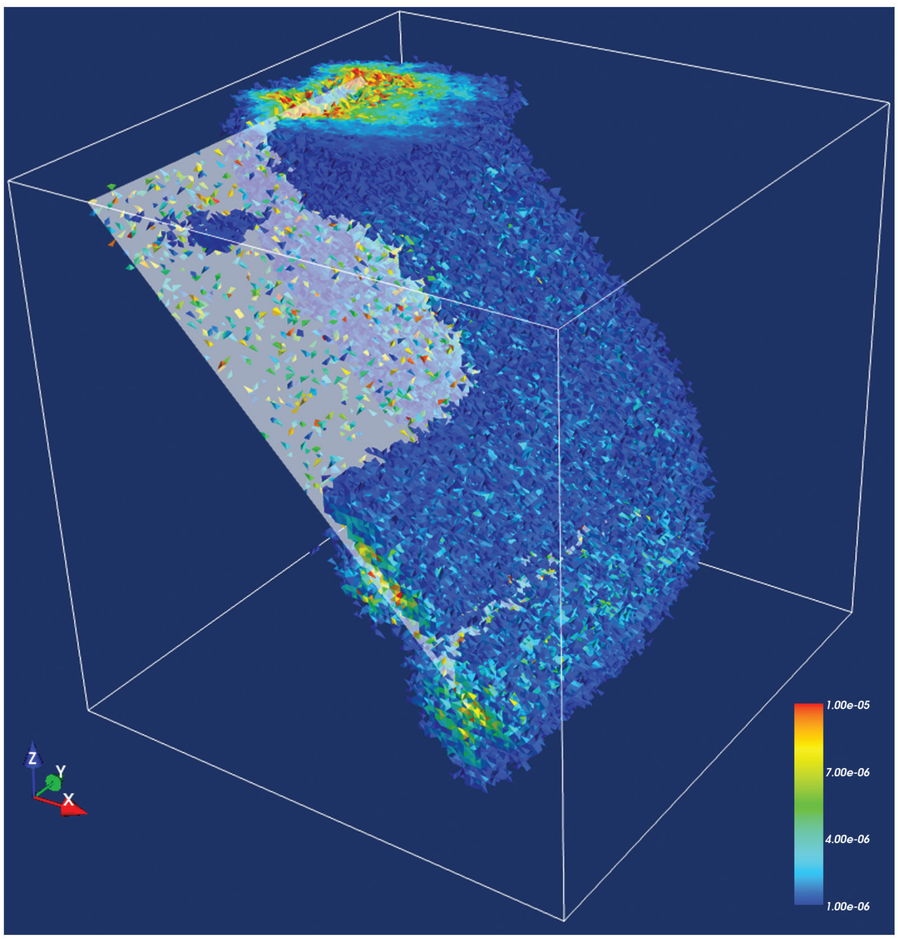
COMPUTATIONAL INFRASTRUCTURE FOR GEODYNAMICS (CIG)

Cigma

User Manual
Version 0.9



Luis Armendariz
Susan Kientz

www.geodynamics.org

Cigma

Version 0.9

July 20, 2007

Contents

List of Figures

Chapter 1

Introduction

1.1 About Cigma

The CIG Model Analyzer (Cigma) consists of a suite of tools intended to facilitate the comparison of numerical models. CIG has developed Cigma in response to demand from the short-term tectonics community for a simple tool that can perform rigorous error analysis on their FEM codes. The long-term goal for Cigma, however, is for it to be used for nearly all geodynamics modeling codes.

In general, Cigma is intended for three types of tasks, namely (1) error analysis, (2) benchmarking, and (3) code verification.

There are two ways in which Cigma can help you with error analysis. It can take a random sampling of points inside a domain of interest and analyze the pointwise differences between physical fields, or otherwise perform an integration of the errors over a discretized version of the domain. This comparison can take place even when the meshes are not compatible.

In benchmarking, Cigma can help the geodynamics community agree on a standard solution to specific problems by facilitating the process of comparing different numerical codes against each other.

Lastly, as an automated tool, Cigma can help developers in creating regression tests to ensure that software changes do not affect the consistency of the results.

At its core, Cigma draws from a variety of libraries, particularly the Tetrahedral Mesh Comparator (TMC) (`www.sci.utah.edu/~bavoil/research/tetsimp/tmc/`) from the University of Utah, which itself draws from the GTB Graphics Toolbox library (`sf.net/projects/gtb`). Cigma extends and generalizes the functionality therein to handle other types of elements as well as adding the ability to compare vector fields.

1.2 Citation

Computational Infrastructure for Geodynamics (CIG) is making this source code available to you in the hope that the software will enhance your research in geophysics. This is a brand-new code and at present no papers are published or at press for use as citations other than this manual, which is cited as follows:

> Armendariz, L., and S. Kientz. *Cigma User Manual.* Pasadena, CA: Computational Infrastructure of Geodynamics, 2007. URL: geodynamics.org/cig/software/cs/cigma/cigma.pdf

CIG requests that in your oral presentations and in your papers that you indicate your use of this code and acknowledge the author of the code and CIG (`geodynamics.org`).

