



TOPSPIN

Processing Reference Guide

Copyright (C) 2004 by Bruker BioSpin GmbH

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means without the prior consent of the publisher.

Part Number H9776SA1 V2/April 16th 2004

Product names used are trademarks or registered trademarks of their respective holders.

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

Address: Bruker BioSpin GmbH
Software Department
Silberstreifen
D-76287 Rheinstetten
Germany
Phone: +49 (721) 5161 455
Fax: +49 (721) 5161 943
E-mail: nmr-software-support@bruker-biospin.de
FTP: <ftp.bruker.de> / <ftp.bruker.com>
WWW: www.bruker-biospin.de / www.bruker-biospin.com

Chapter 1	Introduction	P-3
	1.1 About this manual.	P-3
	1.2 Conventions	P-3
	1.3 About dimensions.	P-4
	1.4 About time and frequency domain data	P-4
	1.5 About raw and processed data	P-5
	1.6 About digitally filtered Avance data.	P-7
	1.7 Usage of processing commands in AU programs	P-8
	1.8 Clicking commands from the TOPSPIN menu	P-8
Chapter 2	TOPSPIN parameters	P-9
	2.1 About TOPSPIN parameters	P-9
	2.2 Parameter values.	P-11
	2.3 Parameter files	P-12
	2.4 List of processing parameters.	P-13
	2.5 Processing status parameters	P-32
	2.6 Relaxation parameters	P-38
Chapter 3	1D Processing commands	P-43
Chapter 4	2D processing commands	P-131
Chapter 5	3D processing commands	P-245
Chapter 6	Print/Export commands	P-291
Chapter 7	Analysis commands	P-305
Chapter 8	Dataset handling	P-335
Chapter 9	Parameters, lists, AU programs	P-371
Chapter 10	Conversion commands	P-397
Chapter 11	TOPSPIN Interface/processes	P-423
	Index	

Chapter 1

Introduction

1.1 About this manual

This manual is a reference to TOPSPIN 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

ProcPars - 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

<tshome> - the TOPSPIN home directory (default C:\Bruker under Windows)

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

TOPSPIN 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 \ 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:

```
<dir>/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:

```
<dir>/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*, *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
- 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 use the **XCMD** macro which takes an **TOPSPIN** command as argument. Examples are:

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

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

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

1.8 Clicking commands from the TOPSPIN menu

This manual describes all processing commands as they can be entered on the command line. However, they can also be clicked from the **TOPSPIN** popup menus. Most commands can be found under the **Processing** or **Analysis** menu. The corresponding command line commands are specified in square brackets.

Chapter 2

TOPSPIN parameters

2.1 About TOPSPIN parameters

TOPSPIN parameters are divided in acquisition and processing 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 by entering **edp** or clicking the *Procvars* tab, 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, by entering **dpa** or clicking the *Acqvars* tab. 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, by **dpp** or by clicking the *Procvars* tab. Most processing status parameters get the value of the correspond-

ing 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

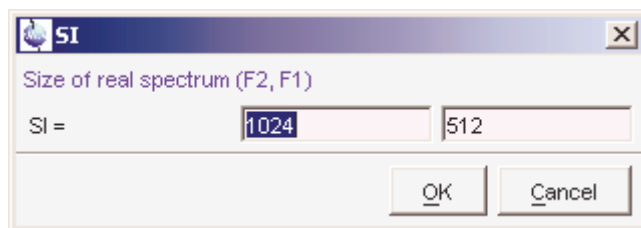
Parameters which are set or modified by processing commands. These can be:

- 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. For example, the parameter SI:

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

The dimensionality of the dataset is automatically recognized. For example, for a 2D dataset the following dialog box is offered:



Although status parameters are normally not changed by the user, a command like **s si** allows you to do that. This, however, could make the dataset inconsistent. Likewise, if an acquisition status parameter is changed with **s**, 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 editor opened by entering **edp** or clicking **Procpars**. 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

TOPSPIN parameters are stored in various files in the dataset directory tree.

In a 1D dataset:

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

acq - acquisition parameters

acqus - acquisition status parameters

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

proc - processing parameters

procs - processing status parameters

In a 2D dataset:

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

acq - F2 acquisition parameters

acq2 - F1 acquisition parameters

acqus - F2 acquisition status parameters

acq2s - F1 acquisition status parameters

<dir>/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

In a 3D dataset:

```
<dir>/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

<dir>/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
```

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 composite 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 from the parameter editor which can be opened by entering **edp** or clicking **Procpars**. Alternatively, you can set parameters by entering 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 **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.

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

ASSWID - region excluded from second highest peak search

- used in 1D datasets
- takes a float value (Hz, default is 0)
- interpreted by **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 *sfil/qfil*.

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)

COROFFS - correction offset for FID baseline correction

- 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:
`<tshome>/exp/stan/nmr/filt/1d`
 and must be set up from a command shell. One standard file called `three-point` is delivered with TOPSPIN.

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:

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

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).

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 **li**, **lipp**, **lippf**
- INTBC has no effect on integrals which were created interactively in the **Integration** mode.

INTSCL - scale 1D integrals relative to a reference dataset

- used in 1D datasets
- takes an integer value
- interpreted by *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 in the integration mode.

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.

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***, **li**, **lipp***
- only peaks with an intensity smaller than MAXI will appear in the peak list. MAXI can also be set from the **pp** dialog box and, interactively, in peak picking mode.

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 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****, ***li***, ***lipp****
- only peaks with an intensity greater than MI will appear in the peak list. MI can also be set from the ***pp*** dialog box and, interactively, in peak picking mode.

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.

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:

$$\text{OFFSET} = (\text{SFO1}/\text{SF}-1) * 1.0\text{e}6 + 0.5 * \text{SW} * \text{SFO1}/\text{SF}$$

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 *gf*, the equation:

$$\text{OFFSET} = (\text{SFO1}/\text{SF}-1) * 1.0\text{e}6$$

is used.

PC - peak picking sensitivity

- used in 1D datasets
- takes a float value
- interpreted by **pp***, **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 in Phase correction mode 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 in Phase correction mode on 1D and 2D datasets
- interpreted by **pk**, **xfb_p**, **xf2_p**, **xf1_p**, **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 correction mode. 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****, ***li***, ***lipp****
- the values of PSCAL have the following meaning:.

PSCAL	Peak used as reference for vertical scaling
<i>global</i>	The highest peak of the entire spectrum.
<i>preg</i>	The highest peak within the plot region.
<i>ireg</i>	The highest peak within the regions specified in the <i>reg</i> file. If the <i>reg</i> file does not exist, <i>global</i> is used.
<i>pireg</i>	as <i>ireg</i> , but the peak must also lie within the plot region.
<i>sreg</i>	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, <i>global</i> is used.
<i>psreg</i>	as <i>sreg</i> but the peak must also lie within the plot region.
<i>noise</i>	The intensity of the noise.

Table 2.7

- For PSCAL = *ireg* or *pireg*, the *reg* file is interpreted. The *reg* file can be created in interactive ***integration*** mode 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 TOPSPIN. 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***, **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***, **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.CDCI3. For all common nucleus/solvent combinations, a scaling region file is delivered with TOPSPIN. 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 $0 < \text{TDoff} < \text{TD}$ 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 $\text{TDoff} < 0$ -TDoff zeroes are prepended at the beginning and:
 - for $\text{SI} < (\text{TD} - \text{TDoff})/2$ 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** or **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.

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
<i>em</i>	Exponential	LB	<i>em</i>
<i>gm</i>	Gaussian	GB, LB	<i>gm</i>
<i>sine</i>	Sine	SSB	<i>sinm</i>
<i>qsine</i>	Sine squared	SSB	<i>qsin</i>
<i>trap</i>	Trapezoidal	TM2, TM1	<i>tm</i>
<i>sinc</i>	Sine	SSB, GB	<i>sinc</i>
<i>qsinc</i>	Sine squared	SSB, GB	<i>qsinc</i>
<i>traf</i>	Traficante (JMR, 71 , 1987, 237)		<i>traf</i>
<i>trafs</i>	Traficante (JMR, 71 , 1987, 237)		<i>trafs</i>

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. TOPSPIN only runs on computers with byte order little endian. However, TOPSPIN's predecessor XWIN-NMR also runs on SGI workstations which are big 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 **s 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 **s 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 commands **xfb big** and **tf3 big** on a Windows or Linux PC store the data in big endian although the computer is little 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 TOPSPIN 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***, ***apk0f*** in 1D datasets
- set interactively in Phase correction mode 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 in Phase correction mode 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.

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.

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.

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 2.9 shows the EDGUESS table for an inversion recovery experiment, with 2 components. The initial guess for I[0] must be such that the total value of all 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

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. 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*.

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

- used in pseudo 2D relaxation datasets
- takes the value *area* or *intensity* (default is *intensity*)
- interpreted by **pd**, **ct1**, **dat1** and **simfit**
- Before you run **pd**, both the integral ranges and peak positions should be determined (see **rspec** 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 FITTYP. 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** (pseudo 2D data)
- Starting with START, every INC point (1D) or row (pseudo 2D) is used for relaxation analysis.

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** (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** (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 point or row used is $START + n*INC$.

Chapter 3

1D Processing commands

This chapter describes all TOPSPIN 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.

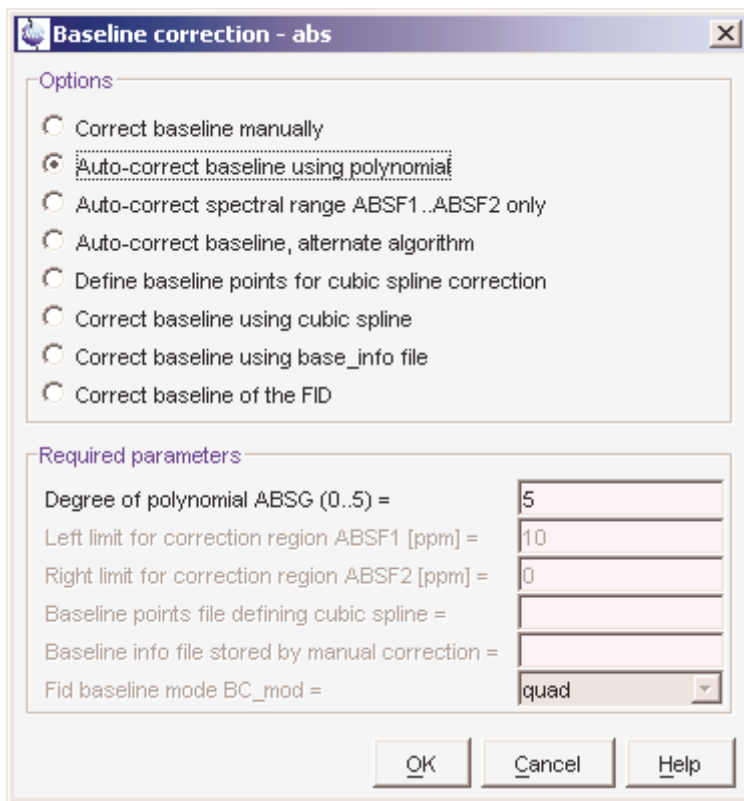
abs, absf, absd, bas

NAME

- abs - automatic baseline correction
- absf - automatic baseline correction of the plot region
- absd - automatic baseline correction with a different algorithm
- bas - open the baseline correction dialog box

DESCRIPTION

Baseline correction commands can be started on the command line or from the baseline correction dialog box. The latter is opened with the command **bas**



This dialog box offers several options, each of which selects a certain command

for execution.

Auto-correct baseline using polynomial

This option selects the command **abs** for execution. It 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 `in-trng` (integral regions). The remaining part of the spectrum is considered baseline and used to fit the polynomial function.

abs also interprets the parameters ABSL, AZFW, AZFE and ISEN. Since these parameters apply to integration rather than baseline correction, they do not appear in the **bas** dialog box. They do appear in the integration dialog box (command **int**). Data points greater than $ABSL * (\text{standard deviation})$ 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.

Auto-correct spectral range ABSF1 .. ABSF2 only

This option selects the command **absf** for execution. It works like **abs** except that it only corrects the spectral region which is determined by the processing parameters ABSF1 and ABSF2.

Auto-correct baseline, alternate algorithm

This option selects the command **absd** for execution. It works like **abs**, except that it uses a different algorithm¹. 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

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


the baseline around the small peak which can then be integrated. Usually **absd** is followed by **abs**.

To display the integral regions determined by one of the above commands:

1. Right-click inside the data window and select *Spectrum components*
2. Check the entry *Integrals* and click *OK*

The integral regions are also used by various commands which calculate spectral integrals like **li**, **lipp** and **plot**.

If you run a command like **abs** from the command line, you have to make sure that the required parameters are already set. Click the *Procpars* tab or enter **edp** to do that.

If automatic baseline correction does not give satisfactory results, you can apply an interactively determined polynomial, exponential, sine or spline baseline correction. This can be started with the first entry of the **bas** dialog box, by clicking the  button in the toolbar or by entering **.bas1** on the command line.

The **bas** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **bas** dialog box, with **edp** or by typing **absg**, **absf1** etc.:

ABSG - degree of the polynomial (input of **abs**, **absf**, **absd**)

ABSF1 - low field (left) limit of the region corrected by **absf**

ABSF2 - high field (right) limit of the region corrected by **absf**

set from the **int** dialog box, with **edp** or by typing **abs1**, **azfw** etc.:

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

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

1r - real processed 1D data

proc - processing parameters

OUTPUT FILES

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

1r - real processed 1D data

procs - processing status parameters

intrng - integral regions (output of ***abs***, ***absf*** , ***absd***)

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

ABS

ABSD

ABSF

SEE ALSO

bcm, sab, bc, .basl

add, addfid, addc, adsu

NAME

add - add two datasets multiplying one of them with DC
addfid - add two FIDs multiplying one of them with DC
addc - add the constant DC to the current dataset
adsu - open the add/subtract dialog box

DESCRIPTION

Addition commands can be entered on the command line or started from the add/subtract/multiply dialog box. The latter is opened with the command **adsu**.

This dialog box offers several options, each of which selects a certain command for execution.

Add a 1D spectrum point-wise

This option selects the command **add** for execution. It adds the second dataset, multiplied with the constant DC, to the current dataset. **add** performs a point to point addition which is independent of the spectrum calibration. The result is stored in the current dataset. DC can be set by entering **dc** on the command line or in the *Procpars* pane. If the second dataset has not been defined yet, the add/subtract dialog box is opened. Here you can define the second dataset and start the **add** command. **add** works on raw or on processed data, depending on the value of DATMOD. For DATMOD = raw, **add** adds the raw data of the current and second dataset but stores the result as processed data in the current dataset. As such, the raw data of the current dataset are not overwritten.

Add a 1D spectrum ppm/Hz-wise

This option selects the command **duadd** for execution. It works like add, except that it 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.

duadd is useful when the two input spectra are:

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

Add / subtract - add

Options

- ☒ Add a spectrum of same size point-wise: current + DC * second
- ☐ Add a spectrum of same or different size ppm/Hz-wise: current + DC * second
- ☐ Add a FID: current + DC * second
- ☐ Add a constant
- ☐ Multiply with 1D spectrum/fid: current * second
- ☐ Multiply with constant
- ☐ Multiply with -1
- ☐ Divide by a 1D spectrum/fid: current / second

Required parameters

Constant DC =

NAME (2nd spectrum) =

EXPNO =

PROCNO =

USER =

DIR =

Shift 2nd spectrum by: [ppm] =

Apply command to raw / processed data: DATMOD =

Add corresponding ppm or hz values =

OK Cancel Help

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

Furthermore, the second spectrum can be shifted by a user defined number of ppm. The parameter *ppm* or *hz* is only relevant 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.

Add an FID

This option selects the command **addfid** for execution. It adds two 1D raw datasets multiplying one of them with the factor DC. The result is stored in the current dataset. It works like **add** with DATMOD = raw, except that it overwrites the raw data.

Add a constant

This option selects the command **addc** for execution. It adds the value of DC to the current dataset. It works on raw or processed data, depending on the value of DATMOD. The result is stored as processed data in the current dataset.

If you run a command like **add** from the command line, you have to make sure that the required parameters are already set. Click the **Procpars** tab or enter **edp** to do that.

The **adsu** command can be used on 1D or 2D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **adsu** dialog box, with **edp** or by typing **dc**, **datmod** etc.:

DC - multiplication factor

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

INPUT FILES

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

fid - current raw data (input of **add/addc** if DATMOD = raw)

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

1r, 1i - current processed data (input of **add/addc** if DATMOD = proc)

proc - processing parameters

curdat2 - definition of the second dataset

<dir2>/data/<user2>/nmr/<name2>/<expno2>/

fid - second raw data (input of **add** if DATMOD = raw, **addfid**)

<dir2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

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

OUTPUT FILES

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

fid - current raw data (output of **addfid**)

audita.txt - acquisition audit trail (output of **addfid**)

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

lr, li - current processed data (output of **add** and **addc**)

procs - processing status parameters

auditp.txt - processing audit trail (output of **add** and **addc**)

USAGE IN AU PROGRAMS

ADD

ADDFID

ADDC

SEE ALSO

mul, mulc, div

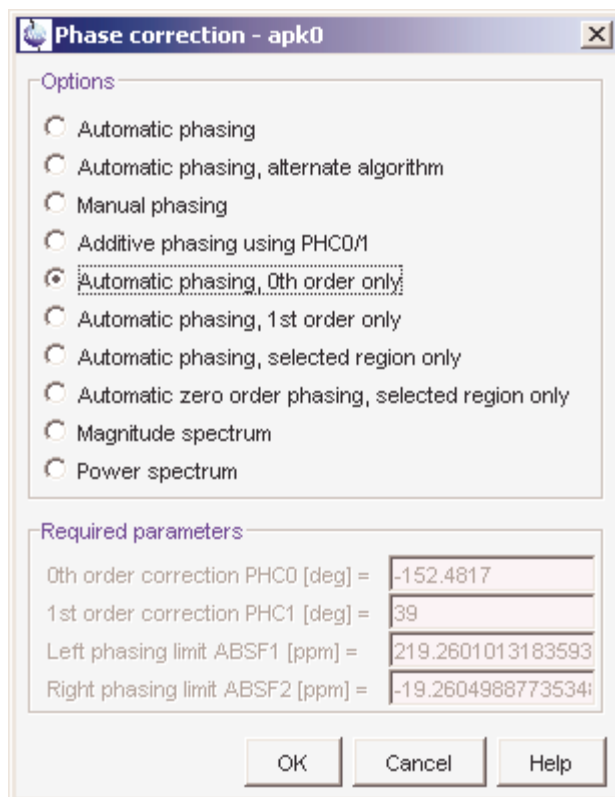
apk0, apk1, apk0f, ph

NAME

apk0 - zero order automatic phase correction
apk1 - first order automatic phase correction
apk0f - zero order automatic phase correction based on a certain spectral region
ph - open the phase correction dialog box

DESCRIPTION

Phase correction commands can be entered on the command line or started from the phase correction dialog box. The latter is opened with the command *ph*.



This dialog box has several options, each of which selects a certain command for execution.

Automatic phasing, 0th order only

This option selects the command **apk0** for execution. It works like **apk** except that it only performs the zero order phase correction.


Automatic phasing, 1st order only

This option selects the command **apk1** for execution. It works like **apk** except that it only performs the first order phase correction.

Automatic zero order phasing, selected region 0th order only

This option selects the command **apk0f** for execution. It works like **apk** except that it only performs the zero order phase correction.

If you run a command like **apk0f** from the command line, you have to make sure that the required parameters are already set. Click the **Procpars** tab or enter **edp** to do that.

If automatic phase correction does not give satisfactory results, you can perform interactive phase correction. This can be started with the entry **Manual phasing** in the **ph** dialog box, by clicking the  button in the toolbar or by entering **.ph** on the command line.

The **ph** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **ph** dialog box, with **edp** or by typing **absf1**, **absf2** etc.:

ABSF1 - low field (left) limit of the region used by **apk0f**

ABSF2 - high field (right) limit of the region used by **apk0f**

OUTPUT PARAMETERS

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

PHC0 - zero order phase correction value (output of **apk0** and **apk0f**)

PHC1 - first order phase correction value (output of **apk1**)

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

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

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

proc - processing parameters

OUTPUT FILES

<dir>/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

APK0

APK1

APK0F

SEE ALSO

apk, apks, apkf, pk, mc, ps, .ph

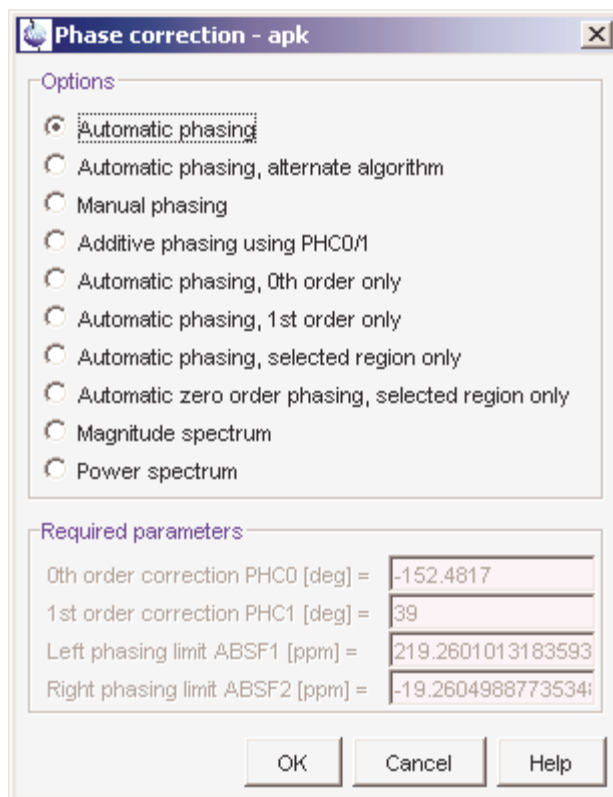
apk, apks, apkf, ph

NAME

apk - automatic phase correction
apks - automatic phase correction with a different algorithm
apkf - automatic phase correction based on a certain spectral region
ph - open the phase correction dialog box

DESCRIPTION

Phase correction commands can be entered on the command line or started from the phase correction dialog box. The latter is opened with the command *ph*.



This dialog box has several options, each of which selects a certain command for execution.

Automatic phasing

This option selects the command **apk** for execution. It 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 processing parameters (**edp**) and as processing status parameters (**dpp**).


Automatic phasing, alternate algorithm

This option selects the command **apks** for execution. It works like **apk** except that it uses a different algorithm which gives better results on certain spectra.

Automatic phasing, selected region only

This option selects the command **apkf** for execution. It 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**.

If you run a command like **apkf** from the command line, you have to make sure that the required parameters are already set. Click the **Procvars** tab or enter **edp** to do that.

If automatic phase correction does not give satisfactory results, you can perform interactive phase correction. This can be started with the entry **Manual phasing** in the **ph** dialog box, by clicking the  button in the toolbar or by entering **.ph** on the command line.

The **ph** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **ph** dialog box, 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**, **s 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

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

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

proc - processing parameters

OUTPUT FILES

<dir>/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

SEE ALSO

apk0, apk1, apk0f, pk, mc, ps, .ph

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** evaluates 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.

When executed on a 2D or 3D dataset, **bc** prompts you for the row and output procno. Alternatively, it can be entered with up to four arguments:

bc <row> <procno> n y

process the specified row and store it under the specified procno. The last two arguments are optional: **n** prevents changing the display to the output 1D data, **y** causes a possibly existing data to be overwritten without warning.

When executed on a dataset with 2D or 3D raw data but 1D processed data¹, **bc** takes one argument:

bc <row>

process the specified row and store it under the current procno.

bc same

process the same row as the previous processing command and store it under the current procno. The **same** option is automatically used by the AU program macro BC. When used on a regular 1D dataset (i.e. with 1D raw data) it has no effect.

bc can also be started from the baseline dialog box which is opened with the command **bas**.

INPUT PARAMETERS

set from the **bas** dialog box, 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

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

fid - raw data (time domain)

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

proc - processing parameters

1. Usually a result of **rsr**, **rsc** or 1D processing command on that 2D or 3D dataset.

OUTPUT FILES

`<dir>/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

bas

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. Click  or enter **.bas1** to change to baseline correction mode.
2. Fit the baseline of the spectrum with a *polynomial*, *exponential* or *sine* function. Click-hold the button **A** and move the mouse to determine the zero order correction. Do the same with the buttons **B**, **C** etc. for higher order corrections until the line matches the baseline of the spectrum.
3. Click  to return. The command **bcm** is automatically executed.

The interactively determined baseline function is stored in the file `base_info`. This file can be stored for general usage with the command **wmisc**. After that, you can read it with **rmisc** on another dataset and run **bcm** to perform the same baseline correction.

bcm can be started from the command line or from the baseline dialog box which is opened with the command **bas**.

INPUT FILES

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

1r - real processed 1D data

proc - processing parameters

base_info - baseline correction coefficients

OUTPUT FILES

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

1r - real processed 1D data

procs - processing status parameters

`auditp.txt` - processing audit trail

USAGE IN AU PROGRAMS

BCM

SEE ALSO

`bas`, `sab`, `.basl`

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

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

fid - raw data (input if DATMOD = raw)

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

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

proc - processing parameters

OUTPUT FILES

<dir>/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

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.

All composite processing commands can be found under the menu:

Processing → More Transforms → Shortcuts

INPUT FILES

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

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

acqus - acquisition status parameters

<dir>/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

<dir>/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, wm

NAME

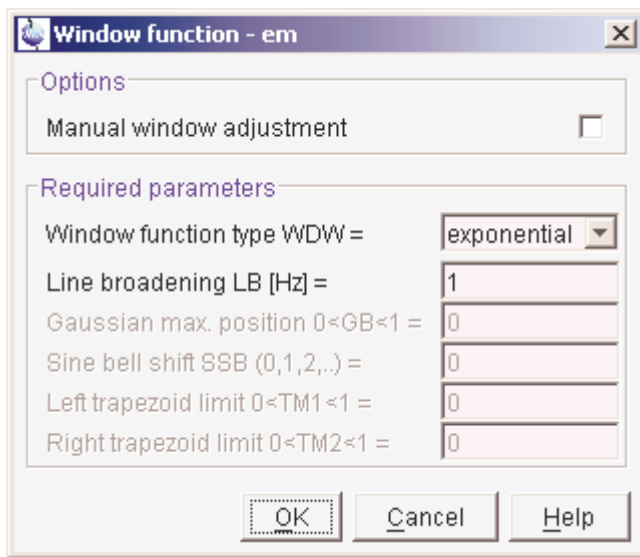
em - exponential window multiplication of the FID

gm - Gaussian window multiplication of the FID

wm - Open the window function dialog box

DESCRIPTION

Window multiplication commands can be entered on the command line or started from the window function dialog box. The latter is opened with the command **wm**.



The parameter section of this dialog box offers several window functions, each of which selects a certain command for execution.

Exponential multiplication

This function selects the command **em** for execution. It 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.

Gaussian multiplication

This function selects the command **gm** for execution. It performs an Gaussian window multiplication of the FID. The result is a Gaussian lineshape after Fourier transform. This lineshape has sharper edges than the lineshape caused by **em**. **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 this 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 with *Manual window adjustment*. 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.

em and **gm** implicitly perform a baseline correction of the FID, according to the processing parameter BC_mod.

When executed on 2D or 3D data, **em** and **gm** take up to four arguments, e.g.:

em <row> <procno> n y

process the specified row and store it under the specified procno. The last two arguments are optional: **n** prevents changing the display to the output 1D data, **y** causes a possibly existing data to be overwritten without warning.

When executed on a dataset with 2D or 3D raw data but 1D processed data¹, **em** and **gm** take one argument, e.g.:

em <row>

process the specified row and store it under the current procno.

em same

process the same row as the previous processing command and store it under the current procno. The **same** option is automatically used by the AU program macros EM and GM. When used on a regular 1D dataset (i.e. with 1D raw data) it has no effect.

If you run a command like **em** from the command line, you have to make sure that the required parameters are already set. Click the **Procpars** tab or enter **edp** to do that.

The **wm** command can be used on 1D or 2D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **wm** dialog box, with **edp** or by typing **lb**, **bc_mod** 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 **s swh**:

SWH - spectral width

INPUT FILES

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

fid - raw data (input if **lr**, **li** do not exist or are Fourier transformed)

