G A S P A N

INTERACTIVE AND AUTOMATIC ANALYSIS OF GAMMA-RAY AND PARTILE SPECTRA

FRIEDRICH RIESS
Sektion Physik der Universität München, 80799 München, W. Germany

July 15, 2006

A Tutorial

Warning: this tutorial is not complete

Introduction

The automatic analysis of gamma-ray spectra from Ge-detectors is a challenging task. Reliable programs are needed to handle the large amount of data produced in todays experiment. The program *gaspan* has been designed to be used either interactively to analyze single lines in a spectrum or to analyze complicated spectra for example spectra with a mix of lines with different line widths, or in an automatic mode to analyze large amounts of spectra with many lines. It can be used to analyze all line structures with Gaussian or nearly Gaussian line shapes such as spectra from NaI- or Ge-detectors, solid state particle detectors and spectra from magnetic spectrograph-s.

The following introduction should help the user to become familiar with the different possibilities of using the program. A description of the fit function and other programming consideration are put in the appendices of this manual.

The program *gaspan* is designed to run at UNIX systems. The source code can be downloaded from the URL

http://www.cip.physik.uni-muenchen.de/~riess

in the file "gaspan-yy-mm-dd.tgz". The spectra used as examples in this tutorial are located in the files "testsuite-yy-mm-dd.tgz" respectively "examples.tgz". A binary file for Linux systems can be also found in this URL.

This tutorial splits into four chapters:

- "1. Interactive Peak Fitting"
  The user learns in this chapter about the command structure and the philosophies of *gaspan* . Examples of simple interactive fitting procedures will be demonstrated.
  This chapter is thought for a student who likes to analyze a few lines in a simple spectrum in relation to his academical training.

- "2. Advanced Peak Fitting"
  This chapter deals with more complicated cases as lines with large asymmetries as well as with spectra of high line density. Automatic analysis of spectra will be explained.
  This chapter is thought for students who have to do spectra analysis in the frame work of a more elaborate work (Master, Diploma or PhD.).

- "3. Special Cases"
  Special cases as for example spectra containing groups of lines with different line widths as well as the generation of background corrected gate spectra from coincidence matrices are discussed.

- "Appendix"
  brings a summary of commands, a lesson how to write and incorporate an own spectrum read routine, a presentation of the fit function as well as other information about the program.

Contents

1 Interactive Peak Fitting

1.1 First Steps

1.1.1 Getting Started

The program *gaspan* if installed properly, can be executed by just typing "gaspan<cr>". It will respond by

```
GASPAN, Version dd.mm.yy
A program to fit gamma and particle spectra
Author: Friedrich Riess, email: friedrich.riess@physik.uni-muenchen.de
URL: http://www.physik.uni-muenchen.de/~riess/
URL: http://www.cip.physik.uni-muenchen.de/~riess/
GSP>
```

`dd.mm.yy` is the version number specified by the date of the release and the present tutorial is valid for versions newer than August 2005.

1.1.2 Reading a Data Set

Data sets i.e. spectra can be ASCII or binary files. *gaspan* has encorporated several data formats which are described in detail in appendix 4.2 and you are asked to have a look at this appendix when you want to read your own data sets. For the moment we will work with spectra found in the archive file "testsuite.tgz" which can be found in the URL http://www.physik.uni-muenchen.de/∼riess/.

To be able to read the spectrum stored in the file "ta182.dat" we need the command

```
set files -file=ta182.dat
```

Note: the data set is not accessed yet i.e. there will be only a warning and no error message if the file is not present.

1.1.3 Displaying the spectrum

It is now possible to look at the spectrum:

```
show spectrum
```

will display this 8k spectrum in units of 4096 channels (see fig. 1,2)[1].

At the top of the figure you find on the left side the name of the data set with an arrow pointing to the sub spectrum number, in the middle the date and time the graphic has been generated and on the right the version of *gaspan* The string "Tutorial" has been inserted via the command

```
set display -comment="Tutorial"
```

The command `show spectrum` has the first and the last channel to be displayed as optional

[1] The display will be partially or totally empty if the graphics window was hidden by other windows. Just type <cr> in the command mode in order to restore it (this possibility to recover the display does not work for versions before the year 2005).

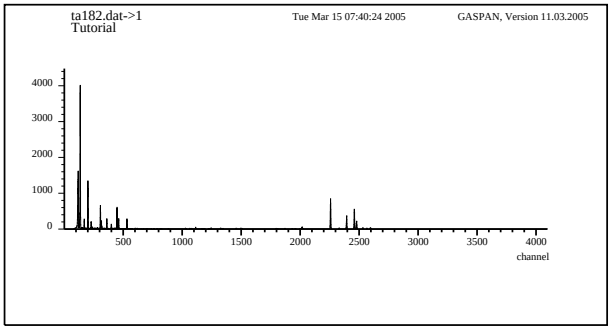


Figure 1: The first 4096 channels

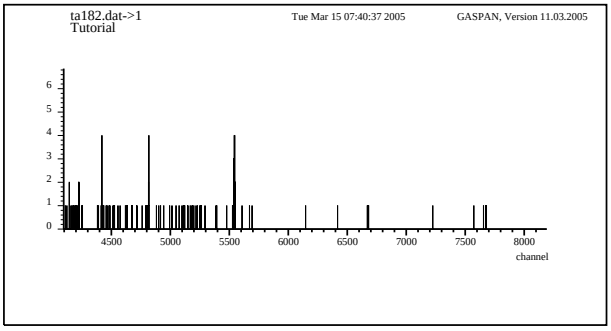


Figure 2: channels 4097 to 8192

argument:

```
show spectrum 420,500
```

will display just this region.

