H3DTD – MUMPS

A Program Library for Forward Modelling of Multi-Transmitter, Time-Domain Electromagnetic Data over 3D structures.

Version 1.6

Developed under the MITEM consortium Research Project
Multi-Source Inversion of Time Domain Electromagnetic Data

UBC Geophysical Inversion Facility
Department of Earth and Ocean Sciences
University of British Columbia
Vancouver, British Columbia

`http://www.eos.ubc.ca/research/ubcgif/`

Introduction

H3DTD is a software package for solving Maxwell's equations in the time domain. The equations are discretized in time using backward Euler method and discretized in space by using a finite volume technique on a staggered grid. The sources can be grounded dipoles or loop currents that reside in the air, on the surface, or inside the earth. The responses can be any combination of components of *E*, *H*, or d*B*/d*t.* The transmitter waveform is user-defined and there are no restrictions on the length or shape of the waveform. Data can be simulated in the "on-time" or "off-time". The earth model is an arbitrary 3D conductivity defined on a structured rectangular mesh. The earth can also have an arbitrary 3D magnetic susceptibility.

The solutions are achieved by factorizing the forward modelling matrix and hence it is possible to simulate data from many sources. This is one of the major benefits of this approach. H3DTD, although it requires significant computing resources, can be run on single modern laptop or desktop computer. Factorization of the forward modelling matrix is facilitated via the MUMPS software for which documentation and downloads can be found at the following website:

`http://graal.ens-lyon.fr/MUMPS/`

The MUMPS routines are built-in to H3DTD and do not need to be installed separately. However, since the forward modelling matrix is large and its decomposition is computationally intensive, H3DTD is most efficiently run on an array of computers, or on a single multi-core computer with lots of RAM (~ 12GB). The parallel implementation is carried out using the Message Passing Interface standard (MPI). MPI installation and usage will be discussed further in this document.

To obtain quality numerical results from the algorithm, the problem must be discretized properly in both space and time. An important step in determining whether discretization is sufficiently accurate is to compare the fields from H3DTD with those from another algorithm. For this purpose we use a 1D code, developed at UBC and "field-tested" for almost a decade now. To assist the user in designing an appropriate mesh and time stepping, and to help validate the mesh, we have generated GUI utilities that are supplied with this package. A list of the major components of this package is provided below.

List of the input files and Graphical User Interface utilities (GUI's)

Programs and GUI's	Input files	File names
H3DTD.exe	Input control file	h3dtd.inp (fixed name)
	Model files (conductivity, susceptibility)	model.con (user specified) model.sus (user specified)
	Tx-Rx location file	trx_loc.txt (user specified) recv_loc.txt (user specified)
Mesh_builder.exe GUI	Mesh file	mesh.txt (user specified)
Wave_builder.exe GUI	Wave file	wave.txt (user specified)
	Time gates file	gates.txt (user specified)

Table 1. List of H3DTD components and control files (green means these files are not mandatory to run the code.

Setting up the control files

"H3DTD.exe" is the executable for the 3D forward simulation. It has to be run in a separate folder that also contains the input file "h3dtd.inp". The remainder of the input files do not have any strict naming convention and can be located in any hard drive or network location under any name, as long as they are properly entered in the "h3dtd.inp" file.

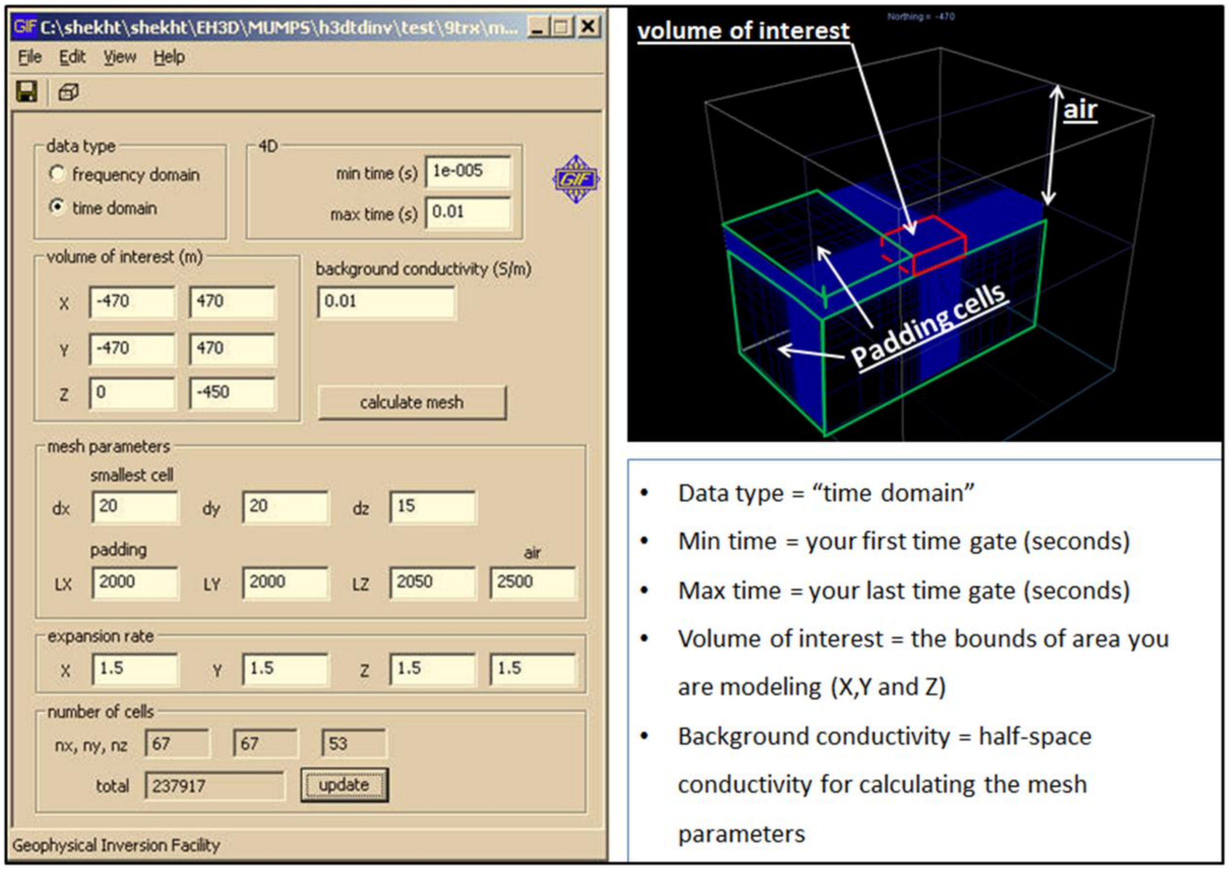
The input control file has the following 6 line format, as follows:

```
../../mesh.txt  ! mesh file
FILE ../../background.con  ! conductivity file
0  ! susceptibility file
trx_loc.txt  ! transmitter location file
wave.txt  ! wave file
gates.txt   ! output time values
```

In this file there is several supplementary control files listed with full paths to their locations.