acqu - acquisition status parameters

1. Usually a result of **rsr**, **rsc** or a previous 1D processing command on that 2D or 3D data.

<dir>/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

<dir>/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

GM

SEE ALSO

sinm, qsin, sinc, qsinc, tm, traf, trafs

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:

```
<tshome>/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 TOPSPIN and needs to be done on operating system 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

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

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

proc - processing parameters

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

digital filtering file(s)

OUTPUT FILES

<dir>/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, fmc

NAME

fp - 1D Fourier transform +phase correction

fmc - 1D Fourier transform + magnitude calculation

DESCRIPTION

The composite processing command **fp** is a combination of **ft** and **pk**, i.e. it performs a 1D Fourier transform and a phase correction.

fmc is a combination of **ft** and **mc**, i.e. it performs a 1D Fourier transform and a magnitude calculation.

fp and **fmc** automatically perform an FID baseline correction according to BC_mod.

All composite processing commands can be found under the menu:

Processing → More Transforms → Shortcuts

INPUT AND OUTPUT PARAMETERS

see the commands **ft**, **pk** and **mc**.

INPUT FILES

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

fid - raw data (input if **lr**, **li** do not exist or are Fourier transformed)

acqus - acquisition status parameters

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

lr, li - processed data (input if they exist but are not Fourier transformed)

proc - processing parameters

OUTPUT FILES

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

lr, li - processed 1D data (real, imaginary)

procs - processing status parameters

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

FP

FMC

SEE ALSO

ef, efp, gf, gfp

ft, ftf

NAME

ft - 1D Fourier transform

ftf - open the Fourier transform dialog box

DESCRIPTION

The command **ft** Fourier transforms a 1D dataset or a row of a 2D or 3D dataset. It can be started from the command line or from the Fourier transform dialog box. The latter is opened with the command **ftf**

Fourier transform - ft

Options

☒ Standard Fourier transform

☐ Advanced Fourier transform

Required parameters

Size of real spectrum SI [pnts] =	8192
# of fid data points to be used TDef =	0
Index of first output point of strip transform STSR =	0
Total # of output points of strip transform STSI =	0
Fid linear prediction (LP) mode ME_mod =	No LP
# of LP coefficients NCOEF =	0
# of fid data points contributing to backward LP LPBIN =	0
# of fid data points to be predicted TDOFF =	0
Reverse spectrum REVERSE =	No
Weighting factor for first fid point (A*X only) FCOR =	0.5
Apply 5th order phase correction (A*X only) PKNL =	Yes

OK Cancel Help

This dialog box offers two options both of which select the **ft** command for execution.

Standard Fourier transform

This option only allows you to set the parameter SI, the size of the real spectrum.

Advanced Fourier transform

This option allows you to set all FT related parameters.

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). 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 than 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 by entering **dpp** or clicking the **Procvars** tab.

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 . Note that **ft** does not evaluate the processing

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

parameter FT_mod but it does store the Fourier transform mode, as evaluated from the acquisition mode, in the processing status parameter FT_mod. However, 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 becomes the last and the last point becomes 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 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. Backward 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. Note that the command **trf** 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.

When executed on a 2D or 3D dataset, **ft** takes up to three arguments:

ft <row> <procno> *y*

process the specified row and store it under the specified procno. The last argument is optional: *y* causes a possibly existing data to be overwritten without warning.

If you run a command like **ft** from the command line, you have to make sure that the required parameters are already set. Click the **Procvars** tab or enter **edp** to do that.

The **ftf** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **ftf** dialog box, 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 *s 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 *s ft_mod, s 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 *s bytordp*:

BYTORDP - data storage order

INPUT FILES

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

fid - raw data (input if *lr*, *li* do not exist or are Fourier transformed)

acqus - acquisition status parameters

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

lr, *li* - processed data (input if they exist but are not Fourier transformed)

proc - processing parameters

OUTPUT FILES

lr, *li* - processed 1D data (real, imaginary)

procs - processing status parameters

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

FT

SEE ALSO

trf, trfp, ift, ht

gdcon, ldcon, mdcon, ppp, dconpl, dcon

NAME

gdcon - Gaussian deconvolution
ldcon - Lorentzian deconvolution
mdcon - mixed Gaussian/Lorentzian deconvolution
ppp - generate peak list for deconvolution
dconpl - show the result of the last deconvolution
dcon - open the deconvolution dialog box

DESCRIPTION

Deconvolution commands can be entered on the command line or started from the deconvolution dialog box. The latter is opened with the command **dcon**.

Line deconvolution - ldcon

Options

- ☒ Use Lorentzian shape
- ☐ Use Gaussian shape
- ☐ Use mixed shape, auto peak pick into file 'peaklist'
- ☐ Use mixed shape, use peaks from file 'peaklist'
- ☐ Generate file 'peaklist', no deconvolution
- ☐ Re-Display peak list from last deconvolution
- ☐ Display the Lorentz/Gauss curves of the last deconvolution

Required parameters

Left deconvolution limit F1P [ppm] =	16.456960678100586
Right deconvolution limit F2P [ppm] =	-4.106147707164844
Minimum intensity MI [rel] =	1
Maximum intensity MAXI [rel] =	10000
Peak detection sensitivity PC =	1
Peak overlapping factor AZFV [ppm] =	0.1
Destination PROCNO for fitted data =	999

OK Cancel Help

This offers several options, each of which selects a certain command for execution.

Use Lorentzian shape

This option selects the command **ldcon** for execution. It 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.

Use Gaussian shape

This option selects the command **gdcon** for execution. It 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.

Use mixed shape, auto peak pick into file 'peaklist'

This option selects the command **mdcon auto** for execution. It first picks the peaks for deconvolution and stores them in the `peaklist` file. Then it deconvolves the spectrum by fitting a mixed Lorentzian/Gaussian function to these peaks. This command is typically used to deconvolve spectra which cannot be approximated by a pure Lorentzian or a pure Gaussian lineshape.

Use mixed shape, use peaks from file 'peaklist'

This option selects the command **mdcon** for execution. It works like **mdcon auto** except that it uses an existing `peaklist` file. This file must have been created before with **mdcon auto** or **ppp**.

Generate peaklist, no deconvolution

This option selects the command **ppp** for execution. It picks the peaks for deconvolution and stores the result in the file `peaklist`. **ppp** is implicitly executed by **mdcon auto**.

Re-Display peaklist from last deconvolution

This option selects the command **dconpl** for execution. It shows the peaklist (file `dconpeaks.txt`) which was created with the last deconvolution on the current dataset.

Display the Lorentz/Gauss curves of the last deconvolution

This option selects the command **dconpl v** for execution. It shows the in-

dividually fitted peaks and their sum.

The deconvolution commands only work on the displayed region, as expressed by the parameters F1P and F2P. Furthermore, they select peaks according to the peak picking parameters MI, MAXI and PC. They also evaluate 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 deconvolution 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.
- the fitted lineshape which is shown together with the original spectrum in multi-display mode.

All deconvolution commands start by opening the deconvolution dialog box. Note that **ldcon**, **gdcon**, **mdcon** and **ppp** open the dialog window with the respective command selected whereas **dcon** selects the last executed deconvolution command.

INPUT PARAMETERS

set from the **dcon** dialog box, with **edp** or by typing **azfw**, **f1p** etc.:

AZFW - minimum distance in ppm for peaks to be fitted independently
F1P - low field (left) limit of the deconvolution region (= plot region)
F2P - high field (right) limit of the deconvolution region (= plot region)
MI - minimum relative intensity (cm) for peak picking
MAXI - maximum relative intensity (cm) for peak picking
PC - peak picking sensitivity

INPUT FILES

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

1r - real processed 1D data
dconpeaks.txt - peak list (input of **dconp1**)
peaklist - peak list (input of **mdcon**)
proc - processing parameters

OUTPUT FILES

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

1r - real processed 1D data

dconpeaks.txt - peak list (output of *ldcon*, *gdcon*, *mdcon*)

peaklist - peak list (output of *ppp* and *mdcon auto*)

procs - processing status parameters

USAGE IN AU PROGRAMS

LDCON

GDCON

MDCON

PPP

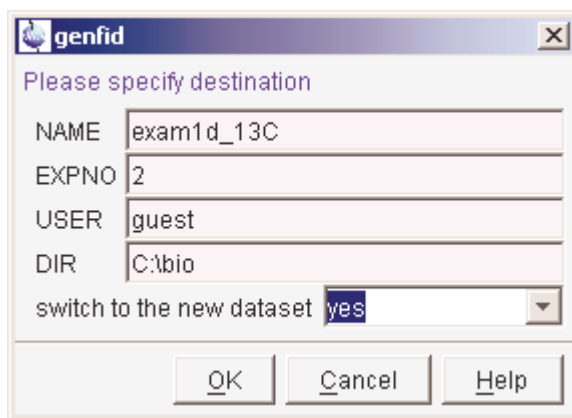
genfid

NAME

genfid - generate pseudo-raw 1D data

DESCRIPTION

The command **genfid** generates pseudo-raw data from processed data. When entered without arguments, it opens a dialog box where you can specify the destination dataset.



genfid is normally used in combination with the command **ift** which performs an inverse Fourier transform, converting a spectrum into an FID. Actually, **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 (types **s td** or **dpa**) is set accordingly; $TD = 2 * SI$.

genfid takes arguments and can be used as follows:

1. **genfid <expno>**

The FID will be stored under the specified *expno*.

2. **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.

You can use any other combination of arguments as long they are entered in the correct order. 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

<dir>/data/<user>/nmr/<name>/<expno1>/pdata/<procno>/

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

OUTPUT FILES

<dir>/data/<user>/nmr/<name>/<expno2>/

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.

All composite processing commands can be found under the menu:

Processing → More Transforms → Shortcuts

INPUT AND OUTPUT PARAMETERS

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

INPUT FILES

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

fid - raw data (input if **lr**, **li** do not exist or are Fourier transformed)

acqus - acquisition status parameters

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

lr, li - processed data (input if they exist but are not Fourier transformed)

proc - processing parameters

OUTPUT FILES

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

lr, li - 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

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

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

1r - real processed 1D data

OUTPUT FILES

<dir>/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

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

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

OUTPUT FILES

<dir>/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

ls, rs

NAME

ls - left shift 1D data NSP points

rs - right shift 1D data NSP points

DESCRIPTION

The command **ls** shifts 1D 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.

rs shifts 1D 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** and **ls** work 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 but also changes from real (absorption) to imaginary (dispersion) or vice versa. Note that this 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

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

fid - raw data (input if DATMOD = raw)

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

lr, li - processed 1D data (input if DATMOD = proc)

proc - processing parameters

OUTPUT FILES

`<dir>/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

RS

SEE ALSO

pk

mc

NAME

mc - 1D magnitude calculation

DESCRIPTION

The command **mc** calculates the magnitude spectrum of a 1D dataset. 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.

mc can also be started from the phase correction dialog box which is opened with **ph**.

INPUT FILES

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

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

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

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

OUTPUT FILES

<dir>/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

ps, pk, apk, trf, trfp

mul, mulc, nm, div, adsu

NAME

mul - multiply two 1D datasets
mulc - multiply 1D data with a constant
nm - negate 1D data
div - divide two 1D datasets
adsu - open the add/subtract/multiply dialog box

DESCRIPTION

Multiplication commands can be entered on the command line or started from the add/subtract/multiply dialog box. The latter is opened with **adsu**.

This dialog box offers several options, each of which selects a certain command for execution.

Multiply with 1D spectrum/fid

This option selects the command **mul** for execution. It multiplies the current dataset with the second dataset. The result is stored in the current dataset.

Multiply with constant

This option selects the command **mulc** for execution. It multiplies the current data with the value of DC.

Multiply with -1

This option selects the command **nm** for execution. It negates the current data which means all data points are multiplied by -1.

Divide by 1D spectrum/fid

This option selects the command **div** for execution. It divides the current dataset by the second dataset.

mul/div perform a complex multiplication/division on complex spectra. This requires that for both the current and second dataset:

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

This is the case for most data that have been acquired in Avance spectrometers.

Add / subtract - mul

Options

- ☐ Add a spectrum of same size point-wise: current + DC * second
- ☐ Add a spectrum of same or different size ppm/Hz-wise: current + DC * second
- ☐ Add a FID: current + DC * second
- ☐ Add a constant
- ☒ Multiply with 1D spectrum/fid: current * second
- ☐ Multiply with constant
- ☐ Multiply with -1
- ☐ Divide by a 1D spectrum/fid: current / second

Required parameters

Constant DC =

NAME (2nd spectrum) =

EXPNO =

PROCNO =

USER =

DIR =

Shift 2nd spectrum by: [ppm] =

Apply command to raw / processed data: DATMOD =

Add corresponding ppm or hz values =

OK Cancel Help

If the above requirements are not fulfilled, real and imaginary data are multiplied/divided pointwise. When a complex operation has been performed, this is reported in the audit trail output file.

mul, ***div***, ***mulc*** and ***nm*** work on raw or on processed data, depending on the value of DATMOD. The result is always stored as processed data in the current dataset. The raw data are not overwritten.

When ***mul*** and ***div*** are started from the command line, they will run without user interaction if the second dataset is already defined (file curdat2). If this is not defined, the ***adsu*** dialog box will be opened. When you run a multiplica-

tion or division command from the command line, make sure that the required parameters are set. Click the *Procpars* tab or enter *edp* to do that.

The *adsu* command can be used on 1D or 2D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the *adsu* dialog box, with *edp* or by typing *dc*, *datmod* etc.:

DC - multiplication factor (input of *mulc*)

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

INPUT FILES

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

fid - raw data (input if DATMOD = raw)

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

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

proc - processing parameters

curdat2 - definition of the second dataset

<dir2>/data/<user2>/nmr/<name2>/<expno2>/

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

<dir2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

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

OUTPUT FILES

<dir>/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

MULC

NM

DIV

SEE ALSO

add, addc, addfid

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:

$$\begin{aligned}R0(i) &= R(i) \cos a(i) - I(i) \sin a(i) \\ I0(i) &= I(i) \cos a(i) + R(i) \sin a(i)\end{aligned}$$

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 values are known. They can be determined, interactively, in Phase correction mode 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 for example AU program **proc_noe**).

pk applies but does not change the processing parameters **PHC0** and **PHC1** (**edp**). It does, however, change the corresponding processing status parameters **PHC0** and **PHC1** (**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 contain 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**) after Fourier transform. This 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**.

The command **pk** can also be started from the phase correction dialog box which is opened with **ph**.

INPUT PARAMETERS

set from the **ph** dialog box, 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

can be viewed with **dpp** or by typing **s phc0**, **s phc1** etc.:

PHC0 - zero order phase correction value (frequency independent)

PHC1 - first order phase correction value (frequency dependent)

INPUT FILES

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

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

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

lr, **li** - processed 1D data (input if they exist)

proc - processing parameters

OUTPUT FILES

`<dir>/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

`mc`, `ps`, `apk`, `trf`, `trfp`

prguide

NAME

prguide - open the Processing Guide

DESCRIPTION

The command **prguide** opens the TOPSPIN Processing Guide (see Figure 3.1). This contains a workflow for processing data, especially suited for new or occasional users. In *Automatic mode*, the Processing Guide will simply execute a processing command when you click the corresponding button. This requires the processing parameters to be set correctly. In interactive mode (*Automatic mode* unchecked), the Processing Guide will, at each step, open a dialog box offering you the available options and required parameters. For example, the phase correction button offers various automatic algorithms as well as an option to switch to interactive phasing mode.

Experienced users normally enter the individual processing commands from the command line. This requires that, for each command, the processing parameters are set correctly.

The Processing Guide can be used for 1D and 2D processing.

SEE ALSO

aqguide, saguide, t1guide

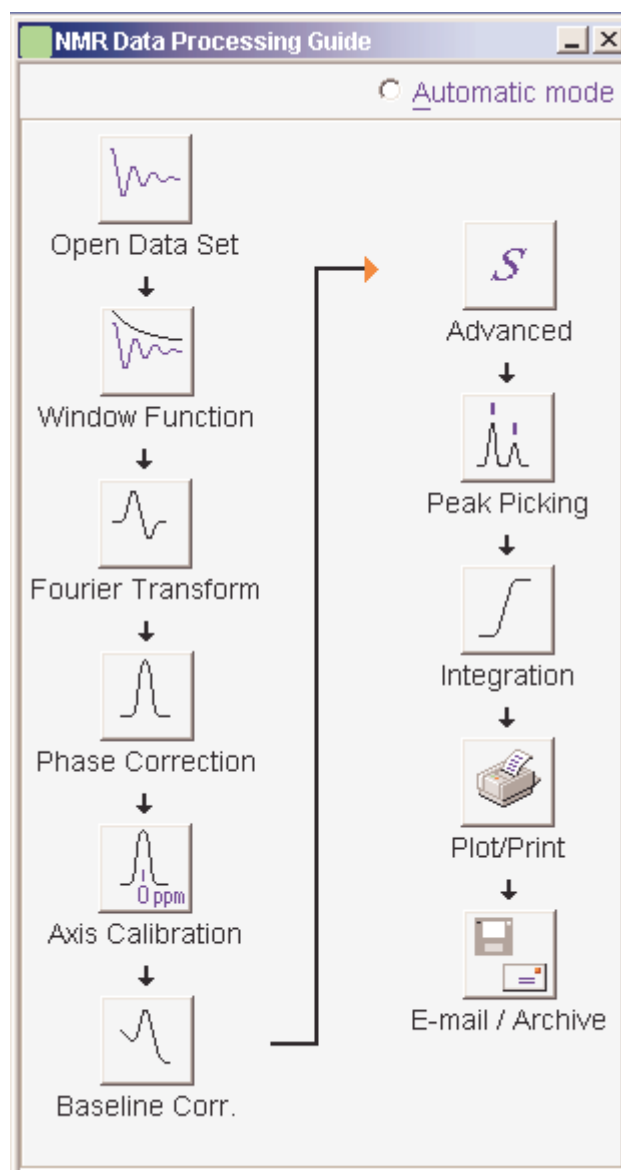


Figure 3.1

ps

NAME

ps - calculate 1D power spectrum

DESCRIPTION

The command **ps** calculates the power spectrum of the 1D 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. If no processed input data exist, **ps** works on the raw data. The result is always stored as the real processed data.

ps can also be started from the phase correction dialog box which is opened with **ph**.

INPUT FILES

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

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

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

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

OUTPUT FILES

<dir>/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, apk, trf, trfp

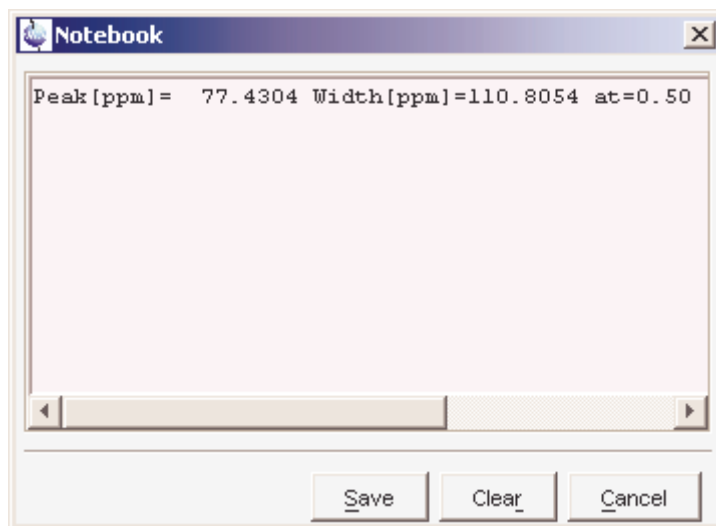
peakw

NAME

peakw - calculate the width of the highest peak in the displayed region

DESCRIPTION

The command **peakw** calculates the peak width at half height of the highest peak of the displayed region. The result is stored in the notebook and displayed on the screen:



The command can also be used with one argument, the height at which the width must be calculated:

peakw <height>

For example, **peakw 0.66** calculates the width of the highest peak in the displayed regions at 66% of the height.

OUTPUT FILES

<userprop>/notebook.txt - notebook text file

SEE ALSO

nbook

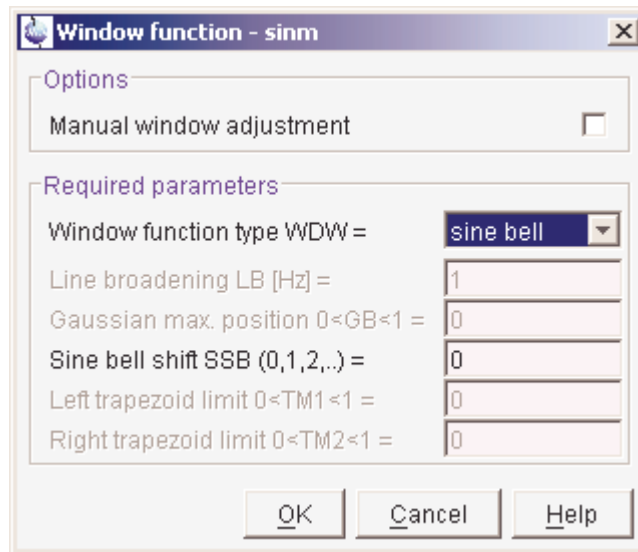
sinm, qsin, sinc, qsinc, wm

NAME

sinm - 1D sine window multiplication
qsin - 1D sine squared window multiplication
sinc - 1D sinc squared window multiplication
qsinc - 1D sinc squared window multiplication
wm - open the window multiplication dialog box

DESCRIPTION

Window multiplication commands can be started from the command line or from the window function dialog box. The latter is opened with the command **wm**:



This dialog box offers several window functions, each of which selects a certain command for execution.

Sine bell

This window function selects the command ***sinm*** for execution. It performs a sine window multiplication, according to the function:

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

where

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

where AQ is an acquisition status parameter and SSB 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.

Squared sine bell

This window function selects the command **qsine** for execution. It performs a sine squared window multiplication, according to the function:

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

where

where AQ is an acquisition status parameter and SSB 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

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

that all values smaller than 2 have the same effect as $SSB = 1$, namely a pure sine function.

Sinc

This window function selects the command ***sinc*** for execution. It 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.

Squared sinc

This window function selects the command ***qsinc*** for execution. It performs a sinc squared window multiplication, according to the function:

where

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

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

and SSB and GB are processing parameters.

The ***sin*** commands implicitly perform a baseline correction of the FID, according to the processing parameter BC_mod.

When executed on 2D or 3D data, the ***sin*** commands take up to four arguments, e.g.:

sinm <row> <procno> n y

process the specified row and store it under the specified procno. The last two arguments are optional: **n** prevents changing the display to the output 1D data, **y** causes a possibly existing data to be overwritten without warning.

When executed on a dataset with 2D or 3D raw data but 1D processed data¹, the **sin*** commands take one argument:

sinm <row>

process the specified row and store it under the current procno.

sinm same

process the same row as the previous processing command and store it under the current procno. The **same** option is automatically used by the AU pro-

1. Usually a result of **rsr**, **rsc** or a previous 1D processing command on that 2D or 3D data.

gram macros **SIN**. When used on a regular 1D dataset (i.e. with 1D raw data) it has no effect.

If you run a command like *sinm* from the command line, you have to make sure that the required parameters are already set. Click the *Procvars* tab or enter *edp* to do that.

The *wm* command can be used on 1D or 2D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the *wm* dialog box, with *edp* or by typing *ssb*, *gb* etc.:

SSB - sine bell shift

GB - Gaussian broadening factor (input of *sinc* and *qsinc*)

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

AQ - Acquisition time (input of *sinm* and *qsinc*)

INPUT FILES

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

fid - raw data (input if *lr*, *li* do not exist or are Fourier transformed)

acqus - acquisition status parameters

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

lr, li - processed data (input if they exist but are not Fourier transformed)

proc - processing parameters

OUTPUT FILES

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

lr, li - processed 1D data (real, imaginary)

procs - processing status parameters

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

SINM

QSIN

SINC

QSINC

SEE ALSO

em, gm, tm, traf, trafs

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 parts 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

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

fid - raw data (input if DATMOD = raw)

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

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

proc - processing parameters

OUTPUT FILES

<dir>/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

SEE ALSO

ft, trf




sab

NAME

sab - spline baseline correction

DESCRIPTION

The command **sab** performs a spline baseline correction. This is based on a pre-defined set of data points which are considered to be a part of the baseline. The regions between these points are individually fitted. In order to execute **sab**, the baseline points must have been determined. You can do this as follows:

1. Click  or enter **.bas1** to change to baseline correction mode.
2. Click  to switch to *Define baseline points* mode
(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 left mouse button at several positions which are part of the baseline.
4. Click  to return. The command **sab** is automatically executed.

The set of baseline points is saved in the file `bas1pnts`. This file can be stored for general usage with the command **wmisc**. After that, you can read it with **rmisc** on another dataset and run **sab** to perform the same baseline correction.

sab can be started from the command line or from the baseline dialog box which is opened with the command **bas**.

INPUT FILES

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

1r - real processed 1D data

bas1pnts - baseline points (points and ppm values)

OUTPUT FILES

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

1r - real processed 1D data

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

SAB

SEE ALSO

bas, .basl, bcm

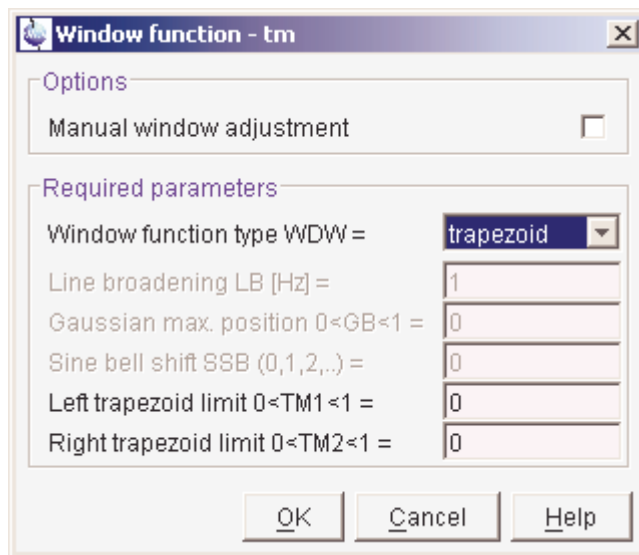
tm, traf, trafs, wm

NAME

tm - trapezoidal window multiplication of the FID
traf - Traficante window multiplication of the FID
trafs - Traficante window multiplication of the FID
wm - Open the window function dialog box

DESCRIPTION

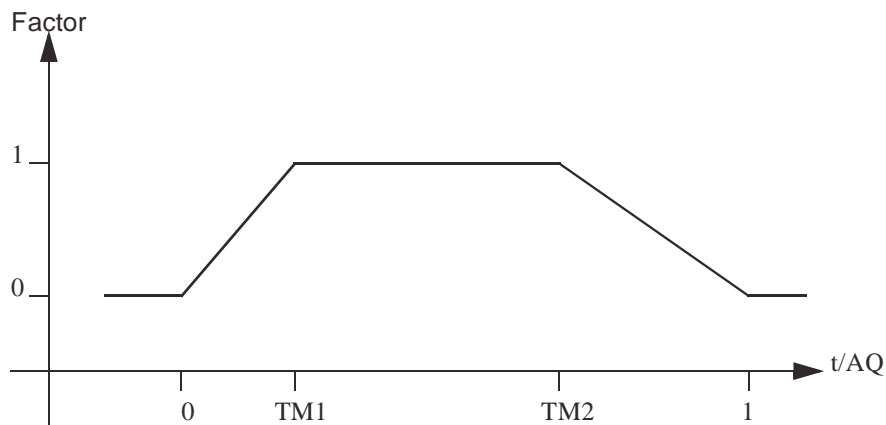
Window multiplication can be executed from the command line or from the window function dialog box. The latter is opened with the command **wm**:



This dialog box offers several window functions, each of which selects a certain command for execution.

Trapezoid

This function selects the command **tm** for execution. It performs a trapezoidal window multiplication of the FID. The rising and falling edge of this function are defined by the processing parameters TM1 and TM2. These represent a fraction of the acquisition time as displayed below.



tm automatically performs an FID baseline correction according to BC_mod.

Traficante and trafic.s/n

This function selects the commands **traf** and **trafs**, respectively, for execution. The algorithms used by these commands are described by D. D. Traficante and G. A. Nemeth in J. Magn. Res., **71** (1987) 237).

tm, **traf** and **trafs** implicitly perform a baseline correction of the FID, according to the processing parameter BC_mod.

When executed on 2D or 3D data, **tm** and **traf*** take up to four arguments, e.g.:

```
tm <row> <procno> n y
```

process the specified row and store it under the specified procno. The last two arguments are optional: *n* prevents changing the display to the output 1D data, *y* causes a possibly existing data to be overwritten without warning.

When executed on a dataset with 2D or 3D raw data but 1D processed data¹, **tm** and **traf*** take one argument, e.g.:

