Using Uni3D version v0.23: a manual

Ch. Jäger and Th. Hertweck

email: *Christoph.Jaeger@gpi-uka.de*

keywords: *Kirchhoff migration, true-amplitude*

ABSTRACT

Kirchhoff migration is a widely investigated and often utilized imaging tool. By applying suitable weight functions in the migration process it is possible to compensate the spherical divergency effect contained in the seismograms and, thus, to end up with so-called true-amplitude migrated images. If a prestack migration has been performed, such images are suitable to retrieve AVO (amplitude versus offset) or AVA (amplitude versus angle) information which can be helpful for a further characterization of reflectivity contrasts in the subsurface. `Uni3D` *is a program by which such imaging problems can be addressed.*

INTRODUCTION

In the past decades, 2D and 3D reflection imaging has become a familiar topic in the world of seismic exploration. One of the oldest migration tools is Kirchhoff migration which has its roots in the graphical migration scheme of Hagedoorn (1954) that was later related to the wave equation by Schneider (1978). Kirchhoff migration can compensate for the geometrical spreading effect of propagating waves and is thus capable to produce so-called true-amplitude migrated images. Although newer migration techniques exist that can under certain circumstances provide better images, the Kirchhoff method is still applied very often. Reasons for this are, e.g., its flexibility which allows target-oriented processing and the handling of irregular acquisition geometries and topographic variations and its relatively low computational costs (see also the article by Hertweck et al. in this issue). In the Unified Approach Theory by Hubral et al. (1996), Tygel et al. (1996), and Jaramillo et al. (1998) it is shown that by combining Kirchhoff true-amplitude migration and its asymptotic inverse process, Kirchhoff true-amplitude demigration, it is possible to solve various imaging problems.

To study possible applications of this theory, the imaging tool `Uni3D` consisting of Kirchhoff true-amplitude migration and demigration algorithms was developed at the Geophysical Institute at Karlsruhe University. In former versions of `Uni3D`, SEPlib was used as input and output data format. Unfortunately, SEPlib cannot handle irregular trace increments or topographic variations in a proper way. The reason for this is that SEPlib has only one global header, traces are addressed making use of constant increments. For this we now make use of Seismic Un*x (SU) data format. Since every trace in an SU data set has an individual header containing information such as shot- and receiver-position and elevation, there is no restriction to a regular acquisition geometry. Traces recorded on a non-flat surfaces can easily be handled.

The change from SEPlib to SU input data entailed the necessity to rewrite major parts of the source code of `Uni3D`. This work is still in progress. In the current version v0.23 of the program the complete demigration part is missing and migration is limited to the 2D case. But now we are able to migrate directly from topography and are no longer restricted to handle data recorded on regular grids. `Uni3D` v0.23 is able to perform 2D prestack and poststack true-amplitude migration for constant velocity models as well as for laterally and vertically inhomogeneous media as is illustrated in Figure 1. We now also make use of input/output routines that are very similar to the ones utilized by the CRS stack. This compatibility of parts of the software is facilitated by C++ and its ability to substitute object oriented programming.

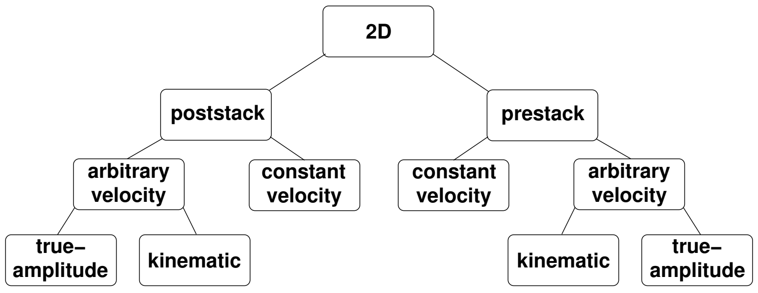

Figure 1: Migration tasks that can be addressed by `Uni3D` v0.23. Constant velocity migration is performed true-amplitude by default. Algorithms for migration of 3D data sets as well as for 2D demigration exist, but are not yet adapted to handle SU data.

