prompts for an *expno* under which the output will be stored
2. ***genser <expno>***
   stores the output under the specified *expno*.
3. ***genser <expno> <name> y***
   stores the output under the specified *name* and *expno*. The last argument (*y*) causes ***genser*** to overwrite possibly existing data.
4. ***genser y <expno> <name>***
   same as 3.
5. ***genser <expno> <name> <user> <du> y n***
   The output will be stored under the specified *expno*, *name*, *user* and *disk unit*. The second last argument (*y*) causes ***genser*** to overwrite possibly existing. The last argument (*n*) causes XWIN-NMR to keep the display on

the input dataset rather than change it to the output dataset.

You can use any other combination of arguments as long as the datapath arguments are entered in the order <*expno*> <*name*> <*user*> <*du*>. The flags *y* and *n* can occur at an arbitrary position. The processed data number (*procno*) of the new dataset is always set to 1.

genser can be useful if you want to reprocess a 2D spectrum, for example with different processing parameters, but the raw data do not exist any more. An example of such a procedure is:

xif2 (if the data are Fourier transformed in F2)

xif1 (if the data are Fourier transformed in F1)

genser (to create the pseudo-raw data)

edp (to set the processing parameters)

xfb (to process the pseudo-raw data)

If the input data are processed but not Fourier transformed, you can skip the first two steps.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed time domain data

OUTPUT FILES

<*du*>/data/<*user*>/nmr/<*name*>/<*expno*>/

`ser` - pseudo-raw time domain data
`audita.txt` - acquisition audit trail

USAGE IN AU PROGRAMS

GENSER(expno)

SEE ALSO

xif2, xif1, genfid

proj

NAME

proj - calculate the full projections of a 2D dataset in both dimensions

DESCRIPTION

The command ***proj*** calculates the projections of a 2D dataset in both the F2 and F1 dimension. A projection is a 1D trace where every point has the highest intensity of all points of the corresponding orthogonal trace in the 2D spectrum. The calculation includes both positive and negative projections and is done over the entire spectrum.

The command ***proj*** is not used very often because all projections are already calculated during 2D processing, e.g. by the commands ***xfb***, ***xfbp***, ***abs2***, ***abs1*** etc. In fact, all 2D processing commands (re)calculate the projections to keep the processed 2D data and the projections consistent. If, however, the data have been manipulated by a third party program, the projections no longer match the processed data and can be recalculated with ***proj***.

Note that projections calculated by ***proj*** are part of the 2D dataset. They can be viewed from the 2D utilities menu by clicking the ***p*** or ***n*** button for positive and negative projections, respectively. Furthermore, projections can be stored a 1D datasets with the commands ***rhpp***, ***rhnp***, ***rvpp***, ***rvnp***.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`p2r1` - positive projection of the F1 dimension
`n2r1` - negative projection of the F1 dimension
`p2r2` - positive projection of the F2 dimension
`n2r2` - negative projection of the F2 dimension

On a magnitude or power spectrum, only the files `p2r1` and `p2r2` are created.

USAGE IN AU PROGRAMS

PROJ

SEE ALSO

f1sum, f2sum, f1disco, f2disco, f1projn, f1projp, f2projn, f2projp, rhpp, rhnp, rvpp, rvnp

ptilt

NAME

ptilt - tilt a 2D spectrum by shifting the data in the F2 dimension

DESCRIPTION

The command ***ptilt*** tilts a 2D spectrum about a user defined angle, by shifting the data points in the F2 dimension. It is typically used to correct possible magnet field drifts during long term 2D experiments. The tilt factor is determined by the F2 processing parameter ALPHA which can take a value between -2 and 2. Each row of the 2D matrix is shifted by *n* points where *n* is defined by:

n = tiltfactor * (nsrow/2 - row)

The variables in this equation are defined by:

tiltfactor = ALPHA*SI2 / SI1
nsrow = total number of rows
row = the row number

where SI2 and SI1 represent processing status parameter SI in F2 and F1, respectively.

For F2-ALPHA = 1 and F1-ALPHA = 1:

- the sequence ***ptilt*** - ***ptilt1*** rotates the spectrum by 90°
- the sequence ***ptilt1*** - ***ptilt*** rotates the spectrum by -90°.

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***alpha*** :

ALPHA - tilt factor

set by the initial processing command, e.g. ***xfb***, can be viewed with ***dpp*** :

SI - size of the processed data

F1 parameters

set by the initial processing command, e.g. ***xfb***, can be viewed with ***dpp*** :

SI - size of the processed data

OUTPUT PARAMETERS

F2 parameters

can be viewed with ***dpp*** or by typing ***2s tilt*** :

TILT - shows whether ***tilt***, ***ptilt*** or ***ptilt1*** was done (true or false)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

PTILT

SEE ALSO

ptilt1, tilt

ptilt1

NAME

ptilt1 - tilt a 2D spectrum by shifting the data in the F1 dimension

DESCRIPTION

The command ***ptilt1*** tilts a 2D spectrum about a user defined angle, by shifting the data points in the F1 dimension. The tilt factor is determined by the F1 processing parameter ALPHA which can take a value between -2 and 2. Each column of the 2D matrix is shifted by *n* points where *n* is defined by:

n = tiltfactor * (nscol/2 - row)

The variables in this equation are defined by:

tiltfactor = ALPHA*SI1/ SI2
nscol = total number of columns
col = the column number

where SI2 and SI1 represent processing status parameter SI in F2 and F1, respectively.

For F2-ALPHA = 1 and F1-ALPHA = 1:

- the sequence ***ptilt*** - ***ptilt1*** rotates the spectrum by 90°
- the sequence ***ptilt1*** - ***ptilt*** rotates the spectrum by -90°.

The command ***ptilt1*** is used in the AU program shear which can be viewed with the command ***edau shear.***

INPUT PARAMETERS

F1 parameters

set by the user with ***edp*** or by typing ***1 alpha*** :

ALPHA - tilt factor

set by the initial processing command, e.g. ***xfb***, can be viewed with ***dpp*** :

SI - size of the processed data

F2 parameters

set by the initial processing command, e.g. ***xfb***, can be viewed with ***dpp*** :

SI - size of the processed data

OUTPUT PARAMETERS

F2 parameters

can be viewed with ***dpp*** or by typing ***2s tilt*** :

TILT - shows whether ***tilt***, ***ptilt*** or ***ptilt1*** was done (true or false)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

PTILT1

SEE ALSO

ptilt, tilt

rev2, rev1

NAME

rev2 - reverse a 2D spectrum in the F2 dimension
rev1 - reverse a 2D spectrum in the F1 dimension

DESCRIPTION

The command ***rev2*** reverses the spectrum in the F2 dimension. This means, each row is mirrored about the central column.

The command ***rev1*** reverses the spectrum in the F1 dimension. This means, each column is mirrored about the central row.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

REV2

REV1

SEE ALSO

rv

rhpp, rhnp, rvpp, rvnp

NAME

rhpp - read horizontal (F2) positive projection
rhnp - read horizontal (F2) negative projection
rvpp - read vertical (F1) positive projection
rvnp - read vertical (F1) negative projection

SYNTAX

rhpp [<procno> [n]]

rhnp, rvpp and rvnp have the same syntax as rhpp

DESCRIPTION

The command ***rhpp*** reads the full positive projection of a 2D spectrum in the F2 dimension and stores it as a 1D dataset. A projection is a 1D trace where every point has the highest intensity of all points of the corresponding orthogonal trace in the 2D spectrum. Normally, the projections already exist. They are calculated, implicitly, by any processing command, e.g. by ***xfb*** or, explicitly, by the command ***proj***. If, however, the projections do not exist, ***rhpp*** automatically executes ***proj*** to create them.

rhpp only takes the projection of the the first quadrant data (file `2rr`) and stores it as real 1D data (file `1r`).

rhpp takes two arguments:

procno - the processed data number of the resulting 1D dataset
chdata - change the display to the output 1D dataset or not (*y* or *n*, default *y*)

Here are some examples of its usage:

rhpp
stores the projection under data name ~TEMP

rhpp <procno>
stores the projection under the specified *procno* of the current data name

rhpp <procno> n
stores the projection under the specified *procno* but does not change the display to that procno

The other three command work like ***rhpp*** except that:

rhnp calculates the full negative projection in F2
rvpp calculates the full positive projection in F1
rvnp calculates the full negative projection in F1

A special case is the command ***rvpp*** or ***rvnp*** on a hypercomplex 2D dataset (MC2 ≠ QF) that has been processed in F2 only. Suppose you would perform the following command sequence:

xf2 - to process the data in F2 only

ls si - to check the F1 size of the 2D data (hit **Enter** to accept it)

ls mc2 - to check status MC2 (≠ QF) → click ***Cancel***

rvpp - to store the F1 projection in ~TEMP and change to that dataset

ls si - to check the size of the resulting 1D dataset (hit **Enter** to accept it)

You will see that the size of the 1D data is only half the F1 size of the 2D data. The reason is that ***rvpp*** unshuffles the input data (file `2rr`). As such, ***rvpp*** behaves like the command ***rsc***. If you want to prevent the unshuffling of the input data (file `2rr`), you can use the following trick. Set the status parameter MC2 to QF before you run ***rvpp*** :

ls mc2 → click ***QF***

Then, the size of the 1D data equals the F1 size of the 2D data. In XWIN-NMR 3.1 and earlier, this trick is not needed because in these versions ***rvpp*** does not unshuffle the input data for MC2 ≠ QF.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`p2r1` - positive horizontal (F2) projection copied by ***rhpp***
`n2r1` - negative horizontal (F2) projection copied by ***rhnp***
`p2r2` - positive vertical (F1) projection copied by ***rvpp***
`n2r2` - negative vertical (F1) projection copied by ***rvnp***

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<*procno*>/

`1r` - 1D spectrum containing the projection

`auditp.txt` - processing audit trail

If the commands are used without arguments, the files are stored in:

<du>/data/<user>/nmr/~*TEMP*/1/pdata/1/

USAGE IN AU PROGRAMS

RHPP(procno)

RHNP(procno)

RVPP(procno)

RVNP(procno)

For all these macros counts that if procno = -1, the projection is written to the dataset ~TEMP

SEE ALSO

f2sum, f1sum, f2disco, f1disco, f2projn, f2projp, f1projn, f1projp, proj

rsc

NAME

rsc - read a column from a 2D spectrum and store it as a 1D spectrum

SYNTAX

rsc [<column> [<procno>] [n]]

DESCRIPTION

The command ***rsc*** reads a column from a 2D spectrum and stores it as a 1D spectrum.The column must be specified as a number between 1 and F2-SI. The latter is the F2 processing status parameter SI that can be viewed with ***ls si***.

rsc is normally entered on the 2D dataset. It then takes three arguments and can be used as follows:

rsc
prompts for the column number and stores it under data name ~TEMP

rsc <column>
stores the specified column under data name ~TEMP

rsc <column> <procno>
stores the specified column under the current data name, the current expno and the specified procno. It changes the display to the output 1D data.

rsc <column> <procno> n
stores the specified column under the current data name, the current expno and the specified procno. It does not change the display to the output 1D data.

After ***rsc*** has read a column and the display has changed to the destination 1D dataset, a subsequent ***rsc*** command can be entered on this 1D dataset. This takes two arguments and can be used as follows:

rsc
prompts for the column number and reads this column from the 2D dataset from which the current 1D dataset was extracted

rsc <column>
reads the specified column from the 2D dataset from which the current 1D dataset was extracted

rsc <column> <procno>
reads the specified column from the 2D dataset that resides under the current data name [1], the current expno and the specified procno. Specifying the procno allows you to read a column from a 2D dataset other than the one from which the current 1D dataset was extracted. Furthermore, the AU macro RSC requires two arguments, no matter if it is used on a 1D or on a 2D dataset.

A special case is a 2D dataset that has been Fourier transformed in F2 but not in F1. ***rsc*** then stores 1D processed data that are in the time domain rather than the frequency domain. Below are five different examples of this case.

Example 1

A 2D dataset is Fourier transformed in F2, column 17 (time domain) is extracted and stored under the same name and expno, in procno 2. The resulting 1D dataset is Fourier transformed.

On the 2D dataset, enter the following commands:

xf2 - to Fourier transform in F2 only

rsc 17 2 - to read column 17 to procno 2 and switch to that dataset

ft - to Fourier transform the resulting 1D data according to FnMODE

Explanation: the 1D data shares the expno, and the acquisition parameters in it, with the source 2D dataset. 1D processing commands automatically recognize that this 1D dataset is a column from a 2D dataset. The command ***ft*** interprets the F1 acquisition parameter FnMODE to determine the Fourier transform mode.

Example 2

A 2D dataset with F1 acquisition mode *States* is Fourier transformed in F2. Column 17 (time domain) is extracted and stored under data name ~TEMP. The resulting 1D dataset is Fourier transformed.

On the 2D dataset, enter the following commands:

ls fnmode - check the FnMODE value (*States*) → click ***Cancel***

xf2 - to Fourier transform in F2 only

1. However, if the current data name is ~TEMP, ***rsc <column> <procno>*** reads from the specified procno in the dataset from which the current 1D dataset was extracted.

ls mc2 - check the MC2 value (*States*) → click ***Cancel***

rsc 17 - read column 17 to ~TEMP and switch to that dataset

ls aq_mod - check the AQ_mod value (*qsim*) → click ***Cancel***

ft - Fourier transform the resulting 1D data according to AQ_mod

Explanation: the source 2D and the destination 1D have a separate a set of acquisition parameters. ***rsc*** reads the F1 status parameter MC2 of the 2D data and translates that to the corresponding AQ_mod of the 1D data. 1D processing commands recognizes this 1D dataset as regular 1D data. This means, for example, that ***ft*** interprets the AQ_mod to determine the Fourier transform mode.

Example 3

A 2D dataset with an F1 acquisition mode States-TPPI is Fourier transformed in F2. Column 17 (time domain) is extracted and stored under data name ~TEMP. The resulting 1D dataset is Fourier transformed.

On the 2D dataset, enter the following commands:

ls fnmode - check the FnMODE value (*States-TPPI*) → click ***Cancel***

xf2 - to Fourier transform in F2 only

ls mc2 - check the MC2 value (*States-TPPI*) → click ***Cancel***

rsc 17 - to read column 17 to ~TEMP and switch to that dataset

ft_mod - check the FT_mod value (*fsc*) → click ***Cancel***

trfp - to Fourier transform the resulting 1D data according to FT_mod

Explanation: the source 2D and the destination 1D have a separate a set of acquisition parameters. Since there is no value for AQ_mod that corresponds to States-TPPI, ***rsc*** sets the processing parameter FT_mod instead of the acquisition status parameter AQ_mod. As such, the resulting 1D dataset can only be Fourier transformed correctly with ***trfp***.

Example 4

A 2D dataset with an F1 acquisition mode QF is Fourier transformed in F2. Column 17 (time domain) is extracted and stored under data name ~TEMP. From the 2D dataset, enter the following commands:

1s fnmode - check the FnMODE value (*QF*) → click ***Cancel***

xf2 - to Fourier transform in F2 only

1s mc2 - check the MC2 value (*QF*) → click ***Cancel***

rsc 17 - to read column 17 to ~TEMP and switch to that dataset

1s si - check the size of the 1D dataset → hit **Enter**

Explanation: for FnMODE = QF the 2D storage mode is different than for other values (see the description of ***xfb***). As such, the size of the resulting 1D data is twice as large as for other values of FnMODE. If 2D imaginary data (file `2ii`) exist, 1D imaginary (file `1i`) are created. Only in that case, the 1D data can be Fourier transformed.

Example 5

From a 3D dataset, a plane is extracted and, from this plane a column is extracted.

On the 3D dataset, enter the following commands:

xf2 s13 48 2 - to read the F3-F1 plane 48 to procno 2

rsc 19 3 - to read, from plane 48, column 19 to procno 3

ft : to Fourier transform the resulting 1D data according to FnMODE

Explanation: the 3D, 2D and 1D dataset are stored in three different procnos all under the same expno, i.e. they share the same acquisition parameters. 1D processing commands automatically recognize that the 1D dataset is a column from an F3-F1 plane that was extracted from a 3D dataset. As such, ***ft*** interprets the F1 parameter FnMODE to determine the Fourier transform mode. Note that F1 is the third dimension of the 3D dataset. The parameter handling, however, is transparent to the user.

On a 1D data that was extracted from a 2D, you can enter ***edc2 used_from*** to view the row or column number (parameter ROW_COL) and the datapath of the source 2D data. On any 1D dataset, you can enter ***edc2 used_from*** to specify the 2D dataset from which you want to read a row or column.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - 2D processed data

OUTPUT FILES

If ***rsc*** is entered with less than 2 arguments, on a 2D dataset:

<du>/data/<user>/nmr/~*TEMP*/1/pdata/1/

`1r`, `1i` - 1D spectrum
`used_from` - datapath of the source 2D data and the column no.
`auditp.txt` - processing audit trail

If ***rsc*** is entered with 2 or more arguments, on a 2D dataset:

<du>/data/<user>/nmr/<name>/<expno>/pdata/<*procno*>/

`1r`, `1i` - 1D spectrum
`used_from` - datapath of the source 2D data and the column no.
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

RSC(column, procno)

If procno = -1, the column is written to the dataset ~TEMP

SEE ALSO

rsr, wsr, wsc, rser, rser2d, wser

rser

NAME

rser - read a row from 2D or 3D raw data and store it as a 1D FID

SYNTAX

rser [<row> [<expno> [<procno>]] [n]]

DESCRIPTION

The command ***rser*** reads a row from 2D or 3D raw data (a series of FIDs) and stores it as a 1D dataset. For 2D, the row must be specified as a number between 1 and F1-TD. The latter is the F1 acquisition status parameter TD that can be viewed with ***ls td***.

rser is normally entered on the 2D dataset. It then takes four arguments and can be used as follows:

rser
prompts for the row number and stores it under data name ~TEMP

rser <row>
stores the specified row under data name ~TEMP

rser <row> <expno>
stores the specified row under the current data name and the specified expno and then changes the display to this expno

rser <row> <expno> n
stores the specified row under the current data name and the specified expno but does not change the display to this expno

rser <row> <expno> <procno> n
stores the specified row under the current data name, specified expno and the specified procno. It does not change the display the output expno. Note that although the procno is specified, the extracted row is still stored in the <expno> directory. The <procno> directory merely contains processing parameter files.

After ***rser*** has read a row and the display has changed to the destination 1D dataset, a subsequent ***rser*** command can be entered on this 1D dataset. This

takes three arguments and can be used as follows:

rser
prompts for the row number and reads this row from the 2D dataset from which the current 1D dataset was extracted

rser <row>
reads the specified row from the 2D dataset from which the current 1D dataset was extracted

rser <row> <expno>
reads the specified row from the 2D dataset that resides under the current data name [1], the specified expno and procno 1.

rser <row> <expno> <procno>
reads the specified row from the 2D dataset that resides under the current data name [1], the specified expno and the specified procno.

Note that on 3D data, ***rser*** does not distinguish between the F2 and F1 dimension and treats the 3D dataset as a large 2D dataset. This implies that the row number must lie between 1 and (F2-TD) * (F1-TD).

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`ser` - 2D raw data

OUTPUT FILES

If ***rser*** is entered with 2 or more arguments, on a 2D dataset:

<du>/data/<user>/nmr/<name>/<*expno*>/

`fid` - 1D FID
`audita.txt` - acquisition audit trail

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/1/

`used_from` - datapath of the source 2D data and the row no.

If ***rser*** is entered with less than 2 arguments, on a 2D dataset:

1. However, if the current data name is ~TEMP, the input dataset is the one from which the current 1D dataset was extracted except for the specified expno (procno).

<du>/data/<user>/nmr/~*TEMP*/1/

`fid` - 1D FID

<du>/data/<user>/nmr/~*TEMP*/1/pdata/1

`used_from` - datapath of the source 2D data and the row no.

USAGE IN AU PROGRAMS

RSER(row, expno, procno)

If expno = -1, the row is written to the dataset ~TEMP

SEE ALSO

wser, wserp, rser2d, rsr, rsc, wsr, wsc

rser2d

NAME

rser2d - read a plane from 3D raw data and store it as a 2D pseudo-raw data

SYNTAX

rser2d [<direction> [<plane> [<expno>]] [n]]

DESCRIPTION

The command ***rser2d*** reads a plane from 3D raw data (a series of FIDs) and stores it as a pseudo 2D dataset. The plane must be specified as a number greater than or equal to 1.

rser2d only exists in XWIN-NMR 3.1 and newer.

The command ***rser2d*** takes 7 arguments and can be used as follows:

rser2d
prompts for the plane direction, the plane number and the output expno and then stores this plane under that expno

rser2d s23
prompts for the F2F3-plane number and the output expno and then stores this plane under that expno

rser2d s13 <plane>
prompts for the output expno and then stores the specified F1F3-plane under that expno

rser2d s23 <plane> <expno>
stores the specified F2F3-plane under the specified expno

rser2d s23 <plane> <expno> <name> <user> <disk> n
stores the specified F2F3-plane under the specified dataset but does not change the display to this expno

In contrast to ***rser***, ***rser2d*** can only be entered on the source dataset, not on the destination dataset.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`ser` - 3D raw data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/

`ser` - 2D pseudo raw data
`audita.txt` - acquisition audit trail

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/1/

`used_from` - datapath of the source 3D data and the plane number

USAGE IN AU PROGRAMS

RSER2D (direction, plane, expno, name, user, disk)

SEE ALSO

rser, wser, wserp, rsr, rsc, wsr, wsc

rsr

NAME

rsr - read a row from a 2D spectrum and store it as a 1D spectrum

SYNTAX

rsr [<row> [<procno>] [n]]

DESCRIPTION

The command ***rsr*** reads a row from a 2D spectrum and stores it as a 1D spectrum. The row must be specified as a number between 1 and F1-SI. The latter is the F1 processing status parameter SI that can be viewed with ***1s si***.

rsr is normally entered on the 2D dataset. It then takes three arguments and can be used as follows:

rsr
prompts for the row number and stores it under data name ~TEMP

rsr <row>
stores the specified row under data name ~TEMP

rsr <row> <procno>
stores the specified row under the current data name, the current expno and the specified procno. It changes the display to the output 1D data.

rsr <row> <procno> n
stores the specified row under the current data name, the current expno and the specified procno. It does not change the display to the output 1D data.

After ***rsr*** has read a row and the display has changed to the destination 1D dataset, a subsequent ***rsr*** command can be entered on this 1D dataset. This takes two arguments and can be used as follows:

rsr
prompts for the row number and reads this row from the 2D dataset from which the current 1D dataset was extracted

rsr <row>
reads the specified row from the 2D dataset from which the current 1D dataset was extracted

rsr <row> <procno>
reads the specified row from the 2D dataset that resides under the current data name [1], the current expno and the specified procno. Specifying the procno allows you to read a row from a 2D dataset other than the one from which the current 1D dataset was extracted. Furthermore, the AU macro RSR requires two arguments, no matter if it is used on a 1D or on a 2D dataset.

On a 1D data that was extracted from a 2D, you can enter ***edc2 used_from*** to view the row or column number (parameter ROW_COL) and the datapath of the source 2D data. On any 1D dataset, you can enter ***edc2 used_from*** to specify the 2D dataset from which you want to read a row or column.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - 2D processed data

OUTPUT FILES

If ***rsr*** is entered with less than 2 arguments, on a 2D dataset:

<du>/data/<user>/nmr/~*TEMP*/1/pdata/1/

`1r`, `1i` - 1D spectrum
`used_from` - datapath of the source 2D data and the row no.
`auditp.txt` - processing audit trail

If ***rsr*** is entered with 2 or more arguments, on 2D a dataset:

<du>/data/<user>/nmr/<name>/<expno>/pdata/<*procno*>/

`1r`, `1i` - 1D spectrum
`used_from` - datapath of the source 2D data and the row no.
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

RSR(row, procno)

If procno = -1, the row is written to the dataset ~TEMP

1. However, if the current data name is ~TEMP, ***rsr <row> <procno>*** reads from the specified procno in the dataset from which the current 1D dataset was extracted.

SEE ALSO

rsc, wsr, wsc, rser, rser2d, wser

sub1d1, sub1

NAME

sub1d1 - subtracts a 1D spectrum from each column of a 2D spectrum
sub1 - as sub1d1, but the subtraction is dependent on the sign of the data points

DESCRIPTION

The command ***sub1d1*** subtracts a 1D spectrum from each column of the current 2D spectrum. Before you apply this command, you must first specify the 1D spectrum as the second dataset with the command ***edc2***.

sub1 works like ***sub1d1***, except that it first compares the intensity of each data point of the 1D spectrum with the intensity of the corresponding data point in the 2D spectrum. If they have opposite signs, no subtraction is done and the 2D data point remains unchanged. If they have the same sign and the 1D data point is smaller than the 2D data point, the subtraction is done. If the 1D data point is greater than the 2D data point, the latter is set to zero. As such, the sign of the 2D data points always remains the same.

sub1d1 and ***sub1*** only work on the real data. After using them, the imaginary data no longer match the real data and cannot be used for phase correction.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

<du2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

`1r` - real processed 1D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

SUB1D1

SUB1

SEE ALSO

sub1d2, sub2

sub1d2, sub2

NAME

sub1d2 - subtracts a 1D spectrum from each row of a 2D spectrum
sub2 - as sub1d2, but the subtraction is dependent on the sign of the data points

DESCRIPTION

The command ***sub1d2*** subtracts a 1D spectrum from each row of the current 2D spectrum. Before you apply this command, you must specify the 1D spectrum as the second dataset with the command ***edc2***.

sub2 works like ***sub1d2***, except that the subtraction is dependent on the relative intensities of the 2D and 1D data points. ***sub2*** compares the intensity of each data point of the 1D spectrum with the intensity of the corresponding data point in the 2D spectrum. If they have opposite signs, no subtraction is done and the 2D data point remains unchanged. If they have the same sign and the 1D data point is smaller than the 2D data point, the subtraction is done. If the 1D data point is greater than the 2D data point, the latter is set to zero. As such, the sign of the 2D data points always remains the same.

sub2 and ***sub1d2*** only work on the real data. After using them, the imaginary data no longer match the real data and cannot be used for phase correction.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

<du2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

`1r` - real processed 1D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

SUB1D2

SUB2

SEE ALSO

sub1d1, sub1

sym

NAME

sym - symmetrize a 2D spectrum about the diagonal

DESCRIPTION

The command ***sym*** symmetrizes a 2D spectrum about a diagonal from the lower left corner (data point 1,1) to the upper right corner (data point F2-SI, F1-SI). It compares each data point with the corresponding data point on the other side of the diagonal and determines which one has the lowest (most negative) intensity. Then both data points are set to that intensity. Table 4.1 shows the intensities of four pairs of data points before and after ***sym***:

before sym	after sym
-370000, 12000	-370000, -370000
1000, -700	-700, -700
18000, 6000	6000, 6000
-13000, -8000	-13000, -13000

Table 4.1

sym only works on the real data. After using it, the imaginary data no longer match the real data and cannot be used for phase correction.

sym is typically used on magnitude cosy spectra.

OUTPUT PARAMETERS

F2 parameters

can be viewed with ***dpp*** or by typing ***2s symm*** :

SYMM - type of symmetrization (*no*, *sym*, *syma* or *symj*) done

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

SYM

SEE ALSO

syma, symj

syma

NAME

syma - symmetrize a 2D spectrum about the diagonal, leaving the sign the same

DESCRIPTION

The command ***syma*** symmetrizes a spectrum about the diagonal from the lower left corner (data point 1,1) to the upper right corner (data point F2-SI, F1-SI). As opposed to ***sym***, it compares each data point with the corresponding data point on the other side of the diagonal and determines which one has the lowest absolute intensity. Then both data points are set to that intensity while each point keeps its original sign. Table 4.2 shows the intensities of four pairs of data points before and after ***syma***:

before sym	after sym
-370000, 12000	-12000, 12000
1000, -700	700, -700
18000, 6000	6000, 6000
-13000, -8000	-8000, -8000

Table 4.2

syma only works on the real data. After using it, the imaginary data no longer match the real data and cannot be used for phase correction.

syma is typically used on phase sensitive cosy spectra.

OUTPUT PARAMETERS

can be viewed with ***dpp*** or by typing ***2s symm*** :

SYMM - type of symmetrization (*no*, *sym*, *syma* or *symj*) done

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

SYMA

SEE ALSO

sym, symj

symj

NAME

symj - symmetrize a 2D spectrum about the horizontal line through the middle

DESCRIPTION

The command ***symj*** symmetrizes a 2D spectrum about a horizontal line through the middle. It is similar to ***sym***, i.e. it compares each data point with the corresponding data point on the other side of the horizontal line and determines which one has the lowest (most negative) intensity. Then both data points are set to that intensity. Table 4.3 shows the intensities of 5 pairs of data points before and after ***symj***:

before symj	after symj
-370000, 12000	-370000, -370000
1000, -700	-700, -700
18000, 6000	6000, 6000
-13000, -8000	-13000, -13000
-8000, -25000	-25000, -25000

Table 4.3

symj only works on the real data. After using it, the imaginary data no longer match the real data and cannot be used for phase correction.

symj is typically used on J-resolved spectra which have been tilted with the command ***tilt***.

OUTPUT PARAMETERS

F2 parameters

can be viewed with ***dpp*** or by typing ***2s symm*** :

SYMM - type of symmetrization (*no*, *sym*, *syma* or *symj*) done

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

SYMJ

SEE ALSO

sym, syma, tilt

tilt

NAME

tilt - tilt a 2D spectrum

DESCRIPTION

The command ***tilt*** tilts 2D spectrum. Each row of the 2D matrix is shifted by the value:

n = tiltfactor * (nsrow/2 - row)

The variables in this equation are defined as:

tiltfactor = (SW_p1/SI1) / (SW_p2/SI2)
nsrow = total number of rows
row = the row number

where SW_p1, SI1, SW_p2 and SI2 represent the processing status parameters SW_p and SI in F1 and F2, respectively.

The upper half of the spectrum is shifted to the right, the lower half to the left. Furthermore, this is a circular shift, i.e. the data points which are cut off at the right edge of the spectrum are appended at the left edge and vice versa.

INPUT PARAMETERS

F2 and F1 parameters

set by initial processing command, e.g. ***xfb***, can be viewed with ***dpp***:

SW_p - spectral width of the processed data
SI - size of the processed data

OUTPUT PARAMETERS

F2 parameters

can be viewed with ***dpp*** or by typing ***2s tilt*** :

TILT - shows whether ***tilt***, ***ptilt*** or ***ptilt1*** was done (true or false)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TILT

SEE ALSO

ptilt, ptilt1, symj

WSC

NAME

wsc - write spectrum column; replace a 2D column by a 1D spectrum

SYNTAX

wsc [<column> [<procno> [<expno> [<name> [<user> [<disk>]]]]]]

DESCRIPTION

The command ***wsc*** replaces one column of 2D processed data by 1D processed data. It is normally used in combination with ***rsc*** in the following way:

1. run ***rsc*** to extract column *x* from a 2D spectrum
2. manipulate the resulting 1D data with 1D processing commands
3. run ***wsc*** to replace column *x* of the 2D data with the manipulated 1D data

wsc can be entered on the source 1D dataset or on the destination 2D dataset.

Examples of the usage of ***wsc*** on the source 1D dataset:

wsc
prompts for the column of the destination 2D data which must be replaced by the current 1D data. The 2D dataset is the one from which the 1D dataset was extracted.

wsc <column>
the specified column of the destination 2D data is replaced by the current 1D data. The 2D dataset is the one from which the current 1D dataset was extracted.

wsc <column> <procno>
the specified column of the destination 2D data is replaced by the current 1D data. The 2D dataset must reside under the current data name [1], the current expno and the specified procno.

Examples of usage of ***wsc*** on the destination 2D dataset:

1. However, if the current data name is ~TEMP, ***wsc <column> <procno>*** writes to the specified procno in the dataset from which the current 1D dataset was extracted.

wsc <column>
the specified column of the current 2D processed data is replaced. The source 1D data must reside under the data name ~TEMP

wsc <column> <procno>
the specified column of the current 2D processed data is replaced. The source 1D data must reside under the current data name, the current expno and the specified procno.

Although ***wsc*** is normally used as described above, it allows you to specify a full dataset path in the following way:

wsc <column> <procno> <expno> <name> <user> <disk>

When entered on a 1D dataset, the arguments specify the destination 2D dataset. When entered on a 2D dataset, the arguments specify the source 1D dataset. If only certain parts of the destination 2D datapath are specified, e.g. the expno and name, the remaining parts are the same as in the current 1D datapath.In AU programs, ***wsc*** must always have 6 arguments (see USAGE IN AU PROGAMS below) [1].

On a 1D data that was extracted from a 2D, you can enter ***edc2 used_from*** to view the row or column number (parameter ROW_COL) and the datapath of the source 2D data. On any 1D dataset, you can enter ***edc2 used_from*** to specify the 2D dataset to which you want to write a row or column.

INPUT FILES

<du>/data/<user>/nmr/~TEMP/1/pdata/1

`1r`, `1i` - 1D processed data
`used_from` - datapath of the 2D data (input of ***wsc*** on a 1D dataset)

or

<du>/data/<user>/nmr/<name>/<expno>/pdata/<*procno*>

`1r`, `1i` - 1D processed data
`used_from` - datapath of the 2D data (input of ***wsc*** on a 1D dataset)

1. In XWIN-NMR 3.0 and older, the expno could not be specified. It was taken from the used_from file (see ***edc2 used_from***)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<*procno*>

`2rr`, `2ri` - processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

WSC(column, procno, expno, name, user, disk)

SEE ALSO

rsc, wsr, rsr, wser, wserp, rser, rser2d

wser

NAME

wser - replace a row of 2D raw data by 1D raw data

SYNTAX

wser [<row> [<expno> [<procno> [<name> [<user> [<du>]]]]]]

DESCRIPTION

The command ***wser*** replaces one row of 2D raw data by 1D raw data. It can be entered on the source 1D dataset or on the destination 2D dataset and takes up to six arguments.

Examples of the usage of ***wser*** on the source 1D dataset:

wser
prompts for the row of the 2D raw data which must be replaced by the current 1D FID. The destination 2D dataset is the one from which the current 1D dataset was extracted.

wser <row>
the specified row of the 2D raw data is replaced by the current 1D FID. The destination 2D dataset is the one from which the current 1D dataset was extracted.

wser <row> <expno>

the specified row of the 2D raw data is replaced by the current 1D FID. The 2D dataset must reside under the current data name, the specified expno and procno 1.

wser <row> <expno> <procno> <name> <user> <du>

the specified row of the 2D raw data is replaced by the current 1D FID. The 2D dataset must reside under the pathname specified by second to sixth argument.

Note that if the destination 2D datapath is not specified, it is the 2D dataset from which the current 1D dataset was extracted (see ***edc2 used_from***). However, if certain parts of the 2D datapath are specified, e.g. the expno, the remaining parts are the same as in the current 1D datapath [1].

Examples of usage of ***wser*** on the destination 2D dataset:

wser <row> <expno>
the specified row of the current 2D raw data is replaced. The source 1D dataset must reside under the current data name, specified expno and procno 1.

wser <row> <expno> <procno> <name> <user> <du>

the specified row of the current 2D raw data is replaced. The source 1D dataset must reside under the pathname specified by second to sixth argument.

Note that the display always remains on the current dataset, no matter if ***wser*** is entered from the source 1D or from the destination 2D dataset.

On a 1D dataset which was extracted from a 2D, you can use the command ***edc2 used_from*** to view the datapath of the source 2D data.

INPUT FILES

<du>/data/<user>/nmr/~TEMP/1/

`fid` - 1D raw data

<du>/data/<user>/nmr/~TEMP/1/pdata/1

`used_from` - datapath of the 2D data (input of ***wser*** on a 1D dataset)

or

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - 1D raw data

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`used_from` - datapath of the 2D data (input of ***wser*** on a 1D dataset)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/

`ser` - raw 2D data
`audita.txt` - acquisition audit trail

1. However, if the current data name is ~TEMP, the destination 2D dataset is the one from which the current 1D dataset was extracted

USAGE IN AU PROGRAMS

WSER(row, name, expno, procno, disk, user)

Note that the order of the arguments in AU programs is different from the order on the command line.

SEE ALSO

rser, wserp, wsr, wsc, rsr, rsc, rser2d

wserp

NAME

wserp - replace a row of 2D raw data by 1D processed data

SYNTAX

wserp [<row> [<expno> [<procno> [<name> [<user> [<du>]]]]]]

DESCRIPTION

The command ***wserp*** replaces one row of 2D raw data by processed 1D data. It can be entered on the source 1D dataset or on the destination 2D dataset and takes up to six arguments.

Examples of the usage of ***wserp*** on the source 1D dataset:

wserp
prompts for the row of the 2D raw data to be replaced by the current 1D processed data. The destination 2D dataset is the one from which the current 1D dataset was extracted.

wserp <row>
the specified row of the 2D raw data is replaced by the current 1D processed data. The 2D dataset is the one from which the current 1D dataset was extracted.

wserp <row> <expno> <procno>
the specified row of the 2D raw data is replaced by the current 1D processed data. The 2D dataset must reside under the current data name, the specified expno and the specified procno.

wserp <row> <expno> <procno> <name> <user> <du>
the specified row of the 2D raw data is replaced by the current 1D processed data. The 2D dataset must reside under the pathname specified by second to sixth argument.

Note that if the destination 2D datapath is not specified, it is the 2D dataset from which the current 1D dataset was extracted (see ***edc2 used_from***). However, if certain parts of the 2D datapath are specified, e.g. the expno and procno, the remaining parts are the same as in the current 1D datapath [1].

Examples of its usage on the destination 2D dataset are:

wserp <row> <expno> <procno>
the specified row of the current 2D raw data is replaced by processed 1D data which must reside under the current data name, the specified expno and the specified procno.

wserp <row> <expno> <procno> <name> <user> <du>
the specified row of the current 2D raw data is replaced by processed 1D data. The 1D dataset must reside under the pathname specified by second to sixth argument.

On a 1D dataset which was extracted from a 2D, you can use the command ***edc2 used_from*** to view the data path of the source 2D data.

INPUT FILES

<du>/data/<user>/nmr/~TEMP/1/pdata/1/

`1r`, `1i` - 1D processed data (real, imaginary)
`used_from` - datapath of the 2D data (input of ***wserp*** on a 1D dataset)

or

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - 1D processed data (real, imaginary)
`used_from` - datapath of the 2D data (input of ***wserp*** on a 1D dataset)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/

`ser` - raw 2D data
`audita.txt` - acquisition audit trail

USAGE IN AU PROGRAMS

WSERP(row, name, expno, procno, disk, user)

Note that the order of the arguments in AU programs is different from the order on the command line.

---

1. However, if the current data name is ~TEMP, the destination 2D dataset is the one from which the current 1D dataset was extracted

SEE ALSO

rser, rser2d, wser, wsr, wsc, rsr, rsc

wsr

NAME

wsr - write spectrum row; replace a row of a 2D spectrum by a 1D spectrum

SYNTAX

wsr [<row> [<procno> [<expno> [<name> [<user> [<disk>]]]]]]

DESCRIPTION

The command ***wsr*** replaces one row of 2D processed data by 1D processed data. It is normally used in combination with ***rsr*** in the following way:

- run ***rsr*** to extract row *x* from a 2D spectrum
- manipulate the resulting 1D data with 1D processing commands
- run ***wsr*** to replace row *x* of the 2D data with the manipulated 1D data

wsr can be entered on the source 1D dataset or on the destination 2D dataset.

Examples of the usage of ***wsr*** on the source 1D dataset:

wsr
prompts for the row of the destination 2D data which must be replaced by the current 1D data. The 2D dataset is the one from which the current 1D dataset was extracted.

wsr <row>
the specified row of the destination 2D data is replaced by the current 1D data. The 2D dataset is the one from which the current 1D dataset was extracted.

wsr <row> <procno>
the specified row of the destination 2D data is replaced by the current 1D data. The 2D dataset must reside under the current data name [1], the current expno and the specified procno.

Examples of usage of ***wsr*** on the destination 2D dataset:

wsr <row>
the specified row of the current 2D processed data is replaced. The source 1D

1. However, if the current data name is ~TEMP, ***wsr <row> <procno>*** writes to the specified procno in the dataset from which the current 1D dataset was extracted.

data must reside under the data name ~TEMP.

wsr <row> <procno>
the specified row of the current 2D processed data is replaced. The source 1D data must reside under the current data name, the current expno and the specified procno.

Although ***wsr*** is normally used as described above, it allows you to specify a full dataset path in the following way:

wsr <row> <procno> <expno> <name> <user> <disk>

When entered on a 1D dataset, the arguments specify the destination 2D dataset. When entered on a 2D dataset, the arguments specify the source 1D dataset. If only certain parts of the destination 2D datapath are specified, e.g. the expno and name, the remaining parts are the same as in the current 1D datapath. In AU programs, ***wsr*** must always have 6 arguments (see USAGE IN AU PROGAMS below) [1].

On a 1D data that was extracted from a 2D, you can enter ***edc2 used_from*** to view the row or column number (parameter ROW_COL) and the datapath of the source 2D data. On any 1D dataset, you can enter ***edc2 used_from*** to specify the 2D dataset to which you want to write a row or column.

INPUT FILES

<du>/data/<user>/nmr/~TEMP/1/pdata/1

`1r`, `1i` - 1D processed data
`used_from` - datapath of the 2D data (input of ***wsr*** on a 1D dataset)

or

<du>/data/<user>/nmr/<name>/<expno>/pdata/<*procno*>

`1r`, `1i` - 1D processed data
`used_from` - datapath of the 2D data (input of ***wsr*** on a 1D dataset)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<*procno*>

1. In XWIN-NMR 3.0 and older, the expno could not be specified. It was taken from the used_from file (see ***edc2 used_from***)

`2rr`, `2ir` - processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

WSR(row, procno, expno, name, user, disk)

SEE ALSO

wsc, rsr, rsc, wser, wserp, rser, rser2d

xf1

NAME

xf1 - process data in the F1 dimension

DESCRIPTION

The command ***xf1*** processes a 2D dataset in the F1 dimension. This involves a Fourier transform which transforms time domain data (FID) into frequency domain data (spectrum). Depending on the F1 processing parameters BC_mod, WDW, ME_mod and PH_mod, ***xf1*** also performs baseline correction, window multiplication, linear prediction and phase correction, respectively. These steps are described in detail for the command ***xfb***.

Normally, 2D data are processed with the command ***xfb*** which performs a Fourier transform in both dimensions, F2 and F1. In some cases, however, it is useful to process the data in two separate steps using the sequence ***xf2*** - ***xf1***, for example to view the data after processing them in F2 only.

If you run ***xf1*** without running ***xf2*** first, a warning that the F2 transform has not been done will appear. When the command has finished the data are in the time domain in F2 and in the frequency domain in F1. The opposite case, however, is more usual, i.e. data which have only been processed with ***xf2***.

xf1 takes the same options as ***xfb***.

The F1 Fourier transform mode and data storage mode depends on the F1 acquisition mode (see INPUT PARAMETERS below and the description of ***xfb***).

INPUT PARAMETERS

F2 and F1 parameters

set by ***xf2***, can be viewed with ***dpp*** or by typing ***2s si***, ***1s si*** etc.:

SI - size of the processed data
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
TDeff - number of raw data points to be used for processing
TDoff - first point of the FID used for processing (default 0)

If ***xf2*** has not been done, ***xf1*** uses the ***edp*** parameters set by the user.

F1 parameters

set by the user with ***edp*** or by typing ***bc_mod*** etc.

BC_mod - FID baseline correction mode
BCFW - filter width for BC_mod = sfil or qfil
COROFFS - correction offset for BC_mod = spol/qpol or sfil/qfil
ME_mod - FID linear prediction mode
NCOEF - number of linear prediction coefficients
LPBIN - number of points for linear prediction
TDoff - number of raw data points predicted for ME_mod = LPb*
WDW - FID window multiplication mode
LB - Lorentzian broadening factor for WDW = em or gm
GB - Gaussian broadening factor for WDW = gm, sinc or qsinc
SSB - Sine bell shift for WDW = sine, qsine, sinc or qsinc
TM1, TM2 - limits of the trapezoidal window
PH_mod - phase correction mode
PHC0 - zero order phase correction value for PH_mod = pk
PHC1 - first order phase correction value for PH_mod = pk
FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)
REVERSE - flag indicating to reverse the spectrum

set by the ***xf2***, can be viewed with ***dpp*** or by typing ***1s mc2*** :

MC2 - Fourier transform mode (input of ***xf1*** on processed data)

set by the acquisition, can be viewed with ***dpa*** or by typing ***1s fnmode***:

FnMODE - Acquisition mode (input of ***xf1*** on raw data)

OUTPUT PARAMETERS

F1 parameters

can be viewed with ***dpp*** or by typing ***1s ft_mod*** etc.:

FT_mod - Fourier transform mode
FTSIZE - Fourier transform size

F2 parameters

can be viewed with ***dpp*** or by typing ***2s ymax_p***, ***2s ymin_p*** etc.:

YMAX_p - maximum intensity of the processed data
YMIN_p - minimum intensity of the processed data

S_DEV - standard deviation of the processed data
NC_proc - intensity scaling factor

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`ser` - raw data (input if `2rr` does not exist or is Fourier transformed in F1)
`acqu2s` - F1 acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed data (input if it exists but is not processed in F1)
`2ir` - second quadrant imaginary processed data (input if FnMODE ≠ QF)
`2ii` - second quadrant imaginary processed data (input if FnMODE = QF)
`proc` - F2 processing parameters
`proc2` - F1 processing parameters

Note that if ***xf1*** uses only `2rr` as input if it is executed before ***xf2***.

