GêBR: a free seismic processing interface

Ricardo Biloti, State University of Campinas & INCT-GP, Brazil

This paper was prepared for presentation at the Twelfth International Congress of the Brazilian Geophysical Society, held in Rio de Janeiro, Brazil, August 15-18, 2011.

Abstract

This article presents GêBR, a free graphical interface created for the processing of seismic data, focused on teaching and research in this area. Besides allowing the creation and management of processing streams from predefined modules, GêBR allows adding new modules, directly by the user. This feature turns GêBR an interface to gather algorithms and programs produced by innovative research.

The GêBR Project

In mid-1990, Eduardo Filpo, a geophysicist of the Brazilian oil company, Petrobras, was doctoral student in the graduate program in Geophysics of Federal University of Bahia, Brazil. It was he who first realized the need for a free interface for seismic processing, able to integrate scientific output of that group in the area of Geophysics within a platform in a way that makes it accessible beyond the limits of the research group (Filpo, 2009 and Filpo, 2009b).

Over ten years later, in 2003, a first prototype interface was created by the Eduardo Filpo to assist in training new professionals who would work in seismic data processing. The prototype, named CIGEF, in allusion to the Introductory Course in Geophysics (*Curso Introdutório em Geofíisica*, in Portuguese), taught to new geophysicists hired by Petrobras, was a proof of concept. It demonstrated how beneficial for learning would be to have a graphical interface for processing, so that students could concentrate exclusively on the issue seismic processing itself, and no longer allocate time to the technicalities involved in creating, editing, submission and control of shell scripts, as was usual until then.

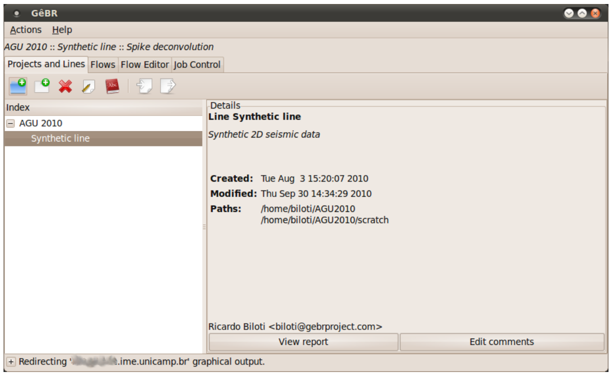
In 2005, another prototype, now already called GêBR, was used in other training for employees of Petrobras, which took place in the Federal University of Parana, Brazil. With that interface was already possible to complete the processing of two lines of real data. In mid 2007, aiming to build a solid foundation for the development of GêBR interface, a new version was written completely from scratch, by Ricardo Biloti. It's over this version that is built the current interface GêBR and other related programs.

October 2007 marked the official launch of Project GêBR[1] to the community with its public presentation at the X International Congress of the Brazilian Geophysical Society, in Rio de Janeiro. Initially created to host the development of GêBR interface, the GêBR Project ended up gaining a larger dimension. Currently, the project aims to encourage the integration of the Brazilian Geophysical community, providing a graphical interface for processing seismic data which, besides being useful for teaching students and professionals, could also be used as an aggregating point and vehicle for the dissemination of research results, accelerating and enhancing their impact within the academic community and the society as a whole.

The GêBR interface

The main and most visible result of the GêBR Project is its graphical seismic processing interface. Unlike other free interfaces for creating seismic processing flows, such as TkSU[2], GêBR is not only designed to the creation and submission of a processing flows, but also to the complete management of the processing of a 2D seismic data.

The GêBR is organized into four tabs (see Figure 1): (1) Projects and Lines, (2) Flows, (3) Flow Editor, and (4) Job Control. The first two tabs are designed to facilitate the organization of the work, allowing the management of several projects, each of which with their own lines and processing flows. The third tab is where the processing flows are assembled from processing modules already available in the interface, the so called *menus*. After this stage of assembly, each execution of a flow, a job is created. The fourth tab is where the monitoring and management of these jobs takes place.



AGU 2010 :: Synthetic line :: Spike deconvolution

Line Synthetic line

Synthetic 2D seismic data

Created: Tue Aug 3 15:20:07 2010
Modified: Thu Sep 30 14:34:29 2010
Paths: /home/biloti/AGU2010
/home/biloti/AGU2010/scratch

Ricardo Biloti <biloti@gebrproject.com>

Redirecting '[redacted].ime.unicamp.br' graphical output.

Figure 1: GêBR interface. At left, it is shown the project *AGU 2010* with one line, *Synthetic line*. At right, it is show some detailed information about that line, which is selected.