```
tm <row>
```

process the specified row and store it under the current procno.

1. Usually a result of **rsr**, **rsc** or a previous 1D processing command on that 2D or 3D data.

tm same

process the same row as the previous processing command and store it under the current procno. The ***same*** option is automatically used by the AU program macro TM. When used on a regular 1D dataset (i.e. with 1D raw data) it has no effect.

If you run a command like ***tm*** from the command line, you have to make sure that the required parameters are already set. Click the ***Procvars*** tab or enter ***edp*** to do that.

The ***wm*** command can be used on 1D or 2D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the ***wm*** dialog box, with ***edp*** or by typing ***tm1***, ***lb*** etc.:

TM1 - the end of the rising edge of a trapezoidal window (input of ***tm***)

TM2 - the start of the falling edge of a trapezoidal window (input of ***tm***)

LB - Lorentzian broadening factor (input of ***traf****)

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

AQ - acquisition time (input of ***tm***)

INPUT FILES

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

fid - raw data (input if ***lr***, ***li*** do not exist or are Fourier transformed)

acqus - acquisition status parameters

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

lr, ***li*** - processed data (input if they exist but are not Fourier transformed)

proc - processing parameters

OUTPUT FILES

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

lr, ***li*** - 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

trf, trfp

NAME

trf - user defined 1D processing of raw data

trfp - user defined 1D 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 *lr*, *li*). The even points are stored as real data (file *lr*) and the odd points as imaginary data (file *li*). 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 to 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.

When executed on a 2D or 3D dataset, **trf** takes up to four arguments:

trf <row> <procno> n y

process the specified row and store it under the specified procno. The last two arguments are optional: **n** prevents changing the display to the output 1D data, **y** causes a possibly existing data to be overwritten without warning.

When executed on a dataset with 2D or 3D raw data but 1D processed data¹, **trf** takes one argument:

trf <row>

process the specified row and store it under the current procno.

trf same

process the same row as the previous processing command and store it under the current procno. The **same** option is automatically used by the AU program macro TRF. When used on a regular 1D dataset (i.e. with 1D raw data), it has no effect.

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

1. Usually a result of **rsr**, **rsc** or a previous 1D processing command on that 2D or 3D data.

SSB - Sine bell shift for WDW = sine, qsine, sinc or qsinc
 TM1, TM2 - limits of the trapezoidal window for WDW = trap
 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 **s td** :

TD - time domain; number of raw data points

OUTPUT PARAMETERS

can be viewed with **dpp** or by typing **s ft_mod, s 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 **s bytordp**:

BYTORDP - data storage order

INPUT FILES

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

fid - raw data (input of **trf**)

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

1r, 1i - processed 1D data (input of **trfp**)

proc - processing parameters

OUTPUT FILES

<dir>/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

`ftf, ft, bc, em, pk`

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

STSI - strip size (input if DATMOD = proc)

INPUT FILES

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

fid - raw data (input if DATMOD = raw)

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

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

proc - processing parameters

OUTPUT FILES

<dir>/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

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

fid - raw data (input if DATMOD = raw)

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

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

proc - processing parameters

OUTPUT FILES

<dir>/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 TOPSPIN 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.

abs2, abst2, absd2, absot2, bas

NAME

abs2 - automatic baseline correction in the F2 dimension (rows)
 abst2 - automatic selective baseline correction in the F2 dimension (rows)
 absd2 - automatic baseline correction in F2 with a different algorithm
 absot2 - automatic selective baseline correction in F2 with a different algorithm
 bas - open the baseline correction dialog box

DESCRIPTION

Baseline correction commands can be started from the command line, by entering **abs2**, **abst2** etc. or from the baseline dialog box. The latter is opened with the command **bas**:

Baseline correction - abs

Options

- ☒ Auto-correct baseline using polynomial
- ☐ Auto-correct baseline, shift correction region
- ☐ Auto-correct baseline, alternate algorithm
- ☐ Auto-correct baseline, shift correction region, altern. algo.
- ☐ Correct baseline using correction result from 1D row/column

Required parameters (F2 and F1)

Apply to axis: F2, F1 ☒ ☐

Degree of polynomial ABSG (0.5) =	5	5
Left limit for correction region ABSF1 [ppm] =	1000	1000
Right limit for correction region ABSF2 [ppm] =	-1000	-1000
Left limit of correction region, last row/col SIGF1 [ppm] =	0	0
Right limit of correction region, last row/col SIGF2 [ppm] =	0	0

OK Cancel Help

This dialog box offers several options, each of which selects a certain command for execution. The command further depends on the selected dimension. Here

we describe the commands for the F2 dimension.

F2 Auto-correct baseline using polynomial

This option selects the command **abs2** for execution. It 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.

F2 Auto-correct baseline, shift correction region

This option selects the command **abst2** for execution. It performs an automatic selective baseline correction in the F2 dimension. This means it corrects 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.

F2 Auto-correct baseline, alternate algorithm

This option selects the command **absd2** for execution. It works like **abs2**, except that it uses a different algorithm¹. 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**.

F2 Auto-correct baseline, shift correction region, alternate algorithm

This option selects the command **absot2** for execution. It works like **abst2**, except that it has a different algorithm which applies a larger correction.

If you run a command like **abs2** from the command line, you have to make sure

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

that the required parameters are already set. Click the *Procvars* tab or enter **edp** to do that.

The **bas** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **bas** dialog box, with **edp** or by typing **absg**, **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

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

2rr - real processed 2D data

proc - F2 processing parameters

OUTPUT FILES

<dir>/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

ABS2

ABST2

ABSD2

ABSOT2

SEE ALSO

abs1, abst1, absd1, absot1

abs1, abst1, absd1, absot1, bas

NAME

abs1 - automatic baseline correction in the F1 dimension (columns)
 abst1 - automatic selective baseline correction in the F1 dimension (columns)
 absd1 - automatic baseline correction in F1 with a different algorithm
 absot1 - automatic selective baseline correction in F1 with a different algorithm
 bas - open the baseline correction dialog box

DESCRIPTION

Baseline correction can be started from the command line, with **abst2**, **abst1** etc., or from the baseline dialog box. The latter is opened with the command **bas**

Baseline correction - abs1

Options

- ☒ Auto-correct baseline using polynomial
- ☐ Auto-correct baseline, shift correction region
- ☐ Auto-correct baseline, alternate algorithm
- ☐ Auto-correct baseline, shift correction region, altern. algo.
- ☐ Correct baseline using correction result from 1D row/column

Required parameters (F2 and F1)

	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Apply to axis: F2, F1		
Degree of polynomial ABSG (0..5) =	5	5
Left limit for correction region ABSF1 [ppm] =	1000	1000
Right limit for correction region ABSF2 [ppm] =	-1000	-1000
Left limit of correction region, last row/col SIGF1 [ppm] =	0	0
Right limit of correction region, last row/col SIGF2 [ppm] =	0	0

OK Cancel Help

This dialog box offers several options, each of which selects a certain command for execution. The command further depends on the selected dimension. Here we describe the commands for the F1 dimension.

F1 Auto-correct baseline using polynomial

This option selects the command **abs1** for execution. It 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**.

F1 Auto-correct baseline, shift correction region

This option selects the command **abst1** for execution. It 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**.

F1 Auto-correct baseline, alternate algorithm

This option selects the command **absd1** for execution. It works like **abs1**, except that it uses a different algorithm¹. 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**.

F1 Auto-correct baseline, shift correction region, alternate algorithm

This option selects the command **absot1** for execution. It works like **abst1**, except that it has a different algorithm which applies a larger correction.

If you run a command like **abs1** from the command line, you have to make sure that the required parameters are already set. Click the **Procpars** tab or enter **edp** to do that.

The **bas** command can be used on 1D, 2D or 3D data. It recognizes the data di-

mensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **bas** dialog box, 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

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

2rr - real processed 2D data

proc2 - F1 processing parameters

OUTPUT FILES

<dir>/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

ABS1

ABST1

ABSD1

ABSOT1

SEE ALSO

abs2, abst2, absd2, absot2

add2d, adsu

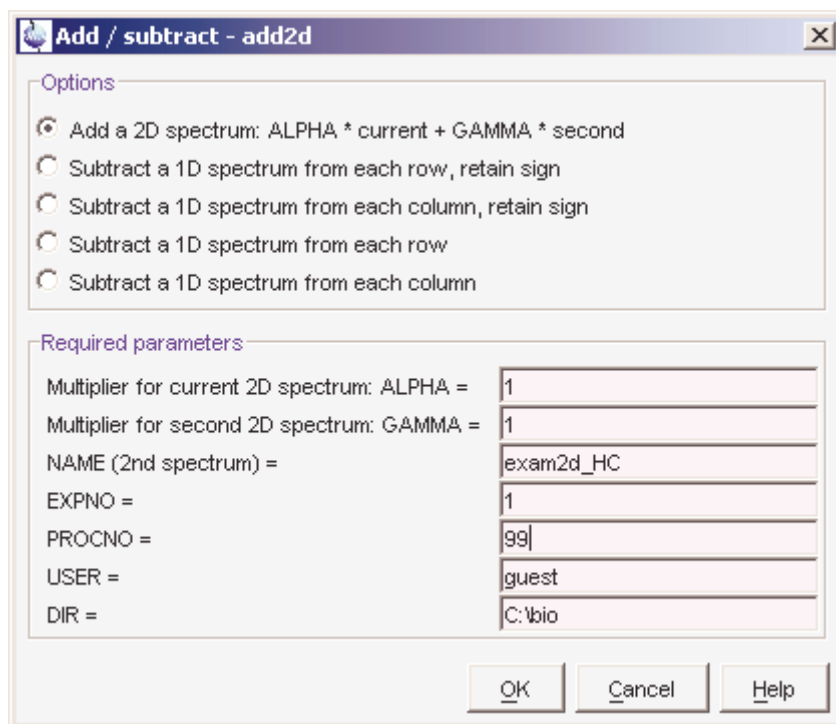
NAME

add2d - add or subtract two 2D datasets

adsu - open the add/subtract dialog box

DESCRIPTION

Add commands can be started from the command line or from the add/subtract dialog box. The latter is opened with the command **adsu**



This dialog box offers several options, each of which selects a certain command for execution.

Add a 2D spectrum

This option selects the command **add2d** for execution. It adds the second

dataset to the current 2D dataset. **add2d** performs the following operation:

$$\text{current} = \text{ALPHA} * \text{current} + \text{GAMMA} * \text{second}$$

where ALPHA and GAMMA are processing parameters. Both real and imaginary data are added. The result overwrites the current processed data.

Caution: the two 2D datasets to be added must have equal sizes.

For ALPHA = 1 and GAMMA = -1, the spectra are subtracted.

If you run a command like **add2d** from the command line, you have to make sure that the required parameters are already set. Click the **Procpars** tab or enter **edp** to do that. If the second dataset has not been defined yet, **add2d** opens the add/subtract (**adsu**) dialog box.

The **adsu** command can be used on 1D or 2D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **adsu** dialog box, with **edp** or by typing **alpha**, **gamma** etc.:

ALPHA - multiplication factor of the current spectrum

GAMMA - multiplication factor of the second spectrum

INPUT FILES

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

2rr, 2ir, 2ri, 2ii - processed data of the current dataset

proc - F2 processing parameters

<dir2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

2rr, 2ir, 2ri, 2ii - processed data of the second dataset

OUTPUT FILES

<dir>/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

addser

NAME

addser - add or subtract two 2D or two 3D raw datasets

DESCRIPTION

The command **add2ser** adds or subtracts two 2D or two 3D raw datasets. The input data are the current dataset and the second dataset. The latter one must be defined with the **edc2** command. **addser** performs the following operation:

$$\text{current} = \text{ALPHA} * \text{current} + \text{GAMMA} * \text{second}$$

where ALPHA and GAMMA are processing parameters. The result is stored in the current dataset, i.e. the current raw data are overwritten.

Note that the two input datasets to be added must have equal sizes.

For ALPHA = 1 and GAMMA = -1, the raw 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>/

ser - raw data of the current dataset

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

proc - F2 processing parameters

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

ser - raw data of the second dataset

OUTPUT FILES

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

`ser` - raw data

`audita.txt` - acquisition audit trail

USAGE IN AU PROGRAMS

`ADD SER`

SEE ALSO

`add`, `addfid`, `add2d`

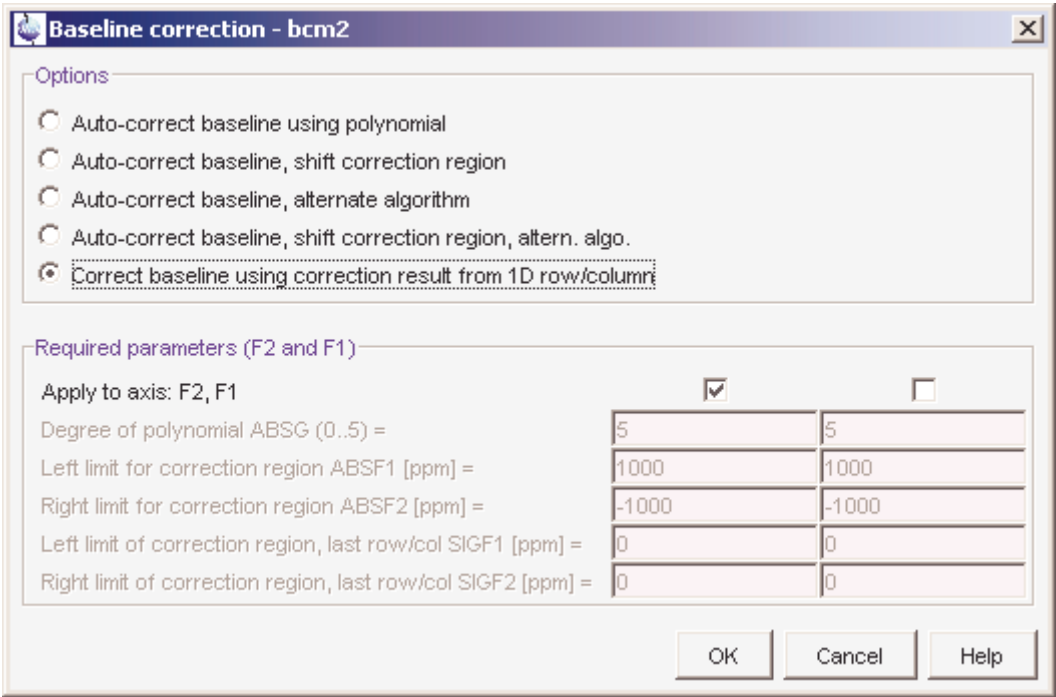
bcm2, bcm1

NAME

- bcm2 - user defined baseline correction in the F2 dimension (rows)
- bcm1 - user defined baseline correction in the F1 dimension (columns)

DESCRIPTION

Baseline correction commands can be started from the command line or from the baseline dialog box. The latter is opened with the command *bas*








This dialog box offers several options, each of which selects a certain command for execution.

F2 correct baseline, using correction result from 1D row

This option selects the command *bcm2* for execution. It performs a baseline correction in the F2 dimension by subtracting a polynomial, sine or exponen-

tial function. Before you can use **bcm2**, you must first do the following:

1. Read a row with **rsr** (TOPSPIN will switch to the 1D data window)
2. Click  or enter **.bas1** to switch to baseline mode.
3. Click ,  or  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). Click-hold button **A** and move the mouse to determine the zero order correction. Do the same with the buttons **B**, **C** for higher order corrections until the line matches the baseline of the spectrum.
5. Click  to leave baseline mode.
6. Select the 2D data window.

Then you can enter **bcm2** to perform the baseline correction.

F1 correct baseline, using correction result from 1D row

This option selects the command **bcm1** for execution. It 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.

bcm2* commands only works on the real data. After applying them, the imaginary data no longer match the real data and cannot be used for phase correction.

INPUT FILES

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

2rr - real processed 2D data

base_info - baseline correction coefficients

OUTPUT FILES

<dir>/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

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* → *Application Manuals*.

INPUT FILES

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

difflist - list of gradient amplitudes in Gauss/cm

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

2rr - 2D data processed in F2 only

dosy - DOSY processing parameters

OUTPUT FILES

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

2rr - 2D processed data

auditp.txt - processing audit trail

SEE ALSO

eddosy, dosy3d

f2disco, f1disco, proj

NAME

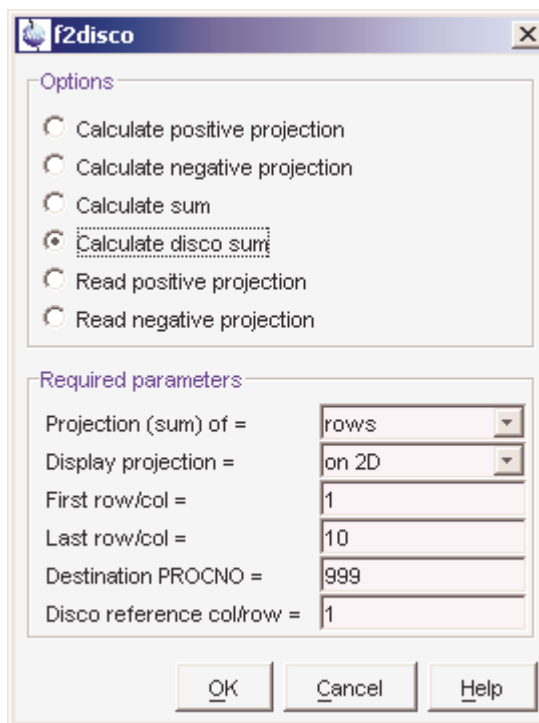
f2disco - calculate disco projection of the F2 dimension

f1disco - calculate disco projection of the F1 dimension

proj - open the projections dialog box

DESCRIPTION

The disco projection commands open the projections dialog box selecting the corresponding command.



This dialog box has several options, each of which selects a certain command for execution.

Calculate disco sum (of rows)

This option selects the command **f2disco** for execution. Like **f2sum**, it 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.

Calculate disco sum (of columns)

This option selects the command **f1disco** for execution. It 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

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

2rr, 2ir, 2ri, 2ii - processed data

OUTPUT FILES

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

1r, 1i- 1D spectrum containing the F1 disco projection

auditp.txt - processing audit trail

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, f2sum, rhpp

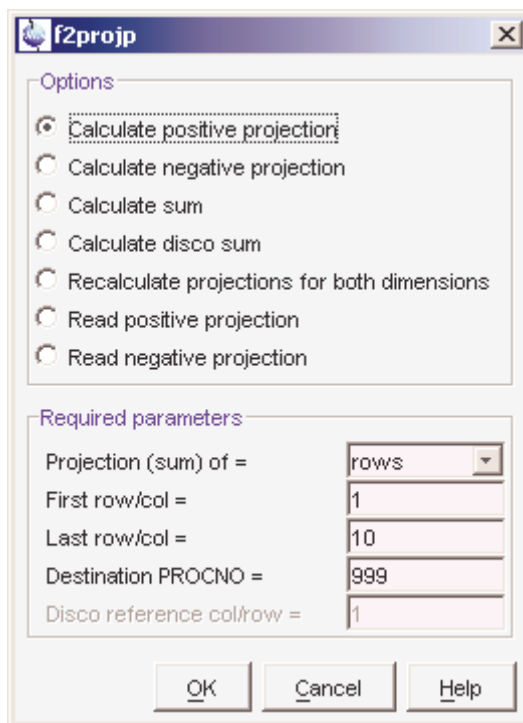
f2projn, f2projp, f1projn, f1projp, proj

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
proj - open the projections dialog box

DESCRIPTION

The projection commands open the projections dialog box selecting the corresponding command.



This dialog box has several options, each of which selects a certain command for execution.

Calculate positive projection (of rows)

This option selects the command **f2projp** for execution. It calculates the partial 1D projection of the 2D dataset in the F2 dimension

Calculate positive projection (of columns)

This option selects the command **f1projp** for execution. It calculates the partial 1D projection of the 2D dataset in the F2 dimension

Calculate negative projection (of rows)

This option selects the command **f2projn** for execution. It calculates the partial 1D projection of the 2D dataset in the F2 dimension

Calculate negative projection (of columns)

This option selects the command **f1projn** for execution. It calculates the partial 1D projection of the 2D dataset in the F2 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. Partial means that only a specified range of rows (or columns) is 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.

A special case is the command **f1projp** or **f1projn** on a hypercomplex 2D dataset ($MC2 \neq QF$) that has been processed in F2 only. Suppose you would perform the following command sequence:

xf2 - to process the data in F2 only

s si - to check the F1 size of the 2D data → click *Cancel*

s mc2 - to check status MC2 ($\neq QF$) → click *Cancel*

f1projp - to store the F1 projection in ~TEMP and change to that dataset

s si - to check the size of the resulting 1D dataset → click *Cancel*

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** :

s mc2 → click ***QF***

Then, the size of the 1D data equals the F1 size of the 2D data.

INPUT FILES

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

2rr - processed data

OUTPUT FILES

<dir>/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

<dir>/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:

<dir>/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, f2sum, rhpp

f2sum, f1sum, proj

NAME

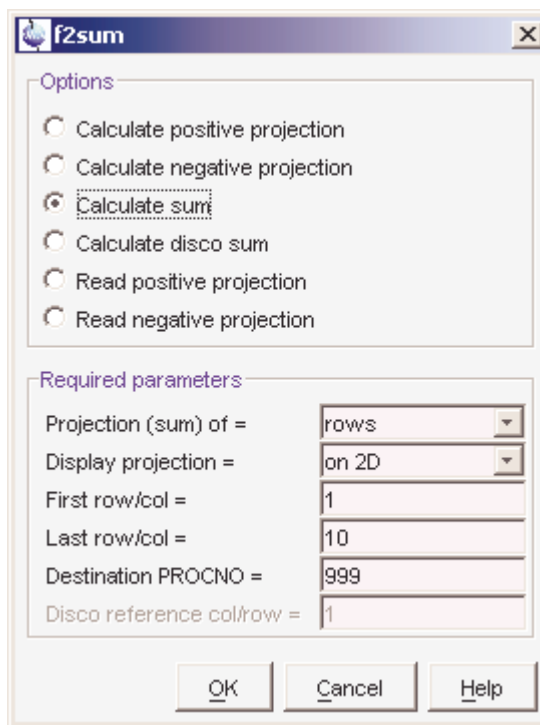
f2sum - calculates the sum of a range of rows (F2)

f1sum - calculates the sum of a range of columns (F1)

proj - open the projections dialog box

DESCRIPTION

The projection sum commands open the projections dialog box selecting the corresponding command.



This dialog box has several options, each of which selects a certain command for execution.

Calculate sum (of rows)

This option selects the command ***f2sum*** for execution. It calculates the sum of all rows within a region specified by the parameters.

Calculate sum (of columns)

This option selects the command ***f1sum*** for execution. It calculates the sum of all columns within a region specified by the parameters.

INPUT FILES

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

2rr, 2ir, 2ri, 2ii - processed data

OUTPUT FILES

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

1r, 1i- 1D spectrum containing the sum

auditp.txt - processing audit trail

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

f2projn, f2disco, rhpp

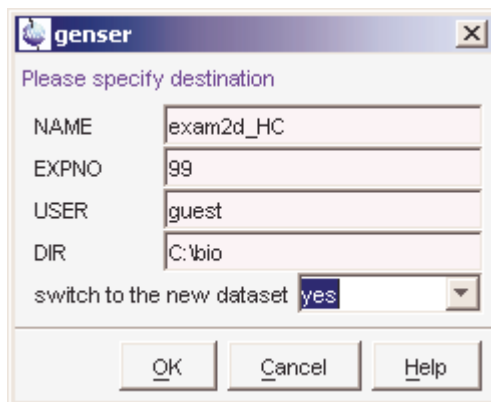
genser

NAME

genser - generate pseudo-raw 2D data

DESCRIPTION

The command **genser** generates pseudo-raw data from processed 2D data. When opened without arguments, genser opens the following dialog box:



Here, you specify the output dataset and click **OK** to actually execute the command. **genser** 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 arguments and can be used as follows:

1. **genser**

opens a dialog box where you can specify the output data.

2. *genser* <expno>

stores the output under the specified *expno*.

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

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

2rr, 2ir, 2ri, 2ii - processed time domain data

OUTPUT FILES

<dir>/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

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

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

2rr, 2ir, 2ri, 2ii - processed data

OUTPUT FILES

<dir>/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, proj

NAME

rhpp - calculate horizontal (F2) positive projection
rhnp - calculate horizontal (F2) negative projection
rvpp - calculate vertical (F1) positive projection
rvnp - calculate vertical (F1) negative projection
proj - open the projections dialog box

SYNTAX

r*p [<procno> [n]]

DESCRIPTION

The projection commands can be started from the command line or from the projection dialog box selecting the corresponding command.

This dialog box has several options, each of which selects a certain command for execution.

Read positive projection (on rows)

This option selects the command **rhpp** for execution. It calculates the full positive projection of a 2D spectrum in the F2 dimension and stores it as a 1D dataset.

Read positive projection (on columns)

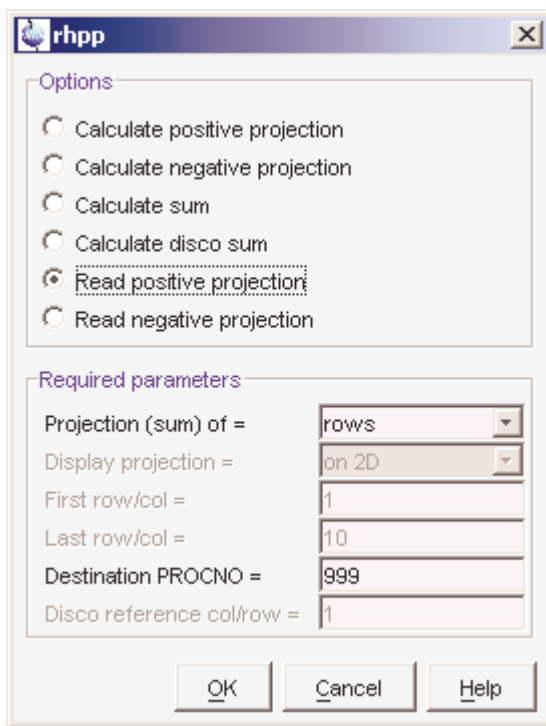
This option selects the command **rvpp** for execution. It calculates the full positive projection of a 2D spectrum in the F1 dimension and stores it as a 1D dataset.

Read negative projection (on rows)

This option selects the command **rhnp** for execution. It calculates the full negative projection of a 2D spectrum in the F2 dimension and stores it as a 1D dataset..

Read negative projection (on columns)

This option selects the command **rvnp** for execution. It calculates the full negative projection of a 2D spectrum in the F1 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.

r*p commands only take the projection of the first quadrant data (file 2rr) and store it as real 1D data (file 1r)

r*p commands can be started from the command line. They open a dialog box like:

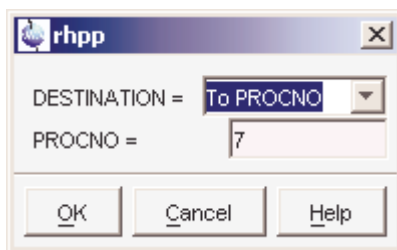
The required arguments can also be specified on the command line.

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 three other **r*p** command have the same syntax.

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

s si - to check the F1 size of the 2D data → click **Cancel**

s mc2 - to check status MC2 (≠ QF) → click **Cancel**

rvpp - to store the F1 projection in ~TEMP and change to that dataset

s si - to check the size of the resulting 1D dataset → click **Cancel**

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** :

s mc2 → click **QF**

Then, the size of the 1D data equals the F1 size of the 2D data.

INPUT FILES

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

2rr - real processed 2D data

OUTPUT FILES

<dir>/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:

`<dir>/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

`f2projn`, `f2sum`, `f2disco`

rsc

NAME

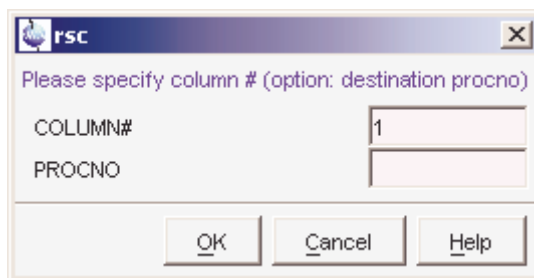
rsc - read a column from a 2D spectrum and store it as a 1D spectrum

SYNTAX

rsc [<column> [<procno>] [n]]

DESCRIPTION

The command **rsc** reads a column from a 2D spectrum and stores it as a 1D spectrum. When entered on a 2D dataset without arguments, **rsc** opens a dialog box where you can specify the column number and the procno of the output data.



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 **s si**. The PROCNO can be any number other than the current PROCNO. If the PROCNO field is left empty, the output dataset is stored under data name ~TEMP.

When entered on a 2D dataset, **rsc** takes up to three arguments and can be used as follows:

rsc

opens the above dialog box

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

opens the above dialog box

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 name¹, 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.

rsr can also be started from the dialog box that is opened with the command ***slice***.

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

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.

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:

s fnmode - check the FnMODE value (*States*) → click **Cancel**

xf2 - to Fourier transform in F2 only

s mc2 - check the MC2 value (*States*) → click **Cancel**

rsc 17 - read column 17 to ~TEMP and switch to that dataset

s 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:

s fnmode - check the FnMODE value (*States-TPPI*) → click **Cancel**

xf2 - to Fourier transform in F2 only

s 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:

s fnmode - check the FnMODE value (*QF*) → click **Cancel**

xf2 - to Fourier transform in F2 only

s mc2 - check the MC2 value (*QF*) → click **Cancel**

rsc 17 - to read column 17 to ~TEMP and switch to that dataset

s si - check the size of the 1D dataset → click **Cancel**

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 2*i*) exist, 1D imaginary (file 1*i*) 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.

INPUT FILES

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

2rr, 2ir, 2ri, 2ii - 2D processed data

OUTPUT FILES

If no output procno is specified:

<dir>/data/<user>/nmr/~TEMP/1/pdata/1/

1r, 1i - 1D spectrum

used_from - data path of the source 2D data and the column no.

auditp.txt - processing audit trail

If the output procno is specified:

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

1r, 1i - 1D spectrum

used_from - data path 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, wserp, slice

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. It opens a dialog box where you can specify the FID number and the expno of the output data.



For 2D data, 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 **s td**.

rser is normally entered on the 2D dataset. It then takes up to 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

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

opens the above dialog box where you can specify the row number and the procno of 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.

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).

rser can also be started from the dialog box that is opened with the command **slice**.

INPUT FILES

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

ser - 2D or 3D raw data

OUTPUT FILES

If the output expno is specified:

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

fid - 1D FID

audita.txt - acquisition audit trail

<dir>/data/<user>/nmr/<name>/<expno>/pdata/1/

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).

used_from - data path of the source 2D data and the row no.

If no output expno is specified:

<dir>/data/<user>/nmr/~TEMP/1/

fid - 1D FID

<dir>/data/<user>/nmr/~TEMP/1/pdata/1

used_from - data path 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, slice

rsr

NAME

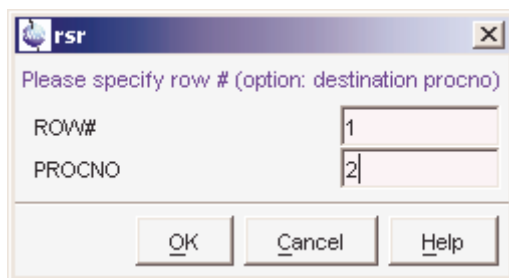
rsr - read a row from a 2D spectrum and store it as a 1D spectrum

SYNTAX

rsr [<row> [<procno>] [n]]

DESCRIPTION

The command **rsr** reads a row from a 2D spectrum and stores it as a 1D spectrum. When entered on a 2D dataset without arguments, **rsr** opens a dialog box where you can specify the row number and the procno of the output data.



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 **s si**. The PROCNO can be any number other than the current PROCNO. If the PROCNO field is left empty, the output dataset is stored under data name ~TEMP.

When entered on a 2D dataset, **rsr** takes up to three arguments and can be used as follows:

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

opens the dialog box where you can specify the row and procno of the 2D data

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.

rsr can also be started from the dialog box that is opened with the command **slice**.

INPUT FILES

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

2rr, 2ir, 2ri, 2ii - 2D processed data

OUTPUT FILES

If no procno is specified:

<dir>/data/<user>/nmr/~TEMP/1/pdata/1/

1r, 1i - 1D spectrum

used_from - data path of the source 2D data and the row no.

auditp.txt - processing audit trail

If the output procno is specified:

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.

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

1r, 1i - 1D spectrum

used_from - data path 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

SEE ALSO

rsc, wsr, wsc, rser, rser2d, wser, wserp, slice

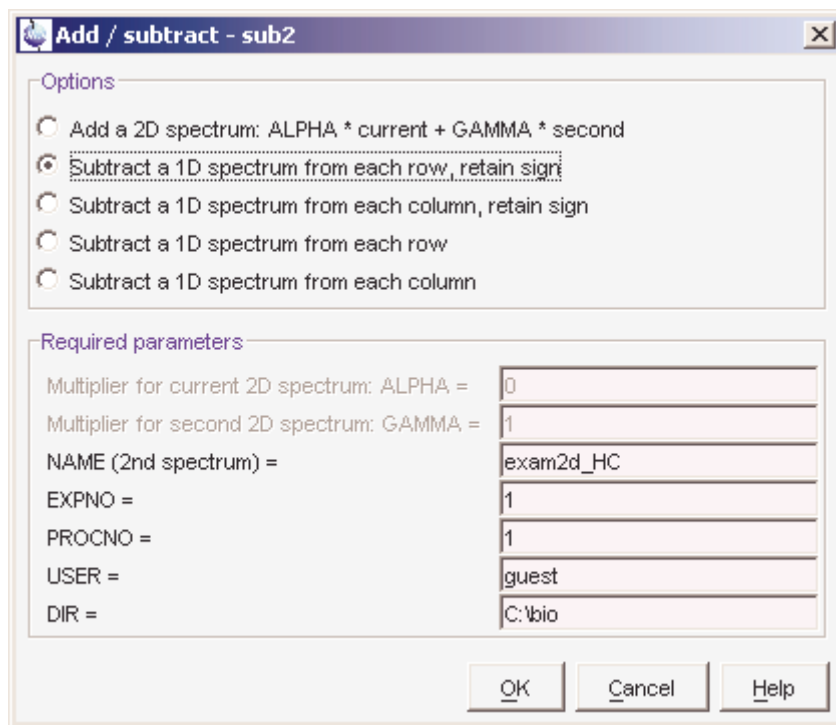
sub2, sub1, sub1d12, sub1d1, adsu

NAME

sub2 - subtract a 1D spectrum from each row of a 2D spectrum, retain sign
sub1 - subtract a 1D spectrum from each column of a 2D spectrum, retain sign
sub1d2 - subtract a 1D spectrum from each row of a 2D spectrum
sub1d1 - subtract a 1D spectrum from each column of a 2D spectrum
adsu - open the add/subtract dialog box

DESCRIPTION

Subtracting a 1D data from a 2D data can be started from the command line or from the add/subtract dialog box. The latter is opened with the command **adsu**.



This dialog box offers several options, each of which selects a certain command for execution.

Subtract a 1D spectrum from each row, retain sign

This option selects the command **sub2** for execution. It subtracts a 1D dataset from each row of the current 2D spectrum. 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.

Subtract a 1D spectrum from each column, retain sign

This option selects the command **sub1** for execution. It works like **sub2** except that it subtracts the 1D second dataset from each column of the current 2D spectrum.

Subtract a 1D spectrum from each row

This option selects the command **sub1d2** for execution. It subtracts a 1D dataset from each row of the current 2D spectrum. Unlike **sub2**, it does not compare intensities.

Subtract a 1D spectrum from each column

This option selects the command **sub1d1** for execution. It subtracts a 1D dataset from each column of the current 2D spectrum. Unlike **sub1**, it does not compare intensities.

The **sub*** commands 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.

If the second dataset has not been defined yet, the **sub*** commands open the add/subtract (**adsu**) dialog box.

The **adsu** command can be used on 1D or 2D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT FILES

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

2rr - real processed 2D data

<dir2>/data/<user2>/nmr/<name2>/<expno2>/pdata/<procno2>/

1r - real processed 1D data

OUTPUT FILES

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

2rr - real processed 2D data

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

SUB2

SUB1

SUB1D2

SUB1D1

SEE ALSO

add2d

sym, syma, symj, symt

NAME

sym - symmetrize a 2D spectrum about the diagonal

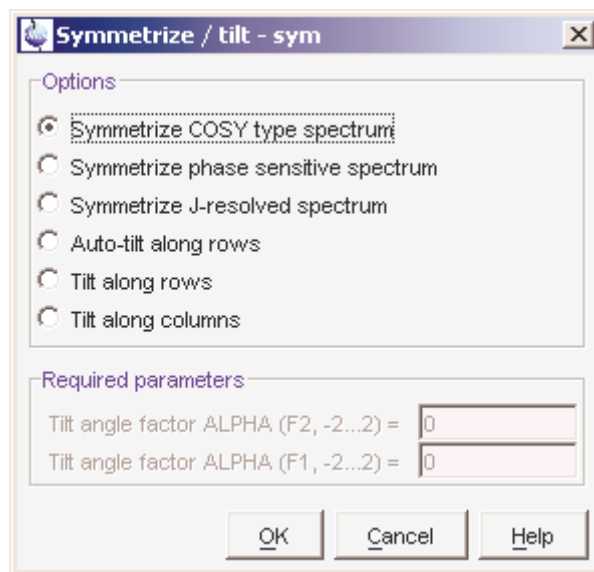
syma - symmetrize a 2D spectrum about the diagonal, leaving the sign the same

symj - symmetrize a 2D spectrum about the horizontal line through the middle

symt - open the symmetrization and tilt dialog box

DESCRIPTION

All **sym*** commands open the symmetrize/tilt dialog box:



This dialog box offers several options, each of which selects a certain command for execution.

Symmetrize COSY type spectrum

This option selects the command **sym** for execution. It 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. In fact, **sym** first opens the symmetrization dialog box with the **sym** command selected. 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 is typically used on magnitude cosy spectra.

Symmetrize phase sensitive spectrum

This option selects the command **syma** for execution. It works like **sym** except that 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 is typically used on phase sensitive cosy spectra.

Symmetrize J-resolved spectrum

This option selects the command **symj** for execution. It 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 is typically used on J-resolved spectra which have been tilted with the command **tilt**.

sym* commands only work on the real data. After using it, the imaginary data no longer match the real data and cannot be used for phase correction.

When executed from the command line, the command **sym**, **syma** and **symj** select the corresponding option in the dialog box. This means, you can just click **OK** or hit **Enter** to start the command. In contrast, **symt** selects the last used symmetrization command.

OUTPUT PARAMETERS

can be viewed with **dpp** or by typing **s symm** :

SYMM - type of symmetrization (*no*, *sym*, *syma* or *symj*) done

INPUT FILES

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

2rr - real processed 2D data

OUTPUT FILES

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

2rr - real processed 2D data

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

SYM

SYMA

SYMJ

SEE ALSO

tilt, ptilt, ptilt1

tilt, ptilt, ptilt1, symt

NAME

tilt - tilt a 2D spectrum

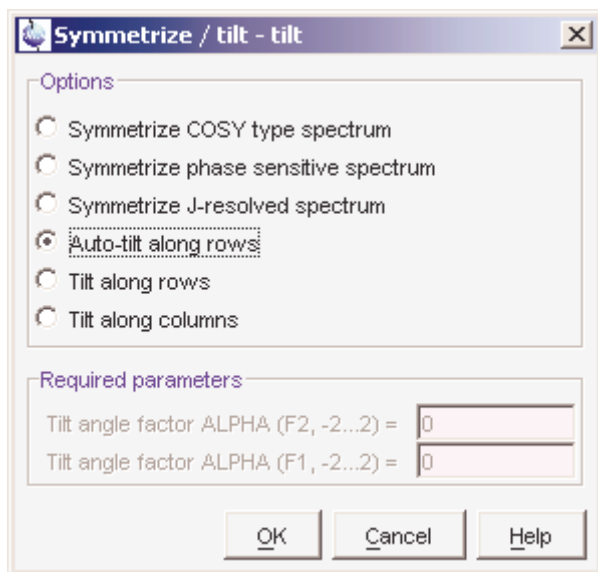
ptilt - tilt a 2D spectrum by shifting the data in the F2 dimension

ptilt1 - tilt a 2D spectrum by shifting the data in the F1 dimension

symt - open the symmetrize/tilt dialog box

DESCRIPTION

All ****tilt**** commands open the symmetrize/tilt dialog box: .



This dialog box offers several options, each of which selects a certain command for execution.

Auto-tilt along rows

This option selects the command ***tilt*** for execution. It tilts 2D spectrum, shifting each row of the 2D spectrum by the value:

$$n = \text{tiltfactor} * (\text{nsrow}/2 - \text{row})$$

The variables in this equation are defined as:

$$\text{tiltfactor} = (\text{SW_p1}/\text{SI1}) / (\text{SW_p2}/\text{SI2})$$

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.

Tilt along rows

This option selects the command **ptilt** for execution. It 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:

$$n = \text{tiltfactor} * (\text{nsrow}/2 - \text{row})$$

The variables in this equation are defined by:

$$\text{tiltfactor} = \text{ALPHA} * \text{SI2} / \text{SI1}$$

nsrow = total number of rows
row = the row number

where SI2 and SI1 are processing status parameter SI in F2 and F1, respectively.

Tilt along columns

This option selects the command **ptilt1** for execution. It 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:

$$n = \text{tiltfactor} * (\text{nscol}/2 - \text{row})$$

The variables in this equation are defined by:

$$\text{tiltfactor} = \text{ALPHA} * \text{SI1} / \text{SI2}$$

nscol = total number of columns

col = the column number

where SI2 and SI1 are 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***.

When executed from the command line, the command ***tilt***, ***ptilt*** and ***ptilt1*** select the corresponding option in the dialog box. This means, you can just click ***OK*** or hit ***Enter*** to start the command. In contrast, ***symt*** selects the last used tilt command.

INPUT PARAMETERS

set from the ***symt*** dialog box, with ***edp*** or by typing ***alpha*** :

ALPHA - tilt factor (used by ***ptilt*** and ***ptilt1***)

set by initial processing command, e.g. ***xfb***, can be viewed with ***dpp***:

SW_p - spectral width of the processed data (used by ***tilt***)

SI - size of the processed data

OUTPUT PARAMETERS

can be viewed with ***dpp*** or by typing ***s tilt*** (currently wrong):

TILT - shows whether ***tilt***, ***ptilt*** or ***ptilt1*** was done (true or false)

INPUT FILES

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

2rr - real processed 2D data

OUTPUT FILES

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

2rr - real processed 2D data

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

TILT

PTILT

PTILT1

SEE ALSO

sym, syma, symj

WSC

NAME

wsc - write spectrum column; replace a 2D column by a 1D spectrum

SYNTAX

wsc [<row> [<procno>]]

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> <dir>**

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 data path are specified, e.g. the expno and name, the remaining parts are the same as in the current 1D data path. In AU programs, ***wsc*** must always have 6 arguments (see USAGE IN AU PROGRAMS below).

rser can also be started from the dialog box that is opened with the command ***slice***.

INPUT FILES

<dir>/data/<user>/nmr/~TEMP/1/pdata/1

1r, 1i - 1D processed data

used_from - data path of the 2D data (input of ***wsc*** on a 1D dataset)

or

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

1r, 1i - 1D processed data

used_from - data path of the 2D data (input of ***wsc*** on a 1D dataset)

OUTPUT FILES

<dir>/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, dir)

SEE ALSO

rsc, wsr, rsr, wser, wserp, rser, rser2d, slice

wser

NAME

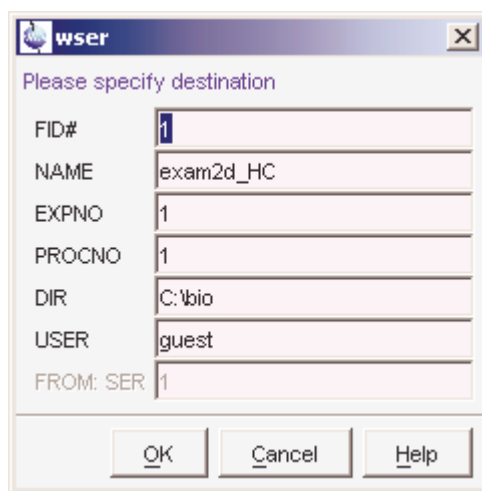
wser - replace a row of 2D raw data by 1D raw data

SYNTAX

wser [<row> [<expno>]]

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. When entered on a 1D dataset, **wser** opens the following dialog box :



Here, you can enter the FID number to be replaced and the destination data path.

Usage of **wser** with arguments on the destination 1D dataset:

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.

Usage of **wser** with arguments 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.

INPUT FILES

<dir>/data/<user>/nmr/~TEMP/1/

fid - 1D raw data

<dir>/data/<user>/nmr/~TEMP/1/pdata/1

used_from - data path of the 2D data (input of **wser** on a 1D dataset)

or

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

fid - 1D raw data

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

used_from - data path of the 2D data (input of **wser** on a 1D dataset)

wser can also be started from the dialog box that is opened with the command **slice**.

OUTPUT FILES

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

ser - raw 2D data

audita.txt - acquisition audit trail

USAGE IN AU PROGRAMS

WSER(row, name, expno, procno, dir, user)

Note that the order of the arguments in AU programs is different from the order on the command line.

SEE ALSO

wserp, rser, rser2d, wsr, wsc, rsr, rsc, slice

wserp

NAME

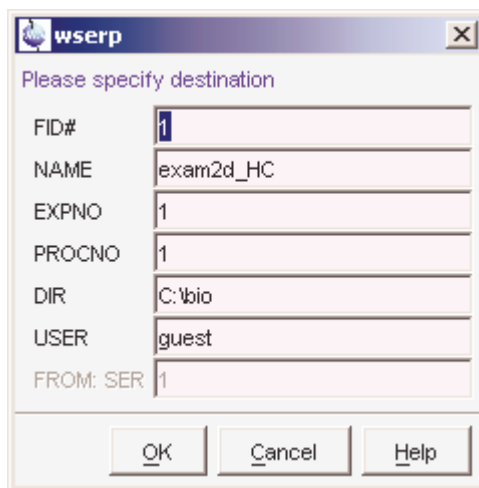
wserp - replace a row of 2D raw data by 1D processed data

SYNTAX

wserp [<row> [<expno>]]

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. When entered on a 1D dataset, **wser** opens the following dialog box:



The screenshot shows a Windows-style dialog box titled "wserp". The main text inside the dialog is "Please specify destination". Below this text are several input fields, each with a label to its left: "FID#" with the value "1", "NAME" with "exam2d_HC", "EXPNO" with "1", "PROCNO" with "1", "DIR" with "C:\bio", "USER" with "guest", and "FROM: SER" with "1". At the bottom of the dialog are three buttons: "OK", "Cancel", and "Help".

Here, you can enter the FID number to be replaced and the destination data path.

Usage of **wserp** with arguments on the destination 1D dataset:

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>

the specified row of the 2D raw data under the specified expno is replaced by the current 1D processed data. The 2D dataset *name*, *user* and *dir* are the same as in the dataset as the current 1D data were extracted from.

wser can also be started from the dialog box that is opened with the command **slice**.

INPUT FILES

<dir>/data/<user>/nmr/~TEMP/1/pdata/1/

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

used_from - data path of the 2D data (input of **wserp** on a 1D dataset)

or

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

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

used_from - data path of the 2D data (input of **wserp** on a 1D dataset)

OUTPUT FILES

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

ser - raw 2D data

audita.txt - acquisition audit trail

USAGE IN AU PROGRAMS

WSERP(row, name, expno, procno, dir, user)

Note that the order of the arguments in AU programs is different from the order on the command line.

SEE ALSO

wser, rser, rser2d, wsr, wsc, rsr, rsc, slice

WSR

NAME

wsr - write spectrum row; replace a row of a 2D spectrum by a 1D spectrum

SYNTAX

wsr [<row> [<procno>]]

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 name¹, 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.

wsr can also be started from the dialog box that is opened with the command ***slice***.

INPUT FILES

<dir>/data/<user>/nmr/~TEMP/1/pdata/1

1r, 1i - 1D processed data

used_from - data path of the 2D data (input of ***wsr*** on a 1D dataset)

or

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

1r, 1i - 1D processed data

used_from - data path of the 2D data (input of ***wsr*** on a 1D dataset)

OUTPUT FILES

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

2rr, 2ir - processed 2D data

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

WSR(row, procno, expno, name, user, dir)

SEE ALSO

wsc, rsr, rsc, wser, wserp, rser, rser2d, slice

xf1

NAME

xf1 - process data in the F1 dimension

DESCRIPTION

The command **xf1** processes a 2D dataset in the F1 dimension. It can be started from the command line or from the Fourier transform dialog box. The latter is opened with the command **ftf**.

xf1 Fourier 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**).

The **ftf** command can be used on 1D or 2D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

F2 and F1 parameters

set by **xf2**, can be viewed with **dpp** or by typing **s si**, **s 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

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 from the **ftf** dialog box, 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 for WDW = trap

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 **s mc2** :

MC2 - Fourier transform mode (input of **xf1** on processed data)

set by the acquisition, can be viewed with **dpa** or by typing **s fnmode**:

FnMODE - Acquisition mode (input of **xf1** on raw data)

OUTPUT PARAMETERS

F1 parameters

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

FT_mod - Fourier transform mode

FTSIZE - Fourier transform size

F2 parameters

can be viewed with **dpp** or by typing **s ymax_p, s 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

<dir>/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

<dir>/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

<dir>/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)

procs - F2 processing status parameters

proc2s - F1 processing status parameters

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

XF1

SEE ALSO

xf2, xfb, xtrf, xtrfp1

xfbm, xf2m, xf1m, ph

NAME

xfbm - calculate the magnitude spectrum in the F2 and F1 dimension

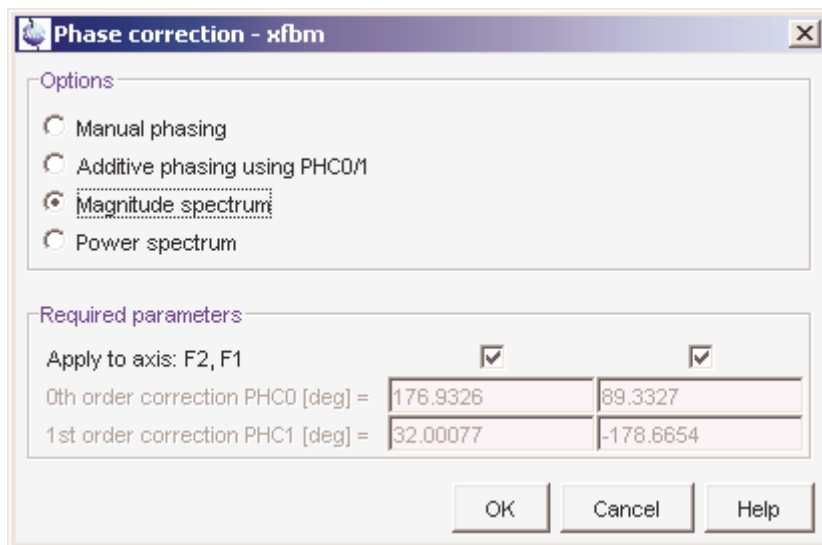
xf2m - calculate the magnitude spectrum in the F2 dimension

xf1m - calculate the magnitude spectrum in the F1 dimension

ph - open the phase correction dialog box

DESCRIPTION

The magnitude spectrum commands can be started from the command line or from the phase correction dialog box. The latter is started with the command **ph**:



This dialog box offers several options, each of which selects a certain command for execution.

Magnitude spectrum in F2

This option selects the command **xf2m** for execution. It calculates the real and F2-imaginary data according to:

Magnitude spectrum in F1

$$rr = \sqrt{rr^2 + ir^2}$$

$$ri = \sqrt{ri^2 + ii^2}$$

This option selects the command ***xf1m*** for execution. It calculates the real and F1-imaginary data according to according to:

$$rr = \sqrt{rr^2 + ri^2}$$

$$ir = \sqrt{ir^2 + ii^2}$$

Magnitude spectrum in F12 and F1

This option selects the command ***xfbm*** for execution. It calculates the real and F1/F2-imaginary data according to according to:

$$rr = \sqrt{rr^2 + ir^2 + ri^2 + ii^2}$$

where:

rr = real data (2rr file)

ir = F2-imaginary data (2ir file)

ri = F1-imaginary data (2ri file)

ii = F2/F1-imaginary data (2ii file)

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.

The **ph** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT FILES

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

2rr, 2ir, 2ri, 2ii - processed 2D data

OUTPUT FILES

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

2rr, 2ir, 2ri, 2ii - processed 2D data

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

XFBM

XF2M

XF1M

SEE ALSO

xf2ps, xf1ps, xfbps

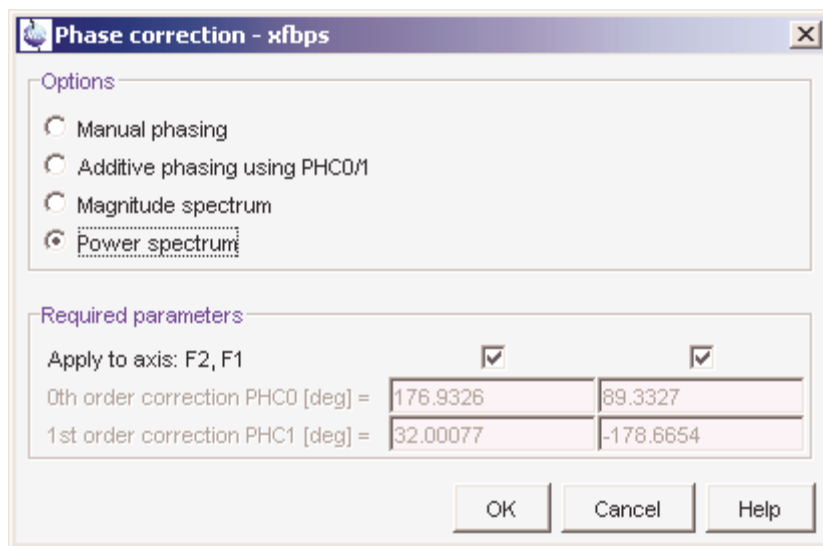
xfbps, xf2ps, xf1ps, ph

NAME

xfbps - calculate the power spectrum in the F2 and F1 dimension
xf2ps - calculate the power spectrum in the F2 dimension
xf1ps - calculate the power spectrum in the F1 dimension
ph - open the phase correction dialog box

DESCRIPTION

The commands **xf*ps** calculate the magnitude spectrum. They can be started from the command line or from the phase correction dialog box. The latter is started with the command **ph** : .



This dialog box offers several options, each of which selects a certain command for execution.

Power spectrum in F2

This option selects the command **xf2ps** for execution. It recalculates the real and F2-imaginary data according to:

$$rr = rr^2 + ir^2$$

$$ri = ri^2 + ii^2$$

Power spectrum in F1

This option selects the command ***xf1ps*** for execution. It recalculates the real and F1-imaginary data according to:

$$rr = rr^2 + ri^2$$

$$ir = ir^2 + ii^2$$

Power spectrum in F2 and F1

This option selects the command ***xf1ps*** for execution. It recalculates the real according to:

$$rr = rr^2 + ir^2 + ri^2 + ii^2$$

where:

rr = real data (2rr file)

ir = F2-imaginary data (2ir file)

ri = F1- imaginary data (2ri file)

ii = F2/F1-imaginary data (2ii file)

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.

The ***ph*** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT FILES

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

2rr, 2ir, 2ri, 2ii - processed 2D data

OUTPUT FILES

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

2rr, 2ir, 2ri, 2ii - processed 2D data

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

XFBPS

XF2PS

XF1PS

SEE ALSO

xfbm, xf2m, xf1m

xf2

NAME

xf2 - process data in the F2 dimension

DESCRIPTION

The command **xf2** processes a 2D dataset in the F2 dimension. It can be started from the command line or from the Fourier transform dialog box. The latter is opened with the command **ftf**.

xf2 Fourier 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 from the **ftf** dialog box, 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 **s td, s td:**

TD - time domain; number of raw data points

F2 parameters

set from the **ftf** dialog box, 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 for WDW = trap

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 **s 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 **s fnmode :**

FnMODE - Fourier transform mode

OUTPUT PARAMETERS

F2 and F1 parameters

can be viewed with **dpp** or by typing **s si, s tdeff** 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 **s ft_mod, s 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 **s bytordp**:

BYTORDP - byte order of the processed data

F1 parameters

set by the acquisition, can be viewed with **dpp** or by typing **s mc2** :

MC2 - Fourier transform mode

INPUT FILES

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

ser - raw data (input if 2rr does not exist or is Fourier transformed in F2)
 acqu - F2 acquisition status parameters
 acqu2s - F1 acquisition parameters

<dir>/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

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

2rr - first quadrant real processed data

2ir - second quadrant imaginary processed data (output if FnMODE \neq QF)

2ii - second quadrant imaginary processed data (output if FnMODE = QF)

procs - F2 processing status parameters

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

XF2

SEE ALSO

xf1, xfb, xtrf, xtrf2

$\mathbf{xfb, ftf}$

NAME

\mathbf{xfb} - process 2D data in the F2 and F1 dimension

\mathbf{ftf} - open the Fourier transform dialog box

DESCRIPTION

The command \mathbf{xfb} processes a 2D dataset or a plane of a 3D dataset. It can be started from the command line or from the Fourier transform dialog box. The latter is opened with the command \mathbf{ftf}

Options	
<input checked="" type="radio"/> Standard Fourier transform	
<input type="radio"/> Advanced Fourier transform	

Required parameters (F2 and F1)	
Enable transform for one/both dimensions	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
Size of real spectrum SI [pnts] =	256 256
Fid baseline correction mode [pnts] =	no no
Filter width for BC_mod=sfil/qfil [ppm] =	1 1
Correction offset for BC_mod=sfil/qfil COROFFS [Hz] =	0 0
# of fid data points to be used TDef =	0 0
Index of first output point of strip transform STSR =	0 0
Total # of output points of strip transform STSI =	0 0
Fid linear prediction (LP) mode ME_mod =	No LP No LP
# of LP coefficients NCOEF =	0 32
# of fid data points contributing to backward LP LPBIN =	0 0
# of fid data points to be predicted TDOFF =	0 0
Reverse spectrum REVERSE =	No No
Weighting factor for first fid point (A*X only) FCOR =	0.5 0.5
Apply 5th order phase correction (A*X only) PKNL =	Yes

OK Cancel Help

This dialog box offers two options both of which select the **xfb** command for execution, provided the F2 and F1 dimension are both enabled.

Standard Fourier transform

This option only allows you to set the parameter SI, the size of the real spectrum.

Advanced Fourier transform

This option allows you to set all Fourier transform related parameters.

xfb Fourier 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

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

transformed according to the acquisition status parameter FnMODE as shown in table 4.5. **xfb** does not evaluate the processing parameter 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 in Phase correction mode 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. little endian on Windows PCs. Note that TOPSPIN's predecessor XWIN-NMR on SGI UNIX workstations stores data in big endian. The byte order is stored in the processing status parameter BYTORDP which can be viewed with ***s 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 command ***xdim***. 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, the output procno and, if applicable, for the permission to overwrite existing data. Alternatively, you can enter this information as arguments on the command line, for example:

```
xfb s23 17 2 y
```

will read the F3-F2 plane number 17 and store it under procno 2, overwriting possibly existing data.

When executed on a dataset with 3D raw data but 2D processed data¹, ***xfb*** takes one argument:

```
xfb <plane>
```

1. Usually a result of a previous 2D processing command on that 3D dataset.

process the specified plane and store it under the current procno.

xfb same

process the same plane as the previous processing command and store it under the current procno. The ***same*** option is automatically used by the AU program macro XFB. When used on a regular 2D dataset (i.e. with 2D raw data), it has no effect.

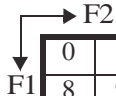
Normally, ***xfb*** stores the entire spectral region as determined by the spectral width. You can, however, do a so-called strip transform which means that only a certain region of the spectrum is stored. This can be done by setting the parameters STSR and STSI which represent the strip start and strip 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.

F1

F2

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



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

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.

FnMODE = QF

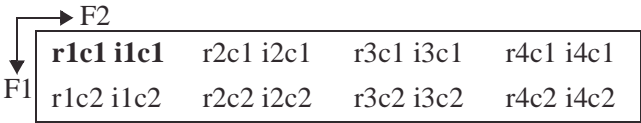
xfb performs complex (two-quadrant) processing. In F2 the data are acquired phase sensitive, in F1 non-phase sensitive. In the example below, the following parameter settings are used:

In F2: TD = 8, SI is 4
In F1: TD = 2, SI = 2

Furthermore, the following notation is used for individual data points:

rncm : point *n* of FID *m*. This point is real in F2 and complex in F1
incm : point *n* of FID *m*. This point is imaginary in F2 and complex in F1

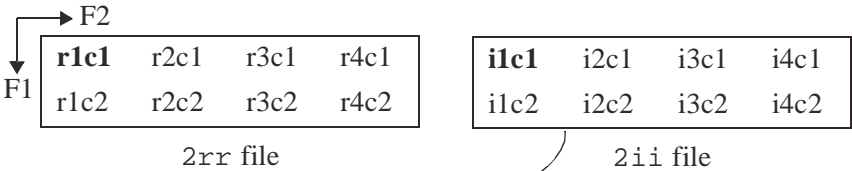
Input F2 processing
(raw data)



ser file

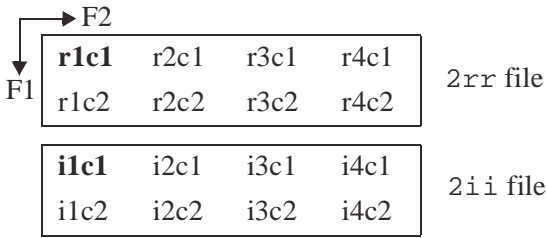
For F2 processing, **r1c1**, **i1c1** is the first complex input point, r2c1, i2c1 the second etc.

Output F2 processing = Input F1 processing

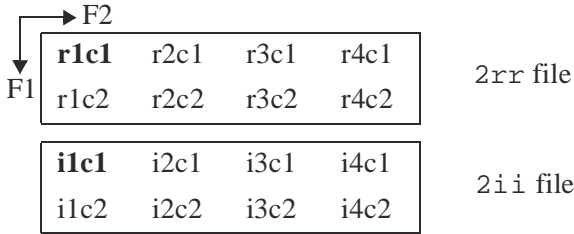


Below, the F1 input data are simply redisplayed in vertical order, with the first complex input point in bold.

Input F1 processing



Output F1 processing



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:

- **rnrm** : point *n* of FID *m*. This point is real in F2 and F1
- **inrm** : point *n* of FID *m*. This point is imaginary in F2 and real in F1
- **rnim** : point *n* of FID *m*. This point is real in F2 and imaginary in F1
- **inim** : point *n* of FID *m*. This point is imaginary in F2 and F1

Input F2 processing
(raw data)

F2

F1

r1r1	ilr1	r2r1	i2r1	r3r1	i3r1	r4r1	i4r1
r1i1	il i1	r2i1	i2i1	r3i1	i3i1	r4i1	i4i1
r1r2	ilr2	r2r2	i2r2	r3r2	i3r2	r4r2	i4r2
r1i2	il i2	r2i2	i2i2	r3i2	i3i2	r4i2	i4i2

ser file

For F2 processing, **r1r1**, **ilr1** 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

ilr1	i2r1	i3r1	i4r1
il i1	i2i1	i3i1	i4i1
ilr2	i2r2	i3r2	i4r2
il i2	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

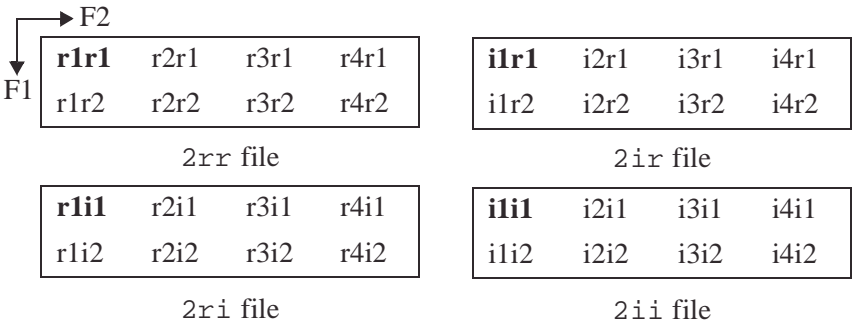
r1r1	r2r1	r3r1	r4r1
r1i1	r2i1	r3i1	r4i1
r1r2	r2r2	r3r2	r4r2
r1i2	r2i2	r3i2	r4i2

2rr file

ilr1	i2r1	i3r1	i4r1
il i1	i2i1	i3i1	i4i1
ilr2	i2r2	i3r2	i4r2
il i2	i2i2	i3i2	i4i2

2ir file

Output F1 processing



FnMODE = Echo-Antiecho

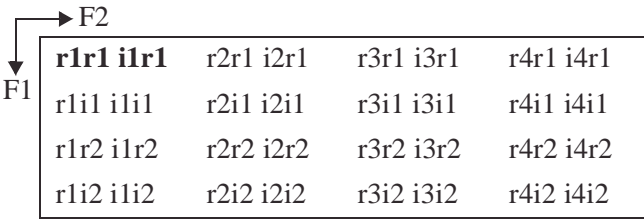
xfb performs hypercomplex (four-quadrant) processing. Both in F2 and F1, the data are acquired phase sensitive. In the example below, the following parameters settings are used:

In F2: TD = 8, SI is 4
In F1: TD = 4, SI = 2

Furthermore, the following notation is used for individual data points:

- **rnrm** : point *n* of FID *m*. This point is real in F2 and F1
- **inrm** : point *n* of FID *m*. This point is imaginary in F2 and real in F1
- **rnim** : point *n* of FID *m*. This point is real in F2 and imaginary in F1
- **inim** : point *n* of FID *m*. This point is imaginary in F2 and F1

Input F2 processing (raw data)

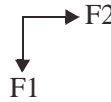


ser file

For F2 processing, **r1r1**, **i1r1** is the first hypercomplex input data point, r2r1,

i2r1 the second etc.

Output F2 processing = Input F1 processing



A diagram showing an arrow pointing from 'F1' to 'F2'. The 'F1' label is at the bottom left, and the 'F2' label is at the top left, with an arrow pointing from 'F1' to 'F2'.

-i1r1-i1i1	-i2r1-i2i1	-i3r1-i3i1	-i4r1-i4i1
-r1r1+r1i1	-r2r1+r2i1	-r3r1+r3i1	-r4r1+r4i1
-i1r2-i1i2	-i2r2-i2i2	-i3r2-i3i2	-i4r2-i4i2
-r1r2+r1i2	-r2r2+r2i2	-r3r2+r3i2	-r4r2+r4i2

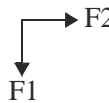
2rr file

r1r1+r1i1	r2r1+r2i1	r3r1+r3i1	r4r1+r4i1
-i1r1+i1i1	-i2r1+i2i1	-i3r1+i3i1	-i4r1+i4i1
r1r2+r1i2	r2r2+r2i2	r3r2+r3i2	r4r2+r4i2
-i1r2+i1i2	-i2r2+i2i2	-i3r2+i3i2	-i4r2+i4i2

2ir file

Below, the F1 input data are simply redisplayed, with the first F1 complex input points in bold.

Input F1 processing



A diagram showing an arrow pointing from 'F1' to 'F2'. The 'F1' label is at the bottom left, and the 'F2' label is at the top left, with an arrow pointing from 'F1' to 'F2'.

-i1r1-i1i1	-i2r1-i2i1	-i3r1-i3i1	-i4r1-i4i1
-r1r1+r1i1	-r2r1+r2i1	-r3r1+r3i1	-r4r1+r4i1
-i1r2-i1i2	-i2r2-i2i2	-i3r2-i3i2	-i4r2-i4i2
-r1r2+r1i2	-r2r2+r2i2	-r3r2+r3i2	-r4r2+r4i2

2rr file

r1r1+r1i1	r2r1+r2i1	r3r1+r3i1	r4r1+r4i1
-i1r1+i1i1	-i2r1+i2i1	-i3r1+i3i1	-i4r1+i4i1
r1r2+r1i2	r2r2+r2i2	r3r2+r3i2	r4r2+r4i2
-i1r2+i1i2	-i2r2+i2i2	-i3r2+i3i2	-i4r2+i4i2

2ir file

Output F1 processing

<div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; transform: rotate(-90deg);">F1</div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; transform: rotate(90deg);">F2</div> </div> </div>				
	-i1r1-i1i1	-i2r1-i2i1	-i3r1-i3i1	-i4r1-i4i1
	-i1r2-i1i2	-i2r2-i2i2	-i3r2-i3i2	-i4r2-i4i2
	2rr file			
	r1r1+r1i1	r2r1+r2i1	r3r1+r3i1	r4r1+r4i1
	r1r2+r1i2	r2r2+r2i2	r3r2+r3i2	r4r2+r4i2
	2ir file			
	-r1r1+r1i1	-r2r1+r2i1	-r3r1+r3i1	-r4r1+r4i1
	-r1r2+r1i2	-r2r2+r2i2	-r3r2+r3i2	-r4r2+r4i2
	2ri file			
	-i1r1+i1i1	-i2r1+i2i1	-i3r1+i3i1	-i4r1+i4i1
	-i1r2+i1i2	-i2r2+i2i2	-i3r2+i3i2	-i4r2+i4i2
	2ii file			

Note that:

- for $\text{FnMODE} \neq \text{QF}$, zero filling once in F1 is done when $\text{SI} = \text{TD}$. For $\text{FnMODE} = \text{QF}$, zero filling once in F1 is done when $\text{SI} = 2 * \text{TD}$.
- $\text{FnMODE} = \text{QF}$ 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.
- $\text{FnMODE} = \text{Echo-Antiecho}$ is equivalent to $\text{FnMODE} = \text{States}$, except that two consecutive FIDs (rows of the 2D raw data) are linearly combined according to the following rules:

$$\begin{aligned}
 \text{re0} &= -\text{im1} - \text{im0} \\
 \text{im0} &= \text{re1} + \text{re0} \\
 \text{re1} &= \text{re1} - \text{re0} \\
 \text{im1} &= \text{im1} - \text{im0}
 \end{aligned}$$

- the command **xfb n** does not store imaginary data after F1 processing.

The **ftf** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

F2 and F1 parameters

set from the **ftf** dialog box, 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 for WDW = trap

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 **s td** etc.:

TD - time domain; number of raw data points

F2 parameters

set from the **ftf** dialog box, 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 **s 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 **s fnmode** :

FnMODE - F1 Acquisition transform mode

set by the acquisition, can be viewed with **dpa** or by typing **s 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 **s si**, **s tdeff** 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 **s ymax_p**, **s 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 **s bytordp**:

BYTORDP - byte order of the processed data

INPUT FILES

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

ser - raw data (input if 2rr does not exit or is Fourier transformed)

<dir>/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
acqus - 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

<dir>/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:

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

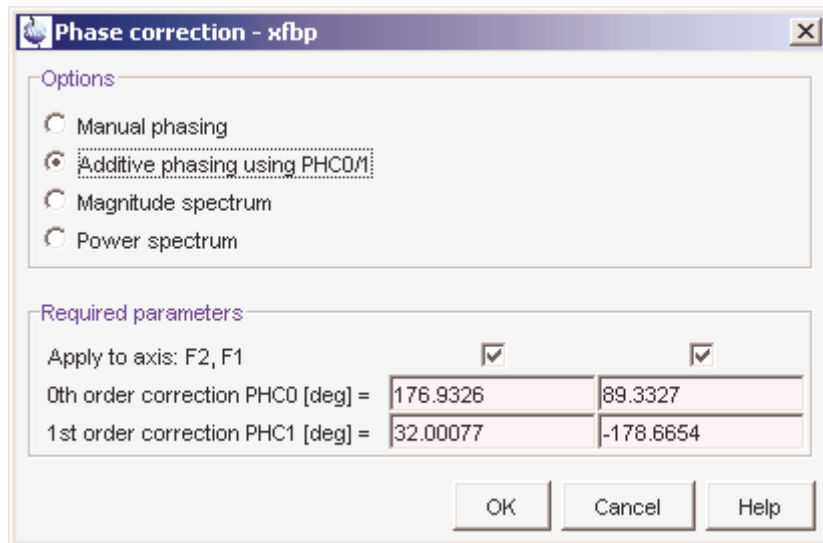
xfbp, xf2p, xf1p, ph

NAME

xfbp - 2D phase correction in the F2 and F1 dimension
 xf2p - phase correction in the F2 dimension
 xf2p - phase correction in the F2 dimension
 ph - open the phase correction dialog box

DESCRIPTION

2D phase correction can be started from the command line or from the phase correction dialog box. The latter is opened with the command **ph**:



This dialog box offers several options, each of which selects a certain command for execution.

Additive phasing using PHC0/1 in F2 and F1


This option selects the command **xfbp** for execution. It performs a zero and first order 2D phase correction in the F2 and F1 dimension. **xfbp** works like the 1D command **pk**. This means it does not calculate the phase values, it simply applies the current values of PHC0 and PHC1.

Additive phasing using PHC0/1 in F2


This option selects the command **xf2p** for execution. It works like **xfbp** except that it only corrects the phase in the F1 dimension.

Additive phasing using PHC0/1 in F1

This option selects the command **xf1p** for execution. It works like **xfbp** except that it only corrects the phase in the F2 dimension.

xf*p are only useful when the PHC0 and PHC1 values are known. If they are not, you can perform 2D interactive phase correction. To do that select the option **Manual Phasing** in the **ph** dialog box or click  in the toolbar. The interactive phase correction procedure is described in the TOPSPIN Users Guide.

The phase values can also be determined by reading a row (**rsr**) and/or column (**rsc**), determining the phase values in phase correction mode. 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.

The phase values can also be determined by the 1D interactive phase correction of a row or column. To do that, read a row (**rsr**) and/or column (**rsc**), enter the option **Manual Phasing** in the **ph** dialog box or click  in the toolbar. The interactive phase correction procedure is described in the TOPSPIN Users Guide.

Alternatively, you can phase correct a row or column with **apk** and view the calculated phase values with **dpp**. Then you can go back to the 2D dataset, set the determined phase values with **edp** and run **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.

The **ph** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **ph** dialog box, with **edp** or by typing **phc0**, **phc1**:

PHC0 - zero order phase correction value (frequency independent)

PHC1 - first order phase correction value (frequency dependent)

OUTPUT PARAMETERS

can be viewed with **dpp** or by typing **s phc0**, **s phc1**:

PHC0 - zero order phase correction value (frequency independent)

PHC1 - first order phase correction value (frequency dependent)

INPUT FILES

<dir>/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

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

2rr, ir, 2ri, 2ii - processed 2D data

procs - F2 processing status parameters

proc2s - F1 processing status parameters

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

XFBP

XF2P

XF1P

SEE ALSO

xfb, xf2, xf1, xtrf, xtrfp, xtrfp2, xtrfp1

xht2, xht1

NAME

xht2 - Hilbert transform of 2D data in the F2 dimension

xht1 - Hilbert transform of 2D data in the F1 dimension

DESCRIPTION

The command **xht2** performs a Hilbert transform of 2D data in the F2 dimension.

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**, **xf2** or **xf1**. If, however, if the imaginary data were not stored (**xfb n**) or have been deleted (**deli**), you can (re)create them with **xht2** or **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**, **abs2**, **sub***, **sym** or third party software.

INPUT FILES

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

2rr - real processed 2D data

2ir - second quadrant imaginary data (if existing, input of **xht1**)

2ri - third quadrant imaginary data (if existing, input of **xht2**)

OUTPUT FILES

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

2rr - real processed 2D data

2ir - second quadrant imaginary data (output of **xht2**, created from 2rr)

2ri - third quadrant imaginary data (output of **xht1**, created from 2rr)

2ii - fourth quadrant imaginary data

auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

XHT2

XHT1

SEE ALSO

xfb, xf2, xf1

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

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

2rr, ir, 2ri, 2ii - processed 2D data

OUTPUT FILES

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

2rr, ir, 2ri, 2ii - processed 2D data

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:

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

- 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 still 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 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 acqui-

sition mode (see INPUT PARAMETERS below and the description of **xfb**).

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 for WDW = trap

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 **s td** :

TD - time domain; number of raw data points

F1 dimension

set by the acquisition, can be viewed with **dpa** or by typing **s fnmode**:

FnMODE - Acquisition mode

OUTPUT PARAMETERS

F2 and F1 parameters

can be viewed with **dpp** or by typing **s 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 **s ymax_p, s 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 **s bytordp**:

BYTORDP - byte order of the processed data

INPUT FILES

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

ser - raw data

acqus - F2 acquisition status parameters

acqus - F1 acquisition status parameters

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

proc - F2 processing parameters

proc2 - F1 processing parameters

OUTPUT FILES

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

2rr, 2ir, 2ri, 2ii - processed 2D data
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 F1 dimension

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**

An alternative 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 for WDW = trap

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 **s 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 **s ymax_p, s 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 **s bytordp**:

BYTORDP - byte order of the processed data

INPUT FILES

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

2rr, 2ir, 2ri, 2ii - processed 2D data

proc - F2 processing parameters

proc2 - F1 processing parameters

OUTPUT FILES

<dir>/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

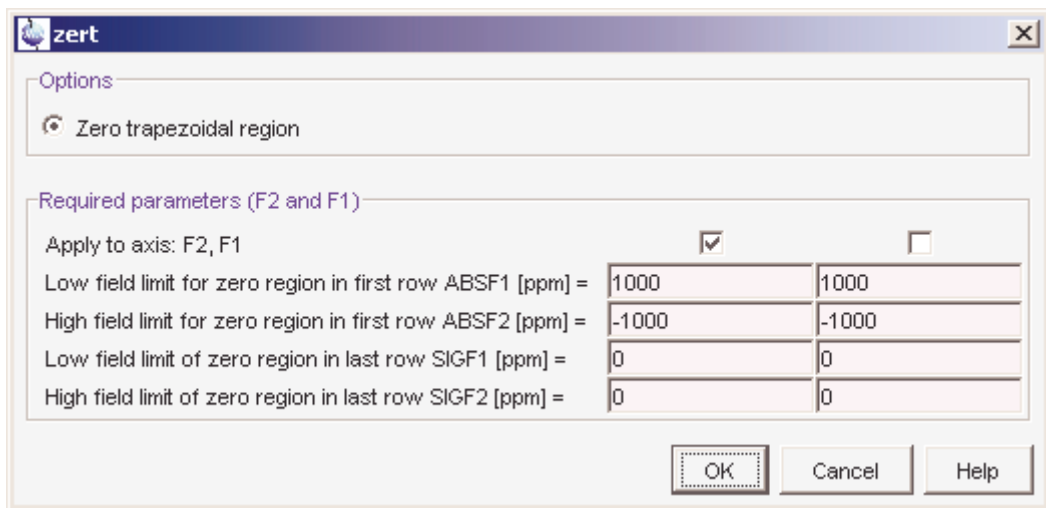
zert2, zert1, zert

NAME

zert2 - zero a trapezoidal region of each row (F2) of 2D data
 zert1 - zero a trapezoidal region of each column (F1) of 2D data
 zert - open the zero region dialog box

DESCRIPTION

The zero region commands can be started from the command line or from the zero region dialog box. The latter is opened with the command **zert**.



This dialog box offers only one option which can be used in the F2 or F1 dimension.

Zero trapezoidal region in F2

This option selects the command **zert2** for execution. The trapezoidal region to be zeroed 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.

Zero trapezoidal region in F1

This option selects the command **zert1** for execution. The trapezoidal region to be zeroed 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

set from the **zert** dialog box, 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

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

2rr - real processed 2D data

proc2 - F1 processing parameters

OUTPUT FILES

<dir>/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

ZERT2

ZERT1

SEE ALSO

abst2, abst1

Chapter 5

3D processing commands

This chapter describes all TOPSPIN 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 under ***Help*** → ***Application Manuals***.

INPUT FILES

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

difflist - list of gradient amplitudes in Gauss/cm

<dir>/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

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

3rrr - 2D processed data

auditp.txt - processing audit trail

SEE ALSO

eddosy, dosy2d

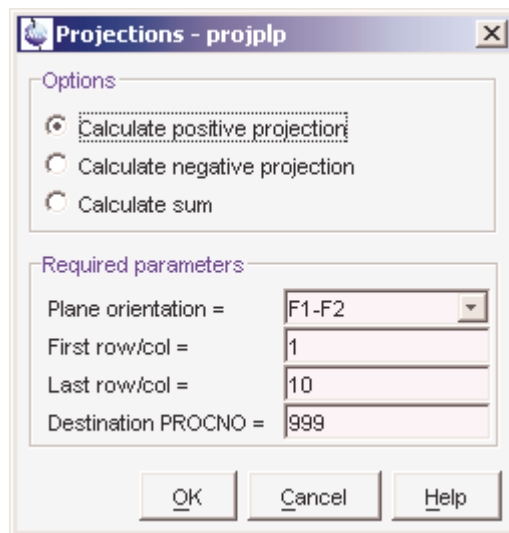
projplp, projpln, sumpl

NAME

projplp - calculate positive projection
projpln - calculate negative projection
sumpl - calculate sum projection

DESCRIPTION

The commands ***projplp***, ***projpln*** and ***sumpl*** open the following dialog box.



They calculate the positive, negative and sum projection, respectively. They require 4 parameters:

Plane orientation: F1-F2, F1-F3 or F2-F3.

First plane: the first plane considered in the calculation

Last plane: the last plane considered in the calculation

Destination PROCNO: the procno where the 2D output dataset is stored

INPUT FILES

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

3rrr - real processed 3D data

OUTPUT FILES

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

2rr - real processed 2D data

SEE ALSO

rpl, wpl, rser2d

r12, r13, r23, slice

NAME

r12 - read an F1-F2 plane from 3D processed data and store it as 2D data
r13 - read an F1-F3 plane from 3D processed data and store it as 2D data
r23 - read an F2-F3 plane from 3D processed data and store it as 2D data
slice - open the read plane dialog box

DESCRIPTION

The commands **r12**, **r13** and **r23** read a plane from 3D processed data and store it as a 2D dataset.

r12, **r13** and **r23** are equivalent to the commands **rp1 12**, **rp1 13** and **rp1 23**, respectively (see the description of **rp1**).

When entered without arguments, they open the following dialog box

Here you can specify three required parameters:

Plane orientation: F1-F2, F1-F3 or F2-F3. This parameter determines which of the commands **r12**, **r13** or **r23** is executed.

Plane number: the maximum plane number is the SI value in the direction orthogonal to the plane orientation.

Destination PROCNO: the procno where the output 2D dataset is stored

The parameters can also be entered as arguments on the command line. In that case, the command is executed without opening the dialog box. For example:

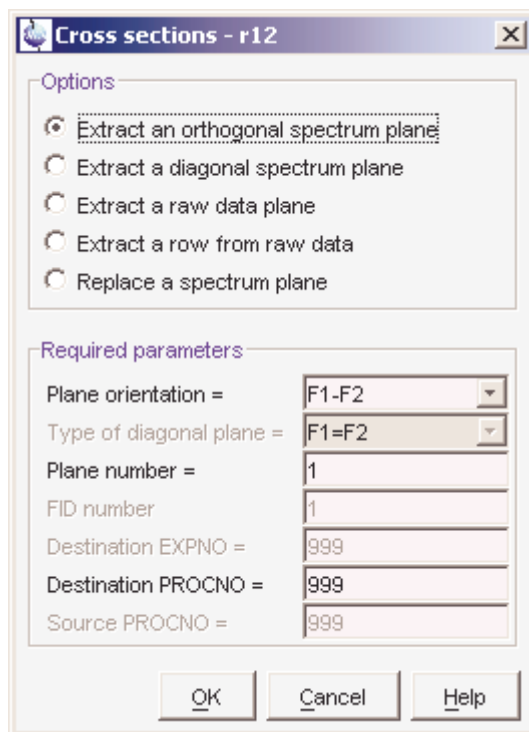
r12 10 999

reads an F1-F2 plane number 10 and stores it in PROCNO 999. Note that the Plane orientation is not specified as an argument but part of the command name.

This dialog box offers several options, each of which selects a certain command for execution. For each command a table below 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, an extracted plane can be further processed with 2D processing commands like ***xf2***, ***xf1***, ***xf2p*** etc.

Extract an orthogonal spectrum plane in F1-F2

This option selects the command ***r12*** for execution. It reads an F1-F2 plane

from a 3D dataset and stores it as a 2D dataset.

3D data processed with	3D input data			2D output data	
	F3	F2	F1	F2	F1
tf3	real+imag	FID	FID	FID ^a	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 5.1 *r12* input/output data

a. The two channels of the raw data are split and only one channel is displayed.

Extract an orthogonal spectrum plane in F1-F3

This option selects the command *r13* for execution. It reads an F1-F3 plane from a 3D dataset and stores it as a 2D dataset.

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 5.2 *r13* input/output data

Extract an orthogonal spectrum plane in F2-F3

This option selects the command *r23* for execution. It reads an F2-F3 plane

from a 3D dataset and stores it as a 2D dataset.

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 5.3 *r23* input/output data

INPUT FILES

<dir>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/
3rrr, 3irr, 3rir, 3rri, 3iii - processed 3D data

OUTPUT FILES

<dir>/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, procno)
for example R12(64, 1)

R13(plane, procno)
for example R13(64, 1)

R23(plane, procno)
for example R23(64, 1)

SEE ALSO

r12d, r13d, r23d, rpl, wpl

r12d, r13d, r23d

NAME

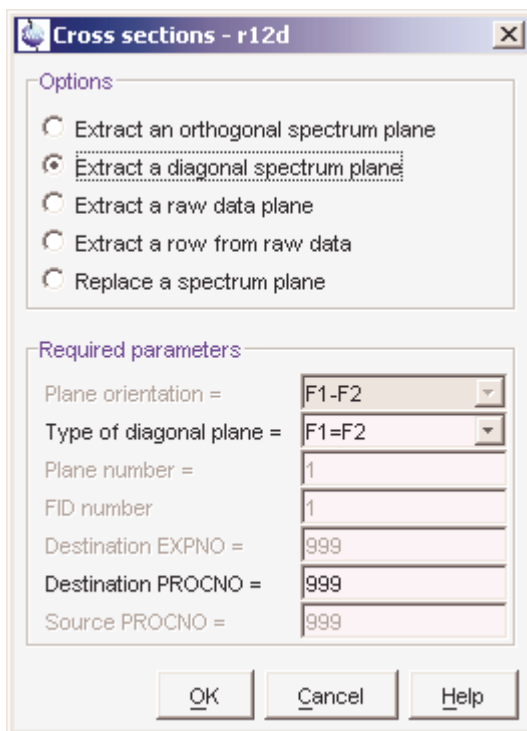
r12d - read a diagonal F1=F2 plane from 3D data and store it as 2D data

r13d - read a diagonal F1=F3 plane from 3D data and store it as 2D data

r23d - read a diagonal F2=F3 plane from 3D data and store it as 2D data

DESCRIPTION

Read plane commands can be started from the command line or from the read plane dialog box. The latter is opened with the command ***slice***



This dialog box offers several options, each of which selects a certain command for execution.

Extract an diagonal spectrum plane in F1-F2

This option selects the command ***r12d*** for execution. It reads the diagonal F1=F2 plane from a 3D dataset and stores it as a 2D dataset.

Extract an diagonal spectrum plane in F1-F3

This option selects the command ***r13d*** for execution. It reads the diagonal F1=F3 plane from a 3D dataset and stores it as a 2D dataset.

Extract an diagonal spectrum plane in F2-F3

This option selects the command ***r23d*** for execution. It reads the diagonal F2=F3 plane from a 3D dataset and stores it as a 2D dataset.

r12d, ***r13d*** and ***r23d*** only store the real data.

INPUT FILES

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

3rrr - real processed 3D data

OUTPUT FILES

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

2rr - real processed 2D data

SEE ALSO

r12, r13, r23, rpl, wpl

rpl

NAME

rpl - read a plane from 3D processed data and store it as 2D data

DESCRIPTION

The command **rpl** reads a plane from 3D processed data and stores it in a different PROCNO as a 2D dataset.

rpl takes up to four arguments:

<plane direction> : 12, 13, 23, 21, 31 or 32

The digits refer to the F3, F2 and F1 axes of the 3D data. Note that the order of the two digits is relevant:

- the first digit is the 3D axis that corresponds to the 2D-F1 axis
- the last digit is the 3D axis that corresponds to the 2D-F2-axis

This means that for the values 21, 31 and 32, the data are transposed, storing rows as columns and vice versa (see below).

<plane number> : 1 - SI

SI is the 3D size in the direction orthogonal to the plane orientation

<procno> :

destination 2D PROCNO (source 3D PROCNO if **rpl** is entered on the 2D destination dataset)

n : optional argument

prevents the destination dataset from being displayed/activated

Arguments which are not specified on the command line will be prompted for, except for the **n** argument.

Examples:

rpl 23 10 999

read F2-F3 plane 10 and store it in PROCNO 999.

rpl 32 10 999

read F2-F3 plane 10 and store it in PROCNO 999, transposing the data.

rpl 12 64 101

read F1-F2 plane 64 and store it in PROCNO 101.

rpl 12 64

read F1-F2 plane 64, prompt the user for the destination PROCNO

rpl 31 1 10 n

read an F1-F3 plane number 1 and store it in PROCNO 10, transposing the data. Do not display/activate the destination dataset.

When ***rpl*** has been executed once, it can be repeated from the destination 2D processed dataset to read different planes. ***rpl*** now requires one argument only; the *Plane number*. By default, the same *Plane Direction* and *3D Source dataset (PROCNO)* are used as with the previous ***rpl*** command (as defined in the `used_from` file) of the 2D dataset. You can, however, use two or three arguments to specify a different *Plane Direction* and/or *3D Source PROCNO*.

Examples:

rpl

prompt the user for the plane number. Use the Plane Direction and Source 3D PROCNO as defined in the current 2D dataset.

rpl 11

read plane 11. Use the Plane Direction and Source 3D PROCNO as defined in current 2D dataset.

rpl 31 11

read F1-F3 plane 11, transposing the data. Use the Source 3D PROCNO as defined in current 2D dataset.

rpl 13 11 2

read F1-F3 plane 11 from the 3D dataset under PROCNO 2

As described above, the ***rpl*** argument *Plane direction* determines whether the data are transposed. Data transposition is sometimes required to match nuclei when you compare a 3D plane with a 2D dataset. Example: you have a 3D NOESYHSQC (F3-1H, F2-13C, F1-1H) and want to compare an F2-F1 plane

with a 2D HSQC (F2-1H, F1-13C).

rp1 12: The plane is stored as a 2D dataset with F2-13C, F1-1H which cannot be directly compared with the a HSQC.

rp1 21: The plane is stored as a 2D dataset with F2-1H, F1-13C which can be directly compared with the a HSQC.

The tables below show the relation between 3D input data and 2D output data for 3D data which are processed in one, two or three directions .

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 5.4 **rp1 13** input/output data

3D data processed with	3D input data			2D output data	
	F3	F2	F1	F2	F1
tf3	real+imag	FID	FID	FID ^a	real+imag
tf3, tf2	real	real+imag	FID	FID ^a	real
tf3, tf2, tf1	real	real	real+imag	real+imag	real
tf3, tf1, tf2	real	real+imag	real	real	real

Table 5.5 **rp1 31** input/output data

a. The two channels of the raw data are split and only one channel is displayed.

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 *rp1 23* input/output data

3D data processed with	3D input data			2D output data	
	F3	F2	F1	F2	F1
tf3	real+imag	FID	FID	FID ^a	real+imag
tf3, tf2	real	real+imag	FID	real+imag	real
tf3, tf2, tf1	real	real	real+imag	real	real
tf3, tf1, tf2	real	real+imag	real	real+imag	real

Table 5.7 *rp1 32* input/output data

a. The two channels of the raw data are split and only one channel is displayed.

3D data processed with	3D input data			2D output data	
	F3	F2	F1	F2	F1
tf3	real+imag	FID	FID	FID ^a	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 5.8 *rp1 12* input/output data

a. The two channels of the raw data are split and only one channel is displayed.

3D data processed with	3D input data			2D output data	
	F3	F2	F1	F2	F1
tf3	real+imag	FID	FID	FID ^a	FID
tf3, tf2	real	real+imag	FID	FID ^a	real+imag
tf3, tf2, tf1	real	real	real+imag	real+imag	real
tf3, tf1, tf2	real	real+imag	real	real	real+imag

Table 5.9 *rp1 21* input/output data

a. The two channels of the raw data are split and only one channel is displayed.

Note that the channels of F2 raw output data are split and stored in separate files. This allows the output data to be processed with **xf2** or **xfb**. F2 raw output data are especially created when data transposition takes place, i.e. when **rp1** takes the argument **32**, **31** or **21**.

Note that the command **rp1** replaces the old commands **r12**, **r13** and **r23** which do not allow data transposition. For compatibility reasons, these commands are still available. Note however, their usage in AU programs has changed compared to TOPSPIN 1.1 and XWIN-NMR (see the description of **r12**)

The behaviour of the 3D command **rp1** is similar to the 2D commands **rsr** and **rsc**, in the sense that it can be entered from the source and destination dataset.

INPUT FILES