1.1.4 Fitting Gaussian peaks

A fit to the lines in this region can be obtained with the command

```
fit 420,500,2
```

gaspan will do a line search in this region, do a fit with a Gaussian plus a quadratic background (polynomial with degree 2), print the result and show the fit.

```
 Working on ta182.dat-->1
 Searching for peaks in region     420    500
          centroid                        area                width
    447.497 +/- 0.036          2223.      +/-    48.       3.41 +/-  0.06
    461.873 +/- 0.053          1051.      +/-    33.       3.41 +/-  0.06
 No error in this fit
next: <cr>, next spec: ESC, p(rint), s(ave)>
```

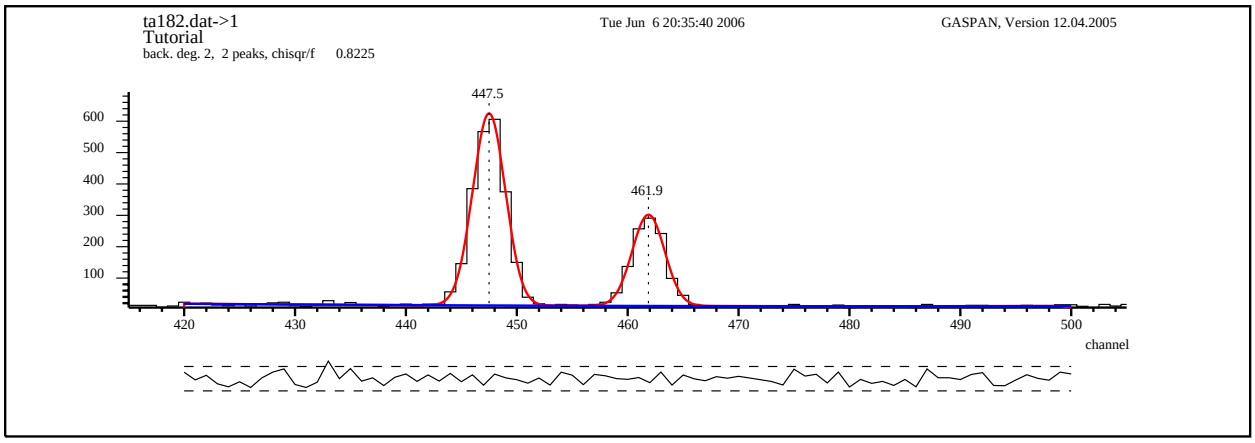


Figure 3: Fit with data points as histogram

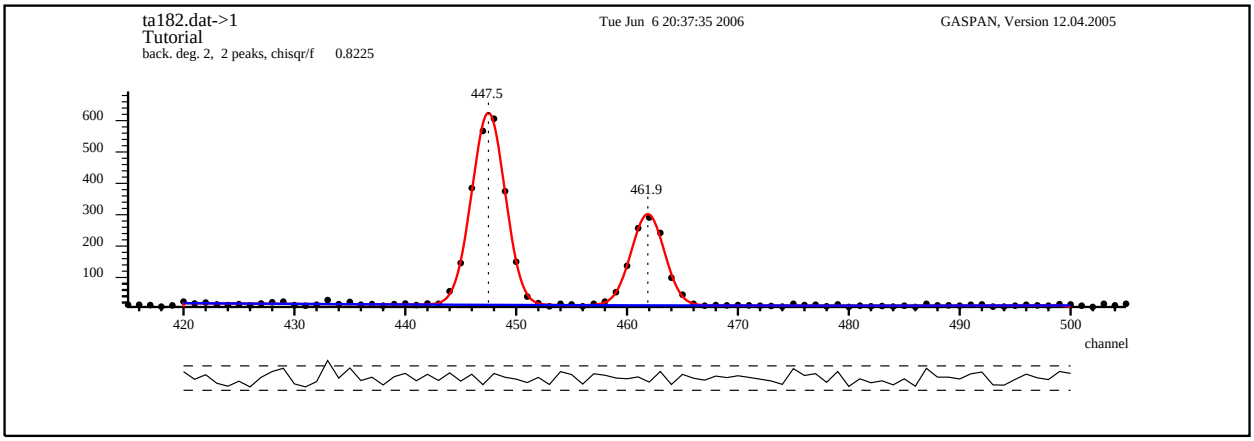


Figure 4: The data are shown as points with error bars

The display (see fig.3) shows the spectrum extended below and above the fit region as a histogram, the fit to the background as a blue line and the fit to the spectrum in red. The bottom part shows the residue spectrum of this fit[2] which is analyzed to detect small lines or doublets. The 99 % confidence limit is shown too. In the additional text line you see the degree of the background polynomial used, the number of lines found and chi-square normalized by the degree of freedom. If you do not like histogramming a spectrum you can plot the data with error bars (fig.4) with the commands

```
set display -errorbar
show fit
```

From now on all spectra will be shown with error bars. You can return to the histogramming mode with the command

```
set display -histogram
```

The program will analyse the residual spectrum for more lines and add them to the fitting procedure if some are found. Look at the region 210,340:

```
fit 210,340,2
```

You get seven lines in this fit (fig.5):

```
 Working on ta182.dat-->1
 Searching for peaks in region     210    340
          centroid                      area                width
    220.957 +/- 0.624           33.     +/-    12.       3.27 +/-  0.05
    228.670 +/- 0.071          644.     +/-    27.       3.27 +/-  0.05
    234.073 +/- 0.196          156.     +/-    17.       3.27 +/-  0.05
    282.807 +/- 0.239          102.     +/-    14.       3.27 +/-  0.05
    306.762 +/- 0.034         2303.     +/-    49.       3.27 +/-  0.05
    314.919 +/- 0.067          833.     +/-    31.       3.27 +/-  0.05
    319.226 +/- 0.199          190.     +/-    20.       3.27 +/-  0.05
 No error in this fit
next: <cr>, next spec: ESC, p(rint), s(ave)>
```

