gsas_language

Sven C. Vogel [*]
Manuel Lujan Jr. Neutron Scattering Center
Los Alamos National Laboratory, Los Alamos, New Mexico, 87545, U.S.A.
sven@lanl.gov

December 2010[†]

If you used gsaslanguage and found it helpful, please reference this paper:

S. C. Vogel,
gsaslanguage: a GSAS script language for automated Rietveld refinements of diffraction data,
J. Appl. Cryst. (2011). 44, 873-877
`http://dx.doi.org/10.1107/S0021889811023181`

[*]This work could not have been possible without the efforts of Drs Allen C. Larson and Robert B. Von Dreele to create and maintain GSAS. I am indebted to Nina J. Lane (Drexel University) for the majority of the effort to generate this manual - you wouldn't read this without her! The Lujan Neutron Scattering Center at the Los Alamos Neutron Science Center is funded by the Department of Energy's Office of Basic Energy Science. The Los Alamos National Laboratory is operated by the Los Alamos National Security LLC under the DOE Contract of DE-AC52-06NA25396.

[†]First drafted September 2009

Abstract

The commands of the gsas_language software are documented. The software utilizes the GSAS package and is intended to simplify automation of the analysis of large numbers of datasets, document refinement strategies and foster efficient exchange of refinement strategies between users and beam line scientists. Use of the bash shell and standard Unix text processing tools, available natively on Linux and Mac OSX platforms and via the free cygwin software on Windows systems, make this software platform independent.

Contents

Chapter 1

Introduction

Automation of data analysis of diffraction data is arguably one of the biggest challenges since the advent of high throughput facilities such as modern synchrotron or neutron facilities. Besides shear sample throughput, nowadays in many cases parametric studies are undertaken during which hundreds or thousands of runs at different pressures, temperatures, stress or other external parameters are acquired. Similarly, for studies of kinetics, data are taken at fixed short time intervals for prolonged periods, generating large numbers of runs that need to be analyzed. While Rietveld analysis [**?**, **?**] has become the standard way to extract crystallographic information from such data, automation of such data analysis is by no means a trivial task and frequently the obstacle for inexperienced or even experienced users of the Rietveld technique towards publication. While it was suggested to incorporate the parameters describing a series of datasets during a parametric study, e.g. the coefficient of thermal expansion or kinetic parameters of phase transformations, into the refinement of the whole dataset ([**?**], [**?**]), this technique can not be generalized and is not available in mainstream Rietveld packages. On the other hand, the role of beam line personnel at large scale facilities also includes, or should include, assistance of the users with data analysis. In view of the large numbers of user visits at many facilities nowadays, this task demands efficient knowledge transfer on e.g. the refinement strategy for the particular type of data from a particular instrument from experts to users with experience ranging from non-existent to very advanced, but for other instruments. The analysis of powder diffraction data using a complex model with tens or in some cases even hundreds of parameters makes this analysis technique somewhat and outlier from other materials characterization techniques with much smaller numbers of parameters. Therefore, for beginners it is often cumbersome to understand the complex interplay of parameters describing a diffraction dataset. While the general rule that the next parameter to be varied on should be the one that is deemed to have the largest contribution to difference curve holds for specific instruments and techniques, the sequence and type of parameters that need to be varied can be considered specific to the instrument. Guidance on the topic of refinement strategies exists for users of the Rietveld method (e.g. [?]), even for specialized topics like magnetic structure analysis using GSAS [?], but highly specialized instruments or sample environments will always require special details that need to be communicated to every user. The learning curve for the particular Rietveld codes can also be fairly steep for the beginner, and while this issue can be partially circumvented by hiding the input- and output-specific problems behind a graphical user interfaces (e.g. [?]), such interfaces do not assist in automation of the analysis. Some refinement techniques, such as the determination of site occupation factors as described by [?], require repetitive refinements with slight variations of e.g. the d-spacing range used in the refinement and could possibly be more wide-spread used if automation would be more readily available. Similarly, occasionally it is required to try several proposed structures and compare their match with a given dataset, providing another application of automation to ensure that the identical refinement strategy was applied with each structure model. Frequently, documentation of the applied refinement strategy is not available and users may even find themselves unable to reproduce a particular successful refinement. On the other hand, black-box automation of refinement with minimal user interaction may cause problems even for experienced users when the hard-coded refinement strategy does not work. Guidance of the user from e.g. the correlation matrix or the evolution of the reduced χ^2 or the parameter shift, that may indicate refinement problems, are typically not obvious for beginners but may contain very valuable indications as to what possible refinement problems might be. Finally, there is no standard way to communicate and exchange successful refinement strategies that would be useful to at least serve as a starting point for other users and problems or in order to remote advise to a user by an experienced user.

In this paper, we describe our approach to address these problems, allowing users to focus on the actual refinement rather than having to deal with the specific obstacles of adding histograms or phase information to the refinement. To address the above issues, we developed a script language that allows to program a refinement of diffraction data using the GSAS [?] package as the back-end. These scripts can be re-used with minimal modifications, either for analysis of similar datasets during a parametric study, or for similar experiments undertaken by other users. They generate an overview file for documentation of the refinement process as well as initial plots of graphs of any refined parameter. The scripts are using the bash shell, both to program commands controlling the analysis as well as programming the actual analysis, and are therefore available on Windows (using the cygwin package, available for free at www.cygwin.com), and for Mac or Linux without installing any software other than the collection of scripts and possibly Latex or gnuplot and of course GSAS. The bash environment in combination with Unix text processing tools allows validation of input (e.g. existence of input files such as instrument parameter files or data files), powerful on-the-fly extensions (no compilation necessary), as well as efficient extraction of results from various GSAS output files (list files, parameter-value-esd files etc.). If other script languages, e.g. Tcl/Tk are installed, crystallographic codes coded in these languages, can be easily included. Using a script language allows also to implement fairly specific checks and validations on common mistakes, such as refinement of the ZERO instrument parameter instead of DIFC in neutron time-of-flight refinements or variation of calibration parameters and lattice parameters, and to provide the user with feedback and warnings. The bash shell creates very little overhead to the execution of the GSAS commands, allowing for efficient analysis of large numbers of datasets. Conditional refinements, i.e. variation of certain phase specific parameters only above a certain threshold for the weight/volume fraction of the phase, can be implemented.

Chapter 2

Installation

2.1 Prerequisites

- This package aims at automating GSAS, therefore, a running GSAS is required. To test this, at the command line from which you wish to run this (mouse junkies - this will NOT be your cup of tea - but maybe we can pull you into the great land of command lines) change into a folder with GSAS data and try to plot it in rawplot by typing in `rawplot`. If this fails with a file not found error, GSAS is either not installed or not in the search path[1]. If it plots, but doesn't have text around the graph, the PGPLOT_FONT variable is not set. Another missing bit could be the GSAS variable, in which case GSAS wouldn't know where to look for the scattering length etc. All this is documented somewhat in the README file coming with GSAS, but who reads those these days... Please note that PC-GSAS and EXPGUI are setting the environment variables only for the current session, that is not enough for what we want to do here, so a running PC-GSAS or EXPGUI is not sufficient.

- The 2nd prerequisite is a working understanding of GSAS by the user. If you don't know what GSAS does and how it works, automation will be of little help, it might be even dangerous due to the GIGO problem[2]... We strongly suggest you work at least through the excellent tutorials that come with GSAS to get an idea what GSAS does. To encourage that, the examples we provide here are the tutorials from the GSAS manual cast into scripts.

[1]On windows systems, go to the control panel, advanced setup, environment variables, and add ";`c:\gsas\exe`" at the end of the search path. On Unix systems, add ":/usr/local/gsas/exe" to you PATH variable in something like `~/.bashrc` or so.