The ***mesh file*** can be located in any folder and can have arbitrary name, as long as it is in ASCII format. The design of the mesh can be handled by the GUI utility *"Meshbuilder.exe".* Selection of mesh parameters is very important and it will be discussed further in this document. In figure 1 there is a screenshot of Meshbuilder GUI with explanations of described parameters"

In setting the mesh parameters, the cell size of "volume of interest" ("smallest cell" in the menu) should depend on the actual geometry of the survey (primarily on density of stations defined by line spacing and sampling rate). The user has to maintain a balance between saving computing time (by coarsening the mesh) and getting more accurate solution of the forward simulation by making the mesh finer. The padding distance, depends on the latest data acquisition time gate and is calculated automatically equal to diffusion distance defined by equation (1):



data type: ( ) frequency domain [✓ time domain]

4D: min time (s) 1e-005; max time (s) 0.01

volume of interest (m): X -470 470; Y -470 470; Z 0 -450

background conductivity (S/m): 0.01

mesh parameters: smallest cell dx 20, dy 20, dz 15; padding LX 2000, LY 2000, LZ 2050, air 2500

expansion rate: X 1.5, Y 1.5, Z 1.5, 1.5

number of cells: nx, ny, nz 67 67 53; total 237917

- Data type = "time domain"
- Min time = your first time gate (seconds)
- Max time = your last time gate (seconds)
- Volume of interest = the bounds of area you are modeling (X,Y and Z)
- Background conductivity = half-space conductivity for calculating the mesh parameters

Figure 1. Mesh file builder GUI parameters setup and graphical representation.

(1) $$\mathbf{D(t) = 1250^*}\sqrt{t/\sigma} \quad \text{(meters)}$$

In this equation t is the latest time gate and σ – is the conductivity of the background half-space in Siemens per meter (S/m).

The background conductivity in meshbuilder is only used to calculate the diffusion radius and is not being further assigned for the forward starting model. The expansion rate is the geometric progression coefficient used for building the padding cells. Generally values between 1.3 and 1.6 are reasonable choices for the expansion rate.

The format of the mesh file is:

```
nx ny nz                      (number of cells in the X, Y, and Z directions)
x0 y0 z0                      (coordinates of the top south west corner of the mesh)
dx_1 dx_2 ... dx_nx           (cell widths in X)
dy_1 dy_2 ... dy_ny           (cell widths in Y)
dz_1 dz_2 ... dz_nz           (cell widths in Z)
```

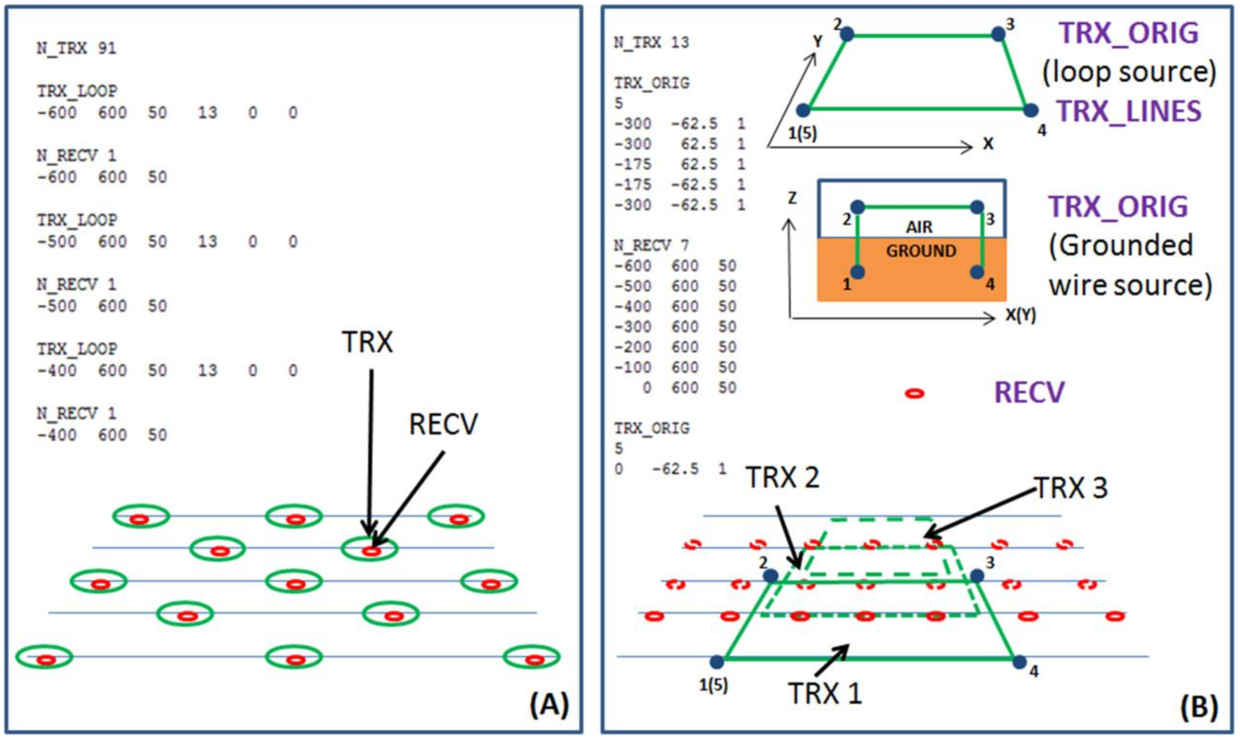
The conductivity and the susceptibility ***model files*** together compose the starting model for the forward simulation. They are referenced in the second and third lines of the input control file, and each of them is defined in the usual GIF format; that is, a single column of values with one value for each cell. The first value corresponds to the top south-western cell. The values are ordered such that Z (depth) changes the fastest, followed by X (easting), followed by Y (northing). For constant half-space model parameters, a single value entered in the input control file can be used as a substitute to the model files. When a file is entered, the line in the input file should start with "FILE", and when a constant value is entered, the line should start with VALUE. For example, in our "h3dtd.inp" file example above, electrical conductivity model is set to be read from a file "*background.con*", located two directories up from the working directory, while the magnetic susceptibility is set to zero (SI units). Instead of "0", "NULL" can be used. Therefore all cells below Z=0 will then be set to these values. The cells above Z=0 will be set to 10^{-8} S/m and zero susceptibility (air).

One of the most important features of the H3DTD is the ability to simulate data from multiple transmitters. This feature facilitates simulation of airborne time domain surveys. A ***location file*** is used to define the transmitter (TRX) and receiver (RECV) locations and fully defines the geometry of the survey.

```
N_TRX     1                                (Number of transmitters is 1)

TRX_ORIG                                   (Transmitter type set to "Original")
5                                          (5 nodes describing TRX geometry)
 -2.00000E+01 -4.20000E+02   7.5           (Coordinates of the nodes 1 to 5)
 -2.00000E+01 -3.40000E+02   7.5
  6.00000E+01 -3.40000E+02   7.5
  6.00000E+01 -4.20000E+02   7.5
 -2.00000E+01 -4.20000E+02   7.5
INCLUDE recv_loc.txt                       (Calling for receiver geometry file)
```

Figure 2 shows and explains the structure of the file. The file contains coordinates for all transmitters and receivers and its architecture is transmitter-based. First, the number of transmitters (N_TRX) is specified. Then, for each transmitter location, all of the receivers must be listed, followed by the next transmitter location, followed by relevant receiver array (see figure 2), and so on. The number of receivers is specified as N_RECV. Each receiver location is defined by the coordinates X, Y, Z. If multiple transmitters share a common receiver array, the



```
N_TRX 91

TRX_LOOP
-600   600   50    13    0    0

N_RECV 1
-600   600   50

TRX_LOOP
-500   600   50    13    0    0

N_RECV 1
-500   600   50

TRX_LOOP
-400   600   50    13    0    0

N_RECV 1
-400   600   50
```

```
N_TRX 13

TRX_ORIG
5
-300   -62.5  1
-300    62.5  1
-175    62.5  1
-175   -62.5  1
-300   -62.5  1

N_RECV 7
-600   600   50
-500   600   50
-400   600   50
-300   600   50
-200   600   50
-100   600   50
   0   600   50

TRX_ORIG
5
0    -62.5   1
```