[2]See appendix 4.3 for an explanation

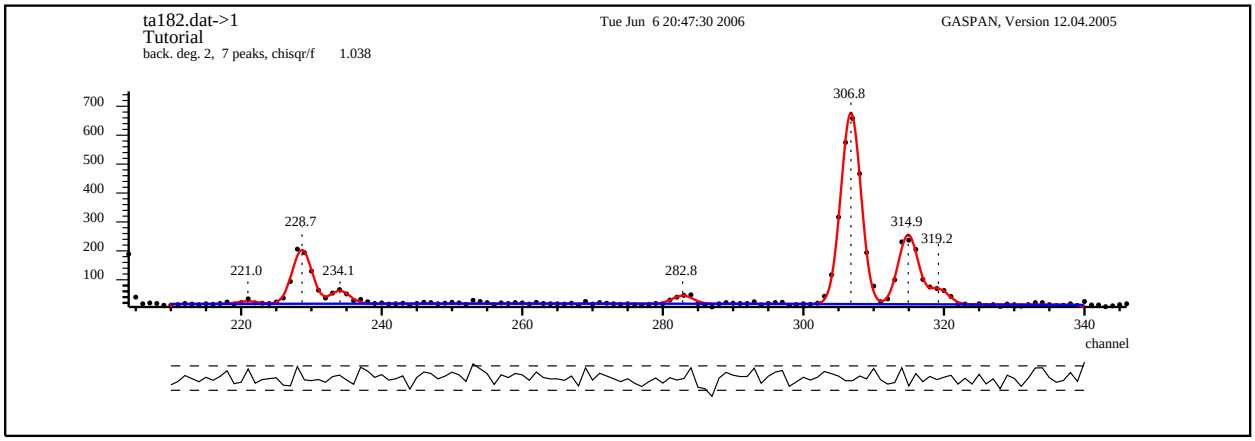


Figure 5: Another region

The fit looks reasonable. An analysis of this case shows that not all lines are found by the peak search routine. This will be discussed later in more detail.

1.1.5 Fitting asymmetric lines (I)

One problem in gamma ray spectra are asymmetric peaks or an asymmetric background. Try the following commands:

```
set file -file=eu152.dat
fit 235,345
```

and you get as result a bunch of weak lines[3]

```
 Working on eu152.dat-->1
 Searching for peaks in region     235    345
          centroid                      area                width
    259.180 +/- 0.327          153.     +/-    42.       3.05 +/-  0.01
    266.293 +/- 0.251          233.     +/-    45.       3.05 +/-  0.01
    270.440 +/- 0.176          402.     +/-    50.       3.05 +/-  0.01
    274.443 +/- 0.169          478.     +/-    54.       3.05 +/-  0.01
    278.210 +/- 0.165          539.     +/-    57.       3.05 +/-  0.01
    282.016 +/- 0.146          709.     +/-    64.       3.05 +/-  0.01
    285.468 +/- 0.152          743.     +/-    66.       3.05 +/-  0.01
    289.133 +/- 0.118         1007.     +/-    73.       3.05 +/-  0.01
    295.282 +/- 0.005        12425.e+01 +/-    55.e+01   3.05 +/-  0.01
    298.896 +/- 0.287           80.e+01 +/-    26.e+01   3.05 +/-  0.01
 No error in this fit
next: <cr>, next spec: ESC, p(rint), s(ave)>
```

and the fit shown in fig.6. Quit clearly there is the problem of weak lines in the presence of a strong line.

[3]Note: the format in the output of the value of the area is chosen such that the error is given with a two digit precision.

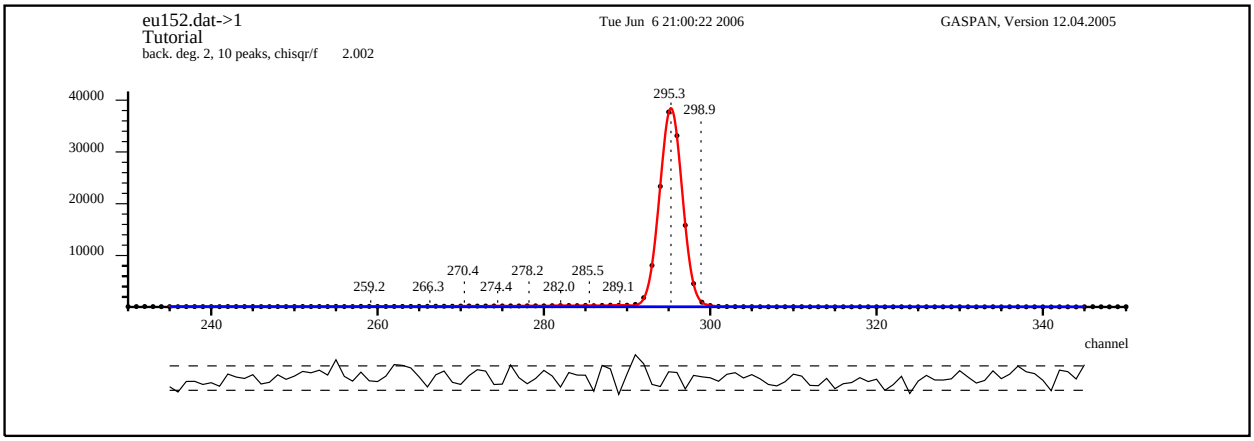


Figure 6: Are these peaks real?

Are these lines real or are they fake? You can blow up the background by using a logarithmic scale (fig.7):

```
set display -logarithmic
show fit
```

and you will recognize that all lines but the one at channel 295.3 are fakes. the lower lines are obviously added in order to fill up an asymmetric background.