[2]garbage in, garbage out

2.2 GSAS installation

The best starting point to install GSAS and EXPGUI is `https://subversion.xor.aps.anl.gov/trac/EXPGUI/`. From there, go to the ”New installation instructions” for your operating system. In order to use the subversion automatic update and installation mode, on Unix systems you may have to add the lines

```
http-proxy-host = proxyout.lanl.gov
http-proxy-port = 8080
```

into the subversion server configuration stored in `/.subversion/servers`. Otherwise you will get an error message that subversion could not connect to the server at APS.

2.3 Linux or Windows?

We found that the same stuff runs faster when a given system runs under Linux. Here are some numbers on a 28 histogram refinement, including 8th order spherical harmonics texture. On both platforms, the same data was analyzed with the same script. The platform was a Dell Optiplex 960 (Quadcore Intel CPU, 3GHz, 4GB RAM). No other major tasks were running on the machine.

	Linux	Windows
OS	OpenSuse	Windows XP Professional
Version	11.2, Kernel 2.6.31.14-0.6 SMP	5.1.2600 SP3 Build 2600
total runtime [s]	2497	2652
average GENLES cycle time [s]	11.2(2.7)	12.4 (3.2)
Minimum GENLES cycle time [s]	4.1	4.8
Maximum GENLES cycle time [s]	18.5	20.1

On average, each GENLES cycle is 9% faster when running Linux with a maximum improvement of 17%. The overall runtime is reduced by 6.2% when running Linux.

2.4 Linux/Mac/Unix Installation

This should be fairly quick:

- Download the zip file with the latest scripts from `https://github.com/Svennito/gsaslanguage.git`.
- Copy the scripts into an appropriate folder and add it to the path[3]. Done.

[3]Add something like
```
PATH=$PATH:/usr/local/gsas/exe:~/gsaslanguage
export PATH
```
to your `/.bashrc` file if you installed GSAS in `/usr/local/gsas` and gsaslanguage in a folder of the same name in your home folder, adapt as needed by your installation. If you are there, also add

- If you want to get the very latest version and you have subversion installed on your system, navigate to the folder under which you want gsaslanguage installed, and type
  `svn checkout https://github.com/Svennito/gsaslanguage.git`

- Ok, maybe not completely. Make sure you have LaTeX installed (punch in `latex` at a command prompt, it should come up with a LaTeX prompt). If you want to plot the results of a multi-run analysis using gsas_plot_overview, make sure you have gnuplot installed, try typing `gnuplot` at a shell prompt.

2.5 Windows Installation

This is a bit more lengthy since many of the Unix standard software, like a bash shell, LaTeX and gnuplot don't come with the standard Windows.

- Copy the scripts into an appropriate folder, e.g. `c:\gsaslanguage`

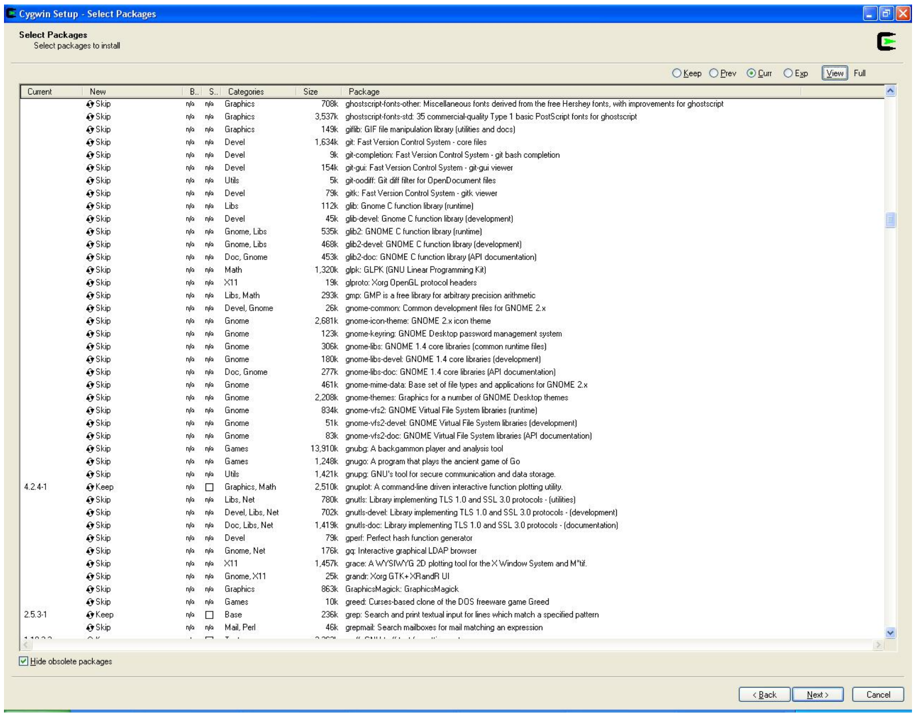
- Since we need a bash shell, install the cygwin package from (www.cygwin.com). Download the setup executable and store it in the same folder where you will install cygwin (e.g. `c:\cygwin`). That way it can be used for future additions and updates. Provide your internet options. If you are working at a U.S. government lab, choose a cygwin site at another government lab (.gov) since they are connected with a high-speed backbone. Once you see the "select packages" screen (see screenshot below), click on the "View" button in the top right corner until you see the "Full" view. Besides the basic cygwin, select to install also gnuplot and bc. Make sure to follow the UNIX end-of-line convention, if in doubt (getting weird error messages when running the gsas scripts such as "unexpected end of file" or so), run "dos2unix" on your script from the command line.

- Get the GSAS scripts from `https://github.com/Svennito/gsaslanguage.git`. Add the folder name to the search path (Control Panel – System – Advanced – Environment Variables – PATH)

- Install LaTeX (go to miktex.org to download the basic installation for a Windows system).

- You may have to install the LaTeX package fullpage manually if your firewall does not allow LaTeX to install it on the fly. If you are using Miktex (Basic MiKtex 2.7), install it into a subfolder of your Miktex installation and refresh the Filename database in the Miktex options.

`export gsas=/usr/local/gsas`
`export PGPLOT_FONT=$GSAS/pgl/grfont.dat`.Run `source ~/.bashrc` to activate your changes for the current shell, new shells will read it from the `~/.bashrc` file. Try `echo $PATH` to see if it was added.