1.3 Support

Cigma development is supported by a grant from the National Science Foundation to CIG, managed by the California Institute of Technology.

Chapter 2

Installation and Getting Help

2.1 Getting Help

For help, send an e-mail to the CIG Computational Science Mailing List (`cig-cs@geodynamics.org`). You can subscribe to the `cig-cs` mailing list and view archived discussions at the CIG Mail Lists web page (`geodynamics.org/cig/lists`). If you encounter any bugs or have problems installing Cigma, please submit a report to the CIG Bug Tracker (`geodynamics.org/bugs`).

2.2 Installation from Source

To use Cigma, download the source package (in the form of a compressed tar file) from the CIG Cigma web page (`geodynamics.org/cig/software/packages/cs/cigma`). This step will require the GNU C and C++ compilers. After unpacking the source and installing the dependencies, issue the following commands

```
$ make
$ sudo make install
```

2.2.1 HDF5

HDF5 is available for download from The HDF Group (`hdfgroup.org/HDF5`). Binaries can be obtained at hdfgroup.org/HDF5/release/obtain5.html (`hdfgroup.org/HDF5/release/obtain5.html`). To install from source, download the latest stable version of this library (currently 1.6.5) and issue the following commands

```
$ tar xvfz hdf5-1.6.5
$ cd hdf5-1.6.5
$ ./configure
$ make
$ sudo make install
```

2.2.2 PyTables

PyTables is a Python extension module that builds on top of the HDF5 library. It provides a convenient scripting interface to manipulate HDF5 files. It is available from PyTables (`www.pytables.org`).

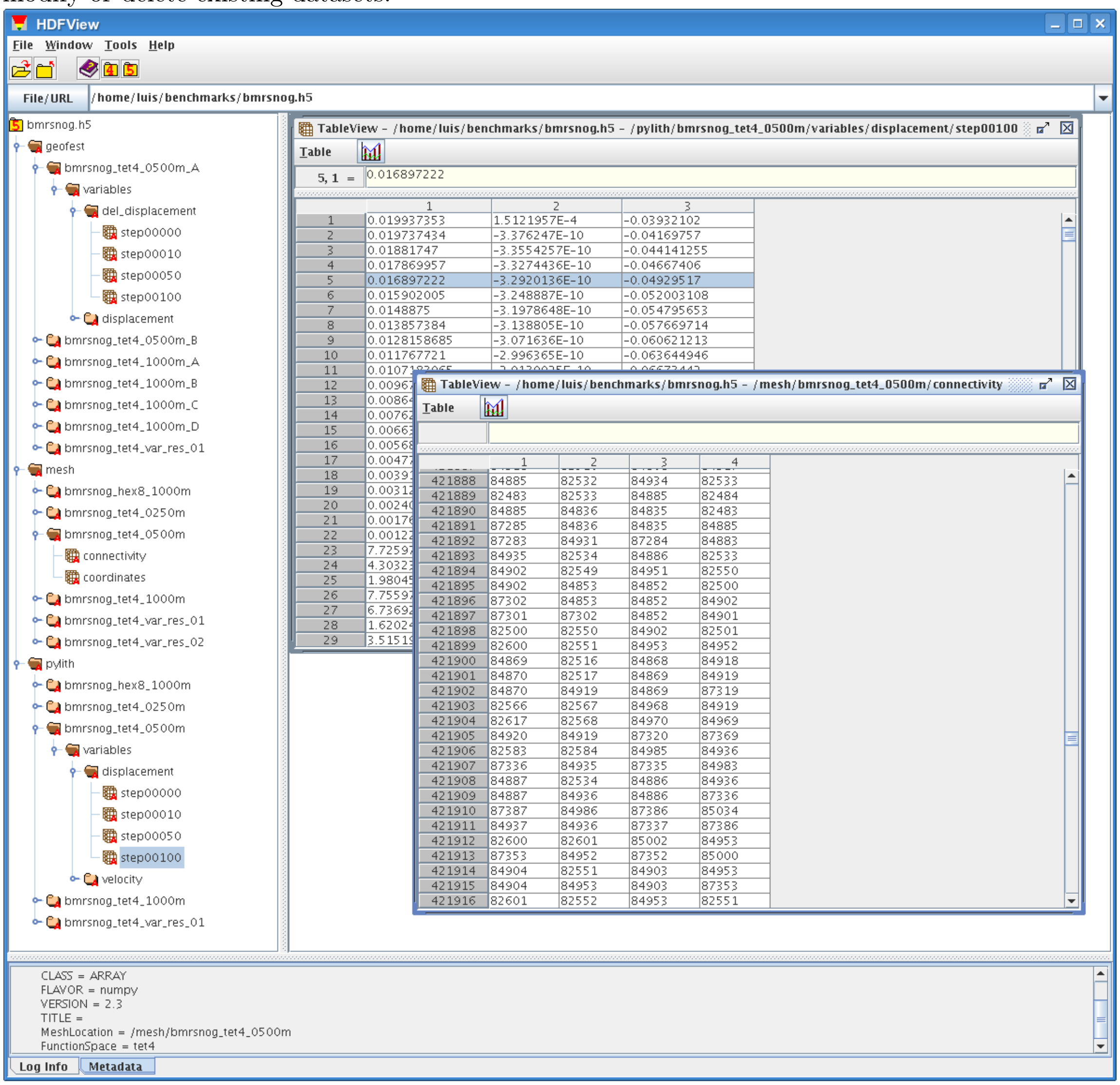
To install this extension from source, download the latest stable version (currently 2.0) and issue the following commands

```
$ tar xvfz pytables-2.0
$ cd pytables-2.0
$ sudo python setup.py install
```

