# Seismic Location Base Model Software Design Document Version 3.0

Sandy Ballard
Sandia National Laboratories
May 6, 2013

The Seismic Location Base Model (SLBM) software library computes seismic travel time through a 2½D velocity model. In this document, the main elements of the software are described. In the first section, the model grid used to store the velocity model through which travel times are to be calculated is described. This is followed by sections that describe how travel times are calculated for phases Pn/Sn and for Pg/Lg. There is a final section that describes the architecture of the software.

## Model Grid

The SLBM Earth Model is defined on a tessellation that spans the entire globe at a constant resolution of approximately 1º (Figure 1). In the range 0.5º ≤ latitude ≤ 84.5º and -19.5º ≤ longitude ≤ 149.5º, Earth model information is obtained from the Unified Model and everywhere else, Earth model information is obtained from the CRUST2.0 model (http://mahi.ucsd.edu/Gabi/rem.html ).

Velocity of the Crust and Top of the Mantle

Both the Unified and the CRUST2.0 models are defined on regular latitude, longitude grids. The Unified model has 7 crustal layers defined on a 1º × 1º grid. The CRUST2.0 model also has 7 crustal layers defined on a 2º × 2º grid. Unfortunately, the definitions of the 7 crustal layers do not agree exactly in the two models (see Table 1).

**Table 1 - Comparison of layer definitions in the Unified, CRUST2.0 and SLBM models.**

| Layer # | Unified Model | CRUST2.0 | SLBM |
|---|---|---|---|
| 0 | Water | Water | Water |
| 1 | Sediment1 | Ice | Sediment1 |
| 2 | Sediment2 | Soft sediment | Sediment2 |
| 3 | Sediment3 | Hard sediment | Sediment3 |
| 4 | Upper crust | Upper crust | Upper crust |
| 5 | Middle crust | Middle crust | Middle crust (Pn, Sn) |
| 6 | Lower crust | Lower crust | Middle crust (Pg, Lg) |
| 7 | Mantle | Mantle | Lower crust |
| 8 | | | Mantle |

The nodes in the SLBM model are not located at the same positions on the Earth as the nodes in the 2 models upon which it is based. Bilinear interpolation was used to interpolate layer thickness and velocity information from the Unified and CRUST2.0 models, at the positions of the SLBM grid nodes.

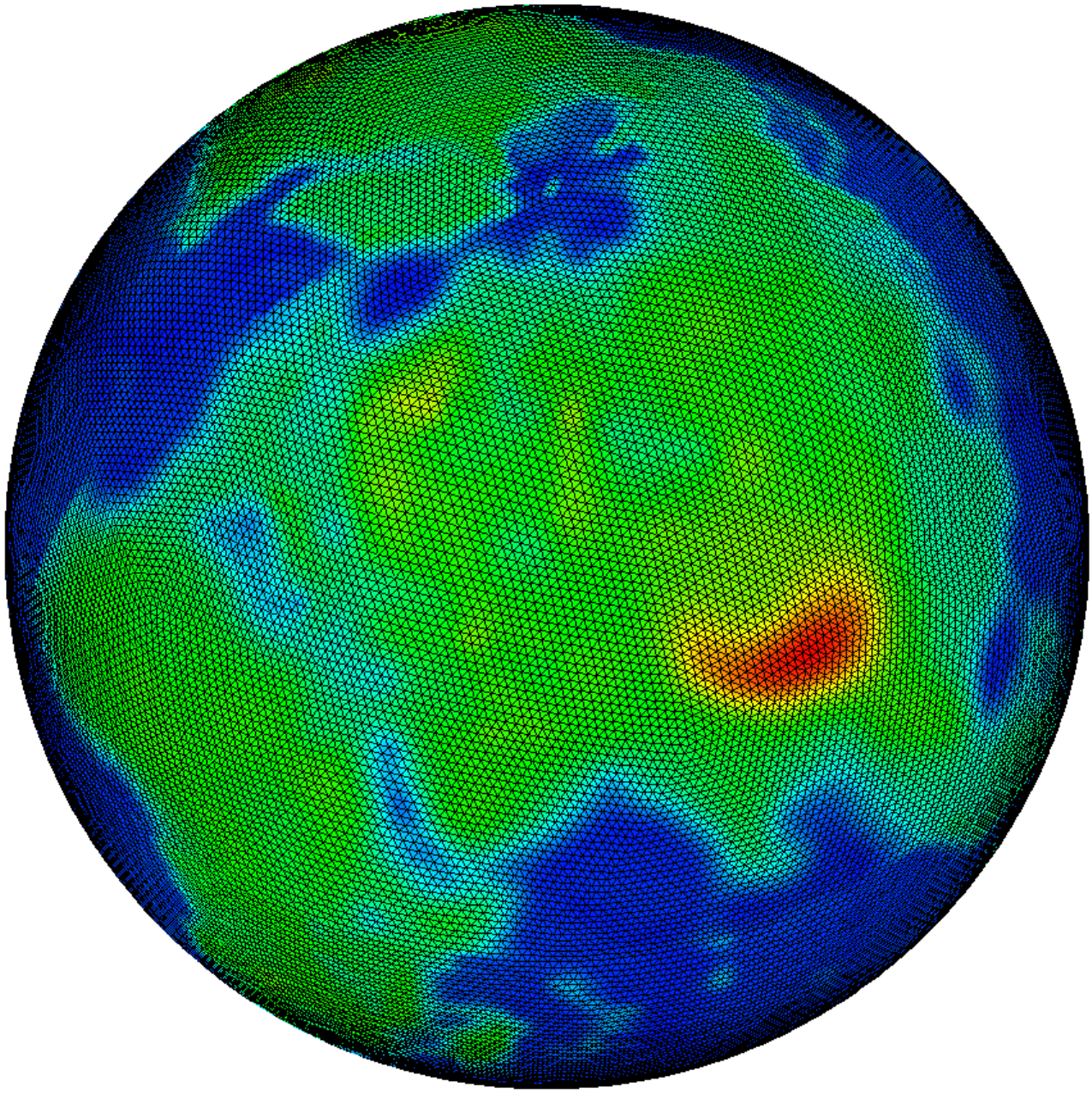

Figure 1 – Orthographic projection of the Earth showing a portion of the SLBM tessellation grid. Colors are contours of the Moho depth.