Cygwin Setup - Select Packages

Select Packages
Select packages to install

Keep Prev [✓ Curr] Exp View Full

Current	New	B..	S..	Categories	Size	Package
	Skip	n/a	n/a	Graphics	708k	ghostscript-fonts-other: Miscellaneous fonts derived from the free Hershey fonts, with improvements for ghostscript
	Skip	n/a	n/a	Graphics	3,537k	ghostscript-fonts-std: 35 commercial-quality Type 1 basic PostScript fonts for ghostscript
	Skip	n/a	n/a	Graphics	149k	giflib: GIF file manipulation library (utilities and docs)
	Skip	n/a	n/a	Devel	1,634k	git: Fast Version Control System - core files
	Skip	n/a	n/a	Devel	9k	git-completion: Fast Version Control System - git bash completion
	Skip	n/a	n/a	Devel	154k	git-gui: Fast Version Control System - git-gui viewer
	Skip	n/a	n/a	Utils	5k	git-oodiff: Git diff filter for OpenDocument files
	Skip	n/a	n/a	Devel	79k	gitk: Fast Version Control System - gitk viewer
	Skip	n/a	n/a	Libs	112k	glib: Gnome C function library (runtime)
	Skip	n/a	n/a	Devel	45k	glib-devel: Gnome C function library (development)
	Skip	n/a	n/a	Gnome, Libs	535k	glib2: GNOME C function library (runtime)
	Skip	n/a	n/a	Gnome, Libs	468k	glib2-devel: GNOME C function library (development)
	Skip	n/a	n/a	Doc, Gnome	453k	glib2-doc: GNOME C function library (API documentation)
	Skip	n/a	n/a	Math	1,320k	glpk: GLPK (GNU Linear Programming Kit)
	Skip	n/a	n/a	X11	19k	glproto: Xorg OpenGL protocol headers
	Skip	n/a	n/a	Libs, Math	293k	gmp: GMP is a free library for arbitrary precision arithmetic
	Skip	n/a	n/a	Devel, Gnome	26k	gnome-common: Common development files for GNOME 2.x
	Skip	n/a	n/a	Gnome	2,681k	gnome-icon-theme: GNOME 2.x icon theme
	Skip	n/a	n/a	Gnome	123k	gnome-keyring: GNOME Desktop password management system
	Skip	n/a	n/a	Gnome	306k	gnome-libs: GNOME 1.4 core libraries (common runtime files)
	Skip	n/a	n/a	Gnome	180k	gnome-libs-devel: GNOME 1.4 core libraries (development)
	Skip	n/a	n/a	Doc, Gnome	277k	gnome-libs-doc: GNOME 1.4 core libraries (API documentation)
	Skip	n/a	n/a	Gnome	461k	gnome-mime-data: Base set of file types and applications for GNOME 2.x
	Skip	n/a	n/a	Gnome	2,208k	gnome-themes: Graphics for a number of GNOME Desktop themes
	Skip	n/a	n/a	Gnome	834k	gnome-vfs2: GNOME Virtual File System libraries (runtime)
	Skip	n/a	n/a	Gnome	51k	gnome-vfs2-devel: GNOME Virtual File System libraries (development)
	Skip	n/a	n/a	Gnome	83k	gnome-vfs2-doc: GNOME Virtual File System libraries (API documentation)
	Skip	n/a	n/a	Games	13,910k	gnubg: A backgammon player and analysis tool
	Skip	n/a	n/a	Games	1,248k	gnugo: A program that plays the ancient game of Go
	Skip	n/a	n/a	Utils	1,421k	gnupg: GNU's tool for secure communication and data storage.
4.2.4-1	Keep	n/a	☐	Graphics, Math	2,510k	gnuplot: A command-line driven interactive function plotting utility.
	Skip	n/a	n/a	Libs, Net	780k	gnutls: Library implementing TLS 1.0 and SSL 3.0 protocols - (utilities)
	Skip	n/a	n/a	Devel, Libs, Net	702k	gnutls-devel: Library implementing TLS 1.0 and SSL 3.0 protocols - (development)
	Skip	n/a	n/a	Doc, Libs, Net	1,419k	gnutls-doc: Library implementing TLS 1.0 and SSL 3.0 protocols - (documentation)
	Skip	n/a	n/a	Devel	79k	gperf: Perfect hash function generator
	Skip	n/a	n/a	Gnome, Net	176k	gq: Interactive graphical LDAP browser
	Skip	n/a	n/a	X11	1,457k	grace: A WYSIWYG 2D plotting tool for the X Window System and M*tif.
	Skip	n/a	n/a	Gnome, X11	25k	grandr: Xorg GTK+ XRandR UI
	Skip	n/a	n/a	Graphics	863k	GraphicsMagick: GraphicsMagick
	Skip	n/a	n/a	Games	10k	greed: Curses-based clone of the DOS freeware game Greed
2.5.3-1	Keep	n/a	☐	Base	236k	grep: Search and print textual input for lines which match a specified pattern
	Skip	n/a	n/a	Mail, Perl	46k	grepmail: Search mailboxes for mail matching an expression

✓ Hide obsolete packages

Figure 2.1: Selecting packages during Cygwin installation

2.6 Testing an Installation

On all platforms it's worth to try a few things to be sure GSAS and other programs are installed correctly before executing scripts. Try these:

- Type `expedt` and hit enter. EXPEDT should ask "Enter the experiment name for EXPEDT ¿". Hit Ctrl-c to exit. If this does not work, either GSAS is not installed or the GSAS executables are not in the search path.

- Change to the folder with the GSAS demo data by typing "cd $GSAS/example". If you get a message like "no such file or directory", your GSAS variable was not set properly. Try `echo $GSAS` to see if it points to the GSAS folder.

- If you got there, list the files in there by typing `ls` followed by enter.

- Go back to your home folder by typing `cd` followed by enter. Start RAWPLOT by typing `rawplot` followed by enter. Select a suitable screen option (a on Linux, c on Windows), take the default of the next question, load the `nickel.raw` file from the previous folder, e.g. type
`/usr/local/gsas/example/nickel.raw`

. Take the default of the next question by hitting enter, load the `inst_tof.prm` instrument parameter file by typing
`/usr/local/gsas/example/inst_tof.prm`
Type `b 1 p` followed by enter. If you don't see graphics, your PG graphics server is not running properly. If you see graphics, but no labels, your `PGPLOT_FONT` variable is not set properly. If you see graphics with text/labels, GSAS plotting routines will work.

- Type `latex` to see if you have LaTeX installed. If you get a LaTeX prompt, it works, if not you'll have to install it for your system.
- type `gnuplot` to see if you have gnuplot. Again, if you get a prompt, it works and if not you'll have to install it.

Chapter 3

Examples

In this chapter we try to familiarize new users with the concepts of the gsas_language. This section is neither intended as a tutorial for Rietveld refinements nor as a GSAS tutorial. There are several excellent sources out there to help with that, some of them are

- The tutorials in the GSAS manual
- GSAS/EXPGUI Alumina example `http://www.ncnr.nist.gov/xtal/software/expgui/tutorial3/`
- Argonne Powder Diffraction Crystallography Resources `http://www.aps.anl.gov/Xray_Science_Division/Powder_Diffraction_Crystallography/index.html`
- Armel Le Bail's website `http://www.cristal.org/`
- Advanced Certificate in Powder Diffraction on the Web `http://pd.chem.ucl.ac.uk/pdnn/pdindex.htm`
- Robert Dinnebier's Rietveld course `http://www.fkf.mpg.de/xray/html/rietveld_refinement.html`

This section is also not a introduction to usage of the bash shell. There are plenty of resources on the web, but maybe a few commands can get you started in case you are new to this:

- Command line completion with the tab key is your friend, e.g. if you type `rawp` and then hit the tab key, on most systems it will complete the word to `rawplot`. If there are more than one possibilities to complete, nothing will happen, but hitting tab twice will give a list of possible completions such that you can type one more letter and try again.
- `cd`: Changes the directory/folder, e.g. `cd mydata/nickel`.
- `cp`: Copies files, e.g. `cp fila1.txt fileb.txt` makes a copy of filea1.txt that is named fileb.txt.

- `ls`: Lists files, e.g. `ls -lrt` lists all files in the current folder in the long version (with time stamps, size etc.), sorted by time in reverse order - so you get the latest files last and can see what has happened during e.g. execution of a script.
- `man`: The Unix help system, provides information for a command, e.g. `man ls` lists the options and explanations for the `ls` command.

3.1 Ready to roll?

To check if you are readty to go, follow these steps.

- Open a shell (either bash on Unix/Mac or cygwin bash on Windows).
- Type `expedt` and hit enter. If you are prompted for a filename, you are good to go. If a file not found message occurs, GSAS is either not installed or not in the search path. Go back to installation.
- Change to a folder with GSAS data (using the `cd` command, you can use the tab key to complete command lines), type `rawplot` and try to plot a data set. If you do not see the data and labels etc. the `PGPLOT_FONT` variable is not set. Go back to installation.
- Type `latex`. If a file not found message occurs, go back to installation, otherwise type `q` to get out of LaTeX.
- Type `gnuplot`. If a file not found message occurs, go back to installation, otherwise type `quit` to get out of gnuplot.

If all of the above work, you are ready to roll.