Figure 2. **(A)** Tx-Rx geometry for airborne TDEM survey (TRX_LOOP is used as an example); **(B)** Tx-Rx geometry for ground loop multiple source survey (TRX_ORIG is used as an example).

latter can be included in a separate 3-column *"recv_loc.txt"* file and specified by for each transmitter by adding "*INCLUDE recv_loc.txt*" instead of "*N_RECV*".
The transmitter label (i.e., TRX_ORIG) indicates the type of transmitter. There are four options for describing the transmitter source:

- TRX_LOOP: circular loop with arbitrary orientation
- TRX_MAGNETIC_DIPOLE: magnetic dipole
- TRX_ORIG: connecting individual wire segments to form a grounded or inductive source.
- TRX_LINES: primary field is generated from analytic expressions for wires in free space.

TRX_LOOP: an analytical circular loop (Figure 2A). This type is described by X, Y, Z location, circular loop radius. This type is a good approximation for large airborne transmitters (VTEM, HeliGeotem, AeroTEM II and IV, HoisTEM, NewTEM, etc). The following is an example of the TRX_LOOP source description format:

```
TRX_LOOP
x y z  radius theta alpha
```

Here x, y and z are the coordinates, `"radius"` is the loop radius, `"theta"` is the vertical angle from positive Z (up) axis and `"alpha"` is the vertical angle from positive Y (North) axis. For a loop parallel to XY plane use alpha = theta = 0.

TRX_MAGNETIC_DIPOLE : an analytical magnetic dipole (Figure 2A). This is a good approximation of small radius airborne TDEM systems (AeroTEM I, II) and small loop ground TDEM systems (Geonics EM61, EM63, etc). The following is an example of the TRX_MAGNETIC_DIPOLE source description format:

```
TRX_MAGNETIC_DIPOLE
x y z theta alpha  m
```

In this file format, x, y and z are the coordinates, `"theta"` is the vertical angle from positive Z (up) axis, `"alpha"` is the vertical angle from positive Y (North) axis and `"m"` is the dipole moment of the transmitter, which should be listed in SI units. Similar to previous example, for a loop parallel to XY plane use alpha = theta = 0.

TRX_ORIG : the "original" distributed current source – closed loops or grounded wires (Figure 3B). This type is a good approximation of any conventional large square loop system (ground and airborne), including, Crone, Geonics (EM47, 57, 67), Zonge, GeoTEM, MegaTEM, MegaTEM II, SkyTEM, etc. The source in this case is described by number of nodes (4 for grounded wire or n>4 for closed loop) and XYZ coordinates for each node (see example below).

```
TRX_ORIG
n
x1 y1 z1
x2 y2 z2
:
xn yn zn
```

Conceptually the source involves a uniform current density within the cells where the source current flows. The grounded wire (Figure 2B) is strictly defined over the air/ground boundary, with nodes 1 and 4 being in the ground domain and 2, 3, in the air. For a closed loop (Figure 2B), the first and last nodes must be identical (i.e., a square loop is specified with 5 nodes). Current in the transmitter is assumed to be 1 Ampere.

TRX_LINES: an analytical general closed loop of line currents. It is designed to handle arbitrary complex transmitters with user defined number of nodes. The format for this source type is same as for the TRX_ORIG, the difference is in how the transmitter currents are handled.

The format of the ***wave file*** is similar to previous UBC-GIF codes (EH3DTD). The same wave file is used for all transmitters. Users will likely have their own wave file for their transmitter. The discretization of the waveform in the "on-time" and discretization in the off-time is of great importance. A new factorization of the modelling matrix is required whenever the stepping time "Δt" is changed. The most computationally efficient discretization is when a single value of Δt can be used for the full time range of interest. Most systems consist of an "on-time" portion (exponential, half-sign, ramp, etc) followed by an "off-time". The on-time portion of the waveform may be modelled using a large value of Δt. Data acquired in the off-time often spans a few decades of time (Figure 3).

Maxwell's equations must be time-stepped with relatively small values of Δt. The stepping time region begins one decade prior to the user-defined earliest time, and extends until the latest data time is defined. It is divided into logarithmic segments, usually a decade in length. Each segment is time stepped with a uniform Δt (linearly spaced). Generally 10-15 time steps are

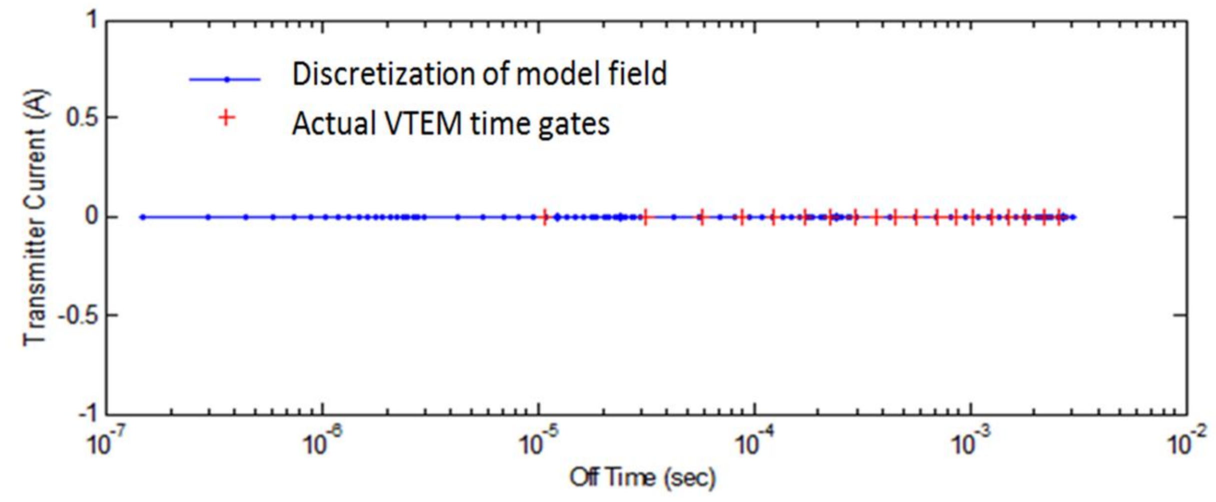


Figure 3. Example: off-time steps (blue) shown vs actual off-time data acquisition time gates for VTEM system (red).

adequate for each decade in time. The total computation time depends upon the number of factorizations, the number of time steps, and the number of transmitters.

A "Wavebuilde.exe" GUI has been generated to assist the user in generating the wave files. Figures 4, 5 and 6 show the user interface generating different types of waveforms. Among the GUI settings there are some general parameters, which specify any waveform and some other parameters specific to every waveform type in particular. Among the general parameters are the following:

Max/min: "min" and "max" values denote the beginning and end values of the time window through which equations are time-stepped. "min" should be smaller (typically about a decade) smaller than the first datum time. "max" can correspond to the last datum time.