```
<dir>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/
```

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

OUTPUT FILES

```
<dir>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/
```

2rr, 2ir, 2ri, 2ii - processed 2D data

auditp.txt - processing audit trail

```
<dir>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/
```

used_from - data path of the source 3D data and the plane number

SEE ALSO

wpl, r12, r13, r23, rser, wser, wserp, projplp, projpln, sumpl

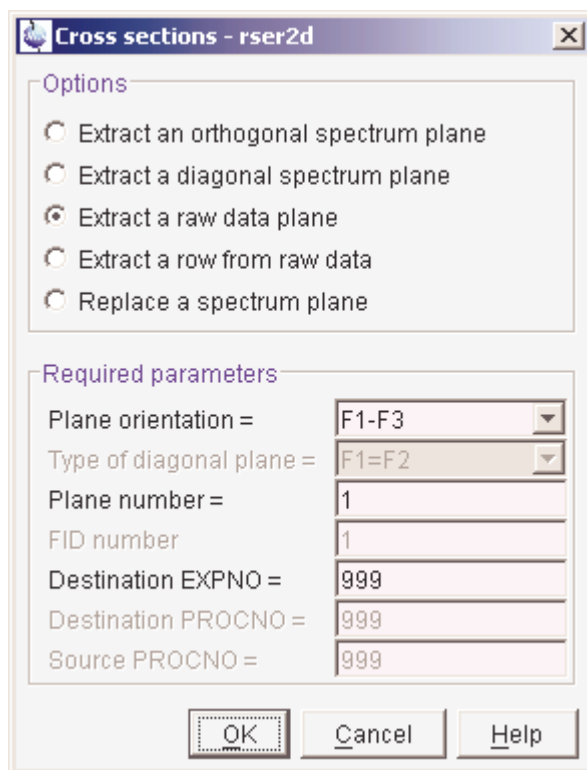
rser2d

NAME

rser2d - read a plane from 3D raw data and store it as a 2D pseudo-raw data

DESCRIPTION

The command **rser2d** reads a plane from 3D raw data (a series of FIDs) and stores it as a pseudo 2D dataset. When entered without arguments, it opens the following dialog box:



Here you can specify three required parameters:

Plane orientation: F1-F3 or F2-F3 (must contain acquisition (F3) direction)

Plane number: the maximum plane number is the TD value in the direction

orthogonal to the plane orientation

Destination EXPNO: the expno where the output 2D dataset is stored

The parameters can also be entered as arguments on the command line. In that case, the command is executed without opening the dialog box. For example:

rser2d s23 10 999

reads an F3-F2 plane number 10 and stores it in EXPNO 999

In contrast to ***rser***, ***rser2d*** can only be entered on the source dataset, not on the destination dataset.

INPUT FILES

<dir>/data/<user>/nmr/<name>/<expno1>/

ser - 3D raw data

OUTPUT FILES

<dir>/data/<user>/nmr/<name>/<expno2>/

ser - 2D pseudo raw data

audita.txt - acquisition audit trail

<dir>/data/<user>/nmr/<name>/<expno2>/pdata/1/

used_from - data path of the source 3D data and the plane number

USAGE IN AU PROGRAMS

RSER2D (direction, plane, expno, name, user, dir)

SEE ALSO

rser, wser, wserp, rpl, wpl

tabs3, tabs2, tabs1

NAME

tabs3 - automatic baseline correction in the F3 dimension

tabs2 - automatic baseline correction in the F2 dimension

tabs1 - automatic baseline correction in the F1 dimension

DESCRIPTION

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.

tabs2 works like **tabs3** except that corrects data in the F2 dimension using the F2 parameters ABSG, ABSF2 and ABSF1.

tabs1 works like **tabs3** except that corrects data in the F1 dimension using the F1 parameters ABSG, ABSF2 and ABSF1.

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 limit of the correction region

ABSF2 - high field limit of the correction region

INPUT FILES

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

3rrr - real processed 3D data

proc - F3 processing parameters

OUTPUT FILES

<dir>/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

TABS2

TABS1

SEE ALSO

abs2, abs1

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 5.10. **tf3** does not evaluate the

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 5.10

processing parameter FT_mod! However, it stores the Fourier transform mode in the processing status parameter FT_mod.

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. $0 < \text{STSR} < \text{SI}$: only the processed data between STSR and STSR+STSI are stored (if STSI = 0, STSR is ignored and SI points are stored)
7. $0 < \text{STSI} < \text{SI}$: 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**)

xdim

3D spectra are stored in the so-called subcube format. The size of the subcubes is calculated by **tf3** and depends on the size of the spectrum and the available memory. The option **xdim** allows you to use predefined subcube sizes. It causes **tf3** to interpret the F3, F2 and F1 processing parameter XDIM which can be set with the command **xdim**. The actually used subcube sizes, whether predefined or calculated, are stored as the F3, F2 and F1 processing status parameter XDIM and can be viewed with **dpp**. Predefining subcube sizes is, for example, used to read the processed data with third party software which can not interpret the processing status parameter XDIM.

big/little

tf3 stores the data in the data storage order of the computer it runs on, e.g. little endian on Windows PCs. Note that TOPSPIN's predecessor XWIN-NMR on SGI UNIX workstations stores data in big endian. 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 top level data directory, typically a different disk. The rest of the data directory path is the same as that of the raw data.

If the specified top level 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** 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 for WDW = trap
 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 **s 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 **s fnmode** etc.:

FnMODE - Fourier transform mode

OUTPUT PARAMETERS

F3 parameters

can be viewed with **dpp** or by typing **s si, s tdef** 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 **s bytordp**:

BYTORDP - byte order of the processed data

F3, F2 and F1

can be viewed with **dpp** or by typing **s si, s stsi** 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 *s mc2*, *s mc2* etc.:

MC2 - Fourier transform mode

INPUT FILES

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

ser - raw data

acqus - F3 acquisition status parameters

acqui2s - F2 acquisition status parameters

acqui3s - F1 acquisition status parameters

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

proc - F3 processing parameters

proc2 - F2 processing parameters

proc3 - F1 processing parameters

OUTPUT FILES

<dir>/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* is the top level data directory

SEE ALSO

tf2, tf1

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 sta-

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 5.11

tus parameter MC2 as shown in table 5.11. **tf2** does not evaluate the processing parameter FT_mod! However, it stores the Fourier transform mode as it was evaluated from MC2 in the processing status parameter 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 spec-

trum 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**.

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 **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 for WDW = trap

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 **s si**, **s stsi** 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 **s mc2** :

MC2 - Fourier transform mode

F1 parameters

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

TD - time domain; number of raw data points

OUTPUT PARAMETERS

F2 parameters

can be viewed with **dpp** or by typing **s ft_mod** :

FT_mod - Fourier transform mode

FTSIZE - Fourier transform size

F3 parameters

can be viewed with **dpp** or by typing **s ymax_p, s 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

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

acqu2s - F2 acquisition status parameters

<dir>/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

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

3rrr - real processed 3D data
3rir - real/imaginary data (if MC2 \neq 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

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

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 5.12

parameter MC2 as shown in table 5.11. **tf1** does not evaluate the processing parameter FT_mod! However, it stores the Fourier transform mode as it was evaluated from MC2 in the processing status parameter 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 \neq 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**.

INPUT PARAMETERS

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 for WDW = trap

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 **ssi**, **sstsi** 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 **s mc2** :

MC2 - Fourier transform mode

OUTPUT PARAMETERS

F1 parameters

can be viewed with **dpp** or by typing **s ft_mod** :

FT_mod - Fourier transform mode

FTSIZE - Fourier transform size

F3 parameters

can be viewed with **dpp** or by typing **s 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

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

acq2s - F1 acquisition status parameters

<dir>/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

<dir>/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

tf3p, tf2p, tf1p

NAME

tf3p - 3D phase correction in the F3 dimension
 tf2p - 3D phase correction in the F2 dimension
 tf1p - 3D phase correction in the F1 dimension

DESCRIPTION

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**. These commands will automatically switch to the resulting 2D data window. To enter the 2D phase mode, click the  button in the toolbar or enter **.ph** on the command line. Phase correct the plane as described in the TOPSPIN Users Manual. Return to the 3D dataset and set (with **edp**) the parameters PHC0 and PHC1 to the values which you determined for the plane.

tf2p works like **tf3p** except that it works in the F2 dimension applying the F2 parameters PHC0 and PHC1. These can be determined on a 2D plane extracted with **r23** or **r12**.

tf1p works like **tf3p** except that it works in the F1 dimension applying the F1 parameters PHC0 and PHC1. These can be determined on a 2D plane extracted with **r13** or **r12**.

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 is already done as a part of the commands **tf3**, **tf2** and **tf1**, if PH_mod = pk and PHC0 and PHC1 are set.

INPUT 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

can be viewed with **dpp** or by typing **s phc0**, **s 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

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

3rrr - real processed 3D data
3irr - F3 imaginary processed data (input of **tf3p**)
3rir - F2 imaginary processed data (input of **tf2p**)
3rri - F1-imaginary processed data (input of **tf1p**)

OUTPUT FILES

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

3rrr - real processed 3D data
3irr - F3 imaginary processed data (output of **tf3p**)
3rir - F2 imaginary processed data (output of **tf2p**)
3rri - F1-imaginary processed data (output of **tf1p**)
auditp.txt - processing audit trail

USAGE IN AU PROGRAMS

TF3P(store_imag)
where *store_image* can be y or n

TF2P(store_imag)
where *store_image* can be y or n

TF1P(store_imag)
where *store_image* can be y or n

SEE ALSO

tf3, tf2, tf1, xf2p, xf1p, pk

tht3, tht2, tht1

NAME

tht3 - 3D Hilbert transform of 3D data in the F3 dimension

tht2 - 3D Hilbert transform of 3D data in the F2 dimension

tht1 - 3D Hilbert transform of 3D data in the F1 dimension

DESCRIPTION

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**.

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**.

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. 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 Hilbert transform. Imaginary data do not match the real data if the latter have been manipulated after the Fourier transform, for example by baseline correction or third party software.

INPUT FILES

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

3rrr - real processed 3D data

OUTPUT FILES

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

3irr - F3 imaginary processed data (output of **tht3**)

3rir - F2 imaginary processed data (output of **tht2**)

3rri - F1-imaginary processed data (output of **tht1**)

auditp.txt - processing audit trail

SEE ALSO

tf3, tf2, tf1

wpl

NAME

wpl - write 2D processed dataset to plane of a 3D processed dataset

DESCRIPTION

The command **wpl** replaces a plane of a 3D processed dataset with a 2D processed dataset. It is usually, but not necessarily, used to write back a plane extracted with **rpl**, either modified and/or to a different plane number.

wpl takes up to three arguments:

<plane direction> : 12, 13, 23, 21, 31 or 32

The digits refer to the F3, F2 and F1 axes of the 3D data. Note that the order of the two digits is relevant:

- the first digit is the 3D axis that corresponds to the 2D-F1 axis
- the last digit is the 3D axis that corresponds to the 2D-F2-axis

This means that for the values 21, 31 and 32, the data are transposed, i.e. rows are stored as columns and vice versa (see below).

<plane number> : 1 - SI

SI is the 3D size in the direction orthogonal to the plane orientation

<procno>

destination 3D PROCNO (source 3D PROCNO if **rpl** is entered on the 2D destination dataset)

Normally, **wpl** is entered from a 2D dataset which is actually a plane extracted from a 3D dataset. In this case, **wpl** prompts for two arguments only, *Plane number* and *3D Destination PROCNO* only. The *Plane direction* is taken from the 2D dataset (*used_from* file). The two arguments can also be specified on the command line. If, however, you specify a three arguments, the *Plane direction* is taken from the first argument rather than from the 2D dataset.

Examples:

wpl

prompt the user for the plane number and destination PROCNO, take the

plane direction from the 2D dataset and write the current 2D data accordingly.

wpl 11 1

write the current 2D data to plane 11 of the 3D dataset in PROCNO 1. Use the plane direction defined in the 2D dataset.

wpl 13 11 2

write the current 2D data to F1-F3 plane 11 of the 3D data in PROCNO 2. Ignore the plane direction defined in the 2D dataset

Note that if the `used_from` file does not exist, for example because the 2D dataset is not an extracted plane, **wpl** will prompt the user for the Plane orientation.

wpl can also be entered on the destination 3D dataset. In this case, it will prompt the user for all three arguments. Alternatively, these can be entered on the command line.

Examples:

wpl 23 10 999

write the 2D data in PROCNO 999 to F2-F3 plane 10 of the current 3D data

wpl 12 32 101

write the 2D data in PROCNO 101, to the F1-F2 plane 32 of the current 3D data

wpl 12

Prompt the user for the PROCNO of the source 2D dataset and for the plane number. Write the 2D dataset to the specified F1-F2 plane accordingly.

The behaviour of the 3D command **wpl** is similar to the 2D commands **wsr** and **wsc**, in the sense that it can be entered from the source and destination dataset.

INPUT FILES

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

2rr, 2ir, 2ri, 2ii - processed 2D data

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

used_from - data path of the source 3D data and the plane number

OUTPUT FILES

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

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

auditp.txt - processing audit trail

SEE ALSO

rpl, r12, r13, r23, rser, wser, wserp, projplp, projpln, sumpl

Chapter 6

Print/Export commands

This chapter describes TOPSPIN print, plot and export commands. Printing can be done directly from the TOPSPIN interface or from the Plot Editor. The data window can be exported into a graphics file. Command are available for setting the plot title and, for 2D and 3D data, the contour levels.

autoplot

NAME

autoplot - plot the current data according to a Plot Editor layout

DESCRIPTION

The command **autoplot** plots the current dataset according to a Plot Editor layout. The layout must be specified with the processing parameter LAYOUT. This layout can be a standard Plot Editor layout which is delivered with TOPSPIN or a user defined layout which you have set up from the Plot Editor.

The Plot editor 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 chosen layout. Note that **autoplot** always uses the current dataset and ignores the first dataset defined in `portfolio.por`.

INPUT FILES

<tsHOME>/plot/layouts/*.xwp - Bruker library Plot Editor layouts

<dir>/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 - Plot Editor portfolio (input file if it exists)

For a 2D dataset, the files `2rr`, `proc2s` and `clevels` are also input.

USAGE IN AU PROGRAMS

AUTOPLOT

SEE ALSO

plot, print, prnt

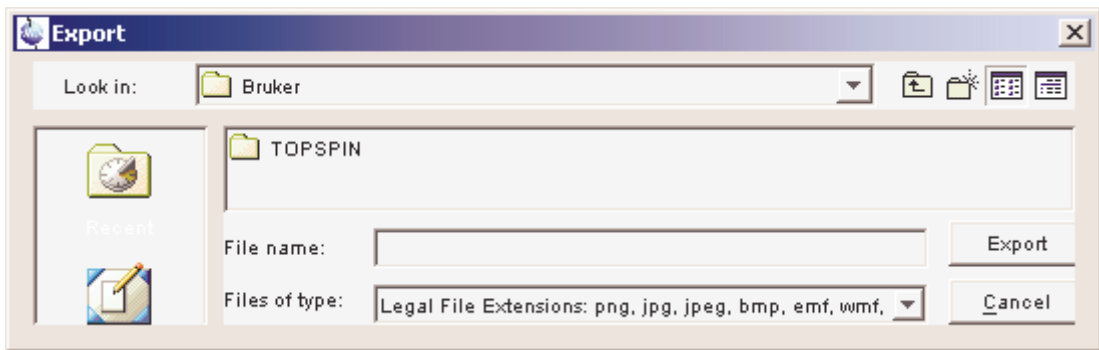
exportfile

NAME

exportfile - store (export) a data window as graphics file

DESCRIPTION

The command **exportfile** saves the contents of a data window in a graphics file of selectable type, e.g. .png, .tif, .wmf etc. It opens an Explorer window.



Here you can:

- Click or type the output file
- Click **Exportfile**

The resolution of such a *screen dump* equals the resolution of your screen. When you import a graphics file into another program, you may lose information when resizing the graphics.

Entering **exportfile** on the command line is equivalent to clicking **File** → **Export....**

OUTPUT FILES

<outputdir>

outputfile[.png, .jpg, .jpeg, .bmp, .emf, .wmf] - graphics file

SEE ALSO

plot, autoplot, prnt, print

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.

The dialog box titled "exam2d_HC 1 1 C:\bio guest" contains a table of contour levels and a section for required parameters.

1	262783.0	-1000000.0
2	473009.4	-1000000.0
3	851416.9	-1800000.0
4	1532550.5	-1800000.0

Required parameters

Calculation method

☒ Multiply with increment
☐ Add increment

Contour level sign

☒ Positive & Negative
☐ Positive
☐ Negative

	Positive	Negative
Base level	3000000.0	-2102264.0
Level increment	1.800	1.800
Number of levels		16

Fill Clear

OK Cancel Help

These are the levels which appear on a plot created with the Plot Editor (command ***plot***). They also appear on the screen.


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 TOPSPIN toolbar by clicking the  button.

INPUT AND OUTPUT FILES