New releases of the code will be made available to our sponsors on the password protected area of the WIT homepage `www.wit-consortium.de`. Some migration results that were obtained using `Uni3D` can be found in the article by Hertweck et al. in this issue.

In the following sections we will concentrate on the technical aspects of the developed program such as its installation and compilation and give a detailed description of all parameters that can (or must) be specified by the user.

REQUIREMENTS

The program `Uni3D` is written in C++ and should comply with the current C++ language standard. We tried to avoid platform specific features, but nevertheless, it can be necessary to make some minor changes in the source code since some environments expect, e.g., `.h` suffices for specific C++ header files. The source code was up to now successfully compiled on the machine architectures and operating systems shown in Table 1.

architecture	system	OS	compiler
i686	PC, AMD Athlon	SuSE Linux 8.1, 8.0	GNU gcc/g++ 3.2.1, 2.95.3
MIPS	SGI O2	SGI IRIX 6.5	MIPSpro CC 7.3.1.1m
MIPS	SGI Origin 3200	SGI IRIX 6.5	MIPSpro CC 7.3.1.1m

Table 1: System configurations with successful installations.

The input and output files are in the Seismic Un*x (SU) format, traveltime tables are stored in the SEPlib data format. Both, SU and SEPlib, are complete and freely distributed seismic data processing software packages. SU was developed by the Center for Wave Phenomena at the Colorado School of Mines. It is freely available at `http://www.cwp.mines.edu/cwpcodes/`. SEPlib was developed by the Stanford Exploration Project (SEP) at Stanford University and can be downloaded from `http://sepwww.stanford.edu/software/seplib/`. Note, however, that the program `Uni3D` does not explicitly require these packages—all relevant I/O routines are included in the source code. Nevertheless, we recommend to install them for visualization, pre-processing, etc.

INSTALLATION, COMPILATION

The program `Uni3D` is distributed with a well-documented `Makefile` in which you might have to adjust the compiler and the compiler and linker flags according to your needs. Be sure to have the subdirectories `opt` and `dbg` in the directory in which the source code and the `Makefile` is located. If necessary, all source dependencies are updated by typing `make dep`. Then, the program can be compiled via `make all` or `make debug`. The first yields the optimized executable `Uni3D` while the latter produces the debugging version `Uni3Ddbg`. For further applications of the `Makefile`, simply type `make`.

PARAMETERS

Parameters common for all configurations are listed together with their default values and a short description in Table 2. If no default value is given, the corresponding parameter is mandatory unless it is marked as optional. Additional parameters may have to be specified depending on the chosen configuration. They are listed in Table 3. Again, if no default value is provided, they must be specified. In the following, all parameters are explained in more detail, grouped thematically into subsections. The parameter names with their expected values in brackets are marked as **boldface**.

Parameter=`Type`	Default	Description
Configuration		
dim=`int`	none	input is 2.5D (0) or 3D (1)
task=`int`	none	perform migration (0) or demigration (1)
vel=`int`	none	assume const. velocity (0), use a dynamic GFT (1), or use a kinematic GFT (2)
stack=`int`	none	input is prestack (0) or poststack (1)
Input/Output		
in=`string`	none	name of input file
out=`string`	none	name of output file
Target zone		
xmin=`int`	none	minimum x [unit] in target zone
xmax=`int`	none	maximum x [unit] in target zone
dx=`int`	none	x increment [unit] in target zone
zmin=`int`	none	minimum z [unit] in target zone
zmax=`int`	none	maximum z [unit] in target zone
dz=`int`	none	z increment [unit] in target zone
unit=`int`	1	unit for above values: m (1) or 0.1m (0)
Migration aperture, taper		
aprad=`int`	FZ based	aperture radius [m]
aptap=`int`	FZ based	aprad + taper length [m]
intap=`int`	FZ based	input taper region [m]
tau=`float`	0.09	estimation of wavelet length [s]
theta=`int`	45	maximum dip to be imaged [deg]
Miscellaneous		
verbose=`int`	0	show only warnings and errors (0) or show additional information (1)
par=`string`	optional	name of parameter file
datapath=`string`	optional	common input/output datapath
logfile=`string`	optional	name for logfile