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed data
`2ir` - third quadrant imaginary processed data (output if FnMODE ≠ QF)
`2ii` - fourth quadrant imaginary processed data (output if FnMODE ≠ QF)
`2ii` - second quadrant imaginary processed data (output if FnMODE = QF)
`p2r1`, `p2r2`, `n2r1`, `n2r2` - positive and negative projections
`procs` - F2 processing status parameters
`proc2s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XF1

SEE ALSO

xf2, xfb, xf1p, xf1m, xf1ps, xht1, xif1, xtrfp1

xf1m, xf2m, xfbm

NAME

xf1m - calculate the magnitude spectrum in the F1 dimension
xf2m - calculate the magnitude spectrum in the F2 dimension
xfbm - calculate the magnitude spectrum in the F2 and F1 dimension

DESCRIPTION

The commands ***xf*m*** calculate the magnitude spectrum.

xf1m replaces the first and second quadrant data according to:

$$rr = \sqrt{rr^2 + ri^2}$$

$$ir = \sqrt{ir^2 + ii^2}$$

xf2m replaces the first and third quadrant data according to:

$$rr = \sqrt{rr^2 + ir^2}$$

$$ri = \sqrt{ri^2 + ii^2}$$

xfbm replaces the first quadrant data according to:

$$rr = \sqrt{rr^2 + ir^2 + ri^2 + ii^2}$$

where:

rr = real data (first quadrant)
ir = second quadrant imaginary data
ri = third quadrant imaginary data
ii = fourth quadrant imaginary data

First and second quadrant data can be created from raw data with ***xf2***. Third and fourth quadrant data cane be created with ***xf1***, from the first and second quadrant data, respectively. Note that the command ***xfb*** is a combination of ***xf2*** and ***xf1*** and creates all four quadrants.

The commands ***xf*m*** are, for example, used to convert a phase sensitive spectrum to magnitude spectrum. This is useful for data which cannot be phased properly or data which are not phase sensitive but have been acquired as such.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`p2r1`, `p2r2` - positive projections
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XF2M

XF1M

XFBM

SEE ALSO

xf2ps, xf1ps, xfbps

xf1p

NAME

xf1p - phase correction in the F1 dimension

DESCRIPTION

The command ***xf1p*** performs a first and second order phase correction in the F1 dimension, applying the values of PHC0 and PHC1. It works like the 1D command ***pk***. This means ***xf1p*** does not calculate the phase values, it simply applies the current values of PHC0 and PHC1. Therefore, ***xf1p*** is only useful when these values are known.

If the phase values are not known, phase correction can be done, interactively, from the 2D ***phase*** menu. When you have determined the phase correction in F1, and return to the main menu, you are asked if you want to do an ***xf1p***. If click ***OK***, ***xf1p*** is automatically performed.

The phase values can also be determined by reading a column (***rsc***) and determining the phase values from the 1D ***phase*** menu. On returning to the main menu, you have the option ***Save as 2D*** which will store the phase values in the 2D dataset where the 1D was extracted from. Alternatively, you can read a row with ***rsc***, phase correct the resulting 1D data with ***apk*** and set the calculated phase values (with ***edp***) on the 2D dataset. After the phase values are set, you can run an ***xf1p*** to apply them. (see also the AU programs ***psysexpro1*** and ***psysphasf1***).

xf1p uses but does not change the processing parameters PHC0 and PHC1 (***edp***). It does, however, change the corresponding processing status parameters (***dpp***), by adding the applied phase values.

INPUT PARAMETERS

F1 parameters

set by the user with ***edp*** or by typing ***1 phc0***, ***1 phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

OUTPUT PARAMETERS

F1 parameters

can be viewed with ***dpp*** or by typing ***1s phc0***, ***1s phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `ir`, `2ri`, `2ii` - processed 2D data
`proc2` - F1 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `ir`, `2ri`, `2ii` - processed 2D data
`p2r1`, `p2r2`, `n2r1`, `n2r2` - positive and negative projections
`proc2s` - F1 processing parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XF1P

SEE ALSO

xf2p, xfbp

xf1ps, xf2ps, xfbps

NAME

xf1ps - calculate the power spectrum in the F1 dimension
xf2ps - calculate the power spectrum in the F2 dimension
xfbps - calculate the power spectrum in the F2 and F1 dimension

DESCRIPTION

The commands ***xf*ps*** calculate the power spectrum.

xf1ps recalculates the first and second quadrant data according to:

$$rr = rr^2 + ri^2$$

$$ir = ir^2 + ii^2$$

xf2ps recalculates the first and third quadrant data according to:

$$rr = rr^2 + ir^2$$

$$ri = ri^2 + ii^2$$

xfbps recalculates the first quadrant data according to:

$$rr = rr^2 + ir^2 + ri^2 + ii^2$$

where:

rr = real data (first quadrant)
ir = second quadrant imaginary data
ri = third quadrant imaginary data
ii = fourth quadrant imaginary data

First and second quadrant data can be created from raw data with ***xf2***. Third and fourth quadrant data can be created with ***xf1***, from the first and second quadrant

data respectively. Note that the command ***xfb*** is a combination of ***xf2*** and ***xf1*** and creates all four quadrants.

The commands ***xf*ps*** is, for example, used in special cases to convert a phase sensitive spectrum to a magnitude spectrum. This is useful for data which cannot be phased properly or data which are not phase sensitive but have been acquired as such.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`p2r1`, `p2r2` - positive projections
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XF1PS

XF2PS

XFBPS

SEE ALSO

xf2m, xf1m, xfbm

xf2

NAME

xf2 - process data in the F2 dimension

DESCRIPTION

The command ***xf2*** processes a 2D dataset in the F2 dimension. This involves a Fourier transform which transforms time domain data (FID) into frequency domain data (spectrum). Depending on the F2 processing parameters BC_mod, WDW, ME_mod and PH_mod, ***xf2*** also performs baseline correction, window multiplication, linear prediction and phase correction, respectively. These steps are described in detail for the command ***xfb***.

Normally, 2D data are processed with the command ***xfb*** which performs a Fourier transform in both dimensions, F2 and F1. In some cases, however, 2D data must only be processed in the F2 dimension. Examples are T1 or T2 data or a 2D dataset which has been created from a series on 1D datasets.

Even if a 2D dataset must be processed in both dimension, it is sometimes useful to do that in two separate steps using the sequence ***xf2*** - ***xf1***. The result is exactly the same as with ***xfb*** with one exception; ***xfb*** performs a quad spike correction (see ***xfb***) and the sequence ***xf2*** - ***xf1*** does not.

xf2 takes the same options as ***xfb***.

xf2 can also be used to process one 2D plane of a 3D spectrum (see ***xfb***).

INPUT PARAMETERS

F2 and F1 parameters

set by the user with ***edp*** or by typing ***si***, ***stsr***, ***1 si*** etc.:

SI - size of the processed data
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
TDeff - number of raw data points to be used for processing
TDoff - first point of the FID used for processing (default 0)
XDIM - submatrix size (only used for the command ***xf2 xdim***)

set by the acquisition, can be viewed with ***dpa*** or by typing ***2s td***, ***1s td***:

TD - time domain; number of raw data points

F2 parameters

set by the user with ***edp*** or by typing ***bc_mod*** etc.

BC_mod - FID baseline correction mode
BCFW - filter width for BC_mod = sfil or qfil
COROFFS - correction offset for BC_mod = spol/qpol or sfil/qfil
ME_mod - FID linear prediction mode
NCOEF - number of linear prediction coefficients
LPBIN - number of points for linear prediction
TDoff - number of raw data points predicted for ME_mod = LPb*
WDW - FID window multiplication mode
LB - Lorentzian broadening factor for WDW = em or gm
GB - Gaussian broadening factor for WDW = gm, sinc or qsinc
SSB - Sine bell shift for WDW = sine, qsine, sinc or qsinc
TM1, TM2 - limits of the trapezoidal window
PH_mod - phase correction mode
PHC0 - zero order phase correction value for PH_mod = pk
PHC1 - first order phase correction value for PH_mod = pk
FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)
REVERSE - flag indicating to reverse the spectrum

set by the acquisition, can be viewed with ***dpa*** or by typing ***2s aq_mod***:

AQ_mod - acquisition mode (determines the Fourier transform mode)

F1 parameters

set by the acquisition, can be viewed with ***dpa*** or by typing ***1s fnmode*** :

FnMODE - Fourier transform mode

OUTPUT PARAMETERS

F2 and F1 parameters

can be viewed with ***dpp*** or by typing ***2s si***, ***1s si*** etc.:

SI - size of the processed data
TDeff - number of raw data points that were used for processing
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform

FTSIZE - Fourier transform size
XDIM - submatrix size

F2 parameters

can be viewed with ***dpp*** or by typing ***2s ft_mod***, ***2s ymax_p*** etc.:

FT_mod - Fourier transform mode
YMAX_p - maximum intensity of the processed data
YMIN_p - minimum intensity of the processed data
S_DEV - standard deviation of the processed data
NC_proc - intensity scaling factor

can only be viewed by typing ***2s bytordp***:

BYTORDP - byte order of the processed data

F1 parameters

set by the acquisition, can be viewed with ***dpp*** or by typing ***1s mc2*** :

MC2 - Fourier transform mode

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`ser` - raw data (input if `2rr` does not exist or is Fourier transformed in F2)
`acqus` - F2 acquisition status parameters
`acqu2s` - F1 acquisition parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - processed data (input if it exists but is not Fourier transformed in F2)
`proc` - F2 processing parameters
`proc2` - F1 processing parameters

Note that if `2rr` is input, `2ri` is also input if ***xf1*** has been done.

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - first quadrant real processed data
`2ir` - second quadrant imaginary processed data (output if FnMODE ≠ QF)
`2ii` - second quadrant imaginary processed data (output if FnMODE = QF)
`p2r1`, `p2r2`, `n2r1`, `n2r2` - positive and negative projections

`procs` - F2 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XF2

SEE ALSO

xf1, xfb, xf2p, xf2m, xf2ps, xht2, xif2, xtrfp2

xf2p

NAME

xf2p - phase correction in the F2 dimension

DESCRIPTION

The command ***xf2p*** performs a first and second order phase correction in the F2 dimension, applying the values of PHC0 and PHC1. It works like the 1D command ***pk***. This means ***xf2p*** does not calculate the phase values, it simply applies the current values of PHC0 and PHC1. Therefore, ***xf2p*** is only useful when these values are known.

If the phase values are not known, phase correction can be done, interactively, from the 2D ***phase*** menu. When you have determined the phase correction in F2 and return to the main menu, you are asked if you want to do an ***xf2p***. If click ***OK***, ***xf2p*** is automatically performed.

The phase values can also be determined by reading a row (***rsr***) and determining the phase values from the 1D ***phase*** menu. On returning to the main menu, you have the option ***Save as 2D*** which will store the phase values in the 2D dataset where the 1D was extracted from. Alternatively, you can read a row with ***rsr***, phase correct the resulting 1D data with ***apk*** and set the calculated phase values (with ***edp***) on the 2D dataset. After the phase values are set, you can run an ***xf2p*** to apply them.

xf2p uses but does not change the processing parameters PHC0 and PHC1 (***edp***). It does, however, change the corresponding processing status parameters (***dpp***), by adding the applied phase values.

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***phc0***, ***phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

OUTPUT PARAMETERS

F2 parameters

can be viewed with ***dpp*** or by typing ***2s phc0***, ***2s phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `ir`, `2ri`, `2ii` - processed 2D data
`p2r1`, `p2r2`, `n2r1`, `n2r2` - positive and negative projections
`proc` - F2 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `ir`, `2ri`, `2ii` - processed 2D data
`procs` - F2 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XF2P

SEE ALSO

xf1p, xfbp

xfb

NAME

xfb - process 2D data in the F2 and F1 dimension

DESCRIPTION

The command ***xfb*** processes a 2D dataset in the F2 and F1 dimension. This involves a Fourier transform which transforms time domain data into frequency domain data. Depending on the processing parameters BC_mod, WDW, ME_mod and PH_mod, ***xfb*** also performs baseline correction, window multiplication, linear prediction and spectrum phase correction.

The processing steps done by ***xfb*** can be described as follows:

1. Baseline correction of the 2D time domain data
   Each row and/or column is baseline corrected according to BC_mod. This parameter takes the value *no*, *single*, *quad*, *spol*, *qpol sfil* or *qfil*. More details on BC_mod can be found in chapter 2.4.
2. Linear prediction of the 2D time domain data
   Linear prediction is done according to ME_mod. This parameter takes the value *no*, *LPfr*, *LPfc*, *LPbr* or *LPbc*. Usually, ME_mod = no, which means no prediction is done. Forward prediction (*LPfr* or *LPfc*) can, for example, be used to extend truncated FIDs. Backward prediction (*LPbr* or *LPbc*) can be used to improve the initial data points of the FID. Linear prediction is only performed for NCOEF > 0. Furthermore, LPBIN and, for backward prediction, TDoff play a role (see these parameters in chapter 2.4).
3. Window multiplication of the 2D time domain data
   Each row and/or column is multiplied with a window function according to WDW. This parameter takes the value *em*, *gm*, *sine*, *qsine*, *trap*, *user*, *sinc*, *qsinc*, *traf* or *trafs*. More details on WDW can be found in chapter 2.4.
4. Fourier transform of the 2D time domain data
   Each row is Fourier transformed according to the acquisition status parameter AQ_mod as shown in table 4.4. Each column (F1) is Fourier transformed according to the acquisition status parameter FnMODE as shown in table 4.5. ***xfb*** does <u>not</u> evaluate the processing parameter

F2 AQ_mod	Fourier transform mode	F2 status FT_mod
qf	forward, single, real	fsr
qsim	forward, quad, complex	fqc
qseq	forward, quad, real	fqr
DQD	forward, quad, complex	fqc

Table 4.4

F1 FnMODE	Fourier transform mode	F1 status FT_mod
QF	forward, quad, complex	fqc
QSEQ	forward, quad, real	fqr
TPPI	forward, single, real	fsr
States	forward, quad, complex	fqc
States-TPPI	forward, single, complex	fsc
Echo-AntiEcho	forward, quad, complex	fqc

Table 4.5

FT_mod! However, it stores the Fourier transform mode as it was evaluated from AQ_mod (F2) or FnMODE (F1) in the processing status parameter FT_mod. If, for some reason, you want to Fourier transform a spectrum with a different mode, you can set the processing parameter FT_mod (with ***edp***) and use the command ***xtrf*** (see ***xtrf***). More details on FT_mod can be found in chapter 2.4.

5. Phase correction of the 2D spectrum according to PH_mod. This parameter takes the value *no*, *pk*, *mc* or *ps*. For PH_mod = pk, ***xfb*** applies the values of PHC0 and PHC1. This is only useful if the phase values are known. If they are not, you can do an interactive phase correction from the ***phase*** menu after ***xfb*** has finished. More details on PH_mod can be found in chapter 2.4.

The size of the processed data is determined by the processing parameter SI; SI real and SI imaginary points are created. A typical value for SI is TD/2 in which case, all raw data points are used and no zero filling is done. In fact, several parameters control the number of input and output data points, for example:

1. SI > TD/2: the raw data are zero filled before the Fourier transform

2. SI < TD/2: only the first 2*SI raw data points are used
3. 0 < TDeff < TD: only the first TDeff raw data points are used
4. 0 < TDoff < TD: the first TDoff raw data points are cut off at the beginning and TDoff zeroes are appended at the end (corresponds to left shift).
5. TDoff < 0: -TDoff zeroes are prepended at the beginning. Note that:
   - for SI < (TD-TDoff)/2 raw data are cut off at the end
   - for DIGMOD=digital, the zeroes would be prepended to the group delay which does not make sense. You can avoid that by converting the raw data with ***convdta*** before you process them.
6. 0 < STSR < SI: only the processed data between STSR and STSR+STSI are stored (if STSI = 0, STSR is ignored and SI points are stored)
7. 0 < STSI < SI: only the processed data between STSR and STSR+STSI are stored.

Note that only in the first case the processed data contain the total information of the raw data. In all other cases, information is lost.

xfb performs a quad spike correction which means that the central data point of the spectrum is replaced by the average of the neighbouring data points in the F1 dimension. Note that the quad spike correction is skipped if you process the data with the sequence ***xf2*** - ***xf1***.

xfb evaluates the parameter FCOR. The first point of the FIDs is multiplied with the value of FCOR which lies between 0.0 and 2.0. For digitally filtered Avance data, FCOR is only used in the F1 dimension. In F2, it has no effect because the first point is part of the group delay and, as such, is zero. However, A*X data or Avance data measured with DIGMOD = analog, FCOR is used in F1 and F2.

xfb evaluates the F2 parameter PKNL. On A*X spectrometers, PKNL = true causes a non linear 5th order phase correction of the raw data. This corrects possible errors caused by non linear behaviour of the analog filters. On Avance spectrometers, PKNL must always be set to TRUE. For digitally filtered data, it causes ***xfb*** to handle the group delay of the FID. For analog data it has no effect.

xfb evaluates the F2 and F1 parameter REVERSE. If REVERSE = TRUE, the spectrum will be reversed in the corresponding dimension, i.e. the first data point becomes the last and the last data point becomes the first. The same effect can be obtained with the commands ***rev2*** and/or ***rev1*** after ***xfb***.

xfb is normally used without options. There are, however, several options available:

n

xfb normally stores real and imaginary processed data. However, the imaginary data are only needed for phase correction. If the parameters PHC0 and PHC1 are set correctly, then you don't need to store the imaginary data. The option ***n*** allows you to do that. This will save processing time and disk space. If you still want to do a phase correction, you can create imaginary data from the real data with a Hilbert transform (see ***xht2*** and ***xht1***).

nc_proc value

xfb scales the data such that, i.e. the highest intensity of the spectrum lies between 2^{28} and 2^{29}. The intensity scaling factor is stored in the processing status parameter NC_proc and can be viewed with ***dpp***. The option ***nc_proc*** causes ***xfb*** to use a specific scaling factor. However, you can only scale down the data by entering a greater (more positive) value than the one ***xfb*** would use without this option. If you enter a smaller (more negative) value, the option will be ignored to prevent data overflow. The option ***nc_proc last*** causes ***xfb*** to use the current value of the status processing parameter NC_proc, i.e. the value set by the previous processing step on this dataset.

raw/proc

xfb works on raw data if no processed data exist or if processed data exist and have been Fourier transformed in F2 and/or F1. One of them is usually true, i.e. the data have not been processed yet or they have been processed, for example with ***xfb***. If, however, the data have been processed with ***xtrf*** with FT_mod = no, they are not Fourier transformed and a subsequent ***xfb*** will work on the processed data. The ***raw*** option causes ***xfb*** to work on the raw data, no matter what. The ***proc*** option causes ***xfb*** to work on the processed data. If these do not exist or are Fourier transformed, the command stops and displays an error message. In other words, the option ***proc*** prevents ***xfb*** to work on raw data.

big/little

xfb stores the data in the data byte order (big or little endian) of the computer it runs on, e.g. big endian on SGI UNIX workstations and little endian on PCs. The byte order is stored in the processing status parameter BYTORDP

which can be viewed with ***2s bytordp***. The option ***big*** or ***little*** allows you to predefine the byte order. This, for example, is used to read processed data with third party software which can not interpret BYTORDP. This option is only evaluated when ***xfb*** works on the raw data.

xdim

Large 2D spectra are stored in the so-called submatrix format. The size of the submatrices are calculated by ***xfb*** and depend on the size of the spectrum and the available memory. The option ***xdim*** allows you to use predefined submatrix sizes. It causes ***xfb*** to interpret the F2 and F1 processing parameter XDIM which can be set with the commands ***2 xdim*** and ***1 xdim***, respectively. The actually used submatrix sizes, whether predefined or calculated, are stored as the F2 and F1 processing status parameter XDIM and can be viewed with ***dpp***. Predefining submatrix sizes is, for example, used to read the processed data with third party software which can not interpret the processing status parameter XDIM. This option is only evaluated when ***xfb*** works on the raw data.

xfb can also be used to process one 2D plane of a 3D spectrum. This can be a plane in the F3-F2 or in the F3-F1 direction. The output 2D data are stored in a separate procno. When the current dataset is a 3D, ***xfb*** will prompt you for the plane direction, the plane number and the output procno. Alternatively, you can enter this information as arguments on the command line, for example:

```
xfb s23 17 2
```

will read the F3-F2 plane number 17 and store it under procno 2.

Normally, ***xfb*** stores the entire spectral region as determined by the spectral width. You can, however, do a so-called strip transform which means that only a certain region of the spectrum is stored. This can be done by setting the parameters STSR and STSI which represent the strip start and strip size, respectively. They both can take a value between 0 and SI. The values which are actually used can be a little different. STSI is always rounded to the next multiple of 16. Furthermore, when the data are stored in submatrix format (see below), STSI is rounded to the next higher multiple of the submatrix size. Type ***dpp*** to check this; if XDIM is smaller than SI, then the data are stored in submatrix format and STSI is a multiple of XDIM.

F2 →
F1 ↓

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	33	34	35	36	38	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Table 4.6 2D data in sequential storage format

Depending on size of the processed data and the available computer memory, ***xfb*** stores the data in sequential or submatrix format. Sequential format is used when the entire dataset fits in memory, otherwise submatrix format is used. ***xfb*** automatically calculates the submatrix sizes such that one row(F2) of submatrices fits in the available memory. The calculated submatrix sizes are stored in the processing status parameter XDIM (type ***dpp***). Table 4.6 and 4.7 shows the alignment of the data points for sequential and submatrix format, respectively. This example shows a dataset with the following sizes: F2 SI = 16, F1 SI = 16, F2 XDIM = 8, F1 XDIM = 4. The storage handling is completely transparent to the user and is only of interest when the data are interpreted by third party software.

As can be seen in table 4.5, the acquisition mode in F1 (FnMODE) determines the Fourier transform mode. Furthermore, FnMODE determines the data storage mode. The description below demonstrates the difference in data storage between a data set with FnMODE = QF and one with FnMODE ≠ QF.

F2 →
F1 ↓

0	1	2	3	4	5	6	7	32	33	34	35	36	37	38	39
8	9	10	11	12	13	14	15	40	41	42	43	44	45	46	47
16	17	18	19	20	21	22	23	48	49	50	51	52	53	54	55
24	25	26	27	28	29	30	31	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	96	97	98	99	100	101	102	103
72	73	74	75	76	77	78	79	104	105	106	107	108	109	110	111
80	81	82	83	84	85	86	87	112	113	114	115	116	117	118	119
88	89	90	91	92	93	94	95	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	160	161	162	163	164	165	166	167
136	137	138	139	140	141	142	143	168	169	170	171	172	173	174	175
144	145	146	147	148	149	150	151	176	177	178	179	180	181	182	183
152	153	154	155	156	157	158	159	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	224	225	226	227	228	229	230	231
200	201	202	203	204	205	206	207	232	233	234	235	236	237	238	239
208	209	210	211	212	213	214	215	240	241	242	243	244	245	246	247
216	217	218	219	220	221	222	223	248	249	250	251	252	253	254	255

Table 4.7 2D data in 8*4 submatrix storage format

FnMODE = QF

xfb performs complex (two-quadrant) processing. In F2 the data are acquired phase sensitive, in F1 non-phase sensitive. In the example below, the following parameter settings are used:

In F2: TD = 8, SI is 4
In F1: TD = 2, SI = 2

Furthermore, the following notation is used for individual data points:

r*n***c***m* : point *n* of FID *m*. This point is real in F2 and complex in F1
i*n***c***m* : point *n* of FID *m*. This point is imaginary in F2 and complex in F1

Input F2 processing
(raw data)

F2 →
F1 ↓

r1c1 i1c1	r2c1 i2c1	r3c1 i3c1	r4c1 i4c1
r1c2 i1c2	r2c2 i2c2	r3c2 i3c2	r4c2 i4c2

`ser` file

For F2 processing, **r1c1**, **i1c1** is the first complex input point, r2c1, i2c1 the second etc.

Output F2 processing = Input F1 processing

F2 →
F1 ↓

r1c1	r2c1	r3c1	r4c1
r1c2	r2c2	r3c2	r4c2

`2rr` file

i1c1	i2c1	i3c1	i4c1
i1c2	i2c2	i3c2	i4c2

`2ii` file

Below, the F1 input data are simply redisplayed in vertical order, with the first complex input point in bold.

Input F1 processing

F2 →
F1 ↓

r1c1	r2c1	r3c1	r4c1
r1c2	r2c2	r3c2	r4c2

`2rr` file

i1c1	i2c1	i3c1	i4c1
i1c2	i2c2	i3c2	i4c2

`2ii` file

Output F1 processing

F2 →
F1 ↓

r1c1	r2c1	r3c1	r4c1
r1c2	r2c2	r3c2	r4c2

`2rr` file

i1c1	i2c1	i3c1	i4c1
i1c2	i2c2	i3c2	i4c2

`2ii` file

FnMODE ≠ QF

xfb performs hypercomplex (four-quadrant) processing. Both in F2 and F1, the data are acquired phase sensitive. In the example below, the following parameters settings are used:

In F2: TD = 8, SI is 4
In F1: TD = 4, SI = 2

Furthermore, the following notation is used for individual data points:

- **r***n***r***m* : point *n* of FID *m*. This point is real in F2 and F1
- **i***n***r***m* : point *n* of FID *m*. This point is imaginary in F2 and real in F1
- **r***n***i***m*: point *n* of FID *m*. This point is real in F2 and imaginary in F1
- **i***n***i***m* : point *n* of FID *m*. This point is imaginary in F2 and F1

Input F2 processing
(raw data)

F2 →
F1 ↓

r1r1 i1r1	r2r1 i2r1	r3r1 i3r1	r4r1 i4r1
r1i1 i1i1	r2i1 i2i1	r3i1 i3i1	r4i1 i4i1
r1r2 i1r2	r2r2 i2r2	r3r2 i3r2	r4r2 i4r2
r1i2 i1i2	r2i2 i2i2	r3i2 i3i2	r4i2 i4i2

`ser` file

For F2 processing, **r1r1**, **i1r1** is the first hypercomplex input data point, r2r1, i2r1 the second etc.

Output F2 processing = Input F1 processing

F2 →
F1 ↓

r1r1	r2r1	r3r1	r4r1
r1i1	r2i1	r3i1	r4i1
r1r2	r2r2	r3r2	r4r2
r1i2	r2i2	r3i2	r4i2

`2rr` file

i1r1	i2r1	i3r1	i4r1
i1i1	i2i1	i3i1	i4i1
i1r2	i2r2	i3r2	i4r2
i1i2	i2i2	i3i2	i4i2

`2ir` file

Below, the F1 input data are simply redisplayed, with the first F1 complex input points in bold.

Input F1 processing

F2 →
F1 ↓

r1r1	r2r1	r3r1	r4r1
r1i1	r2i1	r3i1	r4i1
r1r2	r2r2	r3r2	r4r2
r1i2	r2i2	r3i2	r4i2

`2rr` file

i1r1	i2r1	i3r1	i4r1
i1i1	i2i1	i3i1	i4i1
i1r2	i2r2	i3r2	i4r2
i1i2	i2i2	i3i2	i4i2

`2ir` file

Output F1 processing

F2 →
F1 ↓

r1r1	r2r1	r3r1	r4r1
r1r2	r2r2	r3r2	r4r2

`2rr` file

i1r1	i2r1	i3r1	i4r1
i1r2	i2r2	i3r2	i4r2

`2ir` file

r1i1	r2i1	r3i1	r4i1
r1i2	r2i2	r3i2	r4i2

`2ri` file

i1i1	i2i1	i3i1	i4i1
i1i2	i2i2	i3i2	i4i2

`2ii` file

Note that:

- for FnMODE ≠ QF, zero filling once in F1 is done when SI = TD. For FnMODE = QF, zero filling once in F1 is done when SI = 2*TD.
- FnMODE = QF is normally used on magnitude or power data. For this purpose, the F1 processing parameter PH_mod must be set to MC or PS, respectively. Note that in these cases, no imaginary data are stored after F1 processing.
- the command ***xfb n*** does not store imaginary data after F1 processing.

INPUT PARAMETERS

F2 and F1 parameters

set by the user with ***edp*** or by typing ***bc_mod***, ***bcfw***, ***1 bc_mod*** etc.

BC_mod - FID baseline correction mode

BCFW - filter width for BC_mod = sfil or qfil

COROFFS - correction offset for BC_mod = spol/qpol or sfil/qfil

ME_mod - FID linear prediction mode

NCOEF - number of linear prediction coefficients
LPBIN - number of points for linear prediction
TDoff - number of raw data points predicted for ME_mod = LPb*
WDW - FID window multiplication mode
LB - Lorentzian broadening factor for WDW = em or gm
GB - Gaussian broadening factor for WDW = gm, sinc or qsinc
SSB - Sine bell shift for WDW = sine, qsine, sinc or qsinc
TM1, TM2 - limits of the trapezoidal window
PH_mod - phase correction mode
PHC0 - zero order phase correction value for PH_mod = pk
PHC1 - first order phase correction value for PH_mod = pk
SI - size of the processed data
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
TDeff - number of raw data points to be used for processing
TDoff - first point of the FID used for processing (default 0)
FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)
REVERSE - flag indicating to reverse the spectrum
XDIM - submatrix size (only used for the command ***xfb xdim***)

set by the acquisition, can be viewed with ***dpa*** or by typing ***2s td***, ***1s td*** etc.:

TD - time domain; number of raw data points

F2 parameters

set by the user with ***edp*** or by typing ***pknl*** :

PKNL - group delay handling (Avance) or filter correction (A*X)

set by the acquisition, can be viewed with ***dpa*** or by typing ***2s aq_mod***.:

AQ_mod - acquisition mode (determines the Fourier transform mode)

F1 parameters

set by the acquisition, can be viewed with ***dpa*** or by typing ***1s fnmode*** :

FnMODE - F1 Acquisition transform mode

set by the acquisition, can be viewed with ***dpa*** or by typing ***1s fnmode*** :

MC2 - FT mode in F1 (only used if F1-FnMODE = undefined)

OUTPUT PARAMETERS

F2 and F1 parameters

can be viewed with ***dpp*** or by typing ***2s si***, ***1s si*** etc.:

SI - size of the processed data
TDeff - number of raw data points that were used for processing
FTSIZE - Fourier transform size
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
XDIM - submatrix size
FT_mod - Fourier transform mode

F2 parameters

can be viewed with ***dpp*** or by typing ***2s ymax_p***, ***2s ymin_p*** etc.:

YMAX_p - maximum intensity of the processed data
YMIN_p - minimum intensity of the processed data
S_DEV - standard deviation of the processed data
NC_proc - intensity scaling factor

can only be viewed by typing ***2s bytordp***:

BYTORDP - byte order of the processed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`ser` - raw data (input if `2rr` does not exit or is Fourier transformed)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data (input if it exists but is not Fourier transformed)
`proc` - F2 processing parameters
`proc2` - F1 processing parameters
`acqus` - F2 acquisition status parameters
`acqu2s` - F1 acquisition status parameters

Note that if `2rr` is input, then `2ir` and `2ri` can also be input, depending on the processing status of the data.

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

For FnMODE ≠ QF:

`2rr` - real processed 2D data
`2ir` - second quadrant imaginary processed data
`2ri` - third quadrant imaginary processed data
`2ii` - fourth quadrant imaginary processed data

For FnMODE = QF:

`2rr` - real processed 2D data
`2ii` - second quadrant imaginary processed data

For all values of FnMODE:

`p2r1` - positive projection of the F2 dimension
`n2r1` - negative projection of the F2 dimension
`p2r2` - positive projection of the F1 dimension
`n2r2` - negative projection of the F1 dimension
`dsp` - compressed data file for display (output if SI > 256)
`dsp.ext` - expanded region of the compressed data for display
`procs` - F2 processing status parameters
`proc2s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XFB

If you want to use XFB with an option, you can do that with XCMD, e.g. XCMD("xfb raw")

SEE ALSO

xf2, xf1, xfbp, xfbm, xfbps, xtrf, xtrf2, xtrfp2, xtrfp1, xif2, xif1

xfbp

NAME

xfbp - 2D phase correction in the F2 and F1 dimension

DESCRIPTION

The command ***xfbp*** performs a zero and first order 2D phase correction in the F2 and F1 dimension, applying the values of PHC0 and PHC1. It works like the 1D command ***pk***. This means ***xfbp*** does not calculate the phase values, it simply applies the current values of PHC0 and PHC1. Therefore, ***xfbp*** is only useful when these values are known.

If the phase values are not known, phase correction can be done, interactively, from the 2D ***phase*** menu. When you have determined the phase correction in both F2 and F1, and return to the main menu, you are asked if you want to do an ***xfbp***. If click OK, ***xfbp*** is automatically performed.

The phase values can also be determined by reading a row (***rsr***) and/or column (***rsc***), determining the phase values from the 1D ***phase*** menu. On returning to the main menu, you have the option ***Save as 2D*** which will store the phase values in the 2D dataset where the 1D was extracted from. Alternatively, you can read a row with ***rsr***, or a column with ***rsc***, phase correct the resulting 1D data with ***apk*** and set the calculated phase values (with ***edp***) on the 2D dataset. After the phase values are set, you can run an ***xfbp*** to apply them.

xfbp uses but does not change the processing parameters PHC0 and PHC1 (***edp***). It does, however, change the corresponding processing status parameters (***dpp***), by adding the applied phase values.

INPUT PARAMETERS

F2 and F1 parameters

set by the user with ***edp*** or by typing ***phc0***, ***phc1***, ***1 phc0*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

OUTPUT PARAMETERS

F2 and F1 parameters

can be viewed with ***dpp*** or by typing ***2s phc0***, ***1s phc0*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `ir`, `2ri`, `2ii` - processed 2D data
`procs` - F2 processing status parameters
`proc2s` - F1 processing status parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `ir`, `2ri`, `2ii` - processed 2D data
`p2r1`, `p2r2`, `n2r1`, `n2r2` - positive and negative projections
`procs` - F2 processing status parameters
`proc2s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XFBP

SEE ALSO

xf2p, xf1p, xtrf, xtrfp, xtrfp2, xtrfp1

xht1

NAME

xht1 - Hilbert transform of 2D data in the F1 dimension

DESCRIPTION

The command ***xht1*** performs a Hilbert transform of 2D data in the F1 dimension. Hilbert transform creates imaginary data from the real data.

Imaginary data are required for phase correction. They are normally created during Fourier transform with ***xfb*** or ***xf1***. If, however, if the imaginary data were not stored (***xfb n***) or have been deleted (***deli***), you can (re)create them with ***xht1***.

Hilbert transform can also be used if the imaginary data exist but do not match the real data. This is the case when the latter have been manipulated after Fourier transform, for example by ***abs1***, ***sub1***, ***sym*** or third party software.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`2ir` - second quadrant imaginary data (input if they exist)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2ri` - third quadrant imaginary data (created from `2rr`)
`2ii` - fourth quadrant imaginary data (created from `2ir` if this exists)
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XHT1

SEE ALSO

xht2

xht2

NAME

xht2 - Hilbert transform of 2D data in the F2 dimension

DESCRIPTION

The command ***xht2*** performs a Hilbert transform of 2D data in the F2 dimension. Hilbert transform creates imaginary data from the real data.

Imaginary data are required for phase correction. They are normally created during Fourier transform with ***xfb*** or ***xf2***. If, however, if the imaginary data were not stored (***xfb n***) or have been deleted (***deli***), you can (re)create them with ***xht2***.

Hilbert transform can also be used if the imaginary data exist but do not match the real data. This is the case when the latter have been manipulated after Fourier transform, for example by ***abs2***, ***sub2***, ***sym*** or third party software.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`2ri` - third quadrant imaginary data (input if they exist)

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2ir` - second quadrant imaginary data (created from `2rr`)
`2ii` - fourth quadrant imaginary data (created from `2ri` if this exists)
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XHT2

SEE ALSO

xht1

xif2, xif1

NAME

xif2 - inverse Fourier transform of 2D data in the F2 dimension
xif1 - inverse Fourier transform of 2D data in the F1 dimension

DESCRIPTION

The command ***xif2*** performs an inverse Fourier transform in the F2 dimension. This means frequency domain data (spectrum) are transformed into time domain data (FID).

xif1 performs an inverse Fourier transform in the F1 dimension.

Note that after ***xif2*** or ***xif1*** (or both), the data are still stored as processed data, i.e. the raw data are not overwritten. You can, however, create pseudo-raw data with the command ***genser*** which creates a new dataset.

Inverse Fourier transform can also be done with the commands ***xtrfp***, ***xtrfp2*** and ***xtrfp1***. This can be done as follows:

1. Type ***dpp*** and check the status FT_mod
2. Type ***edp*** to set the processing parameters; set BC_mod, WDW, ME_mod and PH_mod to *no* and FT_mod to the inverse equivalent of the status FT_mod
3. Perform ***xtrfp***, ***xtrfp2*** or ***xtrfp1***

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `ir`, `2ri`, `2ii` - processed 2D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `ir`, `2ri`, `2ii` - processed 2D data
`p2r1`, `p2r2`, `n2r1`, `n2r2` - positive and negative projections
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XIF2

XIF1

SEE ALSO

genser, xtrfp, xtrfp2, xtrfp1

xtrf, xtrf2

NAME

xtrf - user defined processing of 2D raw data in the F2 and F1 dimension
xtrf2 - user defined processing of 2D raw data in the F2 dimension

DESCRIPTION

The command ***xtrf*** performs user defined processing of the raw data in both the F2 and F1 dimension. It works like ***xfb***, except for the following differences:

- the Fourier transform is performed according to the processing parameter FT_mod, whereas the acquisition status parameter AQ_mod is ignored. This, for example allows you to process the data without Fourier transform (FT_mod = no). Furthermore, you can choose a Fourier transform mode different from the one that would be evaluated from the acquisition mode. This feature is not used very often because the Fourier transform as evaluated from the acquisition mode is usually the correct one. If, however, you want to manipulate the acquisition mode of the raw data, you can Fourier transform the data with one FT_mod, inverse Fourier transform them with a different FT_mod. Then you can use ***genser*** to create pseudo-raw data with a different acquisition mode than the original raw data. Table 4.8 shows a list of values of FT_mod:
- a baseline correction is performed according to BC_mod. This parameter can take the value *no*, *single*, *quad*, *spol*, *qpol*, *sfil* or *qfil*. ***xtrf*** evaluates BC_mod for the baseline correction mode (e.g. quad, qpol or qfil) and for the detection mode (e.g. single or quad, spol or qpol, sfil or qfile). Note that ***xfb*** evaluates the acquisition status parameter AQ_mod for the detection mode. More details on BC_mod can be found in chapter 2.4.
- when all parameters mentioned above are set to *no*, no processing is done but the raw data are stored as processed data and displayed on the screen. This means the raw data are converted to submatrix format (files `2rr`, `2ir`, `2ri` and `2ii`) and scaled according to the vertical resolution. The intensity scaling factor is stored in the processing status parameter NC_proc and can be viewed with ***dpp***. The size of these processed data and the number of raw data points which are used are determined by the

FT_mod	Fourier transform mode
no	no Fourier transform
fsr	forward, single channel, real
fqr	forward, quadrature, real
fsc	forward, single channel, complex
fqc	forward, quadrature, complex
isr	inverse, single channel, real
iqr	inverse, quadrature, real
isc	inverse, single channel, complex
iqc	inverse, quadrature, complex

Table 4.8

parameters SI, TDeff and TDoff, as described for the command ***xfb***. For example, if 0 < TDeff < TD, the processed data are truncated. This allows you to create pseudo-raw data with a smaller size than the original raw data (see also ***genser***).

The F1 Fourier transform mode and data storage mode depends on the F1 acquisition mode (see INPUT PARAMETERS below and the description of ***xfb***).

The command ***xtrf2*** works like ***xtrf*** except that it only works in the F2 dimension.

xtrf and ***xtrf2*** take the same options as ***xfb***.

xtrf can be used to do a combination of forward and backward prediction. Just run ***xtrf*** with ME_mod = LPfc and ***xtrfp*** (or ***xfb***) with ME_mod = LPbc.

INPUT PARAMETERS

F2 and F1 dimension

set by the user with ***edp*** or by typing ***si***, ***bc_mod***, ***bcfw*** etc.:

SI - size of the processed data
TDeff - number of raw data points to be used for processing
TDoff - first point of the FID used for processing (default 0)
FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)
BC_mod - FID baseline correction mode

BCFW - filter width for BC_mod = sfil or qfil
COROFFS - correction offset for BC_mod = spol/qpol or sfil/qfil
ME_mod - FID linear prediction mode
NCOEF - number of linear prediction coefficients
LPBIN - number of points for linear prediction
TDoff - number of raw data points predicted for ME_mod = LPb*
WDW - FID window multiplication mode
LB - Lorentzian broadening factor for WDW = em or gm
GB - Gaussian broadening factor for WDW = gm, sinc or qsinc
SSB - Sine bell shift for WDW = sine, qsine, sinc or qsinc
TM1, TM2 - limits of the trapezoidal window
FT_mod - Fourier transform mode
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
REVERSE - flag indicating to reverse the spectrum
PKNL - group delay handling (Avance) or filter correction (A*X)
PH_mod - phase correction mode
PHC0 - zero order phase correction value for PH_mod = pk
PHC1 - first order phase correction value for PH_mod = pk

set by the acquisition, can be viewed with ***dpa*** or by typing ***2s td***, ***1s td*** :

TD - time domain; number of raw data points

F1 dimension

set by the acquisition, can be viewed with ***dpa*** or by typing ***1s fnmode***:

FnMODE - Acquisition mode

OUTPUT PARAMETERS

F2 and F1 parameters

can be viewed with ***dpp*** or by typing ***2s si***, ***1s si*** etc.:

SI - size of the processed data
TDeff - number of raw data points that were used for processing
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
XDIM - submatrix size

F2 parameters

can be viewed with ***dpp*** or by typing ***2s ymax_p***, ***2s ymin_p*** etc.:

YMAX_p - maximum intensity of the processed data
YMIN_p - minimum intensity of the processed data
S_DEV - standard deviation of the processed data
NC_proc - intensity scaling factor

can only be viewed by typing ***2s bytordp***:

BYTORDP - byte order of the processed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`ser` - raw data
`acqus` - F2 acquisition status parameters
`acqu2s` - F1 acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - F2 processing parameters
`proc2` - F1 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`p2r1`, `p2r2`, `n2r1`, `n2r2` - positive and negative projections
`procs` - processing status parameters
`proc2s` - processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XTRF

XTRF2

SEE ALSO

xtrfp, xtrfp2, xtrfp1, xfb, xf2, xf1

xtrfp, xtrfp2, xtrfp1

NAME

xtrfp - user defined processing of 2D processed data in the F2 and F1 dimension
xtrfp2 - user defined processing of 2D processed data in the F2 dimension
xtrfp1 - user defined processing of 2D processed data in the F1dimension

DESCRIPTION

The command ***xtrfp*** performs a user defined processing of processed data both the F2 and F1 dimension. It works like ***xtrf***, except that it only works on processed data. If processed data do not exist, an error message is displayed. If processed data do exist, they are further processed according to the parameters BC_mod, WDW, ME_mod, FT_mod and PH_mod as described for ***xtrf***.

xtrfp2 works like ***xtrfp***, except that it only works in the F2 dimension.

xtrfp1 works like ***xtrfp***, except that it only works in the F1 dimension.

The ***xtrfp**** commands can, for example, be used to perform multiple additive baseline corrections. This can be necessary if the raw data contain multiple frequency baseline distortions. You cannot do this with ***xfb*** or ***xtrf*** because these commands always work on the raw data, i.e. they are not additive.

xtrfp, ***xtrfp2*** and ***xtrfp1*** can also be used for inverse Fourier transform. This can be done as follows:

1. Type ***dpp*** to check the status FT_mod
2. Type ***edp*** to set the processing parameters; set BC_mod, WDW, ME_mod and PH_mod to *no* and FT_mod to the inverse equivalent of the status FT_mod
3. Perform ***xtrfp***, ***xtrfp2*** or ***xtrfp1***

However, a simpler way to do an inverse Fourier transform is the usages of the commands ***xif2*** and ***xif1***.

INPUT PARAMETERS

F2 and F1 parameters

set by the user with ***edp*** or by typing ***bc_mod***, ***bcfw*** etc.:

BC_mod - FID baseline correction mode
BCFW - filter width for BC_mod = sfil or qfil
COROFFS - correction offset for BC_mod = spol/qpol or sfil/qfil
ME_mod - FID linear prediction mode
NCOEF - number of linear prediction coefficients
LPBIN - number of points for linear prediction
TDoff - number of raw data points predicted for ME_mod = LPb*
WDW - FID window multiplication mode
LB - Lorentzian broadening factor for WDW = em or gm
GB - Gaussian broadening factor for WDW = gm, sinc or qsinc
SSB - Sine bell shift for WDW = sine, qsine, sinc or qsinc
TM1, TM2 - limits of the trapezoidal window
FT_mod - Fourier transform mode
PH_mod - phase correction mode
PHC0 - zero order phase correction value for PH_mod = pk
PHC1 - first order phase correction value for PH_mod = pk
FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)
REVERSE - flag indicating to reverse the spectrum

set by a previous processing command, e.g. ***xtrf***, can be viewed with ***dpp*** :

SI - size of the processed data
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
TDeff - number of raw data points to be used for processing
TDoff - first point of the FID used for processing (default 0)

set by the acquisition, can be viewed with ***dpa*** or by typing ***2s td***, ***1s td*** :

TD - time domain; number of raw data points

F1 parameters

set by a previous processing command, e.g. ***xtrf***, can be viewed with ***dpp*** :

MC2 - Fourier transform mode

OUTPUT PARAMETERS

F2 parameters

can be viewed with ***dpp*** or by typing ***2s ymax_p***, ***2s ymin_p*** etc.:

YMAX_p - maximum intensity of the processed data

YMIN_p - minimum intensity of the processed data
S_DEV - standard deviation of the processed data
NC_proc - intensity scaling factor

can only be viewed by typing ***2s bytordp***:

BYTORDP - byte order of the processed data

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`proc` - F2 processing parameters
`proc2` - F1 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`procs` - F2 processing status parameters
`proc2s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

XTRFP

XTRFP2

XTRFP1

SEE ALSO

xtrf, xtrf2, xfb, xf2, xf1

zert1

NAME

zert1 - zero a user defined region of each column (F1) of 2D data

DESCRIPTION

The command ***zert1*** sets the intensity of all 2D data points in a user defined region to zero. This region is defined as follows:

- only the columns between F2-ABSF2 and F2-ABSF1 are zeroed
- the part (region) of each column which is zeroed shifts from column to column. The first column is zeroed between F1-ABSF2 and F1-ABSF1. The last column is zeroed between F1-SIGF2 and F1-SIGF1. For intermediate columns, the low field limit is an interpolation of F1-ABSF2 and F1-SIGF2 and the high field limit is an interpolation of F1-ABSF1 and F1-SIGF1.

zert1 works exactly like ***abst1***, except that the data points are zeroed instead of baseline corrected.