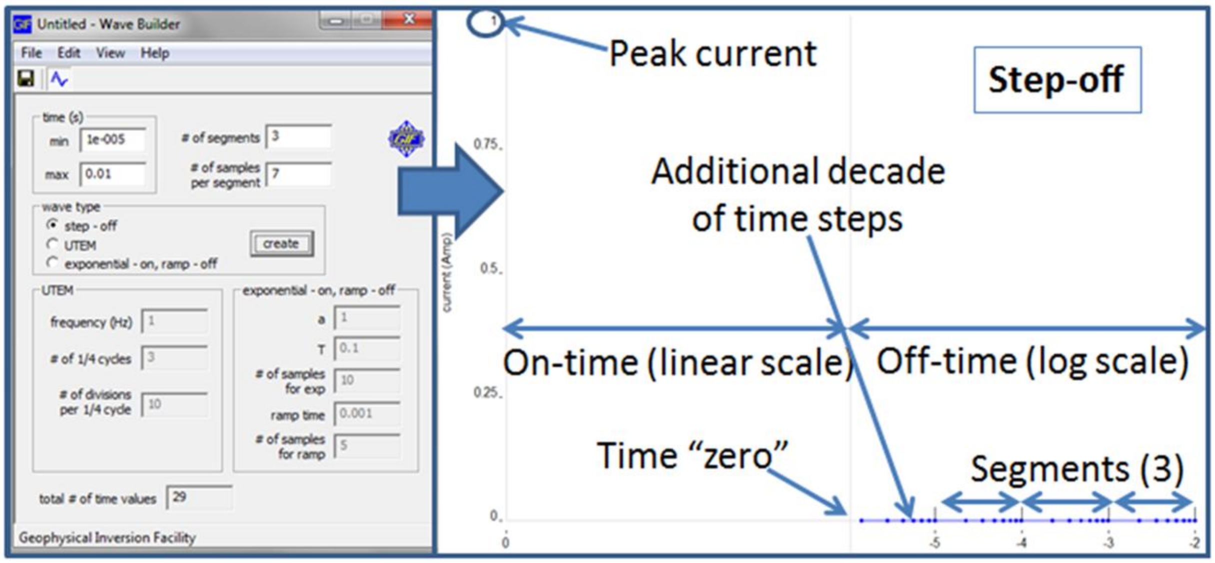
#segments and # of samples per segment are the number of logarithmically spaced segments and linearly spaced samples per each segment

Wave type: there are three options included in the present GUI.

- Step off
- UTEM
- Exponential plus a ramp

Each of these is described in further detail below. These are accompanied by explanations of the parameters of the interface that are of relevance and also plots of the times at which the equations are solved. For each waveform type all time values are referenced to the “time zero”, which is the beginning of both: factorization time steps and data acquisition time gates.

Step-off (Figure 4). The current prior to t=0 is assumed to be uniform and solution of the steady state fields are generated by the program.



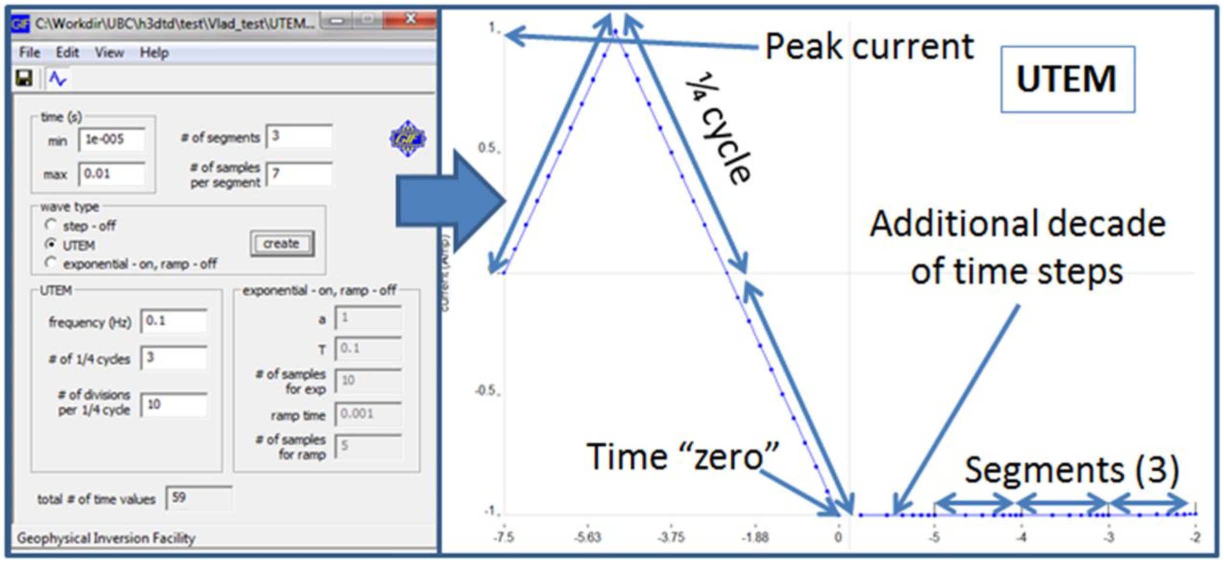
time (s): min 1e-005, max 0.01
# of segments 3
# of samples per segment 7
wave type: [✓ step - off] UTEM exponential - on, ramp - off
UTEM: frequency (Hz) 1; # of 1/4 cycles 3; # of divisions per 1/4 cycle 10
exponential - on, ramp - off: a 1; T 0.1; # of samples for exp 10; ramp time 0.001; # of samples for ramp 5
total # of time values 29

Figure 4. An example of “Step-off” waveform.

This reverse Heaviside function is the simplest waveform to model. No actual geophysical system uses this exact waveform; however recorded signals can often be deconvolved to conform to this waveform type. Peak current (maximum current) data are generally sampled only in the log domain.

UTEM (Figure 5). Also often referred to as “sawtooth”. This is another simple waveform, which is periodic defined by the frequency and the amplitude. For UTEM waveform there is no off-time, all measurements acquired in the on-time. The forward modelling begins with all field values set to zero and hence ¾ of a cycle is usually modelled prior to data collection times. The

¾ cycle can be modelled in linear time steps and following data times in logarithmic steps. The


C:\Workdir\UBC\h3dtd\test\Vlad_test\UTEM...

File Edit View Help

time (s) min 1e-005 max 0.01

# of segments 3

# of samples per segment 7

wave type: ( ) step - off [✓ UTEM] ( ) exponential - on, ramp - off

create

UTEM: frequency (Hz) 0.1; # of 1/4 cycles 3; # of divisions per 1/4 cycle 10

exponential - on, ramp - off: a 1; T 0.1; # of samples for exp 10; ramp time 0.001; # of samples for ramp 5