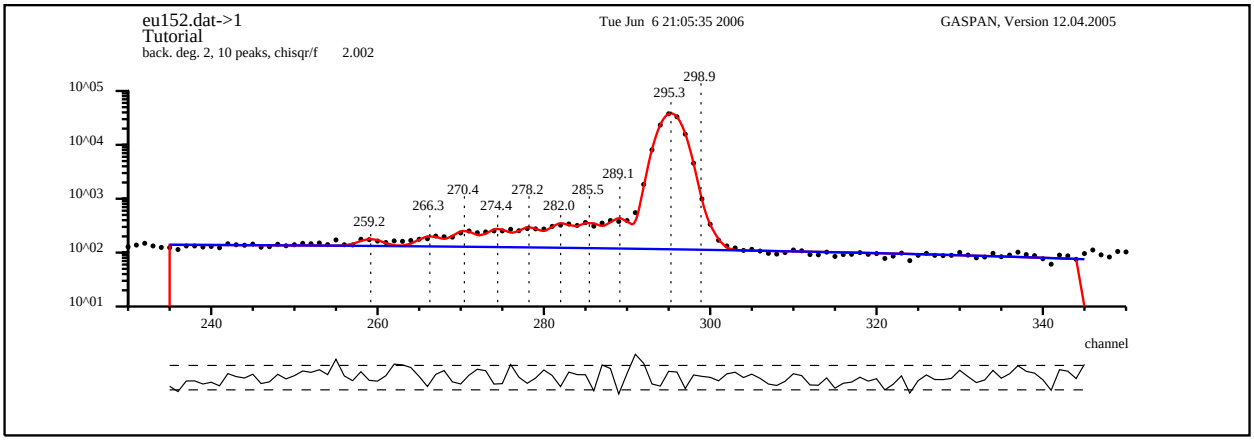


Figure 7: ...definitely not!

You might try an odd degree of the background polynomial

```
fit 235,345,3
```

but you will see that this does not help. There is an asymmetric background contribution from the line itself: an exponentially decaying tail with a large decay constant. This is called in the program a "background tail". We can include such a tail in the fit but there might be other asymmetric contributions. The easiest way to handle this problem is to include all asymmetries:

```
set tail -all
```

Now you get a perfect fit to the background and the lower peaks have disappeared (fig.8):

```
Working on eu152.dat-->1
```

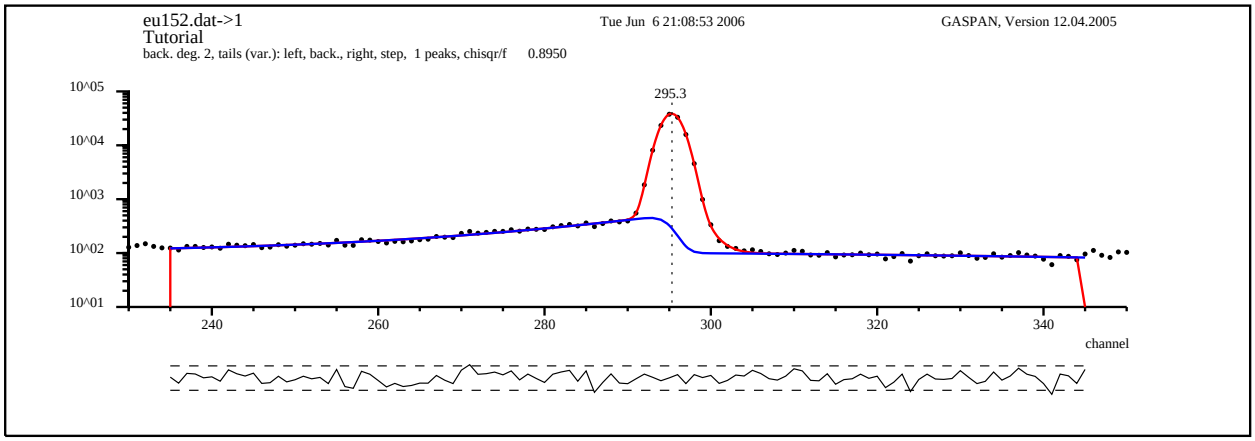


Figure 8: A fit with all asymmetries

```
Searching for peaks in region     235    345
         centroid                          area                    width
   295.328 +/- 0.005          12381.e+01 +/-    36.e+01    2.98 +/-  0.01
Input value for left tail decay constant too large.
```

Note the change in the areas! Obviously we get an error message: it turns out that there is no left peak tail in this case (see **??**) but this should not bother us at the moment

Give the command

```
save options
```

in order to save the current setup and leave *gaspan* with the command

```
quit
```

1.1.6 The command structure

Control of the program is done by keyboard input. There are several basic commands:

`set`	set an option or define a variable or a file name.
`show`	show data or options or other things
`fit`	fit a region interactively
`go`	fit a region automatically
`help`	get help about a command
`save`	save options into a file
`get`	get options from a file
`do`	execute a shell command
`quit`	quit the program
`exit`	quit the program

Commands can be abbreviated but the abbreviation must be unique. The command "`show`" can be abbreviated to "`sho`" or "`sh`" but not to "`s`" because "`s`" is no longer unique. The following error message will be printed if an abbreviated command is not unique:

```
-->CMD: command ambiguous: s
```

and the command will be printed up to the letter were the ambiguity starts. The commands are on a mnemonic basis and will therefore be written in the examples in full length[4].

Most of the basic commands are followed by an identifier and - in most cases - the identifier is followed by an option.

1.1.7 The History Mechanism

You can recall previous commands with the up arrow key and edit them with the left and right arrow keys and the backspace key. The history will be saved into the file ".gaspan" in the current directory when leaving *gaspan* and will be accessible in the next gaspan session[5].

1.1.8 Getting Help

The help command gives you online information about the commands in *gaspan* . It has the general structure:

```
help [command [subcommand [option]]]
```

You will get the message

```
--> Help file not found, notify operator
```

if the help file "gaspan.hlp" is not installed correctly but you will get an overview over the commands in any case. You can get information about the listed commands by typing `help command` and even without a help file you will see the subcommand or the options of a command. If you have access to the file "gaspan.hlp" you might copy it either into the current directory or into /tmp/ in order to access it.
You can also get access to the help file with the command

```
set file -helpfile=PATH/gaspan.hlp
```

where PATH is the path to the gaspan.hlp file.
Commands are shown in inverse video. An option which is followed by an equal sign requests further input:

`(%s)`	expects a string
`(%s,%d)`	expects a string followed by an integer number
`(%f,%f)`	expects two floating point numbers.