Some intervals in the Unified Model have zero thickness and velocity, which is a problem when using bilinear interpolation. Consider a Unified model cell defined by 4 nodes where one of the nodes has zero thickness and velocity. In general, the interpolation coefficient of the zero node will be non-zero so the zero velocity value will be erroneously averaged in to the interpolated value. To circumvent this difficulty, every node in the Unified Model was scanned for zero thickness intervals. When a zero thickness interval was discovered, the node's eight neighbors were visited. If any of the eight neighbors had non-zero thickness, then the zero thickness node was assigned the average value of all the non-zero thickness neighboring nodes.

Velocity Gradient in the Mantle

The Unified model has 16 layers defined for the mantle while only a single mantle layer is desired for the SLBM model. The mantle layer for the SLBM model was assigned velocity values equal to $V_{top,8}$, and the SLBM P and S velocity gradients were computed as

$$\frac{dV}{dz} = \frac{V_{top,8} - V_{bottom,10}}{Z_{top,8} - Z_{bottom,10}}$$

where $V_{top,8}$ is the P or S velocity at the top of the mantle, $Z_{top,8}$ is the depth of the Moho, $V_{bottom,10}$ is the velocity at the bottom of layer 10 in the Unified Model, and $Z_{bottom,10}$ is the depth of the bottom of layer 10 in the Unified Model, which corresponds to a depth of 135 km.

The CRUST2.0 Model does not specify a velocity gradient for the mantle so both P and S wave velocity gradients were set to velocity gradients calculated from the IASP91 travel time model, $1.17647 \times 10^{-4}$ sec$^{-1}$ for P and $3.52941 \times 10^{-4}$ sec$^{-1}$ for S.

# Model Storage Format

## SLBM Version 2

In SLBM Version 2, the model was stored in a custom binary format. SLBM Version 3 can still load and save models in the old Version 2 format. SLBM version 2 models are stored in 3 binary files and 8 ascii files. The first binary file contains the definition of the tessellation which is comprised of the latitudes and longitudes of the nodes of the tessellation and a list of the indexes of the nodes that comprise each triangle. The second file defines a list of 'stacks' which are defined below. The third file defines the connectivity between the nodes of the tessellation and the stacks. The relationship between stacks and nodes is one-to-many in the sense that each stack may belong to one or more nodes but each node must be connected to exactly one stack.

For the unified/crust2.0 hybrid model, there are 24,862 stacks, 40,962 nodes, 81,920 triangles and 24 data items per stack. The model requires about 4.4 MB of disk space and occupies about 10.7 MB of memory when loaded.

*Stacks*

A stack is a description of the layers that would be encountered at one or more locations in the model (see Figure 2). It describes the depth of the layer interfaces, relative to the surface of the solid Earth, the P and S velocities of the layers and the P and S velocity gradients in the mantle. A stack is not associated with any particular position on the Earth (nodes do that; see next section). A stack is consists of the following information:

P velocity, S velocity of layer 2
Depth of top, P velocity, S velocity of layer 3
Depth of top, P velocity, S velocity of layer 4
Depth of top, P velocity, S velocity of layer 5
Depth of top, Pn velocity, Sn velocity, Pg velocity, Lg velocity of layer 6
Depth of top, P velocity, S velocity of layer 7
Depth of top, P velocity, S velocity, P gradient and S gradient of layer 8

Depths are in km relative to the surface of the solid Earth, velocities in km/sec and gradients in $sec^{-1}$. The true depths of the interfaces, at any position on the earth, are determined by subtracting the elevation of the topographic/bathymetric surface at the specified position.

*Nodes*

A node associates a particular position on the earth with a reference to a stack. Note that latitude is geographic latitude as defined for a GRS80 ellipsoid. Elevation corresponds to the topography for dry land and to the bathymetry in the oceans. For layers consisting of solid material (layers 2 through 8) the interface depths at a node are determined by subtracting the elevation of the node from the depth of the layer as recorded in the referenced stack. If a node has a positive water thickness, and the elevation of the node is negative, then the thickness of the water layer is set to the absolute value of the elevation in libslbm. If the elevation of a node is positive, or if the nominal water thickness is less than or equal to zero, then the water thickness is set to zero.

*Triangles*

Each record lists the indexes of the three nodes that define the corners of the node, listed in clockwise order. The node indexes refer to the order in which the nodes were listed in the section of the file where the nodes were described. Node indexes are zero based.

SLBM assumes that the first 42 nodes specified in the file are nodes that are included in what are called ‘special’ triangles. These triangles are special in the sense that they are approximately evenly spaced over the surface of the earth.

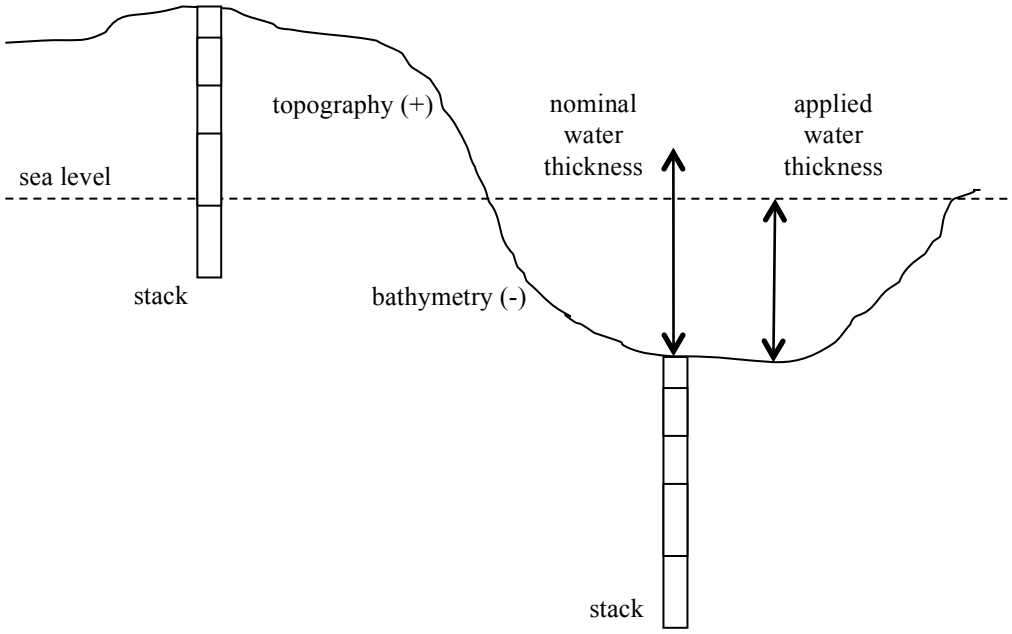


Figure 2 – Illustration of how various model components relate to each other in the vertical dimension. The top of Layer 2 in each Stack is connected to the topography.