[1] http://www.gebrproject.com

[2] http://www.henrythorson.com/interface.htm

Simplicity

The simplicity of the GêBR GUI is not a negative point, indicating lack of resources, but it is its greatest advantage, making it feasible for new users to take advantage of the interface almost immediately. In each tab there is always a toolbar with buttons for implementing key actions associated with the context. Moreover, to avoid overloading the interface and also to make access to functionalities faster, context menus (accessible through the right mouse button) are available, presenting shortcuts for actions relating only the element (or group of elements) selected.

Extensibility

Another strength of the GêBR interface is that it is not static. New processing modules can be incorporated into the interface. This feature allows that GêBR acts as a dissemination vehicle for the new research products, in the sense that new programs can be immediately incorporated into the interface, being in this way readily available to use.

In principle, any program that can be executed through the command line of a shell can be integrated to GêBR. The more the program is controlled by options provided in the command line, the better the integration of it with GêBR. Currently, approximately 400 programs are already integrated into GêBR, mainly from Seismic Unix processing package (Stockwell, 1999).

The integration of a new program to GêBR is straightforward. It is done through an auxiliary interface, also developed under the GêBR Project, known as DéBR. To complete this integration it is not necessary to have access to the source code of the program, but only have access to its usage documentation.

Organizational resources

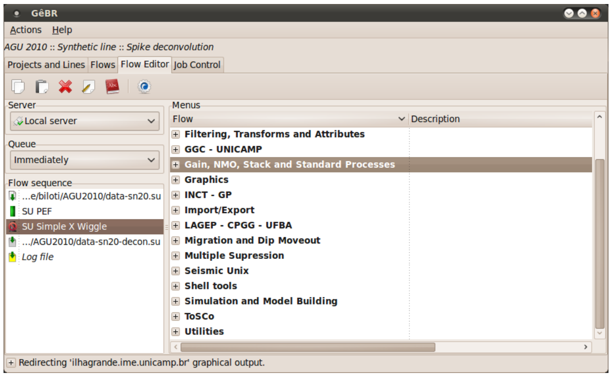
To help users organize their work within the GêBR interface there is a hierarchy of information to be respected and explored. The first level of this hierarchy is inhabited by *projects*. Projects are responsible for hosting groups of *lines*, an entity that occupies the second level of the hierarchy. Both for projects as well as for lines, it is possible to define basic properties, like the owner or responsible for the project, a title and a short description. For lines is still possible to define preferred paths. These paths are used during the construction of processing flows, to facilitate the organization of the input and output files. The third level of the hierarchy is populated by *processing flows*. Just like for projects and lines, it is also possible to set basic properties for each flow.

Beside the lists of projects and lines or flows, there is always a summary table displaying the main details about the selected element. Besides the properties of the element, for flows, for example, it is also shown when the flow was last ran, in which server, and which files are configured as input or output. GêBR also yields detailed reports for lines and flows containing the list of all the flows of the line, tables summarizing the setup of each program of each flow, which are the input and output files for the flow, among other information. In addition, there is also placeholder for users to add their own comments on the element.

Each execution of a processing flow, a job is created and displayed on the last tab GêBR interface. When selecting a specific job in that tab, the user has access to all information generated by the processing flow, like the current state of flow (submitted, waiting in a queue for execution, running, completed, or stopped), the command line that was actually used to run the job, error messages, run time, etc. Furthermore, users can stop a running flow or cancel the execution of a flow which is waiting in a queue to start its execution.

Resources to assembly and configuration of flows

To facilitate the construction of processing flows, the menus are sorted by categories (see Figure 2). The same menu can belong to more than one category, making it easy to find. If the user still can not find the menu he or she wants, the user can do incremental search by the name of the menu. After finding the desired menu, by requesting the inclusion of it in the processing flow, all programs that make up the menu will be included in the flow, independently, allowing the user to disable or even remove some of them.



AGU 2010 :: Synthetic line :: Spike deconvolution

Server: Local server

Queue: Immediately

Flow sequence
...e/biloti/AGU2010/data-sn20.su
SU PEF
SU Simple X Wiggle
.../AGU2010/data-sn20-decon.su
Log file