```
<dir>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/
```

```
  clevels - 2D contour levels
```

SEE ALSO

plot, limits2d

edti

NAME

edti - set the dataset title

DESCRIPTION

The command **edti** allows you to define the dataset title. Entering this command is equivalent to clicking the *Title* tab. Changes in the title will automatically appear in the data window after clicking the *Spectrum* or *Fid* tab.

The title defined with will also appear on plots created with **prnt** or **auto-plot**.

The command **edti** replaces the formerly used command **setti** which is still available.

INPUT FILES

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

title - plot title

OUTPUT FILES

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

title - plot title

SEE ALSO

plot, prnt, print, autoplot

plot

NAME

plot - start the Plot Editor

DESCRIPTION

The command **plot** starts the Plot Editor. The main window of the Plot Editor consists of a drawing area, a menu bar and a toolbar which offers various graphical objects. Here you can display objects like FIDs, one- or two-dimensional NMR spectra, Stacked Plots, parameter lists and titles. You can add integral curves and peak lists to a spectrum, combine several spectra to a stacked plot, draw projections around a 2D spectrum.

Furthermore, the Plot Editor offers a set of so-called graphic primitives like lines, text, rectangles and bezier curves. You can place these objects anywhere on the screen and change their appearance. They can be superimposed on NMR-related graphics. All objects can be moved and resized interactively and for each object a range of editing modes is available.

The Topspin command **autoplot** allows you to plot a spectrum using a Plot Editor layout.

For a full description, please refer to the Plot Editor online help.

SEE ALSO

print, prnt, autoplot

print

NAME

print - print the current dataset

DESCRIPTION

The command **print** opens the following dialog box

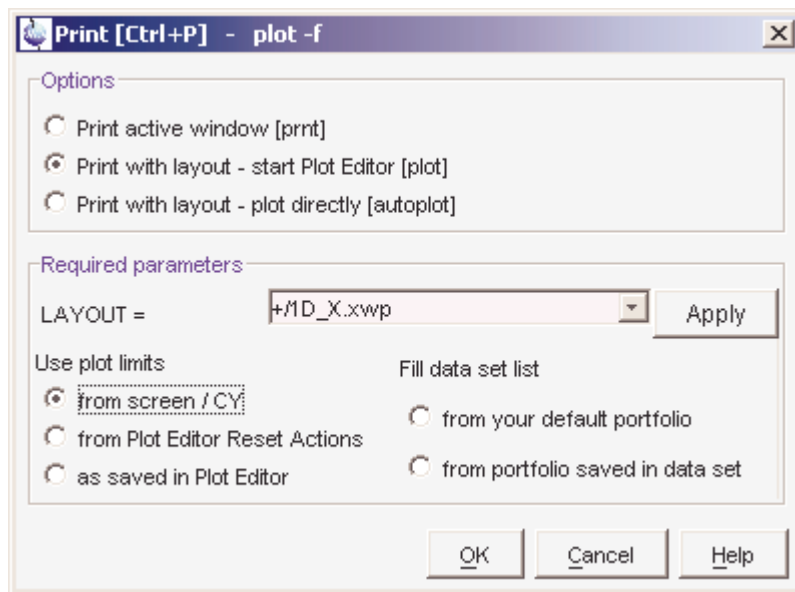


Figure 6.1

Here, you can choose from three print options:

- **Print active window [prnt]**

The data window is printed as it is displayed on the screen. Before printing starts, the operating system print dialog box will appear where you can, for example, select the printer and printer properties.

- **Print with layout - start Plot Editor [plot]**

If you select this option and click **OK**, the Plot Editor will be started. This option is equivalent to entering **plot** on the TOPSPIN command line.

- ***Print with layout - plot directly*** [***autoplot***]

Selecting this option activates the Plot Editor layout list box. Select the desired layout and click **OK** to print. Standard layouts are delivered with TOPSPIN. They use the Windows default printer. User defined layouts use the printer defined in the Plot Editor. On a 1D dataset, only 1D layouts are listed, on a 2D dataset only 2D layouts are listed etc.

For the last two options, the following Required Parameters are available:

Use plot limits:

- ***from screen/ CY***
the plot limits and maximum intensity are used as they are on the screen (processing parameter F1P, F2P and CY, respectively)
- ***from Plot Editor Reset Actions***
the plot limits and maximum intensity are set according to the Plot Editor Reset Actions (right-click inside the Plot Editor data field and choose ***Automation*** to set the Reset Actions).
- ***as saved in Plot Editor***
the plot limits and maximum intensity are set in the specified layout

Fill dataset list:

- ***from your default portfolio***
the portfolio contains the current TOPSPIN dataset plus the data from the default Plot Editor portfolio
- ***from port folio saved in dataset***
the portfolio contains the current TOPSPIN dataset plus the data from the portfolio stored in this dataset

The ***Apply*** button stores the indicated layout to the processing parameter LAYOUT. As such, this layout will be used by the ***autoplot*** command when it is entered from the command line.

For each Option/Required Parameter combination, the corresponding command line command is shown in the title bar of the dialog box. In the example above this is the command ***plot -f***.

INPUT FILES

see the description of ***prnt***, ***plot*** and ***autoplot***

SEE ALSO

prnt, plot, autoplot

prnt

NAME

prnt - print the current dataset

DESCRIPTION

The command ***prnt*** prints the current dataset as it is shown on the screen. Before printing starts, the operating system print dialog box will appear. Here you can, for example, select the printer and printer properties.

SEE ALSO

print, plot, autoplot

Chapter 7

Analysis commands

This chapter describes TOPSPIN analysis commands for 1D, 2D and 3D data. Although they do not really process (manipulate) the data, they are part of the processing part of TOPSPIN. 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 ***edti*** and ***pps***). None of them, however change the processed data.

int2d, int

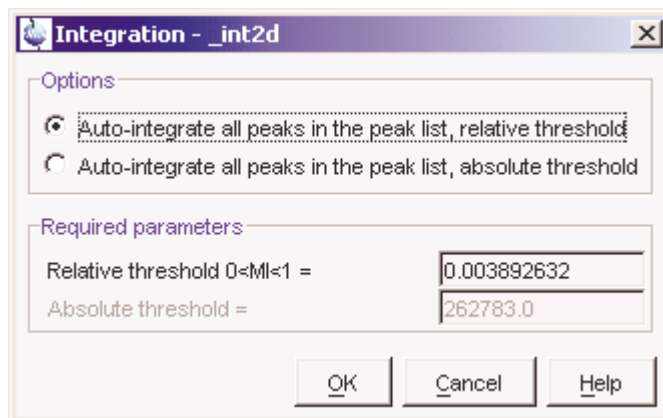
NAME

int2d - calculate 2D integrals

int - open the integral dialog box

DESCRIPTION

The command **int2d** calculates 2D integrals. It opens the following dialog box.



Here you can choose between integration using a relative or absolute threshold and set the required parameters. The calculated integrals will be marked in the data field and can be listed by clicking the **Integrals** tab.

The **int** command can be used on 1D or 2D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT FILES

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

2rrr - real processed 2D data

OUTPUT FILES

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

integ_points.txt - data points of integral regions

`integrals.txt` - peaks, integral regions and integral values

SEE ALSO

li

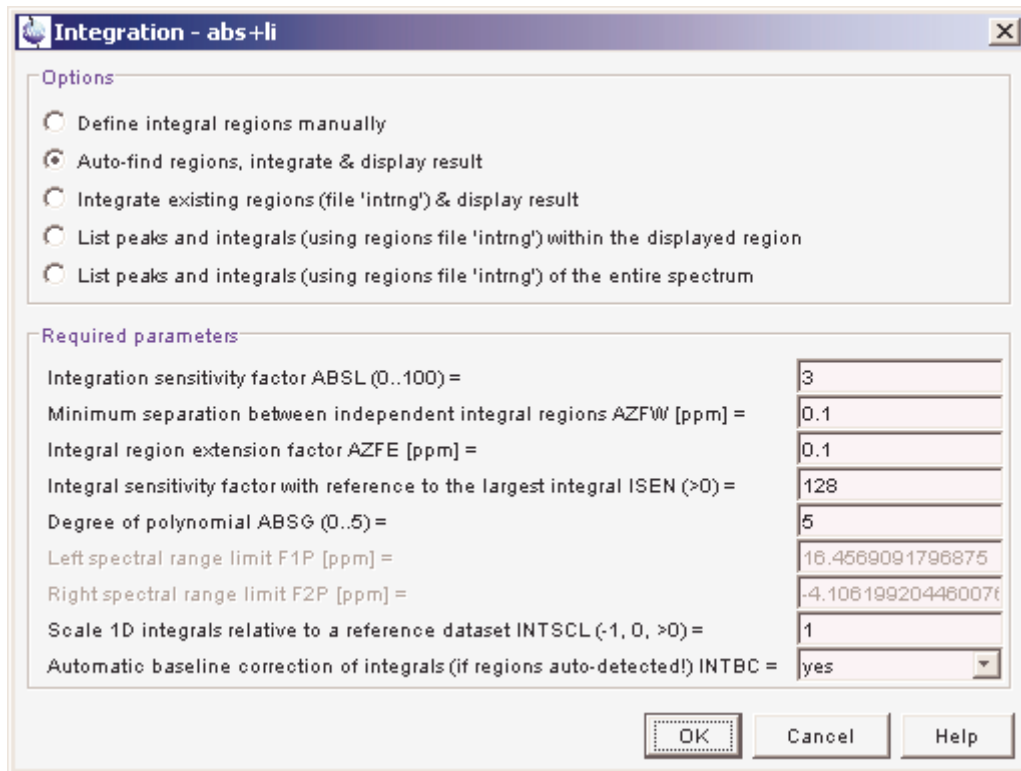
li, lipp, lippf, int

NAME

- li - list integrals of a 1D or 2D dataset
- lipp - list integrals and peaks of the regions defined by F1P,F2P
- lippf - list integrals and peaks of the full spectrum
- int - open the integral dialog box

DESCRIPTION

Integral commands can be entered from the command line or started from the integration dialog box. The later is opened with the command **int**



The image shows a Windows-style dialog box titled "Integration - abs+li". It contains two main sections: "Options" and "Required parameters".

Options:

- ☐ Define integral regions manually
- ☒ Auto-find regions, integrate & display result
- ☐ Integrate existing regions (file 'intrng') & display result
- ☐ List peaks and integrals (using regions file 'intrng') within the displayed region
- ☐ List peaks and integrals (using regions file 'intrng') of the entire spectrum

Required parameters:

Integration sensitivity factor ABSL (0..100) =	3
Minimum separation between independent integral regions AZFW [ppm] =	0.1
Integral region extension factor AZFE [ppm] =	0.1
Integral sensitivity factor with reference to the largest integral ISEN (>0) =	128
Degree of polynomial ABSG (0..5) =	5
Left spectral range limit F1P [ppm] =	16.4569091796875
Right spectral range limit F2P [ppm] =	-4.106199204460076
Scale 1D integrals relative to a reference dataset INTSCL (-1, 0, >0) =	1
Automatic baseline correction of integrals (if regions auto-detected!) INTBC =	yes

At the bottom right are three buttons: "OK", "Cancel", and "Help".

This dialog box has several options, each of which selects a certain command for execution.

Auto-find regions, integrate & display results

This option executes the command sequence **abs** - **li**. The command **abs** determines the integral regions creating the 'intrng' file. The command **li** calculates the integral value for each integral region and shows the result in the *Integrals* pane.

Integrate existing regions and display results

This option executes the command sequence **abs** - **li**. The command **abs** determines the integral regions creating the 'intrng' file. The command **li** calculates the integral value for each integral region and shows the result in the *Integrals* pane.

List peaks and integrals within the displayed region

This option executes the command **lipp**. It works like **li**, except that it also performs peak picking and shows a list of integral regions and peaks.

List peaks and integrals of the entire spectrum

This option executes the command **lippf**. It works like **lipp**, except that it only determines the integrals and peaks within the region F1P - F2P.

The **li*** commands evaluate 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. Furthermore, the parameter INTBC is evaluated. For INTBC = yes, an automatic baseline correction (slope and bias) of the integrals is performed. This, however, is only done when the integral regions were determined with **abs**, not if they were determined interactively.

The **int** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set with **edp**, from the **int** dialog box or by typing **intscl**, **intbc** etc.:

INTSCL - scale 1D integrals relative to a reference dataset

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

F1P - low field (left) region in ppm (input for ***lipp***)

F2P - high field (right) region in ppm (input for ***lipp***)

INPUT FILES

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

1r - real processed 1D data

intrng - 1D integral regions (created by ***abs*** or interactive integration)

OUTPUT FILES

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

integrals.txt - ascii file containing the output of ***li****

peaks - peak file (output of ***lipp****)

USAGE IN AU PROGRAMS

LI

LIPP

LIPPF

SEE ALSO

int2d

mana

NAME

mana - Switch to multiplet analysis mode

DESCRIPTION

The command **mana** switches to multiple analysis mode. It is equivalent to clicking: *Analysis* → *Multiplet Analysis*. Multiplet analysis is completely described in the Topspin Users Guide.

pps, ppl, pph, ppj, pp

NAME

pps - perform peak picking and show peaks on the screen
ppl - perform peak picking in predefined regions
pph - perform peak picking and also show an intensity histogram
ppj - perform peak picking and store peaks in JCAMP-DX format
pp - open the peak picking dialog box

DESCRIPTION

Peak picking commands can be started from the command line or from the peak picking dialog box.

All peak picking command open the dialog box with the corresponding option selected. The command **pp**, however, selects the last used option.

Auto-Pick peaks on displayed spectrum region

This option selects the command **pps** for execution. It determines all peaks

Peak picking - pps

Options

- ☒ Auto-Pick peaks on displayed spectrum region
- ☐ Define regions / peaks manually, adjust MI, MAXI
- ☐ Auto-Pick peaks in predefined regions (file 'peakrng')
- ☐ Like 1st option, but peak list with histogram
- ☐ Like 1st option, but peak list in JCAMP format
- ☐ Calculate width of currently displayed peak

Required parameters

Left picking limit F1P [ppm] = 15.8339946457606

Right picking limit F2P [ppm] = -6.40097655892449

Intensity of reference peak CY [rel] = -3

Minimum intensity MI [rel] = 0

Maximum intensity MAXI [rel] = 10000

Peak detection sensitivity PC = 1.4

Fraction of peak height for width calc. [0...1] = 0.5

Pick positive / negative peaks PSIGN = pos.

Reference peak selection mode PSCAL = sreg

Region file for PSCAL = sreg/psreg: SREGLST = 13C.CDCI3

OK Cancel Help

within the displayed 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 7.1

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 displayed region as expressed 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 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 7.2 shows all possible values of PSCAL and the corresponding regions. For PSCAL = ireg or pireg, the

PSCAL	Peak used as reference for vertical scaling
<i>global</i>	The highest peak of the entire spectrum.
<i>preg</i>	The highest peak within the plot region.
<i>ireg</i>	The highest peak within the regions specified in the reg file. If it does not exist, <i>global</i> is used.
<i>pireg</i>	as <i>ireg</i> , but the peak must also lie within the plot region.
<i>sreg</i>	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, <i>global</i> is used.
<i>psreg</i>	as <i>sreg</i> but the peak must also lie within the plot region.
<i>noise</i>	The intensity height of the noise of the spectrum.

Table 7.2

reg file is interpreted. To create a reg file click  to switch to integration mode, click  and select *Save regions to 'reg'*. The reg file 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.CDCI3. For most common nucleus/solvent combinations, a scaling region file is delivered with TOPSPIN. They can be viewed or edited with **edlist sc1**. 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.

pps 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

Auto-Pick peaks in predefined regions (file 'peakrng')

This option selects the command **pp1** for execution. It pick the peaks in predefined regions. To define those regions:

- click **Define regions/peaks manually** in the peaks dialog box or click the  button in the toolbar to switch to peak picking mode
- click the  button and drag the cursor inside the data window to defined the regions
- right-click inside the data window and select Pick Peaks on ranges or enter ppl on the command line

Like 1st option but peak list with histogram

This option selects the command **pph** for execution. It works like **pps** except that it also shows an intensity histogram. This allows you to get a quick overview over the intensity distribution.

Like 1st option but peak in JCAMP format

This option selects the command **ppj** for execution. It works like **pps** 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 7.3

The **pp** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

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 **edp** or by typing *f1p*, *f2p* etc.:

F1P - low field (left) plot region in ppm

F2P - high field (right) plot region in ppm

INPUT FILES

<dir>/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

<tshome>/exp/stan/nmr/lists/scl/

<SREGLST> - regions containing the reference peak if PSCAL = *sreg/psreg*

OUTPUT FILES

<dir>/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

peakhist.txt - peak list with histogram, created by **pph**

pp.dx - peak list in JCAMP-DX format created by **ppj**

USAGE IN AU PROGRAMS

PPS

PPL

PPH

PPJ

SEE ALSO

peakw, ppp, lipp, lippf

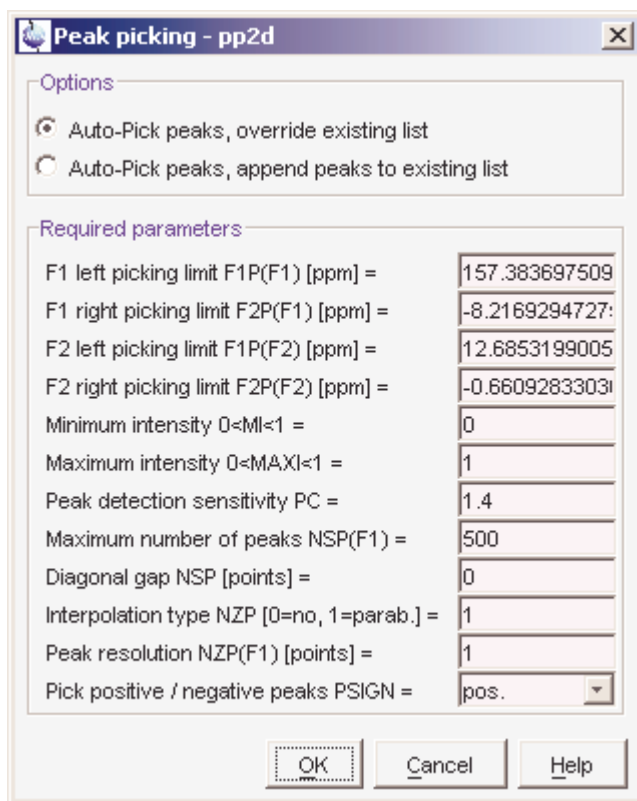
pp2d, pp

NAME

pp2d - perform 2D peak picking
pp - open the peak picking dialog box

DESCRIPTION

2D peak picking can be started from the command line or from the peak picking dialog box. The latter can be opened with the command **pp**.



This dialog box has two options, each of which selects a certain command for execution.

Auto-Pick peaks, overwrite existing list

This option selects the command **pp2d** for execution. It performs peak picking according to the parameters in the dialog box. **pp2d** overwrites a possibly existing peak list.

Auto-Pick peaks, append peaks to existing list

This option selects the command **pp2d append** for execution. It performs peak picking according to the parameters in the dialog box. **pp2d append** appends the found peaks to a possibly existing peak list.

The found peaks are marked in the data window.

The **pp** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **pp** dialog box:

F1P - low field (left) limit of the peak picking region in F2 and F1

F2P - high field (left) limit of the deconvolution region F2 and F1

MI - minimum relative intensity (cm)

MAXI - maximum relative intensity (cm)

PC - peak picking sensitivity

PSIGN - peak sign (pos, neg, or both)

INPUT FILES

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

2rr - real processed 2D data

proc - F2 processing parameters

OUTPUT FILES

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

peak.txt - 2D peak list

SEE ALSO

pp3d, pps, ppl, pph, ppj

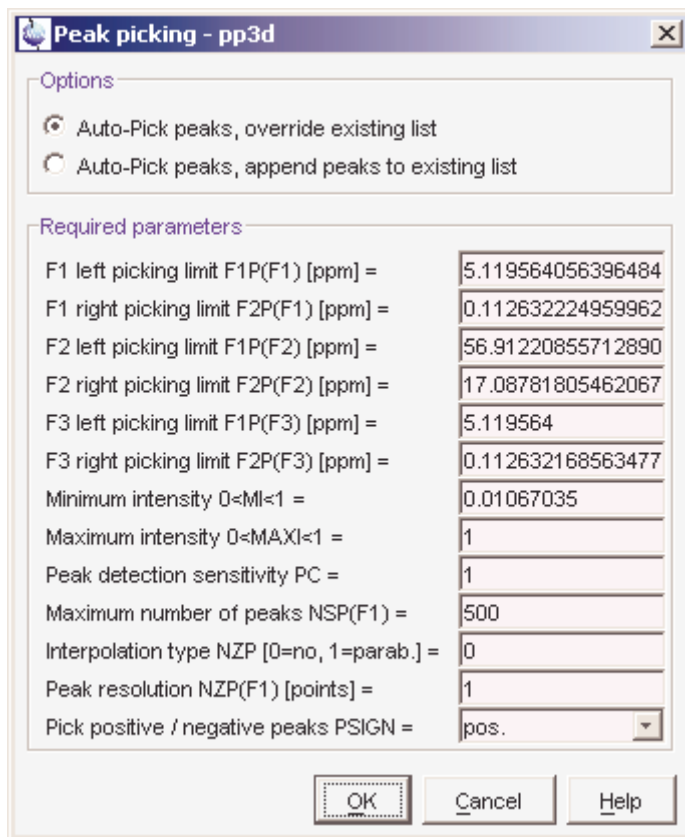
pp3d, pp

NAME

pp3d - perform 3D peak picking
pp - open the peak picking dialog box

DESCRIPTION

3D peak picking can be started from the command line or from the peak picking dialog box. The latter can be opened with the command **pp**.



This dialog box has two options, each of which selects a certain command for execution.

Auto-Pick peaks, overwrite existing list

This option selects the command **pp3d** for execution. It performs peak picking according to the parameters in the dialog box. **pp3d** overwrites a possibly existing peak list.

Auto-Pick peaks, append peaks to existing list

This option selects the command **pp3d append** for execution. It performs peak picking according to the parameters in the dialog box. **pp3d append** appends the found peaks to a possibly existing peak list.

The found peaks are marked in the data window as long as it is in 2D image or 2D contour display mode.

The **pp** command can be used on 1D, 2D or 3D data. It recognizes the data dimensionality and opens a dialog box with the appropriate options and parameters.

INPUT PARAMETERS

set from the **pp** dialog box:

F1P - low field (left) limit of the peak picking region in F3, F2 and F1
F2P - high field (left) limit of the deconvolution region in F3, F2 and F1
MI - minimum relative intensity (cm)
MAXI - maximum relative intensity (cm)
PC - peak picking sensitivity
PSIGN - peak sign (pos, neg, or both)

INPUT FILES

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

3rrr - real processed 2D data
proc - F3 processing parameters

OUTPUT FILES

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

peak.txt - 3D peak list

SEE ALSO

pp2d, pps, ppl, pph, ppj

saguide

NAME

saguide - open the solids analysis guide

DESCRIPTION

The command **saguide** opens a dialog box with a workflow for Solids Line Shape Analysis. This procedure is completely described in the TOPSPIN Users Guide. To open this:

click *Help* → *Users Guide*

SEE ALSO

sola

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 processing 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:

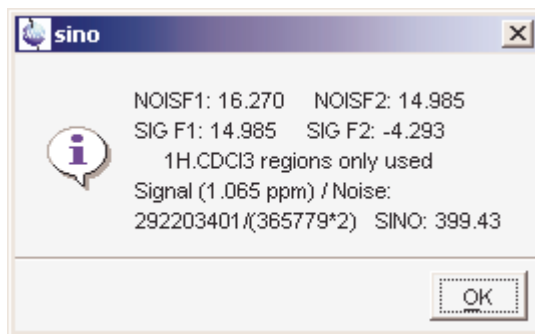
where N is the total number of points in the noise region, n = (N-1)/2, and y(i) is the nth point in the noise region. The limits of the noise region is determined by the processing parameters NOISF1 and NOISF2. If they are equal, the first 1/16th of the spectrum is used as the noise region.

The parameters SIGF1, SIGF2, NOISF1 and NOISF2 can be set from the command line, from the **Procpars** tab (command **edp**) or, interactively, in Signal/Noise display mode. The latter can be entered by clicking **Analysis** → **Signal/Noise Calculation** or by entering **.sino** on the command line.

$$noise = \sqrt{\frac{\sum_{i=-n}^n y(i)^2 - \frac{1}{N} \left(\left(\sum_{i=-n}^n y(i) \right)^2 + \frac{3 \cdot \left(\sum_{i=1}^n i(y(i) - y(-i)) \right)^2}{N^2 - 1} \right)}{N - 1}}$$

sino internally performs a peak picking to determine the highest peak in the signal region.

The result of **sino** appears on the screen, for example:



sino noprint does not show the result on the screen. 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 **s sino** or **dpp**.

sino real skips the magnitude calculation and works on the real data. Note that **sino** without argument first performs a magnitude calculation and then calculates the signal to noise ratio on the magnitude data.

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 store 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 in *.sino* display mode, with *edp* or by typing *noisf1, noise2* 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 *s nuc1* etc.:

NUC1 - observe nucleus

SOLVENT - sample solvent

OUTPUT PARAMETERS

can be viewed with *dpp* or by typing *s sino* :

SINO - signal to noise ratio

INPUT FILES

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

1r - real processed 1D data

1i - imaginary processed data (not used for *sino real*)

proc - processing parameters

<tshome>/exp/stan/nmr/lists/scl/

<NUC1.SOLVENT> - scaling region file

OUTPUT FILES

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

procs - processing status parameters

USAGE IN AU PROGRAMS

SINO

SEE ALSO

mc, abs

sola

NAME

sola- switch to solids lineshape analysis mode

DESCRIPTION

The command **sola** switches to solids lineshape analysis mode. This procedure is completely described in the TOPSPIN Users Guide. To open this:

click *Help* → *Users Guide*

SEE ALSO

saguide

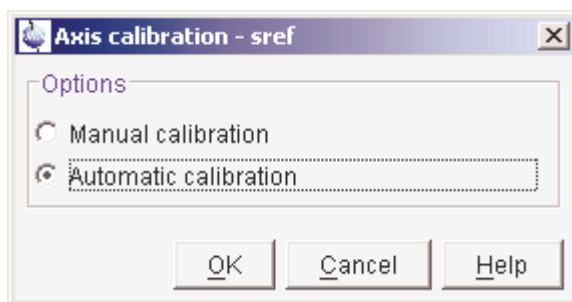
sref, cal

NAME

sref - calibrate the spectrum; set the TMS signal to 0 ppm
cal - open the calibration dialog box


DESCRIPTION

Spectrum calibration can be started from the command line with **sref** or from the calibration dialog box which is opened with the **cal** command.



This dialog box offers two options, one for manual and one for automatic calibration.

Manual calibration

This option selects the **.cal** command for execution. This is equivalent to clicking the  button in the toolbar and switches to interactive calibration mode. Click inside the data window at the reference peak, enter the frequency value in the appearing dialog box and click **OK**.

Automatic calibration

This option selects the **sref** command for execution. It 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 TOPSPIN 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 , ^{13}C and ^{31}P spectrum with C_6D_6 as a solvent. Table 7.4 shows the corresponding entry in the **edlock** table:

Solvent	Field	Lock power	Nucleus	Distance [ppm]	Ref. [ppm]	Width [ppm]	RShift [ppm]
C_6D_6	-150	-15.0					
			^1H	7.28	0.0	0.5	0.000
			^2H	7.28	0.0	0.5	0.000
			^{13}C	128.0	0.0	5.0	0.220
			^{31}P	0.00	10.5	5.0	13.356

Table 7.4

case #1 - calibration of a ^1H spectrum: A spectrum was acquired while being locked on C_6D_6 . **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 ^{13}C spectrum: A spectrum was acquired while being locked on C_6D_6 . **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 ^{31}P spectrum: A spectrum was acquired while being locked on C_6D_6 . **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 **s 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

<tshome>/conf/instr/<instrum>/

2Hlock - edlock table for 2H locked samples

19Flock - edlock table for 19F locked samples

OUTPUT FILES

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

proc - processing parameters

procs - processing status parameters

t1guide

NAME

t1guide - open the relaxation analysis guide

DESCRIPTION

The command **t1guide** opens a dialog box with a workflow for relaxation analysis including T1/T2. This procedure is completely described in the TOPSPIN Users Guide. To open this:

click *Help* → *Users Guide*

Chapter 8

Dataset handling

This chapter describes all TOPSPIN commands which can be used to read or write or delete datasets.

copy

NAME

copy - copy the data path of the current dataset to the Clipboard

DESCRIPTION

The command ***copy*** copies the data path of the current dataset to the clipboard. This corresponding dataset can then be opened in a different window with the command ***paste***.

Entering ***copy*** on the command line is equivalent to clicking ***File*** → ***Copy*** in the menu.

SEE ALSO

paste

del, dela, delp, deldat, delete

NAME

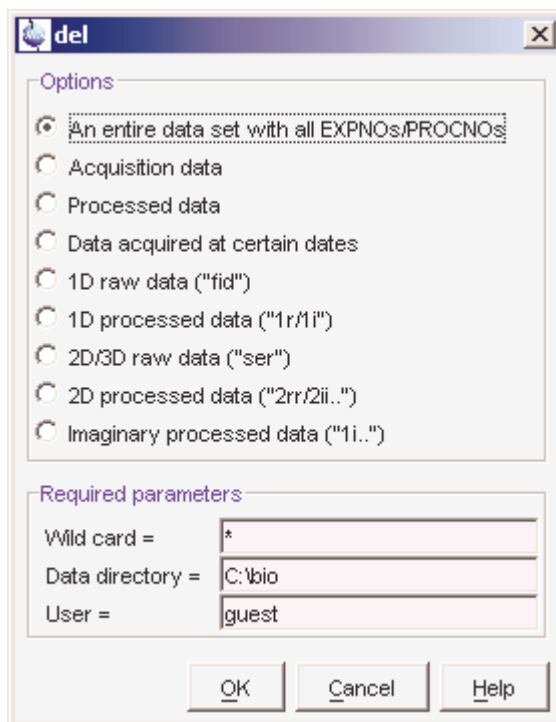
del - delete data
dela - delete acquisition data (raw data)
delp - delete processed data
deldat - delete data acquired at certain dates
delete - open the delete dialog box

SYNTAX

del* [<name>]

DESCRIPTION

Delete commands can be started from the command line or from the delete dialog box. The latter is opened with the command **delete** :



This dialog box has several options, each of which selects a certain command for execution.

The commands **del**, **dela**, **delp** and **deldat** allow you to display a list of datasets. Such a list 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 **Delete...** to actually delete them.

An entire dataset with all EXPNOs/PROCNOs

This option selects the command **del** for execution. It lists datasets, only showing the dataset name. To delete data, mark one or more datasets and click **Delete**. The marked datasets are entirely deleted, including data files, parameter files and the data name directory.

Acquisition data

This option selects the command **dela** for execution. It lists datasets showing a separate entry for each experiment number (expno). Each entry shows the dataset *name*, *expno*, *file* and *size*. Datasets which do not contain raw data are displayed with the *file* entry "no raw data". To delete data, mark one or more datasets and click one of the following buttons:

- **Delete selected EXPNO(s)** to delete the expno directory
- **Delete selected FILE(S)** to delete the raw data files only

Processed data

This option selects the command **delp** for execution. It lists datasets showing a separate entry for each processed data number (procno). Each entry shows the dataset *name*, *expno*, *procno*, *files* and *size*. Datasets which do not contain processed data are displayed with in the *files* entry "no processed data". To delete data, mark one or more datasets and click one of the following buttons:

- **Delete selected PROCNO(s)** to delete the procno directories
- **Delete selected FILE(S)** to delete the processed data files only

Data acquired at certain dates

This option selects the command **deldat** for execution. It prompts the user

for a time range as specified in table 8.1. Depending on the time range you

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

select, you are further prompted for one or two specific dates. A list of datasets which were measured within the specified time range is displayed with a separate entry for each experiment number (expno).

When started from the command line, **del*** commands can take one argument which may contain wild cards. Examples:

del *exam1d**

list all datasets whose name starts with *exam1d*

del *exam1d???*

list all datasets whose name is *exam1d* plus three extra characters

del* commands only list and delete the datasets of current user. The current user here refers to the *user* part of the data path of the currently selected dataset.

Please distinguish:

- the user part of the data path
- the owner of the dataset
- the user who runs TOPSPIN

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 TOPSPIN 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.

OUTPUT FILES

For *de1a* → *Delete selected FILES(S)*:

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

audita.txt - acquisition audit trail

For *de1p* → *Delete selected FILES(S)*:

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

auditp.txt - processing audit trail

SEE ALSO

delf, dels, delser, del2d, deli

delf, dels, delser, del2d, deli, delete

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
delete - open the delete dialog box

SYNTAX

del* [<name>]

DESCRIPTION

Delete commands can be started from the command line or from the delete dialog box. The latter is opened with the command **delete**

This dialog box has several options, each of which selects a certain command for execution.

The commands **delf**, **dels**, **delser**, **del2d** and **deli** display a list of datasets. Such a 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 one or more datasets to mark them for deletion and then click **Delete..** to actually delete them.

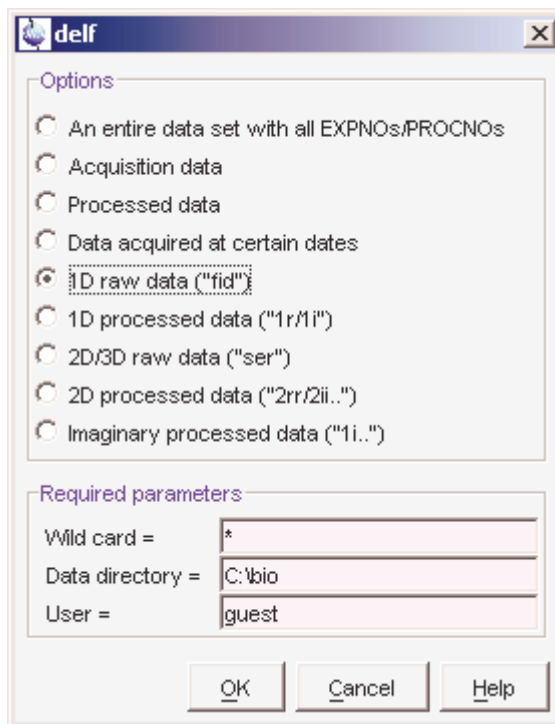
1D raw data

This option selects the command **delf** for execution. It lists 1D datasets which contain raw data showing a separate entry for each experiment number (expno). Each entry shows the dataset *name*, *expno*, *file* and *size*. To delete data, mark one or more datasets and click one of the following buttons:

- **Delete selected EXPNO(s)** to delete the expno directory
- **Delete selected FILE(S)** to delete the raw data files only

1D processed data

This option selects the command **dels** for execution. It lists 1D datasets which contain processed data showing a separate entry for each processed



data number (procno). Each entry contains the dataset *name*, *procno*, *files* and *size*. To delete data, mark one or more datasets and click one of the following buttons:

- **Delete selected PROCNO(s)** to delete the procno directories
- **Delete selected FILE(S)** to delete the processed data files only

2D/3D raw data

This option selects the command **delser** for execution. It lists 2D and 3D datasets which contain raw data showing a separate entry for each experiment number (expno). Each entry shows the dataset *name*, *expno*, *file* and *size*. To delete data, mark one or more datasets and click one of the following buttons:

- **Delete selected EXPNO(s)** to delete the expno directory
- **Delete selected FILE(S)** to delete the raw data files only

2D processed data

This option selects the command **del2d** for execution. It lists 2D datasets which contain processed data showing a separate entry for each processed data number (procno). Each entry shows the dataset *name*, *procno*, *files* and *size*. To delete data, mark one or more datasets and click one of the following buttons:

- **Delete selected PROCNO(s)** to delete the procno directories
- **Delete selected FILE(S)** to delete the processed data files only

Imaginary processed data

This option selects the command **deli** for execution. It lists datasets which contain 1D, 2D or 3D imaginary data showing a separate entry for each processed data number (procno). Each entry shows the dataset *name*, *expno*, *procno*, *files* and *size*. Only the imaginary processed data files are deleted. Raw data, processed data and parameter files are kept. To delete data, mark one or more datasets and click the button:

- **Delete selected 'i' FILE(S)** to delete the imaginary processed data files

When started from the command line, **del*** commands can take one argument which may contain wild cards. Examples:

del f exam1d*

list all datasets whose name starts with *exam1d*

del f exam1d???

list all datasets whose name is *exam1d* plus three extra characters

del* commands only list and delete the datasets of current user. The current user here refers to the *user* part of the data path of the currently selected dataset. Please distinguish:

- the user part of the data path
- the owner of the dataset
- the user who runs TOPSPIN

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 TOPSPIN 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.

OUTPUT FILES

For *del*/*delser* → Delete selected FILES(S):

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

audita.txt - acquisition audit trail

For *del*s/*del*2d/*del*i → Delete selected FILES(S):

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

auditp.txt - processing audit trail

SEE ALSO

del, dela, delp, deldat

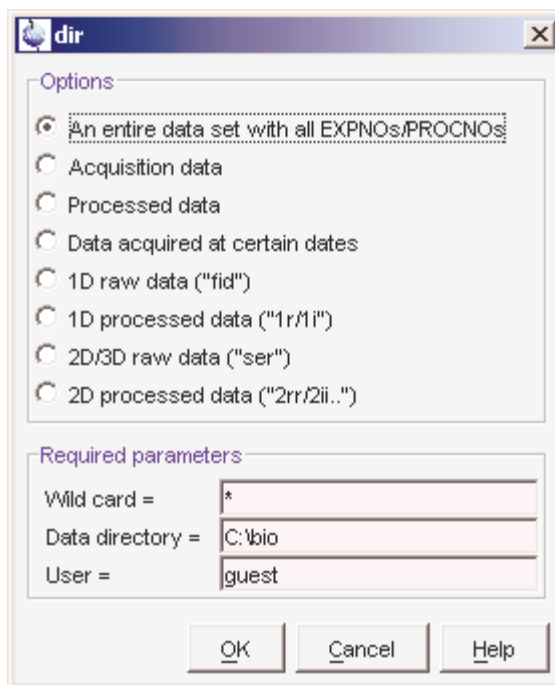
dir, dira, dirp, dirdat, browse

NAME

dir - list data
dira - list acquisition data (raw data)
dirp - list processed data
dirdat - list data acquired at certain dates
browse - open the data directory dialog box

DESCRIPTION

Commands to list data directories can be started from the command line or from the directory dialog box. The latter is opened with the command **browse** :



This dialog box has several options, each of which selects a certain command for execution.

The commands **dir**, **dira**, **dirp** and **dirdat** display all datasets containing

raw and/or processed data as well as empty datasets which only contain parameter files. You can mark one or more entries in the list and click:

- ***Display***
to display the data in the current data window

or

- ***Display in new window***
to display the data in a new data window

When multiple entries were marked, they will be shown in one data window in multi-display mode.

An entire dataset with all EXPNOs/PROCNOs

This option selects the command ***dir*** for execution. It lists datasets, showing the data names only.

Acquisition data

This option selects the command ***dira*** for execution. It lists datasets showing a separate entry for each expno. Each entry shows the dataset *name*, *expno*, *file* and *size*. The entry *file* refers to the data files and can be *fid* (1D raw data), *ser* (2D or 3D raw data) or *no raw data*.

Processed data

This option selects the command ***dirp*** for execution. It lists datasets showing a separate entry for each processed data number (procno). Each entry shows the dataset *name*, *expno*, *procno*, *files* 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*.

Data acquired at certain dates

This option selects the command ***dirdat*** for execution. It prompts the user

for a time range as specified in table 8.2. Depending on the time range you

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

select, you are further prompted for one or two specific dates. A list of datasets which were measured within the specified time range is displayed with a separate entry for each expno.

When started from the command line, **dir*** commands can take one argument which may contain wild cards. Examples:

dir exam1d*

list all datasets whose name starts with *exam1d*

dir exam1d???

list all datasets whose name is *exam1d* plus three extra characters

INPUT FILES

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

fid - 1D raw data

ser - 2D or 3D raw data

<dir>/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, open, find, search, re, rep, rew, repw

dirf, dirs, dirser, dir2d, browse

NAME

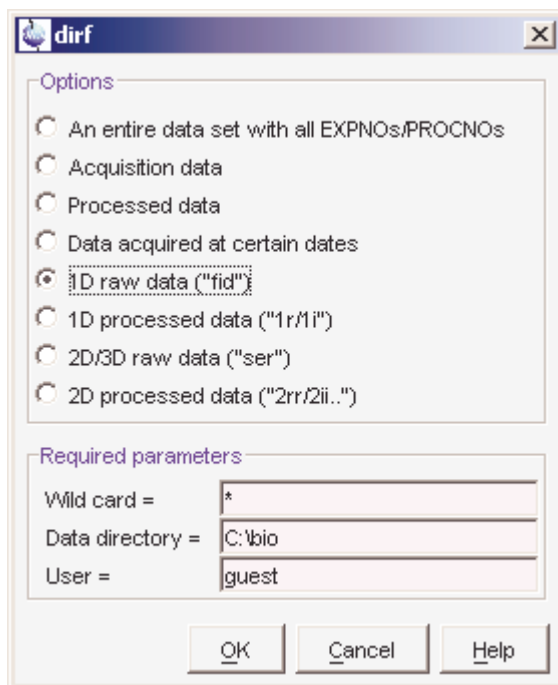
dirf - list FIDs (1D raw data)
dirs - list spectra (1D processed data)
dirser - list serial data (2D and 3D raw data)
dir2d - list 2D processed data
browse - open the dir* dialog box

SYNTAX

dir* [<name>]

DESCRIPTION

The **dir*** commands display a list of datasets according to certain criteria. They can be started from the command line or from the **browse** dialog box.



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. You can mark one or more datasets in the list and click:

- ***Display***
to display the data in the current data window
- or
- ***Display in new window***
to display the data in a new data window

When multiple entries were marked, they will be shown in one data window in multi-display mode.

1D raw data

This option selects the command ***dirf*** for execution. It lists 1D datasets which contain raw data showing a separate entry for each experiment number (expno). Each entry shows the dataset *name*, *expno*, *file* and *size*.

1D processed data

This option selects the command ***dirs*** for execution. It lists 1D datasets which contain processed data showing a separate entry for each processed data number (procno). Each entry shows the dataset *name*, *procno*, *files* and *size*.

2D/3D raw data

This option selects the command ***dirser*** for execution. It lists 2D and 3D datasets which contain raw data showing a separate entry for each experiment number (expno). Each entry shows the dataset *name*, *expno*, *file* and *size*.

2D processed data

This option selects the command ***dir2d*** for execution. It lists 2D datasets which contain processed data showing a separate entry for each processed data number (procno). Each entry shows the dataset *name*, *procno*, *files* and *size*.

INPUT FILES

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

fid - 1D raw data

ser - 2D or 3D raw data

<dir>/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, browse, find, search, re, rep, rew, repw

find, search

NAME

find, search - find TOPSPIN data

DESCRIPTION

The command ***find*** [***search***, ***Ctrl-f***] opens a dialog box:



The image shows a Windows-style dialog box titled "Find data". It contains several input fields and dropdown menus for specifying search criteria. The fields are: NAME (containing "exam"), EXPNO, PROCNO, USER (containing "g"), Title, Pulse Prog., Dimension (a dropdown menu showing "1D+2D+3D"), and Data type (a dropdown menu showing "spectra"). Below these fields is a section labeled "Data directories" containing a list box with three entries: "C:\tsa", "C:\bio" (which is highlighted with a blue background), and "C:\ts1". At the bottom of the dialog are three buttons: "OK", "Reset mask", and "Cancel".

Here you can specify the search criteria and data directories to be searched. Note that all entries will be found that start with the characters specified in the criteria fields. The above search, for example will find all data names that start with "exam" of all users that start with "g". After doing that, you can:

- Click the ***OK*** button to start the actual search

or

- Click the **Reset mask** button to reset the default search criteria.

A search will result in a dialog box that shows a list of found datasets.



The search result can be used in several ways:

To display the properties of one dataset:

1. Click dataset to select it
2. Click **Properties**

To select all found datasets:

Click **Select all**

To display one of the found datasets:

1. Click one dataset to select it

Optionally: click **Properties** to view the datasets properties

2. Click **Display**

to display the selected dataset in the current data window

Note that if the search result consist of only one dataset, this is automatically selected and you can skip step 1

To add the found datasets to the portfolio:

1. Do one of the following:
 - Click one dataset to select it

- Hold the Control key and left-click several datasets to select these datasets
 - Hold the Shift key and left-click two datasets to select these datasets and all datasets in between
 - Click **Select all** to select all datasets in the search result
2. Then do one of the following:
- Click **Add to portfolio** to extend the current portfolio
 - Click **Replace portfolio** to replace the current portfolio

INPUT FILES

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

fid - 1D raw data

acqu - acquisition parameters

acqu - acquisition status parameters

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

1r, 1i - processed 1D data

proc - processing parameters

procs - processing status parameters

Note that these are only the main 1D data files.

SEE ALSO

open, new, re, rep, rew, repw, dir*

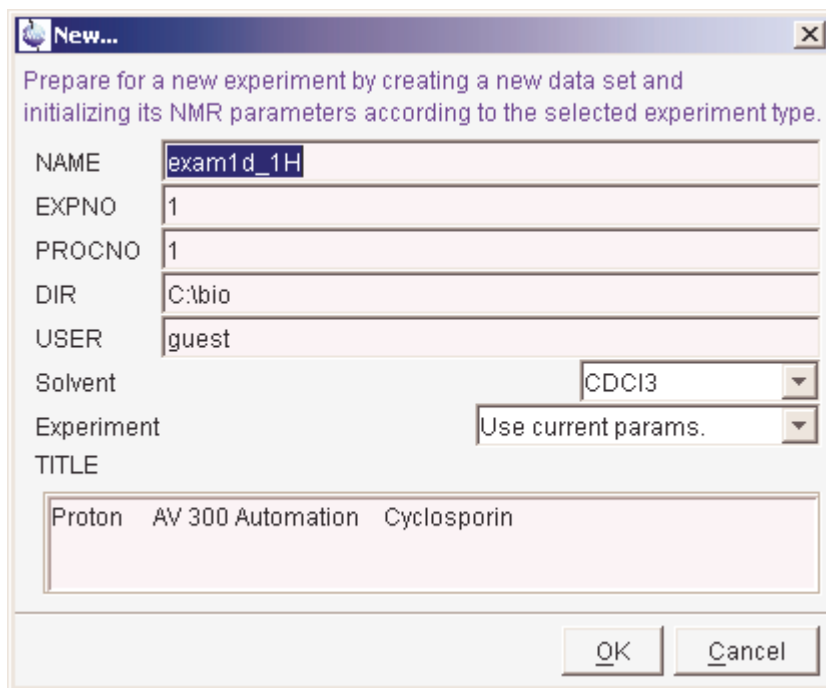
new

NAME

new - define a new dataset

DESCRIPTION

The command **new** [*Ctrl-n*] opens a dialog box in which you can define a new dataset.



The screenshot shows a 'New...' dialog box with the following fields and values:

Field	Value
NAME	exam1d_1H
EXPNO	1
PROCNO	1
DIR	C:\bio
USER	guest
Solvent	CDCl3
Experiment	Use current params.
TITLE	Proton AV 300 Automation Cyclosporin

Buttons: OK, Cancel

Here, you can specify the dataset *name*, *expno*, *procno*, *dir* (disk unit) and *user*. Note that these are all parts of the data pathname:

C:\bio\data\nmr\user\name\expno\data\procno

Furthermore, you can select:

- ***Solvent***
sets the acquisition parameter SOLVENT. Default is the solvent of the current dataset.
- ***Experiment*** (=parameter set)
copies the acquisition and processing parameters. Default is "Use current parameters".

When you click **OK**, the dataset is created and made the current data window. If the specified dataset already exists, you will be prompted to overwrite this or not. Note that this will only overwrite the parameters, not the data files.

new is equivalent to the command **edc**.

INPUT FILES

<tshome>/prog/curdir/<user>/

curdat - current dataset parameters

If ***Experiment = Use current params:***

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

acqu - acquisition parameters

acqu - acquisition status parameters

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

proc - processing parameters

procs - processing status parameters

If ***Experiment ≠ Use current params.:***

<tshome>/exp/stan/nmr/par/<experiment>/

acqu - acquisition parameters

proc - processing parameters

OUTPUT FILES

<tshome>/prog/curdir/<user>/

curdat - current dataset parameters

If the dataset specified with **new** does not exist yet, the current dataset is copied:

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

acqu - acquisition parameters

acqus - acquisition status parameters

<dir>/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

open, find, search, re, rep, rew, repw, dir*

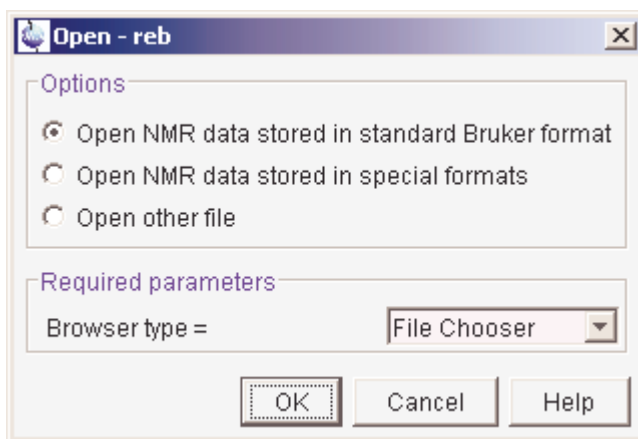
open

NAME

open - open a dataset, pulse program, AU program etc.

DESCRIPTION

Opening data, parameters, lists and various other files can be started from the command line or from the open dialog box. [The latter is opened with the command **open** [*Ctrl-o*].



This dialog box has two options each with several file types. Each file type selects a certain command for execution.

Open NMR data stored in standard Bruker format

This option allows you to open Bruker format data in the following ways:

- File chooser [**reb**]
- RE dialog [**re**]
- PROCNO dialog [**rep**]

Open NMR data stored in special formats

This option allows you to open the following NMR data types (formats):

- JCAMP-DX [**fromjdx**]

- Zipped TOPSPIN [*fromzip*]
- WIN-NMR [*winconv*]
- A3000 [*conv*]
- VNMR [*vconv*]
- JNMR [*jconv*]

Open other file:

This option allows you to open the following lists and programs:

- Pulse programs [*edpul*]
- Au programs [*edau*]
- Gradient programs [*edgrad*]
- CPD programs [*edcpd*]
- Miscellaneous files [*edmisc*]
- Parameter lists [*edlist*]
- Python program [*edpy*]

The corresponding command line commands are specified in square brackets.

After clicking **OK**, a new dialog box will appear according to the selected option and file type.

SEE ALSO

re, rep, fromjdx, fromzip, winconv, conv, vconv, jconv, edpul, edau, edgrad, edcpd, edmisc, edlist, edmac

paste

NAME

paste - open the dataset that was previously copied

DESCRIPTION

The command **paste** opens the dataset which was previously copied from the a TOPSPIN data window or from the Windows Explorer. This involves two steps:

1. Copy

In the Windows 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 and click **Copy**

or

In TOPSPIN:

- Select the source data window
- Click **File** → **Copy** or type **copy**

2. Paste

In TOPSPIN:

- Click **File** → **Paste** or type **paste**

Note that if you select and copy a the dataset in the Windows 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, TOPSPIN tries to read whatever is currently stored in the Clipboard. If that is a data path, TOPSPIN will read it, otherwise you will get an error message.

INPUT FILES

<dir>/data/<user>/<name1D>/nmr/<expno>/

fid - 1D raw data

<dir>/data/<user>/<name1D>/nmr/<expno>/pdata/<procno>/

1r, 1i - processed 1D data

<dir>/data/<user>/<name2D>/nmr/<expno>/

ser - 2D raw data

<dir>/data/<user>/<name2D>/nmr/<expno>/pdata/<procno>/

2rr, 2ir, 2ri, 2ii - processed 2D data

clevels - 2D contour levels

Furthermore the parameter files acqu, acqu, proc etc. are read as a part of the entire dataset.

SEE ALSO

copy

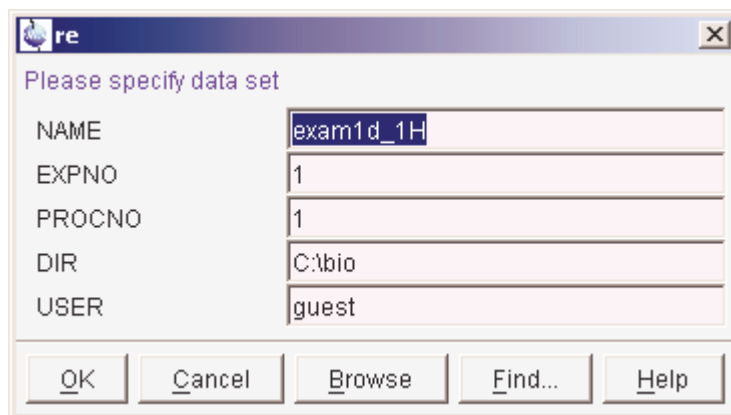
re, rep, rew, repw

NAME

re - read data name and/or experiment number (expno)
rep - read data processed data number (procno)
rew - read data name and/or experiment number (expno) in a new window
repw - read data processed data number (procno) in a new window

DESCRIPTION

The command **re** allows you to read and display a new dataset.



It opens a dialog box where you can specify the data path variables. A full data path is:

`<dir>/data/<user>/nmr/<name>/<expno>/pdata/<procno>`

re replaces the dataset in the current data window (if it exists).

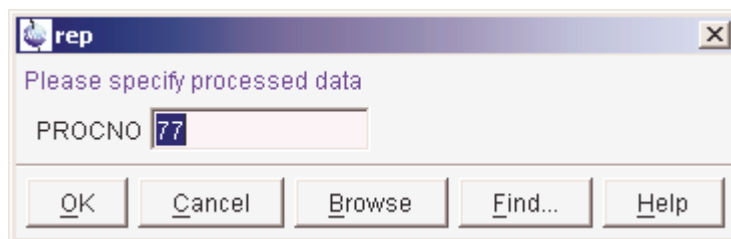
The data path variables can also be specified on the command line. In this case, the dialog box is not opened and the missing data path variables are taken from the current dataset. Examples:

```
re <name>
re <expno>
re <name> <expno>
re <expno> <procno>
```

```
re <name> <expno> <procno>  
re <name> <expno> <procno> <dir> <user>
```

Note that the first alphanumeric argument is always interpreted as the name and the first numeric argument as experiment number.

The command **rep** allows you to read a new processed data number (procno) of the current dataset. .



The destination procno can also be specified on the command line, e.g.:

```
rep 77
```

The commands **rew** and **repw** work like **re** and **rep** respectively, except that they open the specified dataset in a new window rather than in the currently active window.

INPUT FILES

```
<dir>/data/<user>/nmr/<name1D>/<expno>/
```

fid - 1D raw data

acqu - acquisition parameters

acqus - acquisition status parameters

```
<dir>/data/<user>/<name1D>/nmr/<expno>/pdata/<procno>/
```

1r, 1i - processed 1D data

proc - processing parameters

procs - processing status parameters

Note that these are only the main files of a 1D dataset.

SEE ALSO

open, dir*, new, find

reopen

NAME

reopen - reopen the current dataset in a new data window

DESCRIPTION

The command ***reopen*** reopens the current dataset in a new data window. This is, for example, convenient to view various regions or various objects (spectrum, fid, parameters etc.) of the same dataset. Multiple data windows are indicated with a number in square brackets, e.g. [1], in the title bar:

Entering ***reopen*** on the command line is equivalent to clicking ***File*** → ***Reopen*** in the menu.

SEE ALSO

open

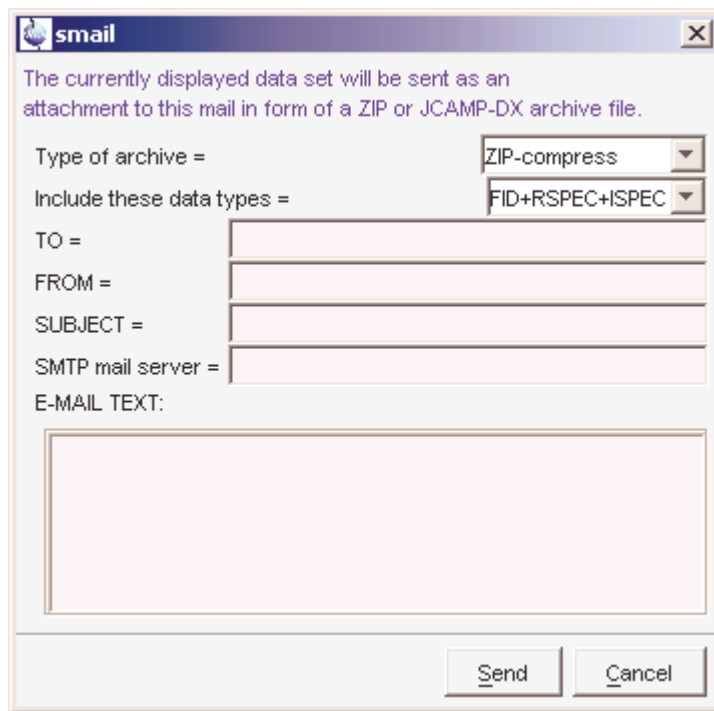
smai

NAME

smai - send the current dataset by Email

DESCRIPTION

The command **smai** sends the current dataset by Email. It opens a dialog box where you can specify the required information or accept the default values.



In the dialog box, you must select the:

- Archive type: ZIP or JCAMP
- Data type(s) included: FID, Spectrum and/or Parameters

For ZIP format data you can choose between compression and no compression.

For JCAMP format, you can choose between the following compression mode:

- *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*)

For the included data types, you have the following choices:

- FID+RSPEC+ISPEC: raw + real and imaginary processed data
- FID+RSPEC: raw + real processed data
- FID: raw data
- RSPEC+ISPEC: real and imaginary processed data
- RSPEC: real processed data
- PARAMS: parameter files

Before you can send the data you must fill in the fields:

- **To:** the email address of the recipient
- **From:** you own email address
- **SMTP mail server:**
- **Subject:**
- **Text:**

INPUT FILES

<tshome>/prog/curdir/<user>/

curdat - current data parameters

If data type includes FID :

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

fid - 1D raw data

ser - 2D raw data

If data type includes RSPEC :

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

1r - real processed 1D data
2rr - real processed 2D data

If data type includes ISPEC :

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

1i - imaginary processed 1D data
2ir - F2-imaginary processed 2D data
2ri - F1-imaginary processed 2D data
2ii - F2/F1-imaginary processed 2D data

All other files which are part of a dataset like parameter files, audit trails files etc. are sent for all data types.

OUTPUT FILES

<userhome>/<mydata.dx> - TOPSPIN data in JCAMP-DX format

<userhome>/<mydata.bzip> - TOPSPIN data in ZIP format

SEE ALSO

tojdx, tozip

wrpa, wra, wrp

NAME

wrpa - copy a dataset
wra - copy raw data
wrp - copy processed data

DESCRIPTION

The command **wrpa** writes (copies) a dataset. It opens a dialog box where you can specify the destination dataset:



When you click **OK**, 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
- <dir> - 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 path¹, except for the last one

1. The data path of the foreground dataset is displayed above the TOPSPIN data field

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>
wrpa <expno>
wrpa <name> <expno>
wrpa <name> <expno> <procno>
wrpa <name> <expno> <procno> <dir> <user> y
```