1.1.9 Showing the Status

There a several commands showing the status of *gaspan* . The command

```
show version
```

outputs the current version. The command

```
show news
```

prints a list of changes with respect to the last published version.
Each program has restrictions due to dimensions of arrays etc. The command

```
show limitations
```

prints a list of these restrictions and last not least you get a list of the currently active options, identifiers and qualifiers with the command

[4] In the summary of commands, appendix 4.1, a star in the command shows the position where the command is no longer ambiguous.

[5] The history algorithm was written by *Bertram Stanzel* †.

```
show options
```

1.1.10 Quit the program

The program can be quit only with the commands "`quit`" or "exit". **You can not quit the program with "CTRL C" respectively the "intr" command** (use the command "stty -a" on your system to see which character interrupts a program). This is because "CTRL C" is used to stop a running fit and to return to the command interpreter of *gaspan* .

1.2 Output Files

Note: You must have write permission in the directory you are running *gaspan* .

1.2.1 The Gaspan Data Summery File

Look at the inventory of your current directory and you will find the files "gsp1ta182.dat" and "gsp1eu152.dat". These are ASCII files which contain all information about the last fit: the options as well as the results of the fit. These files are called "gaspan data summary file". The filename of these files start always with the string "gsp", followed by a number which denotes the sub-spectrum[6], followed by the filename of the data set analyzed (without any path specifications).

A gaspan data summary file will be generated in any case. It starts with a header showing the options and parameters of the last fit, followed by the fit result. The line starting with R contains the fit region, the degree of the background polynomial a variable which indicates an error code if an error during the fit occurred and the normalized chi-square. The positions of the peaks are given with a 3 digits precision. Its error is set to a minimum value of 0.001 and a maximum value of 99.999. It follows the area under the peak. The precision of these numbers depend on the error of the area. They are given in such a representation that the error shows 2 digits. Finally the width is printed with a two digit precision.
Note: with each `fit` (and `go`) command the "gaspan data summery file" is generated new, hence old data are lost unless you give the command

```
set peak -append
```

but you should be aware that the options are saved only during the generation of the "gaspan data summary file".

1.2.2 Handling the display

There are many display commands allowing manupulation of the displayed spectrum. Please inspect the summary of commands 4.1.3 and 4.1.2. Only three of the commands will be discussed here in more detail:

`set display -saveall=`*filename*

Each display you will see on the screen is saved as Postscript file in the file *filename*. You can vue the pictures for example with the program "gv" (`gv filename`) and extract single pictures by selecting the command "Save Marked" in order to include them in a text file.

[6] *gaspan* accepts more than one spectrum in a data file. see chapter 4.2.

As an example, LaTeX allows you to include postscript files with the package "epsf" and the command sequence:

```
\begin{figure}[h]
\epsfxsize15cm
\leavevmode
\epsffile{filename}
\caption{\label{xxx} text}}
\end{figure}
```

The command is canceled with the command

`set display -nosaveall` (default)

There are two commands which control the window size and its contents:

`set display -windowsize=x,y`

defines the size of the generated window in fractions of the screen size i.e. x,y=1.,1. will generate a window covering the total screen. It should be evident that there is also a minimum window size (at present: `x,y=0.575,0.35` which is adapted for a screen resolution of 1280x1024).

```
set display -residuum
set display -noresiduum
```

defines if the residue spectrum is to be shown or not. The size of the spectrum will be extended into the space reserved for the residue spectrum if this is not shown (see fig. 9 and 10)

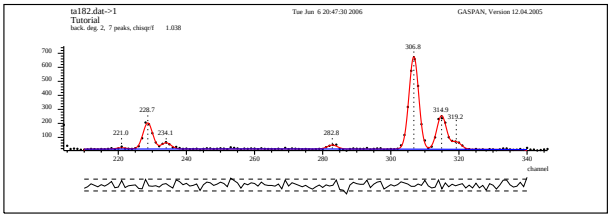


Figure 9: same as fig. 5

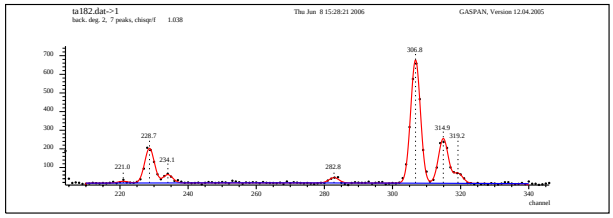


Figure 10: same as fig. 5 but without residue spectrum

1.2.3 Spectra files

The command `set fit -out` will generate 2 data files "fitNfile.dat" and "bckNfile.dat" which contain the fit-and background- spectra. N is the sub-spectrum number of the data file "file".

2 Advanced Peak Fitting (not implemented yet)

3 Special Cases (not implemented yet)

4 Appendix

4.1 Summary of commands

All commands can be abbreviated as long as they are unique. A star in the following list of commands denotes this position.