total # of time values 59

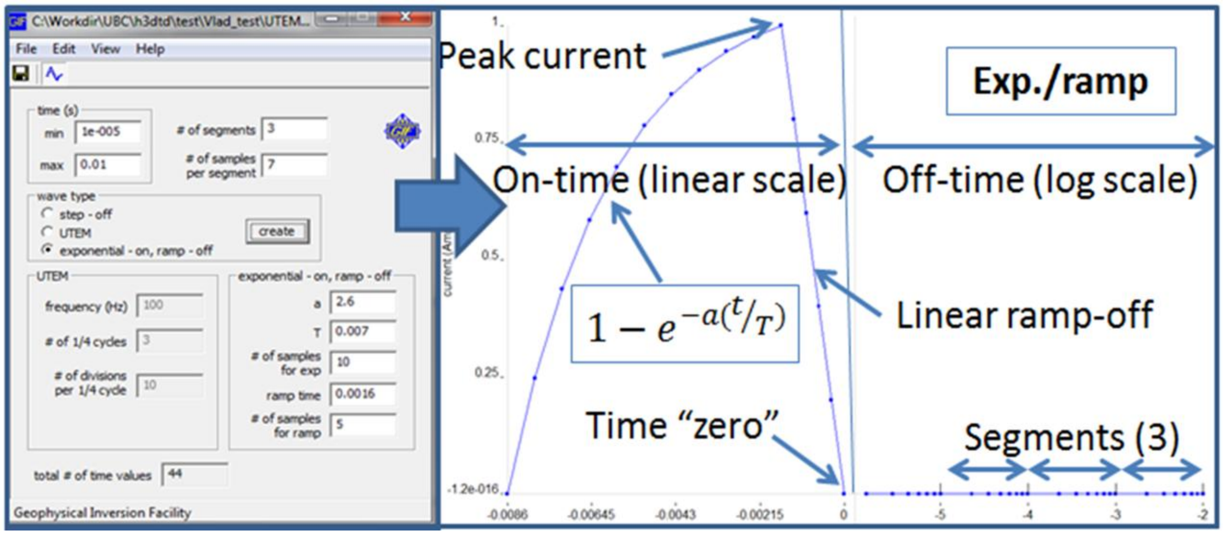
Geophysical Inversion Facility

Figure 5. An example of "UTEM" waveform.

number of quarter cycles to be modelled prior to data collection is governed by a user-defined parameter which has to be an odd number.

It is convenient to have the data times begin at t=0 and so the UTEM waveform will begin at some negative time. The time stepping for the data channel portion of the waveform is controlled by the "max-min" values on the interface.

Exponential-on/ramp-off (Figure 6) This is a general expression that can approximate the waveform from many systems.


C:\Workdir\UBC\h3dtd\test\Vlad_test\UTEM...

File Edit View Help

time (s) min 1e-005 max 0.01

# of segments 3

# of samples per segment 7

wave type: ( ) step - off ( ) UTEM [✓ exponential - on, ramp - off]

create

UTEM: frequency (Hz) 100; # of 1/4 cycles 3; # of divisions per 1/4 cycle 10

exponential - on, ramp - off: a 2.6; T 0.007; # of samples for exp 10; ramp time 0.0016; # of samples for ramp 5

total # of time values 44

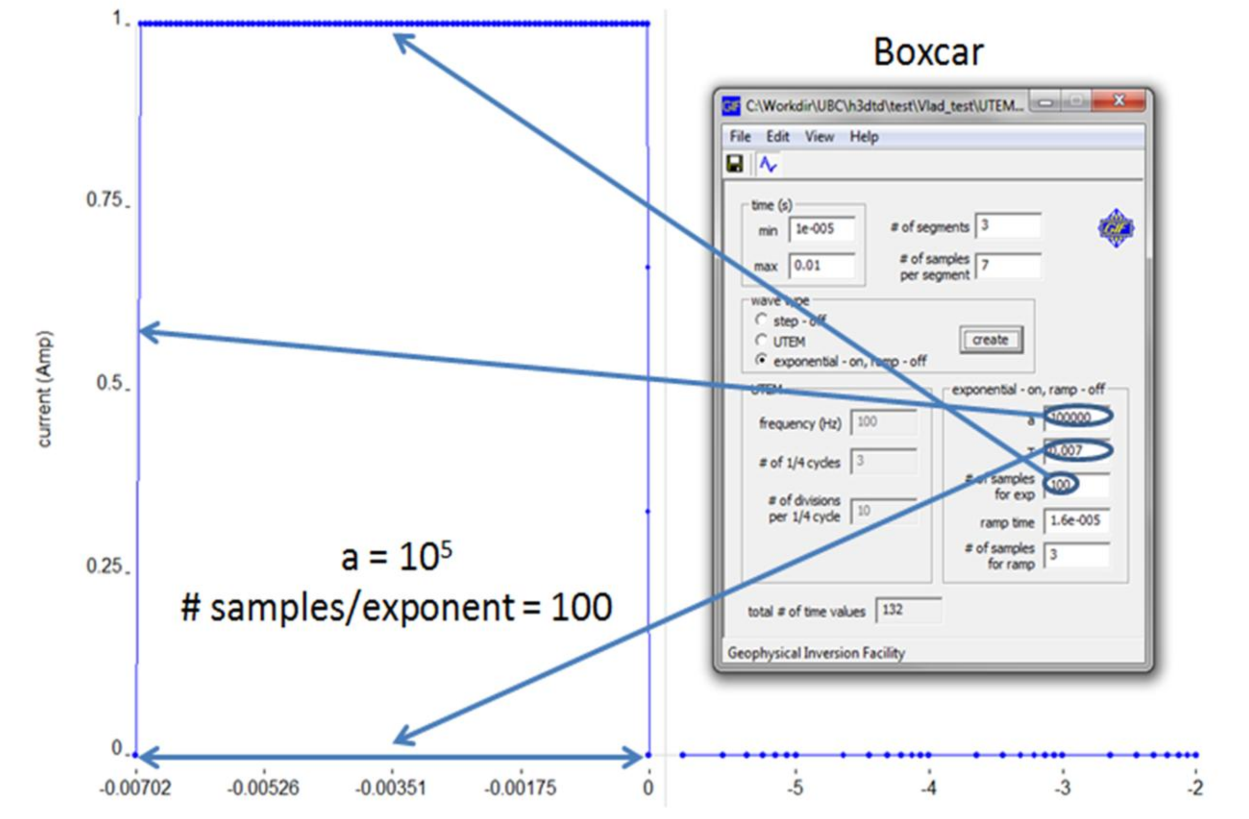
Geophysical Inversion Facility

Figure 6. An example of "Exponential rise/ramp-off" waveform.

This is an exponential rise in current followed by a ramp off. The ramp-on is defined by the equation (2):

(2) $\mathbf{I = 1 - } e^{-a(t/T)}$, where I is the current, **a** is the exponential decay coefficient, **T** is a user defined parameter equal to the desired length of the exponential ramp-on. Thus the exponential ramp is defined for 0<t<T. For later times the waveform is a ramp-off and its length is designated in the box "ramp time" in the interface. The end of the ramp is adjusted so that it corresponds to "t=0" and subsequent times are logarithmically generated by the "max-min" portion of the interface.