Table 2: Parameters which are common for all configurations

Input/Output

As was mentioned before, the input data is expected to be in SU format. It is read from the file **in (string)**, which can contain a global path or a path relative to the directory in which the executable `Uni3D` is located. SU trace headers that are utilized by `Uni3D` are the shot and receiver x-coordinates (SX, GX), the coordinate scaling factor SCALCO, shot and receiver elevation (SELEV, GELEV), and the time sampling and time offset (DT, DELRT). The shot/receiver elevation is positive when the shot/receiver is located above the plane z=0, negative below. Please note that in the current version of `Uni3D` the input is assumed to be recorded on a straight line (namely the x-axis) and y-coordinates are ignored.

The output filename is specified by the parameter **out (string)**, which can again contain a path. In the output, the trace headers F1, D1, F2, and D2 are set to the minimum values and increments in [m] in the z-

Parameter=Type	Default	Description
If vel=0		
vconst=`int`	none	constant velocity [m/s]
If vel=1 or vel=2		
v0=`int`	none	estimate of minimum velocity [m/s]
gft=`string`	none	name of greens function table
If stack=0		
minoffset=`int`	none	minimum input offset to be considered
maxoffset=`int`	none	maximum input offset to be considered
outoff=`char`	n	keep offsets in output (y) or stack output over offsets (n)
offbinsize=`int`	from data	size of output offset bin [m]

Table 3: Parameters which depend on the chosen configuration

and x-direction, respectively. Aditionally, the x-position [m] is stored also in SX. If a prestack migration (**stack ([0,1]) = 0**) has been performed and offset stacking is omitted during migration (**outoff ([y,n]) = y**) OFFSET is computed.

Target zone

The migration target zone has to be specified by the minimum and maximum values in x- and z-direction together with the desired sampling of these axes. The relevant parameters are **xmin (int)**, **xmax (int)**, **dx (int)**, **zmin (int)**, **zmax (int)**, and **dz (int)**. We assume the z-axis pointing downwards, z-values are thus positive. The unit of the above values is meter unless the parameter **unit ([0,1])** is set to '1'—then, the unit for the above parameters is set to 0.1m.

Constant velocity or arbitrary velocity field?

If the parameter **vel ([0,1,2])** is set to '0', migration is performed assuming a constant wave-propagation velocity. This velocity must be assigned in m/s to the parameter **vconst (int)**.

True-amplitude migration for arbitrary velocity models requires not only traveltime tables but tables containing certain additional wavefield attributes as, e.g., the geometrical spreading or the KMAH index. Theses quantities are normally retrieved by dynamic ray tracing. A different approach to obtain weights needed in true-amplitude imaging, namely to express them purely in terms of traveltime, is investigated by the WIT group in Hamburg (see, e.g., Vanelle and Gajewski (2002)). For situations in which a compensation for spherical divergence is not needed, we have implemented purely kinematic migration schemes that require only kinematic traveltime tables. If the velocity model is simple so that the consideration of only first-arrivals is sufficient, these tables can be computed very fast using, e.g., eikonal solvers. Migration for constant velocity models is by default implemented in a true-amplitude way because the necessary weight functions can be computed analytically without much additional computational effort.

For migration with arbitrary velocity models a traveltime table for kinematic migration (**vel ([0,1,2]) = 2**) or, if true-amplitude processing is desired (**vel ([0,1,2]) = 1**), a table containing further wavefield attributes (see below) must exist. The respective filename has to be assigned to the parameter **gft (string)**. These traveltime tables, which we also call Green's function tables (GFTs), have to be in the SEPlib data format and must have the following "shape": axis 1 = para, axis 2 = z, axis 3 = x and axis 4 = sx, where x and z define the subsurface target zone and sx contains the shot positions at the surface. The values z, x, and sx must be given in km. For a kinematic GFT, axis 1 contains simply the one-way traveltime from the source at sx to a depth point defined by (x,z). In case of a dynamic GFT, additional wavefield attributes are stored on axis 1 besides f1 = 0, which is again traveltime. They are: f1 = 1: ray take-off angle at the source, f1 = 2: geometrical spreading, f1 = 3: KMAH index and f1 = 4: angle to vertical at depth point. That is, for every shot position a three-dimensional cube has to be stored as is depicted in Figure 3. If such a dynamic traveltime table is given, spherical divergency effects of the recorded input data are compensated by a