## SLBM Version 3

In SLBM version 3, the model is stored in GeoTess format. GeoTess is an open source general-purpose model parameterization and software support system for the managing 2D and 3D Earth models. For more information about GeoTess visit the website at www.sandia.gov/geotess.

While the model storage format is different, the basic content of the model is the same. It is deployed on a uniform triangular tessellation with each triangle having an edge length of approximately 1° (Figure 1). The nodes of the new format are located at exactly the same geographic positions as in the old format.

The concept of a *geostack* has been abandoned in the version 3 parameterization. There is a radial profile at each vertex of the GeoTessGrid. This removes the necessity to store the water thickness and elevation at each profile. These values have been incorporated directly into the radial positions of the profile nodes at each grid vertex.

The primary advantage of the GeoTessModel parameterization is that GeoTess implements the natural neighbor and linear interpolation algorithms for the interpolation of model values in the geographic dimensions. SLBM version 2 was only able to interpolate model values using linear interpolation.

Another advantage of the GeoTess parameterization is that the GeoTessExplorer application can be used to extract information from the model. In format 3, the file *geotessmodel* can be used as input to GeoTessExplorer, as can the models stored in format 4. See the GeoTess website to download GeoTessExplorer.

## RSTT Model File Formats

The following formats are defined. Formats 1 and 2 are available in RSTT version 2. Formats 3 and 4 are added in RSTT version 3.

1. RSTT version 1 ascii format. This is the format we used in SLBM version 1 and early in the development of version 2. It was loaded and saved using the methods *loadVelocityModel*() and *saveVelocityModel*(). This format was never widely used. In SLBM versions 1 and 2 the format did not include the uncertainty files but in SLBM version 3 the uncertainty information has been appended onto the bottom of the files. Use of this format is not recommended.
2. RSTT version 2 binary directory format. This is the principal format used with SLBM version 2. A model is stored in a directory with the same name as the model. The directory contains the following files:
    a. *Geostack – layer boundaries and velocities, and mantle gradient information.*
    b. *Connectivity – connection between grid nodes and geostacks*
    c. *Uncertainty_Pn_TT.txt*
    d. *Uncertainty_Pn_Sh.txt*
    e. *Uncertainty_Sn_TT.txt*
    f. *Uncertainty_Sn_Sh.txt*
    g. *Uncertainty_Pg_TT.txt*
    h. *Uncertainty_Pg_Sh.txt*
    i. *Uncertainty_Lg_TT.txt*
    j. *Uncertainty_Lg_Sh.txt*

    The grid definition is stored in a separate directory called *tess* that resides at the same directory level as the model directory. The grid definition consists of two files. The name of the first one is the MD5 hash of the grid information and is the only file used by RSTT. The second file is named *<md5_hash>_adjacency* and is not used by RSTT.
3. RSTT version 3 binary directory format. This is identical to the RSTT version 2 binary directory format except that the two files *geostack* and *connectivity* are replaced with a single binary file called *geotessmodel*. The *geotessmodel* file is in GeoTessModel format, but extended to include the average P and S velocity of the mantle. The grid definition is still stored in a separate file in the *../tess* directory and uses the MD5 hash for the name, but the grid file is in standard GeoTessGrid format.
4. RSTT version 3 single-file format. All model information is stored in a single file. This includes the model values, the grid geometry, grid connectivity and the uncertainty information. This format is an extension of the standard GeoTessModel binary format. The model values, grid geometry and grid connectivity is standard GeoTess. The standard

GeoTess format is extended to add the average P and S velocity of the mantle and all of the uncertainty information. If the file extension is '.ascii' then the file will be saved/loaded in ascii text format, otherwise it is saved/loaded in binary format.

Note that the single file in format 4 and the file *geotessmodel* in format 3, are GeoTessModel files that have been extended to include additional information specific to RSTT. The custom information includes the average mantle velocities (2 values of type double) and, in the case of format 4, all of the uncertainty information. The GeoTessExplorer Java application can be applied to these files to extract model information, just like any other GeoTessModel file. While GeoTessExplorer cannot access the custom information specific to RSTT, it can access all the velocity and gradient information in the model, the layer boundaries, etc.

## IO Methods

In this section, the main IO methods implemented in SLBM are described. The first two subsections describe how input and output were managed in SLBM version 2 and the following two subsections describe the changed implemented in version 3.

### *Input – Version 2*

```
1. loadVelocityModel ( string modelID )
```

Load a model in format 1 (version 1 ascii format). This method has been seldom used since early in the development of Version 2.

```
2. loadVelocityModelBinary( string modelDirectory )
```

Load a model stored in format 2 (version 2 binary directory format). This is the only method widely used by current RSTT users.

### *Output – Version 2*

```
1. saveVelocityModel (string modelFile)
```

Save a model in format 1 (version 1 ascii format). The uncertainty files were not output by this method. This method has been seldom used since early in the development of Version 2.

```
2. specifyOutputDirectory( string modelDirectory )
3. saveVelocityModelBinary()
```

Save a model in format 2 (version 2 binary directory format ). Saving a model was a two-step process. The application first specifies the name of the directory where the model is to be saved and then calls *saveVelocityModelBinary()* to actually write the files.

### *Input – Version 3*

```
1. loadVelocityModel ( string modelSpecifier )
```

This is now the preferred method to call to load a model from file(s). It will determine the format of the model on the file system (Formats 1-4 described above) and load the model.

```
2. loadVelocityModelBinary( string modelDirectory )
```

This method is deprecated in Version 3 but is still available. It simply calls *loadVelocityModel(modelSpecifier)* described above.