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>
```

Note that **wrpa**, **wra** and **wrp** only work if user who started TOPSPIN has the permission to create the destination dataset.

INPUT AND OUTPUT FILES

For **wrpa** and **wra**:

```
<dir>/data/<user>/nmr/<name>/<expno>/
fid - 1D raw data
acqu - acquisition parameters
acqu - acquisition status parameters
```

For **wrpa**, **wra** and **wrp**:

```
<dir>/data/<user>/<name>/nmr/<expno>/pdata/<procno>/
1r, 1i - processed 1D data
```

proc - processing parameters
procs - processing status parameters
auditp.txt - processing audit trail

Note that apart from data and parameters several other files are copied.

USAGE IN AU PROGRAMS

WRA(expno)

WRP(procno)

WRPA(name, expno, procno, diskunit, user)

Note that these macros overwrite possibly exiting data.

SEE ALSO

new, open, re, rep, rew, repw, dir*

Chapter 9

Parameters, lists, AU programs

This chapter describes all TOPSPIN 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 TOPSPIN

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

<tshome>/exp/stan/nmr/au/src/*

AU programs (source files)

OUTPUT FILES

<tshome>/prog/au/bin/*

AU programs (executable files)

SEE ALSO

expinstall, cplbruk, cpluser, edau, xau, xaua, xaup, delau

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 - lists Bruker AU programs; double-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*.

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

<tshome>/exp/stan/nmr/au/src/*

AU programs (source files)

OUTPUT FILES

<tshome>/prog/au/bin/*

AU programs (executable files)

SEE ALSO

expinstall, compileall, edau, xau, xaua, xaup, delau

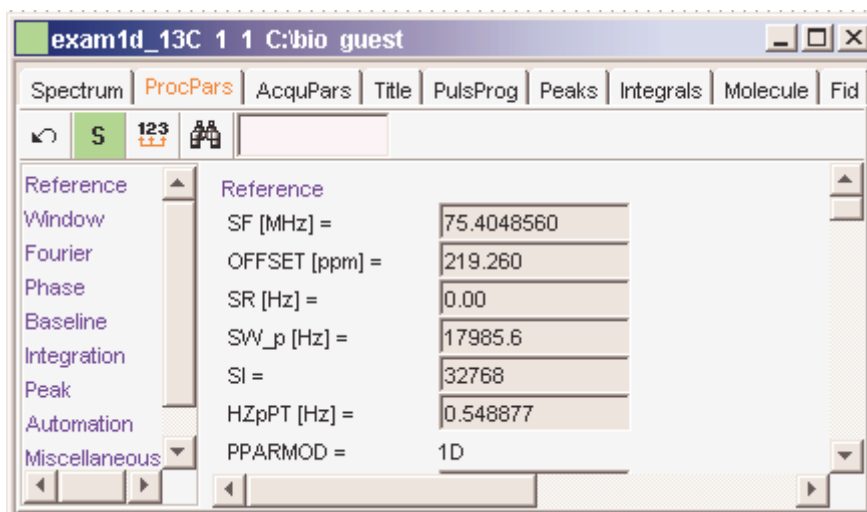
dpp

NAME

dpp - display the processing status parameters



DESCRIPTION

The command **dpp** displays the processing status parameters. Entering **dpp** is equivalent to clicking the **ProcPars** tab in the data window and clicking the **S** button.



The processing status parameters are set by processing commands and represent the status of the processed data. As such, they can only be viewed in the **dpp** window.

The following buttons are available:

-  Undo the last modification (unused for status parameters)
- S** Switch between processing and processing status parameters
-  Change raw dataset dimensionality (parameter PARMODE)



Search for the parameter specified in the search field

Processing status parameters can also be viewed by entering their names on the command line. For example:

s ft_mod

display the processing status parameter FT_mod

s nc_proc

display the processing status parameter NC_proc

INPUT FILES

<tshome>/classes/prop/

pared.prop - parameter properties file

<tshome>/exp/stan/nmr/form/

proc.prop - processing parameter format file

<dir>/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, dpa

edau, xau, delau

NAME

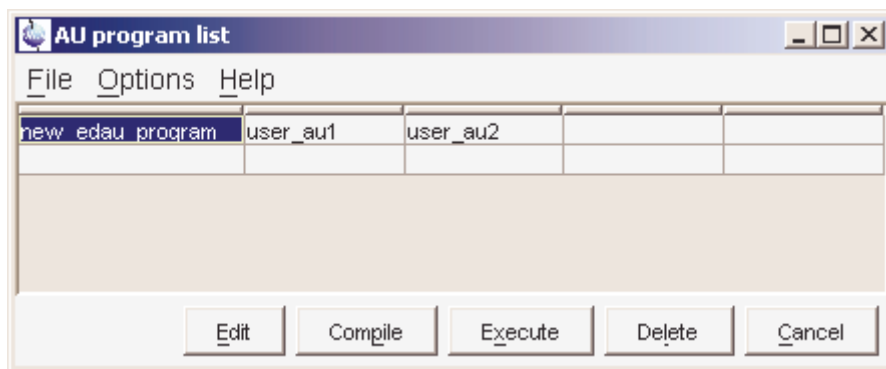
edau - edit an AU program
xau - execute an AU program
delau - delete an AU program

SYNTAX

edau [<name>]
xau [<name>]
delau [<name>]

DESCRIPTION

When entered without arguments, the AU program commands **edau**, **xau** and **delau** all open the AU program dialog box:



Depending on the currently selected option, the Bruker and/or User AU programs are listed. The dialog box allows you to edit, compile, execute and delete AU programs, as well as copy AU programs or create new ones.

Edit the selected AU program

Click **Edit**

Equivalent to double-clicking the AU program name, clicking **File** → **Edit** or entering **edau <name>** on the command line.

Compile the selected AU program

Click *Compile*

Equivalent to clicking *File* → *Compile* or entering **cplbruk** <name> on the command line.

Execute the selected AU program

Click *Execute*

Equivalent to entering <name> or **xau** <name> on the command line.

Delete the selected AU program

Click *Delete*

Equivalent to clicking *File* → *Delete* or entering **delau** <name> on the command line. Note that both the source and binary AU program are deleted.

List all Bruker AU programs:

Click *Options* → *Bruker defined*

List all User AU programs:

Click *Options* → *User defined*

Show the AU program comment lines (short descriptions):

Click *Options* → *Comment on/off*

When you edit a Bruker AU program, it is shown in view mode which means it cannot be modified. However, if you click *Save as..* and store it under a different name, the stored file is automatically opened in edit mode. When you edit a User defined AU program, it is opened in edit mode and can be modified. Whenever you close the AU program view or edit window with *Save* or *Cancel*, you are prompted to *Compile* the AU program. If you don't, you can compile the AU program in a separate step, e.g. with **cplbruk**.

When **edau** is entered on the command line with an argument, the corresponding AU program will be opened. If it does not exist it will be created. If the argument contains wildcards, the AU dialog box is opened showing the matching AU programs. For example, **edau a*** displays 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 TOPSPIN is installed.

edau uses the editor which is defined in the TOPSPIN User Preferences. To change it, enter **set**, click *Miscellaneous* and select or change the editor.

AU programs are usually executed simply by entering their names. The command **xau** is only needed in three cases:

- the AU program has not been compiled yet
- an TOPSPIN command with the same name exists
- to call an Au program from another AU program (using the macro XAU)

AU programs run in background and several of them can run simultaneously. 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 AU reference manual (available as TOPSPIN online help).

INPUT FILES

```
<tshome>/exp/stan/nmr/au/src/*
```

AU program source files

OUTPUT FILES

```
<tshome>/exp/stan/nmr/au/src/*
```

AU program source files

```
<tshome>/prog/au/bin/*
```

AU program executable binary files

SEE ALSO

expinstall, comepileall, cpluser

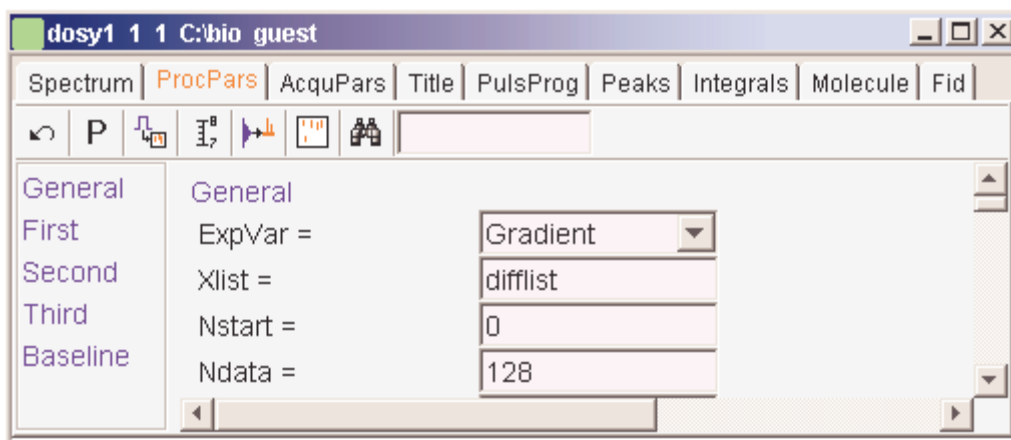
eddosy

NAME

eddosy - edit DOSY processing parameters







DESCRIPTION

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 following buttons are available:

-  Undo the last modification. Can be used repeatedly.
-  Switch to processing parameters
-  Copy parameters from experiment (AU program **setdiffparm**)
-  Get display limits from dataset
-  Execute Fourier Transform (command **xf2**)
-  Start fitting



Search for the parameter specified in the search field

For more information on **eddosy**, refer to the manual DOSY under *Help* → *Application Manuals*.

INPUT FILES

<tshome>/exp/stan/nmr/form/

dosy.e - format file for **eddosy**

INPUT AND OUTPUT FILES

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

dosy - DOSY processing parameters

SEE ALSO

dosy2d, dosy3d

edmac, xmac, delmac

NAME

edmac - edit a macro
xmac - execute a macro
delmac - delete a macro

SYNTAX

edmac [<name>]
xmac [<name>]
delmac [<name>]

DESCRIPTION

The macro commands ***edmac***, ***xmac*** and ***delmac*** allow you to edit or create, execute and delete a TOPSPIN macro, respectively.

Macros are text files which contain a sequence of TOPSPIN commands as they would be entered on the command line. 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.

When entered on the command line, all three ****mac*** commands open a dialog box which shows all available macros. You can select one to edit, execute or delete it. Alternatively, you can specify the macro name as an argument on the command line:

edmac <name> - edit the macro <name>
xmac <name> - execute the macro <name>
<name> - execute the macro <name>
delmac <name> - delete the macro <name>

As opposed to AU programs, only a few macros are delivered by Bruker.

If you enter the command with an argument, this may contain wildcards, e.g. **edmac a*** displays a list of all macros which start with *a*.

INPUT AND OUTPUT FILES

<tshome>/exp/stan/nmr/lists/mac/*

TOPSPIN macros

SEE ALSO

edau, xau, delau, edpy, xpy

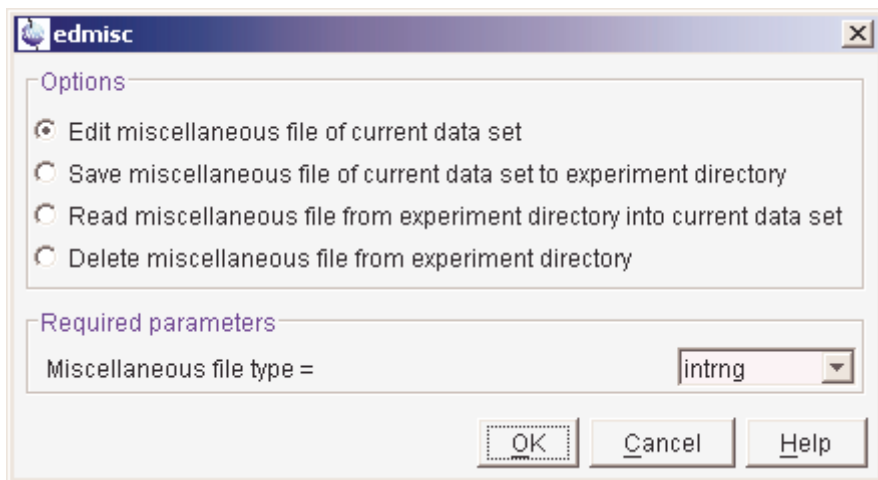
edmisc, rmisc, wmisc, delmisc

NAME

edmisc - edit miscellaneous lists
rmisc - read miscellaneous lists
wmisc - write miscellaneous lists
delmisc - delete miscellaneous lists

DESCRIPTION

The commands **misc* allow you to read, edit, write or delete miscellaneous lists. When entered without arguments, they all open the same dialog box.



This dialog box offers several options, each of which selects a certain command for execution.

Edit miscellaneous file of current data set

This option selects the command *edmisc* for execution. It allows you to edit

files in the current dataset. The lists which can be edited are shown in table

list type	contains
intrng	integral regions, created by interactive integration or automatic baseline correction (<i>abs</i>). Used for spectrum display, print and integral listing.
base_info	<i>polynomial</i> , <i>sine</i> or <i>exponential</i> baseline function, created from the baseline mode (<i>.bas1</i>). Used by the baseline correction command <i>bcm</i> .
baslpnts	baseline points created by <i>def-pts</i> from the baseline mode (<i>.bas1</i>). Used by the spline baseline correction command <i>sab</i> .
peaklist	peak information, created by the command <i>ppp</i> and <i>mdcon auto</i> . Used by the mixed deconvolution command <i>mdcon</i> .
reg	plot regions, created from the integration menu (command <i>integ</i>). Used by <i>pp</i> , <i>lipp</i> when PSCAL=ireg or pireg.

Table 9.1 Miscellaneous list types

9.1.

Save miscellaneous file of current data set to experiment directory

This option selects the command *wmisc* for execution. It allows you to write miscellaneous files for general usage.

Read miscellaneous file from experiment directory into current data set

This option selects the command *rmisc* for execution. It allows you to read files which have been stored with *wmisc* to the current dataset.

A typical usage of the **misc* commands for the file type *intrng* is:

1. Perform 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

The selected list is copied to the current dataset.

rmisc takes two arguments and can be used as follows:

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.

INPUT/OUTPUT DIRECTORIES

<tshome>/exp/stan/nmr/lists

intrng - integral range files

baslpnts - spline baseline points file

base_info - pol. exp. or sine baseline function files

peaklist - peak information files

reg - plot region files

<dir>/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

edlist

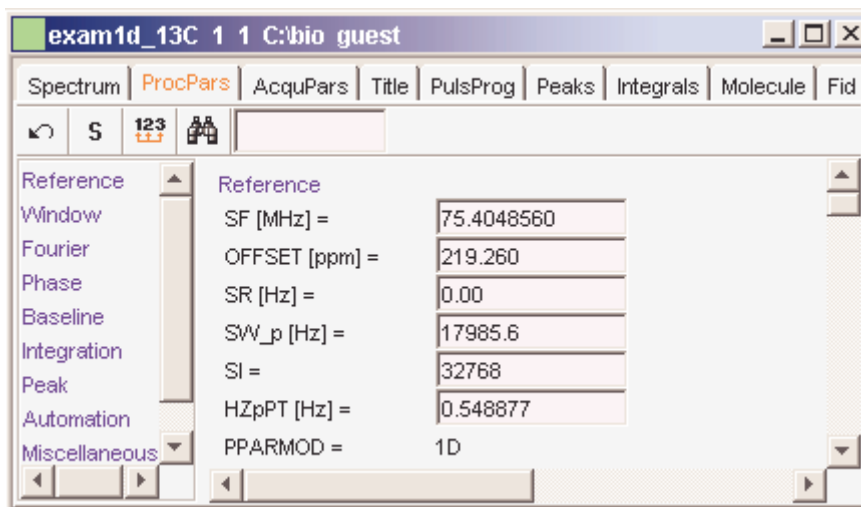
edp

NAME

edp - edit processing parameters





DESCRIPTION

The command **edp** opens a dialog box in which you can set all processing parameters.



Entering **edp** on the command line is equivalent to clicking **ProcPars** in the tab bar of the data window.

The following buttons are available:

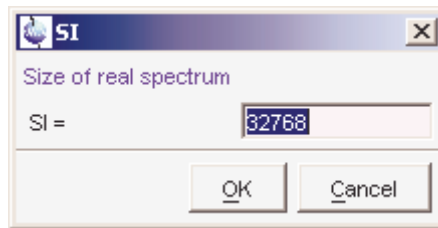
-  Undo the last modification. Can be used repeatedly.
-  Switch to processing status parameters
-  Change raw dataset dimensionality (parameter PPARMOD)
-  Search for the parameter specified in the search field

Inside the parameter editor, you can do the following actions:

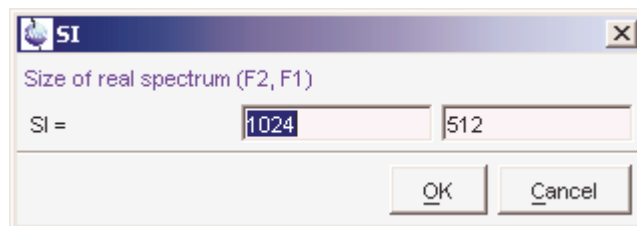
- click a processing step, e.g. **Window** at the left of the dialog box. The step becomes highlighted and the corresponding parameters will appear in the right part of the dialog box.
- click in a parameter field, e.g. **SI** to set the parameter value. It is automatically stored.
- hit the **Tab** key to jump to the next parameter field
- hit **Shift-Tab** to jump to the previous parameter field
- use the scroll bar at the right of the dialog box to move to parameters further up or down in the dialog box.

Note that you can also set parameters by entering there names on the command line. A dialog window will appear where you can enter the parameter value(s). For example:

si
on a 1D dataset.



or on a 2D dataset:



Alternatively, you can specify the parameter value as an argument on the command line, For example;

si 4k
The size will be set to 4k

INPUT AND OUTPUT PARAMETERS

All processing parameters.

INPUT FILES

<tshome>/classes/prop/

pared.prop - parameter properties file

<tshome>/exp/stan/nmr/form/

proc.e - format file for **edp**

<dir>/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

<dir>/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

dpp

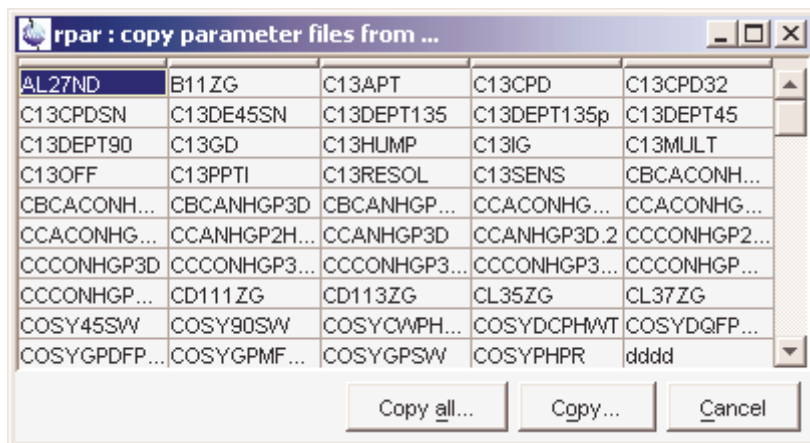
rpar

NAME

rpar - read a parameter set

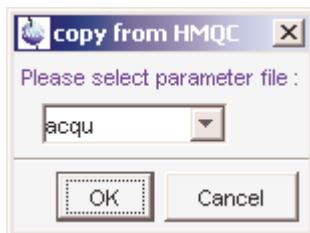
DESCRIPTION

The command **rpar** reads a parameter set (experiment) to the current dataset. When it is entered without arguments, **rpar** opens a dialog box with a list of available parameter sets.



Here you can select a parameter set and click **Copy** or **Copy all** to copy it to the current dataset.

- Clicking **Copy** will open a dialog box where you can select individual parameter files, for example acquisition or processing:



- Clicking **Copy All** results in copying all parameter files

rpar can be used with arguments

- **rpar <name>**
opens a dialog box where you can select individual parameter files of the parameter set <name>. Upon clicking **OK**, this file is copied to the current dataset.
- **rpar <name> all**
reads all parameter files of the parameter set <name> to the current dataset.

The first argument may contain wildcards, e.g.:

rpar C* shows all parameter sets beginning with the letter C

After reading a parameter set with **rpar**, you can modify parameters of the various types with the commands:

- **eda** - acqu parameters
- **edp** - processing 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 TOPSPIN 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> *all*

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

<tsHOME>/exp/stan/nmr/par/<1D parameter set>/

acqu - acquisition parameters

proc - processing parameters

outd - output device parameters

<tsHOME>/exp/stan/nmr/par/<2D parameter set>/

acqu - F2 acquisition parameters

acqu2 - F1 acquisition parameters

proc - F2 processing parameters

proc2 - F1 processing parameters

outd - output device parameters

3D parameter sets also contain the files `acqu3` and `proc3` for the third dimension.

OUTPUT FILES

<dir>/data/<user>/nmr/<1D data name>/<expno>/

acqu - acquisition parameters

<dir>/data/<user>/nmr/<1D data name>/<expno>/pdata/<procno>/

proc - processing parameters

outd - output device parameters

<dir>/data/<user>/nmr/<2D data name>/<expno>/

acqu - F2 acquisition parameters

acqu2 - F1 acquisition parameters

<dir>/data/<user>/nmr/<2D data name>/<expno>/pdata/<procno>/

proc - F2 processing parameters

proc2 - F1 processing parameters
outd - output device parameters

USAGE IN AU PROGRAMS

RPAR(name, type)

SEE ALSO

wpar, delpar, expinstall

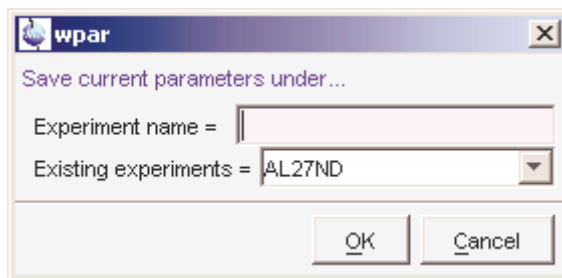
wpar

NAME

wpar - write a parameter set

DESCRIPTION

The command **wpar** stores the parameters of the current dataset in a parameter set. It opens a dialog box where you can enter a new experiment name or select an existing experiment:



Bruker standard experiment names should not be used when storing your own experiments with **wpar**. The reason is that they are overwritten again when a new version of TOPSPIN is installed.

wpar is often used in the following way:

1. Define a new dataset with the command **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. Store the parameters with **wpar** under a new experiment name for general usage.

USAGE IN AU PROGRAMS

WPAR(name, type)

SEE ALSO

rpar, expinstall

Chapter 10

Conversion commands

This chapter describes all TOPSPIN conversion commands. These are commands which convert one data format to another. Described are the conversion of Bruker Aspect 2000/3000 to TOPSPIN, of Varian and Jeol to TOPSPIN, of Avance to AMX, of TOPSPIN to JCAMP-DX and of JCAMP-DX to TOPSPIN.

conv

NAME

conv - convert Aspect 2000/3000 data to TOPSPIN format

SYNTAX

conv [<station> [<filename> | ? | *]]

DESCRIPTION

The command **conv** converts Aspect 2000/3000 data to the TOPSPIN 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 send 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:

/<tshome>/conf/instr/

where <tshome> is the directory where TOPSPIN 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:

<tshome>/conf/instr/<station>/disnmr.conf

For DISMSL data, you can create the `dismsl.conf` file with the TOP-

SPIN 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 directory of the output data (the converted TOPSPIN dataset)

`<station>`

this can be freely chosen, e.g. the name of the source spectrometer

`<user>`

this must be the desired user (USER part of the data directory) of the output data

conv uses the **new** parameters DIR 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 ms11** on the TOPSPIN dataset:

`/x/data/joe/nmr/exam1d/1/pdata/1`

then **conv** searches for input data in the directory:

`/x/bbruknet/ms11/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:

`<tshome>/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 TOPSPIN 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 ms11** on the TOPSPIN 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/ms11/joe
```

If, however, the `destination` file does not exist, **conv** searches for input data in the directory:

```
/x/bruknet/ms11/joe
```

Converting data

conv must be entered on a TOPSPIN dataset where:

- USER corresponds to the `<user>` part of the input directory

- DIR corresponds to the <dir> part of the input directory. This, however, does not count if you work under LINUX 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 TOPSPIN data parameters (command **new**):

- DIR is set to the data directory of the foreground TOPSPIN 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
<tshome>/conf/<station>/disnmr.conf - DISNMR configuration
<tshome>/conf/<station>/dismsl.conf - DISMSL configuration
/usr/local/lib/destination - destination file (UNIX only)

OUTPUT FILES

<dir>/data/<user>/nmr/<name>/<expno>/
 fid - Avance type 1D raw data
 ser - Avance type 2D or 3D raw data
 acqu - acquisition parameters
 acqu2 - acquisition status parameters

<dir>/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

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> <dir> 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. The processed data number (procno) of the new dataset cannot be chosen, it is always set to 1.

INPUT FILES

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

fid - Avance type 1D raw data

ser - Avance type 2D or 3D raw data

acqu - acquisition parameters

acqu - acquisition status parameters

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

proc - processing parameters

procs - processing status parameters

OUTPUT FILES

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

fid - AMX type 1D raw data

ser - AMX type 2D or 3D raw data

acqu - acquisition parameters

acqu3 - acquisition status parameters

audita.txt - acquisition audit trail

<dir>/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

conv, vconv, jconv

fromjdx

NAME

fromjdx - convert a JCAMP-DX datafile to TOPSPIN format

SYNTAX

fromjdx [<pathname> [o]]

DESCRIPTION

The command **fromjdx** converts a JCAMP-DX data file to an TOPSPIN dataset. JCAMP-DX is a standard ascii exchange format for spectroscopic data.

fromjdx supports the conversion of 1D data (raw or processed) and 2D data (raw or processed-real).

fromjdx takes three arguments and can be used as follows:

fromjdx

prompts for the pathname of the JCAMP-DX input file and converts it and stores it under the lowest empty expno and procno 1.

fromjdx <pathname>

converts the JCAMP-DX file specified by the pathname and stores it under the lowest empty expno and procno 1

fromjdx <pathname> y

converts the JCAMP-DX file specified by the pathname and stores it under expno 1 and procno 1. Possibly existing data are overwritten (y).

In the examples above, **fromjdx** stores the output dataset in the directory:

<DU>/data/<USER>/nmr/<NAME>/<EXPNO>/pdata/<PROCNO>

where

<DU> - the data directory of the current dataset

<USER> - the user of the currently current dataset

<NAME> - the name of the JCAMP-DX file but without the extension .dx

Further examples:

fromjdx <pathname> du

converts the JCAMP-DX file specified by the pathname and stores it under

the DIR, USER, NAME, EXPNO and PROCNO as specified in the input JCAMP-DX file.

fromjdx <pathname> user

converts the JCAMP-DX file specified by the pathname and stores it under the DU of the current dataset and the USER, NAME, EXPNO and PROCNO as specified in the input JCAMP-DX file.

fromjdx <pathname> name

converts the JCAMP-DX file specified by the pathname and stores it under the DU and USER of the active dataset and the NAME, EXPNO and PROCNO as specified in the input JCAMP-DX file.

fromjdx <pathname> expno

converts the JCAMP-DX file specified by the pathname and stores it under the DU, USER and NAME of the active dataset and the EXPNO and PROCNO as specified in the input JCAMP-DX file.

fromjdx <pathname> procno

converts the JCAMP-DX file specified by the pathname and stores it under the DU, USER and NAME of the active dataset, EXPNO 1 and the PROCNO as specified in the input JCAMP-DX file.

All the above examples can be used with the **y** option to overwrite possibly existing data.

INPUT FILES

<pathname>/<mydata.dx> - TOPSPIN data in JCAMP-DX format

OUTPUT FILES

For 1D and 2D data:

<tshome>/prog/curdir/<user>/

curdat - current data parameters

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

audita.txt - acquisition audit trail (if input file contains raw data)

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

auditp.txt - processing audit trail (if input file contains processed data)

outd - output device parameters

`title` - title file (see ***setti***)

For 1D data:

`<dir>/data/<user>/nmr/<name>/<expno>/`

`fid` - 1D raw data (if input file contains 1D raw data)

`acqu` - acquisition parameters

`acqu`s - acquisition status parameters

`<dir>/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

`proc`s - processing status parameters

For 2D data:

`<dir>/data/<user>/nmr/<name>/<expno>/`

`ser` - 2D raw data (input if Output Data = raw)

`acqu` - F2 acquisition parameters

`acqu2` - F1 acquisition parameters

`acqu`s - F2 acquisition status parameters

`acqu2s` - F1 acquisition status parameters

`<dir>/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

`proc`s - F2 processing status parameters

`proc2s` - F1 processing status parameters

`clevels` - 2D contour levels

USAGE IN AU PROGRAMS

`FROMJDX(name, overwrite)`

for example `FROMJDX("/tmp/mydata.dx", "o")`

SEE ALSO

`tojdx`, `tozip`, `fromzip`

fromzip

NAME

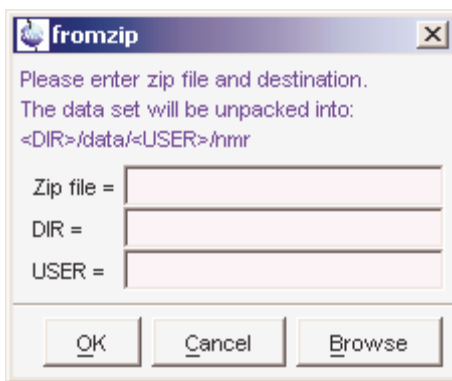
fromzip - unzip and display a zipped TOPSPIN dataset

SYNTAX

fromzip [<pathname> [o]]

DESCRIPTION

The command **fromzip** opens a dialog box to unzip a ZIP TOPSPIN dataset.



Here you can enter the ZIP file (pathname) and the DIR and USER part of the output data path.

fromzip takes up to three arguments and can be used as follows:

fromzip

opens the above dialog box.

fromzip <pathname>

converts the ZIP file specified by the pathname and stores it under the data path from which it was originally created

fromzip <pathname> <dir> <user>

converts the ZIP file specified by the pathname and stores it under the specified <dir> and <user> and the name, expno and procno as stored in the ZIP archive.

In the examples above, **fromzip** stores the output dataset in the directory:

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

The TOPSPIN dataset created by fromzip becomes the active dataset.

INPUT FILES

<pathname>/<mydata.bzip> - TOPSPIN data in ZIP format

OUTPUT FILES

For 1D and 2D data:

<tshome>/prog/curdir/<user>/

curdat - current data parameters

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

audita.txt - acquisition audit trail (if input file contains raw data)

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

auditp.txt - processing audit trail (if input file contains processed data)

outd - output device parameters

title - title file (see **setti**)

For 1D data:

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

fid - 1D raw data (if input file contains 1D raw data)

acqu - acquisition parameters

acqus - acquisition status parameters

<dir>/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:

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

ser - 2D raw data (input if Output Data = raw)

acqu - F2 acquisition parameters
acqu2 - F1 acquisition parameters
acqu_s - F2 acquisition status parameters
acqu_{2s} - F1 acquisition status parameters

<dir>/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
proc_s - F2 processing status parameters
proc_{2s} - F1 processing status parameters
cleve_ls - 2D contour levels

USAGE IN AU PROGRAMS

FROMZIP

SEE ALSO

tozip, tojdx, fromjdx

jconv

NAME

jconv - convert Jeol type data to Bruker TOPSPIN type data

SYNTAX

jconv [<inputfile>]

DESCRIPTION

The command **jconv** converts data from Jeol spectrometers to TOPSPIN 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*, *dir* and *user*, i.e. for the TOPSPIN 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*, *dir* and *user*, i.e. the TOPSPIN 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 TOPSPIN equivalent. First, the JNMR parameter EXMOD is interpreted. If it is set to a certain name, ***jconv*** checks the existence of an TOPSPIN 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 TOPSPIN equivalent and overwrites the values of the parameter set which was previously copied. The parameters of the TOPSPIN 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 TOPSPIN 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

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

fid - TOPSPIN 1D raw data

acqu - TOPSPIN acquisition parameters

acqu_s - TOPSPIN acquisition status parameters

audita.txt - acquisition audit trail

<dir>/data/<user>/nmr/<name>/<expno>/pdata/1/

proc - TOPSPIN processing parameters

proc_s - TOPSPIN processing status parameters

jnm 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

tojdx

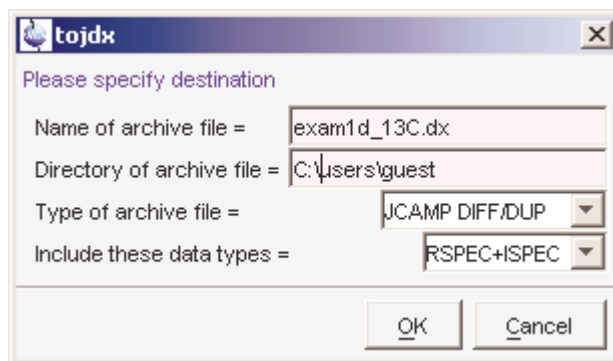
NAME

tojdx - convert a TOPSPIN 1D or 2D dataset to JCAMP-DX format

DESCRIPTION

The command **tojdx** converts an TOPSPIN dataset to JCAMP-DX format. JCAMP-DX is a standard ascii exchange format for spectroscopic data.

When **tojdx** is entered without argument, it will open a dialog box.



in which you can enter the required information. This includes:

Name of the archive file

The filename should have the extension .dx. This allows you to open it with TOPSPIN with drop & drag.

Directory of the archive file

Any directory.

Type of archive folder

For JCAMP format, you can choose between the following archive types:

- *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*)

Include these data types

For the included data types, you have the following choices:

- FID: raw data
- RSPEC: real processed data
- RSPEC+ISPEC: real and imaginary processed data
- PARAMS: parameter files

INPUT FILES

For 1D and 2D data:

<tshome>/prog/curdir/<user>/

curdat - current data parameters

For 1D data:

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

fid - 1D raw data (input if Output Data = raw)

acqus - acquisition status parameters

<dir>/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:

<dir>/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

<dir>/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> - TOPSPIN 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, fromzip, tozip

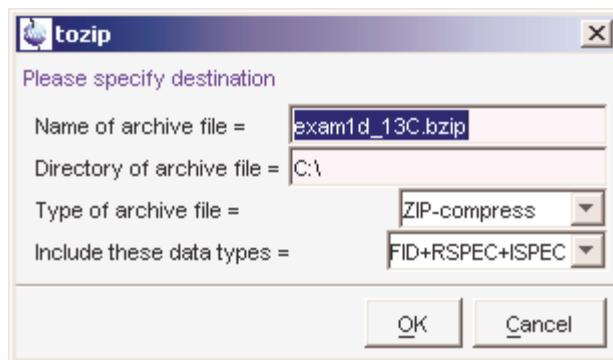
tozip

NAME

tozip - convert an TOPSPIN dataset to ZIP format

DESCRIPTION

The command **tozip** converts an TOPSPIN dataset to ZIP format. It opens a dialog box where you can enter the required information



This information includes:

Type of archive:

- *ZIP-compress*
- ZIP-no compress

Data types included:

- FID+RSPEC+ISPEC: raw + real and imaginary processed data
- FID+RSPEC: raw + real processed data
- FID: raw data
- RSPEC+ISPEC: real and imaginary processed data
- RSPEC: real processed data

INPUT FILES

If Data type includes FID

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

fid - 1D raw data

ser - 2D or 3D raw data

If Data type includes RSPEC

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

1r - real processed 1D data

2rr - real processed 2D data

3rrr - real processed 3D data

If Data type includes ISPEC

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

1i - imaginary processed 1D data

2ir, 2ri, 2ii - imaginary processed 2D data

3irr, 3rir, 3iii - imaginary processed 3D data

The parameter files `acqu*` and `proc*` are stored for all data types.