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***absf1***, ***absf2*** etc.:

ABSF1 - low field limit that defines the first column to be zeroed
ABSF2 - high field limit that defines the last column to be zeroed

F1 parameters

set by the user with ***edp*** or by typing ***1 absf1***, ***1 absf2*** etc.:

ABSF1 - low field limit of the zero region in the first row
ABSF2 - high field limit of the zero region in the first row
SIGF1 - low field limit of the zero region in the last row
SIGF2 - high field limit of the zero region in the last row

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`p2r1`, `p2r2`, `n2r1`, `n2r2` - positive and negative projections
`proc2` - F1 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`proc2s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

ZERT1

SEE ALSO

zert2, abst1

zert2

NAME

zert2 - zero a user defined region of each row (F2) of 2D data

DESCRIPTION

The command ***zert2*** sets the intensity of all 2D data points in a user defined region to zero. This region is defined as follows:

- only the rows between F1-ABSF2 and F1-ABSF1 are zeroed
- the part (region) of each row which is zeroed shifts from row to row. The first row is zeroed between F2-ABSF2 and F2-ABSF1. The last row is zeroed between F2-SIGF2 and F2-SIGF1. For intermediate rows, the low field limit is an interpolation of F2-ABSF2 and F2-SIGF2 and the high field limit is an interpolation of F2-ABSF1 and F2-SIGF1.

zert2 works exactly like ***abst2***, except that the data points are zeroed instead of baseline corrected.

INPUT PARAMETERS

F1 parameters

set by the user with ***edp*** or by typing ***1 absf1***, ***1 absf2*** etc.:

ABSF1 - low field limit that defines the first row to be zeroed
ABSF2 - high field limit that defines the last row to be zeroed

F2 parameters

set by the user with ***edp*** or by typing ***absf1***, ***absf2*** etc.:

ABSF1 - low field limit of the zero region in the first row
ABSF2 - high field limit of the zero region in the first row
SIGF1 - low field limit of the zero region in the last row
SIGF2 - high field limit of the zero region in the last row

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

`proc` - F2 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`p2r1`, `p2r2`, `n2r1`, `n2r2` - positive and negative projections
`procs` - F2 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

ZERT2

SEE ALSO

zert1, abst2

Chapter 5

3D processing commands

This chapter describes all XWIN-NMR 3D processing commands. They only work on 3D data and store their output in processed data files. 3D raw data are never overwritten.

We will often refer to the three dimensions of a 3D dataset as the F3, F2 and F1 dimension. F3 is always the acquisition dimension. For processed data, F2 and F1 are always the second and third dimension, respectively. For raw data, this order can be the same or reversed as expressed by the acquisition status parameter AQSEQ. 3D processing commands which work on raw data automatically determine their storage order from AQSEQ.

The name of a 3D processing command expresses the dimension in which it works, e.g. ***tf3*** works in F3, ***tf2*** in F2 and ***tf1*** in the F1 dimension. The command ***r12*** reads an F1-F2 plane, ***r13*** an F1-F3 plane etc.

For each command, the relevant input and output parameters are mentioned.

Furthermore, the relevant input and output files and their location are mentioned. Although file handling is completely transparent, it is sometimes useful to know which files are involved and where they reside. For example, if you have permission problems or if you want to process or interpret your data with third party software.

dosy3d

NAME

dosy3d - process a 3D DOSY dataset

DESCRIPTION

The command ***dosy3d*** processes a 3D DOSY dataset.

DOSY is a special representation of diffusion measurements. Instead of generating just numbers using the T1/T2 fitting package (i.e. diffusion coefficients and error values), the DOSY processing gives pseudo 3D data where the F2 or F1 axis displays diffusion constants rather than NMR frequencies.

At the time of this writing, only exponential fitting (up to 3 exponentials) is supported.

For more information on ***dosy3d***, refer to the manual "DOSY and Diffusion" under ***Help → Other topics***.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`difflist` - list of gradient amplitudes in Gauss/cm

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - 3D data which are processed in F3 and F2 or in F3 and F1
`dosy` - DOSY processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - 2D processed data
`auditp.txt` - processing audit trail

SEE ALSO

eddosy, dosy2d

tabs3

NAME

tabs3 - automatic baseline correction in the F3 dimension

DESCRIPTION

The command ***tabs3*** performs an automatic baseline correction in the F3 dimension, by subtracting a polynomial. The degree of the polynomial is determined by the F3 parameter ABSG which has a value between 0 and 5, with a default of 5. ***tabs3*** works like ***absf*** in 1D and ***abs2*** in 2D. This means that it only corrects a certain spectral region which is determined by the parameters ABSF1 and ABSF2.

INPUT PARAMETERS

F3 parameters

set by the user with ***edp*** or by typing ***absg***, ***absf1*** etc.:

ABSG - degree of the polynomial to be subtracted (0 to 5, default of 5)
ABSF1- low field limit of the correction region
ABSF2 - high field limit of the correction region

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`proc` - F3 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`procs` - F3 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TABS3

SEE ALSO

tabs2, tabs1, tf3

tabs2

NAME

tabs2 - automatic baseline correction in the F2 dimension

DESCRIPTION

The command ***tabs2*** performs an automatic baseline correction in the F2 dimension, by subtracting a polynomial. The degree of the polynomial is determined by the F2 parameter ABSG which has a value between 0 and 5, with a default of 5. ***tabs2*** works like ***absf*** in 1D and ***abs2*** in 2D. This means that it only corrects a certain spectral region which is determined by the parameters ABSF1 and ABSF2.

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***2 absg***, ***2 absf1*** etc.:

ABSG - degree of the polynomial to be subtracted (0 to 5, default is 5)
ABSF1- low field limit of the correction region
ABSF2 - high field limit of the correction region

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`proc2` - F2 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`proc2s` - F2 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TABS2

SEE ALSO

tabs3, tabs1, tf2

tabs1

NAME

tabs1 - automatic baseline correction in the F1 dimension

DESCRIPTION

The command ***tabs1*** performs an automatic baseline correction in the F1 dimension, by subtracting a polynomial. The degree of the polynomial is determined by the F1 parameter ABSG which has a value between 0 and 5, with a default of 5. ***tabs1*** works like ***absf*** in 1D and ***abs1*** in 2D. This means that it only corrects a certain spectral region which is determined by the parameters ABSF1 and ABSF2.

INPUT PARAMETERS

F1 parameters

set by the user with ***edp*** or by typing ***1 absg***, ***1 absf1*** etc.:

ABSG - degree of the polynomial to be subtracted (0 to 5, default is 5)
ABSF1- low field limit of the correction region
ABSF2 - high field limit of the correction region

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`proc3` - F1 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`proc3s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TABS1

SEE ALSO

tabs3, tabs2, tf1

tf3

NAME

tf3 - process 3D data in the F3 dimension

DESCRIPTION

The command ***tf3*** processes a 3D dataset in the F3 dimension. F3 is the first dimension of a 3D dataset, i.e. the acquisition dimension. ***tf3*** always performs a Fourier transform which transforms time domain data (FID) into frequency domain data (spectrum). Depending on the processing parameters BC_mod, WDW, ME_mod and PH_mod, it also performs baseline correction, window multiplication, linear prediction and spectrum phase correction.

The processing steps done by ***tf3*** can be described as follows:

1. Baseline correction of the F3 time domain data
   Each row is baseline corrected according to BC_mod. This parameter takes the value *no*, *single*, *quad*, *spol*, *qpol sfil* or *qfil*. More details on BC_mod can be found in chapter 2.4.
2. Linear prediction of the F3 time domain data
   Linear prediction is done according to ME_mod. This parameter takes the value *no*, *LPfr*, *LPfc*, *LPbr* or *LPbc*. Usually, ME_mod = no, which means no prediction is done. Forward prediction (*LPfr* or *LPfc*) can, for example, be used to extend truncated FIDs. Backward prediction (*LPbr* or *LPbc*) can be used to improve the initial data points of the FID. Linear prediction is only performed if NCOEF > 0. Furthermore, the parameters LPBIN and, for backward prediction, TDoff play a role (see these parameters in chapter 2.4).
3. Window multiplication of the F3 time domain data
   Each row is multiplied with a window function according to WDW. This parameter takes the value *em*, *gm*, *sine*, *qsine*, *trap*, *user*, *sinc*, *qsinc*, *traf* or *trafs*. More details on WDW can be found in chapter 2.4.
4. Fourier transform of the F3 time domain data
   Each row is Fourier transformed according to the acquisition status parameter AQ_mod as shown in table 5.1. ***tf3*** does not evaluate the processing parameter FT_mod! However, it stores the Fourier transform mode in the processing status parameter FT_mod.

AQ_mod	Fourier transform mode	status FT_mod
qf	forward, single, real	fsr
qsim	forward, quad, complex	fqc
qseq	forward, quad, real	fqr
DQD	forward, quad, complex	fqc

Table 5.1

5. Phase correction of the F3 frequency domain data
   Each row is phase corrected according to PH_mod. This parameter takes the value *no*, *pk*, *mc* or *ps*. For PH_mod = pk, ***tf3*** applies the values of PHC0 and PHC1. This is only useful if the phase values are known. You can determine them by typing ***xfb*** on the 3D data to process a 23 or 13 plane and do a phase correction on the resulting the 2D dataset. Then you go back to the 3D dataset and set the F3 parameters PHC0 and PHC1 (with ***edp***) to the values calculated on the 2D data. More details on PH_mod can be found in chapter 2.4.

The size of the processed data is determined by the processing parameter SI; SI real and SI imaginary points are created. A typical value for SI is TD/2 in which case, all raw data points are used and no zero filling is done. In fact, several parameters control the number of input and output data points, for example:

1. SI > TD/2: the raw data are zero filled before the Fourier transform
2. SI < TD/2: only the first 2*SI raw data points are used
3. 0 < TDeff < TD: only the first TDeff raw data points are used
4. 0 < TDoff < TD: the first TDoff raw data points are cut off and TDoff zeroes are appended at the end
5. TDoff < 0: -TDoff zeroes are prepended at the beginning. Note that:
   - for SI < (TD-TDoff)/2 raw data are cut off at the end
   - for DIGMOD=digital, the zeroes would be prepended to the group delay which does not make sense. You can avoid that by converting the raw data with ***convdta*** before you process them.
6. 0 < STSR < SI: only the processed data between STSR and STSR+STSI are stored (if STSI = 0, STSR is ignored and SI points are stored)
7. 0 < STSI < SI: only the processed data between STSR and STSR+STSI

are stored.

Note that only in the first case the processed data contain the total information of the raw data. In all other cases, information is lost.

Before you run ***tf3***, you must set the processing parameter SI in all three dimensions F3, F2 and F1. The command ***tf2*** does not evaluate the F2 processing parameter SI, it evaluates the processing status parameter SI as it was set by ***tf3***.

tf3 evaluates the parameter FCOR. The first point of the FIDs is multiplied with the value of FCOR which lies between 0.0 and 2.0. For digitally filtered Avance data, FCOR has no effect in F3 because the first point is part of the group delay and, as such, is zero. In that case, it only plays a role in the F2 and F1 dimension (see ***tf2*** and ***tf1***). However, on A*X data or Avance data measured with DIGMOD = analog, there is no group delay and FCOR also plays a role in F3.

tf3 evaluates the F3 parameter PKNL. On A*X spectrometers, PKNL = true causes a non linear 5th order phase correction of the raw data. This corrects possible errors caused by non linear behaviour of the analog filters. On Avance spectrometers, PKNL must always be set to TRUE. For digitally filtered data, it causes ***tf3*** to handle the group delay of the FID. For analog data it has no effect.

tf3 evaluates the F3 parameter REVERSE. If REVERSE = TRUE, the spectrum will be reversed in F3, i.e. the first data point becomes the last and the last data point becomes the first.

tf3 can be used with the following command line options:

n

tf3 will not store the imaginary data. Without this option, ***tf3*** will ask you whether or not to store the imaginary data. Imaginary data are only needed for phase correction. If the phase values are already known and PHC0 and PHC1 are set accordingly, ***tf3*** will phase correct the spectrum and there is no need to store the imaginary data. This will save processing time and disk space. If you still need to do a phase correction after ***tf3***, you can create imaginary data from the real data with a Hilbert transform (see ***tht3***)

a

tf3 will store the imaginary data if there is enough disk space available. If not, it will only store the real data.

c

this option allows you to set the subcube sizes to predefined values. ***tf3*** will prompt you for the subcube size in each dimension. Processed 3D data are stored in the so-called subcube format whenever the data are larger than the computer memory. Without the option ***c***, subcubes are made as large possible and the calculated subcube sizes are stored in the processing status parameter XDIM (type ***dpp***). If, however, you want to read the processed data with third party software which can not interpret this parameter, you can use the ***c*** option to predefine the subcube sizes.

big/little

tf3 stores the data in the data storage order of the computer it runs on, e.g. big endian on SGI UNIX workstations and little endian on Windows PCs. The storage order is stored in the processing status parameter BYTORDP (type ***2s bytordp***). If, however, you want to read the processed data with third party software which can not interpret this parameter, you can use the ***big/little*** option to predefine the storage order.

p<du>

the option p allows you to store the processed data on a different disk unit than the current data disk unit (***edc*** parameter DU). Examples of <DU> are:

u, x, par2, usr/people/guest (under UNIX)
D:, E:\x, w, xwdat\spect (under Windows)

If the specified directory does not exist, it will be created.

Normally, ***tf3*** stores the entire spectral region as determined by the spectral width. However, you can do a so-called strip transform which means that only a certain region of the spectrum is stored. This can be done by setting the parameters STSR and STSI which represent the strip start and strip size, respectively. They both can take a value between 0 and SI. The value which are actually used can be a little different. STSI is always rounded to the next higher multiple of 16. Furthermore, when the data are stored in subcube format (see below), STSI is rounded to the next multiple of the subcube size. Type ***dpp*** to check this; if XDIM is smaller than SI, then the data are stored in subcube format and STSI is a multiple of XDIM.

Depending on size of the processed data and the available computer memory, ***tf3*** stores the data in sequential or subcube format. Sequential format is used

when the entire dataset fits in memory. Subcube format is used this is not the case. ***tf3*** automatically calculates the subcube sizes such that one row (F3) of subcubes fits in the available memory. Furthermore, one column (F2) and one tube (F1) of subcubes must fit in the available memory. The calculated subcube sizes are stored in the processing status parameter XDIM (type ***dpp***). The alignment of the data points for sequential and subcube format is the extension of the alignment in a 2D dataset as it is shown in table 4.6 and 4.7. The storage handling is completely transparent to the user and is only of interest when the data are interpreted by third party software.

INPUT PARAMETERS

F3, F2 and F1 parameters

set by the user with ***edp*** or by typing ***si***, ***stsr***, ***2 si***, ***1 si*** etc.:

SI - size of the processed data
STSR - strip start: first output point of strip transform
STSI - number of output points of strip transform
TDeff - number of raw data points to be used for processing
TDoff - first point of the FID used for processing (default 0)

F3 parameters

set by the user with ***edp*** or by typing ***bc_mod***, ***bcfw*** etc.:

BC_mod - FID baseline correction mode
- BCFW - filter width for BC_mod = sfil or qfil
- COROFFS - correction offset for BC_mod = spol/qpol or sfil/qfil

ME_mod - FID linear prediction mode
- NCOEF - number of linear prediction coefficients
- LPBIN - number of points for linear prediction
- TDoff - number of raw data points predicted for ME_mod = LPb*

WDW - FID window multiplication mode
- LB - Lorentzian broadening factor for WDW = em or gm
- GB - Gaussian broadening factor for WDW = gm, sinc or qsinc
- SSB - Sine bell shift for WDW = sine, qsine, sinc or qsinc
- TM1, TM2 - limits of the trapezoidal window

PH_mod - phase correction mode
- PHC0 - zero order phase correction value for PH_mod = pk
- PHC1 - first order phase correction value for PH_mod = pk

FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)

REVERSE - flag indicating to reverse the spectrum
PKNL - group delay handling (Avance) or filter correction (A*X)

set by the acquisition, can be viewed with ***dpa*** or ***3s aq_mod*** etc.:

AQ_mod - acquisition mode (determines the status FT_mod)
AQSEQ - acquisition sequence (3-2-1 or 3-1-2)
TD - time domain; number of raw data points

F2 and F1 parameters

set by the acquisition, can be viewed with ***dpa*** or by typing ***2s fnmode*** etc.:

FnMODE - Fourier transform mode

OUTPUT PARAMETERS

F3 parameters

can be viewed with ***dpp*** or by typing ***3s si***, ***3s tdeff*** etc.:

SI - size of the processed data
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
FTSIZE - Fourier transform size
FT_mod - Fourier transform mode
YMAX_p - maximum intensity of the processed data
YMIN_p - minimum intensity of the processed data
S_DEV - standard deviation of the processed data
NC_proc - intensity scaling factor

can only be viewed by typing ***3s bytordp***:

BYTORDP - byte order of the processed data

F3, F2 and F1

can be viewed with ***dpp*** or by typing ***3s si***, ***2s si***, ***1s si*** etc.:

SI - size of the processed data
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
TDeff - number of raw data points that were used for processing
TDoff - first point of the FID used for processing (default 0)
XDIM - subcube size

F2 and F1 parameters

can be viewed with ***dpp*** or by typing ***2s mc2***, ***1s mc2*** etc.:

MC2 - Fourier transform mode

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`ser` - raw data
`acqus` - F3 acquisition status parameters
`acqu2s` - F2 acquisition status parameters
`acqu3s` - F1 acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`proc` - F3 processing parameters
`proc2` - F2 processing parameters
`proc3` - F1 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`3irr` - real/imaginary processed data (for FnMODE ≠ QF)
`3iii` - real/imaginary processed data (for FnMODE = QF)
`procs` - F3 processing status parameters
`proc2s` - F2 processing status parameters
`proc3s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TF3(store_imag, partition)

where *store_image* can be *y* or *n* and *partition* must be something like:
u, *x* or *usr/xwin* under UNIX
D: or *E:\tmp* under Windows NT

SEE ALSO

tf2, tf1, xf2, xfb, tf3p, tabs3, tht3

tf2

NAME

tf2 - process 3D data in the F2 dimension

DESCRIPTION

The command ***tf2*** processes a 3D dataset in the F2 dimension. This involves a Fourier transform which transforms time domain data (FID) into frequency domain data (spectrum). Depending on the processing parameters BC_mod, WDW, ME_mod and PH_mod, ***tf2*** also performs baseline correction, window multiplication, linear prediction and spectrum phase correction.

The processing steps done by ***tf2*** can be described as follows:

tf2 only works on data which have already been processed with ***tf3***. It performs the following processing steps in the F2 dimension:

1. Baseline correction of the F2 time domain data
   Each column is baseline corrected according to BC_mod. This parameter takes the value *no*, *single*, *quad*, *spol*, *qpol sfil* or *qfil*. More details on BC_mod can be found in chapter 2.4.
2. Linear prediction of the F2 time domain data
   Linear prediction is done according to ME_mod. This parameter takes the value *no*, *LPfr*, *LPfc*, *LPbr* or *LPbc*. Usually, ME_mod = no, which means no prediction is done. Forward prediction in F2 (*LPfr* or *LPfc*) can, for example, be used to extend truncated FIDs. Backward prediction (*LPbr* or *LPbc*) is not used very often in F2. Linear prediction is only performed for NCOEF > 0. Furthermore, LPBIN and, for backward prediction, TDoff play a role (see these parameters in chapter 2.4).
3. Window multiplication of the F2 time domain data
   Each column is multiplied with a window function according to WDW. This parameter takes the value *em*, *gm*, *sine*, *qsine*, *trap*, *user*, *sinc*, *qsinc*, *traf* or *trafs*. More details on WDW can be found in chapter 2.4.
4. Fourier transform of the F2 time domain data
   Each column is Fourier transformed according to the F2 processing status parameter MC2 as shown in table 5.2. ***tf2*** does <u>not</u> evaluate the processing parameter FT_mod! However, it stores the Fourier transform mode as it was evaluated from MC2 in the processing <u>status</u> parameter

F2 status MC2	Fourier transform mode	status FT_mod
QF	forward, quad, real	fqc
QSEQ	forward, quad, real	fqr
TPPI	forward, single, real	fsr
States	forward, quad, complex	fqc
States-TPPI	forward, single, complex	fsc
Echo-AntiEcho	forward, quad, complex	fqc

Table 5.2

FT_mod (type ***dpp***). Note that status MC2 is set by ***tf3*** to the value of the acquisition status parameter FnMODE.

5. Phase correction of the F2 frequency domain data.
Each column is phase corrected according to PH_mod. This parameter takes the value *no*, *pk*, *mc* or *ps*. For PH_mod = pk, ***tf2*** applies the values of PHC0 and PHC1. This is only useful if the phase values are known. You can determine them by typing ***xfb*** on the 3D data to process a 23 or 12 plane and do a phase correction on the resulting the 2D dataset. Then you go back to the 3D dataset and set the F2 parameters PHC0 and PHC1 (with ***edp***) to the values calculated on the 2D data. More details on PH_mod can be found in chapter 2.4.

The F2 processing parameter SI determines the size of the processed data in the F2 dimension. This must, however, be set before ***tf3*** is done and cannot be changed after ***tf3***. See ***tf3*** for the role of TD, TDeff and TDoff.

tf2 can do a strip transform according to the F2 parameters STSR and STSI (see ***tf3***).

tf2 evaluates the F2 parameter FCOR. The first point of the FIDs is multiplied with the value of FCOR which is a value between 0.0 and 2.0. As such, FCOR allows you to control the DC offset of the spectrum.

tf2 evaluates the F2 parameter REVERSE. If REVERSE = TRUE, the spectrum will be reversed in F2, i.e. the first data point becomes the last and the last data point becomes the first.

tf2 evaluates the F2 status parameter MC2. For MC2 ≠ QF, ***tf2*** uses the file `3rrr` as input and the files `3rrr` and `3rir` as output. For MC2 = QF, ***tf2*** uses

the files `3rrr` and `3iii` as input and output. The role of MC2 is described in detail for the 2D processing command ***xfb***.

tf2 can be used with the following command line options:

n

tf2 will not store the imaginary data (see ***tf3***)

a

tf2 will store the imaginary data if there is enough disk space available. If not, it will only store the real data.

The F1 Fourier transform mode and data storage mode depends on the F1 acquisition mode (see INPUT PARAMETERS below and the description of ***xfb***).

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***2 bc_mod***, ***2 bcfw*** etc.:

BC_mod - FID baseline correction mode
- BCFW - filter width for BC_mod = sfil or qfil
- COROFFS - correction offset for BC_mod = spol/qpol or sfil/qfil

ME_mod - FID linear prediction mode
- NCOEF - number of linear prediction coefficients
- LPBIN - number of points for linear prediction
- TDoff - number of raw data points predicted for ME_mod = LPb*

WDW - FID window multiplication mode
- LB - Lorentzian broadening factor for WDW = em or gm
- GB - Gaussian broadening factor for WDW = gm, sinc or qsinc
- SSB - Sine bell shift for WDW = sine, qsine, sinc or qsinc
- TM1, TM2 - limits of the trapezoidal window

PH_mod - phase correction mode
- PHC0 - zero order phase correction value for PH_mod = pk
- PHC1 - first order phase correction value for PH_mod = pk

FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)

REVERSE - flag indicating to reverse the spectrum

F3, F2 and F1 parameters

set by ***tf3***, can be viewed with ***dpp*** or by typing ***3s si***, ***2s si*** etc.:

SI - size of the processed data
STSR - strip start: first output point of strip transform
STSI - strip size: number of output points of strip transform
TDeff - number of raw data points to be used for processing
TDoff - first point of the FID used for processing (default 0)

F2 parameters

set by the ***tf3***, can be viewed with ***dpp*** or by typing ***2s mc2*** :

MC2 - Fourier transform mode

F1 parameters

set by the acquisition, can be viewed with ***dpa*** or by typing ***1s td*** etc.:

TD - time domain; number of raw data points

OUTPUT PARAMETERS

F2 parameters

can be viewed with ***dpp*** or by typing ***2s ft_mod*** :

FT_mod - Fourier transform mode
FTSIZE - Fourier transform size

F3 parameters

can be viewed with ***dpp*** or by typing ***3s ymax_p***, ***3s ymin_p*** etc.:

YMAX_p - maximum intensity of the processed data
YMIN_p - minimum intensity of the processed data
S_DEV - standard deviation of the processed data
NC_proc - intensity scaling factor

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`acqu2s` - F2 acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - processed 3D data (Fourier transformed in F3)
`3iii` - real/imaginary processed data (if MC2 = QF)
`proc2` - F2 processing parameters

`procs`, `proc2s`, `proc3s` - F3, F2, F1 processing status parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`3rir` - real/imaginary data (if MC2 ≠ QF)
`3iii` - real/imaginary processed data (if MC2 = QF)
`procs` - F3 processing status parameters
`proc2s` - F2 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TF2(store_imag)
where *store_image* can be *y* or *n*

SEE ALSO

tf3, tf1, xf1, xfb, tf2p, tabs2, tht2

tf1

NAME

tf1 - process 3D data in the F1 dimension

DESCRIPTION

The command ***tf1*** processes a 3D dataset in the F1 dimension. This involves a Fourier transform which transforms time domain data (FID) into frequency domain data (spectrum). Depending on the processing parameters BC_mod, WDW, ME_mod and PH_mod, ***tf1*** also performs baseline correction, window multiplication, linear prediction and spectrum phase correction.

The processing steps done by ***tf1*** can be described as follows:

tf1 only works on data which have already been processed with ***tf3*** and possibly with ***tf2***. It performs the following processing steps:

1. Baseline correction of the F1 time domain data
   Each tube is baseline corrected according to BC_mod. This parameter takes the value *no*, *single*, *quad*, *spol*, *qpol sfil* or *qfil*. More details on BC_mod can be found in chapter 2.4.
2. Linear prediction of the F1 time domain data
   Linear prediction is done according to ME_mod. This parameter takes the value *no*, *LPfr*, *LPfc*, *LPbr* or *LPbc*. Usually, ME_mod = no, which means no prediction is done. Forward prediction in F1 (*LPfr* or *LPfc*) can, for example, be used to extend truncated FIDs. Backward prediction (*LPbr* or *LPbc*) is not used very often in F1. Linear prediction is only performed for NCOEF > 0. Furthermore, LPBIN and, for backward prediction, TDoff play a role (see these parameters in chapter 2.4).
3. Window multiplication of the F1 time domain data
   Each tube is multiplied with a window function according to WDW. This parameter takes the value *em*, *gm*, *sine*, *qsine*, *trap*, *user*, *sinc*, *qsinc*, *traf* or *trafs*. More details on WDW can be found in chapter 2.4.
4. Fourier transform of the F1 time domain data
   Each tube is Fourier transformed according to the F1 processing status parameter MC2 as shown in table 5.2. ***tf1*** does <u>not</u> evaluate the processing parameter FT_mod! However, it stores the Fourier transform mode as it was evaluated from MC2 in the processing <u>status</u> parameter

F1 MC2	Fourier transform mode	status FT_mod
QF	forward, quad, real	fqc
QSEQ	forward, quad, real	fqr
TPPI	forward, single, real	fsr
States	forward, quad, complex	fqc
States-TPPI	forward, single, complex	fsc
Echo-AntiEcho	forward, quad, complex	fqc

Table 5.3

FT_mod (type ***dpp***). Note that status MC2 is set by ***tf3*** to the value of the acquisition status parameter FnMODE.

5. Phase correction of the F1 frequency domain data.
Each column is phase corrected according to PH_mod. This parameter takes the value *no*, *pk*, *mc* or *ps*. For PH_mod = pk, ***tf1*** applies the values of PHC0 and PHC1. This is only useful if the phase values are known. You can determine them by typing ***xfb*** on the 3D data to process a 13 or 12 plane and do a phase correction on the resulting the 2D dataset. Then you go back to the 3D dataset and set the F1 parameters PHC0 and PHC1 (with ***edp***) to the values calculated on the 2D data. More details on PH_mod can be found in chapter 2.4.

The F1 processing parameter SI determines the size of the processed data in the F1 dimension. This must, however, be set before ***tf3*** is done and cannot be changed after ***tf3***. See ***tf3*** for the role of TD, TDeff and TDoff.

tf1 can do a strip transform according to the F1 parameters STSR and STSI (see ***tf3***).

tf1 evaluates the F1 parameter FCOR. The first point of the FIDs is multiplied with the value of FCOR which is a value between 0.0 and 2.0. As such, FCOR allows you to control the DC offset of the spectrum.

tf1 evaluates the F1 parameter REVERSE. If REVERSE=TRUE, the spectrum will be reversed in F1, i.e. the first data point becomes the last and the last data point becomes the first.

tf1 evaluates the F1 status parameter MC2. For MC2 ≠ QF, ***tf1*** uses the file `3rrr` as input and the files `3rrr` and `3rri` as output. For MC2 = QF, ***tf1*** uses

the files `3rrr` and `3iii` as input and output. The role of MC2 is described in detail for the 2D processing command ***xfb***.

tf1 can be used with the following command line options:

n

tf1 will not store the imaginary data (see ***tf3***)

a

tf1 will store the imaginary data if there is enough disk space available. If not, it will only store the real data.

INPUT PARAMETERS

F1 parameters

set by the user with ***edp*** or by typing ***1 bc_mod***, ***1 bcfw*** etc.:

BC_mod - FID baseline correction mode
- BCFW - filter width for BC_mod = sfil or qfil
- COROFFS - correction offset for BC_mod = spol/qpol or sfil/qfil

ME_mod - FID linear prediction mode
- NCOEF - number of linear prediction coefficients
- LPBIN - number of points for linear prediction
- TDoff - number of raw data points predicted for ME_mod = LPb*

WDW - FID window multiplication mode
- LB - Lorentzian broadening factor for WDW = em or gm
- GB - Gaussian broadening factor for WDW = gm, sinc or qsinc
- SSB - Sine bell shift for WDW = sine, qsine, sinc or qsinc
- TM1, TM2 - limits of the trapezoidal window

PH_mod - phase correction mode
- PHC0 - zero order phase correction value for PH_mod = pk
- PHC1 - first order phase correction value for PH_mod = pk

FCOR - first (FID) data point multiplication factor (0.0-2.0, default 0.5)
REVERSE - flag indicating to reverse the spectrum

F3, F2 and F1 parameters

set by ***tf3***, can be viewed with ***dpp*** or by typing ***3s si***, ***2s si*** etc.:

SI - size of the processed data
STSR - strip start: first output point of strip transform

STSI - strip size: number of output points of strip transform
TDeff - number of raw data points to be used for processing
TDoff - first point of the FID used for processing (default 0)

F1 parameters

set by the ***tf3***, can be viewed with ***dpp*** or by typing ***1s mc2*** :

MC2 - Fourier transform mode

OUTPUT PARAMETERS

F1 parameters

can be viewed with ***dpp*** or by typing ***1s ft_mod*** :

FT_mod - Fourier transform mode
FTSIZE - Fourier transform size

F3 parameters

can be viewed with ***dpp*** or by typing ***3s ymax_p*** etc.:

YMAX_p - maximum intensity of the processed data
YMIN_p - minimum intensity of the processed data
S_DEV - standard deviation of the processed data
NC_proc - intensity scaling factor

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`acqu2s` - F1 acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - processed 3D data (Fourier transformed in F1)
`3iii` - real/imaginary processed data (if MC2 = QF)
`proc3` - F1 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`3rir` - real/imaginary data (if MC2 ≠ QF)
`3iii` - real/imaginary processed data (if MC2 = QF)

`proc3s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TF1(store_imag)
where *store_image* can be *y* or *n*

SEE ALSO

tf3, tf2, xf1, xfb, tf1p, tabs1, tht1

tf3p

NAME

tf3p - phase correction in the F3 dimension

DESCRIPTION

The command ***tf3p*** performs a phase correction in the F3 dimension applying the values of PHC0 and PHC1. These values must first be determined, for example on a 2D plane. A plane can be extracted with ***r23*** or ***r13*** which will automatically change the XWIN-NMR display to the resulting 2D dataset. You can enter the 2D ***phase*** menu and determine the phase values by correcting the spectrum. After returning to the 3D dataset, you can set PHC0 and PHC1 (with ***edp***) to the values which you determined for the 2D plane.

tf3p can only be done:

- directly after ***tf3*** (not after ***tf2*** or ***tf1)***
- if the F3 imaginary data exist

The F3 imaginary data exist if they have been stored in the previous step, e.g. with ***tf3 y*** or ***tf3p y***. If they do not exist, you can create them from the real data with a Hilbert transform (command ***tht3***).

Phase correction in F3 is already done as a part of ***tf3*** if PH_mod = pk and PHC0 and PHC1 are set (see ***tf3***).

INPUT PARAMETERS

F3 parameters

set by the user with ***edp*** or by typing ***phc0***, ***phc1*** etc.

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

OUTPUT PARAMETERS

F3 parameters

can be viewed with ***dpp*** or by typing ***3s phc0***, ***3s phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)

PHC1 - first order phase correction value (frequency dependent)

can be viewed with ***edp*** or by typing ***phc0***, ***phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`3irr` - F3 imaginary processed data
`proc` - F3 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`3irr` - F3 imaginary processed data
`proc` - F3 processing parameters
`procs` - F3 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TF3P(store_imag)
where *store_image* can be *y* or *n*

SEE ALSO

tf2p, tf1p, tf3, xf2p, pk

tf2p

NAME

tf2p - phase correction in the F2 dimension

DESCRIPTION

The command ***tf2p*** performs a phase correction in the F2 dimension applying the values of PHC0 and PHC1. These values must first be determined, for example on a 2D plane. A plane can be extracted with ***r23*** or ***r12*** which will automatically change the XWIN-NMR display to the resulting 2D dataset. You can enter the 2D ***phase*** menu and determine the phase values by correcting the spectrum. After returning to the 3D dataset, you can set PHC0 and PHC1 (with ***edp***) to the values which you determined for the 2D plane.

tf2p can only be done:

- directly after ***tf2*** (not after ***tf3*** or ***tf1)***
- if the F2 imaginary data exist

The F2 imaginary data exist if they have been saved in the previous step, e.g. with ***tf2 y*** or ***tf2p y***. If they do not exist, you can create them from the real data with a Hilbert transform (command ***tht2***).

Phase correction in F2 is already done as a part of ***tf2*** if PH_mod = pk and PHC0 and PHC1 are set (see ***tf2***).

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***2 phc0***, ***2 phc1***etc.

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

OUTPUT PARAMETERS

F2 parameters

can be viewed with ***dpp*** or by typing ***2s phc0***, ***2s phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)

PHC1 - first order phase correction value (frequency dependent)

can be viewed with ***edp*** or by typing ***2 phc0***, ***2 phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`3rir` - F2 imaginary processed data
`proc2` - F2 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`3rir` - F2 imaginary processed data
`proc2` - F2 processing parameters
`proc2s` - F2 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TF2P(store_imag)
where *store_image* can be *y* or *n*

SEE ALSO

tf3p, tf1p, tf2, xf1p, pk

tf1p

NAME

tf1p - phase correction in the F1 dimension

DESCRIPTION

The command ***tf1p*** performs a phase correction in the F1 dimension applying the values of PHC0 and PHC1. These values must first be determined, for example on a 2D plane. A plane can be extracted with ***r13*** or ***r12*** which will automatically change the XWIN-NMR display to the resulting 2D dataset. You can enter the 2D ***phase*** menu and determine the phase values by correcting the spectrum. After returning to the 3D dataset, you can set PHC0 and PHC1 (with ***edp***) to the values which you determined for the 2D plane.

tf1p can only be done:

- directly after ***tf1*** (not after ***tf3*** or ***tf2)***
- if the F1 imaginary data exist

The F1 imaginary data exist if they have been saved in the previous step, e.g. with ***tf1 y*** or ***tf1p y***. If they do not exist, you can create them from the real data with a Hilbert transform (command ***tht1***).

Phase correction in F1 is already done as a part of ***tf1*** if PH_mod = pk and PHC0 and PHC1 are set (see ***tf1***).

INPUT PARAMETERS

F1 parameters

set by the user with ***edp*** or by typing ***1 phc0***, ***1 phc1*** etc.

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

OUTPUT PARAMETERS

F1 parameters

can be viewed with ***dpp*** or by typing ***1s phc0***, ***1s phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)

PHC1 - first order phase correction value (frequency dependent)

can be viewed with ***edp*** or by typing ***1 phc0***, ***1 phc1*** etc.:

PHC0 - zero order phase correction value (frequency independent)
PHC1 - first order phase correction value (frequency dependent)

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`3rri` - F1-imaginary processed data
`proc3` - F1 processing parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data
`3rri` - F1-imaginary processed data
`proc3` - F1 processing parameters
`proc3s` - F1 processing status parameters
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

TF1P(store_imag)
where *store_image* can be *y* or *n*

SEE ALSO

tf3p, tf2p, tf1, xf1p, pk

tht3

NAME

tht3 - Hilbert transform of 3D data in the F3 dimension

DESCRIPTION

The command ***tht3*** performs a Hilbert transform in the F3 dimension creating imaginary data from the real data. The resulting imaginary data can then be used for phase correction with ***tf3p***.

Normally, the imaginary data are created during Fourier transform with ***tf3***. If, however, the imaginary data are missing or do not match the real data and you want to do a phase correction, you can (re)create them with ***tht3***.

Imaginary data do not match the real data if the latter have been manipulated after the Fourier transform, for example by ***tabs3*** or third party software.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3irr` - F3 imaginary processed data
`auditp.txt` - processing audit trail

SEE ALSO

tht2, tht1, tf3, tf3p

tht2

NAME

tht2 - Hilbert transform of 3D data in the F2 dimension

DESCRIPTION

The command ***tht2*** performs a Hilbert transform in the F2 dimension creating imaginary data from the real data. The resulting imaginary data can then be used for phase correction with ***tf2p***.

Normally, the imaginary data are created during Fourier transform with ***tf2***. If, however, the imaginary data are missing or do not match the real data and you want to do a phase correction, you can (re)create them with ***tht2***.

Imaginary data do not match the real data if the latter have been manipulated after the Fourier transform, for example by ***tabs2*** or third party software.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rir` - F2 imaginary processed data
`auditp.txt` - processing audit trail

SEE ALSO

tht3, tht1, tf2, tf2p

tht1

NAME

tht1 - Hilbert transform of 3D data in the F1 dimension

DESCRIPTION

The command ***tht1*** performs a Hilbert transform in the F1 dimension creating imaginary data from the real data. The resulting imaginary data can then be used for phase correction with ***tf1p***.

Normally, the imaginary data are created during Fourier transform with ***tf1***. If, however, the imaginary data are missing or do not match the real data and you want to do a phase correction, you can (re)create them with ***tht1***.

Imaginary data do not match the real data if the latter have been manipulated after the Fourier transform, for example by ***tabs1*** or third party software.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rri` - F1-imaginary processed data
`auditp.txt` - processing audit trail

SEE ALSO

tht3, tht2, tf1, tf1p

r12

NAME

r12 - read an F1-F2 plane from 3D processed data and store it as 2D data

DESCRIPTION

The command ***r12*** reads an F1-F2 plane from a 3D dataset and stores it as a 2D dataset.

r12 takes three arguments and can be used as follows:

r12
prompts for the plane number and output expno and reads the plane

r12 <plane>
prompts for the output expno and reads the specified plane

r12 <plane> <expno>
reads the specified plane and stores it under the specified expno

r12 <plane> <expno> n
reads the specified plane and stores it under the specified expno. The imaginary data are not stored.

Actually, ***r12*** can take fourth argument, ***p<partition>*** which is the partition or drive on which the plane is stored, but this is rarely used (see also ***tf3***).

Table 5.4 shows how the processing state of the output 2D data relates to the processing state of the input 3D data. This table can be interpreted as follows:

`FID` - data have not been Fourier transformed (time domain data)
`real` - data have been Fourier transformed but imaginary data do not exist
`real+imag` - data have been Fourier transformed and imaginary data exist

Depending on the processing state, data can be further processed after ***r12*** with 2D processing commands like ***xf2***, ***xf1***, ***xf2p*** etc.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr`, `3irr`, `3rir`, `3rri`, `3iii` - processed 3D data

3D data processed with	3D input data			2D output data	
	F3	F2	F3	F2	F1
tf3	real+imag	FID	FID	FID	FID
tf3, tf2	real	real+imag	FID	real+imag	FID
tf3, tf2, tf1	real	real	real+imag	real	real+imag
tf3, tf1, tf2	real	real+imag	real	real+imag	real

Table 5.4 ***r12*** input/output data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

R12(plane, expno, imaginary, partition)
for example R12(64, 1, "n", "pD:")

SEE ALSO

r13, r23, r12d, r13d, r23d, r12p, r13p, r23p

r13

NAME

r13 - read an F1-F3 plane from 3D data and store it as 2D data

DESCRIPTION

The command ***r13*** reads an F1-F3 plane from a 3D dataset and stores it as a 2D dataset.

r13 takes 3 arguments and can be used as follows:

r13
prompts for the plane number and output expno and reads the plane

r13 <plane>
prompts for the output expno and reads the specified plane

r13 <plane> <expno>
reads the specified plane and stores it under the specified expno

r13 <plane> <expno> n
reads the specified plane and stores it under the specified expno. The imaginary data are not stored.

Actually, ***r13*** can take fourth argument, ***p<partition>*** which is the partition or drive on which the plane is stored, but this is rarely used (see also ***tf3***).

Table 5.5 shows how the processing state of the 2D data relates to the processing state of the 3D data when ***r13*** was done. This table can be interpreted as follows:

`FID` - data have not been Fourier transformed (time domain data)
`real` - data have been Fourier transformed but imaginary data do not exist
`real+imag` - data have been Fourier transformed and imaginary data exist

Depending on the processing state, data can be further processed after ***r13*** in with 2D processing commands like ***xf2***, ***xf1***, ***xf2p*** etc.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr`, `3irr`, `3rir`, `3rri`, `3iii` - processed 3D data

3D data processed with	3D input data			2D output data	
	F3	F2	F1	F2	F1
tf3	real+imag	FID	FID	real+imag	FID
tf3, tf2	real	real+imag	FID	real	FID
tf3, tf2, tf1	real	real	real+imag	real	real+imag
tf3, tf1, tf2	real	real+imag	real	real	real

Table 5.5 ***r13*** input/output data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

R13(plane, expno, imaginary, partition)
for example R13(64, 1, "n", "pD:")

SEE ALSO

r12, r23, r13d, r12d, r23d, r13p, r12p, r23p

r23

NAME

r23 - read an F2-F3 plane from 3D data and store it as 2D data

DESCRIPTION

The command ***r23*** reads an F2-F3 plane from a 3D dataset and stores it as a 2D dataset.

r23 takes 3 arguments and can be used as follows:

r23
prompts for the plane number and output expno and reads the plane

r23 <plane>
prompts for the output expno and reads the specified plane

r23 <plane> <expno>
reads the specified plane and stores it under the specified expno

r23 <plane> <expno> n
reads the specified plane and stores it under the specified expno. The imaginary data are not stored.

Actually, ***r23*** can take fourth argument, ***p<partition>*** which is the partition or drive on which the plane is stored, but this is rarely used (see also ***tf3***).