### *Output – Version 3*

```
1. saveVelocityModel (string modelFile, int modelFormat=4)
```

Save the model to the file system in the specified format. A model can be saved in any of the model formats (1-4) by specifying the desired format number. The model format specifier is optional and defaults to 4 (version 3 single file format).

```
2. specifyOutputDirectory( string modelDirectory )
3. saveVelocityModelBinary()
```

This pair of methods is deprecated but still available. It saves the model in format 3 (version 3 binary directory format).

## Phases Pn/Sn

The total travel time for a Pn or Sn phase is composed of 4 components: the time required to 1) travel from the source to the Moho, 2) to travel along the Moho from the source to the receiver, 3) to travel from the Moho up to the receiver, and 4) a correction for the mantle gradient and the sphericity of the Earth. The method used to compute the travel time and horizontal offset through the crust is described in subsection 1 below. The algorithm for computing the total travel time for crustal events is described in the second subsection. The calculation of travel time for mantel events is described in the final subsection.

1. Crustal Legs

Calculation of travel time for phases Pn and Sn through the crust is implemented in file CrustalProfile.cc. The crustal legs model the crust as a sequence of *n* layers each of which is characterized by a constant velocity. Equations 3.66 and 3.68 in Lay and Wallace (1995) are integrated across each layer to obtain the horizontal offset and travel time of the ray as it

traverses the layer. The total horizontal offset across the crust, beneath either the source or the receiver, is given by

$$x_{crust} = \sum_{j=1}^{n} \left[ \cos^{-1}(pv_j / r_j) - \cos^{-1}(pv_j / r_{j+1}) \right] \quad (1)$$

Where $p$ is the ray parameter, $v_j$ is the velocity of layer $j$, $r_j$ is the distance from the center of the Earth to the top of layer $j$ and $r_{n+1}$ is the radius of the Moho.

The total travel time across the crust beneath either the source or the receiver is given by

$$t_{crust} = \sum_{j=1}^{n} \left[ \sqrt{r_j^2 / v_j^2 - p^2} - \sqrt{r_{j+1}^2 / v_j^2 - p^2} \right] \quad (2)$$

In equations 1 and 2, the uppermost layer in the Unified Model, which corresponds to the WATER layer, and all layers with zero thickness, are ignored. $r_1$ is the radius of the source/receiver. If $r_1$ is less than the radius of the top of Layer2, then $v_1$ is velocity of the layer in which the source/receiver is located. If $r_1$ is greater than the top of Layer2, then $v_1$ is assumed to be the velocity of the outermost layer in the Earth model that is not composed of WATER and that has non-zero thickness. $r$ is assumed to be monotonically decreasing, which means that zero-thickness layers and layers whose tops are shallower than the source/receiver radius are ignored.

It should also be noted that the interpretation of Layer 6, corresponding to the MIDDLE_CRUST, depends on the phase. There are two versions of the MIDDLE_CRUST Layer in the model, one for Pn/Sn and another for Pg/Lg. Both versions have the same radius of the top of the layer but have different P and S velocities.

2. Travel Time for Crustal Event

If the event occurred in the crust, then the following procedure is implemented to compute the total travel time. This algorithm is implemented in file GreatCircle_Xn.cc, method computeTravelTimeCrust().

1. Subdivide the angular separation of the source and receiver, $d_s$, into $N$ angular increments, each of length $\delta$. Find the radius of the Moho, $r_i$, the velocity at the top of the mantle, $v_i$, and the mantle gradient, $g_i$, at the center of each angular increment.

2. Initialize the ray parameter, $p$, to the smaller of the critical ray parameters below the source and receiver. The critical ray parameter is given by

   $$p = r_{Moho} / v_{mantle}$$

   Where $r_{Moho}$ is the radius of the Moho and $v_{mantle}$ is the velocity at the top of the mantle at the source/receiver location.

3. Compute the angular horizontal offsets below the source and receiver, $x_s$ and $x_r$, using equation 1.

4. If $x_s + x_r > d_s$, then abort the calculation.

5. Calculate the horizontal distance in km between the source and receiver pierce points, $x_m$, the path averaged mantle velocity $V_m$, and the path averaged velocity gradient in the mantle, $G_m$:

$$x_m = \sum_1^N \delta_i r_i$$

$$V_m = \left[ \sum_1^N v_i \delta_i r_i \right] / x_m$$

$$G_m = \left[ \sum_1^N g_i \delta_i r_i \right] / x_m$$

where $\delta_i$ is the portion of each angular increment computed in step 1 that lies between the source and receiver pierce points.

6. If $G_m < 0$, set $G_m = 0$.

7. Calculate $C_m$

$$C_m = \left( \frac{G_m}{V_m} + 1.58 \times 10^{-4} \right)$$

8. Calculate the depth of the turning point below the Moho (Zhao, 1993, Appendix A).

$$H = \frac{\left( x_m^2 C_m^2 / 4 + 1 \right)^{1/2} - 1}{C_m}$$

9. Calculate the turning radius of the ray in the mantle

$$R_{turn} = (r_{Moho,source} + r_{Moho,receiver}) / 2 - H$$

and the velocity at the turning point

$$V_{turn} = V_m + G_m H$$

10. Determine a revised estimate of the ray parameter

$$p = R_{tum} / V_{tum}$$

11. Repeat steps 3 through 10 until *p* stops changing significantly.

12. Compute the travel time of the head wave along the Moho as

$$t_m = \sum_{1}^{N} \delta_i r_i / v_i$$

13. Calculate the *c* parameter (Zhao, 1993)

$$c = G_m / V_o + 1.58 \times 10^{-4}$$

where $V_0$ is the mantle velocity averaged over whole Earth model.

14. Calculate the travel time correction due to the mantle gradient

$$t_\gamma = -\frac{x_m^3 c^2}{24 V_0}$$

15. The total travel time is then given by

$$T_{total} = t_s + t_r + t_m + t_\gamma$$

where $t_s$ and $t_r$ are the travel times for the crustal legs below the source and receiver computed using equation 2.

3. Travel time for Events in the Mantle

If the event occurred in the mantle, then the following procedure is implemented to compute the total travel time. This algorithm is implemented in file GreatCircle_Xn.cc in method computeTravelTimeMantle(). See Figure 3 for help in interpreting the meaning of variables defined below.