3.2 Neutron time-of-flight data: The nickel demo

A good start is probably running the nickel demo. This demo also serves as the skeleton of a refinement strategy for neutron time-of-flight data. It is based on the first tutorial in the GSAS manual; it basically does the same steps described there – except it takes a few seconds to run rather than the 2 hours for the average GSAS newbie or 5 minutes for the experienced GSAS user. And it produces a few graphs. We highly recommend you go through that example as described in the GSAS manual - without knowing GSAS the gsas_language software is of limited use.

Ok, here we go. We assume you installed everything properly. Download the nickel tutorial zip file from the gsas_language website and unzip it into a location of your choice. For your reading convenience, here is the listing of the main script `gsas_analyze`.

```
# initialize the EXP file with the file name of this project
gsas_initialize nickel "Nickel Powder Data Example"
```

```
# add a phase named Nickel with a space group and lattice parameter
gsas_add_phase "Nickel" "f m -3 m" "3.5234"

# add two histograms to the refinement which are banks 2 and 3 of the dataset
# and have a d-spacing range from 0.25 to 3 Angstrom
gsas_add_histogram nickel.raw inst_tof.prm 2 0.25 3
gsas_add_histogram nickel.raw inst_tof.prm 3 0.25 3

# add an atom to phase #1 (parameter are same as during EXPEDT atom addition)
gsas_add_atom 1 "NI 0 0 0 1 / i 0.004"

# do 2 refinement cycles
gsas_refine 2

# change the background for histograms 1 2 to function #1 with 16 coefficients
gsas_change_background 1 1 16
gsas_change_background 2 1 16
gsas_refine 5

# vary the diffractometer constant DIFC
gsas_vary_DIFC 1 C
gsas_vary_DIFC 2 C
gsas_refine 9

# vary the peak profile parameter sigma1 (peak width in TOF experiment)
gsas_vary_sigma1 1 1
gsas_vary_sigma1 2 1
gsas_refine 9

# vary the thermal motion parameter Uiso of phase 1, atom 1, with a damping of 5
gsas_vary_atom 1 1 u 5
gsas_refine 9

# fix the thermal motion parameter of phase 1's atom 1
gsas_vary_atom 1 1 -u

# vary the absorption (function #1 is used) for histogram 1 and 2 with damping 5
gsas_vary_absorption 1 1 y 5
gsas_vary_absorption 2 1 y 5
gsas_refine 9

# clean up and generate the overview file
gsas_done

# Make a copy to generate some dummy data for a graph
```

```
gsas_copy_expfile NICKEL NICKEL_LATTICE_TOO_BIG "same data, just messed up a little"

# change the lattice parameter - this now works in the new file!
gsas_change_lattice 1 3.53
gsas_refine 9

# clean up and generate the overview file
gsas_done

# Make another copy to generate more dummy data for a graph
gsas_copy_expfile NICKEL NICKEL_LATTICE_TOO_SMALL "same data, just messed up a little"

# change the lattice parameter - this now works in the new file!
gsas_change_lattice 1 3.52
gsas_refine 9

# clean up and generate the overview file
gsas_done

# make some graphs, just for the heck of it (if this analysis would have more runs, th
gsas_plot "   1HSCL" "   2HSCL"
gsas_plot "   1ABS1" "   2ABS1"
gsas_plot "DIFC  1 "
gsas_plot "DIFC  2 "
gsas_plot "Rwp     "
gsas_plot "Rp      "
gsas_plot "1  1PF 6" "1  2PF 6"
gsas_plot_overview
```

To run this, open a bash shell and change to that location. In there, start the script Unix-style with
`./gsas_analyze`

You can look at this file with the editor of your choice (if that choice is wordpad or any other Windows editor and you save the file, make sure to run `dos2unix` on it before executing it again). If really everything is installed nicely (GSAS, gnuplot, LaTeX, ghostscript), you should get four Acrobat PDF files. Three are the results of the three refinements (NICKELxxx.pdf) and one is an overview of the parameters (overview.pdf). The script gsas_analyze is documented with comments (starting with # in bash scripts), so not much help needed there. Some more information on the commands is provided in the GSAS Refinement Commands section below.

The result PDF files for the refinements show the data, what was changed, and refinement progress indicators (reduced CHISQ, final variable sum shift) for each refinement step. The data is shown as the usual (non normalized) measured data with fit as well as the plots that are generated if you answer "yes" when POWPLOT asks whether you want to see the error analysis. That allows to get a feeling of what the variation of a certain parameter does to the refinement. At the end of the document, you find a list of the refined parameters with uncertainties, which is exactly the PVE file. The PVE file is a good place to collect the parameters of interest for parameter studies etc., which is why it is not deleted at the end of the refinement. The refinement overview files allow you to document a refinement strategy, and they can also tell whether a parameter helps (i.e. the CHISQ goes down substantially) or not. The plot overview in overview.pdf is just meant as an example to show how results of parametric studies can be quickly visualized – not very meaningful here, but it hopefully gives you an idea how to use it.

3.3 X-ray data: The fluorapatite demo

As a similar skeleton for the refinement of laboratory X-ray data, we show the script that corresponds to the GSAS tutorial on fluorapatite, which you can also download from the gsas_language website.

```
# initialize the EXP file named FAP with a meaningful title
gsas_initialize FAP "GSAS demo for X-ray data, flourapatite"

# add a phase with name, space group and lattice parameters
gsas_add_phase "Flourapatite" "P 63/m" "9.37 6.88"
#read -p "wait after adding phase..."

# add the histogram
gsas_add_histogram fap.xra inst_xry.prm 1 0.8 6
#read -p "wait after adding histograms..."

# add the atoms
gsas_add_atom 1 "CA 1/3   2/3   0     1 CA1 i 0.005"
gsas_add_atom 1 "CA 0.242 0.992 1/4   1 CA2 i 0.005"
gsas_add_atom 1 "P  0.397 0.367 1/4   1 P3  i 0.005"
gsas_add_atom 1 "F  0     0     1/4   1 F4  i 0.010"
gsas_add_atom 1 "O  0.325 0.485 1/4   1 O5  i 0.005"
gsas_add_atom 1 "O  0.591 0.469 1/4   1 O6  i 0.005"
gsas_add_atom 1 "O  0.340 0.258 0.070 1 O7  i 0.005"
#read -p "wait after adding atoms..."

# change the background for histogram 1 to function 5 with 3 coefficients
gsas_change_background 1 5 3
gsas_refine 5 #noplot
#read -p "pause after background..."

# vary the lattice parameter
gsas_vary_lattice 1 y
gsas_refine 5 #noplot
gsas_refine 5 #noplot
#read -p "pause after lattice..."

# vary profile parameters
gsas_vary_LX_LY_trns 1 1 "y y n"
gsas_vary_asym_shft_GP 1 1 "n y n"
gsas_refine 5 #noplot
#read -p "pause after LX, LY and shft..."

gsas_vary_asym_shft_GP 1 1 "y y n"
gsas_refine 5 #noplot
#read -p "pause after asym..."

gsas_change_profile_cutoff 1 1 0.5
gsas_refine 5 #noplot
#read -p "pause after profile cut-off..."

gsas_vary_stec_ptec_sfec 1 1 "y y n"
gsas_refine 5 #noplot
#read -p "pause after stec and ptec..."

gsas_vary_atom 1 1:7 u
gsas_vary_atom 1 1:7 x
gsas_refine 5 #noplot
#read -p "pause after atomic positions and thermal parameters..."

gsas_fourier_maps
gsas_calc_bond_length 1 3
gsas_done
```

In this example you see several lines starting with `#read -p...`. This is some kind of low-level debugging. When the # is removed, the line is not treated as a comment any more and the script will stop there, display the message in the parentheses and continue when the enter key is hit. If one sees a problem, either with the fit such as divergence, or with the script, such as file not found or so, the user can abort the script using `Ctrl-C`. It is good practice to check that things are running as intended upon the first execution of each command - watch the reduced χ^2 and the sum of the shifts to check for oscillations and convergence, make sure filenames are read etc. Once a step is successful, comment out the `read` command.