Table 5.6 shows how the processing state of the 2D data relates to the processing state of the 3D data when ***r23*** was done. This table can be interpreted as follows:

`FID` - data have not been Fourier transformed (time domain data)
`real` - data have been Fourier transformed but imaginary data do not exist
`real+imag` - data have been Fourier transformed and imaginary data exist

Depending on the processing state, data can be further processed after ***r23*** in with 2D processing commands like ***xf2***, ***xf1***, ***xf2p*** etc.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr`, `3irr`, `3rir`, `3rri`, `3iii` - processed 3D data

3D data processed with	3D input data			2D output data	
	F3	F2	F1	F2	F1
tf3	real+imag	FID	FID	real+imag	FID
tf3, tf2	real	real+imag	FID	real	real+imag
tf3, tf2, tf1	real	real	real+imag	real	real
tf3, tf1, tf2	real	real+imag	real	real	real+imag

Table 5.6 ***r23*** input/output data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

R23(plane, expno, imaginary, partition)
for example R23(64, 1, "n", "pD:")

SEE ALSO

r12, r13, r12d, r13d, r23d, r12p, r13p, r23p

r12d

NAME

r12d - read a diagonal F1=F2 plane from 3D data and store it as 2D data

DESCRIPTION

The command ***r12d*** reads the diagonal F1=F2 plane from a 3D dataset and stores it as a 2D dataset.

r12d takes one argument and can be used as follows:

r12d
prompts for the output expno

r12d <expno>
stores the plane under the specified expno

Actually, ***r12d*** can take second argument, ***p<partition>*** which is the partition on which the plane is stored, but this is rarely used (see also ***tf3***).

r12d only stores the real data.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/<procno>/

`2rr` - real processed 2D data

SEE ALSO

r13d, r23d, r12, r13, r23, r12p, r13p, r23p

r13d

NAME

r13d - read a diagonal F1=F3 plane from 3D data and store it as 2D data

DESCRIPTION

The command ***r13d*** reads the diagonal F1=F3 plane from a 3D dataset and stores it as a 2D dataset.

r13d takes one argument and can be used as follows:

r13d
prompts for the output expno

r13d <expno>
stores the plane under the specified expno

Actually, ***r13d*** can take second argument, ***p<partition>*** which is the partition on which the plane is stored, but this is rarely used (see also ***tf3***).

r13d only stores the real data.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/<procno>/

`2rr` - real processed 2D data

SEE ALSO

r12d, r23d, r12, r13, r23, r12p, r13p, r23p

r23d

NAME

r23d - read a diagonal F2=F3 plane from 3D data and store it as 2D data

DESCRIPTION

The command ***r23d*** reads the diagonal F2=F3 plane from a 3D dataset and stores it as a 2D dataset.

r23d takes one argument and can be used as follows:

r23d
prompts for the output expno

r23d <expno>
stores the plane under the specified expno

Actually, ***r23d*** can take second argument, ***p<partition>*** which is the partition on which the plane is stored, but this is rarely used (see also ***tf3***).

r23d only stores the real data.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/<procno>/

`2rr` - real processed 2D data

SEE ALSO

r12d, r13d, r12, r13, r23, r12p, r13p, r23p

r12p

NAME

r12p - read the F1-F2 positive projection from 3D data

DESCRIPTION

The command ***r12p*** calculates the positive F1-F2 projection from a 3D dataset and stores it as a 2D dataset.

r12p takes one argument and can be used as follows:

r12p
prompts for the output expno

r12p <expno>
stores the projection under the specified expno

Actually, ***r12p*** can take second argument, ***p<partition>*** which is the partition on which the plane is stored, but this is rarely used (see also ***tf3***).

r12p only stores the real data.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/<procno>/

`2rr` - real processed 2D data

SEE ALSO

r23p, r13p, r12, r13, r23, r12d, r13d, r23d

r13p

NAME

r13p - read the F1-F3 positive projection from 3D data

DESCRIPTION

The command ***r13p*** calculates the positive F1-F3 projection from a 3D dataset and stores it as a 2D dataset.

r23p takes one argument and can be used as follows:

r13p
prompts for the output expno

r13p <expno>
stores the projection under the specified expno

Actually, ***r13p*** can take second argument, ***p<partition>*** which is the partition on which the plane is stored, but this is rarely used (see also ***tf3***).

r13p only stores the real data.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/<procno>/

`2rr` - real processed 2D data

SEE ALSO

r12p, r23p, r12, r13, r23, r12d, r13d, r23d

r23p

NAME

r23p - read the F2-F3 positive projection from 3D data

DESCRIPTION

The command ***r23p*** calculates the positive F2-F3 projection from a 3D dataset and stores it as a 2D dataset.

r23p takes one argument and can be used as follows:

r23p
prompts for the output expno

r23p <expno>
stores the projection under the specified expno

Actually, ***r23p*** can take second argument, ***p<partition>*** which is the partition on which the plane is stored, but this is rarely used (see also ***tf3***).

r23p only stores the real data.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`3rrr` - real processed 3D data

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/<procno>/

`2rr` - real processed 2D data

SEE ALSO

r12p, r13p, r12, r13, r23, r12d, r13d, r23d

Chapter 6

Analysis and output commands

This chapter describes XWIN-NMR analysis and output commands. Although they do not really process (manipulate) the data, they are part of the processing part of XWIN-NMR. Some of them merely interpret the data and display their output, i.e. they do not change the dataset in any way. Others change parameters (like ***sref*** and ***sino***) or create new files (like ***setti*** and ***ppp***). None of them, however change the processed data.

autoplot

NAME

autoplot - plot the current data according to an XWIN-PLOT layout

DESCRIPTION

The command ***autoplot*** plots the current dataset according to an XWIN-PLOT layout. The layout must be specified with the ***edo*** parameter LAYOUT. This layout can be a standard XWIN-PLOT layout which is delivered with the NMR Suite program or one of your own layouts which you have set up from XWIN-PLOT.

XWIN-PLOT allows you to set up a dataset portfolio and store it under the name `portfolio.por` in the processed data directory (procno). If this file exists, ***autoplot*** will plot:

- the current (foreground) dataset
- the second, third etc. dataset defined in `portfolio.por`

according to the layout define in ***edo***. Note that autoplot always uses the current dataset and ignores the first dataset defined in `portfolio.por`.

In NMR Suite 3.1 and newer ***autoplot*** is used in various processing AU programs (like ***proc_1d***) instead of ***plot***. As such, ICON-NMR automation using XWIN-PLOT layouts.

Under LINUX, the commands ***plot*** and ***autoplot*** are equivalent. They both plot the current data according to an XWIN-PLOT layout.

INPUT FILES

<xwhome>/plot/layouts/*.xwp - Bruker library XWIN-PLOT layouts

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`procs` - processing status parameters
`intrng` - integral regions
`parm.txt` - ascii file containing parameters which appear on the plot
`title` - default title file
`outd` - output device parameters

`portfolio.por` - XWIN-PLOT portfolio (input file is it exists)

For a 2D dataset, the files `2rr`, `proc2s` and `level` are also input.

USAGE IN AU PROGRAMS

AUTOPLOT

SEE ALSO

xwinplot, xwp_lp, xwp_pp

edg, edgx, edgw

NAME

edg - edit plot parameters
edgx - edit extended plot parameters
edgw - edit plot parameters for a white washed stack plot

DESCRIPTION

The command ***edg*** allows you to set plot parameters for parameter oriented plotting in XWIN-NMR (commands ***view***/***plot***). A dialog box is opened which shows a list of plot objects, e.g. spectrum, title, axis, integrals, peaks. For each object, there are two buttons; one to select or deselect the object and one to edit the object. Only selected objects will appear on the plot. When you click one of the edit buttons, e.g. EDSPECT, a new dialog box will appear showing the parameters for that object.

Plot parameters can also be set by entering their names, in lowercase letters, on the command line. For example, entering ***f1p*** allows you to set the value of F1P, a plot parameter of the SPECT object. Entering ***tpos***, allows you to set the value of TPOS, a plot parameter of the TITLE object.

edgx works like ***edg*** except that it allows you to set the extended parameters which are used by the commands ***viewx*** and ***plotx***.

edgw works like ***edg*** except that it allows you to set the parameters for a (white washed) stack plot which are used by the commands ***vieww*** and ***plotw***. Stack plots can be made of a series of 1D experiments which are stored in a 2D dataset.

The alternative to parameter oriented plotting is XWIN-PLOT, a wysiwyg plot editor of the NMR Suite (XWIN-NMR commands ***xwinplot***/***autoplot***). For more information, please refer to the XWIN-PLOT online help.

For a full description of all ***edg**** parameters, please refer to the Complete processing manual which is available as XWIN-NMR online help.

The ***edg**** commands are not supported for XWIN-NMR under LINUX. Parameter oriented plotting is not used there. The ***plot*** command still exist but is executes the command ***autoplot***.

INPUT AND OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`meta` - plot parameters for ***edg***, ***plot*** and ***view***
`meta.ext` - extended plot parameters for ***edgx*** and ***edgw***

SEE ALSO

plot, plots, plotx, plotw, view, viewx, viewww, xwinplot, autoplot

edinfo

NAME

edinfo - setup sample information

DESCRIPTION

The command ***edinfo*** allows you to set up sample information which consists of predefined entries, like *name* and *sample*. This text will appear on a plot created with ***plot*** or ***plots*** [1].

By default, ***edinfo*** opens a dialog box with two entries; *name* and *sample*. After filling out these entries, you have to set the following plot parameters:

TITLE = yes
TITNAM = ../../info

You can do that with the command ***edg***. Note that this will replace the title which was possibly setup with ***setti***. If you want both the title and the predefined information to appear on the plot, you can do that with the AU program ***proc_1dinfo***.

edinfo uses a template which is defined in the file:

`<xwhome>/exp/stan/nmr/lists/info_item`

This file can be modified by the NMR Superuser from the Windows Explorer or from a UNIX shell. You can replace the entries *name* and *sample* and/or increase the number of entries up to 10.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`info` - sample information for plot

<xwhome>/exp/stan/nmr/lists/

`info_item` - template for sample information

1. In XWIN-PLOT, you can use the NMR Text object for this purpose.

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`info` - sample information

SEE ALSO

setti, edg

edlev

NAME

edlev - edit 2D contour levels

DESCRIPTION

The command ***edlev*** opens a dialog box in which you can set the contour levels of a 2D dataset. These are the levels which appear on a plot created with one of the ***plot**** commands. They also appear on the screen if the spectrum is displayed in contour mode (as opposed to intensity mode).

Changing a level can be done by putting the cursor in a field of the right column and enter a new number.

Deleting a level can be done by clicking its number (in the left column). If you enter the value zero for a certain level (in the right column), then this level and all higher (more positive) levels will be deleted.

Adding a level can be done by entering a value in the last field of the right column. The new level will be automatically sorted in.

Creating an equidistant sequence of levels can be done by entering an increment value in the INCR field. The lowest positive level will remain the same and all higher levels will be recalculated according to the increment. The same is done for negative levels if they exist.

Creating a geometric sequence of levels can be done by entering a multiplication factor in the FACT field. The lowest positive level will remain the same and all higher levels will be recalculated according to the multiplication factor. The same is done for negative levels if they exist.

Note that the number of levels can also be change from the XWIN-NMR menu by clicking the ***DefPlot*** button.

INPUT AND OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`level` - 2D contour levels

SEE ALSO

plot, view, limits2d, plotreg

flplot

NAME

flplot - flush (send to the printer) all suspended plots

DESCRIPTION

The command ***flplot*** flushes (sends to the printer) all suspended plots. Suspended plots are plots which have been created with the command ***plots***.

The ***flplot*** command is not supported for XWIN-NMR under LINUX. Parameter oriented plotting is not used there. The ***plot*** command still exist but is executes the command ***autoplot***.

Suspended plots can be removed with the command ***rmplot***.

SEE ALSO

plots, rmplot

int2d, int2dref

NAME

int2d - calculate 2D integrals
int2dref - calculate 2D integrals and prompt for a reference integral

SYNTAX

int2d [filename]
int2dref [filename]

DESCRIPTION

The command ***int2d*** calculates 2D integrals. Before you can apply this command, you must first determine the integral regions from the ***integrate*** menu. When you enter this menu, you are prompted for the name of a file in which the regions are to be stored. After determining the integral regions, this filename can be entered as an argument of the ***int2d*** command. If you omit the argument, ***int2d*** will prompt for the filename. Note that ***int2d*** does not show its result. The command ***li*** sends the result of ***int2d*** to the output device which is specified with ***edo***, for example to the screen.

int2dref works like ***int2d***, except that it prompts for a reference integral number and a reference integral value. The output of ***int2dref*** contains an additional column with integral values which are normalized against the reference.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data

<xwhome>/exp/stan/nmr/lists/roi/

`<filename>`

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`int2d` - ascii file containing the integral regions and integral values

SEE ALSO

li, lipp, lippf

levcalc

NAME

levcalc - calculate 2D contour levels

DESCRIPTION

The command ***levcalc*** calculates the 2D contour levels according to the following criteria:

- The number of positive levels is determined by the processing parameter NLEV. On a phase sensitive spectrum, an equal number of negative levels is calculated.
- The lowest level is set to:

  LEV0 * S_DEV

  where LEV0 must set by the user (type ***edp***) and S_DEV was set by the last processing command (type ***dpp***). For LEV0 = 0, the default value of 4 is used.
- The highest level is set to:

  (TOPLEV/100) * max(YMAX_p, abs(YMIN_p))

  where TOPLEV must be set between 0 and 100 (type ***edp***) and YMAX_p and YMIN_p were set by the last processing command. For TOPLEV = 0, the default value of 85 is used.

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***lev0***, ***toplev*** etc.:

LEV0 - lowest 2D contour level multiplication factor
TOPLEV - highest 2D contour level
NLEV - number of positive contour levels in a 2D spectrum

set by processing, can be viewed with ***dpp*** or by typing ***2s s_dev*** :

S_DEV - standard deviation of the processed data

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`proc` - processing parameters
`procs` - processing status parameters

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`level` - 2D contour levels

USAGE IN AU PROGRAMS

LEVCALC

SEE ALSO

edlev

li

NAME

li - list integrals of a 1D or 2D dataset

DESCRIPTION

The command ***li*** creates a list of integral regions and integral values within those regions. It can be used on a 1D or 2D dataset.

Before you can use ***li***, you must first determine the integral regions. On a 1D dataset, this can be done as a part of the automatic baseline correction (***abs***) or by interactive integration. On a 2D dataset, this must be done by interactive integration followed by the command ***int2d*** or ***int2dref***. ***li*** lists the integral regions of the entire spectrum, independent of the plot region.

On a 1D dataset, ***li*** evaluates the parameter INTSCL if the regions have been determined interactively. For $INTSCL \neq -1$, the current dataset is defined as reference dataset for integral scaling. For $INTSCL = -1$, the integrals of the current dataset are scaled relative to the reference dataset. As such, you can compare the areas of peaks in a series of experiments.

On a 1D dataset, ***li*** evaluates the parameter INTBC. For INTBC = yes, ***li*** performs an automatic baseline correction (slope and bias) of the integrals. This, however, is only done when the integral regions were determined with ***abs***, not if they were determined interactively.

li sends its output to a device specified by parameter CURPRIN which can be set with ***edo***. This can be a printer, a file, the Clipboard or the screen. Furthermore, the output is always stored in the file `int1d` in the processed data directory.

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***intscl***, ***intbc*** etc.:

INTSCL - scale 1D integrals relative to a reference dataset
INTBC - automatic baseline correction of integrals created by ***abs***

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`2rr` - real processed 2D data
`intrng` - 1D integral regions (created by ***abs*** or interactive integration)
`int2d` - ascii file containing the 2D integral regions and integral values

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`int1d` - ascii file containing the output of ***li***
`integrals.txt` - ascii file containing the output of ***li***

USAGE IN AU PROGRAMS

LI

SEE ALSO

lipp, lippf, abs, int2d, int2dref

limits2d

NAME

limits2d - set the 2D plot limits

DESCRIPTION

The command ***limits2d*** allows you to set the 2D plot limits and the number of contour levels. It prompts you for the following:

F1 low field limit (F1LO)
F1 high filed limit (F1HI)
F2 low field limit (F2LO)
F2 high filed limit (F2HI)

Then, it will ask you if you want to change the (number of) levels. If you answer ***yes***, you will be prompted for the number of positive contour levels. Finally, you will be asked if you want to display the contours. If you answer ***yes***, the 2D spectrum will be displayed in contour mode.

The default limits shown by ***limits2d*** are the limits of the 2D spectrum as it is currently displayed on the screen, both in values and units (Hz or ppm). ***limits2d*** will adjust the display to the values you enter and set the plot parameters accordingly. This concerns the parameters F2LO, F2HI, F1LO, F1HI (limits in Hz) and the corresponding parameters F2PLO, F2PHI, F1PLO, F1PHI (limits in ppm).

You can also set these plot parameters with the command ***edg*** or by entering them on the command line (e.g. ***f1lo***, ***f1plo***). Note, however, that this will not automatically adjust the display.

INPUT AND OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`meta` - plot parameters for ***plot*** and ***view***
`level` - 2D contour levels (input if you change the number of levels)

SEE ALSO

edg, edlev, plot, view, plotreg

lipp, lippf

NAME

lipp - list peaks and integrals within the plot region
lippf - list peaks and integrals of the entire spectrum

DESCRIPTION

The command ***lipp*** is a combination of ***li*** and ***pp*** and creates a list of peaks and integrals. Its output contains the integral regions, integral areas, peak positions and peak intensities.

lipp lists all peaks which lie within the plot region and within one of the integral regions whereas ***pp*** shows all peaks within the plot region. For integral regions determined with ***abs***, this is usually the same. However, for interactively determined integral regions which do not cover all peaks, ***lipp*** might show fewer peaks than ***pp***.

lipp sends its output to a device specified by parameter CURPRIN which can be set with ***edo***. This can be a printer, a file, the Clipboard or the screen. Furthermore, the output is always stored in the file `int1d` in the processed data directory.

lippf works like ***lipp*** except that it shows the integrals and peaks of the entire spectrum, not just the plot region.

INPUT PARAMETERS

set by the user with ***edg***, by typing ***f1***, ***f2***, ***f1p***, etc.:

F1 - low field (left) plot region in Hz
F2 - high field (right) plot region in Hz
F1P - low field (left) plot region in ppm
F2P - high field (right) plot region in ppm

set by the user with ***edp***, by typing ***pscal***, ***sreglst*** etc.:

PSCAL - determines the region with the reference peak for vertical scaling
SREGLST - name of the scaling region file used for PSCAL=sreg/psreg
ASSFAC - assign the highest or second highest peak as reference for scaling
ASSWID - region excluded from second highest peak search
INTBC - automatic baseline correction of integrals created by ***abs***

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`proc` - processing parameters
`meta` - plot parameters for ***plot*** and ***view***
`reg` - region with the reference peak when PSCAL = *ireg/pireg*

<xwhome>/exp/stan/nmr/lists/scl/

`<SREGLST>` - scaling region file for PSCAL = *sreg/psreg*

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`int1d` - ascii file containing the output of ***lipp*** or ***lippf***
`peaks` - peak list containing all peaks in the entire spectrum

USAGE IN AU PROGRAMS

LIPP

LIPPF

SEE ALSO

li, abs

lp

NAME

lp - print the parameters of the current dataset

DESCRIPTION

The command ***lp*** prints the parameters of the current dataset, including dataset, output device, acquisition, processing and plot parameters. ***lp*** is a combination of the commands ***lpc***, ***lpo***, ***lpa***, ***lpp*** and ***lpg***. Its output is sent to the current printer as specified with ***edo*** (parameter CURPRIN).

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`acqus` - acquisition status parameters

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`procs` - processing status parameters
`meta` - plot parameters for ***plot*** and ***view***
`outd` - output device parameters

<xwhome>/prog/curdir/<user>/

`curdat` - current data parameters

<xwhome>/exp/stan/nmr/form/curd.l/`normlp` - format file for ***lpc***
<xwhome>/exp/stan/nmr/form/outd.l/`normlp` - format file for ***lpo***
<xwhome>/exp/stan/nmr/form/acqu.l/`normlp` - format file for ***lpa***
<xwhome>/exp/stan/nmr/form/proc.l/`normlp` - format file for ***lpp***
<xwhome>/exp/stan/nmr/form/plot.l/`normlp` - format file for ***lpg***

USAGE IN AUPROGRAMS

LP

SEE ALSO

lpc, lpo, lpa, lpp, lpg, lpgx, lppl

lpa

NAME

lpa - print the acquisition status parameters

DESCRIPTION

The command ***lpa*** prints the acquisition status parameters of the current dataset. The acquisition status parameters are set by acquisition commands and represent the status of the raw data.

The output of ***lpa*** is sent to the current printer as specified with ***edo*** (parameter CURPRIN).

Acquisition status parameters can viewed on the screen with ***dpa***.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`acqus` - acquisition status parameters
`acqu2s` - acquisition parameters second dimension (2D and 3D data only)
`acqu3s` - acquisition parameters third dimension (3D data only)

<xwhome>/exp/stan/nmr/form/acqu.l/

`normlp` - format file for ***lpa***

USAGE IN AUPROGRAMS

LPA

SEE ALSO

dpa, lp, lpp, lpc, lpg, lpgx, lpo, lppl

lpc

NAME

lpc - print the current data path parameters

DESCRIPTION

The command ***lpc*** prints the current data path parameters, including NAME, EXPNO, PROCNO, DU and USER. These parameters represent the variable parts of the data path of the current dataset:

<DU>/data/<USER>/nmr/<NAME>/<EXPNO>/pdata/<PROCNO>/

The output is sent to the printer which is specified with ***edo*** (parameter CURPRIN).

Current data path parameters can be set with ***edc*** and viewed with ***dpc***.

INPUT FILES

<xwhome>/prog/curdir/<user>/

`curdat` - current data parameters

<xwhome>/exp/stan/nmr/form/curd.l/

`normlp` - format file for ***lpc***

USAGE IN AUPROGRAMS

LPC

SEE ALSO

edc, dpc, lp, lpa, lpp, lpg, lpgx, lpo, lppl

lpg

NAME

lpg - print the plot parameters

DESCRIPTION

The command ***lpg*** prints the plot parameters of the current dataset. Its output is sent to the current printer as specified with ***edo*** (parameter CURPRIN).

Plot parameters can be set up with ***edg*** and viewed on the screen with ***dpg***.

Note that the parameters printed by ***lpg*** are only used for the parameter oriented plotting (commands ***view**** and ***plot****), not by XWIN-PLOT.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`meta` - plot parameters for ***plot*** and ***view***

<xwhome>/exp/stan/nmr/form/plot.l/

`normlp` - format file for ***lpg***

USAGE IN AUPROGRAMS

LPG

SEE ALSO

edg, dpg, lp, lpa, lpp, lpc, lpgx, lpo, lppl

lpgx

NAME

lpgx - print the extended plot parameters

DESCRIPTION

The command ***lpgx*** prints the extended plot parameters which are used for the commands ***viewx*** and ***plotx***. The output is sent to the current printer as specified with ***edo*** (parameter CURPRIN).

Extended plot parameters can be set up with ***edgx*** and viewed with ***dpgx***

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`meta.ext` - plot parameters

<xwhome>/exp/stan/nmr/form/plotx.l/

`normlp` - format file for ***lpgx***

USAGE IN AU PROGRAMS

LPGX

SEE ALSO

edgx, dpgx, lp, lpa, lpp, lpc, lpg, lpo, lppl

lpo

NAME

lpo - print the output device parameters

DESCRIPTION

The command ***lpo*** prints the output device parameters. The output is sent to the current printer as specified with ***edo*** (parameter CURPRIN).

The output device parameters can be set up with ***edo*** and viewed with ***dpo***.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`outd` - output device parameters

<xwhome>/exp/stan/nmr/form/outd.l/

`normlp` - format file for ***lpo***

USAGE IN AUPROGRAMS

LPO

SEE ALSO

edo, dpo, lp, lpa, lpp, lpc, lpg, lpgx, lppl

lpp

NAME

lpp - print the processing status parameters

DESCRIPTION

The command ***lpp*** prints the processing status parameters of the current dataset. The output is sent to the current printer as specified with ***edo*** (parameter CURPRIN).

Processing status parameters can viewed with ***dpp***.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`procs` - processing status parameters
`proc2s` - processing status parameters second dimension (2D and 3D only)
`proc3s` - processing status parameters third dimension (3D only)

<xwhome>/exp/stan/nmr/form/proc.l/

`normlp` - format file for ***lpg***

USAGE IN AUPROGRAMS

LPP

SEE ALSO

dpp, lp, lpa, lpp, lpc, lpgx, lpo, lppl

lppl

NAME

lppl - (re)create the file which contains the parameter list for the plot

DESCRIPTION

The command ***lppl*** (re)creates the parameter list which appears on a plot. Its output is an ascii files in the processed data directory of the current dataset. The list includes the acquisition, processing and plot parameters.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`format.temp` - format file for parameters which appear on the plot

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`parm.txt` - ascii file containing parameters which appear on the plot

If the file `format.temp` does not exist, it is created by ***lppl*** using the file `pulseprogram` from the same directory as input. If this does not exist either, it is also created by ***lppl*** from the pulse program which is defined by the acquisition parameter PULPROG. This pulse programs resides in:

<xwhome>/exp/stan/nmr/lists/pp

USAGE IN AU PROGRAMS

LPPL

SEE ALSO

plot, view, xwp_lp, lp, lpc, lpo, lpa, lpp, lpg, lpgx

plot, plots, plotw, plotx

NAME

plot - plot the current dataset
plots - plot suspend
plotw - plot a white washed stack plot of the current dataset
plotx - plot an extended plot of the current dataset

DESCRIPTION

The command ***plot*** plots the current dataset. Before you use ***plot***, you must set up the plot parameters with the command ***edg*** and specify the plotter with the command ***edo***. If you want a title to appear on the plot (plot parameter TITLE = yes), you must define the title with the command ***setti***. Finally, before you actually plot the spectrum, you can preview it on the screen with the command ***view***.

As an alternative to setting up the plot parameters with ***edg***, you can read the predefined plot parameters of a standard parameter set with the command:

```
rpar parameterset plot
```

plots works like ***plot***, except that it suspends the plot which means it is not send to the printer. The output of one or more ***plots*** commands can be sent to the printer with the ***flplot*** command. This allows you to put multiple plots, for example with expansions, on one sheet.

plotx works like ***plot***, except that it plots expansions of the integral regions as defined with ***edgx***.

plotw works like ***plot***, except that it plots a white washed stack plot of a 2D dataset, as defined with ***edgw***.

The setting of the printer or plotter in ***edo*** only counts for the current dataset. If you want to defined a printer or plotter for all datasets, you can do that in the User Interface. Enter the command ***setres*** and specify the printer name in the field ***Plotter***. The setting of the plotter in the User Interface overrules the setting in ***edo***.

For an extended description of the ***plot**** commands and plot parameters, please refer to the Complete processing manual which can be opened from the XWIN-NMR ***Help*** menu.

The command ***plot*** is the conventional, parameter oriented plotting spectra. It is used in many Bruker AU programs and several of which are used in the ICON-NMR automation. For interactive plotting, you can also use the wysiwyg XWIN-PLOT program. XWIN-PLOT can be started from the XWIN-NMR windows menu, by typing ***xwinplot*** on the command line or from the Desktop. The XWIN-NMR command ***autoplot*** command allows you to plot predefined XWIN-PLOT layouts and can be used in AU programs (AUTOPLOT).

The ***plots***, ***plotw*** and ***plotx*** commands are not supported for XWIN-NMR under LINUX. Parameter oriented plotting is not used there. The ***plot*** command still exist but is executes the command ***autoplot***.

INPUT PARAMETERS

all plot parameters

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`acqus` - acquisition status parameters (input if parameters are plotted)
`format.temp` - format file for parameters which appear on the plot (input if `parm.txt` does not exist)

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`1r` - processed 1D data
`meta` - plot parameters for ***plot*** and ***view***
`meta.ext` - extended plot parameters (input of ***plotx***)
`outd` - output device parameters (e.g. printer/plotter)
`title` - default title file (input if TITLE = yes)
`intrng` - integral regions (input if INTEGR = yes)
`parm.txt` - parameters (input if PARAM = yes)
`peaks` - peak list (input if it exists and PLABELS = yes)
`proc` - processing parameters
`procs` - processing status parameters
`reg` - region file (input of ***plotx*** if it exists and for ***plot*** if LIMITS=region)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`parm.txt` - parameters (output if it did not exist and PARAM = yes)
`peaks` - peak list (output if it did not exist and PLABELS = yes)
`reg` - copy of `intrng` file (output of ***`plotx`*** if it did not exists)

USAGE IN AU PROGRAMS

PLOT

PLOTS

PLOTW

PLOTX

SEE ALSO

edg, edgx, edgw, edo, setres, view, vieww, viewx, xwinplot, autoplot

plotreg

NAME

plotreg - display the current plot region

DESCRIPTION

The command ***plotreg*** sets the display to the current plot region.

On a 1D spectrum, the plot region is defined by the plot parameters:

F2 - high field limit in Hz
F1 - low field (left) limit in Hz
F2P - high field (left) limit in ppm
F1P - low field (left) limit in ppm

On a 2D spectrum, the plot region is defined by the plot parameters:

F2LO - F2 low field limit in Hz
F2HI - F2 high field limit in Hz

F1LO - F2 low field limit in Hz
F1HI - F2 high field limit in Hz

F2PLO - F2 low field limit in ppm
F2PHI - F2 high field limit in ppm

F1PLO - F1 low field limit in ppm
F1PHI - F1 high field limit in ppm

You can also set these plot parameters with the command ***edg*** or by entering them on the command line (e.g. ***f2***, ***f2lo***). Note, however, that this will not automatically adjust the display.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`meta` - plot parameters for ***plot*** and ***view***

SEE ALSO

edg, plot, view

pp, pps, pph, ppj, ppp

NAME

pp - peak picking: send peaks to a user defined output device
pps - as pp but show peaks on the screen
pph - as pp but also show an intensity histogram
ppp - as pp but store peaks in special format for mixed deconvolution
ppj - as pp but store peaks in JCAMP-DX format
ppt1 - as pp but store peaks and integral regions for relaxation analysis

DESCRIPTION

The command ***pp*** determines all peaks within the plot region. Its output looks like:

#	ADDRESS	FREQUENCY		INTENSITY
		[Hz]	[PPM]	
1	648.7	3698.825	7.3995	0.17
2	658.4	3687.649	7.3771	0.21

Table 6.1

pp sends its output to a device specified by parameter CURPRIN which can be set with ***edo***. This can be a printer, a file, the Clipboard or the screen. In the latter case, ***pp*** works exactly like ***pps*** (see below).

The peak list is created according to several criteria which are determined by various parameters. A data point is added to the peak list if:

- its intensity is higher than its two neighbouring points
- its relative intensity is smaller than MAXI
- its relative intensity is larger than MI
- its absolute intensity is larger than PC*noise
- it lies within the plot region determined by F2P and F1P

where MAXI, MI and PC are processing parameters and noise is calculated from the first 32th part of the spectrum.

The values of MI and MAXI must be chosen in relation to the plot parameter CY;

the intensity (in cm) of the reference peak. The reference peak is the highest peak in the spectrum or in a a certain part of it. The spectral region which contains reference peak, is determined by the parameter PSCAL. For PSCAL = global, this is entire spectrum. Table 6.2 shows all possible values of PSCAL and the corresponding regions.

PSCAL	Peak used as reference for vertical scaling
global	The highest peak of the entire spectrum.
preg	The highest peak within the plot region.
ireg	The highest peak within the regions specified in the `reg` file. If it does not exist, *global* is used.
pireg	as *ireg*, but the peak must also lie within the plot region.
sreg	The highest peak in the regions specified in scaling region file. This file is specified by the parameter SREGLST. If SREGLST is not set or it specifies a file which does not exist, *global* is used.
psreg	as *sreg* but the peak must also lie within the plot region.
noise	The intensity height of the noise of the spectrum.

Table 6.2

For PSCAL = ireg or pireg, the `reg` file is interpreted. The `reg` file can be created from the ***integrate*** menu and can be viewed or edited with the command ***`edmisc reg`***.

For PSCAL = sreg or psreg, the scaling region file is interpreted. This is used to exclude the solvent peak as reference. The name of a scaling region file is typically of the form NUCLEUS.SOLVENT, e.g. 1H.CDCl3. For most common nucleus/solvent combinations, a scaling region file is delivered with XWIN-NMR. They can be viewed or edited with ***`edlist scl`***. In several 1D standard parameter sets which are used during automation, PSCAL is set to *sreg* and SREGLIST to NUCLEUS.SOLVENT as defined by the parameters NUCLEUS and SOLVENT.

`pp` evaluates the parameter PSIGN which can take three possible value:

- pos - only positive peaks appear in the list
- neg - only negative peaks appear in the list
- both - both positive and negative peaks appear in the list

pps works like ***pp***, except that the peak list is always sent to the screen.

pph works like ***pp*** except that it also shows an intensity histogram. This allows you to get a quick overview over the intensity distribution.

ppj works like ***pp*** except that the peak list is stored in JCAMP-DX format in the file `pp.dx`. This file resides in the processed data directory and can be used for external programs which require JCAMP peak lists. As the file created by ***tojdx*** it contains the acquisition and processing parameters but instead of data points it contains a list of peaks. The last part of the file `pp.dx` looks like:

##NPOINTS= 4	
##PEAK TABLE= (XY..XY)	
2.3241	1.58
2.2962	1.18
1.9943	10.00
1.8725	1.36

Table 6.3

ppp works like ***pp***, except that the peak list is stored in the file `peaklist` which has the following format:

#frequency	half width	%gauss/100
3698.825	8.06	0.0
3687.649	8.06	0.0

Table 6.4

This file is used for mixed Lorentzian/Gaussian deconvolution (see ***mdcon***).

ppt1 creates a peak list for relaxation analysis. It works like ***pp*** but in addition stores the peaks in the file `baslpnts`. This file has the following format:

where the integer on the first line represents the number of points in the spectrum. Furthermore, ***ppt1*** determines the integral regions within the plot region and stores them in the file `intrng`. The files `baslpnts` and `intrng` can be viewed with ***edmisc***. They are used by the relaxation commands ***pd*** and ***pd0***.

```
16384
11295  2.324
11317  2.296
11556  1.994
11653  1.873
```

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***mi***, ***maxi*** etc.:

MI - minimum relative intensity (cm)
MAXI - maximum relative intensity (cm)
PC - peak picking sensitivity
PSIGN - peak sign (pos, neg, or both)
PSCAL - determines the region with the reference peak for vertical scaling
SREGLST - name of the scaling region file used for PSCAL = sreg/psreg
ASSFAC - assign the highest or second highest peak as reference for scaling
ASSWID - region excluded from second highest peak search

set by the user with ***edg*** or by typing ***f1***, ***f2*** etc.:

F1 - low field (left) plot region in Hz
F2 - high field (right) plot region in Hz
F1P - low field (left) plot region in ppm
F2P - high field (right) plot region in ppm

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`proc` - processing parameters
`reg` - region with the reference peak for PSCAL = *ireg* or *pireg*

<xwhome>/exp/stan/nmr/lists/scl/

`<SREGLST>` - regions containing the reference peak if PSCAL = *sreg*/*psreg*

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`peaks` - peak list containing all peaks in the entire spectrum

`peak.txt` - peak list created by pp and pps for XWIN-PLOT [1]
`peaklist` - peak list created by ***ppp*** for ***mdcon***
`pp.dx` - peak list in JCAMP-DX format created by ***ppj***
`baslpnts` - peaks positions calculated by ***ppt1***
`intrng` - integral regions calculated by ***ppt1***

USAGE IN AU PROGRAMS

PP

PPH

PPJ

PPP

Note that ***pps*** cannot be used in AU programs because it sends its output to the screen.

SEE ALSO

lipp, lippf, mdcon, xwp_pp

1. XWIN-NMR 3.1 and newer.

pp2d, pp2dmi

NAME

pp2d - AU program for peak picking on a 2D spectrum
pp2dmi - as pp2d with automatic calculation of MI

DESCRIPTION

The commands ***pp2d*** and ***pp2dmi*** perform peak picking on a 2D dataset. They calculate all peaks within the plot region according to the F2 parameters MI, MAXI, PC and PSIGN. Whereas ***pp2d*** interprets the parameter MI, ***pp2d*** automatically calculates the minimum intensity from the noise value.

pp2d and ***pp2dmi*** are actually AU programs which perform 1D peak picking on a partial projection of the 2D data. Before they can be used, they must be installed (with ***expinstall***) and compiled (with ***edau*** or ***xau***)). For more information, type ***edau pp2d*** and read the header of the AU program.

INPUT PARAMETERS

F2 parameters

set by the user with ***edp*** or by typing ***mi***, ***maxi*** etc.:

MI - minimum relative intensity (cm) (input for ***pp2d***)
MAXI - maximum relative intensity (cm)
PC - peak picking sensitivity
PSIGN - peak sign (pos, neg, or both)

set by the user with ***edg*** or by typing ***f1lo***, ***f1hi*** etc.:

F1LO - F1 low field plot region in Hz
F1HI - F1 high field plot region in Hz
F2LO - F2 low field plot region in Hz
F2HI - F2 high field plot region in Hz

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`2rr` - real processed 2D data
`proc` - F2 processing parameters
`meta` - plot parameters for ***plot*** and ***view***

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`f1projp` - ascii file containing the range of columns and the 1D data path

USAGE IN AU PROGRAMS

XCMD(xau "pp2d")

XCMD(xau "pp2dmi")

SEE ALSO

pp, pph, ppj, ppp, pps

rlut

NAME

rlut - read 2D lookup table

DESCRIPTION

The command ***rlut*** allows you to read a 2D lookup table. ***rlut*** takes one argument and can be used as follows:

rlut
shows a list of lookup tables. If you select one, it is read to the current 2D dataset.

rlut <lut>
reads the specified lookup table to the current 2D dataset

INPUT FILES

<xwhome>/exp/stan/nmr/lut/*

binary files which contain the 2D lookup tables

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

luta - 2D intensity lookup table

SEE ALSO

dellut

rmplot

NAME

rmplot - remove suspended plots which were queued by the command plots

DESCRIPTION

The command ***rmplot*** removes all suspended plots. These are plots which were created by the ***plots*** command and are normally send to the printer with the ***flplot*** command.

USAGE IN AU PROGRAMS

RMPLOT

SEE ALSO

plots, flplot

setti

NAME

setti - set up dataset title

DESCRIPTION

The command ***setti*** allows you to define the plot title which appears on plots created with ***plot*** or with XWIN-PLOT. It also appears in the XWIN-NMR window above the data field and on the plot preview (command ***view***).

By default, the plot title is stored in the file `title` which resides in the processed data directory. ***setti*** always opens the file `title` but you are free to store it under a different name. The parameter TITNAM determines which file is use by the ***plot*** command. By default, TITNAM = title.

setti uses the editor which is defined in the XWIN-NMR User Interface. If you want to use a different editor, type ***setres*** and modify the entry ***Editor***.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`title` - plot title

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`title` - plot title

SEE ALSO

edg, plot, view

sino

NAME

sino - calculate signal to noise ratio of a 1D spectrum

SYNTAX

sino [real] [noprint]

DESCRIPTION

The command ***sino*** calculates the signal to noise ratio of a 1D spectrum according to the formula:

$$SINO = \frac{maxval}{2 \cdot noise}$$

where *maxval* is highest intensity in the signal region. The signal region is determined by the parameters SIGF1 and SIGF2. If SIGF1 = SIGF2, the signal region is defined by:

- the entire spectrum minus the first 16th part (if the scaling region file is not defined)
- the regions defined in the scaling region file NUC1.SOLVENT where NUC1 and SOLVENT are acquisition status parameters.

Standard scaling region files can be installed with ***expinstall*** and can be edited with ***edlist scl***.

The factor *noise* is calculated according to the following algorithm:

$$noise = \sqrt{\frac{\sum_{i=-n}^{n} y(i)^2 - \frac{1}{N}\left(\left(\sum_{i=-n}^{n} y(i)\right)^2 + \frac{3 \cdot \left(\sum_{i=1}^{n} i(y(i) - y(-i))\right)^2}{N^2 - 1}\right)}{N - 1}}$$

where N is the total number of points in the noise region, n = (N-1)/2, and y(i) is the nth point in the noise region. The limits of the noise region is determined by the parameters NOISF1 and NOISF2. If they are equal, the first 1/16th of the spectrum is used as the noise region.

sino internally performs a peak picking to determine the highest peak in the signal region.

The signal region and the noise region can also be set interactively with the buttons ***sigreg*** and ***noisereg*** from the ***utilities*** menu.

sino first performs a magnitude calculation and then calculates the signal to noise ratio on the magnitude data. The command ***sino real*** skips the magnitude calculation and works on the real data.

The result of ***sino*** appears on the screen. If you don't want that, you can use the command ***sino noprint***. The *noprint* option is automatically set when ***sino*** is part of an AU program. The result of ***sino*** is also stored in the processing status parameter SINO which can be viewed with ***ls sino*** or ***dpp***.

The parameter SINO exists as processing parameter (***edp***) and as processing status parameter (***dpp***) and the two an different function. The latter is used to stored the result of the command ***sino*** as discussed above. The former can be used to specify a signal to noise ratio which must be reached in an acquisition (see the parameter SINO in chapter 2.4 and the AU program ***au_zgsino***).

INPUT PARAMETERS

set by the user with ***edp*** or by typing ***noisf1***, ***noisef2*** etc.:

NOISF1 - low field (left) limit of the noise region
NOISF2 - high field (right) limit of the noise region
SIGF1 - low field (left) limit of the signal region
SIGF2 - high field (right) limit of the signal region

set by the acquisition, can be viewed with ***dpa*** or by typing ***ls nuc1*** etc.:

NUC1 - observe nucleus
SOLVENT - sample solvent

OUTPUT PARAMETERS

can be viewed with ***dpp*** or by typing ***ls sino*** :

SINO - signal to noise ratio

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r` - real processed 1D data
`1i` - imaginary processed data (not used for ***sino real***)
`proc` - processing parameters

<xwhome>/exp/stan/nmr/lists/scl/

`<NUC1.SOLVENT>` - scaling region file

OUPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`procs` - processing status parameters

USAGE IN AU PROGRAMS

SINO

SEE ALSO

mc, abs

sref

NAME

sref - calibrate the spectrum; set the TMS signal to 0 ppm

DESCRIPTION

The command ***sref*** calibrates the spectrum by setting the TMS signal of a spectrum to exactly 0 ppm. It works on 1D and 2D spectra.

sref makes use of the lock table. This must be set up once after installing XWIN-NMR with the command ***edlock***.

On 1D spectra, ***sref*** involves three steps which are discussed below.

During the first step ***sref*** sets the value of the processing parameter SF according to the formula:

SF=BF1/(1.0+RShift * 1e-6)

where *RShift* is taken from the ***edlock*** table and BF1 is an acquisition status parameter. Changing SF automatically changes the processing parameters SR, the spectral reference, and OFFSET, the ppm value of the first data points, according to the following relations:

SR = SF - BF1

where BF1 is an acquisition status parameter

OFFSET = (SFO1/SF-1) * 1.0e6 + 0.5 * SW * SFO1/SF

where SW and SFO1 are acquisition status parameters

In fact, the relation for OFFSET depends on the acquisition mode. When the acquisition status parameter AQ_mod is *qsim*, *qseq* or *DQD*, which is usually the case, the above relation count. When AQ_mod is *qf*, the relation OFFSET = (SFO1/SF-1) * 1.0e6 is used.

sref then calculates which data point (between 0 and SI) in your spectrum corresponds to the ppm value *Ref.* from the ***edlock*** table. This data point will be used in the second step. The first step is independent of a reference substance.

During the second step, ***sref*** scans a region around the data point found in the first step for a peak. It will normally find the signal of the reference substance.

The width of the scanned region is defined by the parameter *Width* in ***edlock*** table, so this region is *Ref. +/- 0.5*Width* ppm. This step is necessary because the lock substance (solvent) will not always resonate at exactly the same position relative to the reference shift. The absolute chemical shift of the lock substance (solvent) differs because of differences in susceptibility, temperature, concentration or pH, for instance.

The third step depends on whether or not a peak was found in the second step. If a peak was found, ***sref*** determines the interpolated peak top and shifts its ppm value to the *ref.* value from the ***edlock*** table. The processing parameters OFFSET, SF and SR are changed accordingly. As such, the result of the default (step 1) is slightly corrected in order to set the peak of the reference substance exactly to 0. You can check this by putting the cursor on this peak. If no peak was found, you will get the message: 'sref: no peak found default calibration done'. The result of the default calibration (step 1) is stored without any further correction.

The three cases below show the calibration of a 1H, 13C and 31P spectrum with C6D6 as a solvent. Table 6.5 shows the corresponding entry in the ***edlock*** table:

Solvent	Field	Lock power	Nucleus	Distance [ppm]	Ref. [ppm]	Width [ppm]	RShift [ppm]
C6D6	-150	-15.0					
			1H	7.28	0.0	0.5	0.000
			2H	7.28	0.0	0.5	0.000
			13C	128.0	0.0	5.0	0.220
			31P	0.00	10.5	5.0	13.356

Table 6.5

case #1 - calibration of a 1H spectrum: A spectrum was acquired while being locked on C6D6. ***sref*** will do a default calibration and look for a signal at 0.0 ppm (*Ref.*) in a window of +/- 0.25 ppm. If a peak is found, its chemical shift will be set to 0 ppm.

case #2 - calibration of a 13C spectrum: A spectrum was acquired while being locked on C6D6. ***sref*** will do a default calibration and look for a signal at 0.0 ppm (*Ref.*) in a window of +/- 2.5 ppm. If a peak is found, its chemical shift will be set to 0 ppm.

case #3 - calibration of a 31P spectrum: A spectrum was acquired while being locked on C6D6. ***sref*** will do a default calibration and look for a signal at 10.5 ppm (*Ref.*) in a window of +/- 2.5 ppm. If a peak is found, its chemical shift will be set to exactly 10.5 ppm.

On 2D spectra, ***sref*** calibrates the F2 and F1 dimension and this involves the same steps as described above for 1D spectra.