**Known Information:**

- z = depth of event
- $z_m$ = depth of event below moho
- $m_s$ = moho depth below station
- $m_e$ = moho depth above event
- $d_s$ = distance (km) from station to event (surface)
- $d_m$ = distance (km) from station to event (moho)
    - $d_m = d_s(6371 - m_s)/6371$
- $V_o$ = average moho velocity over entire model (as constant so inversion is linear)

- r = radius ratio of event, $r = r_{mantle}/(r_{mantle} - z_m)$, $r_{mantle}$ is moho radius above event.
- $r_e$ = event radius

**Path Calculations**

- Calculate path-averaged moho velocity: $V_m = \sum_{i=1}^{N} V_{m_i}$, where N is number of nodes along great circle path between source and receiver in model
- Calculate path-averaged mantle gradient: $g_m = \sum_{i=1}^{N} g_{m_i}$
- Calculate path-averaged $c_m = \left(\frac{g_m}{V_m}\right) + 1/r_e$, used for calculating turning-point depth
- Calculate whole model average (used for gradient term) $c = \left(\frac{g_m}{V_o}\right) + 1/r_e$
- Calculate average moho velocity at event depth over whole model, using c above
  - $V_{oz} = V_o(1 + cz_m)$
- Initial ray parameter from path-averaged moho velocity (head wave): $p = \frac{1}{V_m}$
- Calculate $x_s$ = distance from station to moho pierce point along moho (similar to equations (2) and (3) in travel-time calculation (use Snell's law and initial ray parameter)

**Mantle Distance**

- d = distance along moho from station moho pierce point to event location if event translated to moho surface:
  - $d = d_m - x_s$
  - Distance also related to FULL path distance that would go through event at depth
    - ♣ $d = (x_m \pm r x_z)/2$
      - $x_m$ = distance along moho from station pierce point to FULL ray intersection with moho on other end of ray
        $x_m = (2/c_m)\sqrt{(1 + c_m h)^2 - 1}$
      - $x_z$ = distance along event depth radius from event to FULL ray intersection with event depth radius on other end of ray
        $x_z = (2/c_m)\sqrt{(1 + c_m(h - z_m))^2 - 1}$
      - ± indicates upgoing (-) or downgoing (+) ray from event
- Minimize difference with actual distance "d" by varying $H_m$ values (turning point depth below moho) for upgoing and downgoing estimated FULL path (from receiver going through event depth location up to surface). Minimize log10 of squared difference using Brent routine (numerical recipes)
  - $\log\left(\left(\sqrt{(1 + c_m h)^2 - 1} - r\sqrt{(1 + c_m(h - z_m))^2 - 1} - dc_m\right)^2\right)$ (upgoing)

- $\log\left(\left(\sqrt{(1+c_m h)^2 - 1} + r\sqrt{(1+c_m(h-z_m))^2 - 1} - dc_m\right)^2\right)$ (downgoing)

- Determine if the upgoing or downgoing ray based on misfit. Should fit one or the other within machine error.
- Recalculate the $x_m$ and $x_z$ from the resulting $H_m$ value and $z_m$
  - if FINAL iteration, stop here and continue to **Travel Times** below
- Get velocity at turning point depth below moho h. Use path-averaged moho velocity and path-averaged gradient
  - $v_t = V_m + g_m h$
  - New ray parameter, $p = \frac{1}{v_t}$
- Recalculate station crustal travel time term and $x_s$ using new ray paramter
- Go through this process for 1 more iteration to get a FINAL $x_m$ and $x_s$.

**Travel Times**

- Calculate Zhao travel times for complete FULL ray between mantle pierce points (use model c instead of path-averaged c in gradient term). First calculate a time estimate for the part of the FULL ray path that is beyond the source-receiver distance. Use an approximate velocity for the path that is between the source and receiver.
  - $V_{eff} = d/t_{moho}$, (can do this simply because is removed from final time calculation in the end)
  - $d_e = x_m - d$, where $d_e$ is the distance along moho of the path beyond the source-receiver distance
  - $t_e = d_e / V_{eff}$
  - $t_{mz} = t_{moho} + t_e - \frac{c^2 x_m^3}{24 V_o}$,
    where $t_{moho}$ = travel time along Moho between pierce points (or Moho point above mantle event depth)
- Calculate Zhao travel time for segment of complete ray between pierce points at event depth (is same as $t_{mz}$ if $z_m$=0):
  - $t_{zz} = t_{mevent} - \frac{c^2 x_z^3}{24 V_{oz}}$,
    where $t_{mevent}$ = travel time along constant depth $z_m$ below Moho.
    If *isign* = -1 (upgoing ray, no part of $x_z$ is within source-receiver distance, can simplify because is removed from time later),
    $t_{mevent} = \frac{x_z}{V_{eff}(1 + c_m z_m)}$
- Obtain Zhao travel time (gradient term) for ray segment between event and moho pierce point below station (what we will use):
  - $t_{zhao} = \left(t_{mz} + float(isign) t_{zz}\right)/2$,
    - ♣ where *isign* is ±1 depending on whether upgoing (-) or downgoing (+)

- Final travel time for event below the moho is:

$t_{total} = \alpha + t_{zhao}$, where $\alpha$ is crustal time below station.