OUTPUT FILES

<pathname>/<mydata.bzip> - TOPSPIN data in ZIP format

USAGE IN AU PROGRAMS

TOZIP(name, datatype, compmode, title, origin, owner)

for example TOZIP("/tmp/mydata.dx", 0, 2, "mytitle", "BRUKER", "joe")

SEE ALSO

fromzip, tojdx, fromjdx

VCONV

NAME

vconv - convert Varian type data to Bruker TOPSPIN type data

DESCRIPTION

The command **vconv** converts Varian data, which were measured with the VNMR program, to TOPSPIN 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*, *dir* and *user*, i.e. the TOPSPIN 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 TOPSPIN data directory. When the specified data are found, you are prompted for the output *name*, *expno*, *dir* and *user*, i.e. the TOPSPIN data path.

vconv vdata.fid <name> <expno> <dir> <user>

as above but now the dataset *vdata.fid* is converted to the specified TOPSPIN dataset without prompting the user.

vconv <path>/vdata.fid

if the specified dataset is found, you are prompted for the output *name*, *expno*, *dir* and *user*, i.e. the TOPSPIN data path.

vconv <path>/vdata.fid <name> <expno> <dir> <user>

as above but now the dataset *vdata.fid* is converted to the specified TOPSPIN dataset without prompting the user.

If, in the second and fourth example, a part of the TOPSPIN 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 TOPSPIN equivalent. First, the VNMR parameter SEQFIL is interpreted. If it is set to a certain name, **vconv** checks the existence of an TOPSPIN 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 **vconv** converts all VNMR parameters which have an TOPSPIN equivalent and overwrites the values of the parameter set which was previously copied. The parameters of the TOPSPIN 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 TOPSPIN 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(background)	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 `procpa`r is stored in the TOPSPIN 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 TOPSPIN 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

`<dir>/data/<user>/nmr/<vdata>.fid`

or

`<VNMR>/<vdata>.fid/`

`fid` - the VNMR raw data
`procpa`r - the parameters
`text` - title file

OUTPUT FILES

`<dir>/data/<user>/nmr/<name>/<expno>/`

`fid` - TOPSPIN 1D raw data
`acqu` - TOPSPIN acquisition parameters
`acqu`s - TOPSPIN acquisition status parameters
`audita.txt` - acquisition audit trail

`<dir>/data/<user>/nmr/<name>/<expno>/pdata/1`

`proc` - TOPSPIN processing parameters
`procs` - TOPSPIN processing status parameters
`procpa`r - 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

TOPSPIN Interface/processes

This chapter describes commands which are related to the User interface and TOPSPIN processes. Each user can set up his/her own interface including the TOPSPIN 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.

arrange, cascade, icoall, maxall

NAME

arrange - arrange data windows
cascade - cascade data windows
icoall - iconify all data windows
maxall - maximize all data windows

DESCRIPTION

The command ***arrange*** arranges data windows. It is equivalent to clicking ***Window*** → ***Arrange***.

The command ***cascade*** cascades data windows. It is equivalent to clicking ***Window*** → ***Cascade***.

The command ***icoall*** iconifies all data windows. It is equivalent to clicking ***Window*** → ***Iconify all***

The command ***maxall*** maximizes all data windows. It is equivalent to clicking ***Window*** → ***Maximize All***.

SEE ALSO

close, newwin, nextwin

audit, auditcheck

NAME

`audit` - open audit trail handling dialog box
`auditcheck` - check the data consistency

DESCRIPTION

The command ***audit*** opens the audit trail dialog box.



This dialog box has several options, each of which selects a certain command for execution.

View audit trail of the processed data

This option selects the command ***audit proc*** for execution. It shows the processing audit trail file `auditp.txt`. This file is created by the processing command that creates the processed data, e.g. ***em***. 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.

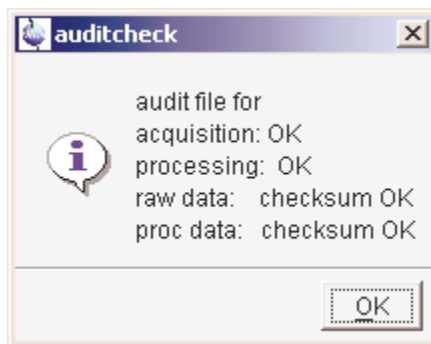
View audit trail of the acquisition data

This option selects the command ***audit acqu*** for execution. It shows the acquisition audit trail file `audita.txt`. This file is created by the acquisition command that creates the raw data, e.g. ***zg***. 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 param-

eters makes an additional entry.

Verify audit trails

This option selects the command ***audit check*** for execution. It performs an audit trail check, i.e. a data consistency check. If both raw and processed data are consistent, you will get the following message:



Add a comment to audit trail

This option selects the command ***audit com*** for execution. It allows you to add a comment to one of the audit trail files (raw or processed).

Each audit trail 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 TOPSPIN)
- *Where* : location where the command started (the computer host name)
- *What* : command and associated parameters, e.g. `<em LB = 0.3 SI = 16384>`

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
```

The command ***audit check*** uses this line for the consistency check.

INPUT FILES

```
<dir>/data/<user>/nmr/<name>/<expno>/
```

`audita.txt` - acquisition audit trail

`<dir>/data/<user>/nmr/<name>/<expno>/pdata/<procno>/`

`auditp.txt` - processing audit trail

Note that these are also the output files for ***audit com***.

SEE ALSO

`gdcheck`

edpy, xpy

NAME

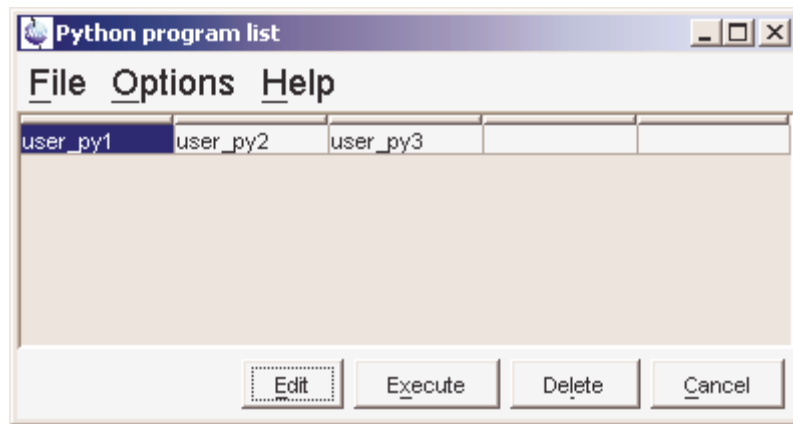
edpy - edit a Python program
xpy - execute a Python program

SYNTAX

edpy [<name> | <pathname>]
xpy [<name> | <pathname>]

DESCRIPTION

The command **edpy** allows you to edit a Python program. When it is entered without argument, it opens the Python dialog box showing all available Python programs:



This dialog box offers several options, each of which selects a certain command for execution.

Edit a Python program

Click *Edit*

Equivalent to double-clicking the Python program name, clicking **File** → **Edit** or entering **edpy <name>** on the command line.

Execute a Python program

Click *Execute*

Equivalent to entering **xpy** *<name>* on the command line. Note that **xpy** can also be started from the **run** dialog box.

Deleting a Python module

Click *Delete*

Equivalent clicking *File* → *Delete*

SEE ALSO

edau, xau

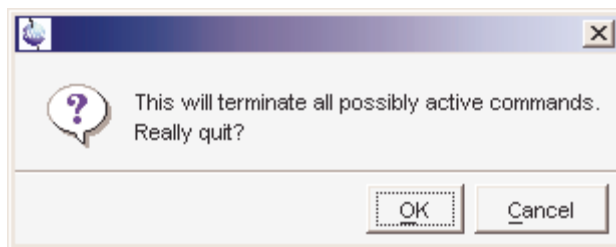
exit

NAME

exit - exit TOPSPIN

DESCRIPTION

The command **exit** exits TOPSPIN and terminates all running processes. Before this happens, you must quit in the following warning with **OK**:



Entering **exit** on the command line is equivalent to clicking *File* → *Exit*.

SEE ALSO

close

expl, run

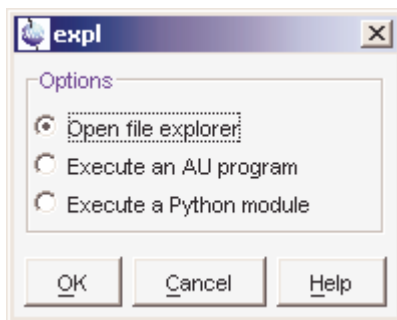
NAME

expl - open the Windows Explorer

run - open the run dialog window

DESCRIPTION

The command **run** opens the run dialog window:



This dialog box has three options, each of which selects a certain command for execution.

Open the file explorer

This option selects the command **expl** for execution. It opens the Windows Explorer showing the processed data files (the files in the PROCNO directory) of the active dataset. Under Linux the KDE konqueror will be opened. If no dataset is open in the TOPSPIN data area, the Explorer will show the users home directory. **expl** allows you access to the current data files as well as the entire data directory tree.

An alternative way to access data files is to right-click inside the data window and select **Files** in the appearing popup menu.

Execute an AU program

This option selects the command **xau** for execution. It opens the AU dialog box showing a list of available AU program. Here you can select an AU program and click **Execute** to execute it. **xau** can also be entered on the command line in which case you can specify the AU program as an argument.

Execute a Python program

This option selects the command ***xpy*** for execution. It prompts you for the pathname of a Python program. Just enter this pathname and click ***OK*** to execute the Python program.

SEE ALSO

xau, xpy

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

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

audit.a.txt - acquisition audit trail

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

audit.p.txt - processing audit trail

USAGE IN AU PROGRAMS

GDCHECK

GDCHECK_RAW

executes the command *gdcheck raw*

AUDITCOMMENTA("user comment")

adds a user comment to the audita.txt file.

AUDITCOMMENTTP("user comment")

adds a user comment to the auditp.txt file.

SEE ALSO

audit, auditcheck

hist

NAME

hist - show the TOPSPIN history and protocol

DESCRIPTION

The command **hist** shows the TOPSPIN protocol and history files. These files only contain information if the protocol function is active. You can switch on this function as follows:

1. Click *Options* → *Preferences* [**set**]
2. Click *Miscellaneous* in the left part of the dialog box.
3. Check the item *Record commands in protocol file*

The protocol file contains TOPSPIN startup information and command information on interface level. The history file contains command information on the level of the command interpreter and application modules. It also contains error messages.

Note that the files `history` and `protocol` are emptied when you restart TOPSPIN which means the history of the previous TOPSPIN session is lost. In case of problems, you should first make a copy of these files before you restart TOPSPIN. Note that a long TOPSPIN session, especially with automation can create very large the `history` and `protocol` files. Therefore, it is useful to regularly check the size of the file or simply restart TOPSPIN after each (automation) session.

OUTPUT FILES

<thome>/prog/curdir/<USER>/

`history` - TOPSPIN history file

<user-home>/<.topspin-hostname>/prop/

`protocol.txt` - TOPSPIN history file

kill, show

NAME

kill, show - show active TOPSPIN commands and allow to kill them

DESCRIPTION

The command ***kill*** displays a list of all active TOPSPIN commands. To kill a command:

- click a command entry
- click the button ***Kill...***

The command ***show*** is equivalent to ***kill***.

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.

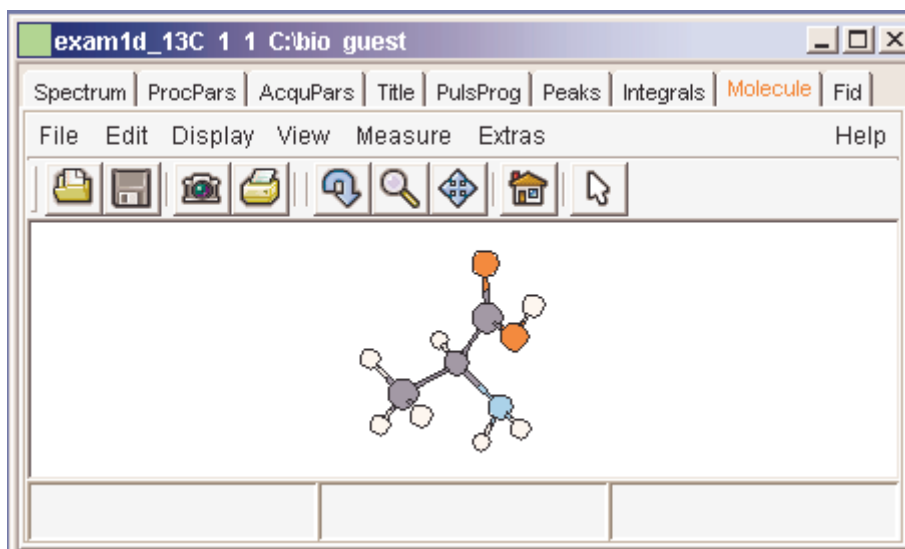
molview

NAME

molview - Open the Jmol molecule structure viewer

DESCRIPTION

The command **molview** opens the *Jmol* molecule structure. Entering this command is equivalent to clicking the **Molecule** tab in the data window. The data window will switch to the following display



The viewer displays the structure file that resides in the expno of the current dataset. If this does not exist, the structure file defined by the acquisition parameter CHEMSTR is displayed. CHEMSTR can define a full pathname or a filename. In the latter case, the file is searched for in the directory defined in the User Preferences. To set this directory:

Click **Options** → **Preferences** and select **Directory pathname**

If no structure file is found, you can open one by clicking:

File → **Open** in the molecule viewer

INPUT PARAMETERS

set by the user with **eda** or by typing **chemstr** :

CHEMSTR - molecule structure filename

INPUT FILES

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

<name> - molecule structure file

acqu - TOPSPIN acquisition parameters

<tshome>/classes/prop/StructureSamples/* - molecule structure files

nbook

NAME

nbook - open the user notebook

DESCRIPTION

The command ***nbook*** opens a user specific notebook. Each user can create and keep his/her own notebook for individual notes, information, settings etc.

INPUT AND OUTPUT FILES

<user-home>/<.topspin-hostname/prop/
notebook.txt - notebook text file

SEE ALSO

peakw

newtop

NAME

newtop - open a new Topspin interface

DESCRIPTION

The command **newtop** opens a new additional TOPSPIN interface. The additional interface is completely equivalent to the one it was started from. Entering **newtop** in the second or in the initial TOPSPIN interface opens another interface etc. The number of TOPSPIN interfaces is only limited by the available computer memory.

When single dataset is displayed in multiple TOPSPIN interfaces, the display in each interface is completely independent from the others. As such, you can display different regions, scalings and data objects. When the dataset is (re)processed from one interface, its display is automatically updated in all TOPSPIN interfaces.

The command **exit** closes the current Topspin interface. Interfaces that were opened from that interface remain open. Entering **exit** in the last open TOPSPIN interface, finishes the entire TOPSPIN session.

The position and geometry of each TOPSPIN interface is saved and restored after restart.

SEE ALSO

exit, newwin, hist

newwin, nextwin, close

NAME

newwin - open a new (empty) data window
nextwin - select the next data window
close - close the current data window

DESCRIPTION

The command **newwin** opens a new empty data window. It is equivalent to clicking *Window* → *New Window*

The command **nextwin** activates the next open data window. It is equivalent to clicking *Window* → *Next Window* or hitting the **F6** key

The command **close** closes the current data window. It is equivalent to clicking *File* → *Close* or hitting **Ctrl-w**.

SEE ALSO

arrange, cascade, icoall, maxall, newtop

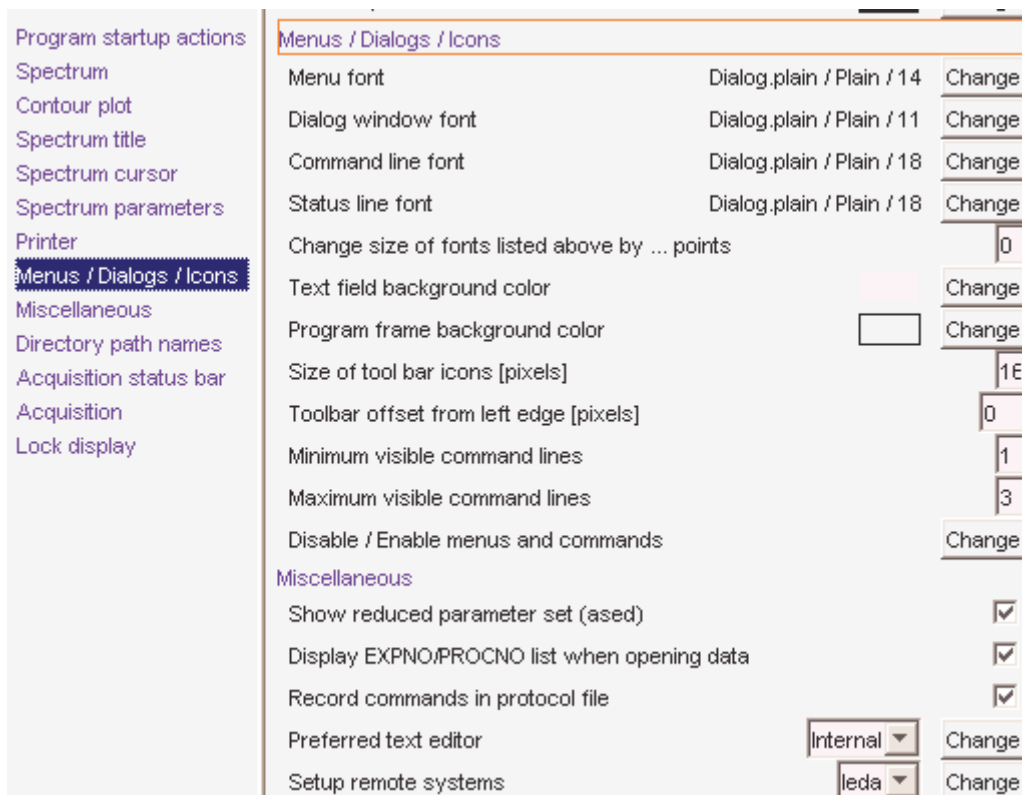
set

NAME

set - edit the TOPSPIN user resources

DESCRIPTION

The command **set** allows you to set user preferences. It opens the following dialog box.



In the left part of the dialog window, you find various category of objects. Click the category of which you want to view/change certain objects. It will become highlighted and the corresponding objects will be displayed at the right part of the dialog box. Some objects can be changed by entering a value, others can be

changed by clicking the ***Change*** button to the right of the object entry.

INPUT AND OUTPUT FILE

<home>/xwinmmr-<hostname>/

resources - ascii file containing all User Interface settings

where

<home> is the users home directory

<hostname> is the hostname of the computer

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 TOPSPIN 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 on - store standard output message

setdef stdout off - do not store standard output messages

The equivalent for standard error messages is **setdef stderr ok/no**.

OUTPUT FILES

<tshome>/prog/curdir/<user>

stdout.num - standard TOPSPIN output file for **setdef stdout ok**

stderr.num - standard TOPSPIN error file for **setdef stderr ok**

swin

NAME

swin - swap the position and geometry of two data windows

DESCRIPTION

The command **swin** swaps the position of two data windows. If the layout contains exactly two data windows, swin simple swaps their position and geometry. If the layout contains more than two data windows, **swin** allows you to swap the currently selected (active) data window with any of the other data windows. The latter can be selected from a list.

swin is typically used after reading a window layout with more than one data window.

SEE ALSO

close, newwin, nextwin, arrange, cascade, icoall, maxall

Index

Symbols

.basl command 385
.png files 294
.tif files 294
.wmf files 294

A

abs command 44, 75, 385
abs1 command 135
abs2 command 132
absd command 44
absd1 command 135
absd2 command 132
absf command 44
absot1 command 135
absot2 command 132
abst1 command 135
abst2 command 132
acquisition
 dimension 4, 16, 131, 245, 265
 mode 24, 75, 76, 122, 233, 330
 parameters 9, 12, 367, 368, 391, 394
 status parameters 9, 10, 12, 13, 24, 28, 324, 330, 391
 time 4, 31, 67, 117
add
 data to portfolio 353
 two 1D datasets 48
 two 1D fids 48
 two 2D datasets 138
 two 2D raw datasets 141
add command 48
add2d command 138
addc command 48
addfid command 48
addition factor 17, 18
addser command 141
adsu command 48, 95, 138, 172

AMX

 format 31, 397, 403
 spectrometer 26, 403
apk command 52, 55, 75
apk0 command 52
apk0f command 52
apk1 command 52
apkf command 55
apks command 55
arrange command 424

AU program

 binaries 372, 373, 379
 compile 372, 373
 install 378
 kill 379
 macros 8
 processing 15
 setup 377
 sources 372, 373, 379

AU reference manual 379

audit command 425
audit trail 433
auditcheck command 425, 433
automatic baseline correction 1D 44
automatic baseline correction 2D 132, 135
automatic mode of the Processing Guide 102
automatic shifting baseline correction 2D 132, 135
autoplot command 292, 298, 301

Avance

 data 18, 31, 211, 267, 402, 403
 spectrometer 7, 25, 26, 76, 211, 397

B

bas command 44, 132, 135
base_info file 385
baseline correction
 1D automatic 14, 44, 46, 75, 385
 1D fid 16, 58, 64, 75, 121, 122
 1D spline 115, 385
 1D user defined 61, 385

- 2D automatic 28, 132, 133, 136, 193, 204
- 2D automatic shifting 133, 136
- 2D FID 209, 234
- 2D user defined 143
- 3D automatic 263
- 3D FID 265, 272, 277
- frequency offset 17
- mode 16
- multiple additive 238
- of integrals 19, 309
- of the FID 17
- basl command 144
- baslpnts file 385
- bc command 58, 75, 121
- bcm command 61, 385
- bcm1 command 143
- bcm2 command 143
- bias correction 309
- big endian 33, 213, 268
- browse command 345, 348
- byte order 32, 33, 213

C

- cal command 329
- calibration
 - automatic 329
 - default 330
 - interactive 23, 27, 29
- cascade command 424
- checksum 433
- chemical shift 330, 331
- circular shift 180
- Clipboard 336, 359
- close command 441
- compileall command 372
- compiling AU programs 373, 379
- composite processing command 13, 64, 72, 86, 100
- contour levels 296
- conv command 358, 398
- convdta command 403
- conversion commands 397
- copy command 336
- correction offset 17
- cosine window multiplication 108
- cplbruk command 373

- cpluser command 373

D

- data
 - mode 17
 - overflow 34, 212
- data window
 - arrange 424
 - cascade 424
 - close 441
 - current 346, 349, 352, 355
 - geometry 445
 - iconify 424
 - maximize 424
 - new 346, 349, 441
 - next 441
 - position 445
 - reopen 363
 - swap 445
- dataset
 - dimensionality 10
 - directory tree 12
 - dosy 3D 246
 - hypercomplex 2D 159
 - inconsistent 436
 - Jeol 411
 - properties 352
 - status 9
 - Varian 419
- dcon command 80
- dconpl command 80
- deconvolution
 - Gaussian 80
 - Lorentzian 80
 - mixed Gaussian/Lorentzian 80, 385
- default
 - calibration 330
 - find criteria 352
 - printer 301
- degree of the polynomial 14, 45, 133, 136, 263
- del command 337
- del2d command 341
- dela command 337
- delau command 377
- deldat command 337

delete
 1D processed data 341
 1D raw data 341
 2D processed data 341
 2D raw data 341
 imaginary data 341
 integral lists 384
 processed data 337
 raw data 337
delete command 337, 341
delf command 341
deli command 229, 341
delmac command 382
delmisc command 384
delp command 337
dels command 341
delser command 341
detection mode 16, 17, 22, 58, 77, 121, 234
diagonal
 line in 2D 175
 plane in 3D 253
digital filtering 18
 acquisition 7
 processing 70
digitally filtered data 7, 18, 26, 76, 211, 267
dimensionality 391, 392
dir command 345
dir2d command 348
dira command 345
dirdat command 345
dirf command 348
dirp command 345
dirs command 348
dirser command 348
disco projection 147
disk space 212, 267
disk unit 367, 401
div command 95
dosy2d command 146
dosy3d command 246
dpa command 375
dt command 63

E

edau command 8, 358, 377

edcpd command 358
eddosy command 380
edgrad command 358
edlev command 296
edlist command 358
edmac command 382
edmisc command 358, 384
edp command 387
edpul command 358
edpy command 358, 428
edti command 298
ef command 64
efp command 64
em command 64, 66, 75, 122
equidistant sequence of contour levels 297
exit command 430
expinstall command 324, 372, 373, 378, 391
expl command 431
exponential
 baseline correction 1D 46, 61, 385
 baseline correction 2D 143
 broadening factor 67
 window multiplication 20, 32, 64, 66, 67
exportfile command 294

F

f1disco command 147
f1projn command 149
f1projp command 149
f1sum command 152
f2disco command 147
f2projn command 149
f2projp command 149
f2sum command 152
files of a dataset 431
filt command 70
filter width 16
find command 351
first order phase correction 24, 35, 52, 99
first point correction 18
fit function 40
fmc command 72
Fourier transform 4, 18, 33
 1D 64, 72, 74, 75, 76, 86, 88, 113, 121
 2D 193, 204, 209, 229, 233

3D 211, 265, 266, 272, 277, 285
Fourier transform mode 18, 21, 33, 76, 122, 210
fp command 72
frequency domain data 4, 5, 75, 84, 154, 193, 204,
209, 231, 265, 266, 272, 277
fromjdx command 357, 405, 408
fromzip command 358
ft command 64, 72, 74, 86
ftf command 74, 208

G

Gaussian

baseline function 17, 58
broadening factor 67, 111
deconvolution 80
lineshape 81
window multiplication 19, 20, 32, 66, 86
gdcheck command 433
gdcon command 80
genfid command 84, 90, 121
genser command 154, 231, 233, 234
geometric sequence of contour levels 297
gf command 86
gfp command 86
gm command 66, 75, 86, 122
graphics files 294
group delay 7, 25, 26, 31, 76, 211, 267

H

Hilbert transform

1D 88
2D 212, 229
3D 267, 282, 285
hist command 435
history
function 435
ht command 88

I

icoall command 424
ift command 84, 90
imaginary data
1D 75, 88, 93, 99, 104, 121
2D 198, 202, 212, 229
3D 267, 282, 285

deleting 343
input parameters 10
int command 306, 308
int2d command 306
integral
extension factor 15
regions 1D 15, 45, 385
scaling 1D 309
sensitivity 20
sensitivity factor 14
values 1D 20
integration
interactive 385
menu 385
intensity
histogram 312, 315
scaling factor 34, 212, 234
value 4
intrng file 45, 385
inverse Fourier transform
1D 84, 90, 122
2D 154, 231, 233, 238

J

JCAMP-DX format 312, 315, 405, 414
jconv command 358, 411
Jeol data 397

K

KDE konqueror 431
kill command 379, 436

L

layout Plot Editor 292
ldcon command 80
least significant byte 33
least square fit 17, 58
left shift 23, 91, 100
li command 46, 308
line broadening factor 67
linear prediction
1D 76, 77, 121, 122
2D 193, 204, 209, 235
3D 265, 272, 277
number of coefficients 22

- number of points 20
- lipp command 46, 308
- lippi command 308
- list
 - of active commands 436
 - of AU programs 431
 - of Bruker AU programs 378
 - of datasets 338, 339, 341, 345, 348
 - of found data 352
 - of integrals 308
 - of Jeol data 411
 - of miscellaneous files 385
 - of parameter sets 390
 - of peaks 2D 318
 - of peaks 3D 321
 - of processing parameters 13
 - of user AU programs 378
 - of Vnmr data 419
 - plot layouts 301
- little endian 33, 213, 268
- lock substance 330
- lock table 329
- Lorentzian
 - broadening factor 20
 - deconvolution 80
 - lineshape 81
- ls command 91, 100

M

- macros
 - in AU programs 8
 - in TOPSPIN 8, 382
- magnet field drifts 180
- magnitude calculation
 - 1D 25, 72, 93, 325
- magnitude spectrum
 - 1D 93
 - 2D 7, 176, 196, 198, 200, 202
- mana command 311
- maxall command 424
- maximum intensity
 - in 1D peak picking 82, 316
 - of a spectrum 38
- mc command 72, 93
- mdcon command 80, 385

- minimum intensity
 - in 1D peak picking 82, 316
 - of a spectrum 38
- miscellaneous lists 384
- mixed Gaussian/Lorentzian deconvolution 80, 385
- mixed sine/cosine function 108
- molview command 437
- most significant byte 33
- mul command 95
- mulc command 95
- multiplication factor
 - 1D 18
 - 2D 14, 19
 - 2D contours 297
 - first point acquisition 18
- multiply two datasets 95

N

- nbook 439
- negate a dataset 95
- new command 354
- new dataset 90, 231, 354, 394, 403
- newtop command 440
- newwin command 441
- nextwin command 441
- nm command 95
- noise region 23, 324

O

- objects
 - of a dataset 363
- open command 357
- orthogonal trace 150, 158
- output parameters 10
- overlapping peaks 45, 67, 81

P

- parameter sets 371, 391, 412, 420
- paste command 359
- peak
 - highest 26, 314
 - second highest 14, 15, 316
 - separation 16
 - sign 27, 315, 316
- peak picking

- 1D 325
 - 2D 318
 - 3D 320
 - maximum intensity 21
 - minimum intensity 22
 - parameters 82
 - sensitivity 24, 82, 316
 - peak.txt file 317
 - peaklist file 385
 - peaks file 317
 - peakw command 106
 - ph command 52, 55, 196, 200, 226
 - phase correction
 - 1D 64, 72, 86, 88, 99, 121, 122
 - 1D automatic 55, 75
 - 2D 193, 204, 209, 212, 226, 229
 - 3D 265, 267, 272, 277, 282, 285
 - automatic 13, 14
 - first order 24, 35, 52, 53
 - interactive 2D 210
 - mode 25
 - multiple 24, 25, 35
 - of 1D raw data 100
 - of raw AMX data 267
 - of raw data 26, 76
 - zero order 24, 35, 52, 53
 - phase sensitive spectrum
 - 2D 176, 198, 202
 - phase values
 - 1D 13, 56, 99
 - 2D 210
 - 3D 266, 267, 273, 278
 - pk command 64, 72, 86, 99
 - plane from 3D data 213, 249, 253, 255, 261, 282, 287
 - plot
 - editor 299
 - layouts 292
 - region 1D 26, 44, 314, 385
 - title 298
 - plot command 46, 299, 300
 - Plot Editor 292, 300
 - polynomial baseline correction
 - 1D spectrum 45, 46, 61, 385
 - 2D spectrum 133, 136, 143
 - 3D spectrum 263
 - fid 17, 58
 - power spectrum
 - 1D 104
 - 2D 7, 200
 - mode 25
 - pp command 21, 312, 318, 320, 385
 - pp2d command 318
 - pp3d command 320
 - pph command 312
 - ppj command 312
 - ppl command 312
 - ppp command 80, 385
 - pps command 312
 - prguide command 102
 - print
 - the active window 300
 - print command 300
 - prnt command 298, 300, 303
 - processed data 5, 6, 7, 17
 - Processing Guide 102
 - processing parameters 9, 12
 - processing status parameters 9, 12, 375
 - PROCNO 431
 - proj command 147, 149, 152, 157
 - projection
 - disco 2D 147
 - negative full 2D 157
 - negative partial 2D 149
 - positive 3D 247
 - positive full 2D 157
 - positive partial 2D 149
 - projpln command 247
 - projplp command 247
 - properties
 - of a printer 300
 - ps command 104
 - pseudo-raw data 84, 90, 154, 231, 261
 - ptilt command 179
 - ptilt1 command 179
- ## Q
- qsin command 107
 - qsinc command 107
 - quad spike correction 204, 211
 - quadrature detection mode 19, 58, 122, 234

R

r12 command 249, 282
 r12d command 253
 r13 command 249, 282
 r13d command 253
 r23 command 249, 282
 r23d command 253
 raw data 5, 6, 7, 17, 33, 50
 re command 357, 361
 reb command 357
 reference
 column for disco projections 148
 data for integral scaling 20, 309
 frequency 27, 29, 330
 peak for frequency calibration 329
 peak for scaling 14, 15, 26, 29, 314, 316
 row for disco projections 148
 shift 330
 substance 330
 reg file 26, 314, 316, 385
 reopen command 363
 rep command 357, 361
 replace current portfolio 353
 repw command 361
 resolution of a screen dump 294
 rev1 command 156, 212
 rev2 command 156, 212
 reverse
 1D spectrum 76, 113
 2D spectrum 156, 212
 3D spectrum 267, 274, 279
 flag 27
 rew command 361
 rhnp command 157
 rhpp command 157
 right shift 23, 91, 100
 rmisc command 384
 rpar command 390, 394, 412, 420
 rpl command 255
 rs command 91, 100
 rsc command 122, 144, 161, 183, 227
 rser command 166
 rser2d command 261
 rsr command 144, 169, 191, 227
 run command 431

rv command 76, 113
 rvnp command 157
 rvpp command 157

S

sab command 115, 385
 saguide command 323
 save
 a data window to a graphics file 294
 scaling region file 26, 29, 314, 316, 324, 326
 screen dump 294
 search command 351
 second dataset 19, 141
 select
 a plot layout 301
 a printer 300
 sequential
 data format 216, 268
 detection mode 75
 set command 442
 setdef command 8, 444
 SGI workstation 33, 213, 268
 shear AU program 181
 Shift key 353
 show command 436
 signal region 28, 324, 325
 signal to noise ratio 28, 35, 324
 simultaneous detection mode 75
 sinc
 squared window multiplication 107
 window multiplication 109
 sinc command 107
 sine
 baseline correction 1D 46, 61
 baseline correction 2D 143
 squared window multiplication 32, 107
 window multiplication 32, 107
 sine bell shift 29, 111
 sine command 122
 single detection mode 17, 19, 58, 75
 sinm command 107
 sino command 324
 slice command 249
 slope correction 309
 smail command 364

- sola command 328
- solvent peak 26, 29, 315
- spline baseline correction 46, 115, 385
- square brackets 363
- sref command 329
- standard deviation 14, 45, 195
- status parameter
 - display 375, 387
- storage order 3D data 245
- strip
 - size 29, 36, 75, 214, 268
 - start 30, 75, 214, 268
 - transform 29, 30, 36
 - transform 1D 75
 - transform 2D 214
 - transform 3D 268, 273, 278
- sub1 command 172
- sub1d1 command 172
- sub1d2 command 172
- sub2 command 172
- subcube format 30, 36, 268
- subcube size 37, 268, 269
- submatrix format 30, 36, 213, 234
- submatrix size 37, 205, 206, 214, 223
- subtract a 1D from a 2D 172
- subtract two 2D datasets 138
- subtract two 2D raw datasets 141
- sumpl command 247
- susceptibility 330
- swin command 445
- sym command 175, 176
- syma command 175
- symj command 175
- symmetrize a 2D spectrum 36, 175
- synt command 175, 179

T

- t1guide command 333
- tabs1 command 263
- tabs2 command 263
- tabs3 command 263
- tf1 command 33, 277, 282
- tf1p command 282, 285
- tf2 command 272, 282
- tf2p command 282, 285

- tf3 command 265, 272, 273, 277, 282
- tf3p command 282, 285
- third party software 213, 216, 229, 245, 268, 269, 285
- tht1 command 285
- tht2 command 285
- tht3 command 267, 282, 285
- tilt a 2D spectrum 179
- tilt command 179
- tilt factor 14, 180
- time domain data 4, 75, 84, 154, 209, 231, 249
- title bar 363
- tm command 117
- tojdx command 315, 414, 417
- trace 131, 150, 158
- traf command 117
- Traficante window multiplication 32, 117
- trafs command 117
- trapezoidal window multiplication 31, 32, 117
- trf command 76, 77, 100, 121
- trfp command 121, 164
- truncated fid 77, 209, 265, 272, 277
- tube of 3D data 269, 277

U

- user defined
 - AU programs 373
 - baseline correction 61, 143
 - parameter sets 391
 - phase values 99
 - plot layouts 301
 - processing 121, 233, 238
 - tilt angle 180
- User Interface 443

V

- Varian data 397, 419
- vconv command 358, 419

W

- weighting coefficients 70
- winconv command 358
- window multiplication
 - 1D 75, 121, 122
 - 1D exponential 64, 66, 67
 - 1D Gaussian 66, 86

- 1D sinc squared 107
- 1D sine 107
- 1D square sine 107
- 1D Trafficante 117
- 1D trapezoidal 117
- 2D 193, 204, 209
- 3D 265, 272, 277
- exponential 20
- Gaussian 19
 - mode 31, 121
- wm command 66, 107, 117
- wmisc command 384
- wpar command 391, 394, 412, 420
- wpl command 287
- wra command 367
- wrp command 367
- wrpa command 367
- wsc command 183
- wser command 186
- wserp command 189
- wsr command 191

X

- xau command 377
- XCMD 8, 225
- xf1 command 193, 204, 211, 229, 250
- xf1m command 196
- xf1p command 226
- xf1ps command 200
- xf2 command 193, 204, 211, 229, 250
- xf2m command 196
- xf2p command 226, 250
- xf2ps command 200
- xfb command 193, 208, 233
- fbm command 196
- fbp command 226
- fbps command 200
- xht1 command 229
- xht2 command 229
- xif1 command 154, 155, 231
- xif2 command 154, 155, 231
- xmac command 382
- xpy command 428
- xtrf command 212, 233, 238
- xtrf2 command 233

- xtrfp command 231, 235, 238
- xtrfp1 command 231, 238
- xtrfp2 command 231, 238

Z

- zero data 126
- zero filling 31, 75, 211, 266
- zero intensity 23, 126
- zero order
 - baseline correction 61, 144
 - phase correction 24, 35, 52, 99
- zert command 241
- zert1 command 241
- zert2 command 241
- zf command 126
- ZIP format 417
- zp command 128