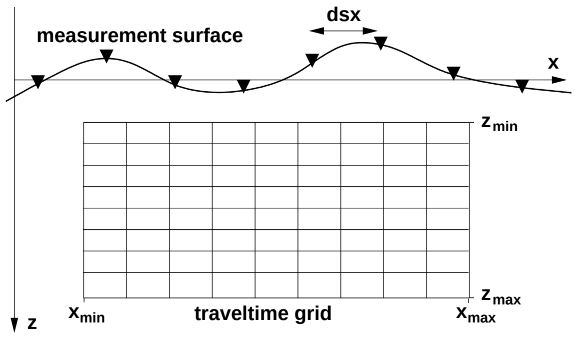

Figure 2: In a kinematic GFT, the traveltime from each shot to each point of the fixed x-z grid is stored. The kinematic GFT is thus 3-dimensional.

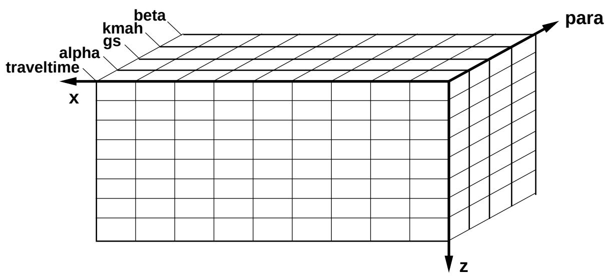

Figure 3: Sketch of the 3-dimensional subset for one single shot position of the 4-dimensional dynamic GFT.

weight function in the stacking process, that is, a true-amplitude migration is performed. Unfortunately, the ray-tracer which we utilize for the construction of the dynamic GFT is not able to provide all quantities needed for the computation of the 2.5D true-amplitude weight function as, e.g., the in-plane and out-of-plane geometrical spreading. Therefore, the 3D weight is implemented even for migration of 2D data, the correct weight is commented out. If a ray-tracer is available which can determine the required parameters, the source code can easily be adapted.

In all tables, the shots must have a constant horizontal spacing dsx but may be located on a surface with varying topography (see Figure 2). The GFT target zone remains fixed for every shot position. The shot increment as well as the x- and z-spacing of the GFT can be chosen larger than the shot increment of the input seismogram and the desired x- and z-spacing of the migration target zone. All parameters are interpolated to the dense output grid during runtime. Please note that multi-pathing is currently not considered in the program.

Prestack migration

If the input seismogram is prestack (**stack ([0,1]) = 0**), the migrated output section is by default stacked over all offsets. If offsets are to be kept in the migrated image, e.g., in order to analyze image gathers, the parameter **outoff ([y,n])** has to be set to 'y'. The output offset bin size is then set by default to the average offset increment found in the input data. If this is not desired, the bin size can be set to the user-defined value **offbinsize (int)** [m].

Migration aperture, tapering

The size of the migration aperture is always limited by the region over which data have been acquired. A further restriction of the migration aperture might be advantageous because (1) it leads to a speedup of the migration, (2) a smaller operator excludes steeper dips, which helps to avoid operator aliasing (see, e.g., Abma et al., 1999), and (3) less summation of data away from the signal reduces the stacking of unwanted noise. For the best possible reduction of aliasing and noise as well as the best computational efficiency, one would like to use a model-based aperture restriction, i.e., one would like to make use of the (projected) Fresnel zone (see, e.g., Schleicher et al., 1997; Sun, 1998; Sun and Bancroft, 2001). Sun suggests that the stacking region should cover the first projected Fresnel zone and the taper region should extend over the second projected Fresnel zone around the stationary point. (The stationary point is the point, where the stacking operator has the same time dip as the reflection event.)