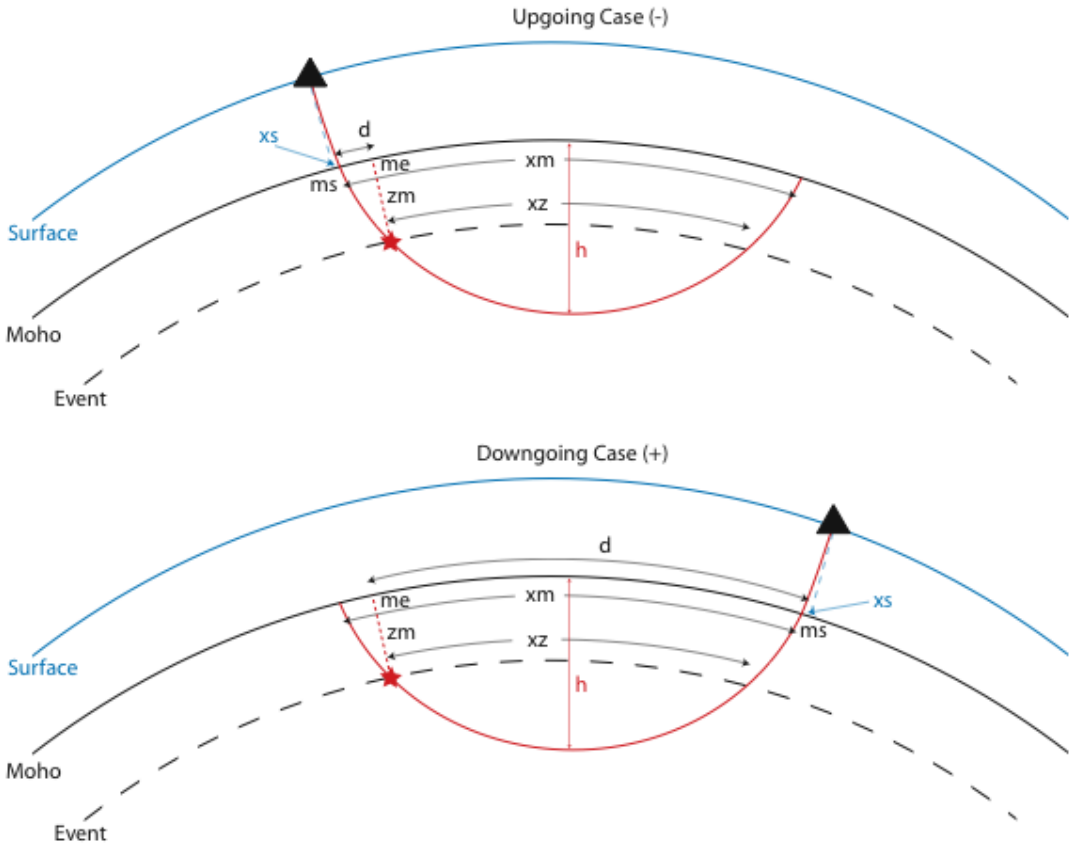


Figure 3 – Illustration of algorithm for computing Pn/Sn travel times through the SLBM model for events in the mantle.

# Phases Pg/Lg

The calculation of travel times for Pg/Lg phases is implemented in file GreatCircle_Xg.cc. Traveltimes are computed using two different methods and the smaller of the two travel times is retained. The first method considers the ray to travel from the source to the top of the middle crust, along the top of the middle crust to a point below the receiver, and from there up to the receiver. The second method assumes a 1D velocity structure equal to that beneath the receiver and uses the TauP method to compute the travel time. Each method is described below, followed by a detailed description of the how the methods are implemented in SLBM.

## *Headwave method*

This method computes travel time for a ray that propagates down (or up) to the middle crust, horizontally at the top of the middle crust, then up to the station. The travel time is given by

$$TT = \sum_{i=1}^{N} \frac{\delta_i r_i}{v_i} + \alpha + \beta$$

where $\delta_i$, $r_i$ and $v_i$ are the angular distance, radius and velocity in each of the $N$ segments comprising the entire great circle path from the source to the receiver (not just the portion between the pierce points at the top of the middle crust). $\alpha$ and $\beta$ are the vertical components of travel time at the source and receiver. $\alpha$ is defined to be

$$\alpha = \sum_{j=1}^{M} \frac{h_j}{v_j}\left(1 - p^2 v_j^2\right)^{1/2}$$

where $v$ and $h$ are the velocity and thickness of each crustal layer between the source and the top of layer 6. When the source is above layer 6, $p$ is the ray parameter defined by the $mean(1/v_{layer6})$ (weighted by the path length of each segment). When the source is in or below layer 6 it is assumed that the ray leaves the source horizontally and intersects the top of layer 6 due simply to earth sphericity. Simple geometry leads to an equivalent ray taking off at ~86° from vertical. Therefore, in this case the ray parameter becomes (only for $\alpha$) the $\sin(86°)/V_{source}$, or $\sim.997/ V_{source}$. Also, when the source is in the lower crust, the medium between the source and the top of layer 6 is assumed to be comprised of a single material with the velocity of the middle crust (the velocity of the lower crust is ignored). If this last assumption was not made, abrupt discontinuities in calculated travel times would occur at the interface between the middle and lower crust.

$\beta$ is defined as

$$\beta = \sum_{k=1}^{M} \frac{h_k}{v_k}\left(1 - p^2 v_k^2\right)^{1/2}$$

where $v$ and $h$ are the velocity and thickness of each crustal layer from the receiver to the top of layer 6. In this case, $p$ is always the $mean(1/v_{layer6})$ (weighted by the path length of each segment).

## *TauP method*

At local distances the Pg and Lg (Sg) travel-times can be calculated as first arriving phases. Further, the full 3-dimensional propagation should be considered when calculating Pg and Lg travel-times. Because the local calculation is applicable in the 0° ~2.5° distance range (depending on local structure) and the SLBM model is 1° resolution, the velocity stack under the station is likely to be a good approximation to the 3-dimensional crust in the SLBM model. Using the 1-dimensional local model, travel-time will be computed using methods similar to those presented in Lay and Wallace (1995) for a spherical 1D Earth. In this method the travel time function is evaluated for stationary points (w.r.t. $p$).

$$T = p\Delta + \tau(p)$$

$$\tau(p) = \int_{r_{bottom}}^{r_{event}} \frac{\sqrt{(ur)^2 - p^2}\,dr}{r} + \int_{r_{bottom}}^{r_{station}} \frac{\sqrt{(ur)^2 - p^2}\,dr}{r}$$

$$\Delta = p\int_{r_{bottom}}^{r_{event}} \frac{dr}{r\sqrt{(ur)^2 - p^2}} + p\int_{r_{bottom}}^{r_{station}} \frac{dr}{r\sqrt{(ur)^2 - p^2}}$$

where $\Delta$ is the station event distance, $T$ is travel time, $r$ is the radial distance from the center of the earth, $r_{bottom}$ is the ray bottoming depth, and $r_{event}$ and $r_{station}$ are the radius of the event and station, respectively, $u$ is model slowness (1/*velocity*), and $p$ is ray parameter.