Figures 7 and 8 show examples of how to approximate boxcar pulse and triangular pulse with off-time, using "exponential ramp-on/linear ramp-off" waveform type by changing the exponential rise **"a"** coefficient and number of samples per exponent.



time (s): min 1e-005; max 0.01

# of segments 3

# of samples per segment 7

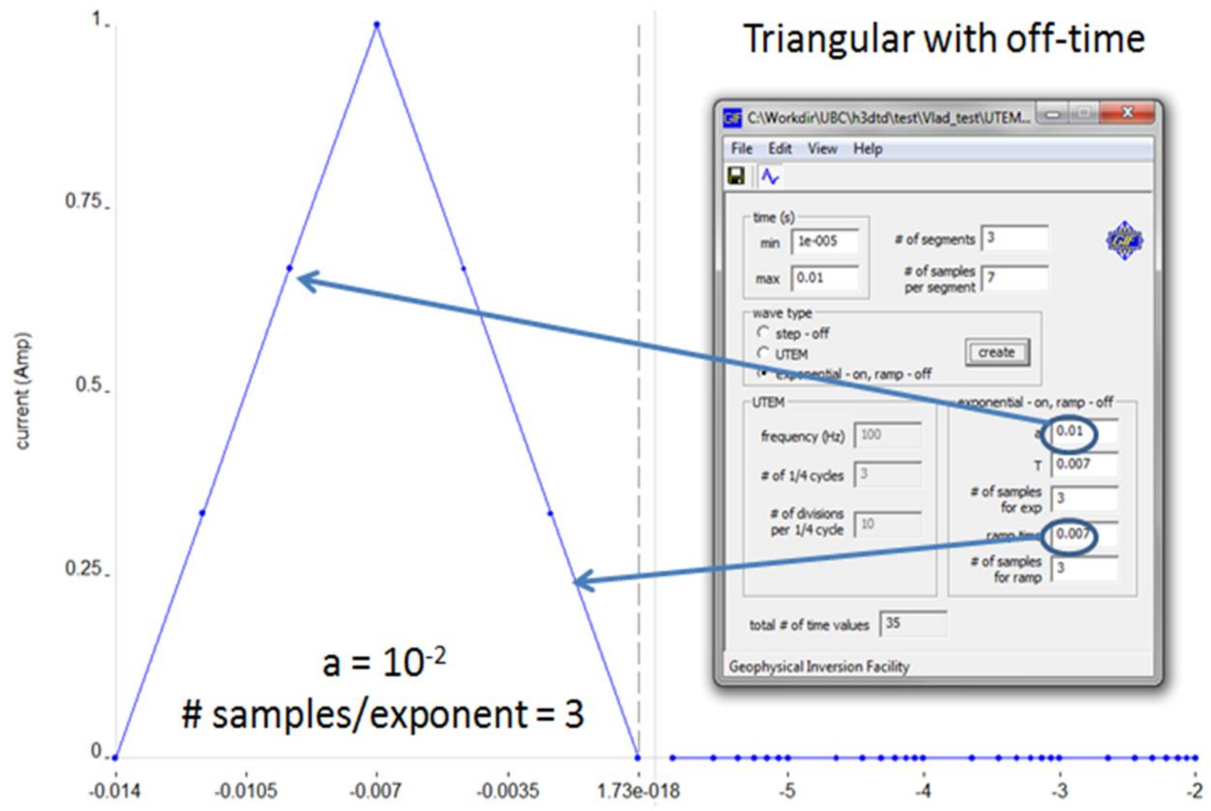
wave type: ( ) step - off; ( ) UTEM; [✓ exponential - on, ramp - off]

UTEM: frequency (Hz) 100; # of 1/4 cycles 3; # of divisions per 1/4 cycle 10

exponential - on, ramp - off: a 100000; T 0.007; # of samples for exp 100; ramp time 1.6e-005; # of samples for ramp 3

total # of time values 132

Figure 7. An example of simulating a boxcar waveform using the exponential ramp-on/linear ramp-off approximation.



time (s)
min 1e-005
max 0.01
# of segments 3
# of samples per segment 7
wave type
( ) step - off
( ) UTEM
[✓ exponential - on, ramp - off]
UTEM
frequency (Hz) 100
# of 1/4 cycles 3
# of divisions per 1/4 cycle 10
exponential - on, ramp - off
a 0.01
T 0.007
# of samples for exp 3
ramp time 0.007
# of samples for ramp 3
total # of time values 35

Figure 8. An example of simulating a triangular pulse with off-time, using the exponential ramp-on/linear ramp-off approximation.

All of the described examples are suitable for data, de-normalized by peak current; however in the *"edit"* menu *"advanced"* features there is a possibility to include user-defined peak current value. The user should be especially careful, when dealing with the dipole transmitter types, which contain peak current values in the dipole moment and ensure that current in the waveform is de-normalized by the peak value.

The waveform file can be saved under user-defined name with arbitrary extension. The format is shown below.

```
0.0000000e+000  1.0000000e+000
1.0000000e-005  0.0000000e+000  15
1.0000000e-004  0.0000000e+000  15
1.0000000e-003  0.0000000e+000  15
1.0000000e-002  0.0000000e+000  15
1.0000000e-001  0.0000000e+000  15
```

This is the example of a step-off current. The first column is the time (in seconds) and the second column is the current (in amperes). The optional third column indicates the number of equal time steps to use. The first time value in the wave file does not necessarily have to be 0. In fact, zero-time should be defined relative to the acquisition time gates, rather than beginning of the pulse in the waveform. The on-time pulse can be in the negative domain. In this particular example there will be 15 equal time steps between 0 and 10^{-5} s. The time steps will therefore be 10^{-5} / 15 = 6.67*10^{-7} s. Between 10^{-4} and 10^{-5} s, there will also be another 15 equal time steps equal to (10^{-4} - 10^{-5}) / 15 = 6.0*10^{-6} s.

The ***time gates file*** is optional. It can be defined in the 6-th line of the h3dtd.inp file, or set to "USE_WAVE" in which case the forward simulation will output model field values for the discretization times set up in the *"wave.txt"* (see figure 3). The following is an example of time gates input format:

```
1.0000e-005
1.5849e-005
2.5119e-005
3.9811e-005
:
1.0000e-002
```

In this format the time gates (seconds) are written in a single column. The name of the file is user-defined and the extension is arbitrary.

Verification of the mesh

In setting the mesh parameters, the user has to maintain a balance between saving computing time (by coarsening the mesh) and getting more accurate solution of the forward simulation by making the mesh finer. In order to verify the validity of a particular mesh, it is suggested to compare the 3D simulated decay curves to the 1D simulated decay curve. Changing the time discretization can also affect the accuracy.

The first step is building a mesh. The importance of the correct mesh design is the key to recovery of meaningful geo-electrical parameters; however a reasonable balance has to be maintained in terms of keeping the mesh discretization at balance with the computing capabilities of the workstation. For mesh validation *"Data_viewer.exe"* GUI can be used. Figure 9 shows how finer discretization of the mesh plays role in the modeling accuracy. In this particular example the computing time differs between fine mesh and coarse mesh by a factor of 10.