Chapter 4

The commands

The following sections describe the commands of the gsas_language. They are arranged by topic and alphabetically therein. An alphabetic list is also provided. For detailed descriptions of the GSAS functions utilized, refer to the GSAS function and references therein. Introduction to more complex concepts such as constraints, texture etc. are beyond the scope of this document.

4.1 Complete list of commands

```
gsas_add_atom
gsas_add_diffuse_scattering
gsas_add_histogram
gsas_add_phase
gsas_calc_bond_length
gsas_change_atom
gsas_change_background
gsas_change_DIFC
gsas_change_Fobs_extraction_flag
gsas_change_lattice
gsas_change_phase_flag
gsas_change_phase_scale
gsas_change_sigma1
gsas_change_sigmas
gsas_change_texture
gsas_constrain_atom
gsas_constrain_phase
gsas_constrain_sigma1
gsas_convert_atom_thermal
gsas_copy_expfile
gsas_delete_atom_constraint
gsas_delete_sigma1_constraint
gsas_done
```

```
gsas_exclude_region
gsas_fourier_maps
gsas_initialize
gsas_plot
gsas_plot_bond
gsas_plot_histograms
gsas_plot_overview
gsas_prepare_mem
gsas_read_phase
gsas_refine
gsas_replace_histogram
gsas_simulate_histogram
gsas_single_peak_fits
gsas_single_peak_fits_make_list
gsas_vary_absorption
gsas_vary_atom
gsas_vary_background
gsas_vary_DIFC
gsas_vary_diffuse_scattering
gsas_vary_histogram_scale
gsas_vary_lattice
gsas_vary_phase
gsas_vary_sigma1
gsas_vary_texture
gsas_vary_UVW
gsas_waterfall_add
gsas_waterfall_plot
```

4.2 Beginning and end of refinement

gsas_copy_expfile

Copies an existing, presumably successful refinement into a new EXP file. This can be used for sequential refinements to use the previous time/temperature/pressure/strain etc. data file as a template for the next one, assuming that the changes will be small. With this, in longer sequences of refinements one does not need to start from scratch for each run. After the EXP file was copied, use the gsas_replace_histogram command to replace the data files referenced in the EXP file, so the EXP file can actually work with the new data. This command replaces gsas_initialize for subsequent refinements. Therefore, gsas_initialize must NOT be called for subsequent runs.

Syntax:

```
gsas_copy_expfile <old_filename> <new_filename> <new_title>
```

Parameters:

- `<old_filename>`: Filename of the old EXP file that is the source. No extension needed, ".EXP" will be added.
- `<new_filename>`: Filename of the new EXP file that will be created. No extension needed, ".EXP" will be added.
- `<new_title>`: Title string for the new refinement which will appear in the head of e.g. histogram plots.

Example:

```
gsas_copy_expfile 0TESLA_BEFORE 1TESLA "Wang, Bi, 1 Tesla magnetic
field"
gsas_replace_histogram "0Tesla_00deg.gda" "1Tesla_00deg.gda"
gsas_replace_histogram "0Tesla_10deg.gda" "1Tesla_10deg.gda"
gsas_replace_histogram "0Tesla_45deg.gda" "1Tesla_45deg.gda"
```

Copies an EXP file 0TESLA_BEFORE.EXP to 1TESLA.EXP and assigns a new title. After that, three data files used in the first EXP file are replaced with the data files of the 2nd EXP file.

gsas_done

This command finishes a refinement. Plots of χ^2, normalized shifts, time per iteration and number of variables are generated using gnuplot and included in the final Acrobat PDF file describing the refinement. Several checks (negative thermal motion, parameters that are zero within their error bars) are performed and added to the refinement report. Temporary files are deleted. If a refinement is aborted and the user wishes to see the overview PDF file, gsas_done can be issued from the command line.

Syntax:

```
gsas_done
```

Parameters:

- None.

Example:

```
gsas_done
```

This would finish a refinement.

gsas_initialize

This command initializes a new refinement by generating a new GSAS EXP file. It deletes temporary files of possible previous refinements and initializes the documentation file of this refinement.

Syntax:

```
gsas_initialize <EXP filename> <title>
```

Parameters:

- `<EXP filename>`: The filename for the GSAS EXP file and assorted files produced by POWPREF, GENLES, etc. Filename must not have an extension. Filename will be automatically converted to capital letters since GSAS expects capital letters on UNIX platforms.
- `<title>`: An arbitrary title to be displayed in the POWPLOT plots etc.

Example:

```
gsas_initialize SIO2_FURNACE_$1C "SiO2, furnace, $1C"
```

When a script with this line is called with a parameter 100, e.g. the GSAS data file for a run at 100C, named 100C.gda, the $1 occurences will be replaced with "100" by bash, generating a EXP filename SIO2_FURNACE_100C.EXP with a run title label "SiO2, furnace, 100C".

gsas_replace_histogram

Replaces all references to a filename with a new filename in the current EXP file. This is typically done after a gsas_copy_expfile command, using a previous refinement as a template for a new refinement. Since the extension for data files in GSAS is not fixed, the extension of the data file has to be provided. This runs a simple text search and replace operation on the current EXP file, so in principle it can be used for other purposes, such as replacing instrument parameter files.

Syntax:

```
gsas_replace_histogram <old_filename> <new_filename>
```

Parameters:

- `<old_filename>`: Filename of the old data file that needs to be replaced.

- `<new_filename>`: Filename of the new data file that will be referenced after this operation.

Example:

```
See example for gsas_copy_expfile.
```

See example for gsas_copy_expfile.

4.3 Adding histograms and phases

gsas_add_atom

Adds an atom to an existing phase of the refinement. The phase can be either added using the gsas_read_phase or gsas_add_phase commands.

Syntax:

```
gsas_add_atom <phase number> "<GSAS atom parameter sequence>"
```

Parameters:

- `<phase number>`: The number of the phase in the main EXP file to which this atom is to be added.

- GSAS atom parameter sequence: Space-separated sequence of element symbol, x, y, z coordinates, site occupation factor, atom label (or forward slash for default), thermal motion flag and value of thermal motion parameter(s). This is the same sequence a user would provide following a "i n" command in the atoms menu in EXPEDT.

Example:

```
gsas_add_atom 1 "NI 0 0 0 1 / i 0.004"
```

This would add a nickel atom to phase 1 on position x=0, y=0, z=0 that is fully occuped (FRAC=1) with the default name that GSAS assigns (the element symbol plus the number of the atom in the sequence). The atom would have isotropic thermal motion with a starting value of $U_{iso} = 0.004$.

gsas_add_histogram

Adds a phase to the refinement by adding the phase information "manually" rather than loading all information from a file. Similar to the addition in GSAS, no atom information is provided in this step (see gsas_add_atom). The existence of the space group and the lattice parameters in the main EXP file is verified after adding this information. Existence of the data and the instrument parameter file are checked before the insertion of the histogram data is attempted. Existence of the histogram data filename in the EXP file is checked after the insertion. This command may be used multiple times if multiple histograms are required for a refinement.

Syntax:

```
gsas_add_histogram <data_file> <par_file> <bank> <min_d> <max_d>
```

Parameters:

- `<datafile>`: Filename of the data file with the histogram data to be loaded (GSAS data file format).
- `<par_file>`: Filename of the instrument parameter file required to interpret the histogram data.
- `<bank>`: Bank number of the histogram to be read.
- `<min_d>, <max_d>`: Minimum and maximum d-spacing to be used for this histogram.