Because it is difficult to determine the center and size of the Fresnel zone for each depth point prior to or during migration, we restrict the aperture to a constant aperture radius which can be defined by setting the parameter **aprad (int)** to the desired value [m]. Note that such a fixed aperture images smaller maximum dips with increasing depth. The simple truncation of the migration operator would lead to artifacts. They are avoided by tapering the operator smoothly to zero in the range between **aprad (int)** and the value defined by **aptap (int)** [m]. To suppress migration artifacts which stem from the boundaries of the input, the input is tapered at its margins over a region defined by **intap (int)** [m].

By default, these parameters are determined from the data: For the zero-offset configuration and a constant velocity medium, the center and the radius of the projected Fresnel zones can be computed depending on a measure of the wavelet length (parameter **tau (int)** [s]) and the maximum dip to be imaged (parameter **theta (int)** [deg]) (Hertweck et al., 2001). The utilized formula depends also on the depth of the considered diffraction point. For the calculation of the default values, this z is set to the maximum depth in the target zone **zmax (int)**.

For prestack data and for arbitrary velocity fields the default values for the parameters **aprad (int)**, **aptap (int)**, and **intap (int)** are computed by means of the same equation (making use of the minimum velocity **v0 (int)** instead of **vconst (int)**). In that cases, the obtained default values are of course only a rule of thumb and might have to be adjusted by the user.

Miscellaneous parameters

By default, the running program writes only warning and error messages to standard error . To receive further information during execution, set the parameter **verbose ([0,1])** to ‘1’. If desired, a log file (**logfile (string)**) in which all these messages are stored can be specified.

Instead of assigning all input and output files with absolute or relative path to the respective parameters, it is possible to specify the parameter **datapath (string)**. This datapath is then applied to all files not beginning with ‘/’ or ‘.’.

To reduce the amount of needed typing and to facilitate reproducible results, all parameter-value pairs can be read from a text file which is then handed over to the program using the parameter **par (string)**.

FUTURE WORK

Some of our developed algorithms as, e.g., the migration of 3D data sets or demigration, have not yet been adapted to handle (the possibly irregular) SU input data and are, thus, not included in the current version of `Uni3D`. Besides of that, the implemented algorithms can clearly still be improved. Our future plans include the avoiding of operator aliasing, the considering of multi-arrivals, the handling of crooked line geometry, and the testing of more sophisticated methods to interpolate the Green’s function tables.

ACKNOWLEDGMENTS

This work was kindly supported by the sponsors of the *Wave Inversion Technology (WIT) Consortium*, Karlsruhe, Germany.

REFERENCES

Abma, R., Sun, J., and Bernitsas (1999). Antialiasing methods in Kirchhoff migration. *Geophysics*, 64(6):1783–1792.

Hagedoorn, J. (1954). A process of seismic reflection interpretation. *Geophys. Prosp.*, 2(2):85–127.

Hertweck, T., Jäger, C., Goertz, A., and Schleicher, J. (2001). Aperture effects in 2.5-D Kirchhoff migration. Submitted to Geophysics; partly published in Annual WIT Report 2001.

Hubral, P., Schleicher, J., and Tygel, M. (1996). A unified approach to 3-D seismic reflection imaging, Part I: Basic concepts. *Geophysics*, 61(3):742–758.

Jaramillo, H., Schleicher, J., and Tygel, M. (1998). Discussion and Errata to: A unified approach to 3-D seismic reflection imaging, Part II: Theory. *Geophysics*, 63(2):670–673.

Schleicher, J., Hubral, P., Tygel, M., and Jaya, M. (1997). Minimum apertures and Fresnel zones in migration and demigration. *Geophysics*, 62(1):183–194.

Schneider, W. (1978). Integral formulation for migration in two and three dimensions. *Geophysics*, 43(1):49–76.

Sun, J. (1998). On the limited aperture migration in two dimensions. *Geophysics*, 63(3):984–994.

Sun, S. and Bancroft, J. (2001). How much does the migration aperture actually contribute to the migration result? In *Expanded Abstracts*.

Tygel, M., Schleicher, J., and Hubral, P. (1996). A unified approach to 3-D seismic reflection imaging, Part II: Theory. *Geophysics*, 61(3):759–775.

Vanelle, C. and Gajewski, D. (2002). True-amplitude migration weights from traveltimes. *Pure and Applied Geophysics*, 159:1583–1599.