INPUT PARAMETERS

set by the acquisition, can be viewed with ***dpa*** or by typing ***ls solvent*** etc.:

SOLVENT - the solvent of the sample
INSTRUM - configuration name (entered during ***cf***) of the spectrometer
LOCNUC - lock nucleus
SFO1 - spectral frequency
NUC1 - measured nucleus
SW - sweep width

OUTPUT PARAMETERS

processing parameters which can be viewed with ***edp***:
processing status parameters which can be viewed with ***dpp*** :

SF - spectral reference frequency
OFFSET - the ppm value of the first data point of the spectrum
SR - spectral reference

INPUT FILES

<xwhome>/conf/instr/<instrum>/

`2Hlock` - edlock table for 2H locked samples
`19Flock` - edlock table for 19F locked samples

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`proc` - processing parameters
`procs` - processing status parameters

view, viewx, vieww, viewmg

NAME

view - preview the output of the command plot
viewx - preview the output of the command plotx
vieww - preview the output of the command plotw
viewmg - preview the output of the commands flplot

DESCRIPTION

The command ***view*** shows a preview of a plot on the screen. It is used in combination with the command ***plot***, i.e. for parameter oriented plotting. After setting all plot parameters with ***edg*** and selecting the printer with ***edo***, ***view*** allows you to check the result before actually plotting the spectrum. If you want, you can change parameters with ***edg***, while the Preview window remains open and click ***Restart*** to see the change.

viewx works like ***view***, except that it shows a preview of the command ***plotx***, which is used for auto expanded plots.

vieww works like ***view***, except that it shows preview of the command ***plotw***, which is used for white washed stack plots.

viewmg works like ***view***, except that it shows a preview of the command ***flplot*** which is used after multiple ***plots*** commands to make several plots on one sheet.

Note that the ***view**** commands cannot be used in combination with the autoplot command which plots the current dataset according to the XWIN-PLOT layout specified in ***edo***.

SEE ALSO

edg, edo, plot, plotx, plotw, plots, flplot

xwinplot

NAME

xwinplot - start XWIN-PLOT; the wysiwyg plot editor

DESCRIPTION

The command ***xwinplot*** starts the XWIN-PLOT program, the WYSIWYG plot editor of the NMR Suite. XWIN-PLOT can also be started from the XWIN-NMR Windows menu or from the desktop.

For a full description of XWIN-PLOT, please refer to the XWIN-PLOT online help.

The XWIN-NMR command ***autoplot*** allows you to plot a spectrum using an XWIN-PLOT layout.

SEE ALSO

autoplot, xwp_lp, xwp_pp

xwp_lp

NAME

xwp_lp - create parameter list for XWIN-PLOT

DESCRIPTION

The command ***xwp_lp*** creates a parameter list for XWIN-PLOT. It must be run if you want the acquisition and processing parameters to appear on the plot.

Instead of running ***xwp_lp*** from XWIN-NMR, you can also create the parameter list from XWIN-PLOT by clicking the ***XWIN-NMR Interface*** menu.

If one of the commands ***plot**** or ***view**** was executed with the ***edg*** parameter PARAM = yes or if ***lppl*** was executed, the parameter lists already exists. However, this also contains the plot parameters (***edg***) and these have no meaning in XWIN-PLOT.Therefore, for XWIN-PLOT, you must recreate the parameter list with ***xwp_lp***.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`format.temp` - format file for parameters which appear on the plot

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`parm.txt` - ascii file containing parameters which appear on the plot

If the file `format.temp` does not exist, it is created by ***xwp_lp*** from using the file `pulseprogram` from the same directory as input. If this does not exist either, it is also created by ***xwp_lp*** from the pulse program which is defined by the acquisition parameter PULPROG. This pulse programs resides in:

<xwhome>/exp/stan/nmr/lists/pp

USAGE IN AU PROGRAMS

XWP_LP

SEE ALSO

xwp_pp, xwinplot, autoplot, lppl

xwp_pp

NAME

xwp_pp - create peak list for XWIN-PLOT

DESCRIPTION

The command ***xwp_lp*** creates a peak list for XWIN-PLOT.

Instead of running ***xwp_lp*** from XWIN-NMR, you can also create the peak list from XWIN-PLOT by clicking the ***XWIN-NMR Interface*** menu.

xwp_pp creates the same list as would be created with the ***pp*** command with the ***edo*** parameter CURPRIN = peak.txt.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`peaks` - peak list containing all peaks (input if it exists)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`peaks` - peak list containing all peaks (output if it did not exist)
`peak.txt` - peak list containing peaks which appear in XWIN-PLOT

USAGE IN AU PROGRAMS

XWP_PP

SEE ALSO

xwp_lp, xwinplot, autoplot, pp, pps

Chapter 7

Relaxation analysis

This chapter describes all XWIN-NMR commands which can be used for relaxation analysis. Most of them must be entered from the Relaxation menu which can be entered from the main menu by clicking ***Analysis*** → ***Relaxation***. Relaxation analysis can be done on T1, T2 and various other type of relaxation experiments. Relaxation curves of up to six components can be fitted.

First we will describe a typical procedure to do a one-component T1 experiment. The individual relaxation commands used here are described in detail in the rest of this chapter.

1. Setup a pseudo 2D dataset where each row corresponds to a certain delay. You can, for example, do this with the commands:
   - ***rpar PROTONT1 all***
     to read a standard T1 parameter set
   - ***edlist vd t1delay***
     to setup a list of delays
2. Enter ***zg*** to run the experiment.
3. Enter ***xf2*** to Fourier transform the pseudo 2D in the F2 dimension
4. Click ***Analysis*** → ***Relaxation*** to switch to the relaxation menu
5. Enter ***edt1*** to set up the relaxation parameters
6. Enter ***rspc*** to extract a row (usually the first) from the pseudo 2D data

7. Determine the peak intensities and integral regions on this row in one of the following ways:
   - automatically with the command ***ppt1***
   - interactively from the baseline menu (enter ***bas1***, click ***def-pts*** and the integral ranges from the integrate menu (click ***integrate***)
8. Click ***Analysis*** → ***Relaxation*** to switch to the relaxation menu
9. Enter ***pd*** or ***pd0*** to pick the data points for relaxation analysis
10. Enter ***ct1*** or ***simfit*** for relaxation analysis of the current peak (determined by the ***edt1*** parameter CURSOR).
11. For multi peak analysis, enter the sequence:
    - ***nxpt***
    - ***ct1*** or ***simfit***

    for each additional peak.

Multi peak analysis can also be done in one step with the commands ***dat1*** or ***simfit all***, thereby replacing step 10 and 11.

The calculated T1 value as well as the intensity of the individual data points are stored in the device determined by the ***edo*** parameter CURPRIN. The relaxation curve is displayed in the XWIN-NMR data field.

ct1

NAME

ct1 - calculate the T1 value of the current peak

DESCRIPTION

The command ***ct1*** calculates the T1 value of the current peak by fitting the relaxation points of that peak. These points are measured in a series of experiments with varying delays (t). The T1 fit is defined by the following formula:

$$I(t) = I(0) + P \times \exp\left(\frac{t}{T1}\right)$$

ct1 determines the best fit by varying I(0), P and T1 in an iterative process according to the Levenberg-Marquardt algorithm.

Before you run ***ct1***, you must first determine the relaxation points with ***pd*** or ***pd0*** (see the description of these commands).

An example of the output of ***ct1*** is shown in table 7.1.

CT1 RESULT

Dataset: C:/data/guest/nmr/t1data/1/pdata/1
INTENSITY fit: I[t]=I[0]+P*exp(-t/T1)
16 points for Peak 1, Cursor Point = 9361, 719.49 Hz, 1.998 ppm
Results Comp. 1
I[0] = 8.074e-01
P = -1.698e+00
T1 = 758.107m
SD = 4.323e-02

tau	*cursor*	*freq*	*ppm*	*integral*	*intensity*
1.000m	9361.61	719.34	1.997	-2.735e+09	-3.1896e+08
50.000m	9361.24	719.43	1.998	-2.2537e+09	-2.5338e+08
100.000m	9361.07	719.47	1.998	-1.523e+09	-1.8606e+08
..	..	..	..	..	..

Table 7.1

Note that this is the result for the first peak in the spectrum which lies at cursor point 9631 and 1.997 ppm. The peak position drifts to higher frequency but less then one cursor point. Both the intensity and integral of the peak is listed for each experiment. The T1 value is calculated on either the intensity or integral distribution, depending on the value of the ***edt1*** parameter FITTYPE. The measurement involves 16 experiments, of which the first three are listed in table 7.1

ct1 calculates T1 for the current peak (the value of the ***edt1*** parameter CURSOR). This is usually the peak whose relaxation points are displayed in the data field. You can calculate T1 for the next peak by running ***nxtp***, which changes display to the corresponding curve, and then run ***ct1*** again. Alternatively, you can calculate T1 for all peaks with the command ***dat1***. In this case, the relaxation points of the last peak will be displayed in the data field.

ct1 can only be used for relaxation curves which consist of a single component. For fitting multiple component curves, you can use the ***simfit*** command.

ct1 sends its output to a device specified by parameter CURPRIN which can be set with ***edo***. This can be the screen, a printer, a file or the Clipboard. Furthermore, the output is stored in the files `ct1t2.out` (or `ct1t2.txt`) and `t1t2.dx` in the processed data directory.

INPUT PARAMETERS

set by the acquisition, can be viewed with ***dpa*** or by typing ***2s vdlist*** :

VDLIST - variable delay list, used when FITYPE = vdlist and the file `vdlist` does not exist in the acquisition data directory

set by the user with ***edt1*** :

FITTYPE - relaxation fit type; peak area or highest intensity
CURSOR - the peak whose relaxation value is calculated

OUTPUT PARAMETERS

set by ***ct1***, can be viewed with ***edt1*** :

Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for T1 calculation

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`vdlist` - input if FITTYPE = vdlist and the file `vdlist` exists

<xwhome>/exp/stan/nmr/lists/vd/

`<name>` - input if FITTYPE = vdlist, the file `vdlist` as described above does not exist and the parameter VDLIST = <name>

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1peaks` - peak positions and intensities (created by ***pd***)
`t1ints` - integral ranges and areas (created by ***pd***)
`t1par` - relaxation parameters (edited with ***edt1***)
`t1elim` - all data points with the eliminated points marked (created by ***pd***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`ct1t2.out` - T1 value and a list of the relaxation points
`ct1t2.txt` - equivalent of `ct1t2.out` in XWIN-NMR 3.1 and newer
`t1t2.dx` - T1 value and a list of the relaxation points in JCAMP-DX format
`t1par` - relaxation parameters

USAGE IN AU PROGRAMS

CT1

SEE ALSO

xf2, edt1, rspc, ppt1, pd, lstp, nxtp, elim, dat1, simfit, ct2, pd0

ct2

NAME

ct2 - calculate the T2 value

DESCRIPTION

The command ***ct2*** calculates the T2 value by fitting relaxation data points. It is typically used for Carr-Purcell type T2 experiments as used in the pulse program `cpmg`. Here, the relaxation points are stored as raw 1D data. The T2 fit is defined by the following formula:

$$I(t) = P \times \exp\left(\frac{t}{T2}\right)$$

ct2 determines the best fit in an iterative process, by varying I(0), P and T2 according to the Levenberg-Marquardt algorithm.

An example of the output of ***ct2*** is shown in table 7.2.

CT2 RESULT

Dataset: C:/data/guest/nmr/t2/1/pdata/1
INTENSITY fit: I[t]=P*exp(-t/T2)
32 points for Peak 1
Results Comp. 1
P = 9.811e-01
T2 = 16.257m
SD = 5.755e-03

tau	*cursor*	*freq*	*ppm*	*integral*	*intensity*
40.000u	2.00	3032.29	8.420	59709	59709
680.000u	34.00	2876.04	7.986	56835	56835
1.320m	66.00	2719.79	7.552	54274	54274
..	..	..	..	..	..

Table 7.2

Note that this output has the same format as the output of ***ct1*** which is used for

multiple peaks in a pseudo 2D spectrum. The reason is that ***ct2***, can also be used on pseudo 2D data, although done very often. In the usual case, of a 1D dataset, there is only one experiment (one peak) and the integral values are the same as the intensity values. In the above case, 32 points of the 1D raw data are used for T2 calculation of which the first three are listed in table 7.2.

Before you run ***ct2***, you must first pick the data points which are used for T2 calculation from the 1D raw data. This can be done with command ***pft2*** (see the description of this command).

ct2 can only be used for relaxation curves which consist of a single component. For fitting multiple component curves, you can use the ***simfit*** command.

ct2 sends its output to a device specified by parameter CURPRIN which can be set with ***edo***. This can be the screen, a printer, a file or the Clipboard. Furthermore, the output is stored in the files `ct1t2.out` (or `ct1t2.txt`) and `t1t2.dx` in the processed data directory.

INPUT PARAMETERS

set by the user with ***edt1*** :

FITTYPE - relaxation fit type; peak area or highest intensity
CURSOR - the peak whose relaxation value is calculated

OUTPUT PARAMETERS

set by ***ct2***, can be viewed with ***edt1*** :

Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for T2 calculation

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1peaks` - data points and intensities (created by ***pft2***)
`t1ints` - same as `t1peaks` (created by ***pft2***)
`t1elim` - all data points with the eliminated points marked (created by ***pd***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`ct1t2.out` - T2 value and a list of the relaxation points
`ct1t2.txt` - equivalent of `ct1t2.out` in XWIN-NMR 3.1 and newer
`t1t2.dx` - T2 value plus relaxation points in JCAMP-DX format
`t1par` - relaxation parameters

USAGE IN AU PROGRAMS

CT2

SEE ALSO

edt1, pft2, lstp, simfit, ct1

dat1

NAME

dat1 - calculate the T1 value for all peaks

DESCRIPTION

The command ***dat1*** calculates the T1 value of all peaks which are selected for T1 analysis. It uses the same algorithm as ***ct1***. In fact, ***dat1*** can be used as an alternative to the sequence:

ct1 - ***nxtp*** - ***ct1*** - ***nxtp*** - ***ct1*** etc.

The difference is that in the above sequence only the result of the last ***ct1*** command is stored.

Like ***ct1***, ***dat1*** can only handle one component distributions. For multiple component distributions, you can use the command ***simfit all***.

The command ***dat2*** exists, but it only makes sense on a pseudo 2D dataset. T2 experiments are, however, usually stored as 1D data. In that case, ***dat2*** gives the same result as ***ct2***.

INPUT PARAMETERS

set by the acquisition, can be viewed with ***dpa*** or by typing ***2s vdlist*** :

VDLIST - variable delay list, used when FITYPE = vdlist and the file `vdlist` does not exist in the acquisition data directory

set by the user with ***edt1*** :

FITTYPE - relaxation fit type; peak area or highest intensity

OUTPUT PARAMETERS

set by ***ct1***, can be viewed with ***edt1*** :

Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for T1 calculation
CURSOR - the peak whose relaxation value is calculated

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`vdlist` - input if FITTYPE = vdlist and the file `vdlist` exists

<xwhome>/exp/stan/nmr/lists/vd/

`<name>` - input if FITTYPE = vdlist, the file `vdlist` as described above does not exist and the parameter VDLIST = <name>

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1peaks` - peak positions and intensities (created by ***pd***)
`t1ints` - integral ranges and areas (created by ***pd***)
`t1par` - relaxation parameters (edited with ***edt1***)
`t1elim` - all data points with the eliminated points marked (created by ***pd***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`ct1t2.out` - T1 value and a list of the relaxation points
`ct1t2.txt` - equivalent of `ct1t2.out` in XWIN-NMR 3.1 and newer
`t1t2.dx` - T1 value and relaxation points in JCAMP-DX format
`t1par` - relaxation parameters

USAGE IN AU PROGRAMS

DAT1

SEE ALSO

ct1, xf2, edt1, rspc, ppt1, pd, pd0, nxtp, lstp, simfit

edt1

NAME

edt1 - edit the relaxation parameters

DESCRIPTION

The command ***edt1*** opens a dialog box in which you can set all relaxation parameters. Although the name of the command refers to T1, ***edt1*** is used for all types of relaxation analysis.

More information about these parameters can be found in chapter 2.6 and at the description of the commands which interpret them (like ***pft2***, ***pd***, ***simfit*** etc.)

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1par` - relaxation parameters

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1par` - relaxation parameters

SEE ALSO

pft2, pd, pd0, simfit, ct1, ct2, dat1, nxtp, lstp

elim

NAME

elim - eliminate points from the relaxation curve

DESCRIPTION

The command ***elim*** allows you to eliminate points from the relaxation curve. After entering ***elim***, you must first click the left mouse button to put the cursor on the curve. If you then click the middle mouse button, the data point that is closest to the click position is removed. Points which have been eliminated are no longer displayed and do not contribute to the relaxation analysis.

The command ***elim*** only exists for historical reasons. In fact, you can eliminate points as described above without entering ***elim*** first.

If the relaxation analysis is done with ***ct1***, ***ct2*** or ***dat1***, the eliminated points still appear in the output list but their intensities are displayed as *****. If the relaxation analysis is done with ***simfit***, the eliminated points do not appear in the output list.

Eliminated points can be restored with the command ***rstp***.

OUTPUT PARAMETERS

set by ***elim***, can be viewed with ***edt1*** :

Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for relaxation analysis

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1peaks` - peak positions and intensities (created by ***pd***)
`t1ints` - integral ranges and areas (created by ***pd***)
`t1elim` - all data points with the eliminated points marked (created by ***pd***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1elim` - all data points with the eliminated points marked (created by ***pd***)
`t1t2.dx` - list of the relaxation points in JCAMP-DX format

SEE ALSO

rstp, pd, pd0, ct1, ct2, dat1, simfit

lstp

NAME

lstp - list the relaxation points of the current peak

DESCRIPTION

The command ***lstp*** lists the relaxation points of the current peak. Each data point is shown with its experimental delay, cursor position, frequency, integral and intensity.

On a 1D dataset, ***lstp*** can be used after ***pft2*** to view the relaxation points before you run ***ct2*** or ***simfit***. Likewise, on a pseudo 2D dataset, ***lstp*** can be used after ***pd*** to view the points before you run ***ct1*** or ***simfit***.

The current peak is determined by the value of the parameter CURSOR and is usually the peak whose relaxation points are currently displayed in the data field.

lstp sends its output to a device specified by parameter CURPRIN which can be set with ***edo***. This can be the screen, a printer, a file or the Clipboard. Furthermore, the output is always stored in the files `ct1t2.out` (or `ct1t2.txt`) and `t1t2.dx` in the processed data directory.

Note that ***lstp*** overwrites the result of ***ct1*** if this command has already been executed. The same counts for ***ct2***.

An example of the output of ***lstp*** is shown in table 7.3. It belongs to the first

tau	*cursor*	*freq*	*ppm*	*integral*	*intensity*
1.000m	9361.61	719.34	1.997	-2.735e+09	-3.1896e+08
50.000m	9361.24	719.43	1.998	-2.2537e+09	-2.5338e+08
100.000m	9361.07	719.47	1.998	-1.523e+09	-1.8606e+08
200.000m	9361.19	719.44	1.998	-1.0528e+09	-1.431e+08
300.000m	9361.25	719.43	1.998	-5.9202e+08	-9.952e+07
500.000m	9361.27	719.42	1.998	1.4766e+08	-2.0653e+07
800.000m	9361.12	719.46	1.998	9.3581e+08	6.5615e+07
1.000s	9361.07	719.47	1.998	1.2997e+09	1.0644e+08

Table 7.3

peak of a pseudo 2D dataset with eight rows corresponding to eight delays. As you can see from the columns 2, 3 and 4, the peak position varies a little between rows but this drift is less than one cursor point.

INPUT PARAMETERS

set by the user with ***edt1*** :

CURSOR - the peak whose relaxation value is calculated

OUTPUT PARAMETERS

set by ***lstp***, can be viewed with ***edt1*** :

Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for relaxation analysis

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1par` - relaxation parameters (edited with ***edt1***)
`t1peaks` - peak positions and intensities (created by ***pd***)
`t1ints` - integral ranges and areas (created by ***pd***)
`t1elim` - all data points with the eliminated points marked (created by ***pd***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`ct1t2.out` - list of data points with time, frequency, intensity and area
`ct1t2.txt` - equivalent of `ct1t2.out` in XWIN-NMR 3.1 and newer
`t1t2.dx` - list of data points in JCAMP-DX format
`t1par` - relaxation parameters (edited with ***edt1***)

SEE ALSO

nxtp, pft2, pd, pd0, ct1, ct2, simfit

lsta

NAME

lsta - list the relaxation points of all peaks

DESCRIPTION

The command ***lsta*** lists the relaxation points of all peaks. Each data point is shown with its experimental delay, cursor position and frequency. For each peak the integral or intensity value is shown, depending on the value of the parameter FITTYPE.

lsta only exists in XWIN-NMR 3.1 and newer.

On a pseudo 2D dataset, ***lsta*** can be used after ***pd*** to view the points before you run ***ct1*** or ***simfit***.

lsta sends its output to a device specified by parameter CURPRIN which can be set with ***edo***. This can be the screen, a printer, a file or the Clipboard. Furthermore, the output is always stored in the files `ct1t2.out` (or `ct1t2.txt`) in the processed data directory.

Note that ***lsta*** overwrites the result of ***ct1*** if this command has already been executed. The same counts for ***ct2***.

An example of the output of ***lsta*** is shown in table 7.4. It belongs to the first

tau	*cursor*	*ppm*	*peak 1*	*peak 2*	*peak 3*
1.000m	9361.61	1.997	-2.8802e+09	-1.2013e+09	-1.4193e+09
50.000m	9361.24	1.998	-2.4338e+09	-1.1453e+09	-1.248e+09
100.000m	9361.07	1.998	-1.5821e+09	-1.0093e+09	-1.1852e+09
200.000m	9361.19	1.998	-1.0784e+09	-8.4397e+08	-9.8166e+08
300.000m	9361.25	1.998	-5.7626e+08	-6.1976e+08	-7.7473e+08
500.000m	9361.27	1.998	2.3674e+08	-4.7259e+08	-4.6758e+08
800.000m	9361.12	1.998	1.082e+09	-7.443e+07	-8.6941e+07
1.000s	9361.07	1.998	1.4717e+09	1.3648e+08	1.2489e+08

Table 7.4

three peaks of a pseudo 2D dataset with eight rows corresponding to eight delays. The values of *cursor* and *ppm* correspond to peak 1. Note that the peak position varies a little between rows but this drift is less than one cursor point.

INPUT PARAMETERS

set by the user with ***edt1*** :

FITTYPE - relaxation fit type; peak area or highest intensity

OUTPUT PARAMETERS

set by ***lsta***, can be viewed with ***edt1*** :

Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for relaxation analysis

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1par` - relaxation parameters (edited with ***edt1***)
`t1peaks` - peak positions and intensities (created by ***pd***)
`t1ints` - integral ranges and areas (created by ***pd***)
`t1elim` - all data points with the eliminated points marked (created by ***pd***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`ct1t2.out` - list of data points with time, frequency, intensity and area
`ct1t2.txt` - equivalent of `ct1t2.out` in XWIN-NMR 3.1 and newer
`t1par` - relaxation parameters (edited with ***edt1***)

SEE ALSO

lstp, nxtp, pft2, pd, pd0, ct1, ct2, simfit

nxtp

NAME

nxtp - go to the next peak in the peak list

DESCRIPTION

The command ***nxtp*** goes to the next peak in the peak list. The intensity distribution of that peak is shown in the data field. ***nxtp*** is used on a pseudo 2D dataset where multiple peaks have been selected for relaxation analysis.

nxtp is used after the peaks have been picked (see ***ppt1***) and the intensity distribution for each peak has been determined (see ***pd***). It is typically part of the following sequence:

pd - ***ct1*** - ***nxtp*** - ***ct1*** - ***nxtp*** - ***ct1*** etc.

or

pd - ***simfit*** - ***nxtp*** - ***simfit*** - ***nxtp*** - ***simfit*** etc.

Instead of stepping through all the peaks, you can also switch to a specific peak by setting the parameter CURSOR with ***edt1***. Furthermore, you can analyse all peaks at once with ***dat1*** or ***simfit all***.

INPUT PARAMETERS

set by ***pd*** or by the user with ***edt1*** :

CURSOR - the peak whose relaxation value is calculated

OUTPUT PARAMETERS

set by ***nxtp***, can be viewed with ***edt1*** :

CURSOR - the peak whose relaxation value is calculated
Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for relaxation analysis

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1par` - relaxation parameters (edited with ***edt1***)
`t1peaks` - peak positions and intensities (created by ***pd***)
`t1ints` - integral ranges and areas (created by ***pd***)
`t1elim` - all data points with the eliminated points marked (created by ***pd***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1t2.dx` - list of the relaxation points in JCAMP-DX format
`t1par` - relaxation parameters

SEE ALSO

pft2, pd, ct1, ct2, dat1, simfit, edt1

pd, pd0

NAME

pd - pick data points for relaxation analysis
pd0 - pick data points for relaxation analysis at constant peak positions

DESCRIPTION

The command ***pd*** picks the data points from a pseudo 2D dataset for relaxation analysis. The pseudo 2D data represent a series of relaxation experiments with different delays where each experiment is stored as one row. Before you run ***pd***, you must perform peak picking and integration on one row (usually the first). For each peak, ***pd*** determines the intensity distribution over a series of experiments, i.e. along a column of the pseudo 2D dataset. The intensity distribution of the first peak is displayed in data field.

pd is usually part of the following sequence:

1. enter ***xf2*** to Fourier transform the pseudo 2D in the F2 dimension
2. click ***Analysis*** → ***Relaxation*** to switch to the relaxation menu
3. enter ***edt1*** to set up the relaxation parameters
4. enter ***rspc*** to extract a row (usually the first) from the pseudo 2D data
5. determine the peak intensities and integral regions on this row in one of the following ways:
   - automatically with the command ***ppt1***
   - interactively from the baseline menu (enter ***bas1***, click ***def-pts*** and the integral ranges from the integrate menu (click ***integrate***)

   see ***ppt1*** for more details
6. click ***Analysis*** → ***Relaxation*** to switch to the relaxation menu
7. enter ***pd*** or ***pd0*** to pick the data points for relaxation analysis
8. enter ***ct1*** or ***simfit*** for relaxation analysis

pd not only determines the intensity distribution but also the integral (area) distribution. Depending on the parameter FITTYPE, one or the other is displayed in the data field. However, both are stored and can be used for relaxation analysis.

The F1 dimension of the 2D dataset is in the time domain but the values of the F1 axis have no relevance. The actual delays as they were used in the experiments, are usually stored in a *vd* list. ***pd*** reads this list to determine the x-axis (time axis) units of its output data. However, a *vp*, *vc* or any other list can also be used. ***pd*** interprets the ***edt1*** parameter LISTTYP for the correct list.

Sometimes the peak positions drift in the course of the relaxation experiment, i.e. the peaks appear at slightly different positions in different rows. Therefore, ***pd*** does not simply take the intensity at the same position in each row. Instead, it searches for the peak maximum. The search range is the peak position determined for one row (START) plus or minus DRIFT. If a maximum is found within that range and its intensity is larger than PC*noise, it will contribute to relaxation curve.

pd0 works like ***pd***, except that it ignores the value of DRIFT and takes the intensity at exactly the peak position as determined for the first row. If there is no drift, ***pd*** and ***pd0*** have the same result.

INPUT PARAMERS

set by the user with ***edt1*** :

NUMPNTS - number of data points used for relaxation analysis
FITTYPE - relaxation fit type; peak area or highest intensity
LISTTYP - type of lists with the experimental delays (tau values)
DRIFT - drift of the peak positions in the course of the experiment
START - first row used for relaxation analysis
INC - row increment for relaxation analysis

set by the user by typing ***pc*** :

PC - peak picking sensitivity

OUTPUT PARAMETERS

set by ***pd*** or ***pd0***, can be viewed with ***edt1*** :

Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for relaxation analysis
CURSOR - the peak whose relaxation value is calculated

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1par` - relaxation parameters (edited with ***edt1***)
`intrng` - integral regions
`baslpnts` - peaks positions (cursor positions and ppm values)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1t2.dx` - list of relaxation points in JCAMP-DX format
`t1peaks` - peak positions and intensities
`t1ints` - integral ranges and areas
`t1par` - relaxation parameters (edited with ***edt1***)
`t1elim` - all data points (used by ***elim***)

USAGE IN AU PROGRAMS

PD

PD0

SEE ALSO

xf2, edt1, rspc, ppt1, ct1, ct2, dat1, dat2, nxtp, simfit

pft2

NAME

pft2 - pick data points for T2 analysis

DESCRIPTION

The command ***pft2*** picks the data points for T2 analysis. It is typically used for Carr-Purcell type experiments where the relaxation points are stored as raw 1D data. Before you run ***pft2***, you must set up the relaxation parameters from the Relaxation menu. Usually, ***pft2*** is part of the following sequence:

1. click ***Analysis*** → ***Relaxation*** to switch to the T2 menu
2. enter ***edt1*** to set the relaxation parameters
3. enter ***pft2*** to pick the data points for T2 analysis
4. enter ***ct2*** to fit the data points and calculate T2

A detailed explanation of the parameters can be found in chapter 2.6.

Although ***pft2*** is only used for T2 data, the parameter editor and several files it uses refer to T1. The reason is that these are also used for T1 and, in fact, several other relaxation type experiments.

INPUT PARAMERS

set by the user with ***edt1*** :

NUMPNTS - number of data points used for T2 calculation
LISTTYP - type of list with delays between data points
START - first data point used for T2 analysis
INC - data point increment for T2 analysis
X_START - x-axis start; time of the first data point used for T2 calculation
LISTINC - time increment between data points

OUTPUT PARAMETERS

set by ***pft2***, can be viewed with ***edt1*** :

Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for T2 calculation

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`fid` - raw 1D data containing the T2 data points

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1par` - relaxation parameters (edited with ***edt1***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1t2.dx` - list of the relaxation points in JCAMP-DX format
`t1peaks` - data points and intensities (created by ***pd***)
`t1ints` - same as `t1peaks`
`t1par` - relaxation parameters (edited with ***edt1***)

USAGE IN AU PROGRAMS

PFT2

SEE ALSO

edt1, ct2, simfit

ppt1

NAME

ppt1 - peak picking and integration for relaxation analysis

DESCRIPTION

The command ***ppt1*** determines the peak positions and integral regions for relaxation analysis. The command must be entered on a row which has been extracted (with ***rspc***) from a pseudo 2D dataset.

ppt1 picks the peaks according to the processing parameters MI, MAXI, PC and PSIGN. These can be set by entering their names (in lowercase letters) on the command line. They cannot be set with ***edp*** because they are 1D parameters and do not appear in the 2D parameter editor.

As an alternative to ***ppt1***, you can do manual peak picking and integration.

Manual peak picking

1. Enter ***basl*** to change to the baseline menu.
2. Click ***def-pts*** to attach the cursor to the spectrum.
   (if the peaks have already been defined, you are first prompted to append to (a) or overwrite (o) the existing file)
3. Move the cursor over the spectrum and click the middle mouse button at the top of the peaks you want to use for relaxation analysis.
4. Click the left mouse button to release the cursor from the spectrum.
5. Click ***return*** → ***Return*** to change to the main menu.

Manual integration

1. Click ***integrate*** to change to the integration menu.
2. Click the left mouse button in the data field to attach the cursor to the spectrum.
3. Click the middle mouse button repeatedly to define the integral regions for all peaks which you selected during manual peak picking.
4. Click ***return*** → ***Save as 'intrng' and return*** to store the integral regions and change to the main menu.

After running ***ppt1*** or manual peak picking and integration, you can change to

the Relaxation menu by clicking ***Analysis*** → ***Relaxation***. Then you can run ***pd*** to determine the intensity distribution over all rows and ***ct1*** or ***simfit*** to calculate the relaxation value.

INPUT PARAMERS

set by the user by typing ***mi***, ***maxi*** etc.:

MI - minimum relative intensity (cm)
MAXI - maximum relative intensity (cm)
PC - peak picking sensitivity
PSIGN - peak sign (pos, neg, or both)

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`1r` - 1D processed data (a row from the pseudo 2D dataset)
`proc` - processing parameters

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`intrng` - integral regions
`baslpnts` - peaks positions (cursor positions and ppm values)

SEE ALSO

xf2, edt1, rspc, pd, ct1, simfit

rspc

NAME

rspc - read a row from a pseudo 2D dataset for relaxation analysis

DESCRIPTION

The command ***rspc*** reads a row from a pseudo 2D dataset for relaxation analysis. It must be entered from the relaxation menu and it automatically changes the display to the 1D processing menu. From there, you can determine the peak positions and integral regions for relaxation analysis (see ***ppt1***).

rspc extracts the row which is specified by the parameter START (default is 1).

INPUT PARAMERS

set by the user with ***edt1*** :

START - the row which is extracted (default is 1; the first row)

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`2rr` - 2D processed data (series of relaxation experiments)
`t1par` - relaxation parameters (edited with ***edt1***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`1r` - 1D processed data (the extracted row)

SEE ALSO

xf2, edt1, ppt1, pd, ct1, simfit

rstp

NAME

rstp - restore the eliminated points of a relaxation curve

DESCRIPTION

The command ***rstp*** restores data points which have been eliminated from the relaxation curve. The eliminated points reappear in the data field.

rstp works on 1D and pseudo 2D data. On the latter, it restores the eliminated data points of all peaks.

Another way of restoring eliminated points is recalculating the relaxation curve(s) with ***pft2*** (1D) or ***pd*** (pseudo 2D)

OUTPUT PARAMETERS

set by ***rstp***, can be viewed with ***edt1*** :

Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for relaxation analysis

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1elim` - all data points with the eliminated points marked
`t1par` - relaxation parameters (edited with ***edt1***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1elim` - all data points
`t1t2.dx` - list of the relaxation points in JCAMP-DX format
`t1par` - relaxation parameters

SEE ALSO

elim, edt1, ct1, ct2, dat1, simfit

simfit

NAME

simfit - simplex fit; multiple component relaxation analysis

DESCRIPTION

The command ***simfit*** calculates the relaxation time, for various relaxation experiments types and multi component relaxation distributions. The differences with ***ct1*** /***ct2*** is that these only work on T1/T2 data with one component relaxation curves.

Table 7.5 shows the experiment types which ***simfit*** can handle and the corresponding fit functions is uses.

Exp. type	Comp	Fit function
uxnmrt1t2	1	I[t] = I[0]+P*exp(t/T1)
invrec	1 - 4	I[t] = I[0]*(1-2A*exp(-t/T1))
satrec	1 - 6	I[t] = I[0]*(1-exp(-t/T1))
cpt1rho	1 - 4	I[t] = I[0]/(1-TIS/T1rho)*(exp(-t/T1rho)-exp(t/TIS))
expdec	1 - 6	I[t] = I[0]*exp(-t/T)
gaussdec	1 - 6	I[t] = I[0]*exp(-SQR(t/T))
lorgauss	1 - 3	I[t] = IL*exp(-t/TL)+IG*exp(-SQR(t/TG))
linear	1 - 6	I[t] = A+B*t
varbigdel	1 - 6	I = I[0]*exp(-D*SQR(2*PI*gamma*G*LD)*(BD-LD/3)*1e4)
varlitdel	1 - 6	I = I[0]*exp(-D*SQR(2*PI*gamma*G*LD)*(BD-LD/3)*1e4)
vargrad	1 - 6	I = I[0]*exp(-D*SQR(2*PI*gamma*G*LD)*(BD-LD/3)*1e4)

Table 7.5

Although ***simfit*** works on 1D and pseudo 2D data, it is, in practice, only used on the latter.

simfit determines the best fit by varying the function variables in an iterative process according to the simplex algorithm.

Before you run ***simfit***, you must set the following parameters with ***edt1*** :

CURSOR: the peak on which the (first) ***simfit*** should run (default 1)

FITTYPE: fit type; peak area of highest intensity

FCTTYPE : the fit function that corresponds to your experiment

COMPNO: the number of components contributing to the relaxation curve.

EDGUESS: the initial guess and step rate of the fit function variables

Which variables appear in the EDGUESS table, depends on the chosen function type. For each function variable, the initial guess (G) and step rate (S) can be set for each component (C). Table 7.6 shows the EDGUESS table for an inversion recovery experiment, with a 2 component relaxation distribution.

GC1I0	0.5	SC1I0	0.05
GC1A	1	SC1A	0.1
GC1T1	2	SC1T1	0.2
GC2I0	0.5	SC2I0	0.05
GC2A	1	SC2A	0.1
GC2T1	2	SC2T1	0.2

Table 7.6

The initial guess for I[0] must be such that the total value of all components does not exceed 1. If there is only one component, I[0] is usually set to 1. The step rate is usually set to about one tenth or the initial guess. If the step rate of a variable is to zero, then this variable is not changed during the iterations.

Note the commands ***ct1***, ***ct2***, ***dat1*** or ***dat2*** do not use the EDGUESS table. They calculate the initial values and step rates of the function variable I[0], P and T1.

simfit can be used in two different ways:

1. Use data points determined by peak picking on one of the rows.

   This is the same procedure as described for ***pd***, except for the last step where ***simfit*** is used instead of ***ct1***.

2. Use the data points from an ascii file created by the user.

   In this case, you have to create the file `t1ascii` with a text editor in the processed data directory (procno). Table 7.7 shows an example of this file

for an inversion recovery experiment.

SIMFIT 2		
0.1	-99	-48
0.5	-90	-44
1.0	-81	-39
2.0	-64	-33
3.0	-48	-23
4.0	-34	-17
6.0	-10	-6
8.0	10	5
10	27	13
15.0	55	26
20.0	73	38
40	97	50

Table 7.7

The first line must consists of the word SIMFIT and the number of peaks. The rest of the file consists of two or more columns where the first column contains the time values and the second column the corresponding intensity or integral values of the first peak. Possible further columns contain the values of further peaks. The command ***simfit asc*** takes its input data from `t1ascii` and fits the data points of the peak that is currently displayed. After that, ***simfit*** without argument will also take its input from the `t1ascii` file. You can, for example, use the sequence:

simfit asc - ***nxtp*** - ***simfit*** - ***nxtp*** - ***simfit*** - etc.

to fit the data points of consecutive peaks. As an alternative, you can also you the command ***simfit asc all*** to fit the data points of all peaks. Note that after ***simfit asc***, not only ***simfit*** but also ***ct1*** and ***dat1*** will take their input from the `t1ascii` file. After running ***pd***, however, ***simfit***, ***ct1*** and ***dat1*** will use the data points which were determined by peak picking one of the rows.

simfit sends its output to a device specified by the parameter CURPRIN

which can be set with ***edo***. This can be the screen, a printer, a file or the Clipboard. Furthermore, the output is always stored in the files `simfit.out` and `t1t2.dx` in the processed data directory.

simfit only works on the current peak and shows the corresponding curve on the screen. This curve often uses only a part of the vertical resolution. The reason is that it is scaled according to the minimum and maximum intensity of all curves (all peaks). Note that a curve fitted with ***ct1*** always uses the full vertical resolution because is scaled according to it's own minimum and maximum intensity.

simfit all fits the data points and calculates the relaxation value for all peaks. The entire result is listed on the device specified by CURPRIN (***edo***) and stored in the file `simfit.out`. The parameter CURSOR is set to the last peak and the corresponding relaxation curve is displayed in the data field.

The result of ***simfit*** can be evaluated by comparing the fitted curve (a line) with the relaxation data points (dots) on the screen. If they reasonably match, the fit has been successful and the relaxation value is reliable. The match can also be judged from the root mean square value (RRS) and standard deviation (SD) which are shown in the ***simfit*** output. If the fitted curve does not match the data points, you can repeat ***simfit*** with different starting values. Enter ***edt1*** → ***EDGUESS*** and change the value of one variable and its increment, for example with a factor of five. Run ***simfit*** and judge the result. If this is still unsatisfactory, change the same variable in the opposite direction or change a different variable. Do this until you have a reasonable fit. With some experience, you'll be able to estimate reasonable initial values for the variables and get a good fit at the first run.

INPUT PARAMETERS

set by the user with ***edt1*** :

CURSOR - the peak whose relaxation value is calculated
EDGUESS - table of initial values and step rates of the function variables
FITTYPE - fit type; peak area or highest intensity
FCTTYPE - function type used for fitting the relaxation curve
COMPNO - number of components contributing to the relaxation curve

OUTPUT PARAMETERS

set by ***simfit***, can be viewed with ***edt1*** :

Y_START - y-axis start; lowest intensity or area
Y_END - y-axis end; highest intensity or area
X_end - x-axis end; time of the last point used for relaxation analysis

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1peaks` - peak positions and intensities (output of ***pd*** or ***simfit asc***)
`t1elim` - all data points with the eliminated points marked (created by ***pd***)
`t1par` - relaxation parameters (edited with ***edt1***)
`t1ascii` - list of data points (only input of ***simfit asc***)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`t1peaks` - peak positions and intensities (output of ***simfit asc***)
`simfit.out` - ***simfit*** output (relaxation value and list of fitted points)
`t1t2.dx` - relaxation value plus relaxation points in JCAMP-DX format
`t1par` - relaxation parameters

USAGE IN AU PROGRAMS

SIMFIT

SIMFITALL

SIMFITASC

SIMFITASCALL

SEE ALSO

ct1, dat1, pd, pd0, nxtp, lstp, elim, rstp

Chapter 8

Dataset handling

This chapter describes all XWIN-NMR commands which can be used to read or write or delete datasets.

browse

NAME

browse - browse for XWIN-NMR data

DESCRIPTION

The command ***browse*** allows you to browse the disk(s) on your computer for XWIN-NMR data. Starting with the available *disk units*, it browses all variable parts of the data path, i.e. *du*, *user*, *name*, *expno* and *procno*. ***browse*** takes one argument which allows you to start at any level of the data path. It can be used as follows:

browse - show all disk units
browse 0 - same as ***browse***
browse 1 - show all users of the current disk unit
browse 2 - same as ***browse***
browse 3 - show all data names of the current user
browse 4 - show all expno's of the current data name
browse 5 - show all procno's of the current expno

The fact that ***browse 2*** is the same as ***browse*** has historical reasons.

INPUT DIRECTORY

<*du*>/data/<*user*>/nmr/<*name*>/<*expno*>/pdata/<*procno*>/

SEE ALSO

dir, dira, dirp, dirdat, dirf, dirs, dirser, dir2d, diro, search, re, rep

del, dela, delp, deldat

NAME

del - delete data
dela - delete acquisition data (raw data)
delp - delete processed data
deldat - delete data acquired at certain dates

SYNTAX

del [<name>] [<du> [<user>]]

dela, delp and deldat have the same syntax as del

DESCRIPTION

The commands ***del***, ***dela***, ***delp*** and ***deldat*** display a list of datasets. This lists includes datasets containing raw and/or processed data as well as empty datasets which only contain parameter files. You can click one or more datasets in the list to mark them for deletion and then click ***execute*** to actually delete them.

When entered without arguments, the ***del**** commands above show all datasets that are stored under the current *du* (disk unit) and *user*.

del displays a list of datasets, only showing the dataset name. The selected datasets are entirely deleted, including data files, parameter files and the data name directory. ***del*** takes a maximum of three arguments. Some examples of its usage are:

del
list all datasets under the current disk unit and user.

del exam1d*
list all datasets whose name starts with *exam1d* and which reside under the current disk unit and user.

del exam1d_?? C:\data joe
list all datasets whose name consists of *exam1d_* and 2 additional characters and which reside under disk unit *C:\data* and user *joe*.

del C:\nmr
list all datasets which reside under disk unit *C:\nmr* and the current user.

del C:\nmr joe
list all datasets which reside under disk unit *C:\nmr* and user *joe*.

del \\host_X\nmrsh joe
list all datasets which reside under user *joe* and disk unit *\\host_x\nmrsh* where *nmrsh* is the share name of a shared directory on computer *host_x*.

del /nmr joe
list all datasets which reside under disk unit */nmr* and user *joe* (on UNIX or LINUX computers).

dela lists datasets showing a separate entry for each experiment number (expno). Each entry contains the dataset *name*, *expno*, *type* and *size*. Datasets which do not contain raw data are displayed with the type "no raw data".

delp lists datasets showing a separate entry for each processed data number (procno). Each entry contains the dataset *name*, *expno*, *procno*, *type* and *size*. Datasets which do not contain processed data are displayed with the type "no processed data".

deldat prompts the user for a time range as specified in table 8.1. Depending

all	all data acquired by the current user
between	data acquired between two specified dates
day	data acquired on the specified date
earlier	data acquired before the specified date
later	data acquired later than the specified date

Table 8.1

on the time range you select, you are further prompted for specific date(s). A list of datasets which were measured within the specified time range is displayed with a separate entry for each experiment number (expno).

dela, ***delp*** and ***deldat*** offer a ***mode*** button which allows you to toggle between the following modes:

- *deleting data only*
  the data are removed but the data directory (including parameters) is kept. As such, the dataset is still available for a new acquisition. For

dela and ***deldat***, only the raw data are removed whereas possible processed data are kept.

- *deleting data + parameters*
  the entire data directory (including data and parameters) is removed. For ***dela***, this concerns the expno directory, including all procno subdirectories. For ***del*** this concerns the entire data name directory, including all expno and procno subdirectories.

If the ***del**** commands are entered without a user as an argument, then only the datasets of current user are deleted. The current user here refers to the *user* part of the data path of the foreground dataset.

Please distinguish:

- the user part of the data path
- the owner of the dataset
- the user who runs XWIN-NMR

Usually these three things are the same, i.e. a user works on his own data. However, the user part of the data path can be any character string and does not have to correspond to a user account on the computer. Furthermore, the user who runs XWIN-NMR might work on someone else's data. In this case, he/she may or may not have the permission to delete this dataset. In the latter case, the ***del**** commands will not delete the dataset but show an error message instead.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - 1D raw data
`ser` - 2D or 3D raw data

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`3rr`, `3irr`, `3rir`, `3iir` - processed 3D data

Note that when the ***mode*** "deleting data + parameters" is selected, the parameter files are also deleted.