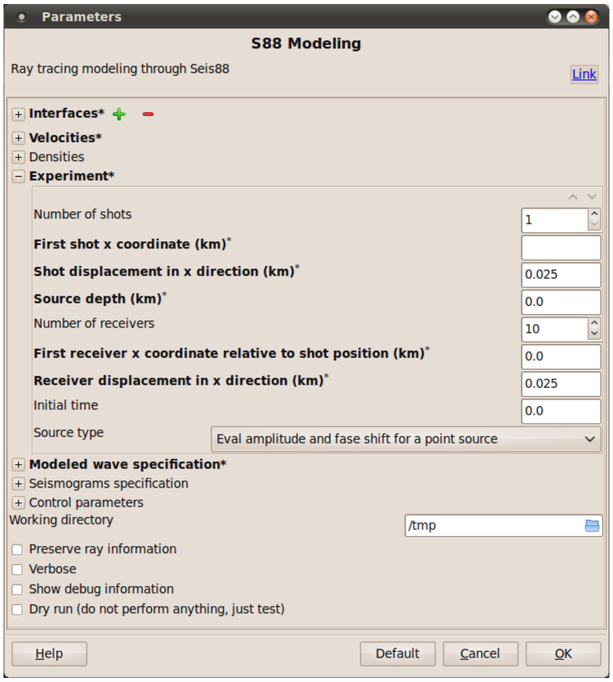
Redirecting 'ilhagrande.ime.unicamp.br' graphical output.

Figure 2: Flow Editor tab. At left in the frame labeled *Flow sequence* it is shown the programs which compound the flow. In this case the second program (*SU Simple X Wiggle*) is disabled. At right, it is shown the list of menu's categories, from which menus can be grabbed to assemble the flow.

The configuration of each program of a processing flow is made through a parameters' dialog specific to the program. Figure 3 shows the parameters' dialog for program *S88 Modeling*. GêBR supports various types of parameters: numeric (integer, real, with or without limitation of range), text, multiple choice within a predefined list of options, Boolean values and files. This specificity allows that the interface correctly validates the values filled out for each parameter, turning the task of configuring programs less error prone.

Through this dialog of parameters, the user yet has access to the documentation of the program in question (see Figure 4), as well as to an external link that refers to a page with more details about the program. By means of a button at the bottom of the dialog, the setting of all parameters can be returned to default values.

Besides being able to fill in directly the value of a parameter, it is possible to employ the *dictionary of parameters*, where values can be labeled to be shared among programs of the same flow, or even between



S88 Modeling

Ray tracing modeling through Seis88

- **Interfaces***
- **Velocities***
- Densities
- **Experiment***

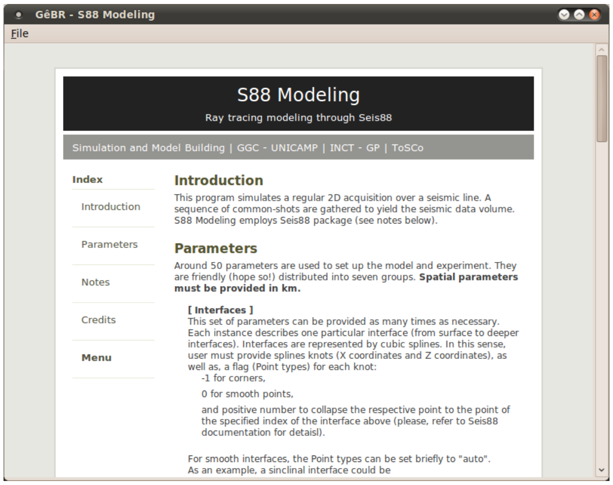
Number of shots	1
First shot x coordinate (km)*	
Shot displacement in x direction (km)*	0.025
Source depth (km)*	0.0
Number of receivers	10
First receiver x coordinate relative to shot position (km)*	0.0
Receiver displacement in x direction (km)*	0.025
Initial time	0.0
Source type	Eval amplitude and fase shift for a point source

- **Modeled wave specification***
- Seismograms specification
- Control parameters

Working directory: /tmp

□ Preserve ray information
□ Verbose
□ Show debug information
□ Dry run (do not perform anything, just test)

Figure 3: Parameters' dialog for the program *S88 Modeling*. Note that groups of parameters are defined to encompass parameters which are related to each other. These groups can be collapsed or expanded. Required parameters are displayed in boldface.



S88 Modeling

Ray tracing modeling through Seis88

Simulation and Model Building | GGC - UNICAMP | INCT - GP | ToSCo

Introduction

This program simulates a regular 2D acquisition over a seismic line. A sequence of common-shots are gathered to yield the seismic data volume. S88 Modeling employs Seis88 package (see notes below).

Parameters

Around 50 parameters are used to set up the model and experiment. They are friendly (hope so!) distributed into seven groups. **Spatial parameters must be provided in km.**

[Interfaces]
This set of parameters can be provided as many times as necessary. Each instance describes one particular interface (from surface to deeper interfaces). Interfaces are represented by cubic splines. In this sense, user must provide splines knots (X coordinates and Z coordinates), as well as, a flag (Point types) for each knot:

-1 for corners,

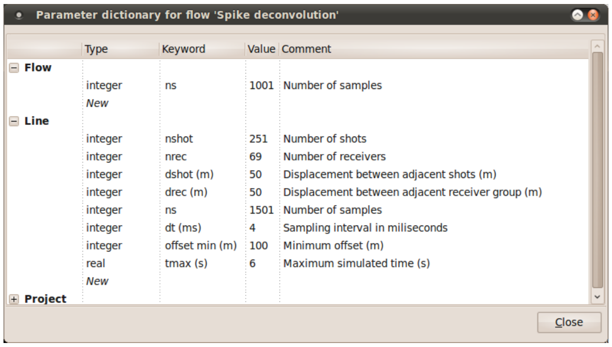
0 for smooth points,

and positive number to collapse the respective point to the point of the specified index of the interface above (please, refer to Seis88 documentation for detaisl).