4.1.1 Defining Data Sets

command	argument	qualifiers	comment
se*t	fil*es	-fi*le=*name*	define name of datafile *name*
		-fi*le=*name,s1*	analyse subspectrum *s1* of datafile *name*
		-fi*le=*name,s1,s2*	analyse all subspectrum between *s1* and *s2* of the datafile *name*
		-fi*le=*name,s1,s2,ds*	analyse all subspectrum between *s1* and *s2* with increment *ds* of the datafile *name*
		-fi*le=*@name*	the file *name* contains a list of datafiles to be analysed
		-fo*rmat=*form*	select the format of the datafile *form* *form* = ascii, dat, daterr etc. see command "show limitations"
		-fo*rmat=search[1)]	derive format from file extension
		-matrix	two dimensional spectrum
		-single_spectrum[1)]	one dimensional spectrum

1) default

4.1.2 Controlling the display

command	argument	qualifiers	comment
se*t	d*isplay	-on[1)]	show spectrum and fit on terminal while fitting
		-of*f	no display of spectra while fitting
		-ho*ld	hold the display after each fit step, continue with <cr>.
		-noh*old[1)]	do not stop after a fit step
		-sa*veall=*filename*	save all displays in the postscript file *filename*
		-nosa*veall[1)]	do not save the displays automatically
		-i*ntermediate	display intermediate steps in the fit procedure
		-noi*ntermediate[1)]	display only the final fit
		-print=" *command*"	define a print command for printing the postscript file of the display. Default: lpr

1) default

4.1.3 Manipulating the Diplay

command	argument	qualifiers	comment
se*t	d*isplay	-er*rorbar	show data points with error bars
		-noe*rrorbar	show data points as histogram
		-hi*stogram[1)]	show data points as histogram
		-li*near[1)]	linear display of fit and spectrum
		-lo*garithmic	logarithmic display of fit and spectrum
		-ch*annel[1)]	mark peaks with channel numbers
		-en*ergy	mark peaks and axis with energies
		-s*ingle_peak	draw each single peak of the fit
		-nos*ingle_peak[1)]	do not draw each single peak of the fit
		-co*mment="*text*"	allows to specify a text line which will be shown on the display
		-co*mment=""[1)]	delete the text line
		-g*auss-position	put peak markers at the position of the gaussian peak
		-ce*ntroid	put peak markers at the position of the centroid of a peak
		-t*ailinclude[1)]	set marker positions according to the command "set tail -include"
		-r*esiduum[1)]	reserve space in the graphic window to show the residue spectrum
		-nor*esiduum	don't display residue spectrum, use space for display of the spectrum
		-w*indowsize=x,y	define the size of the graphic window, x and y are fractions of the screen size
		-y*scale=$ymin,ymax$	restrict display to the range $ymin \leq y \leq ymax$
		-y*scale=$,ymax$	get $ymin$ from the spectrum
		-y*scale=$ymin,$	get $ymax$ from the spectrum
		-y*scale=,[1)]	get $ymin,ymax$ from the spectrum

1) default

4.1.4 Controlling the Fit Procedure

command	argument	qualifiers (argument)	comments
se*t	fit	-res*idue_search[1)]	make a residue search to find doublets and weak peaks
		-nor*esidue_search	do not care about the residue spectrum
		-reg*ion=*min, max*	set analyzing window from channel *min* to *max*
se*t	r*egion	-s*earch[1)]	invoke region search routine
		-l*ist=*name*	get regions from data file *name*
		-f*it=*min,max*	set analyzing window from channel *min* to *max*
se*t	pa*rameter	-noc*heck[1)]	do not check if parameters minimize χ^2
		-c*heck	check if parameters minimize χ^2, return immediately to the fit routine if not
		-a*ll_check	check *all* parameters if they minimize χ^2, select the best improvement if not and return to the fit routine
		-f*ull_check	make a sampling of χ^2 around the current parameters, return immediately to the fit routine if an improvement is found
		-ex*tended_check	make a sampling of χ^2 around all current parameters, return to the fit routine with the best improvement if found
se*t	pe*ak	-s*earch[1)]	enter peak search routine
		-l*ist=*name*	get list of peaks from data file *name*
		-nol*ist	identical to -search
se*t	s*tatistics	-a*llpeaks	include all peaks from a peak list
		-noa*llpeaks[1)]	include only good peaks
		-se*nsitivity=*code*	define sensitivity in peak- and residue search (*code*=normal[1)] or high)
		-st*andard[1)]	assume normal distributed data
		-nos*tandard	assume none normal distributed data
se*t	w*idth	-l*ist	take the value for *FWHM* from the peak list
		-nol*ist[1)]	take the value for *FWHM* from the width calibration
		-r*ange=*value*	set the variation range for *FWHM* to *value*

1) default

4.1.5 Tailering the fit Function

command	argument	qualifiers	comment
se*t	b*ackground	-p*olynomial_ degree=*value*	set degree of background polynomial to *value* ($value = [0, 4]$)
		-m*ax_polynomial_ degree=*value*	the program tries to set the degree of background polynomial but not larger than *value*[1)]
			value = {0,1,2,3 or 4}
		-fix*ed=*value*	fix the background to *value* if the degree of the polynomial is 0
		-fit[1,2)]	include a constant background in the fit
se*t	pe*ak	-ind*ividual[1)]	vary all peak centroids independently
		-c*ommon	shift peak centroids by the same amount
		-f*it[1)]	include peak centroids in the fit
		-nof*it	keep peak centroids fixed
se*t	w*idth	-co*mmon[1)]	fit all widths in a fit region with one common parameter, standard for versions before 2005
		-i*ndividual	each peak has its own width parameter (only in versions newer than 2005)
		-r*ange=*value*	set the variation range for the width to *value* ($value = [0., 99.]$)
se*t	t*ail	-*kind*=*value*	include specified tail into the fit function and set variation range of decay constant to *value* ($value = [0., 0.95]$)
			kind might be one of the following:
			l*eft_peak_tail
			r*ight_peak_tail
			b*ackground_tail
		-s*tep	include a step function into the fit
		-f*it=*kind*	include amplitude into the fit
		-nof*it=*kind*	fix amplitude
		-f*it=step	include amplitude of the step into the fit
		-nof*it=step	fix amplitude of the step
		-gamma	add left- and background tail
		-all	add all tails but step
		-non*e[1)]	fit only with Gaussians
		-no*kind*	exclude specified tail from fit function