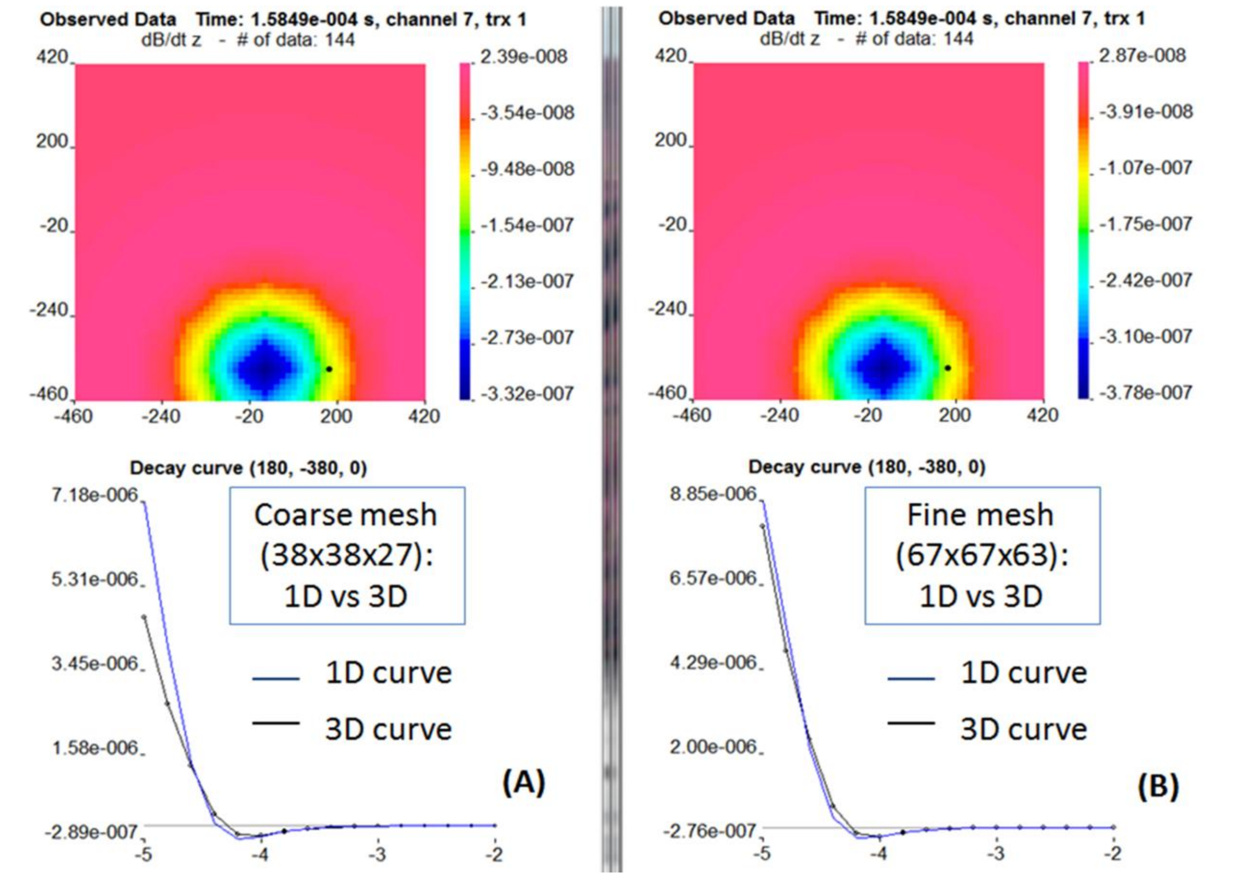


Figure 9. (A) Comparison between dBz/dt decay curves simulated for a one loop source 100 Ohm m half-space using 1D forward modeling and 3D forward modeling and a 38x38x27 mesh; (B) similar comparison for a 67x67x63 mesh.

Running the forward model

It is recommended that for every forward model, a separate folder should be created with all the control files stored there. This folder should be your working directory (further referred to as "*workdir*") Every time any of the modeling parameters are changed, a new "workdir" should be created specifically for the new model settings. Separate from the working directory, an executable directory (*"execdir"*) should exist on the workstation. It is not recommended to keep executable files in the same directory as the control files. The following table is showing, which Files should be kept in which directory (please note that not all files listed in the *"workdir"* may be necessarily needed for running the code, the bare minimum is having the "*h3dtd.inp"; "mesh.txt"; "wave.txt" and "trx_loc.txt",* the rest of the files is optional):

Execdir	**Workdir**
h3dtd.exe	h3dtd.inp
mkl_mc3.dll	mesh.txt
mkl_intel_thread.dll	wave.txt
mkl_def.dll	model.con
mkl_core.dll	model.sus
mkl_blacs_msmpi_lp64.dll	trx_loc.dat
mkl_blacs_mpich2_lp64.dll	rcv_loc.dat
mkl_blacs_lp64.dll	time_gates.txt
mkl_blacs_intelmpi_lp64.dll	machinefile.txt
libiomp5md.dll	

Table 2. List of files specific for *"workdir"* and *"execdir".*

For optimal performance of the forward modelling (H3DTD) it is best to use the Message Passing Interface (MPI), which allows running multiple computational devices in parallel, including commodity clusters, hi-speed networks and multi-core processors on local computers. In order to install the MPI application library, download it from http://www.mcs.anl.gov/research/projects/mpich2/

Linux:

For commodity clusters operated under Linux system, the code can be run on any number of processors listed in a description ASCII file. The following is an example of such description file:

```
Compname01:nProc
Compname02:nProc
Compname03:nProc
```

The description file name is completely arbitrary. `“Compname”` is the network name of the computer to be used and `“nProc”` is the number of processors to be employed for the procedure. If the computer is on the local network and can be directly accessed, then no path is needed to be specified. The following is an example of a command line to be used under Linux operating system in order to start the forward simulation:

```
mpiexec -machinefile machines.txt -n 20 ./h3dtd_mumps
```

In this command line `“-machinefile”` calls for a description file “machines.txt” and “–n 20” indicates a total amount of processors to be used on all the machines listed in the description file.

<u>Windows:</u>

For single multi-core computer usage, under Windows operating system, the command line will look like this:

```
"C:\Program Files\MPICH2\bin\mpiexec.exe" -localonly 4 h3dtd
```

In this line `“-localonly`” limits the computation to only one machine, from which the command line is launched and `“4”` specifies the number of processors to be used..

For running the code on several computers or a network:

- Make sure each computer has the same version of MPI installed.
- The user running the program should have the same “UserID” and “password” on each computer (the user does not have to be logged on to every computer, but has to have a network account set up on each computer with same identification).

- Make your *“workdir”* and *“execdir”* folders shareable, and make sure that *“workdir”* provides full sharing (read and write).
- Make sure your shared folders are visible on each computer then place the input files and executables in sharable folders, as per Table 2.
- The firewall in each computer (except for the host) should be turned off, or else the h3dtd program should be added to the exceptions list.

The command line to start the MPI job is:

```
mpiexec.exe -machinefile machines.txt -n 8 -priority 1 -dir
\\MYCOMP\share \\MYCOMP\share\exe\h3dtd.exe
```

In the above example:

`machines.txt` - file containing the names of the computers to use. Each computer name can be followed be `:p` which indicates the number of processes to start on that machine.

`-n 8` - total number of processes to start.

`-priority 1` - (optional) indicates that all jobs should be started at low priority.

`-dir \\MYCOMP\share` - sharable folder that should be visible on all computers.