Example:

```
gsas_add_histogram nickel.raw inst_tof.prm 2 0.25 3
```

This would add histogram number 2 from the GSAS data file nickel.raw, using the GSAS instrument parameter file inst_tof.prm. The minimum d-spacing for this histogram would be set to 0.25Å and the upper d-spacing limit would be 3Å.

gsas_add_phase

Adds a phase to the refinement by adding the phase information "manually" rather than loading all information from a file. Similar to the addition in GSAS, no atom information is provided in this step (see gsas_add_atom). The existence of the space group and the lattice parameters in the main EXP file is verified after adding this information.

Syntax:

```
gsas_add_phase "<phase name>" "<space group>" "lattice parameters"
```

Parameters:

- `phase name`: The identifier for the phase in GSAS.
- `space group`: The space group of the crystal structure.
- `lattice parameters`: The lattice parameters for the crystal structure.

Example:

```
gsas_add_phase "Nickel" "f m -3 m" "3.5234"
```

This will add a phase with the label `Nickel` that has the cubic space group $Fm\bar{3}m$ and a lattice parameter of 3.5234Å.

gsas_exclude_region

Excludes a region (in the native x-axis unit of the data, i.e. time-of-flight or 2θ) from the refinement. To convert from d-spacing to the native x-axis unit, either convert using the equations in the manual or use the EXPEDT minimum d-spacing function (enter the d-spacing to converted, read the value in the native x-axis unit, then enter the true minimum again). Make sure that an extra excluded region does not overlap with the upper or lower limit, this will cause instabilties as e.g. background functions to diverge.

Syntax:

```
gsas_exclude_region <histogram> <from> <to>
```

Parameters:

- `histogram`: The histogram number for which the excluded region is to be inserted.
- `from`: The lower limit of the excluded region in native units of the x-axis.
- `to`: The upper limit of the excluded region in native units of the x-axis.

Example:

```
gsas_exclude_region 1 16 18
```

Excludes for histogram 1 the region from 16 to 17, either 2θ or time-of-flight in ms. For the HIPPO neutron time-of-flight diffractometer, this command would exclude a background generated from the adjacent WNR facility from around 16-18 ms (1/60 of a second=16.6 ms) in backscattering detector bank 1.

gsas_read_phase

Adds a phase to the refinement by loading a crystal structure from a file. The file with the crystal structure information has to be in GSAS format, CIF files are currently not supported. The existence of the phase file is checked before loading is attempted. The number of the phase during the refinement is determined by the sequence by which the phases are added to the refinement.

Syntax:

```
gsas_read_phase <filename> <phase number>
```

Parameters:

- `filename`: The name of the file (with extension, e.g. `.exp`) from which the crystal structure information is to be read.
- `phase number`: The number of the phase in the GSAS EXP file, typically 1.

Example:

```
gsas_read_phase "Boron_beta_68106.exp" 1
```

This would read phase information (lattice parameters, space group, atom parameters such as positions and thermal motion parameters) from a file name `Boron_beta_68106.exp` (case sensitive!). In that file, phase number 1 would be read.

4.4 Crystal structure refinement

gsas_add_diffuse_scattering

This function allows to use the diffuse scattering function to model a contribution to the "background" by e.g. an amorphous phase.

Syntax:

`gsas_add_diffuse_scattering <histogram> <function type> <amplitude> <radius> <Uiso> <atom type 1> <atom type 2>`

Parameters:

- `<histogram>`: The histogram for which the diffuse scattering term will be added.
- `<function type>`: The diffuse scattering function to be used. Possible options are "1" for diffuse scattering from substitional disorder (e.g. an alloy), "2" for positional disorder (e.g. a glass), and "3" for diffuse scattering caused by vibrational correlation. See the GSAS manual (page 131) for more information.
- `<amplitude>`: The starting value for the amplitude for the particular diffuse scattering term. "1" typically is a good starting value.
- `<radius>`: The starting value for radius or atomic distance for the particular diffuse scattering term.
- `<Uiso>`: The starting value for the isotropic thermal motion parameter for the atoms of this particular diffuse scattering term.
- `<atom type 1>`: The first atom of the pair, has to be a valid element in GSAS.
- `<atom type 2>`: The second atom of the pair, has to be a valid element in GSAS.

Example:

```
EXAMPLE
```

COMMENTS FOR EXAMPLE

gsas_change_atom

This command is used to change values of atom parameters, such as position or thermal motion, but also the atom type to change the element/isotope for a given atom or atom sequence. Using the EXPEDT convention to address several atoms at once, e.g. in the sequence of atoms within a phase or by their element type, multiple atoms can be affected at once.

Syntax:

`gsas_change_atom <phase> <atom range> <parameter name> <value>`

Parameters:

- `phase`: The number of the phase to which the atoms belong that need to be changed.
- `atom range`: An atom range, e.g. the number of a single atom, a sequence in the GSAS atom list of the phase, or a specific type (element or isotope). This parameter can be anything EXPEDT understands.
- `parameter name`: The parameter name, one of `X, Y, Z, UISO, FRAC, U11, U22, U33, U12, U13, U23, TYPE, NAME`.
- `value`: The new value for the parameter of the atoms addressed with the previous parameters.

Example:

```
gsas_change_atom 1 NI UISO 0.004
```

This would change the isotropic thermal motion parameter `UISO` of all atoms of the type `NI` in phase 1 to 0.004Å^2.

gsas_change_background

This command changes the type of the background function used for a given histogram and the number of parameters for that function.

Syntax:

```
gsas_change_background <histogram> <background function> <number
of parameters>
```

Parameters:

- `histogram`: The number of the histogram for which the background will be changed.
- `background function`: The number of the function to be used to describe the background in this histogram. Currently, valid numbers are 1, 2, 4, 5, 6, 7, and 8 (see page 129 in the 2004 or 'modern' GSAS manual).
- `number of parameters`: The number of parameters or coefficients to be used for the selected background function. GSAS supports up to 36 parameters, typically much less are sufficient.

Example:

```
gsas_change_background 2 1 16
```

Changes the background of histogram 2 to GSAS background function number 1 with 16 parameters.

gsas_change_Fobs_extraction_flag

This command allows to change the way the observed structure factors are extracted from the measured data. This allows essentially to switch between

- Rietveld mode: All peak intensities in the model are constrained by the atomic position, thermal motion, texture etc.
- Le Bail mode: Each peak intensity is an independent variable, like in a single peak fit, but with all peak positions constrained by the space group and lattice parameters and peak profile parameters constrained by the implemented functions for them.

While Rietveld mode is obviously useful to perform crystal structure refinements, Le Bail mode is useful when a good structural model is not available. Examples are reliable extraction of lattice parameters in the presence of texture or initialization of lattice and peak profile parameters during the early stages of a crystal structure refinement such that they can be fixed during refinement of intensity dependent parameters later to reduce the number of parameters varied and make the fit more stable. Since in Le Bail mode all peak intensities are independent, all peak intensity dependent parameters such as atomic positions, thermal motion and texture and also the histogram and phase scale factors need to be fixed. Otherwise divergence will occur. Also, each run of POWPREF will reset the peak intensities to their initial values. It is therefore advisable to have lattice parameters etc. fixed when starting the refinement. Once the intensities are properly initialized, lattice parameters can be refined but then POWPREF should only be re-run if they are fixed again.

The peak intensities are not part of the variable set counted in the "final variable sum((shift/esd)**2)" in GENLES. Therefore, a stop in reduction of the "Reduced CHI**2" is a better indicator as to when convergence was achieved. Oscillations are frequent in Le Bail refinement and damping might need to be included. Typically, fixing the refined parameters, then running POWPREF with background only followed by refining the previously varied parameters again further reduces the CHISQ.