For smooth interfaces, the Point types can be set briefly to "auto".
As an example, a sinclinal interface could be

Figure 4: Detailed help for the *S88 Modeling* program.

programs of different flows (see Figure 5). For example, if two or more programs need to know a reference velocity, it would be better to fill this value in the dictionary and then set the appropriate parameters of the those programs through the dictionary. Thus, if necessary, change the value of this property in the dictionary will have the effect of this change automatically propagate to all programs. The definition of the parameters in the dictionary is done by scope. Parameters defined for the flow have priority over parameters set for the line, overlapping them if there are collision of labels. Parameters can also be set through arithmetic expressions, employing values defined within the dictionary of parameters.

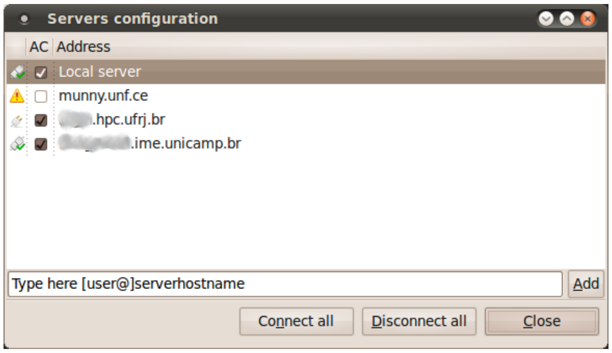


	Type	Keyword	Value	Comment
Flow				
	integer	ns	1001	Number of samples
	New			
Line				
	integer	nshot	251	Number of shots
	integer	nrec	69	Number of receivers
	integer	dshot (m)	50	Displacement between adjacent shots (m)
	integer	drec (m)	50	Displacement between adjacent receiver group (m)
	integer	ns	1501	Number of samples
	integer	dt (ms)	4	Sampling interval in miliseconds
	integer	offset min (m)	100	Minimum offset (m)
	real	tmax (s)	6	Maximum simulated time (s)
	New			
Project				

Figure 5: Dictionary of parameters.

Resources for flow execution

Through GêBR, it is possible to connect to multiple processing servers, allowing the user to submit flows for execution on different machines and to control the progress of each job. Figure 6 shows the dialog where servers are registered. A flow may be subjected to immediate execution or queued, being blocked from running until a specific job has been completed. This feature, common in large machines, like clusters, is natively available on any machine through GêBR.



AC	Address
☑	Local server
□	munny.unf.ce
☑	[redacted].hpc.ufrj.br
☑	[redacted].ime.unicamp.br

Type here [user@]serverhostname

Figure 6: Server's setup dialog. This dialog shows the list of registered servers. It is possible to see which server is actually connect, ask for connection or even stop a server.

Upon a job submission, GêBR itself makes several validations in the generated job to minimize potential problems in the flow setup, alerting the user about them and blocking the submission until the problems have being solved.

Technical aspects

The GêBR interface, as well as other softwares developed by the GêBR Project, is developed mainly in C, employing the graphical toolkit GTK+, among many other common libraries, all of them being highly portable. In fact, to be able to provide an interface to a wide range of systems, it is always a concern of the project choose portable libraries. Yet in this regard, all information filled in the interface is stored as XML, which is a standard format whenever portability is an issue to be considered.

All code produced by the GêBR Project is released as free software, under the terms of the GNU General Public License. This guarantees unrestricted access to the code to everyone, who might be interested.

References

Filpo, E., 2009, GêBR – The geophysical interface of Brazil (*GêBR – A interface geofísical do Brasil*, in Portuguese). Boletim SBGf, No. 1., pp. 10.

Filpo, E., 2009b, A rescue of the history of GêBR (*Um resgate da história da GêBR*, in Portuguese). URL: `http://www.gebrproject.com/documentation/historia-da-gebr`.

Stockwell, J.W., 1999, The CWP/SU: Seismic Unix package, Computers & Geosciences, Vol. 25, No. 4., pp. 415–419.

Acknowledgments

I would like to thank PETROBRAS for financially support this work, through its Research Net in Applied Geophysics and the sponsors of the WIT Consortium.