1) default
2) polynomials with degree greater 0 are allways fitted

4.1.6 Defining Starting Parameters

command	argument	qualifiers	comment
se*t	b*ackground	-fix*ed=*value*	fix the background to *value* if the degree of the polynomial is 0
se*t	pe*ak	-l*ist=*list*	*list* is a the name of a file containing a list of peak positions
se*t	r*egion	-l*ist=*list*	*list* is a the name of a file containing a list of regions
se*t	w*idth	-v*alue=(*value*)	specify a channel independent peak width (*value* = [1., 400.])
		-ca*librate=*file*	fit an up to fourth degree polynomial to the data set of widths in *file*
		-l*ist=*name*	take the widths from the peak list *name*
se*t	pa*rameter	-w*idth=(*values*)	specify the coefficients of a polynomial for the width *values*=[a0[,a1[,a2[,a3[,a4]]]]]
		-*kind*=(*values*)	define tail parameters *kind* might be one of the following: l*eft_peak_tail r*ight_peak_tail b*ackground_tail *values* migh be a list of polynomial coefficients up to 2nd degree for the amplitude a and decay coefficients w: (a0,w0) (a0,a1,w0,w1) (a0,a1,a2,w0,w1,w2)
		-s*tep=(*values*)	*values*: a list of up to 3 polynomial coefficients for the amplitude of the step

1) default

4.1.7 Fit Commands

com.	argument	comments
fit	*xmin,xmax,degree*	fit the region from *xmin* to *xmax* using a background polynomial of degree *degree*
fit	*xmin,xmax*	fit the region from *xmin* to *xmax* using a background polynomial of a degree as specified in the command "set background ..."
fit		fit the region from *xmin* to *xmax* as defined in the command "set region -fit=*xmin,xmax*" using a background polynomial of a degree as specified in the command "set background ..."
		the fit command gets the peak positions either from the peak search routine or from a peak list. Regions are ignored. *xmin,xmax* are interpreted as energies if there is an energy calibration and the command "set display -energy" has been given.
go		fit the region from *xmin* to *xmax* as defined in the command "set region -fit=*xmin,xmax*" Get the peak positions either from the peak search routine or from a peak list. Get the fit regions either from a region search routine or from a list of regions. Get the degree of the background polynomial either from the command "set background ..." or from a list of regions.

4.2 Data Formats

The available data formats for data sets can be asked for with the command "show limitations" or "help set files -format". Data sets can be in character or in binary mode.

4.2.1 Printable data sets

A printable data set is always in character mode. Numbers are recognized in the following representations:

```
123   123.   123.4   1.234e+2   1.234E+02   1234.d-01   1234.D-1
```

Negative numbers are accepted as well.
A line is called a header line if it contains other characters than the one above. Fractional or negative channel numbers are not allowed.

The following file formats are standard:

ascii, file extension: .ascii
the channel contents is listed without channel number. There can be one channel contents per line or several separated by spaces but a line should not be longer than 130 characters. The channel contents are assumed to be listed with increasing channel numbers starting with channel 0, however this channel is ignored. Any number of header lines may precede the data set but at least one header line is required. Several spectra ("sub spectra") might be in one file, separated by headers. Sub spectra are addressed by the sub spectrum number.

dat, file extension: .dat
The file is assumed to be an ASCII file with a header followed by lines containing

[channel [sub_spectrum]] contents

values in [] are optional. Spectra are assumed to start with channel 0 if channel numbers are not given however channel 0 is always ignored. Channel and sub spectra numbers must be integer numbers with either integer or float representation. The contents might be integer or floating point numbers also with exponential representation. Any number of header lines may precede the data set but at least one header line is required. Several spectra in one file are permitted. They must be separated by headers if the sub spectrum number is not present. They are addressed with the sub spectrum number.

daterr, file extension: .daterr
The file is assumed to be a printable file with a header followed by lines containing

[channel [sub_spectrum]] contents error

values in [] are optional. Spectra are assumed to start with channel 0 if channel numbers are not given however channel 0 is always ignored. Channel and sub spectra numbers must be integer numbers with integer or float representation. The contents and its error might be integer or floating point numbers also with exponential representation. Any number of header lines may precede the data set but at least one header line is required. Several spectra in one file are permitted. They must be separated by headers if the sub spectrum number is not present. They are addressed with the sub spectrum number.

4.2.2 Binary data sets

At present there are two binary formats integrated into *gaspan* : The Munich online data format (extension .osp) and a binary data format used in Strasbourg (extension .s).

4.2.3 Writing a spectrum read routine (to be written)

4.3 The Fit Function And Its Parameters

An ideal peak at channel x_i is decomposed into three contributions:

$$f_i = A_G\,\delta(x-x_i) \quad \text{full energy loss peak} \tag{1}$$
$$+A_G\,A_L\,e^{\frac{(x-x_i)}{\tau_L}}\,\Theta(x_i-x) \quad \text{left peak tail} \tag{2}$$
$$+A_G\,A_R\,e^{-\frac{(x-x_i)}{\tau_R}}\,\Theta(x-x_i) \quad \text{right peak tail} \tag{3}$$

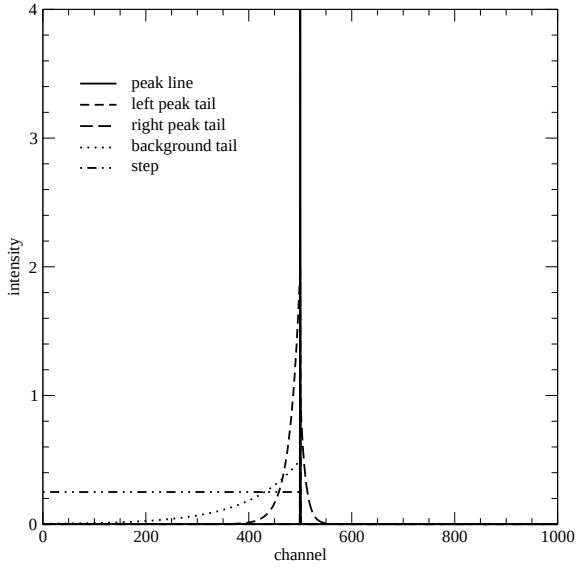
(see Philips and Marlow, NIM 137 (1976) 535 for a discussion)
The background of the peak is decomposed into a left tail of the peak with a large decay constant, a step function[7] and a polynomial up to fourth order in channel number

$$+A_G\,A_B\,e^{\frac{(x-x_i)}{\tau_B}}\,\Theta(x_i-x) \quad \text{background tail} \tag{4}$$
$$+A_G\,A_S\,\Theta(x_i-x) \quad \text{step function} \tag{5}$$
$$+\textstyle\sum_{j=0}^{4}\,a_j\,x^j \quad \text{background polynomial} \tag{6}$$