Syntax:

```
gsas_change_Fobs_extraction_flag <histogram> <extraction flags>
```

Parameters:

- `histogram`: The number of the histogram for which the extraction flags should be changed.
- `extraction flags`: Same as in EXPEDT: ”n” for Rietveld, ”y n” for Le Bail with starting intensities weighed by the current structural model, and ”y y” with initial intensities all set to 1.0. For multiple phases, the right number of ”n”, ”y n”, or ”y y” has to be added into a single string enclosed by double quotes, e.g. ”y n y n y n” for setting three phases to a weighed Le Bail mode.

Example:

```
gsas_change_Fobs_extraction_flag 1 "y n"
```

This command will set the extraction flags for histogram number 1 to Le Bail mode with a model-based initial intensity guess.

gsas_change_lattice

Changes the lattice parameters of a given phase. This might be necessary for instance when the parameters from the phase file read with gsas_read_phase do not match the experimental data, e.g. due to differences in the chemistry.

Syntax:

```
gsas_change_lattice <phase> <lattice parameters>
```

Parameters:

- `phase`: The number of the phase for which the lattice parameters are changed.
- `lattice parameters`: The new lattice parameters. The number of parameters depends on the space-group of the phase, e.g. one for cubic, two for hexagonal, or six for triclinic crystal structures. The parameters are input as in GSAS, e.g. first a, then c for hexagonal, or in the sequence $a, b, c, \alpha, \beta, \gamma$ for triclinic phases. The parameters can be either given as individual numbers, or as a single string encapsulated in double-quotes.

Example:

```
gsas_change_lattice 1 3.14 11.05 OR gsas_change_lattice 1 "3.14
11.05"
```

Sets the lattice parameters of phase 1 to 3.14A and 11.05A. Since there are two parameters given, the phase is either hexagonal, tetragonal, or rhombohedral.

gsas_change_profile_cutoff

Changes the profile cut-off for which the peak profile of a given peak is computed. Peak profiles are defined from $-\infty$ to $+\infty$. To keep a finite computation time, the peak profile computation is cut-off at a certain fraction of the maximum peak intensity. In older instrument parameter files or for low statistics data, this value can be 1.00% and one might wish to change it to 0.10%.

Syntax:

```
gsas_change_profile_cutoff <histogram> <phase> <cut-off value>
```

Parameters:

- `histogram`: The number of the histogram for which the cut-off should be changed.
- `phase`: The number of the phase for which the cut-off should be changed.
- `cut-off value`: The new value for the peak cut-off intensity, in a per-cent value of maximum intensity.

Example:

```
gsas_change_profile_cutoff 2 1 0.1
```

Changes the peak cut-off value for histogram 2, phase 1, to 0.1%.

gsas_change_texture

Changes for a given phase the order of the spherical harmonics representation of the crystal orientation distribution function (ODF - the texture or preferred orientation of a phase) and the sample symmetry of the texture. In general, this function should only be used when sufficient sample directions are probed in a multi-histogram refinement. If this is not the case, any intensity-dependent parameters (atomic structure, phase fractions) will be affected and this may lead to false results. Imposing a sample symmetry will reduce the number of required sample directions/histogram, however, one may wish to verify the existence of such symmetry independently, i.e. with a full texture measurement. In case a symmetry is imposed, the center of symmetry, e.g. fiber axis, has to be in the center of the resulting pole figures and the sample orientation angles may have to be changed.

Syntax:

```
gsas_change_texture <phase> <order> <symmetry>
```

Parameters:

- `phase`: The number of the phase for which to change the texture description.
- `order`: The order of spherical harmonics to be used, must be an even number.
- `symmetry`: The sample symmetry of the spherical harmonics description of the ODF. Valid numbers are 0 (cylindrical symmetry or single fiber texture), 1 (triclinic or no symmetry), 2 (2/m or shear texture), or mmm (orthorhombic or rolling texture).

Example:

```
gsas_change_texture 1 8 1
```

Changes the ODF description for phase 1 to an 8th order spherical harmonics series expansion with no sample symmetry.

gsas_constrain_atom

This command is used to set a given atom parameter (e.g. occupancy, position, thermal motion) for different atoms in a phase to be constrained together during refinements. By default, this would set the different atom parameters to vary together (for example, if you wanted the thermal motion of atoms 2 and 3 to be constrained together since they are crystallographically similar and the refinement diverges otherwise). Note that if you want two atom parameters to be equal, they must start out equal before they are varied when constrained together, otherwise they will vary proportional to one another. In the case of constraining the occupancy, the values will be constrained such that they alway add up to the sum of the their starting occupancies (usually 1). More complex and multi-phase constraints, i.e. constraints affecting atoms from different phases, are also possible. This allows for instance in a mix of two oxides to constrain the oxygen atoms together. Note that differently from the `gsas_change_atom` etc. commands addressing by atomic species is not allowed, whereas sequences, e.g. `1:3`, are possible.

Syntax:

```
gsas_constrain_atom <phase> <atom prm> <atom 1> <atom 2> <atom
3>...

OR
gsas_constrain_atom multi "<constraint term 1>" "<constraint term
2>" ...
```

Parameters:

- `phase`: Phase number for which atom parameters should be constrained. If `phase` is `multi`, constraints over multiple phases may be defined. In this case all following parameters will have to be entries (in quotes or double-quotes) as given during definition of these constraints in EXPEDT (see example).
- `atom prm`: Atom parameters which should be constrained, one of `X, Y, Z, UISO, FRAC, U11, U22, U33, U12, U13, U23`.
- `atom i`: Atom numbers of the atom set that should be constrained, listed separately.
- `constraint term i`: A constraint term as required by EXPEDT consisting of phase number, variable identifier `atom prm`, atom or atom sequence and multiplier.

Example:

```
gsas_constrain_atom 1 U11 2 3
gsas_constrain_atom multi "1 UISO 1:3 1" "2 UISO 1 1"
```

The first example sets the thermal motion parameter U11 for atoms 2 and 3 in phase 1 to be constrained together. The second example constrains the Uiso of atoms 1 to 3 of phase with a multiplier of 1 to the atom 1 of phase 2, also with a multiplier of 1 (the multiplier is the last entry).

gsas_constrain_phase

Constrains a phase to have a constant weight fraction in all histograms. In most cases, this should be set before varying a phase during refinements. Excemptions might be refinements of histograms originating from different instruments and only some of them show an extra phase.

Syntax:

```
gsas_constrain_phase <phase>
```

Parameters:

- `phase`: Phase number for which weight fraction should be constrained.

Example:

```
gsas_constrain_phase 2
```

Sets the weight fraction of phase 2 to be equal in all histograms.

gsas_convert_atom_thermal

Converts thermal motion parameters of atoms from isotropic (UISO) to anisotropic (U11, U22, U33, etc) or vice versa.

Syntax:

```
gsas_convert_atom_thermal <phase> <atom range> <flag>
```

Parameters:

- `phase`: Number of the phase for which parameters should be changed.
- `atom range`: Atom(s) for which thermal factors should be changed. This is a valid GSAS atom range, e.g. the number of a single atom, a sequence in the GSAS atom list of the phase, or a specific type (element or isotope). This parameter can be anything EXPEDT understands. `<t>`, `<s>` or `<s1:s2>`
- `flag`: Flag for thermal factors: `i` for isotropic or `a` for anisotropic.

Example:

```
gsas_convert_atom_thermal 1 1:4 a
```

This would convert the thermal motion parameters to anisotropic for atoms 1, 2, 3 and 4 of phase 1.

gsas_refine

Performs the actual refinement by calling POWPREF and GENLES for a given number of refinement cycles. Each call to this command adds a section to the refinement protocol generated at the very end of the refinement.

Syntax:

```
gsas_refine <cycles> [noplot] [nopowpref]
```

Parameters:

- `cycles`: The number of refinement cycles in GENLES to be performed.
- `noplot`: If the word "noplot" is added after the number of cycles, the time-consuming generation of plots using POWPLOT is skipped.
- `noplot`: If the word "nopowpref" is added after the number of cycles, running of `powpref` will be skipped. This is useful during Le Bail fits during which re-setting the peak intensities by `powpref` may lead to divergence.