OUTPUT FILES

For *`dela`* and *`deldat`*, both with *mode* = *deleting data only*:

<du>/data/<user>/nmr/<name>/<expno>/

`audita.txt` - acquisition audit trail

For *`delp`* with *mode* = *deleting data only*:

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`auditp.txt` - processing audit trail

SEE ALSO

delf, dels, delser, del2d, deli

delf, dels, delser, del2d, deli

NAME

delf - delete FIDs (1D raw data)
dels - delete spectra (1D processed data)
delser - delete serial data (2D and 3D raw data)
del2d - delete 2D processed data
deli - delete imaginary processed data

SYNTAX

delf [<name>] [<du> [<user>]]

dels, delser, del2d and deli have the same syntax as delf

DESCRIPTION

The commands ***delf***, ***dels***, ***delser***, ***del2d*** and ***deli*** display a list of datasets. This list only includes datasets which contain data files. As opposed to commands like ***del*** and ***dela***, they do not show empty datasets. You can click on one or more datasets to mark them for deletion and then click ***execute*** to actually delete them.

When entered without arguments, the ***del**** commands above show all datasets that are stored under the current *du* (disk unit) and *user*. For examples on their syntax, see the description of ***del***.

delf lists 1D datasets which contain raw data showing a separate entry for each experiment number (expno). Each entry contains the dataset *name*, *expno*, *type* and *size*.

dels lists 1D datasets which contain processed data showing a separate entry for each processed data number (procno). Each entry contains the dataset *name*, *procno*, *type* and *size*.

delser lists 2D datasets which contain raw data showing a separate entry for each experiment number (expno). Each entry contains the dataset *name*, *expno*, *type* and *size*.

del2d lists 2D datasets which contain processed data showing a separate entry for each processed data number (procno). Each entry contains the dataset *name*, *procno*, *type* and *size*.

deli lists datasets which contain 1D, 2D or 3D imaginary data showing a separate entry for each processed data number (procno). Each entry contains the dataset *name*, *expno*, *procno*, *type* and *size*. Only the imaginary processed data files are deleted. Raw data, processed data and parameter files are kept.

All these del commands (except for ***deli***) offer a ***mode*** button which allows you to toggle between the following modes:

- *deleting data only*
  the data are removed but the data directory (including parameters) is kept. As such, the dataset is still available for a new acquisition. For ***delf*** and ***delser***, only the raw data are removed whereas possible processed data are kept.
- *deleting data + parameters*
  the entire data directory (including data and parameters) is removed. For ***delf*** and ***delser***, this concerns the expno directory, including all procno subdirectories. For ***dels*** and ***del2d***, this concerns the procno directory.

If the ***del**** commands are entered without a user as an argument, then only the datasets of current user are deleted. The current user here refers to the *user* part of the data path of the foreground dataset. Please distinguish:

- the user part of the data path
- the owner of the dataset
- the user who runs XWIN-NMR

Usually these three things are the same, i.e. a user works on his own data. However, the user part of the data path can be any character string and does not have to correspond to a user account on the computer. Furthermore, the user who runs XWIN-NMR might work on someone else's data. In this case, he/she may or may not have the permission to delete this dataset. In the latter case, the ***del**** commands will not delete the dataset but show an error message instead.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - 1D raw data
`ser` - 2D or 3D raw data

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`3rrr`, `3irr`, `3rir`, `3iir` - processed 3D data

Note that when the ***mode*** "deleting data+parameters" is selected, the parameter files are also deleted.

OUTPUT FILES

For *`delf`* and *`delser`*, both with *mode* = *deleting data only:*

<du>/data/<user>/nmr/<name>/<expno>/

`audita.txt` - acquisition audit trail

For *`dels`* and *`del2d`* both with *mode* = *deleting data only* **and *`deli`*:**

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`auditp.txt` - processing audit trail

SEE ALSO

del, dela, delp, deldat

dir, dira, dirp, dirdat

NAME

dir - list data
dira - list acquisition data (raw data)
dirp - list processed data
dirdat - list data acquired at certain dates

SYNTAX

dir [<name>] [<du> [<user>]]

dira, dirp and dirdat have the same syntax as dir

DESCRIPTION

The commands ***dir***, ***dira***, ***dirp*** and ***dirdat*** display a list of datasets. The list includes datasets containing raw and/or processed data as well as empty datasets which only contain parameter files. When you click a dataset in the list, it is read and becomes the new foreground dataset.

When entered without arguments, the ***dir**** commands above show all datasets that are stored under the current *du* (disk unit) and *user*.

dir lists datasets, showing the data names only. When you click a dataset, it is read and becomes the new foreground dataset. If the selected dataset contains more than one expno, XWIN-NMR looks for the same expno as that of the current foreground dataset. If that does not exist, the lowest available expno is read. ***dir*** takes three arguments. Some examples of its usage are:

dir
list all datasets under the current disk unit and user.

dir exam1d*
list all datasets whose name starts with *exam1d* and which reside under the current disk unit and user.

dir exam1d_?? C:\data joe
list all datasets whose name starts with *exam1d_* and has 2 additional characters and which reside under disk unit *C:\data* and user *joe*.

dir C:\nmr

list all datasets and which reside under disk unit *C:\nmr* and the current user.

dir C:\nmr joe
list all datasets which reside under disk unit *C:\nmr* and user *joe*.

dir \\host_X\nmrsh joe
list all datasets which reside under user *joe* and disk unit *\\host_x\nmrsh* where *nmrsh* is the share name of a shared directory on computer *host_x*.

dir /nmr joe
list all datasets which reside under disk unit */nmr* and user *joe* (on UNIX or LINUX computers).

dira lists datasets showing a separate entry for each expno. Each entry contains the dataset name, expno, type and size. The type refers to the name of the data files and can be *fid* (1D raw data), *ser* (2D or 3D raw data) or *no raw data*. When you click a dataset, it is read and become the new foreground dataset. If more than one procno exist, XWIN-NMR looks for the same procno as that of the current foreground dataset. If that does not exist, the lowest available procno is read.

dirp lists datasets showing a separate entry for each processed data number (procno). Each entry contains the dataset name, expno, procno, type and size. The type refers to the name of the data files and can be *1r 1i* (processed 1D data), *2rr 2ir 2ri 2ii* (2D raw data), *3rrr, 3rri*, ..(processed 3D data) or *no processed data*. When you click a dataset, it is read and becomes the new foreground dataset.

dirdat prompts the user for a time range as specified in table 8.2. Depending

all	all data acquired by the current user
between	data acquired between two specified dates
day	data acquired on the specified date
earlier	data acquired before the specified date
later	data acquired later than the specified date

Table 8.2

on the time range you select, you are further prompted for specific date(s). A list of datasets which were measured within the specified time range is displayed with a separate entry for each expno.

If you specify an argument, then it may contain wildcards; for example ***dir a****** lists all datasets starting with *a* and ***dir cholac??*** all datasets called *cholac* plus two additional characters.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - 1D raw data
`ser` - 2D or 3D raw data

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`3rrr`, `3irr`, `3rir`, `3iir` - processed 3D data

SEE ALSO

dirf, dirs, dirser, dir2d, search, browse, re, rep

dirf, dirs, dirser, dir2d

NAME

dirf - list FIDs (1D raw data)
dirs - list spectra (1D processed data)
dirser - list serial data (2D and 3D raw data)
dir2d - list 2D processed data

SYNTAX

dirf [<name>] [<du> [<user>]]

dirs, dirser and dir2d have the same syntax as dirf

DESCRIPTION

The commands ***dirf***, ***dirs***, ***dirser*** and ***dir2d*** display a list of datasets. This list only includes datasets which contain certain data files. As opposed to commands like ***dir*** and ***dira***, they do not show empty datasets. When you click a dataset in the list, it is read and becomes the new foreground dataset.

When entered without arguments, the ***dir**** commands above show all datasets that are stored under the current *du* (disk unit) and *user*. For examples on their syntax, see the description of ***dir***.

dirf lists 1D datasets which contain raw data showing a separate entry for each experiment number (expno). Each entry contains the dataset *name*, *expno*, *type* and *size*.

dirs lists 1D datasets which contain processed data showing a separate entry for each processed data number (procno). Each entry contains the dataset *name*, *procno*, *type* and *size*.

dirser lists 2D and 3D datasets which contain raw data showing a separate entry for each experiment number (expno). Each entry contains the dataset *name*, *expno*, *type* and *size*.

dir2d lists 2D datasets which contain processed data showing a separate entry for each processed data number (procno). Each entry contains the dataset *name*, *procno*, *type* and *size*.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - 1D raw data
`ser` - 2D or 3D raw data

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`3rrr`, `3irr`, `3rir`, `3iir` - processed 3D data

SEE ALSO

dir, dira, dirp, dirdat, search, browse, re, rep

diro

NAME

diro - list owners (user part of the data path)

SYNTAX

diro [<user>] [<du>]

DESCRIPTION

The command ***diro*** displays a list of users. Here, user refers to the <users> part of the data path like:

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

When entered without argument, the users under the current (foreground) disk unit (du) are displayed. After selecting a user, ***diro*** opens the first dataset of this user and makes it the current dataset.

diro takes two arguments. These allow you to select users under a different disk unit or to use wild cards. Here are some examples:

diro
list all users under the current disk unit (du)

diro a*
list all users under the current disk unit that start with *a*.

diro a??
list all users under the current disk unit that start with *a* and have 2 additional characters

diro C:\nmr
list all users under disk unit *C:\nmr*

diro /nmr
list all users under disk unit */nmr* on a UNIX or LINUX system

diro j* C:\nmr
list all users under disk unit *C:\nmr* that start with *j*

Note that the users shown by ***diro*** are not necessarily user accounts. If fact, they can be any character string. Please distinguish the following things:

- the user who is running XWIN-NMR
- the user part of the data path
- the owner of a dataset

Often the user who runs XWIN-NMR works on his/her own data in a data path with his/her own name and these three items are the same.

INPUT DIRECTORIES

<du>/data/* - user part of the data path

SEE ALSO

dir, dira, dirp, dirdat, dirf, dirs, dirser, dir2d, search, browse, re, rep

edc

NAME

edc - define a new dataset

DESCRIPTION

The command ***edc*** opens a dialog box in which you can define a new dataset, including *disk unit*, *user*, *data name*, *expno* and *procno*. If the specified dataset already exists, it becomes the new foreground dataset, i.e. it is displayed. If it does not exist, it is created as a copy of the current foreground dataset. Note that only the parameter files are copied, not the data files.

edc is identical to the command ***new***.

INPUT FILES

<xwhome>/prog/curdir/<user>/

`curdat` - current dataset parameters

If the dataset specified with ***edc*** does not exist yet, the current dataset is copied:

<du>/data/<user>/<name>/nmr/<expno>/

`acqu` - acquisition parameters
`acqus` - acquisition status parameters

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`proc` - processing parameters
`procs` - processing status parameters

OUTPUT FILES

<xwhome>/prog/curdir/<user>/

`curdat` - current dataset parameters

If the dataset specified with ***edc*** does not exist yet, the current dataset is copied:

<*du*>/data/<*user*>/<*name*>/nmr/<*expno*>/

`acqu` - acquisition parameters

`acqus` - acquisition status parameters

<*du*>/data/<*user*>/<*name*>/nmr/<*expno*>/pdata/<*procno*>/

`proc` - processing parameters
`procs` - processing status parameters

For 2D and 3D data the files `acqu2`, `acqu2s` etc. are also output.

SEE ALSO

new, edc2, search, browse, re, rep, dir, wra, wrp, wrpa, wrd

edc2

NAME

edc2 - define the *second* and *third* dataset

DESCRIPTION

The command ***edc2*** opens a dialog box in which you can define the so called *second* and *third* dataset. These are used for dual display and several algebra commands such as ***add*** and ***mul***. Note that the second and as third dataset can be the same. Furthermore, the second and/or third dataset can be the same as the current (first) dataset. This is, for example, used to add the second dataset to the current dataset and store the result in the current dataset.

In XWIN-NMR 3.1 and newer, ***edc2*** can also be used to define the external projections of a 2D dataset. In previous versions, this was done with the ***edg*** command.

Although ***edc2*** is normally used without arguments, it can take the argument ***used_from***. In that case, a dialog box appears in which you can specify a 2D dataset. Once you have done that, you can switch to that 2D dataset simply by clicking the ***2D*** menu button. It can also be used as follows:

1. read a 1D dataset
2. enter ***edc2 used_from*** and specify a 2D dataset
3. enter ***rsr*** to read a row or ***rsc*** to read a column from that 2D dataset

Note that you can skip step 2 if the current 1D dataset is already a row or column that was extracted from a 2D dataset and you simply want to read another row/column from the same dataset.

INPUT AND OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`curdat2` - definition of second and third dataset
`used_from` - definition of 2D dataset (input of ***edc2 used_from***)

SEE ALSO

edc, new, dual, add, mul, rsr, rsc, rser

edo

NAME

edo - edit the output device parameters

DESCRIPTION

The command ***edo*** opens a dialog box in which you can define the output device parameters of the current dataset. These parameters are:

CURPLOT - the printer or plotter used by the ***plot****** commands
PFORMAT - format file for plotting parameters with the ***plot****** commands
DFORMAT - format file for ***dpa***, ***dpc***, ***dpg***, ***dpgx***, ***dpo***, ***dpp*** and ***dp***
LFORMAT - format file for ***lpa***, ***lpc***, ***lpg***, ***lpgx***, ***lpo***, ***lpp*** and ***lp***
CURPRIN - the printer used by ***lpa***, ***lpc***, ***lpg***, ***lpgx***, ***lpo***, ***lpp*** and ***lp***
LAYOUT - the XWIN-PLOT layout used by the ***autoplot*** command

You can either type in the names of these parameters or click the small arrow to the right of the parameter fields. The latter offers you all currently available entries.

The parameters PFORMAT, DFORMAT and LFORMAT are usually set to their default values, `normpl`, `normdp` and `normlp`, respectively. You can, however, create new files in the respective directories (see below) and set the corresponding parameters to them. Note that the commands which use these files might fail after installing a new XWIN-NMR version. If that happens, Bruker has changed something in the format files and you must also make this change in your own files. You can do that by comparing your format files with the default format files.

The ***edo*** settings only count for the current dataset. If you want to defined a printer or plotter for all datasets, you can do that in the User Interface. Enter the command ***setres*** and specify the printer name in the field ***Plotter***. The setting of the plotter in the User Interface overrules the value of CURPLOT in ***edo***.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`outd` - output device parameters

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`outd` - output device parameters

SEE ALSO

dpo, lpo, setres

paste

NAME

paste - read a dataset from the Windows Explorer

DESCRIPTION

The command ***paste*** reads an XWIN-NMR dataset which was previously selected from the Windows Explorer. This involves the following steps:

In the Explorer:

- Go to a dataset
- Right-click a dataset folder or file, e.g. the data name, expno or procno folder or any file in it
- Click **Copy**

In XWIN-NMR:

- Click ***File*** → ***Paste*** or type ***paste***

Note that you can select any part of a dataset in the Explorer, file or folder. For some files, ***Paste*** will not only read the corresponding dataset but also perform a certain action. For example, when you select the file `fid`, the ***Paste*** command will automatically switch to the acquisition menu. If you select the file `level` file in a 2D dataset, ***Paste*** will call the command ***edlev***.

Note that if you select and copy a the dataset in the Explorer, its data path is copied to the Clipboard. The command ***Paste*** reads this path from the Clipboard. If you run ***Paste*** without first copying a dataset from the Explorer, XWIN-NMR tries to read whatever is currently stored in the Clipboard. If that is a data path, XWIN-NMR will read it, otherwise you will get an error message.

INPUT FILES

<du>/data/<user>/<name1D>/nmr/<expno>/

`fid` - 1D raw data

<du>/data/<user>/<name1D>/nmr/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data

<du>/data/<user>/<name2D>/nmr/<expno>/

`ser` - 2D raw data

<du>/data/<user>/<name2D>/nmr/<expno>/pdata/<procno>/

`2rr`, `2ir`, `2ri`, `2ii` - processed 2D data
`level` - 2D contour levels

Furthermore the parameter files `acqu`, `acqus`, `proc` etc. can be selected in the Explorer and are read as a part of the entire dataset.

SEE ALSO

search, browse, re, rep, dir, dira, dirp, dirdat, dirf, dirs, dirser, dir2d

re, rep

NAME

re - read data name or experiment number (expno)
rep - read data processed data number (procno)

DESCRIPTION

The command ***re*** allows you to read and display a new dataset. It takes five arguments; the five variable parts of a data path:

<*du*>/data/<*user*>/nmr/<*name*>/<*expno*>/<*procno*>

Here are some examples of you can use ***re*** :

```
re <name> <expno> <procno> <du> <user>
re <name>
re <expno>
re <name> <expno>
re <name> <expno> <procno>
re <expno> <procno>
re
```

Note that the first alphanumeric argument is always interpreted as the name and the first numeric argument as experiment number.

If you enter ***re*** without an argument, you will be prompted for the data path and you can enter all or only some of the arguments as shown above.

The command ***rep*** allows you to read a new processed data number (procno) of the current dataset. It takes only one argument; the destination procno.

INPUT FILES

<du>/data/<user>/nmr/<name1D>/<expno>/

`fid` - 1D raw data
`acqu` - acquisition parameters
`acqus` - acquisition status parameters

<du>/data/<user>/<name1D>/nmr/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`proc` - processing parameters

`procs` - processing status parameters
`meta` - plot parameters for ***`plot`*** and ***`view`***

Note that these are only the main files of a 1D dataset.

SEE ALSO

search, browse, dir, dira, dirp, dirdat, dirf, dirs, dirser, dir2d

ren, reno

NAME

ren - rename a dataset name
reno - rename dataset user

DESCRIPTION

The command ***ren*** opens a dialog box with all dataset names in the current data path (current *user* and *disk unit*). You can rename one or more datasets, simply by replacing their names. When you enter a new name followed by Enter, the dataset is immediately renamed. The dialog box can be closed by clicking ***Cancel***.

reno works like ***ren*** except that it allows you to rename a dataset *user* instead of a dataset *name*. Note that *user* refers to a part of the data path. The owner of the dataset on operating system level remains the same.

INPUT DIRECTORIES

For ***ren***:

<du>/data/<user>/nmr/<*old_name*>/<expno>/pdata/<procno>

For ***reno***:

<du>/data/<*old_user*>/nmr/<name>/<expno>/pdata/<procno>

OUTPUT DIRECTORIES

For ***ren***:

<du>/data/<user>/nmr/<*new_name*>/<expno>/pdata/<procno>

For ***reno***:

<du>/data/<*new_user*>/nmr/<name>/<expno>/pdata/<procno>

SEE ALSO

rengp, renmac, renpul, renau, renpar

search

NAME

search - search for dataset

DESCRIPTION

The command ***search*** opens a portfolio editor in which you can you find datasets and select them for display. The editor contains a field for each part of the data path; Directory, User, Name, Expno, Procno. You can select a dataset, simply by clicking an entry in each successive field or accept the default selection. Then click ***Append*** to add the dataset to the Portfolio field and ***Apply*** to actually read the dataset in XWIN-NMR.

The command ***search*** loads the default portfolio which does not necessarily contain all your data directories. If one or more data directories are missing, you can add them as follows:

1. Click ***Edit*** → ***Edit Directory List***
2. A window 'Directory List' will appear:
   a) In the field Directory: enter your data directory, e.g. `/w` or `D:\`
   b) Click ***Add*** → ***OK***
3. In the window 'Portfolio Editor':
   - Click ***File*** → ***Save as Default***

   or

   - Click ***File*** → ***Save as ...***

   and choose you own directory and portfolio name.

The latter part of step 3 allows you to keep several portfolios.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - 1D raw data
`acqu` - acquisition parameters
`acqus` - acquisition status parameters

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`proc` - processing parameters
`procs` - processing status parameters
`meta` - plot parameters for ***plot*** and ***view***

Note that these are only the main 1D data files.

SEE ALSO

re, rep, browse, dir, dira, dirp, dirdat, dirf, dirs, dirser, dir2d

wrpa, wra, wrp, wrd

NAME

wrpa - copy a dataset
wra - copy raw data
wrp - copy processed data
wrd - copy a dataset to a different disk unit

DESCRIPTION

The command ***wrpa*** copies the current dataset to a new data name or experiment number (expno). The entire expno directory is copied including raw data, acquisition parameters, processed data and processing parameters.

wrpa takes six arguments:

<name> - the dataset name
<expno> - the experiment number
<procno> - the processed data number
<du> - the disk unit (data directory)
<user> - the user
y - overwrite the destination dataset if it already exists

All arguments are parts of the destination data path[1], except for the last one which is a flag. You can, but do not have to, specify all of these arguments. If the first argument is a character string, it is interpreted as the destination data name. If the first argument is an integer value, it is interpreted as the destination experiment number. Examples of using ***wrpa*** are:

```
wrpa <name> <expno> <procno> <du> <user> y
wrpa <name>
wrpa <expno>
wrpa <name> <expno>
wrpa <name> <expno> <procno>
wrpa <expno> <procno>
wrpa
```

When ***wrpa*** is entered without arguments, you are prompted for *name*, *expno*, *procno*, *du* and *user*. You can enter one or more arguments in the dialog box, as

1. The data path of the foreground dataset is displayed above the XWIN-NMR data field

described above for the command line arguments.

Note that ***wrpa*** only works if user who started XWIN-NMR has the permission to create the destination dataset. This especially plays a role if a new destination *user* is specified.

wra makes a copy of the current expno directory, including raw data, acquisition parameters, and processing parameters. The command takes two arguments and can be used as follows:

wra - prompts you for the destination experiment number
wra <expno> - copies the raw data to <expno>
wra <expno> y - overwrites existing raw data in <expno>

wrp makes a copy of the current procno directory, including the processed data and processing parameters. The command takes two arguments and can be used as follows:

wrp - prompts you for the destination processed data number
wrp <procno> - copies processed data to <procno>
wrp <procno> y - overwrites existing processed data in <procno>

wrd copies an entire dataset to a different disk unit. It is, for example, useful for archiving datasets. It takes only one argument, the destination disk unit. All other parts of the data path, *name*, *expno*, *procno*, and *user* remain the same.

INPUT AND OUTPUT FILES

For ***wra***, ***wrpa***, and ***wrd***:

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - 1D raw data
`acqu` - acquisition parameters
`acqus` - acquisition status parameters

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`proc` - processing parameters
`procs` - processing status parameters
`meta` - plot parameters for ***plot*** and ***view***
`auditp.txt` - processing audit trail

For ***wrp***:

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`1r`, `1i` - processed 1D data
`proc` - processing parameters
`procs` - processing status parameters
`meta` - plot parameters for ***`plot`*** and ***`view`***
`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

WRA(expno)

WRP(procno)

WRPA(name, expno, procno, diskunit, user)

Note that these macros overwrite possibly exiting data.

SEE ALSO

edc, re, rep

Chapter 9

Parameters, lists, AU programs

This chapter describes all XWIN-NMR commands which handle parameters and parameter sets. Furthermore, you will find commands which are used to read or edit lists like pulse programs, gradient programs, frequency lists etc. and, finally, commands which are used to read, edit or run AU programs. Note that several commands in this chapter are acquisition related rather than processing related. Nevertheless they play a role in the processing part of XWIN-NMR

compileall

NAME

compileall - compile Bruker and User AU programs

DESCRIPTION

The command ***compileall*** compiles all Bruker and User AU programs. In order to compile Bruker AU programs, these must have been installed. This can be done with the command ***expinstall***, with the option "Install Bruker library AU programs/modules" enabled.

For more information on AU programs please refer to the AU reference manual.

INPUT FILES

<xwhome>/exp/stan/nmr/au/src/*

AU programs (source files)

OUTPUT FILES

<xwhome>/prog/au/bin/*

AU programs (executable files)

For more information on AU programs please refer to the AU reference manual.

SEE ALSO

expinstall, cplbruk, cpluser, edau, xau, xaua, xaup, delau, renau

cplbruk, cpluser

NAME

cplbruk - compile Bruker AU programs
cpluser - compile user defined AU programs

SYNTAX

cplbruk [<name> | all]
cpluser [<name> | all]

DESCRIPTION

The command ***cplbruk*** allows you to compile one or more Bruker AU programs. Before you can use it, the command ***expinstall*** must have been executed once, with the option "Install Bruker library AU programs/modules" enabled. Then you can use ***cplbruk*** in three different ways:

cplbruk <name> - compile the Bruker AU program <name>
cplbruk all - compile all Bruker AU programs
cplbruk - a list of Bruker AU programs appears; click one to compile it

If you specify an argument, then it may contain wildcards; for example ***cplbruk a**** compiles all Bruker AU programs which start with *a*.

The command ***cpluser*** works like ***cplbruk***, except that it compiles user defined AU programs.

For more information on AU programs please refer to the AU reference manual.

INPUT FILES

<xwhome>/exp/stan/nmr/au/src/*

AU programs (source files)

OUTPUT FILES

<xwhome>/prog/au/bin/*

AU programs (executable files)

For more information on AU programs please refer to the AU reference manual.

SEE ALSO

expinstall, compileall, edau, xau, xaua, xaup, delau, renau

delpar, delpul, delgp, delsh, delau, delmac

NAME

delpar - delete parameter sets
delpul - delete pulse programs
delgp - delete gradient programs
delsh - delete shim files
delmac - delete macros
dellut - delete 2D lookup tables
delau - delete AU programs

SYNTAX

delpar [<name>]

delpul, delgp, delsh, delmac, dellut and delau have the same syntax as delpar

DESCRIPTION

The command ***delpar*** displays a list of parameter sets, both Bruker and user defined. Each entry shows the parameter set name and the parameter types in that set. You can mark one or more parameter sets for deletion and then click ***Execute*** to actually delete them. Furthermore, you can print the list by clicking the ***Print*** button. If you have accidentally removed Bruker parameter sets, you can re-install them with the command ***expinstall***.

The other ***del**** commands mentioned here all work like ***delpar***, deleting the type of files as specified above. Note that ***delau*** deletes both the selected AU source files and the corresponding executable files. If you accidentally delete Bruker AU programs, pulse programs or gradient programs, you can re-install them with ***expinstall***. In case of AU programs, you must also compile them with ***cplbruk*** or ***xau.***

INPUT DIRECTORIES

<xwhome>/exp/stan/nmr/par - Bruker and user defined parameter sets

<xwhome>/exp/stan/nmr/lists/pp - Bruker and user defined pulse programs

<xwhome>/exp/stan/nmr/lists/gp - Bruker and user defined gradient programs

<xwhome>/exp/stan/nmr/lists/bsms - shim files

<xwhome>/exp/stan/nmr/lists/mac - Bruker and user defined macros

<xwhome>/exp/stan/nmr/au/src - Bruker and user defined source AU programs

<xwhome>/prog/au/bin - Bruker and user defined binary AU programs

USAGE IN AU PROGRAMS

DELPAR(name)

No AU macros are available for the other ***del****** commands.

SEE ALSO

dellist, delmisc

dellist

NAME

dellist - delete various lists

SYNTAX

dellist [<name>]

DESCRIPTION

The command ***dellist*** displays a list of various lists types. Most of them are used in acquisition, e.g. ***vd*** lists contain variable delays which are read by the ***vd*** command in pulse programs. When you click a list type, the available files of that type appear. You can click individual entries to mark them for deletion or click the button ***Select all***. Clicking the ***Execute*** button deletes all marked entries.

All lists which appear with ***dellist*** can be printed by clicking the ***Print*** button.

INPUT FILES

<xwhome>/exp/stan/nmr/lists

`pp` - pulse programs
`cpd` - CPD programs
`gp` - gradient programs
`ds` - dataset lists
`vc` - variable counter lists
`vd` - variable delay lists
`vp` - variable pulse lists
`vt` - variable temperature lists
`f1` - frequency lists
`f2`, `f3` - frequency lists (A*X spectrometers only)
`mac` - XWIN-NMR macros
`roi` - 2D integral regions
`scl` - scaling region files
`masr` - MASR rotation values

SEE ALSO

delmisc, delpar, delpul, delgp, delsh, delau, delmac

delmisc

NAME

delmisc - delete integral, baseline or peak lists

SYNTAX

delmisc [<name>]

DESCRIPTION

The command ***delmisc*** displays a list of five list types, *intrng*, *base_info*, *baslpnts*, *peaklist* and *reg*. These lists are described with the command ***edmisc***. When you click a list type, the available lists of that type appear. You can click one or more entries for deletion and then click ***Execute*** to actually delete them.

All lists which appear with ***delmisc*** can be printed by clicking the ***Print*** button.

INPUT FILES

<xwhome>/exp/stan/nmr/lists/

`intrng` - integral regions for ***view***, ***plot***, ***li***, ***lipp*** etc.
`base_info` - baseline correction coefficients for ***bcm***
`baslpnts` - baseline points for spline baseline correction with ***sab***
`peaklist` - peak list created with ***ppp*** for ***mdcon***
`reg` - plot regions used when PSCAL=ireg or pireg or when LIMITS=region

SEE ALSO

dellist, delpar, delpul, delgp, delsh, delau, delmac

dirpar

NAME

dirpar - list parameter sets

DESCRIPTION

The command ***dirpar*** displays a list of parameter sets, both Bruker and user defined. Each entry shows the parameter set name and the parameter types within that set. You can print the list by clicking the ***Print*** button.

Bruker parameter sets only appear in the list when the command ***expinstall*** has been executed, with the option ***Convert Standard Parameter Sets*** selected.

INPUT FILES

`<xwhome>/exp/stan/nmr/par/*`

Bruker and user defined parameter sets

SEE ALSO

expinstall, rpar, wpar, delpar

dp

NAME

dp - display status parameters

DESCRIPTION

The command ***dp*** shows the status parameters of the current dataset on the screen. These include the dataset, output device, acquisition, processing and plotting parameters. ***dp*** is a combination of the commands ***dpc***, ***dpo***, ***dpa***, ***dpp*** and ***dpg***.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`acqus` - acquisition status parameters

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`procs` - processing status parameters
`meta` - plot parameters for ***plot*** and ***view***
`outd` - output device parameters

<xwhome>/prog/curdir/<user>/

`curdat` - current data parameters

<xwhome>/exp/stan/nmr/form/curd.l/`normdp` - format file for ***dpc***
<xwhome>/exp/stan/nmr/form/outd.l/`normdp` - format file for ***dpo***
<xwhome>/exp/stan/nmr/form/acqu.l/`normdp` - format file for ***dpa***
<xwhome>/exp/stan/nmr/form/proc.l/`normdp` - format file for ***dpp***
<xwhome>/exp/stan/nmr/form/plot.l/`normdp` - format file for ***dpg***

SEE ALSO

dpc, dpo, dpa, dpp, dpg, dpgx

dpa

NAME

dpa - display the acquisition status parameters

DESCRIPTION

The command ***dpa*** displays the acquisition status parameters on the screen. The acquisition status parameters are set by acquisition commands and represent the status of the raw data.

The ***dpa*** dialog box offers the following buttons/fields:

- ***Done***
  Close the ***dpa*** window or, on 2D or 3D data, go to the next dimension
- ***1-Col***
  change to one column display mode
- ***Parameter***
  enter a parameter name (or a part of it), press **Enter** to find and select it
- ***Next***
  select the next parameter which starts with the string in the ***Parameter*** field

Acquisition status parameters can also be viewed by entering their names on the command line. For example:

On a 1D dataset:

1s ns
display the acquisition status parameter NS

On a 2D dataset:

2s td
display the F2 acquisition status parameter TD

1s td
display the F1 acquisition status parameter TD

On a 3D dataset, the preposition ***3s*** can be used for the F3 dimension.

The NMR Superuser can change the ***dpa*** dialog box, for example remove parameter which are not used. This must be done from the Windows Explorer or

from a UNIX shell by editing the file `normdp` (see below).

INPUT FILES

<xwhome>/exp/stan/nmr/form/*acqu.l*/

`normdp` - format file for ***dpa***

On 2D and 3D data the directories *acqu2.l* and *acqu3.l* contain a `normdp` file for the second and third dimension, respectively.

<du>/data/<user>/<name>/nmr/<expno>/

`acqus` - acquisition status parameters

On 2D and 3D data the files `acqu2s` and `acqu3s` are used for the second and third dimension, respectively (see also chapter 2.3).

SEE ALSO

eda, lpa, dp, dpc, dpo, dpp, dpg, dpgx

dpc

NAME

dpc - show the current dataset parameters on the screen

DESCRIPTION

The command ***dpc*** shows the current dataset parameters on the screen.

The current data parameters can be set up with ***edc*** and printed with ***lpc***.

INPUT FILES

<xwhome>/prog/curdir/<user>/

`curdat` - current dataset parameters

<xwhome>/exp/stan/nmr/form/curd.l/

`normdp` - format file for ***dpc***

SEE ALSO

edo, lpo, dp, dpo, dpa, dpp, dpg, dpgx

dpg

NAME

dpg - display the plot parameters

DESCRIPTION

The command ***dpg*** displays the plot parameters on the screen. Only a small subset of all plot parameters, the ones which also appear on a plot, are displayed. All plot parameters can be viewed and modified with ***edg***.

INPUT FILES

<xwhome>/exp/stan/nmr/form/*plot.l*/

`normdp` - format file for ***dpg***

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`meta` - plot parameters for ***plot*** and ***view***

SEE ALSO

edg, lpg, dp, dpc, dpo, dpa, dpp, dpgx

dpgx

NAME

dpgx - display the extended plot parameters

DESCRIPTION

The command ***dpgx*** displays the extended plot parameters on the screen.

INPUT FILES

<xwhome>/exp/stan/nmr/form/*plotx.l*/

`normdp` - format file for ***dpgx***

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`meta.ext` - extended plot parameters for ***plotx***

SEE ALSO

edgx, lpgx, dp, dpc, dpo, dpa, dpp, dpg

dpo

NAME

dpo - display the output device parameters

DESCRIPTION

The command ***dpo*** displays the output device parameters on the screen.

The output device parameters can be set up with ***edo*** and printed with ***lpo***.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`outd` - output device parameters

<xwhome>/exp/stan/nmr/form/outd.l/

`normdp` - format file for ***dpo***

SEE ALSO

edo, lpo, dp, dpc, dpa, dpp, dpg, dpgx

dpp

NAME

dpp - display the processing status parameters

DESCRIPTION

The command ***dpp*** displays the processing status parameters on the screen. The processing status parameters are set by processing commands and represent the status of the processed data.

The ***dpp*** dialog box offers the following buttons/fields:

- ***Done***
  Close the ***dpp*** window or, on 2D or 3D data, go to the next dimension
- ***1-Col***
  change to one column display mode
- ***Parameter***
  allows you to search for a parameter. Just enter the parameters name (or a part of it) and hit the **Enter** key. The ***dpp*** window will scroll to parameters position. If nothing happens, the parameter does not exist or is already on the current page.
- ***Next***
  select the next parameter which starts with the string in the ***Parameter*** field

Processing status parameters can also be viewed by entering their names on the command line. For example:

On a 1D dataset:

1s ft_mod
display the processing status parameter FT_mod

On a 2D dataset:

2s ft_mod
display the F2 processing status parameter FT_mod

1s ft_mod
display the F1 processing status parameter FT_mod

On a 3D dataset, the preposition ***3s*** can be used for the F3 dimension.

The NMR Superuser can change the ***dpp*** dialog box, for example remove parameter which are not used. This must be done from the Windows Explorer or from a UNIX shell by editing the file `normdp` (see below).

INPUT FILES

<xwhome>/exp/stan/nmr/form/*proc.l*/

`normdp` - format file for ***dpp***

On 2D and 3D data the directories *proc2.l* and *proc3.l* contain a `normdp` file for the second and third dimension, respectively.

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`procs` - processing status parameters

On 2D and 3D data the files `proc2s` and `proc3s` are used for the second and third dimension, respectively (see also chapter 2.3).

SEE ALSO

edp, lpp, dp, dpa, dpc, dpo, dpg, dpgx

edau

NAME

edau - create or edit AU programs

SYNTAX

edau [<name>]

DESCRIPTION

The command ***edau*** allows you to list, create or edit AU programs. When used without argument, a two column list appears with the Bruker AU programs on the right side and the user defined AU programs on the left side. When you click a Bruker AU program, it is shown in view mode which means it cannot be modified. When you click a user defined AU program, it is opened with an editor and can be modified. When you close the view window or editor, you are prompted for one of the following options:

(c)ompile - compile the AU program
(b)ack - go back to displaying the list of AU programs
(q)uit - quit without compiling the AU program

When you choose *c* for compiling, the AU program will be compiled using a C compiler. After successful compilation, the AU program can be executed by typing its name.

When ***edau*** is entered with an argument, the specified AU program will be opened. This allows you to write a new AU program or modify an existing one. The argument may contain wildcards, e.g. ***edau a**** displays a list of all AU programs which start with *a*.

Bruker AU programs must be installed once with ***expinstall*** before they can be opened with ***edau***. The installation must be repeated when a new version of XWIN-NMR is installed. By default, Bruker AU programs are opened in view mode, which means they cannot be modified. However, if you enter ***edau*** and click the ***Edit*** button, the NMR Superuser password is requested and the program switches to *Edit* mode. When you now click a Bruker AU program, it will be opened with an editor and can be modified. Nevertheless, we recommend to leave the Bruker AU programs unchanged. If you want a modified version, just create a new AU program, read in the Bruker AU program, modify it to your

needs and store it.

edau uses the editor which is defined in the XWIN-NMR User Interface. If you want to use a different editor, type ***setres*** and modify the entry ***Editor***.

For details on writing, compiling, and executing AU programs please refer to the AU reference manual (available as XWIN-NMR online help).

INPUT FILES

`<xwhome>/exp/stan/nmr/au/src/*`

AU program source files

OUTPUT FILES

`<xwhome>/exp/stan/nmr/au/src/*`

AU program source files

`<xwhome>/prog/au/bin/*`

AU program executable binary files

SEE ALSO

expinstall, comepileall, cpluser, cplbruk, delau, renau

edcgp

NAME

edcgp - edit current gradient program

SYNTAX

edcgp [<name>]

DESCRIPTION

The command ***edcgp*** allows you to create or edit the current gradient program. The current gradient program is defined as the gradient program of the foreground dataset as defined by the acquisition parameter GRDPROG.

edcgp takes one argument and can be used as follows:

- ***edcgp***
  open the current gradient program
- ***edcgp <name>***
  open the gradient program <name> and make it the current gradient program.

If you specify an argument, then it may contain wildcards; for example:

edcgp grad* lists all gradient programs beginning with grad
edcpul [m-z]* lists all gradient programs beginning with m,n,...,z

INPUT PARAMETERS

set by the user with ***eda*** or by typing ***grdprog*** :

GRDPROG - the current gradient program (input of ***edcgp***)

OUTPUT PARAMETERS

can be viewed with ***eda*** or by typing ***grdprog*** :

GRDPROG - the current gradient program (output of ***edcgp <name>***)

INPUT FILES

<xwhome>/exp/stan/nmr/lists/gp/*

gradient programs

<du>/data/<user>/nmr/<name>/<expno>/

`acqu` - acquisition parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`acqu` - acquisition parameters

SEE ALSO

edgp

edcpd

NAME

edcpd - edit composite pulse decoupling (CPD) programs

SYNTAX

edcpd [<name>]

DESCRIPTION

The command ***edcpd*** allows you to list, create or edit CPD programs. If you enter ***edcpd*** without arguments, a list of all CPD programs is displayed. The list includes both the Bruker and the user defined CPD programs. When you click on a CPD program, it is opened with an editor. Alternatively, or you can enter a name in the field "Type New Name" to create a new CPD program. The ***Print*** button allows you to print the list of CPD programs.

If you enter the command with an argument, e.g. ***edcpd <name>***, the CPD program <name> is opened, if it exists, or otherwise it is created. The argument may contain wildcards; e.g. ***edcpd a**** displays a list of all CPD programs which start with *a*.

Bruker CPD programs must be installed with ***expinstall*** before they can be opened with ***edcpd***.

edcpd uses the editor which is defined in the XWIN-NMR User Interface. You can change it by typing ***setres*** and specify it in the field ***Editor***.

INPUT AND OUTPUT FILES

<xwhome>/exp/stan/nmr/lists/cpd/*

Bruker and user defined CPD programs

SEE ALSO

expinstall

edcpul

NAME

edcpul - edit the current pulse program

SYNTAX

edcpul [<name>]

DESCRIPTION

The command ***edcpul*** allows you to create or edit the current pulse program. The current pulse program is defined as the pulse program of the foreground dataset as defined by the acquisition parameter PULPROG.

edcpul takes one argument and can be used as follows:

- ***edcpul***
  open the current pulse program
- ***edcpul <name>***
  open the pulse program <name> and make it the current pulse program.

Bruker pulse programs are opened in view mode which means they cannot be modified. User defined pulse programs are opened with an editor and can be modified.

If you specify an argument, then it may contain wildcards; for example:

edcpul cos* lists all pulse programs beginning with cos
edcpul [m-z]* lists all pulse programs beginning with m,n,...,z

INPUT PARAMETERS

set by the user with ***eda*** or by typing ***pulprog*** :

PULPROG - the current pulse program (input of ***edcpul***)

OUTPUT PARAMETERS

can be viewed with ***eda*** or by typing ***pulprog*** :

PULPROG - the current pulse program (output of ***edcpul <name>***)

INPUT FILES

<xwhome>/exp/stan/nmr/lists/pp

Bruker and user defined pulse programs

<du>/data/<user>/nmr/<name>/<expno>/

`acqu` - acquisition parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`acqu` - acquisition parameters

SEE ALSO

edpul

eddosy

NAME

eddosy - edit DOSY processing parameters

The command ***eddosy*** opens a dialog box in which you can set DOSY processing parameters. These parameters are used by the command ***dosy2d*** and ***dosy3d*** on 2D and 3D data, respectively.

The ***eddosy*** dialog box offers the following buttons/fields:

- ***Save***
  save all parameters
- ***2-Col***
  change to two column display mode
- ***Parameter***
  allows you to search for a parameter. Just enter the parameters name (or a part of it) and hit the **Enter** key. The dialog box will scroll to parameters position. Note that the parameter might be on the current page.
- ***Next***
  select the next parameter which starts with the string in the ***Parameter*** field
- ***Cancel***
  leave ***eddosy*** without saving any changes

The NMR Superuser can change the ***eddosy*** dialog box, for example remove parameter which are not used. This must be done on operating system level by editing the file `dosy.e` (see below).

For more information on ***eddosy***, refer to the manual "DOSY and Diffusion" under ***Help*** → ***Other topics***.

INPUT FILES

<xwhome>/exp/stan/nmr/form/

`dosy.e` - format file for ***eddosy***

INPUT AND OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`dosy` - DOSY processing parameters

SEE ALSO

dosy2d, dosy3d

edgp

NAME

edgp - edit gradient programs

SYNTAX

edgp [<name>]

DESCRIPTION

The command ***edgp*** allows you to list, create or edit gradient programs. If you enter ***edgp*** without arguments, a list of all gradient programs is displayed. This list includes both the Bruker and the user defined gradient programs. When you click a gradient program it will be opened with an editor. Alternatively, you can enter a name in the field "Type New Name" to create a new gradient program. The ***Print*** button allows you to print the list of gradient programs.

If you enter the command with an argument, e.g. ***edgp <name>***, the gradient program <name> is opened, if it exists, or otherwise it is created. The argument may contain wildcards, e.g. ***edgp a**** displays a list of all gradient programs which start with *a*.

Bruker gradient programs must be installed with ***expinstall*** before they can be opened with ***edgp***.

edgp uses the editor which is defined in the XWIN-NMR User Interface. You can change it by typing ***setres*** and specify it in the field ***Editor***.

INPUT AND OUTPUT FILES

<xwhome>/exp/stan/nmr/lists/gp/*

gradient programs

SEE ALSO

edcgp, expinstall

edlist

NAME

edlist - edit various lists

SYNTAX

edlist [<type> [<name>]]

DESCRIPTION

The command ***edlist*** allows you to list, create or edit various lists. Most of these lists are used by the acquisition. For example, a *vd* list is read by the pulse program statement `vd`. The command ***edlist*** can be used in the following ways:

edlist - display all lists types
edist <type> - display all entries of the list type <type>
edlist <type> <name> - create or edit the list <name> of type <type>

The second argument may contain wildcards, e.g. ***edlist vd a**** displays all variable delay lists which start with *a*.

Lists which are used by pulse programs are displayed in table 9.1. The command

list type	contains
vd	variable delay lists
vp	variable pulse lists
f1	frequency lists (Avance)
f1, f2, f3	frequency lists (A*X)
vt	variable temperature lists
vc	variable counter lists
ds	variable dataset lists
masr	MASR rotation values

Table 9.1 Types of parameter lists

edlist also shows other lists like pulse programs, CPD programs, gradient

programs and macros. However, these type of lists are normally opened with the dedicated commands ***edpul***, ***edcpd***, ***edgp*** and ***edmac***, respectively.

SEE ALSO

edmisc, edpul, edcpd, edgp, edmac

edmac

NAME

edmac - edit a macro

SYNTAX

edmac [<name>]

DESCRIPTION

The command ***edmac*** allows you to create or edit a macro. A macro contains a sequence of XWIN-NMR commands. Once created, you can enter the macro name on the command line to execute this sequence. A simple macro for processing and plotting the current dataset is:

```
em
ft
apk
sref
plot
```

All entries in a macro file must be written in lowercase letters.

If you enter ***edmac*** without arguments, a list of existing macros is displayed. When you click a macro it will be opened with an editor. Alternatively, you can enter a name in the field "Type New Name" to create a new macro. The ***Print*** button allows you to print the list of existing macros.

If you enter the command with an argument, e.g. ***edmac <name>***, the macro <name> is opened, if it exists, or otherwise it will be created. The argument may contain wildcards, e.g. ***edmac a**** displays a list of all macros which start with *a*.

edmac uses the editor which is defined in the XWIN-NMR User Interface. If you want to use a different editor, type ***setres*** and modify the entry ***Editor***.