Note: all tail amplitudes are defined relative to the peak amplitude, allowing peaks with different amplitudes in a fit region.
This ideal response function is folded with the real response function

$$\frac{1}{\gamma\sqrt{\pi}}\,e^{(-x^2/\gamma^2)}$$

to yield (with a re-normalization of the tail amplitudes) the most general peak shape function in *gaspan* which can be utilized:



$$\begin{aligned} f(x) = \sum_{i=1}^{npeaks} A_G(i)\cdot\Bigg\{ & e^{-\frac{(x-x_i)^2}{\gamma_i^2}} \\ & +A_L\cdot e^{\frac{(x-x_i)}{\tau_L}}\cdot\mathrm{erfc}\left[\frac{(x-x_i)}{\gamma_i}+\frac{\gamma_i}{2\tau_L}\right] \\ & +A_B\cdot e^{\frac{(x-x_i)}{\tau_B}}\cdot\mathrm{erfc}\left[\frac{(x-x_i)}{\gamma_i}+\frac{\gamma_i}{2\tau_B}\right] \\ & +A_R\cdot e^{-\frac{(x-x_i)}{\tau_R}}\cdot\mathrm{erfc}\left[-\frac{(x-x_i)}{\gamma_i}+\frac{\gamma_i}{2\tau_R}\right] \\ & +A_S\cdot\mathrm{erfc}\left[\frac{(x-x_i)}{\gamma(i)}\right]\Bigg\} \\ & +\textstyle\sum_{l=1}^{degree} A_l\cdot(x-\langle x\rangle)^{l-1} \end{aligned} \tag{7}$$

with

$$\langle x\rangle = 0.5\,(x_{max}+x_{min})$$

[7] the step function is only in versions newer than March 2003.

The full width half maximum ($FWHM$) of a Gaussian is related to γ by

$$FWHM = 2\sqrt{\ln 2} \cdot \gamma = 1.6651092223 \cdot \gamma$$

All parameters but the coefficients A_j of the background polynomial are subject to constraints: The amplitudes $A_G(i)$, A_L, A_R, A_B, and A_S are defined as squares of the corresponding fit parameters

$$A_G(i) = a_G^2(i) \qquad A_K = a_K^2 \cdot A_{Ko}(i),\ K = L,\ R,\ B,\ S$$

limiting them to positive values and enhancing the speed of the fitting procedure. The splitting in the tail amplitudes allows the definition of energy dependent amplitudes.
The variation of the position of the Gaussian peak x_i as well as the width and decay constants are restricted to a certain variation range. The $\sin(x)$-function for such a restriction is recommended in the literature but care has to be taken because of its periodicity hence the function $psin(x) = \sin(\frac{\pi}{2}x)$ is used restricting $|x| \leq 0.95$ allowing still for a none zero slope[8].

The peak positions are restricted by:

$$x_i = x_{io} + \Delta x \cdot psin(px_i)$$

Δx is either set to the average $FWHM$ of the peaks or set to zero, fixing the positions of the peaks. The values of x_{io} are obtained either by a peak search routine or by input and by an analysis of the residuum of a fit. They are redefined after each fit cycle, starting each new cycle with $px_i = 0$.

The width of the peaks are restricted to the range

$$\gamma_i = \gamma_{io} \cdot (1 + b_i \cdot \Delta\gamma \cdot psin(p_\gamma))$$

γ_{io} i.e. $FWHM_{io}$ and $\Delta\gamma$ are input parameters in the range $[1., 400.]$ respectively $[0. : 99.]$[9]. $\Delta\gamma$ is common to all peaks as well as the parameter p_γ. The factor b_i can be set to 0 or 1 for each peak[10]. $FWHM_{io}$ can either be defined individually for each peak or is obtained from a width calibration input into the program.

The tail decay constants are restricted by

$$\tau_K = \tau_{Ko} \cdot (1 + \Delta\tau_K \cdot psin(p_K)) \qquad K = L, R, B$$

Again τ_{Ko} and $\Delta\tau_K$ are input parameters, the latter is restricted to $[0., 1.0]$.

The value for the parameters adapting the experimental data $y(x)$ are obtained by minimizing

$$\chi^2 = \sum_x \left(\frac{(y(x) - f(x))}{\Delta y(x)} \right)^2$$

[8] The function $psin(x) = x * (2 - |x|)$ with $|x| \leq 0.95$ and with the derivative $pcos(x) = 2\,(1. - |x|)$ for $|x| \leq 1$ and $pcos(x) = 0$ for $|x| > 1$ has been used in earlier versions. The present definition is used to comply with the previous definition.

[9] $[0. : 9.]$ in versions before March 2005

[10] Starting with versions March 2005: Each peak has its own width parameter $p\gamma_i$, b_i is obsolete

where $\Delta y(x)$ are the errors of the data points. The value of $\chi^2/degree\_freedom$ should be in the order of 1.
The elements of the sum

$$r(x) = \left\{ \frac{(y(x) - f(x))}{\Delta y(x)} \right\}$$

form a spectrum, the residue spectrum, which can be used to detect small peaks or deviations from a the Gaussian line shape.