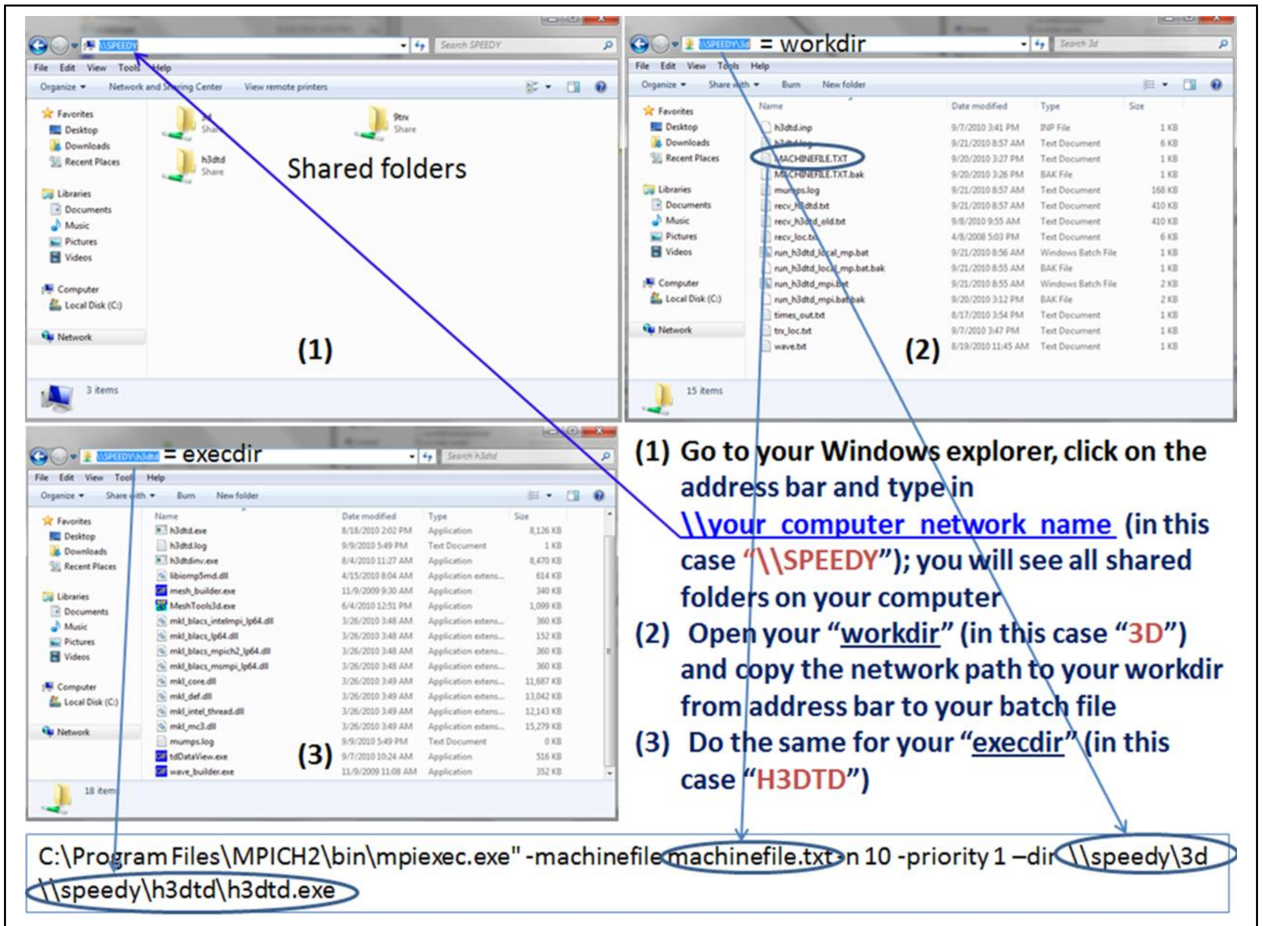
`\\MYCOMP\share\exe\h3dtd.exe` - full path of the executable. Other computers must be able to see it.

For convenience, it is recommended to set up batch files (*.bat) for running H3DTD.exe on a local network. Two examples of such batch files are provided with the documentation. The file *“run_h3dtd_local_mp.bat”* is to be used for running H3DTD on local workstation alone and the file *“run_h3dtd_mpi.bat*” is to be used for running the code on the local network.

In editing the file: *“run_h3dtd_local_mp.bat”* the editable text includes the path to the h3dtd.exe file, which should be set to your “execdir” location. The editable parts of the *“run_h3dtd_mpi.bat”* are listed in Figure 10 and include the following:

- machinefile.txt
- number of processors
- path to “workdir”

- path to "execdir"



```
Name                        Date modified        Type                 Size
h3dtd.inp                   9/7/2010 3:41 PM     INP File             1 KB
h3dtd.log                   9/21/2010 8:57 AM    Text Document        6 KB
MACHINEFILE.TXT             9/20/2010 3:27 PM    Text Document        1 KB
MACHINEFILE.TXT.bak         9/20/2010 3:26 PM    BAK File             1 KB
mumps.log                   9/21/2010 8:57 AM    Text Document        168 KB
recv_h3dtd.txt              9/21/2010 8:57 AM    Text Document        410 KB
recv_h3dtd_old.txt          9/8/2010 9:55 AM     Text Document        410 KB
recv_loc.txt                4/8/2008 5:03 PM     Text Document        6 KB
run_h3dtd_local_mp.bat      9/21/2010 8:56 AM    Windows Batch File   1 KB
run_h3dtd_local_mp.bat.bak  9/21/2010 8:55 AM    BAK File             1 KB
run_h3dtd_mpi.bat           9/21/2010 8:55 AM    Windows Batch File   2 KB
run_h3dtd_mpi.bat.bak       9/20/2010 3:12 PM    BAK File             2 KB
times_out.txt               8/17/2010 3:54 PM    Text Document        1 KB
trx_loc.txt                 9/7/2010 3:47 PM     Text Document        1 KB
wave.txt                    8/19/2010 11:45 AM   Text Document        1 KB
```

```
Name                          Date modified        Type                 Size
h3dtd.exe                     8/18/2010 2:02 PM    Application          8,126 KB
h3dtd.log                     9/9/2010 5:49 PM     Text Document        1 KB
h3dtdinv.exe                  8/4/2010 11:27 AM    Application          8,470 KB
libiomp5md.dll                4/15/2010 8:04 AM    Application extens...  614 KB
mesh_builder.exe              11/9/2009 9:30 AM    Application          340 KB
MeshTools3d.exe               6/4/2010 12:51 PM    Application          1,099 KB
mkl_blacs_intelmpi_lp64.dll   3/26/2010 3:48 AM    Application extens...  360 KB
mkl_blacs_lp64.dll            3/26/2010 3:48 AM    Application extens...  152 KB
mkl_blacs_mpich2_lp64.dll     3/26/2010 3:48 AM    Application extens...  360 KB
mkl_blacs_msmpi_lp64.dll      3/26/2010 3:48 AM    Application extens...  360 KB
mkl_core.dll                  3/26/2010 3:49 AM    Application extens...  11,687 KB
mkl_def.dll                   3/26/2010 3:49 AM    Application extens...  13,042 KB
mkl_intel_thread.dll          3/26/2010 3:49 AM    Application extens...  12,143 KB
mkl_mc3.dll                   3/26/2010 3:49 AM    Application extens...  15,279 KB
mumps.log                     9/9/2010 5:49 PM     Text Document        0 KB
tdDataView.exe                9/7/2010 10:24 AM    Application          516 KB
wave_builder.exe              11/9/2009 11:08 AM   Application          352 KB
```

(1) Go to your Windows explorer, click on the address bar and type in \\your computer network name (in this case "\\SPEEDY"); you will see all shared folders on your computer

(2) Open your "workdir" (in this case "3D") and copy the network path to your workdir from address bar to your batch file

(3) Do the same for your "execdir" (in this case "H3DTD")

```
C:\Program Files\MPICH2\bin\mpiexec.exe" -machinefile machinefile.txt -n 10 -priority 1 –dir \\speedy\3d
\\speedy\h3dtd\h3dtd.exe
```

Figure 10. Block diagram for editing batch file in order to run the code on the network.

In figure 10, the file *"machinefile.txt"* used for executing the code under local network has the same format as the example shown above for Linux environment and should be located in the *"workdir".* Please note that for running the code on multiple network computers, the total number of processors should be specified, equal to the sum of all processors on all computers listed in *machinefile.txt*.

Output files

h3dtd.log - a log file showing the progress of the inversion.

times_out.txt - a file containing predicted data.

Mumps.log - a file showing the factorizations and cpu time.