INPUT AND OUTPUT FILES

<xwhome>/exp/stan/nmr/lists/mac/*

XWIN-NMR macros

SEE ALSO

xmac, delmac

edmisc

NAME

edmisc - edit miscellaneous lists

SYNTAX

edmisc [<type>]

DESCRIPTION

The command ***edmisc*** allows you to list, create or edit miscellaneous lists in the current dataset. It takes one argument and can be used as follows:

edmisc - displays all lists types, you can select one for editing
edmisc <type> - edits or creates the list <type> in the current dataset

The lists which can be edited with ***edmisc*** are shown in table 9.2.

list type	contains
intrng	integral regions, created by interactive integration or automatic baseline correction (***abs***). Used for plotting (***view***, ***plot***) and integral printout (***li***, ***lipp***).
base_info	*polynomial*, *sine* or *exponential* baseline function, created from the baseline menu (***basl***). Used by the baseline correction command ***bcm***..
baslpnts	baseline points created by ***def-pts*** from the baseline menu (***basl***). Used by the spline baseline correction command ***sab***.
peaklist	peak information, created by the command ***ppp***. Used by the mixed deconvolution command ***mdcon.***
reg	plot regions, created from the integration menu (command ***integ***). Used by ***view***, ***plot***, ***pp***, ***lipp*** when PSCAL=ireg or pireg and by ***view*** and ***plot*** when LIMITS=region.

Table 9.2 Miscellaneous list types

Miscellaneous lists reside in the processed data directory of the current dataset.

They can be stored for general usage with the command ***wmisc***. After that, they can be read on other datasets with the command ***rmisc***. A typical sequence would be:

1. Interactive integration - to create a list of integral regions
2. ***edmisc intrng*** - to make manual changes to the integral regions
3. ***wmisc intrng myinteg*** - to store the lists for general usage
4. Read dataset on which you want to apply the same integral regions
5. ***rmisc intrng myinteg*** - read the integral regions you store before

INPUT AND OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`intrng` - integral regions
`base_info` - baseline correction coefficients
`baslpnts` - |baseline points for spline baseline correction with ***sab***
`peaklist` - peak list
`reg` - plot region

SEE ALSO

rmisc, wmisc, delmisc

edp

NAME

edp - edit processing parameters

DESCRIPTION

The command ***edp*** opens a dialog box in which you can set all processing parameters. For most processing steps, however, only a few parameters need to be set, e.g.:

- For any processing step: SI - the size of the processed data
- For FID baseline correction: BC_mod - the baseline correction mode
- For window multiplication: WDW - the window multiplication mode
- For exponential window multiplication: LB - the Lorentzian broadening factor

Note that the main processing step, the Fourier transform does not require a special processing parameter to be set. There is a parameter FT_mod, but it is only used by the processing commands ***trf**** (1D) and ***xtrf**** (2D).

The ***edp*** dialog box offers the following buttons/fields:

- ***Save***
  save all parameters
- ***1-Col***
  change to one column display mode
- ***Parameter***
  allows you to search for a parameter. Just enter the parameters name (or a part of it) and hit the **Enter** key. The ***edp*** window will scroll to parameters position. Note that the parameter might be on the current page.
- ***Next***
  select the next parameter which starts with the string in the ***Parameter*** field
- ***Cancel***
  leave ***edp*** without saving any changes

Note that you can also set 1D parameters by entering there names on the command line, e.g.:

si
You will be prompted to enter the size.
si 4k
The size will be set to 4k

For 2D and 3D datasets, ***edp*** displays a separate column of parameters for each dimension. The ***1-Col*** button is not available. Not all parameters are available for each dimension. For example, PKNL is only available for the first (acquisition) dimension. 2D parameters can also be set from the command line, e.g.:

si 4k
The F2 size will be set to 4k

2 si 4k
The F2 size will be set to 4k

1 si 4k
The F1 size will be set to 4k

The NMR Superuser can change the ***edp*** dialog box, for example remove parameter which are not used. This must be done on operating system level by editing the file `proc.e` (see below).

INPUT AND OUTPUT PARAMETERS

All processing parameters.

INPUT FILES

<xwhome>/exp/stan/nmr/form/

`proc.e` - format file for ***edp***

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`proc` - processing parameters
`proc2` - processing parameters for the second dimension (2D or 3D)
`proc3` - processing parameters for the third dimension (3D)

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>

`proc` - processing parameters
`proc2` - processing parameters for the second dimension (2D or 3D)

`proc3` - processing parameters for the third dimension (3D)

SEE ALSO

edp3, edp2, edp1

edp3, edp2, edp1

NAME

edp3 - edit the F3 processing parameters of a 3D dataset
edp2 - edit the F2 processing parameters of a 2D or 3D dataset
edp1 - edit the F1 processing parameters of a 2D or 3D dataset

DESCRIPTION

The commands ***edp3***, ***edp2*** and ***edp1*** open a dialog box in which you can set processing parameters for the F3, F2 and F1 dimension respectively. ***edp3*** only works on 3D data, ***edp2*** and ***edp1*** work on 2D and 3D data. They work like ***edp***, except that they only show the processing parameters for one dimension of a dataset. Furthermore, they only show a restricted set of processing parameters whereas ***edp*** shows them all.

INPUT AND OUTPUT PARAMETERS

All processing parameters.

INPUT AND OUTPUT FILES

<xwhome>/exp/stan/nmr/form/

`edp3.e` - format file for ***edp3***
`edp2.e` - format file for ***edp2***
`edp1.e` - format file for ***edp1***

<du>/data/<user>/<name2D>/nmr/<expno>/pdata/<procno>

`proc` - F2 processing parameters (input for ***edp2***)
`proc2` - F1 processing parameters (input for ***edp1***)

<du>/data/<user>/<name3D>/nmr/<expno>/pdata/<procno>

`proc` - F3 processing parameters (input for ***edp3***)
`proc2` - F2 processing parameters (input for ***edp2***)
`proc3` - F1 processing parameters (input for ***edp1***)

SEE ALSO

edp

edpul

NAME

edpul - edit a pulse program

SYNTAX

edpul [<name>]

DESCRIPTION

The command ***edpul*** allows you to list, edit or created pulse programs.

edpul without arguments opens a dialog box with two columns of pulse programs. The right column shows the Bruker pulse programs, the left column the user defined pulse programs. When you click a Bruker pulse program, it is opened in view mode which means it cannot be changed. When you click a user defined pulse program, it is opened with an editor and can be modified. Alternatively, you can enter a name in the field "Type New Name" to create a new pulse program.

The ***edpul*** dialog box also contains a button ***Edit*** which allows you to switch to Edit-mode and modify Bruker pulse programs. Be careful: Bruker pulse programs are overwritten when you install a new version of XWIN-NMR. Therefore, we recommend to store modified Bruker pulse programs under a new name.

edpul <name> opens the pulse program <name>. If <name> does not exist, a n empty file is created and opened with an editor.

If you specify an argument, then it may contain wildcards; for example:

edpul cos* lists all pulse programs beginning with *cos*
edpul [m-z]* lists all pulse programs beginning with *m*,*n*,...,*z*

Bruker pulse programs must be installed with ***expinstall*** before they can be opened with ***edpul***.

edpul uses the editor which is defined in the XWIN-NMR User Interface. If you want to use a different editor, type ***setres*** and modify the entry ***Editor***.

INPUT FILES

<xwhome>/exp/stan/nmr/lists/pp/*

pulse programs

SEE ALSO

edcpul, expinstall, edlist

expinstall

NAME

expinstall - install pulse programs, AU programs, parameter sets etc.

DESCRIPTION

The command ***expinstall*** installs pulse programs, AU programs, parameter sets and various other resources for spectrometer usage. It must be performed once after the installation of XWIN-NMR and after ***cf*** has been done. cf and expinstall are typically performed as a part of the ***config*** configuration suite.

expinstall first prompts you for the NMR Superuser password. After it has been entered correctly, you see a list of spectrometer types. The spectrometer type you have defined with ***cf*** is highlighted. You can simply click ***Proceed***, unless you have a reason to choose a different type of spectrometer. ***expinstall*** will then offer you a list of tasks which can be selected or deselected. For routine spectroscopy, you can accept the default selection and click ***Proceed*** to continue. If the task *Convert Standard Parameter Sets* was selected, you will be prompted for some information about the spectrometer. Once this has been entered, ***expinstall*** will start to execute the selected tasks. The full list of tasks is:

- Install pulse programs
  These are used for all experiments.
- Install Bruker library AU programs
  Performing this task makes Bruker AU programs available for editing (***edau***) and compilation (***xau*** or ***cplbruk***).
- Recompile User AU programs
  AU programs must be (re)compiled after the installation of a new XWIN-NMR version because the installation has removed the AU binaries. ***expinstall*** only compiles User AU programs, not Bruker AU programs. The latter can be compiled with ***compileall*** or ***cplbruk all***.
- Install CPD programs
  These are used for composite pulse decoupling experiments.
- Install gradient files
  These are used for gradient experiments.

- Install Library shape files
  These are used for selective excitation experiments.
- Convert standard parameter sets
  Bruker standard parameter sets, as they are delivered with the NMR Suite, were prepared at various field strengths. ***expinstall*** converts them to your spectrometer frequency. This includes the parameters BFx, Ox, SFOx and SW as well as the offsets of the shaped pulses (parameter SPOFFS). Note that for 2D (3D), the SW in F1 (F2 and F1) is kept and the increment IN0 (IN0 and IN10) is adjusted.
- Install standard scaling region files
  These contain the regions in which the reference peak for vertical scaling is searched by commands like ***plot***, ***li***, ***lipp***, ***pp****.
- Enable Define Statements in Pulse programs
  Define statements are pulse program statements like:

  ```
  ;; d11=30m
  ```

  at the beginning of a pulse program. In this form they are not active because lines starting with a '`;`' character are treated as comment. ***expinstall*** removes all occurrences '`;;`', thereby enabling the defined statements. Normal comment lines are not affected because they contain a single '`;`' only.

The NMR Suite is delivered with a set of pulse programs, CPD programs, gradient files, shape files and scaling region files for each spectrometer type (Avance, AMX, ARX etc.). ***expinstall*** installs the set which is needed on your spectrometer type.

If the task *Convert Standard Parameter Sets* is selected, ***expinstall*** will prompt you for the following information:

- Select type of digitizer:
  Click the digitizer that you want to store in the acquisition parameter DIGTYP in all parameter sets.
- Select acquisition mode:
  Click the acquisition mode that you want to store in the acquisition parameter AQ_mod in all parameter sets
- Enter default pre-scan-delay DE:
  Enter the value that you want to store in the acquisition parameter DE in all parameter sets. Normally, you can accept the default value.

- Select printer:
  Click the printer that you want to store in the output (***edo***) parameter CURPRIN in all parameter sets. This printer will be used by commands like ***lpa*** and ***lpp***.
- Select plotter:
  Click the printer that you want to store in the output (***edo***) parameter CURPLOT in all parameter sets. This printer will be used by commands like ***view*** and ***plot***.
- Enter paper format:
  Enter A4, A3, A or B. The entered value will effect various plot (***edg***) parameters like CX, CY and DHEI in all parameter sets. If you enter any value other than A4, A3, A or B, the file:

  <xwhome>/exp/stan/nmr/lists/plotconvpar

  is interpreted for the paper format [1]. This file is delivered with XWIN-NMR and you can change it for your purpose from the Windows Explorer or from a UNIX shell.

INPUT PARAMETERS

If the task *Convert standard parameter sets* is selected, ***expinstall*** uses the following input parameters:

set by the user with ***edsp*** :

DEFRSEL - preferred preamplifier (default routing)
DEF19F - preferred output for 19F (default routing)

from the parameter sets as delivered with XWIN-NMR:

BF1 - BF4 - basic frequencies for channel f1 to f4
SFO1- SFO4 - irradiation (carrier) frequencies for channels f1 to f4
IN0 - increment for delay D0 (2D and 3D parameter sets only)
IN10 - increment for delay D10 (3D parameter sets only)
SW - spectral width in ppm
SPOFFS[0-7] - shaped pulse frequency offset

1. Note that the file plotconvpar.A3 is used for A3 and B and plotconvpar.A4 for A4 and A.

OUTPUT PARAMETERS

If the task *Convert standard parameter sets* is selected, ***expinstall*** stores the following parameters in the parameter sets:

BF1 - BF4 - basic frequencies for channel f1 to f4
SFO1- SFO4 - irradiation (carrier) frequencies for channels f1 to f4
SF - spectral reference frequency
IN0 - increment for delay D0 (2D and 3D parameter sets only)
IN10 - increment for delay D10 (3D parameter sets only)
SW - spectral width in ppm
SPOFFS[0-7] - shaped pulse frequency offset
DIGTYP - digitizer type
DR - digital resolution
DIGMOD - digitizer mode
DECIM - decimation factor of the digital filter
DE - prescan delay
FCUCHAN - routing between logical frequency channels and FCU's
RSEL - routing between FCU's and amplifiers
SWIBOX - routing between Switchbox inputs and Switchbox outputs
PRECHAN - routing between Switchbox outputs and Preamplifier modules
HPMOD - routing between high power amplifiers and Preamplifier modules

INPUT FILES

<xwhome>/conf/instr/<instrum>/specpar - routing parameters

<xwhome>/prog/au/src.exam/* - Bruker AU programs (source files)

<xwhome>/exp/stan/nmr/au/src/* - AU programs (source files)

<xwhome>/exp/stan/nmr/par.avance/* - Bruker parameter sets for Avance

<xwhome>/exp/stan/nmr/par.300/* - Bruker parameter sets for A*X

<xwhome>/exp/stan/nmr/pp.dexam/* - pulse programs for Avance

<xwhome>/exp/stan/nmr/pp.exam/* - pulse programs for AMX

<xwhome>/exp/stan/nmr/cpd.dexam/* - CPD programs for Avance

<xwhome>/exp/stan/nmr/cpd.exam/* - CPD programs for AMX

<xwhome>/exp/stan/nmr/gp.dexam/* - gradient programs for Avance

<xwhome>/exp/stan/nmr/gp.exam/* - gradient programs for AMX

<xwhome>/exp/stan/nmr/wave.dexam/* - shape files for Avance

<xwhome>/exp/stan/nmr/wave.exam/* - shape files for AMX

<xwhome>/exp/stan/nmr/scl.exam/* - scaling region files for Avance/AMX

Depending on the spectrometer type and/or application, expinstall uses various other input folders/files using the extensions:

`.rexam` - high resolution on ARX
`.solids` - solid state on AMX/ASX
`.imag` - micro imaging on AMX
`.tomo` - tomography
`.dsolids` - solid state on Avance
`.dimag` - micro imaging on Avance

OUTPUT FILES

<xwhome>/exp/stan/nmr/au/src/* - Bruker AU programs (source files)

<xwhome>/prog/au/bin/* - AU programs (binary executables)

<xwhome>/exp/stan/nmr/par/* - parameter sets for your spectrometer

<xwhome>/exp/stan/nmr/pp/* - pulse programs for your spectrometer

<xwhome>/exp/stan/nmr/cpd/* - CPD programs for your spectrometer

<xwhome>/exp/stan/nmr/gp/* - CPD programs for your spectrometer

<xwhome>/exp/stan/nmr/wave/* - shape files for your spectrometer

<xwhome>/exp/stan/nmr/scl/* - scaling region files for your spectrometer

SEE ALSO

cf, config, cplbruk, cpluser, compileall, rpar, wpar

renau, renpar

NAME

renau - rename AU programs
renpar - rename parameter sets

DESCRIPTION

The command ***renau*** opens a dialog box with all AU programs, both Bruker and user defined. You can rename one or more AU programs, simply by replacing their names. When you enter a new name followed by Enter, the AU program is immediately renamed. This counts for both the source and the binary file. The dialog box can be closed by clicking ***Cancel***.

renpar works like ***renau***, except that it allows you to rename parameter sets. Parameter sets are directories, so ***renpar*** changes directory names; the names of the parameter files in those directories (*acqu*, *proc*, *meta*, *outd*) are not affected.

INPUT AND OUTPUT FILES

For ***renau***:

<xwhome>/exp/stan/nmr/au/src/*
<xwhome>/prog/au/bin/*

For ***renpar***:

<xwhome>/exp/stan/nmr/par/*

SEE ALSO

ren, reno, rengp, renlist, renlut, renmac, renpul

rengp, renlut, renmac, renpul

NAME

rengp - rename gradient programs
renlut - rename 2D lookup tables
renmac - rename macros
renpul - rename pulse programs

DESCRIPTION

The command ***renpul*** opens a dialog box with all pulse programs, both Bruker and user defined. You can rename one or more pulse programs, simply by replacing their names. When you enter a new name and hit the **Enter** key, the pulse program is immediately renamed. The dialog box can be closed by clicking ***Cancel***.

In the same way, ***renmac***, ***rengp*** and ***renlut*** allow you to rename macros, gradient programs and 2D lookup tables, respectively.

INPUT AND OUTPUT FILES

<xwhome>/exp/stan/nmr/lists/pp/*

pulse programs

<xwhome>/exp/stan/nmr/lists/mac/*

XWIN-NMR macros

<xwhome>/exp/stan/nmr/lists/gp/*

gradient programs

SEE ALSO

ren, reno, renau, renpar

rmisc

NAME

rmisc - read miscellaneous lists

DESCRIPTION

The command ***rmisc*** allows you to read miscellaneous lists (integral, baseline and peak lists) which have previously been stored with ***wmisc***. The selected list is copied to the current dataset.

rmisc takes two arguments and can be used as follows:

rmisc
displays all lists types. If you select a type, all entries of that type will appear. If you select an entry, the corresponding list will be read.

rmisc <type>
shows all entries of the type <type>. If you select an entry, the corresponding list will be read.

rmisc <type> <name>
copies the list <name> of the type <type> will be read.

where type can be *intrng*, *base_info*, *baslpnts* or *peaklist*. These list types are described in detail for the command ***edmisc***.

INPUT FILES

<xwhome>/exp/stan/nmr/lists/intrng/*
<xwhome>/exp/stan/nmr/lists/baslpnts/*
<xwhome>/exp/stan/nmr/lists/base_info/*
<xwhome>/exp/stan/nmr/lists/peaklist/*

OUTPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`intrng` - integral regions
`base_info` - baseline correction coefficients
`baslpnts` - |baseline points for spline baseline correction with ***sab***
`peaklist` - peak list
`reg` - plot region

SEE ALSO

edmisc, wmisc, delmisc

renlist

NAME

renlist - rename various lists

DESCRIPTION

The command ***renlist*** displays a list with various lists types, most of which are used in acquisition. When you click a list type, a dialog box appears with the available entries of that type. You can rename one or more entries, simply by replacing their names. When you enter a new name followed by Enter, the list is immediately renamed. The dialog box can be closed by clicking ***Cancel***.

INPUT AND OUTPUT FILES

<xwhome>/exp/stan/nmr/lists/*

SEE ALSO

ren, reno, rengp, renlut, renmac, renpul

rpar

NAME

rpar - read a parameter set

SYNTAX

rpar [<name> [<type>]]

DESCRIPTION

The command ***rpar*** reads a parameter set to the current dataset. It takes two arguments and can be used as follows:

- ***rpar***
  shows a list of all parameter sets. When you click on a parameter set, you will get a list of parameter types. You can select one or more parameter types and then click ***Copy*** to copy them to the current dataset. Alternatively, you can click ***Copy All*** to copy all parameter types.
- ***rpar <name>***
  shows a list of parameter types for the parameter set <name>. You can select one or more parameter types and then click ***Copy*** to copy them to the current dataset. Alternatively, you can click ***Copy All*** to copy all parameter types.
- ***rpar <name> <type>***
  copies the parameter type <type> of the parameter set <name>

The following parameter types can be specified as the second argument:

- *acqu* - acquisition parameters
- *proc* - processing parameters
- *plot* - graphics and plot parameters
- *outd* - output device parameters
- *all* - all parameters including acqu, proc, plot and outd

The first argument may contain wildcards, e.g.:

rpar C* shows all parameter sets beginning with the letter C
rpar [H-Z]* shows all parameter sets beginning with a letter between H and Z.

After reading a parameter set with ***rpar***, you can modify parameters of the various types with the commands:

- ***eda*** - acqu parameters
- ***edp*** - processing parameters
- ***edg and edgx*** - plot parameters
- ***edo*** - output device parameters

Note that Bruker parameter sets contain all parameter types, but user defined parameter sets contain only those parameter types that were stored when the parameter set was created (see ***wpar***). Usually, however, user defined parameter sets are also stored with all parameter types.

Bruker parameter sets are delivered with XWIN-NMR and installed with the command ***expinstall***.

User defined parameter sets are created with ***wpar***, which stores the parameters of the current dataset under a new or existing parameter set name.

rpar allows you to read parameters sets of various dimensionalities, 1D, 2D, etc. If the dimensionality of the current dataset and the parameter set you want to read are the same, e.g. both 1D, the current parameter files are overwritten. If the current dataset contains data (raw and/or processed data), these are kept. Furthermore, the status parameters are kept so you still have a consistent dataset. However, as soon as you process the data, the new processing parameters are used, the processed data files are overwritten and the processing status parameters are updated. When you start an acquisition, the new acquisition parameters are used, the raw data are overwritten and the acquisition status parameters are updated. If the current dataset is 1D, contains data (raw and/or processed) and you read a 2D parameter set, ***rpar*** will warn you that the current data will be deleted and ask you whether or not you want to continue. However, this warning will not appear if you enter the command with two arguments, i.e.:

```
rpar <name> <type>
```

In that case, data files of a different dimensionality are simply deleted. The reason is that is that ***rpar*** with two arguments is used in automation.

INPUT FILES

<xwhome>/exp/stan/nmr/par/<1D parameter set>/

`acqu` - acquisition parameters
`proc` - processing parameters
`meta` - plot parameters for ***plot*** and ***view***
`meta.ext` - extended plot parameters for ***plotx***
`outd` - output device parameters

<xwhome>/exp/stan/nmr/par/<2D parameter set>/

`acqu` - F2 acquisition parameters
`acqu2`- F1 acquisition parameters
`proc` - F2 processing parameters
`proc2` - F1 processing parameters
`meta` - plot parameters for ***plot*** and ***view***
`meta.ext` - extended plot parameters for ***plotx***
`outd` - output device parameters

3D parameter sets also contain the files `acqu3` and `proc3` for the third dimension but do not contain the file `meta.ext`.

OUTPUT FILES

<du>/data/<user>/nmr/<1D data name>/<expno>/

`acqu` - acquisition parameters

<du>/data/<user>/nmr/<1D data name>/<expno>/pdata/<procno>/

`proc` - processing parameters
`meta` - plot parameters for ***plot*** and ***view***
`meta.ext` - extended plot parameters for ***plotx***
`outd` - output device parameters

<du>/data/<user>/nmr/<2D data name>/<expno>/

`acqu` - F2 acquisition parameters
`acqu2` - F1 acquisition parameters

<du>/data/<user>/nmr/<2D data name>/<expno>/pdata/<procno>/

`proc` - F2 processing parameters
`proc2` - F1 processing parameters
`meta` - plot parameters for ***plot*** and ***view***
`meta.ext` - extended plot parameters for ***plotx***
`outd` - output device parameters

USAGE IN AU PROGRAMS

RPAR(name, type)

SEE ALSO

wpar, dirpar, delpar, renpar, expinstall

wmisc

NAME

wmisc - write miscellaneous lists

DESCRIPTION

The command ***wmisc*** allows you to store miscellaneous lists (integral, baseline and peak lists) which can later be read on a different dataset with ***rmisc***. ***wmisc*** takes two arguments and can be used as follows:

wmisc
displays miscellaneous lists types. If you select a type, all existing entries of that type will appear plus a field ***Type New Name***. If you click an existing list, it will be overwritten, if you enter a new name, a new list will be created.

wmisc <type>
displays existing entries of the type <type> plus a field ***Type New Name***. If you click an existing list, it will be overwritten, if you enter a new name, a new list will be created.

wmisc <type> <name>
writes the list <type> from the current dataset under name <name>. If this already exists, it will be overwritten without warning.

where <*type*> can be *intrng*, *base_info*, *baslpnts* or *peaklist*. These list types are described in detail for the command ***edmisc***.

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`intrng` - integral regions
`base_info` - baseline correction coefficients
`baslpnts` - |baseline points for spline baseline correction with ***sab***
`peaklist` - peak list
`reg` - plot region

OUTPUT FILES

<xwhome>/exp/stan/nmr/lists/intrng/*
<xwhome>/exp/stan/nmr/lists/baslpnts/*

<xwhome>/exp/stan/nmr/lists/base_info/*
<xwhome>/exp/stan/nmr/lists/peaklist/*

SEE ALSO

edmisc, rmisc, delmisc

wpar

NAME

wpar - write a parameter set

SYNTAX

wpar [<name> [< type>]]

DESCRIPTION

The command ***wpar*** stores the parameters of the current dataset in a parameter set. This parameter set is then available for general usage and can be read to any dataset with ***rpar***.

wpar allows you to overwrite an existing parameter set or enter a new name. It takes two arguments an can be used in one of the following ways:

wpar
shows all existing parameter sets plus a field ***Type New Name***. If you click a parameter set, you can select the parameter types you want to store and then click **Copy** to store them. Alternatively, you can click ***Copy All*** to store all types.

wpar <name>
shows a list of parameter types. If the parameter set <name> already exists, the existing types are highlighted. You can select the parameter types you want to store and then click **Copy** to store them. Alternatively, you can click ***Copy All*** to store all types.

wpar <name> <type>
stores parameters of type <type> to parameter set <name>. If <type> already exists in <name>, it is overwritten

The following parameter types are available:

acqu - acquisition parameters (set with ***eda***)
proc - processing parameters (set with ***edp***)
plot - plot parameters (set with ***edg*** and ***edgx***)
outd - output device parameters (set with ***edo***)
all - all parameters including *acqu*, *proc*, *plot* and *outd*

The first argument of ***wpar*** may contain wildcards, for example:

wpar C* lists all parameter sets which begin with the letter C
wpar [H-Z]* lists all parameter sets which begin with a letter between H and Z.

wpar is often used in the following way:

1. Define a new dataset with ***edc*** or ***new***.
2. Enter ***rpar*** to read a Bruker parameter set which defines the experiment you want to do.
3. Modify the acquisition parameters (with ***eda***) to your preference and run the acquisition.
4. Modify processing parameters (with ***edp***) to your preference and process the data.
5. Modify the plot parameters (with ***edg***) to your preference, set the output device parameters (with ***edo***) and plot the dataset [1].
6. Store the parameters with ***wpar*** for general usage.

USAGE IN AU PROGRAMS

WPAR(name, type)

SEE ALSO

rpar, dirpar, delpar, renpar, expinstall

1. Alternatively, you can use XWIN-PLOT whose layout is not part of the parameter set.

xau, xaua, xaup

NAME

xau - execute an AU program
xaua - execute the AU program specified with AUNM
xaup - execute the AU program specified with AUNMP

SYNTAX

xau [<name>]
xaua
xaup

DESCRIPTION

The command ***xau*** allows you to execute an AU program. Normally, however, AU programs are executed simply by entering their names. The command ***xau*** is only needed in two cases:

- the AU program has not been compiled yet
- an XWIN-NMR command with the same name exists

Usually, AU programs have already been compiled with ***edau***, ***compileall***, ***cplbruk*** or ***cpluser***. Furthermore, it is not recommended to give an AU program the same name as an XWIN-NMR command. Before you start writing a new AU program, just type in the name you want to give it to find out if this name is already in use.

AU programs can be executed in three different ways:

xau - a list of all AU programs appears, you can click one to execute it

xau <name> - executes the AU program <name>

<name> - executes the AU program <name>

Furthermore, you can use the commands ***xaua*** and ***xaup*** to execute AU programs. These commands take no argument but execute the AU program which is specified with the parameters AUNM and AUNMP, respectively. In all Bruker parameter sets, these parameters are set to relevant Bruker AU programs, e.g. in the parameter set PROTON, AUNM = au_zg and AUNMP = proc_1d. When parameter sets are used in automation (ICON-NMR), the AU programs specified by

AUNM and AUNMP do acquisition and processing, respectively.

AU programs run in background and several of them can run simultaneously. You can use the command ***follow*** to see which AU programs and which parts within each AU program are currently running. The command ***kill*** can be used to stop a running (or hanging) AU program.

For details on writing, compiling, and executing AU programs please refer to the XWIN-NMR AU reference manual.

INPUT PARAMETERS

set by the user with ***eda*** or by typing ***aunm*** :

AUNM - acquisition AU program name for ***xaua***

set by the user with ***edp*** or by typing ***aunmp*** :

AUNMP - processing AU program name for ***xaup***

INPUT FILES

<du>/data/<user>/<name>/nmr/<expno>/

`acqu` - acquisition parameters (input file for ***xaua***)

<du>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/

`proc` - processing parameters (input file for ***xaup***)

USAGE IN AU PROGRAMS

XAU(name)

XAUA

XAUP

XAUPW

Note that XAUPW waits until the AU program it executes has finished before the next statement is executed whereas XAUP doesn't. XAUA works like XAUPW is this respect.

SEE ALSO

edau, delau, renau, expinstall, compileall, cplbruk, cpluser

xmac

NAME

xmac - execute a macro

SYNTAX

xmac [<name>]

DESCRIPTION

The command ***xmac*** allows you to execute an XWIN-NMR macro. These are text files which contain a sequence of XWIN-NMR commands as they would be entered on the command line. They can be started in three different ways:

xmac - a list of all macros appears, you can click one to execute it

xmac <name> - execute the macro <name>

<name> - execute the macro <name>

The third possibility, just typing in the macro's name, only works if this name is unique, i.e. it is different from all XWIN-NMR commands and AU programs.

As opposed to AU programs, macros are not delivered by Bruker. However, they are very simply to create as is described for the command ***edmac***.

SEE ALSO

edmac, delmac, xau

Chapter 10

Conversion commands

This chapter describes all XWIN-NMR conversion commands. These are commands which convert one data format to another. Described are the conversion of Bruker Aspect 2000/3000 to XWIN-NMR, of Varian and Jeol to XWIN-NMR, of Avance to AMX, of XWIN-NMR to JCAMP-DX and of JCAMP-DX to XWIN-NMR.

conv

NAME

conv - convert Aspect 2000/3000 data to XWIN-NMR format

SYNTAX

conv [<station> [<filename> | ? | *]]

DESCRIPTION

The command ***conv*** converts Aspect 2000/3000 data to the XWIN-NMR format. It takes two arguments, *station* and *filename*, which are both parts of the pathname of the input data:

Before you can use ***conv***, you have to set up its configuration environment once. For each Aspect 2000/3000 station from which you want to convert data, you must create:

- a configuration file
- an input data directory

Setup under Windows

Creating a configuration file

Bruker has configuration files for all spectrometers controlled by Aspect 2000/3000. Just sent an email with your spectrometer specification to:

nmr-software-support@bruker.de

and you'll receive the proper configuration file.

Open the Windows Explorer and go to the folder:

`/<xwhome>/conf/instr/`

where <xwhome> is the directory where XWIN-NMR is installed. There, you must create a new folder with the name of the source spectrometer (see below). Then copy the configuration file to this folder. For example, a DISNMR configuration file would have to be stored as:

`<xwhome>/conf/instr/<station>/disnmr.conf`

For DISMSL data, you can create the `dismsl.conf` file with the XWIN-

NMR command ***convsys***.

Creating an input directory file

conv searches for input data in a directory like:

```
<dir>/bruknet/<station>/<user>/
```

This directory must be created by the user. It contains the variables:

`<dir>`
this must be the desired disk unit (the ***edc*** parameter DU) of the output data (the converted XWIN-NMR dataset)

`<station>`
this can be freely chosen, e.g. the name of the source spectrometer

`<user>`
this must the desired user (the ***edc*** parameter USER) of the output data

conv uses the ***edc*** parameters DU and USER of the current (foreground) dataset to determine the parts `<dir>` and `<user>`, respectively, of the input directory. The part `<station>` is specified as an argument. For example, if you enter ***conv msl1*** on the XWIN-NMR dataset:

```
/x/data/joe/nmr/exam1d/1/pdata/1
```

then ***conv*** searches for input data in the directory:

```
/x/bruknet/msl1/joe
```

Setup under UNIX

Creating a configuration file

For DISNMR data, the file DISNMR.CONF must be transferred from the Aspect 2000/3000 computer to:

```
<dir>/bruknet/<station>/<user>/DISNMR.CONF+
```

Then it must be converted with the standalone program:

```
<xwhome>/prog/bin/config
```

You can also obtain the `disnmr.conf` file from Bruker by sending an email with your spectrometer specification to:

nmr-software-support@bruker.de

For DISMSL data, you can create the `dismsl.conf` file with the XWIN-NMR command ***convsys***. You do not need to transfer this file from the Aspect 2000/3000.

Creating a input directory file

The ***conv*** searches for input data in a directory like:

```
<dir>/<station>/<user>/
```

Under UNIX, A2000/3000 data are usually transferred with Bruknet. This program automatically creates the file:

```
/usr/local/lib/destination
```

with the contents /u/bruknet. If you want to, you can edit this file and replace the contents with any other existing pathname. Bruknet interprets the file `destination` and stores the Aspect 2000/3000 in the corresponding directory thereby creating the necessary subdirectories `<station>` and `<user>`.

conv interprets the file `destination` and searches the corresponding directory for input data. For the rest it works as it does under Windows. For example, if you enter ***conv msl1*** on the XWIN-NMR dataset:

/**x**/data/**joe**/nmr/exam1d/1/pdata/1

and the `destination` file contains the path:

/**u**/bruknet

conv searches for input data in the directory:

/**u**/bruknet/msl1/**joe**

If, however, the destination file does not exist, ***conv*** searches for input data in the directory:

/**x**/bruknet/msl1/**joe**

Converting data

conv must be entered on an XWIN-NMR foreground dataset where:

- USER corresponds to the <user> part of the input directory

- DU corresponds to the <dir> part of the input directory. This, however, does not count if you work under UNIX and use a `destination` file (see above).

If the current dataset does not fulfil this requirement, you have to change datasets before you start data the conversion.

conv takes two arguments and can be entered as follows:

conv
you will be prompted for the station name and filename

conv <station>
you will be prompted for the filename which will then be searched for under the specified station

conv <station> <filename>
the specified filename will be searched for under the specified station and will be converted. If the filename contains a '+', do not specify this.

conv <station> ?
all filenames under the specified station name will be displayed. When you click on a dataset, it will be converted.

conv <station> *
takes the next available file under the specified station and converts it. If no files exists, ***conv*** waits and starts the conversion as soon as a dataset arrives.

The output data of ***conv***, the converted dataset, has the following XWIN-NMR data parameters (command ***edc***):

- DU is set to the disk partition of the foreground XWIN-NMR dataset.
- USER is set to the <user> part of the pathname of the foreground dataset.
- NAME is set to the filename of the Aspect 2000/3000 file without the extension.
- EXPNO is set to filename extension of the Aspect 2000/3000 file.
- PROCNO is set to 1.

Aspect A2000/3000 data can also be converted with the command ***btran***. It works like ***conv*** except that it allows you to choose the destination disk unit and user.

The AU program ***remproc*** converts A3000/2000 data in an infinite loop. If the input directory is empty, it waits for unconverted data which are converted as soon as they arrive. Before you can use this AU program, you must edit it with ***edau remproc*** and define the STATION.

INPUT FILES

<dir>/bruknet/<station>/<user>/* - A2000/3000 data
<xwhome>/conf/<station>/disnmr.conf - DISNMR configuration
<xwhome>/conf/<station>/dismsl.conf - DISMSL configuration
/usr/local/lib/destination - destination file (UNIX only)

OUPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - Avance type 1D raw data
`ser` - Avance type 2D or 3D raw data
`acqu` - acquisition parameters
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>

`1r`, `1i` - converted processed 1D data
`2rr`, `2ir`,`2ri`, `2ii` - converted processed 2D data
`proc` - processing parameters
`procs` - processing status parameters

For 2D data, the parameter files `acqu2`, `acqu2s`, `proc2` and `procs` will also be created.

USAGE IN AU PROGRAMS

CONV(instrname,filename)

SEE ALSO

convsys, vconv, jconv, convdta

convsys

NAME

convsys - create a configuration file for the conversion of DISMSL data

DESCRIPTION

The command ***convsys*** creates a configuration file for the conversion of DISMSL data from an Aspect 2000 or 3000. ***convsys*** first prompts for the NMR Superuser password and then for the MSL station name and its basic proton frequency.

Unlike the shell command <xwhome>/prog/bin/config, which converts an configuration file for DISNMR data, ***convsys*** needs no input file.

The output file `dismsl.conf` is interpreted by the conversion commands ***conv*** and ***btran***.

OUTPUT FILES

<xwhome>/conf/instr/<station>/

`dismsl.conf` - the DISMSL configuration data

SEE ALSO

conv, <xwhome>/prog/bin/config

convdta

NAME

convdta - convert Avance type raw data to AMX type raw data

DESCRIPTION

The command ***convdta*** converts Avance type raw data to AMX type raw data. It can handle 1D, 2D and 3D data. This is useful if you want to process data which have been acquired on an Avance spectrometer on an AMX or ARX spectrometer.

convdta takes up to six arguments and can be used as follows:

1. ***convdta***
   You will be prompted for an expno under which the FID must be stored
2. ***convdta <expno>***
   The FID will be stored under the specified expno.
3. ***convdta <expno> <name> y***
   The output will be stored under the specified *name* and *expno*. The last argument (*y*) causes ***convdta*** to overwrite existing data without a warning.
4. ***convdta <expno> <name> <user> <du> y n***
   The output will be stored under the specified *expno*, *name*, *user* and *diskunit*. The second last argument (*y*) causes ***convdta*** to overwrite existing data without a warning. The last argument (*n*) causes the display to remain on the current dataset rather than change to the output dataset.

You can use any other combination of arguments as long they are entered in the correct order. Note that the last argument in example 3 and the last two arguments in example 5, can only take the values *y* and *n*, respectively. The processed data number (procno) of the new dataset cannot be chosen, it is always set to 1.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - Avance type 1D raw data
`ser` - Avance type 2D or 3D raw data
`acqu` - acquisition parameters

`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>

`proc` - processing parameters
`procs` - processing status parameters

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<*expno*>/

`fid` - AMX type 1D raw data
`ser` - AMX type 2D or 3D raw data
`acqu` - acquisition parameters
`acqus` - acquisition status parameters
`audita.txt` - acquisition audit trail

<du>/data/<user>/nmr/<name>/<*expno*>/pdata/<procno>

`proc` - processing parameters
`procs` - processing status parameters

For 2D and 3D data, the `acqu2`, `acqu3`, `proc2` and `proc3` as well as the corresponding status parameter files are input and output files.

USAGE IN AU PROGRAMS

CONVDTA(expno)

SEE ALSO

btran, convsys, vconv, jconv, convdta

fromjdx

NAME

fromjdx - convert a JCAMP-DX datafile to XWIN-NMR format

SYNTAX

fromjdx [<pathname> [o]]

DESCRIPTION

The command ***fromdjx*** converts a JCAMP-DX data file to an XWIN-NMR dataset. JCAMP-DX is a standard ascii exchange format for spectroscopic data.

XWIN-NMR 3.1 supports the conversion of 1D data (raw or processed) and 2D data (raw or processed-real). XWIN-NMR 3.0 and older only support the conversion of 1D data.

fromjdx takes two arguments and can be used as follows:

fromjdx
prompts for the pathname of the JCAMP-DX input file and converts it.

fromjdx <pathname>
converts the JCAMP-DX file specified by the pathname and stores it under the lowest empty expno and procno

fromjdx <pathname> o
converts the JCAMP-DX file specified by the pathname and stores it under expno 1 and procno 1. Possibly existing data are overwritten (o).

fromjdx stores the output dataset in the directory:

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>

where

<du> - the disk unit of the current (foreground) dataset
<user> - the user of the current (foreground) dataset
<name> - the name of the JCAMP-DX file but without the extension .dx
<expno> - the lowest empty expno or, if the option ’o’ was used, expno 1
<procno> - the lowest empty procno or, if the option ’o’ was used, procno 1

XWIN-NMR always changes the display to the output dataset.

INPUT FILES

<pathname>/<mydata.dx> - XWIN-NMR data in JCAMP-DX format

OUTPUT FILES

For 1D and 2D data:

<xwhome>/prog/curdir/<user>/

`curdat` - current data parameters

<du>/data/<user>/nmr/<name>/<expno>/

`audita.txt` - acquisition audit trail (if input file contains raw data)

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>

`auditp.txt` - processing audit trail (if input file contains processed data)
`meta` - plot parameters for ***plot*** and ***view***
`meta.ext` - plot parameters for ***plotx***
`outd` - output device parameters
`title` - title file (see ***setti***)

For 1D data:

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - 1D raw data (if input file contains 1D raw data)
`acqu` - acquisition parameters
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>

`1r` - real processed 1D data (if input file contains 1D real processed data)
`1i` - imaginary processed 1D data (if input file contains 1D imaginary data)
`proc` - processing parameters
`procs` - processing status parameters

For 2D data:

<du>/data/<user>/nmr/<name>/<expno>/

`ser` - 2D raw data (input if Output Data = raw)
`acqu` - F2 acquisition parameters
`acqu2` - F1 acquisition parameters
`acqus` - F2 acquisition status parameters

`acqu2s` - F1 acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>

`2rr` - real processed 2D data (if input file contains 2D real processed data)
`proc` - F2 processing parameters
`proc2` - F1 processing parameters
`procs` - F2 processing status parameters
`proc2s` - F1 processing status parameters
level - 2D contour levels

USAGE IN AU PROGRAMS

FROMJDX(name, overwrite)
for example FROMJDX("/tmp/mydata.dx", "o")

SEE ALSO

tojdx

tojdx

NAME

tojdx - convert an XWIN-NMR 1D or 2D dataset to JCAMP-DX format

SYNTAX

tojdx [<pathname>[<datatype>[<compmode>[<title>[<origin>[<owner>]]]]]]

DESCRIPTION

The command ***todjx*** converts an XWIN-NMR dataset to JCAMP-DX format. JCAMP-DX is a standard ascii exchange format for spectroscopic data.

In XWIN-NMR 3.1 and newer, ***tojdx*** stores data in JCAMP-DX 6.0 format and supports the conversion of 1D data (raw or processed) and 2D data (raw or processed-real).

In XWIN-NMR 3.0 and older, ***tojdx*** stores data in JCAMP-DX 5.0 format and supports the conversion of 1D data only.

When ***tojdx*** is entered without argument, it will open a dialog box in which you can enter the required information which is:

- Output filename: enter the pathname of the output file. Default is *$home/name.dx*, where *name* is the name of the current XWIN-NMR dataset.
- Output data: click to choose from the following values:
  RAW DATA (=0) : raw data
  SPEC REAL (=1) : real processed data
  SPEC REAL+IMAG (=2) : real + imaginary processed data
  (default = *RAW DATA*)
- Compression mode: click to choose from the following values:
  FIX (=0) - table format
  PACKED (=1) - no spaces between the intensity values
  SQUEEZED (=2) - the sign of the intensity values is encoded in the first digit
  DIFF/DUP (=3) - the difference between successive values is encoded suppressing repetition of successive equal values
  (default = *DIFF/DUP*)

- TITLE: the title as it appears in the output file: enter a character string
- ORIGIN: the origin as it appears in the output file: enter a character string
- OWNER: the owner as it appears in the output file: enter a character string

The default TITLE is the plot title as defined with ***setti***. If no plot title is defined the data name is taken as default. The default ORIGIN and OWNER are taken from the acquisition status parameter files (`acqus`).

The above information can also be entered as arguments of ***tojdx***. If you enter an * character as argument, the default value will be used. Examples are:

```
tojdx C:\temp\mydata.dx 0 2 mytitle BRUKER guest
tojdx D:\nmr\mydata.dx 0 2 mytitle * *
tojdx * 1 * mytitle MYORIGIN joe
tojdx F:\users\guest\mydata.dx * * * * *
```

INPUT FILES

For 1D and 2D data:

<xwhome>/prog/curdir/<user>/

`curdat` - current data parameters

For 1D data:

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - 1D raw data (input if Output Data = raw)
`acqus` - acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>

`1r` - real processed 1D data (input if Output data = SP. REAL)
`1i` - imaginary processed 1D data (input if Output data = SP. REAL+IMAG)
`proc` - processing status parameters (input if Output data = RAW DATA)
`procs` - processing status parameters (input if Output data = SPEC...)

For 2D data:

<du>/data/<user>/nmr/<name>/<expno>/

`ser` - 2D raw data (input if Output Data = RAW DATA)
`acqus` - F2 acquisition status parameters
`acqu2s` - F1 acquisition status parameters

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>

`2rr` - real processed 2D data (input if Output data = SPEC REAL)
`proc` - F2 processing parameters (input if Output data = RAW DATA)
`proc2` - F1 processing parameters (input if Output data = RAW DATA)
`procs` - F2 processing status parameters (input if Output data = SP. REAL)
`proc2s` - F1 processing status parameters (input if Output data = SP. REAL)

OUTPUT FILES

<pathname>/<mydata.dx> - XWIN-NMR data in JCAMP-DX format

USAGE IN AU PROGRAMS

TOJDX(name, datatype, compmode, title, origin, owner)
for example TOJDX("/tmp/mydata.dx", 0, 2, "mytitle", "BRUKER", "joe")

SEE ALSO

fromjdx

jconv

NAME

jconv - convert Jeol type data to Bruker XWIN-NMR type data

SYNTAX

jconv [<inputfile>]

DESCRIPTION

The command ***jconv*** converts data from Jeol spectrometers to XWIN-NMR format.

jconv can handle Jeol EX, GX and ALPHA raw data and works on 1D, 2D and 3D data. Processed data cannot be converted. The conversion of FX FID data has been implemented. FX data must have a numerical extension (like in proton.1) and the name must be specified on the command line, e.g. jconv proton.1. No parameter file is needed for the conversion, the most relevant parameters are extracted from the header of the data file.