The $\tau$ function is called out because it is common to seismology and has an analog in the flat Earth approach defined by Buland and Chapman (1983). It should be noted that this is similar to the taup approach for calculating travel times except that distance ($\Delta$) is known and $p$ must be zeroed-in upon for each possible turning layer (ray path) to arrive at a set of travel times from which the smallest value is chosen. Although the constant and linear velocity forms of the distance equation, given above, have analytic solutions they must be called many times to zero-in on the ray parameter with an acceptable predefined tolerance. This implies that the computational performance of this method should be tested for viability before final acceptance of the approach is granted.

## Implementation in SLBM

In this section, a detailed description of how travel times are computed for phases Pg and Lg is provided. First, the crustal stack at the receiver is retrieved. The water layer, the Pn/Sn middle crustal layer, and all zero thickness layers are removed. The top of the uppermost remaining layer is extended to a radius of 6471 km, which is approximately 100 km above the nominal radius of the earth. This is done to ensure that all sources and receivers will be located within the radial range of the model. The lowest layer in the model is the mantle which is characterized by a linear velocity gradient. The mantle layer extends from the Moho down to a radius of zero (the center of the Earth), or down to a radius where the velocity is 0.01 km/sec, whichever is greater. This velocity stack is used to compute TauP travel times.

When a request for a Pg or Lg travel time is received, SLBM first checks to ensure that both the source and receiver are located above the Moho as determined at each location independently, and throws an exception if that is not the case. If both are located in the crust, then SLBM attempts to compute the travel time using the TauP method. This will fail for any of several reasons. The first reason it might fail is that there may exist low velocity zones in the sedimentary layers in the crust which result in shadow zones. In the SLBM model, it is true that the velocities in the upper, middle and lower crust increase monotonically with depth, but it is not uncommon for the sedimentary layers that overlie the upper crust to have decreasing velocity with depth. A second way that TauP can fail is if the ray turns below the bottom of the middle

crust. While it is possible to calculate a TauP travel time for rays that bottom deeper than the bottom of the middle crust, it is not useful to do so in this application.

After the TauP travel time has been calculated, travel times are computed using the Headwave Method described above. There is one complication that can arise. If the velocity of the upper crust is greater than the velocity of the middle crust minus .1 km/sec, then reduce all velocities above the middle crust by a fudge factor such that the velocity of the upper crust is equal to the velocity of the middle crust minus 0.1 km/sec.

If either of the computed travel times is invalid, the valid travel time is retained. If both travel times are valid, the smaller of the two is retained. If neither is valid, an exception is thrown.

## Derivatives of Travel Time With Respect to Source Location

Since SLBM is intended for use in locating seismic events, and many seismic event location codes require the derivatives of travel time with respect to source position, SLBM provides methods to compute these derivatives. Specifically, SLBM has methods to return the following quantities: horizontal slowness, derivatives of travel time with respect to source latitude, longitude and depth. The numerical algorithms used to compute these quantities are described herein.

Figures 4 and 5 illustrate a vertical cross section and a map view of the great circle path connecting the source and receiver, respectively. While typical source-receiver distances range from about 2 to 15 degrees, separation between the auxiliary points and the source, $\delta x$ and $\delta z$, are 0.57° and 0.1 km, respectively. Expressions for the various derivatives are as follows:

Horizontal slowness
$$\frac{\partial T}{\partial x} = \frac{T_{x+} - T_s}{\delta x} \tag{1}$$

Travel time with respect to depth
$$\frac{\partial T}{\partial z} = \frac{T_z - T_s}{\delta z} \tag{2}$$

Travel time with respect to latitude
$$\frac{\partial T}{\partial \varphi} = \frac{T_\varphi - T_s}{\delta x} \tag{3}$$

Travel time with respect to longitude
$$\frac{\partial T}{\partial \theta} = \frac{T_\theta - T_s}{\delta x} \tag{4}$$

where $T$ is travel time, $T_i$ is the travel time evaluated at position $p_i$, $\varphi$ is latitude and $\theta$ is longitude.


Figure 4 – A vertical cross section containing the great circle path connecting source and receiver. Diamond shapes show the positions of auxiliary locations in the plane of the great circle that are close to the source and used to compute numerical derivatives.

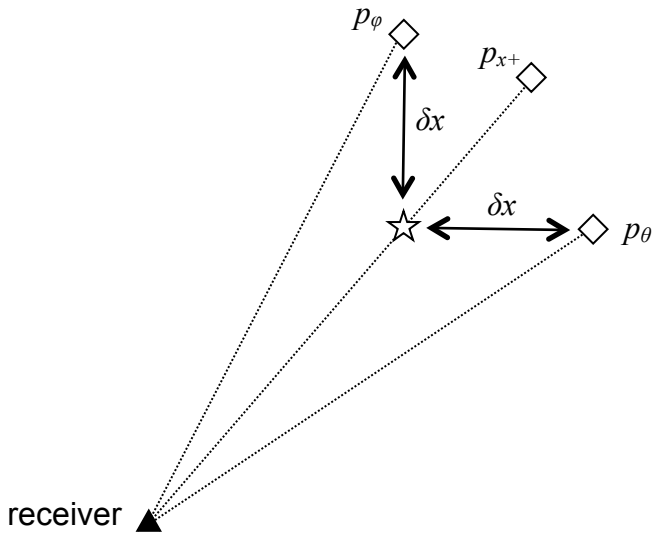


Figure 5 – Map view of source, receiver and auxiliary points showing locations of points used to compute derivatives of travel time with respect to latitude and longitude and depth.

## Software Design

The SLBM software suite consists of a core library, libslbm, written in C++, with interfaces written in C++, C, FORTRAN and Java. Extensive documentation of each class in libslbm and for each of the 4 interfaces, is supplied in html format in the various documentation directories under each interface's main directory. It is recommended that software developers who intend to write code that uses libslbm consult that documentation. This section will provide a general description of the architecture of the classes that comprise the core SLBM library.

Figure 6 is a collaboration diagram for the main classes that make up libslbm. At the bottom of the figure is the SlbmInterface which is the C++ interface to the library. C++ applications and all of the other interfaces access libslbm functionality through this interface.