Example:

```
gsas_refine 2 noplot
```

Performs a least-squares cycle by calling POWPREF, then changing the number of refinement cycles to 2 and calling GENLES. Plots of the fit will not be generated for the overview file.

gsas_vary_DIFC

Changes the refinement flags for the diffractometer constants (DIFCs) for a given histogram. This is needed to e.g. calibrate an instrument or sample environment or to accomodate sample misalignment.

Syntax:

```
gsas_vary_DIFC <histogram> <code>
```

Parameters:

- `histogram`: The number of the histogram for which the new refinement flags are given.
- `code`: The code(s) of the diffractometer constants that should be varied. These depend on the instrument type, e.g. for neutron time-of-flight they are `C` for `DIFC`, `A` for `DIFA`, and `Z` for `ZERO`. The codes for other instrument types can be queried with `expedt`, options `k l o c v`. Combinations are possible, e.g. `CA` would vary `DIFC` and `DIFA` in the above example. To fix diffractometer constants use `" "` (double quotes with space inbetween) as code.

Example:

```
gsas_vary_DIFC 1 C
gsas_vary_DIFC 2 " "
```

Turns on variation of `DIFC` for histogram number 1 and fixes all diffractometer constants for histogram 2.

gsas_vary_histogram_scale

Changes the variation flag for the histogram scale factor. In general, the histogram scale should be varied and is so by default in a new refinement. However, in Le Bail mode all intensity changing parameters should be fixed to avoid zero-matrix elements (message ”... Columns of the N Column matrix are 0.0”) and divergence.

Syntax:

```
gsas_vary_histogram_scale <histogram> <variation flag>
```

Parameters:

- `histogram`: The number of the histogram for which the new refinement flag is given.
- `variation flag`: The flag, either ”y” to vary the scale or ”n” to fix it.

Example:

```
gsas_vary_histogram_scale 1 n
```

Fixes the histogram scale for histogram 1.

gsas_vary_phase

Sets refinement flag for weight fraction of specified phase.

Syntax:

```
gsas_vary_phase <phase#> <flag>
```

Parameters:

- `phase#`: Number of phase for which weight fraction should be varied or fixed.

- `flag`: Variation flag: `<y>` to vary or `<n>` to fix; if no flag then assume `<y>`.

Example:

```
gsas_vary_phase 2 y
```

Sets refinement flag for phase 2 to “y” (vary it).

gsas_vary_profile_parameters

This is the most common command to vary peak profile parameters. For some commonly used peak profiles, special commands are available to vary common parameters such as σ_1 for neutron time-of-flight data. For the general case, i.e. to vary any parameter, this command can be used.

Syntax:

```
gsas_vary_profile_parameters <histogram> <phase> <flags>
```

Parameters:

- `histogram`: The number of the histogram for which the flags should be set.
- `phase`: The number of the phase for which the flags should be set.
- `flags`: The flags, either in sets of flags like ”n y n”, or as a single sequence exactly as queried by EXPEDT, e.g. ”/ / n y n / / /” to not change the first and second set of flags, then change to not varying, varying and not varying the next three parameters, and leaving the rest untouched. The sequence of parameters depends on the profile function used and if in doubt needs to be tried by starting EXPEDT.

Example:

```
gsas_vary_profile_parameters 1 1 "n n" "y y" "y y y" "n n n"
```

Change the variation for peak profile parameters of phase 1 in histogram 1.

gsas_vary_texture

Changes the variation and damping flags for the spherical harmonics description of the ODF for a given phase.

Syntax:

```
gsas_vary_texture <phase> <variation flag> <damping>
```

Parameters:

- `phase`: Number of phase for which the ODF/texture should be varied or fixed.
- `variation flag`: Variation flag: `<y>` to vary or `<n>` to fix.
- `damping`: The damping flag, a number between 0 (no damping) and 9 (maximal damping). This parameter is optional - if not given, the previous damping flag persists (no damping by default).

Example:

```
gsas_vary_texture 1 n
```

Fixes the texture for phase 1.

4.5 Neutron time-of-flight specific

gsas_change_sigmas

Changes the values for the three peak width parameters of neutron time-of-flight profile function number 1. This function can be used to reset the values if needed or initialize to a specific value in case constraints between peak width parameters are introduced that require a certain ratio, e.g. that they are the same, between peak width parameters of different histograms or phases.

Syntax:

```
gsas_change_sigmas <histogram> <phase> <sigma0> <sigma1> <sigma2>
```

Parameters:

- `histogram`: The number of the histogram for which the value of the σ parameters should be changed.
- `phase`: The number of the phase for which the value of the σ parameters should be changed.
- `sigma0,sigma1,sigma2`: The value for each of the three parameters.

Example:

```
gsas_change_sigmas 2 1 0 60 0
```

Changes the peak width parameters σ_0, σ_1, and σ_2 for histogram 2, phase number 1, to 0, 60, and 0, respectively.

gsas_vary_sigma1

Variation of the σ_1 peak-width parameter for neutron time-of-flight data using profile function number 1, the default for many neutron time-of-flight diffractometers.

Syntax:

```
gsas_vary_sigma1 <histogram> <phase> <flag> <damping>
```

Parameters:

- `histogram`: The number of the histogram for which the σ_1 parameter should be varied.
- `phase`: The number of the phase for which the σ_1 parameter should be varied.
- `flag`: The refinement flag, either `y` for varyiable or `n` for fixed.
- `damping`: The damping flag, a number between 0 (no damping) and 9 (maximal damping).

Example:

```
gsas_vary_sigma1 2 1 y 4
```

Varies (`y`) the σ_1 parameter for histogram 2, phase 1 and damps the variation with a damping factor of 4.

4.6 Generating extra infomation

gsas_fourier_maps

Generates all Fourier maps defined in GSAS: Calculated (FCLC), observed (experimental, FOBS) and difference Fourier map (DELF) as well as a 2FDF map (the Fourier map $2F_o - F_c$) . After generation, the Fourier maps are output in `.grd` format, which can be imported for instance into Koichi Momma's software VESTA for viewing Fourier maps in 3D together with the crystal structure for comparison and e.g. identification of potentially missing atoms. You may download VESTA from

`http://www.geocities.jp/kmo_mma/crystal/en/vesta.html`
Open the grd file in VESTA. To modify the contour levels in VESTA, go to Objects – Properties - Isosurfaces, select an isosurface in the list (roughly center of the dialog box, next to new, delete and clear buttons) and modify the level, color etc. above the list. In order to also plot the atoms, go to Edit Data - Structure Parameters and click on import (bottom right) and read the EXP file with the current crystal structure. If you save a VESTA file with a .grd and your current imported EXP file, you may just re-open the VESTA file to obtain updated structure plots after refinement and/or generation of Fourier plots have run again.

Syntax:

```
gsas_fourier_maps <section>
```

Parameters:

- `section`: The type of section for the Fourier map, x, y, or z. The default (if this parameter is not given) is z.

Example:

```
gsas_fourier_maps
```

This will simply generate all Fourier maps in `.grd` format.

gsas_plot_histograms

Plots all histograms of a GSAS refinement. Contrary to the main refinement protocol, no error plots are given. This particularly useful for multi-histogram refinements, e.g. for texture analysis, as it allows to more easily inspect all histograms. The output file is always `all_hist.pdf`, which may be renamed to inspect a longer refinement at various steps after the refinement is done (or crashed...).

Syntax:

```
gsas_plot_histograms
```

Parameters:

- This function has no parameters.

Example:

```
gsas_plot_histograms mv all_hist.pdf After8thOrder.pdf
```

Generates all plots of the current refinement and then renames the resulting PDF file to `After8thOrder.pdf` to indicate at which step the ”snapshot” was taken.

4.7 Plotting of results