Data type	extension of data file	extension of parameter file
EX	.gxd	.gxp
GX	.gxd	.gxp
ALPHA	.nmf	.txt
DELTA	.bin	.hdr
FX	.num (an integer number)	no parameter file

Table 10.1

jconv

shows a list of all entries with the extension *.gxd*, *.nmf* and *.bin* in the directory defined by the environment variable JNMR. If JNMR is not set or points to a directory without any of the above entries, you are prompted for the Jeol data path. After entering a Jeol dataset, you are prompted for the output *name*, *expno*, *disk unit* and *user*, i.e. for the XWIN-NMR data path.

jconv jdata.ext

where *ext* can be *gxd*, *nmf* or *bin*. When the specified dataset is found, you are prompted for the output *name*, *expno*, *disk unit* and *user*, i.e. the XWIN-NMR data path.

jconv fxdata.num

where *num* is an integer number. This converts the FX dataset fxdata.num.

jconv converts all JNMR parameters which have an XWIN-NMR equivalent. First, the JNMR parameter EXMOD is interpreted. If it is set to a certain name, ***jconv*** checks the existence of an XWIN-NMR parameter set with that name. If it exists, it is copied to the destination dataset. If it does not exist, a standard parameter set (*standard1D* for 1D data) is copied. Then ***jconv*** converts all JNMR parameters which have an XWIN-NMR equivalent and overwrites the values of the parameter set which was previously copied. The parameters of the XWIN-NMR parameter set which do not have a JNMR equivalent keep their original values. If you frequently convert Jnmr data, with typical values of EXMOD, you might want to create the XWIN-NMR parameter sets with the corresponding names. This can be done by reading a library parameter set with ***rpar***, modify it with ***eda*** and ***edp*** and then store it with ***wpar***.

INPUT FILES

<$JNMR>/

`<jdata.ext>` - Jeol raw data

If JNMR is not set, the input data path is prompted for.

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - XWIN-NMR 1D raw data
`acqu` - XWIN-NMR acquisition parameters
`acqus` - XWIN-NMR acquisition status parameters
`audita.txt` - acquisition audit trail

<du>/data/<user>/nmr/<name>/<expno>/pdata/1/

`proc` - XWIN-NMR processing parameters
`procs` - XWIN-NMR processing status parameters
`jnmrpar` - original Jeol parameter file

For 2D and 3D data, the raw data are stored in the file `ser` and the additional

parameter files `acqu2(s)`, `acqu3(s)`, `proc2(s)` and `proc3(s)` are created.

USAGE IN AU PROGRAMS

JCONV(jname,uxname,uxexp,uxdisk,uxuser)

SEE ALSO

vconv, conv, convsys, convdta

vconv

NAME

vconv - convert Varian type data to Bruker XWIN-NMR type data

DESCRIPTION

The command ***vconv*** converts Varian data, which were measured with the VNMR program, to XWIN-NMR format.

vconv
shows a list of VNMR data in the directory defined by the environment variable VNMR. If VNMR is not set, you are prompted for the VNMR data path. After selecting a VNMR dataset, you are prompted for the output *name*, *expno*, *disk unit* and *user*, i.e. the XWIN-NMR data path.

vconv vdata.fid
searches for *vdata.fid* in the directory defined by the environment variable VNMR. If VNMR is not set, ***vconv*** searches in the current XWIN-NMR data directory. When the specified data are found, you are prompted for the output *name*, *expno*, *disk unit* and *user*, i.e. the XWIN-NMR data path.

vconv vdata.fid name expno du user
as above but now the dataset *vdata.fid* is converted to the specified XWIN-NMR dataset without prompting the user.

vconv <path>/vdata.fid
if the specified dataset is found, you are prompted for the output *name*, *expno*, *disk unit* and *user*, i.e. the XWIN-NMR data path.

vconv <path>/vdata.fid name expno du user
as above but now the dataset *vdata.fid* is converted to the specified XWIN-NMR dataset without prompting the user.

If, in the second and fourth example, a part of the XWIN-NMR data specification is skipped the remaining parts are prompted for. Furthermore, you do not have to specify the extension *.fid* of the Vnmr dataset.

vconv converts all VNMR parameters which have an XWIN-NMR equivalent. First, the VNMR parameter SEQFIL is interpreted. If it is set to a certain name, ***vconv*** checks the existence of an XWIN-NMR parameter set with that name. If it exists, it is copied to the destination dataset. If it does not exist, a standard pa-

rameter set (*standard1D* for 1D data) is copied. Then ***vconv*** converts all VNMR parameters which have an XWIN-NMR equivalent and overwrites the values of the parameter set which was previously copied. The parameters of the XWIN-NMR parameter set which do not have a VNMR equivalent keep their original values. If you frequently convert Vnmr data, with typical values of SEQFIL, you might want to create the XWIN-NMR parameter sets with the corresponding names. This can be done by reading a library parameter set with ***rpar***, modify it with ***eda*** and ***edp*** and then store it with ***wpar***.

VNMR	XWIN-NMR	VNMR	XWIN-NMR
ct	NS(status)	rfl/rfp	OFFSET
d1	D1	rfl1/rfp1	OFFSET(2D)
date	DATE	rfl2/rfp2	OFFSET(3D)
dfrq	BF2	rp	PHC0
dfrq2	BF3	rp/lp	PHC0/PHC1
dmf	P31	rp1/lp1	PHC0/PHC1(2D)
dn	DECNUC	rp2/lp2	PHC0/PHC1(3D)
dn2	DECBNUC	seqfil	PULPROG
dof	O2	sfrq	BF1
dof2	O3	solvent	SOLVENT
fb	FW	spin	RO
fn	SI	ss	DS
lp	PHC1	sw	SW_h
np	TD	sw1	SW_h(2D)
nt	NS(foreground)	sw2	SW_h(3D)
pp	P3	temp	TE
pslabel	AUNM	tn	NUCLEUS
pw	P0	tof	O1
pw90	P1		

Table 10.2

The original VNMR parameter file `procpar` is stored in the XWIN-NMR processed data directory. You can check this ascii file for possible parameters which could not be converted.

Table 10.2 shows the Varian parameters and there XWIN-NMR equivalent.

vconv can handle Unity and Gemini data acquired with VNMR 4.1 or newer. Data from older Varian spectrometers or acquired with older software versions might also work, but have not been tested by Bruker.

INPUT FILES

<du>/data/<user>/nmr/<vdata>.fid

or

<VNMR>/<vdata>.fid/

`fid` - the VNMR raw data
`procpar` - the parameters
`text` - title file

OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`fid` - XWIN-NMR 1D raw data
`acqu` - XWIN-NMR acquisition parameters
`acqus` - XWIN-NMR acquisition status parameters
`audita.txt` - acquisition audit trail

<du>/data/<user>/nmr/<name>/<expno>/pdata/1

`proc` - XWIN-NMR processing parameters
`procs` - XWIN-NMR processing status parameters
`procpar` - VNMR parameter file

For 2D and 3D data, the raw data are stored in the file `ser` and the additional parameter files `acqu2(s)`, `acqu3(s)`, `proc2(s)` and `proc3(s)` are created.

USAGE IN AU PROGRAMS

VCONV(vname, xwname, xwexpno, xwdisk, xwuser)

SEE ALSO

jconv, conv, convsys, convdta

Chapter 11

XWIN-NMR Interface/processes

This chapter describes commands which are related to the User interface and XWIN-NMR processes. Each user can set up hit/her own interface including the XWIN-NMR menu, colours, printer usage etc. Commands are described for following processes on the screen, storing them in the history file or killing them. Online help is described as far as it can be started from the command line.

auditcheck

NAME

auditcheck - check the audit trail consistency

DESCRIPTION

The command ***auditcheck*** checks the audit trail consistency of the current dataset. This means it checks if the audit trail files exist and if they are internally consistent.

The acquisition command that creates the raw data, usually ***zg***, creates the acquisition audit trail file `audita.txt` and insert the first entry. Any acquisition command that modifies/updates the raw data, e.g. ***go*** makes an additional entry. Furthermore, any command that changes one or more acquisition status parameters makes an additional entry.

The processing command that creates the processed data, e.g. ***em***, creates the processing audit trail file `auditp.txt` and inserts the first entry. Any processing command that modifies/updates the processed data, e.g. ***ft***, makes an additional entry. Furthermore, any command that changes one or more processing status parameters makes an additional entry.

The files `audita.txt` and `auditp.txt` can be viewed from a UNIX shell or from the Windows Explorer using a text editor. Each file entry contains the following elements:

- *Number*: the entry number (1, 2, 3, ...)
- *When* : starting date and time of the command
- *Who* : user who starts the command (the user that started XWIN-NMR)
- *Where* : location where the command started (the computer host name)
- *What* : command and associated parameters, e.g.

The last line of the file is a checksum which looks like:

$$ 24 EB 5D 82 76 AD F2 2B 7E D2 A1 35 7B B5 C4 D5

auditcheck uses this line for the consistency check.

INPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`audita.txt` - acquisition audit trail

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`auditp.txt` - processing audit trail

follow

NAME

follow - show active XWIN-NMR commands, updating the table

DESCRIPTION

The command ***follow*** displays a list of currently active XWIN-NMR commands. For each command, it shows the module which executes the command, the dataset on which the command works, the process status and the operating system process ID. When a command has finished, it disappears from the list. When the last XWIN-NMR command has finished, the list disappears.

SEE ALSO

show, kill

gdcheck

NAME

gdcheck - generate data checksum

DESCRIPTION

The command ***gdcheck*** generates a data checksum. It updates the audit trail files. It takes one argument and can be used as follows:

gdcheck
make the processing audit trail consistent

gdcheck raw
make the acquisition audit trail consistent

gdcheck is, for example, required if a dataset has been manipulated with third party software. In that case the audit trail would be inconsistent, i.e. the command ***auditcheck*** would report an inconsistency error. ***gdcheck*** updates the audit trail file with a new data checksum and adds the entry

Unknown data manipulation detected

After this, ***auditcheck*** would report:

Unknown data manipulation

In 2D and 3D data, ***gdcheck*** adds a data checksum. For 1D data, a data checksum is automatically created by processing commands. In 2D and 3D, however, processing commands do not create a data checksum because this would be too time consuming. If it is required ***gdcheck*** allows you to create it.

INPUT AND OUTPUT FILES

<du>/data/<user>/nmr/<name>/<expno>/

`audita.txt` - acquisition audit trail

<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/

`auditp.txt` - processing audit trail

hist

NAME

hist - set the XWIN-NMR history

SYNTAX

hist off | on | error

DESCRIPTION

The command ***hist*** sets the XWIN-NMR history function. The command takes one argument which can have three possible values:

hist off - no logging
hist on - all processes, commands and error messages are logged
hist error - all commands and error messages are logged

We recommend to set ***hist*** to ***on*** or ***error***. In case of problems, you can easily trace the command(s) which caused them. The information is stored in the file:

`<xwhome>/prog/curdir/<USER>/history`

This file cannot be viewed from XWIN-NMR but, for example, from the Windows Explorer or from a UNIX shell.

Note that the `history` file is emptied when you restart XWIN-NMR which means the history of the previous XWIN-NMR session is lost. In case of problems, you should first make a copy of the history file before you restart XWIN-NMR. Note that a long XWIN-NMR session, especially with automation can create a very large the `history` file. Therefore, it is useful to regularly check the size of the file or simply restart XWIN-NMR after each (automation) session.

The history function can also be set in the User Interface which is opened with the command ***setres***.

OUTPUT FILES

<xwhome>/prog/curdir/<USER>/

`history` - XWIN-NMR history file

hoff, hon

NAME

hoff - switch off the online description of XWIN-NMR buttons
hon - switch on the online description of XWIN-NMR buttons

DESCRIPTION

The command ***hoff*** switches off the online description of a XWIN-NMR buttons. The command ***hon*** switches this function on. The online description appear when the cursor is moved over the button.

Note that the effect of ***hoff*** and ***hon*** only counts for the current XWIN-NMR session. When you (re)start XWIN-NMR, the online description is set according to the User Interface which can be set up with the command ***setres***.

SEE ALSO

setres

kill

NAME

kill - show active XWIN-NMR commands and allow to kill them

DESCRIPTION

The command ***kill*** displays a list of all active XWIN-NMR commands. If you click on a command, it will be killed immediately.

A running acquisition should not be stopped with ***kill*** because this would leave an inconsistent dataset. Instead, the commands ***halt*** or ***stop*** should be used for this purpose.

If you only want to view all active commands without killing any of them, you can do that with the command ***follow*** or ***show cmd***.

SEE ALSO

show cmd, follow, halt, stop

setdef

NAME

setdef - switch error message acknowledgment on/off

DESCRIPTION

The command ***setdef*** is mainly used to switch the error message acknowledgement function on or off. It must be entered in the form:

setdef ackn no - commands continue without acknowledgment

setdef ackn ok - commands require acknowledgment before continuing

Note that (re)starting XWIN-NMR always sets ***setdef ackn*** to its default value which is ***ok***.

setdef can also be used to switch the storage of standard output and standard error message off or on. In this case it must be entered in the form:

setdef stdout ok - store standard output message

setdef stdout no - do not store standard output messages

The equivalent for standard error messages is ***setdef stderr ok/no***.

OUTPUT FILE

<xwhome>/prog/curdir/<user>

`stdout.num` - standard XWIN-NMR output file for ***setdef stdout ok***
`stderr.num` - standard XWIN-NMR error file for ***setdef stderr ok***

setres

NAME

setres - edit the XWIN-NMR user resources

DESCRIPTION

The command ***setres*** opens a dialog box in which you set the resources for your personal XWIN-NMR interface. This includes the following items:

- Menu layout: *standard*, *extended*, *horizontal* or *UXNMR*
- Colors for spectrum, axis, integrals and baseline
- Background color of the data field
- Online help: short description when moving the mouse over buttons
- Status function: *all datasets*, *current dataset* or *off*
- History flag: log all processes and error messages [1]
- Editor: editor used for commands like ***edau***, ***edpul***, ***setti*** etc.
- Plotter: printer used for all datasets (overrules the printer set with ***edo***)
- ZGsafety: acquisition does not overwrite an existing FID

INPUT AND OUTPUT FILE

<home>/.xwinnmr-<hostname>/

`resources` - ascii file containing all User Interface settings

where

<home> is the users home directory
<hostname> is the hostname of the computer

1. The log file is: <xwhome>/prog/curdir/<user>/history

show

NAME

show - display active modules, last datasets and active commands

SYNTAX

show proc|cmd|object

DESCRIPTION

The command ***show*** displays information about active commands, datasets and modules. Depending on the argument, the information is sorted:

1. ***show proc*** lists XWIN-NMR modules which are in memory, the commands they currently execute and the datasets on which these commands are executed.
2. ***show cmd*** lists all commands which are currently executed or which are scheduled for execution, together with the datasets they work on and the modules they use. It works like the command ***follow*** except that it does not update the command table when a command finishes.
3. ***show object*** shows a list of datasets which were read during this XWIN-NMR session, together with the commands and modules that currently work one them. You can click an entry to read the corresponding dataset.

SEE ALSO

follow, status, kill, dir

status

NAME

status - define which command status messages are shown

SYNTAX

status <type>

DESCRIPTION

The command ***status*** defines which command status messages are shown. XWIN-NMR commands display various messages during execution. These messages appear in the status line, the line below the command line at the bottom of the XWIN-NMR window.

The following four different message types are available:

- ***status all***
  Messages of all currently active commands are shown, including messages from commands which do not run on the current foreground data. Note that when several commands run simultaneously, they may send status message at the same moment and only one of them would appear on the screen.
- ***status auto***
  Only messages from commands which work on the current foreground dataset are shown.
- ***status cmd***
  Shows a list of active commands. If you select one, XWIN-NMR disables the status messages for all other active commands.
- ***status no***
  No message will be displayed.

The status can also be set in the User Interface which is opened with the command ***setres***.

SEE ALSO

show, follow

xhelp

NAME

xhelp - open an online help document

SYNTAX

xhelp <document_path>

DESCRIPTION

The command ***xhelp*** allows you to open an online help document from the command line. It uses the Acrobat Reader for display which means it only read files with the extension `.pdf`. ***xhelp*** takes one argument; the pathname to the document. This can be an absolute pathname like in:

xhelp C:\users\guest\yourfile.pdf

It can also be a relative pathname in which must be a sub directory of

<xwhome>/prog/docu/english, for example:

xhelp xwinproc/au.pdf

INPUT FILES

<xwhome>/prog/docu/english/*

Chapter 12

NMR Suite files

This chapter describes the files which are involved in XWIN-NMR processing. Furthermore, the main acquisition related files are described.

For each file, the commands that typically create or modify the file are specified. If no command is specified, the file is delivered with XWIN-NMR and not modified by any command. Furthermore, the commands that typically interpret a file are specified. If no command is specified, the file is meant to be read from a UNIX shell or Windows Explorer or to be used by a external program. Note that the specified commands can be started, manually, from the XWIN-NMR command line or, automatically, from an AU program or ICON-NMR. Files that are created by ICON-NMR are not described here but in the ICON-NMR manual. Note that only the most important commands that access a certain file are mentioned here.

For each file, the file type is specified which has one of the following letters:

- *a* - ascii file. It can be opened by a text editor.
- *j* - JCAMP-DX file. It can be opened with a text editor and interpreted by any software which supports JCAMP-DX.
- *b* - binary file. Data file with consecutive 32-bit integer values.
- *e* - binary executable
- *d* - directory with files and/or sub directories
- *t* - Tcl/Tk script

Besides single command names, the following terms are used:

- *proc. cmds*: processing commands
- *proc. cmds(r)*: processing commands that work on raw data (see chapter 1.5)
- *proc. cmds(p)*: processing commands that work on processed data (see chapter 1.5)
- *<name> menu*: the file is accessed interactively from the menu <name>

1D data files

File	Created or modified by	Interpreted by	Description (file type)
<du>/data/<user>/nmr/<name>/<expno>			
fid	***zg***, ***genfid***	proc. cmds (r)	raw data (b)
format.temp	***zg***	***view***, ***plot***	format for pdata/<procno/*parm.txt* (a)
acqu	***eda***, ***rpar***	***zg***	acquisition parameters (j)
acqus	***zg***	***dpa***, proc. cmds (r)	acquisition status parameters (j)
For other acquisition related files see the Acquisition reference Manual			
<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/			
1r	***em***, ***ft***, ***ef***	proc. cmds (p)	real processed data (b)
1i	***em***, ***ft***, ***ef***	***pk***, ***apk***	imaginary processed data (b)
auditp.txt	proc. cmds		processing audit trail (j)
base_info	***basl*** menu	***bcm***	baseline correction coefficients (a)
baslpnts	***basl*** menu	***sab***	spline baseline correction points (a)
intrng	***abs***, ***rmisc***	***plot***, ***li***	integral regions (a)
meta	***edg***	***view***, ***plot***	plot parameters for ***plot*** (j)
meta.ext	***edgx***	***viewx***, ***plotx***	extended plot parameters for ***plotx*** (j)
outd	***edo***	***plot***, ***lpa*** ***autoplot***	output parameters, e.g. the printer (j)
peaklist	***ppp***	***mdcon***	peak list for mixed deconvolution (a)
parm.txt	***view***, ***plot***	***view***, ***plot***	parameters to be printed on the plot (a)
peaks	***pp***, ***plot***	***pp***, ***plot***	list of all peaks in the entire spectrum (b)
pp.dx	***ppj***		peak list in JCAMP-DX format (j)
proc	***edp***, ***rpar***	proc. cmds	processing parameters (j)
procs	***em***, ***ft***, ***abs***	***dpp***, ***plot***, proc. cmds (p)	processing status parameters (j)

1D data files

File	Created or modified by	Interpreted by	Description (file type)
reg	integrate menu	***plot***, ***plotx***	region with the reference peak if PSCAL = *ireg* or *pireg* (a) or plot limits if LIMITS = region or regions to be expanded with ***plotx***
title	***setti***	***view***, ***plot***	plot title (a)

2D data files

Filename	Created or modified by	Interpreted by	Description (file type)
<du>/data/<user>/nmr/<name>/<expno>			
ser	**zg**, **genser**	proc. cmds (r)	raw data (series of FIDs) (b)
acqu	**eda**, **rpar**	**zg**	F2 acquisition parameters (j)
acqu2	**eda**	**zg**	F1 acquisition parameters (j)
acqus	**zg**	**dpa** proc. cmds (r)	F2 acquisition status parameters (j)
acqu2s	**zg**	**dpa** proc. cmds (r)	F1 acquisition status parameters (j)
<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/			
2rr	**xfb**, **xf2**	proc. cmds (p)	real processed data (b)
2ir	**xfb**, **xf2**	proc. cmds (p)	F2-imaginary processed data (b)
2ri	**xfb**, **xf1**	proc. cmds (p)	F1-imaginary processed data (b)
2ii	**xfb**, **xf1**	proc. cmds (p)	F2/F1-imaginary processed data (b)
proc	**edp**, **rpar**	proc. cmds	F2 processing parameters (j)
proc2	**edp**, **rpar**	proc. cmds	F1 processing parameters (j)
procs	proc. cmds	**dpp**, **plot**, proc. cmds (p)	F2 processing status parameters (j)
proc2s	proc. cmds	**dpp**, **plot**, proc. cmds (p)	F1 processing status parameters (j)
meta	**edg**	**view**, **plot**	plot parameters for **plot** (j)
level	**edlev**	**view**, **plot**	contour levels (b)
int2d	**int2d**	**view**, **plot**, **li**	integral regions and integral values (a)
n2r1	proc. cmds	**view**, **plot**	negative full F1 projection (b)
n2r2	proc. cmds	**view**, **plot**	negative full F2 projection (b)
p2r1	proc. cmds	**view**, **plot**	positive full F1 projection (b)
p2r2	proc. cmds	**view**, **plot**	positive full F2 projection (b)

2D data files

Filename	Created or modified by	Interpreted by	Description (file type)
f1projn	**f1projn**	**view**, **plot**	range of columns and 1D data path of the negative partial F1 projection (a)
f1projp	**f1projp**	**view**, **plot**	range of columns and 1D data path of the positive partial F1 projection (a)
f2projn	**f2projn**	**view**, **plot**	range of rows and 1D data path of the negative partial F2 projection (a)
f2projp	**f2projp**	**view**, **plot**	range of rows and 1D data path of the positive partial F2 projection (a)
f1disco	**f1disco**	**view**, **plot**	range of columns, reference row and 1D data path of F1 disco projection (a)
f2disco	**f2disco**	**view**, **plot**	range of rows, reference column and 1D data path of F2 disco projection (a)
f1sum	**f1sum**	**view**, **plot**	range of columns and 1D data path of F2 sum projection (a)
f2sum	**f2sum**	**view**, **plot**	range of rows and 1D data path of F2 sum projection (a)
portfolio.por	**xwinplot**	**autoplot**	XWIN-PLOT portfolio stored under the dataset procno (a)
t1par	**edt1**	**ct1**, **pft2**, **pd**	Relaxation parameters (j)
ct1t2.out	**ct1**, **ct2**, **dat1**, **lstp**		T1/T2 value and a list of relaxation points in XWIN-NMR 3.0 and older (a)
ct1t2.txt	**ct1**, **ct2**, **dat1**, **lstp**		T1/T2 value and a list of relaxation points in XWIN-NMR 3.1 and newer (a)
t1t2.dx	**ct1**, **ct2**, **dat1**, **lstp**		T1/T2 value and a list of relaxation points in JCAMP-DX format (j)
t1peaks	**pd**, **pft2**	**ct1**, **ct2** **simfit**	peaks positions and intensities (b)
t1ints	**pd**, **pft2**	**ct1**, **ct2**	integral ranges and areas (b)

2D data files

Filename	Created or modified by	Interpreted by	Description (file type)
t1ascii	***simfit asc***	***simfit asc***	list of data points (a)
t1elim	***pd***, ***pft2***	***elim***	eliminated relaxation points (j)

3D data files

Filename	Created or modified by	Interpreted by	Description (file type)
<du>/data/<user>/nmr/<name>/<expno>			
ser	***zg***	proc. cmds (r)	raw data (series of FIDs) (b)
acqu	***eda***, ***rpar***	***zg***	F3 acquisition parameters (j)
acqu2	***eda***, ***rpar***	***zg***	F2 acquisition parameters (j)
acqu3	***eda***, ***rpar***	***zg***	F1 acquisition parameters (j)
acqus	***zg***	***dpa***, proc. cmds (r)	F3 acquisition status parameters (j)
acqu2s	***zg***	***dpa***, proc. cmds (r)	F2 acquisition status parameters (j)
acqu3s	***zg***	***dpa***, proc. cmds (r)	F1 acquisition status parameters (j)
<du>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/			
3rrr	***tf3***	proc. cmds (p)	real processed data (b)
3irr	***tf3***	***tf3p***	F3 imaginary processed data (b)
3rir	***tf2***	***tf2p***	F2 imaginary processed data (b)
3rri	***tf1***	***tf1p***	F1 imaginary processed data (b)
3iii	***tf3***, ***tf2***, ***tf1***	***tf2***, ***tf1***	imaginary data for FnMODE = QF (b)
dsp3d	***display***	***display***	Compressed processed data (b)
proc	***edp***, ***rpar***	proc. cmds	F3 processing parameters (j)
proc2	***edp***, ***rpar***	proc. cmds	F2 processing parameters (j)
proc3	***edp***, ***rpar***	proc. cmds	F1 processing parameters (j)
procs	proc. cmds	***dpp***, ***plot***, proc. cmds (p)	F3 processing status parameters (j)
proc2s	proc. cmds	***dpp***, ***plot***, proc. cmds (p)	F2 processing status parameters (j)
proc3s	proc. cmds	***dpp***, ***plot***, proc. cmds (p)	F1 processing status parameters (j)

User defined settings

File	Created or modified by	Interpreted by	Description (file type)
<userhome>/.xwinnmr-<hostname>/			
resources	**setres**	**zg**, **plot**	XWIN-NMR resource settings (j)
default.por	**search**	**search**	default portfolio (a)
autoshim	**gradshim**	**gradshim**	gradient shimming directory (d)
<xwhome>/prog/curdir/<user>			
history	**hist on**		history of commands and error messages (a)
curdat	**edc**, **new**	**edc**, **new**	currently displayed dataset (j)
individual_user_notebook.txt	***Help***		individual user notebook (a)

AU Programs

Filename	Created or modified by	Interpreted by	Description (file type)
<xwhome>/prog/au/src.exam			
*		**expinstall**	Bruker AU program sources (a)
<xwhome>/prog/au/bin			
*	**compileall cpluser**,**edau**	**xau** [a]	User defined AU executables (e)
*	**compileall cplbruk**,**edau**	**xau**	Bruker AU executables (e)
<xwhome>/prog/include/			
aucmd.h		**edau**	AU macro definitions (a)
<xwhome>/prog/include/inc			
*		**edau**	AU macro and inclusion files (a)
<xwhome>/exp/stan/nmr/au			

a. Note that AU programs can be started by entering **xau <name>** or simply **<name>**

NMR Suite resources

File	Interpreted by	Description (file type)
<xwhome>/prog/app-defaults/		
XWinNmr	XWIN-NMR	XWIN-NMR resources (a)
XWinNmr-Colors	XWIN-NMR	XWIN-NMR colors (a)
XWinPlot	**`xwinplot`**	XWIN-PLOT resources (a)
Pulse-Display	**`pulsdisp`**	Pulse display resources (a)
ShapeTool	**`stdisp`**	Shape tool resources (a)
Edte	**`edte`**	Temperature setup resources (a)
EdAcb	**`edacb`**	Amplifier Control Board setup resources (a)
GsDisp	**`gs`**	Go setup resources (a)
XSearch	**`search`**	Dataset search resources (a)

Documentation

Files	Interpreted by	Description (file type)
<xwhome>/prog/docu/english		
xwin-proc/pdf	***Help***	Manuals for acquisition, processing, AU programs, installation, release letter etc. (d)
avance/pdf	***Help***	Avance spectrometer users guide (d)
xwpman/pdf	***Help***	XWIN-PLOT manuals (d)
iconman/pdf	***Help***	ICON-NMR manuals (d)

NMR Suite executables

File	Description (file type)
<xwhome>/prog	
cpr/xcpu	XWIN-NMR graphics program (e)
cpr/cpr	XWIN-NMR command interpreter (e)
mod	XWIN-NMR modules (d)
tcl	Tcl/Tk scripts for ICON-NMR, GLP, Prosol etc. (d)
au/bin	Bruker and User AU programs executables (d)

Spectrometer configuration

Filename	Created or modified by	Interpreted by	Description (file type)
<xwhome>/conf/instr/			
curinst	**cf**	**cf**	spectrometer name (a)
patchlev			XWIN-NMR patchlevel (a)
hardware.exam	**cf** **makelist**		example hardware list containing all possible hardware devices (a)
nmr_user_ notebook.txt		**Help**	notebook for all NMR users (a)
system_ notebook.txt		**Help**	notebook for system administrator (a)
probehead	**edhead**	**lock**,**edlock**, **getprosol** **edprosol**	current probe definition (a)
probeheads	**edhead**	**edhead**	probe parameter files (d)
<xwhome>/conf/instr/<instrum>			
2Hlock	**edlock**	**lock**, **lopo**	2H lock table (a)
bacs_params	**cfbacs**, **cf**		sample changer information (a)
uxnmr.info	**cf**		spectrometer configuration overview (a)
cortab	**cortab**	**zg**	amplifier power and phase correction tables (d)
uxnmr.par	**cf**		spectrometer configuration (j)
nuclei	**cf**, **ednuc**	**edasp**	nuclei table (a)
scon	**edscon**	**zg**	spectrometer fine tuning settings (j)
specpar	**edsp**, **edasp**	**zg**	routing information
modulenames	**cf**	**edsp**, **edasp**	HPPR modulenames
hardware_ list		**cf**	list of hardware components that are not automatically detected (a)

Spectrometer configuration

Filename	Created or modified by	Interpreted by	Description (file type)
<xwhome>/conf/instr/			
curinst	***cf***	***cf***	spectrometer name (a)
mas_params	***cfmas***	***masi***, ***mase***	information about the MAS unit (a)
prosol	***edprosol***	***getprosol***	probe/solvent dependent parameter files (d)
autoshim	***gradshim***	***gradshim***	gradient shimming files (d)
bbis_bla1	***cf***	***ii***, ***zg***	linear amplifier information (a)
bbis_bla2	***cf***	***ii***, ***zg***	linear amplifier information (a)
etc.			

Format files

File	Created or modified by	Interpreted by	Description (file type)
<xwhome>/exp/stan/nmr/form/			
proc.e		**`edp`**	processing parameters (1D, 2D, 3D) (a)
outd.e		**`edo`**	output parameters (a)
curd.e		**`edc`**	current data path parameters (a)
curd2.e		**`edc2`**	second data path parameters (a)
edp3.e		**`edp3`**	F3 processing parameters (3D) (a)
edp2.e		**`edp2`**	F2 processing parameters (2D, 3D) (a)
edp1.e		**`edp1`**	F1 processing parameters (2D, 3D) (a)
used_ from.e		**`edc2`** **`used_from`**	source 2D data path parameters (a)
<xwhome>/exp/stan/nmr/form/proc.l			
normdp		**`dpp`**	processing status parameters to be viewed (a)
normlp		**`lpp`**	processing status parameters to be printed (a)
normpl		**`view`**, **`plot`**	processing status parameters to be plotted (a)
normplot			all processing status parameters (reference) (a)
<xwhome>/exp/stan/nmr/form/curd.l			
normdp		**`dpc`**	main data path parameters to be viewed (a)
normlp		**`lpc`**	main data path parameters to be printed (a)
normpl		**`view`**, **`plot`**	main data path parameters to be plotted (a)
normplot			all data path parameters (reference) (a)
<xwhome>/exp/stan/nmr/form/plot.l			
normdp		**`dpg`**	main plot parameters to be viewed (a)

Format files

File	Created or modified by	Interpreted by	Description (file type)
normlp		***lpg***	main plot parameters to be printed (a)
normpl		***view***, ***plot***	main plot parameters to be plotted (a)
For acquisition related format files see the Acquisition Reference Manual			

Chapter 13

Graphics commands

This chapter describes XWIN-NMR graphics commands. Although the XWIN-NMR graphics is normally controlled by clicking menu buttons, most commands can also be entered on the command line. If you have the option ***Online Help*** switched ***on*** (see ***setres***), moving the mouse cursor over the menu buttons will show the corresponding command line commands. This feature can also be switched on and off with the commands ***hon*** and ***hoff*** respectively.

1D Main menu

a/r	Toggle between absolute and relative Y-axis units
all	Display the full spectrum
calib	Calibrate the spectrum.
dots	Toggle between dotted and solid line display
dpl1	Set plot region to display region, adjust Hz/cm, keep CX
dpl2	Set plot region to display region, adjust CX, keep Hz/cm
dpl3	Set plot region to display region; adjust F2, keep CX, Hz/cm
dual	Switch to dual display menu
fid	Show 1D raw data (FID)
grid	Switch on/off grid
gridxy	Switch between X/Y/X-Y grid along axis labels
hosc1	Horizontal expansion (*2) around the center
hosc2	Horizontal compression (/2) around the center
hosc3	Horizontal reset to fit the window
hz/ppm	Toggle axis labels between Hz and ppm
integ	Switch to the integral menu
lesh1	Left-shift on screen by 1/2 window
phase	Enter interactive phase correction menu
plot_1d	Set plot region to displayed region (adjust HZCM) and plot
plotreg	Display the current plot region of the spectrum.
real	Display the real processed data
rish1	right-shift data on screen by 1/2 window
shu	Shuffle data on the screen
spec	Display the processed data (spectrum)
sw-sfo1	Set SW to displayed region and O1/O1P to center
to2d	Switch to last 2D spectrum.
to3d	Switch to last 3D spectrum.
unshu	Unshuffled display of data points from two channels

1D Main menu

utilities	Enter utilities mode
v*2	Scale up data by a factor of 2
v*8	Scale up data by a factor of 8
v/2	Scale down data by a factor of 2
v/8	Scale down data by a factor of 8
vdown	Put the data (zero intensity) at the bottom of the window
vreset	Reset vertical scaling to show highest peak
vup	Put data (zero intensity) in the center of the window
yax	Toggle the y-axis on/off.

1D Utilities menu

defpts	Interactively define a list of baseline points for ***sab***.
manipul	Change the intensity of a data point interactively.
noisereg	Set NOISF1 and NOISF2 for ***sino*** to displayed region.
peaklist	Setup ’peaklist’ file for deconvolution (***mdcon***)
return	Return to the main 1D menu
setmaxi	Set maximum intensity (MAXI) for peak picking.
setmi	Set the minimum intensity (MI) for peak picking.
seto1	Set the O1 value interactively.
seto2	Set the O2 value interactively
seto3	Set the O3 value interactively
sigreg	Set SIGF1-SIGF2 for ***sino*** to displayed region

1D Phase menu

biggest	phase (zero order) on the biggest (highest) peak
cursor	bind cursor to spectrum to select reference peak
m*2	Double 1st order phase increment
m/2	Divide 1st order phase increment by 2
mreset	Reset 1st order phase increment to default
return	Return to the main 1D menu

1D Baseline menu

def-pts	Bind cursor to spectrum for defining spline baseline points
expon	Switch to exponential baseline function
polynom	Set baseline function to polynomial
reset	Reset all polinomial coefficients to zero
sine	Switch to sine baseline function

1D Integration menu

ai-cal	Scale integrals relative to dataset scaled with calibint
c*2	scale up the current integral by 2
c/2	scale down the current integral by 2
calibint	calibrate the current integral
clear	clear all integrals
delcur	Clear current integral
i*2	Magnify all integrals by 2
i/2	Reduce all integrals by 2
idown	Vertical reset to show the first point of the current integral
imag	Display the imaginary processed data
iup	Vertical reset to show last point of the current integral
mi*2	Decrease sensitivity for slope/bias correction by 2
mi/2	Increase sensitivity for slope/bias correction by 2
mireset	Reset sensitivity for slope/bias correction to default
read	Read the current integral regions
return	Return to the main 1D menu

Acquisition menu

dspft	show unphased spectrum during acquisition online FT
dspmc	show magnitude spectrum during acquisition online FT
dsppk	show phased spectrum during acquisition online FT
frq	Switch on the online FT
tim	Display time domain.

1D Dual menu

ddiff	Show difference between spectrum 1 and 2.
decexp2	Decrement EXPNO of dataset 2 by 1.
decexp3	Decrement EXPNO of dataset 3 by 1.
decprc2	Decrement PROCNO of dataset 2 by 1.
decprc3	Decrement PROCNO of dataset 3 by 1.
incexp2	Increment EXPNO of dataset 2 by 1.
incexp3	Increment EXPNO of dataset 3 by 1.
incprc2	Increment PROCNO of dataset 2 by 1.
incprc3	Increment PROCNO of dataset 3 by 1.
return	Return to the main 1D menu.
sum	Display sum of first and second dataset.
undo	Undo sum or difference display.
v2*2	Scale up second spectrum data by 2.
v2/2	Scale down second spectrum by 2.
vdown1	Put spect. 1 (zero intensity) at the bottom of the window.
vdown2	Put spect. 2 (zero intensity) at the bottom of the window.
vup1	Put spect. 1 (zero intensity) in the center of the window.
vup2	Put spect. 2 (zero intensity) in the center of the window.

1D Winfunc menu

delta	Set the increment used by the + and - buttons.
em	Switch to exponential window multiplication.
fadjl	Expand FID by 2 keeping it left aligned.
fadjr	Compress FID by 2 keeping it left aligned.
fhreset	Horizontal reset of the FID to fit the window size.
fv*2	Magnify the on-screen FID by 2.
fv/2	Reduce the on-screen FID by 2.
fvdown	Put the FID window at the bottom of the window.
fvreset	Vertical reset of the FID to fit the window size.
fvup	Put the FID in the middle of the window.
gb+	Increase GB by ’delta’, then reprocess the data.
gb-	Decrease GB by ’delta’, then reprocess the data.
gm	Switch to Gaussian multiplication.
lb+	Increase LB by ’delta’ and reprocess the FID.
lb-	Decrease LB by ’delta’ and reprocess the FID.
ph-mod	Define phasing mode used in spectrum calculation.
qsin	Switch to square sine window function.
sinm	Switch to sine window function.
ssb+	Increase SSB by ’delta’ and reprocess the data.
ssb-	Decrease SSB by ’delta’ and reprocess the data.
ssb=1	Set SSB=1 and reprocess the data.
ssb=2	Set SSB=2 and reprocess the data.

2D Main menu

all	Display the full spectrum.
calib	F2 and F1 frequency calibration.
co	Switch to contour display.
defplot	Set plot region to the region displayed on screen.
exp	Display last expanded region.
grid	Switch on/off grid.
hz/ppm	Toggle axis labels between Hz and ppm.
in	Display spectrum as intensity color map.
integ	Switch to the integral menu.
limits2d	Prompt user for plot limits and set display accordingly.
ob	Display spectrum in oblique view.
phase	Switch to the phase menu.
plot_2d	Set plot region to displayed region and plot.
plotreg	Display the current plot region of the spectrum.
posneg1	Toggle between display of positive/negative/both peaks.
serial	Switch to raw 2D data display.
strp	Set STSR, STSI for strip transform to displayed region.
to1d	Switch to last 1D spectrum.
to3d	Switch to last 3D spectrum.
utilities	Enter utilities mode.
v*2	Scale up data by a factor of 2.
v*8	Scale up data by a factor of 8.
v/2	Scale down data by a factor of 2.
v/8	Scale down data by a factor of 8.
vreset	Reset vertical scaling to show highest peak.

2D utilities

+	Go to the next row or column.
-	Go to the previous row or column.
col	Define a column interactively.
f1dcalc	Define region and calculate F1 disco projection.
f1ddisp	Display F1 disco projection.
f1ext	Display F1 external projection (defined with ***edc2***).
f1n	Display F1 negative full projection.
f1p	Display F2 positive full projection.
f1pcalc	Define and calculate F1 positive partial projection.
f1pdisp	Display current F1 positive partial projection.
f1scalc	Define and calculate F1 sum partial projection.
f1sdisp	Display current F1 sum partial projection.
f2dcalc	Define and calculate F2 disco projection.
f2ddisp	Display current F2 disco projection.
f2ext	Display F2 external projection (defined with ***edc2***).
f2n	Display F2 negative full projection.
f2p	Display F2 positive full projection.
f2pcalc	Define and calculate F2 positive partial projection.
f2pdisp	Display current F2 positive partial projection.
f2scalc	Define and calculate F2 sum partial projection.
f2sdisp	Display current F2 sum partial projection.
on/off	Switch display if the 2D spectrum on/off.
row	Define a row interactively.
scan	Scan rows and columns interactively.
vr*2	Scale up selected row/column/projection by 2.
vr/2	Scale down selected row/column/projection by 2.

2D phase menu

big1	phase (zero order) on biggest (highest) peak in window 1.
big2	phase (zero order) on the biggest peak in window 2
big3	phase (zero order) on biggest peak in window 3
col	Select a column in the 2D window
cur1	Bind cursor to spectrum 1 to select reference peak
cur2	Bind cursor to spectrum 2 to select reference peak
cur3	Bind cursor to spectrum 3 to select reference peak
move1	Move current row/column to 1D window 1
move2	Move current row/column to 1D window 2
move3	Move current row/column to 1D window 3
return	Return to the main 2D menu.
row	Select a row in the 2D window
v1/2	Scale down data in active 1D window by 2
v1*2	Scale up data in active 1D window by 2
v1reset	Vertical reset of data in active 1D window

2D serial menu

cols	Show columns of 2D raw data.
next	Show next row/column of 2D raw data.
numb	Prompt for the FID number to be displayed.
prev	Show previous row/column of 2D raw data.
ser	Display first row/column of 2D raw data.

2D Integral menu

clear	Clear all integral regions from the screen.
read	Prompt for an integral region file and read it.
return	Return to the main 2D menu.

3D Main menu

auto_3d	Switch 3D movie on/off.
display	Calculate and display 3D spectrum.
move	Allows to move the cube by moving the mouse.
move_reset	Reset cube position to default.
project	Switch to 2D projection menu.
reset_rot	Reset the cube to default orientation.
reset_siz	Reset the cube to default size.
scale_down	Reduce the on-screen cube.
scale_up	Magnify the on-screen cube.
scan3d	Switch to 2D scan menu.
scan_12	Allow scanning for F1-F2 planes
switch_2d	Create 2D data (procno) from 3D with F3-F2 parameters.
x90	Rotate the cube 90° around the x-axis.
y90	Rotate the cube 90° around the y-axis.
z90	Rotate the cube 90° around the z-axis.

3D Scan menu

auto_23	Auto scan through F2-F3 planes
auto_13	Auto scan through F12-F3 planes
auto_12	Auto scan through F1-F2 planes
scan_23	Interactive scan through F2-F3 planes
scan_13	Interactive scan through F1-F3 planes
scan_12	interactive scan through F1-F2 planes
set_planes	Set initial planes in all 3 dimensions for scanning
stepb_12	Display previous 1-2 plane
stepb_13	Display previous 1-3 plane
stepb_23	Display previous 2-3 plane
stepf_12	Display next 1-2 plane
stepf_13	Display next 1-3 plane
stepf_23	Display next 2-3 plane

Chapter 14

Bruker addresses

Germany

Bruker BioSpin GmbH
Silberstreifen
D-76287 Rheinstetten
Tel: (++49) (721) 51 61 0
Fax: (++49) (721) 51 71 01

Bruker BioSpin Software Department
Silberstreifen
D-76287 Rheinstetten
Tel: (++49) (721) 5161 440
Fax: (++49) (721) 5161 480

http://www.bruker-biospin.de

ftp server: ftp.bruker.de

Email: ut@bruker.de (sales)
mbu@bruker.de (service)
applik@bruker.de (application)
nmr-software-support@bruker.de (software)
license@bruker.de (licenses)

USA

Bruker BioSpin Corporation
15 Fortune Drive
Manning Road
Billerica, MA. 01821-3991

Tel: (++1)(978) 667 9580 195 (center)
(++1)(978) 667 9580 444 (application)
Fax: (++1)(978) 667 6168 (center)
(++1)(978) 667 2955 (application)

http://www.bruker-biospin.com
ftp server: ftp.bruker.com

Email: applab@bruker.com
center@bruker.com
software@bruker.com

Switzerland

Bruker BioSpin AG
Industriestraße 26
CH-8117 Fällanden

Tel: (++41)(1) 8 25 91 11
Fax: (++41)(1) 8 25 96 96

web server: www.bruker.ch
E-Mail: epweb@bruker.ch

all_ap@bruker.ch

France

Bruker BioSpin S.A.
34, rue de l'industrie
F-67166 Wissembourg/Cedex

Tel: (++33)(3) 88 73 68 00
Fax: (++33)(3) 88 73 68 79

E-Mail: support-rmn@bruker.fr (customer support)

England

Bruker BioSpin LTD.
Banner lane

Coventry CV4 9GH
Tel: (++44)(2476) 855200
Fax: (++44)(2476) 465317

Email:service@bruker.co.uk
apps@bruker.co.uk

Our webpage

http://www.bruker-biospin.de/analytic/nmr-dep/about/offices/contact.htm

provides contact addresses of our facilities and offices worldwide

Command - Index

A-xx: Acquisition Reference Manual **P-xx: Processing Reference Manual**

Index