SlbmInterface maintains instances of four main classes within libslbm. The first of these is a single Grid object, which manages the velocity model through which travel times are calculated. Grid has methods to read velocity models from a file, methods to extract velocity profiles from the velocity model, methods to modify the velocity and gradient information in memory, and methods to write the velocity model back out to a new file.. SlbmInterface objects also have access to model uncertainty information through an Uncertainty object.

At this point, it is useful to describe the concept of a Profile. Consider a borehole drilled vertically downward through the velocity model at some point on the surface to the model. That borehole would intersect all of the layers of the model, each of which would be characterized by the depth of the top of the layer, the P and S wave velocities of the layer, and, at the top of the mantle, the P and S velocity gradients of the mantle. This ensemble of information is described as a Profile.

libslbm includes 4 different varieties of Profile. The simplest of these is the GridProfile which includes all the information described in the last paragraph. The locations of GridProfile objects correspond with the locations of the grid nodes in the velocity model. In fact, a Grid object stores the velocity information it manages in a 2 dimensional array of GridProfile objects.

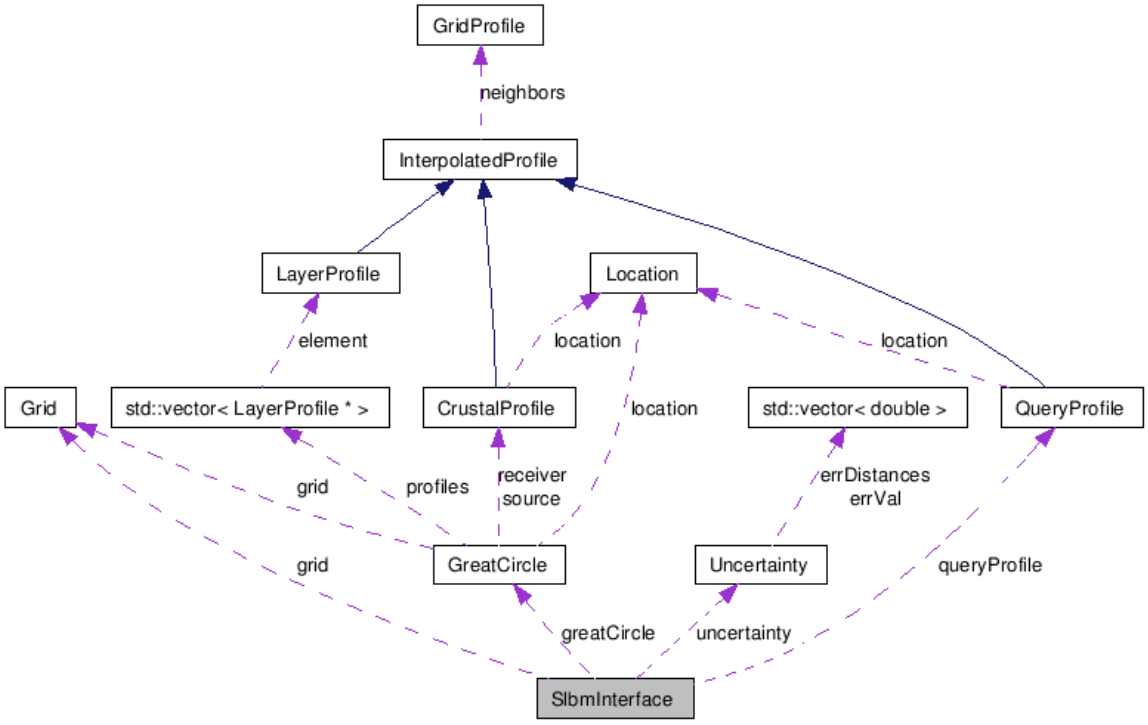


Figure 6 – Collaboration diagram for libslbm. Dark blue arrows represent a public inheritance relation between two classes. Purple dashed arrows represent classes that are contained or used by another class. The purple arrows are labeled with the variable(s) through which the pointed class is accessible.

The other three types of Profiles are all located at positions in latitude, longitude space that do not necessarily coincide with the location of a grid node. Since they all involve interpolating information from nearby grid nodes, they share a base class, InterpolatedProfile, that provides all the functionality necessary to interpolate radius, velocity and velocity gradient information from the velocity grid. The three profile classes that derive off of InterpolatedProfile are QueryProfile, CrustalProfile and LayerProfile. Each is designed for a different purpose and hence stores different amounts of interpolated information.

A QueryProfile object stores the same information as a GridProfile object, which includes everything that is known about a Profile through the velocity model, but contains information that is interpolated from a number of neighboring GridProfiles. SlbmInterface uses a QueryProfile object to respond to all the model query calls provided in the SlbmInterface.

A CrustalProfile object stores only the information needed to calculate travel times through a crustal stack at a specified position, for a single phase. A CrustalProfile object only supplies travel time information for a single phase, so there is no need to compute or store both P and S wave velocities. CrustalProfile objects only compute and store one or the other. Furthermore, since velocity gradient information is not required to compute travel times for crustal legs, interpolated velocity gradient information is neither computed nor stored.

The last type of Profile included in libslbm is the LayerProfile class. These are used to represent the radius, P or S velocity, and possibly P or S velocity gradient information, for a single layer at a single point within the model. These objects are used as the nodes along the head wave interfaces.

Another very important class is the GreatCircle class. The SlbmInterface owns one of these, which it uses to compute the travel time between a source and receiver. A GreatCircle object owns two CrustalProfile objects, one for the source and one for the receiver. It also owns an array of LayerProfile objects which are evenly spaced along the great circle path between the source and receiver, and are positioned at the radius of the head wave interface. There are two flavors of GreatCircle objects, GreatCircle_Xn and GreatCircle_Xg, both of which inherit from the GreatCircle base class. GreatCircle_Xn has a computeTravelTime() method to compute travel times for Pn/Sn phases while GreatCircle_Xg's computeTravelTime() method calculates travel times for Pg/Lg phases.

Location is a utility class used to manipulate the positions of points on or within the Earth where the Earth is represented by a GRS80 ellipsoid. Many classes within libslbm use instances of Location. A complete description of the mathematical basis of the Location class is provided in a separate document included in the SLBM documentation called geovectors.pdf.