2.2.3 HDFView (optional)

NCSA HDFView is a graphical user interface tool for accessing data in your HDF5 files. You can use it for viewing the internal file hierarchy in a tree structure, adding new datasets, and modifying or deleting existing datasets. You can download it from the HDFView home page (`hdf.ncsa.uiuc.edu/hdf-java-html/hdfview`).

Figure 2.1: With HFDFView, you can view the internal file hierarchy in a tree structure, add new datasets, and modify or delete existing datasets.



Chapter 3

Error Analysis

3.1 Introduction

When studying differential equations that represent physical systems we often obtain solutions by using a variety of techniques, most of which are numerical in nature. In carefully designed problems one may of course obtain solutions in explicit analytical form, but for the most part we will deal with approximate solutions given by the Finite Element Method. Without any closed-form solution available, the quality of an approximation can only be assessed relative to other approximations.

Thus, an important part of error analysis lies in the ability to calculate the distance between two putative solutions. Estimating the error between two arbitrarily represented fields is computationally challenging due to the variety of representations that are possible. For example, each field may use its own discretization of the original domain, and may use a different set of shape functions.

3.2 Distance Measures

The simplest possible quantitative measure of the difference between two distinct fields you can make consists of taking the pointwise difference of both fields at a common set of points. While no finite sample of points can perfectly represent a continuum of values, valuable information can be inferred from a statistics analysis of the resulting residual values.

Another useful distance measure can be obtained by using the L_2 norm, defined by the following integral

$$\varepsilon = ||u - v||_{L_2} = \sqrt{\int_\Omega ||u(\vec{x}) - v(\vec{x})||^2 d\vec{x}}$$

This gives us a single global estimate ε representing the distance between the two fields $u(\vec{x})$ and $v(\vec{x})$. Alternatively, you may think of this as the size, or norm, of the residual field $\rho(\vec{x}) = u(\vec{x}) - v(\vec{x})$. If we discretize the domain Ω into finite elements Ω_e, the above integral can be broken up into a sum over local contributions on each element. For efficiency, each contribution can be integrated over a reference element $\hat{\Omega}_e$ defined on a standard coordinate system.

$$\begin{aligned}
\varepsilon^2 &= \sum_{e=1}^{\text{nel}} \varepsilon_e^2 \\
&= \sum_{e=1}^{\text{nel}} \int_{\Omega_e} ||u(\vec{x}) - v(\vec{x})||^2 d\vec{x} \\
&= \sum_{e=1}^{\text{nel}} \int_{\hat{\Omega}_e} ||u(\vec{\xi}) - v(\vec{\xi})||^2 J(\vec{\xi}) d\vec{\xi}
\end{aligned}$$

In general, we won't be able to integrate each local contribution exactly since the two fields u and v may have a representation that's incompatible with the local domain Ω_e. However, we can approximate each ε_e^2 by applying an appropriate quadrature rule with a tolerable truncation error [2].

Assuming we apply the same quadrature rule, with weights w_q and integration points $\vec{\xi}_q$, on every element,

$$\begin{aligned} \varepsilon_e^2 &= \sum_{q=1}^{\mathrm{nq}} w_q ||\hat{\rho}(\vec{x}_q)||^2 \\ &= \sum_{q=1}^{\mathrm{nq}} w_q ||u(\vec{\xi}_q) - v(\vec{\xi}_q)||^2 J(\vec{\xi}_q) \end{aligned}$$

thus we arrive at the final form

$$\varepsilon = \sqrt{\sum_{e=1}^{\mathrm{nel}} \sum_{q=1}^{\mathrm{nq}} w_q ||u(\vec{\xi}_q) - v(\vec{\xi}_q)||^2 J(\vec{\xi}_q)}$$

In calculating the norm of the residual field ρ, Cigma will output each of the local contributions ε_e^2 which by definition are scalar-valued cell quantities over each of their corresponding elements Ω_e.

Chapter 4

Cigma Components

4.1 Mesh

In Cigma, we define a finite element mesh simply by the coordinates, (x_n, y_n, z_n) of its degrees of freedom, and the connectivity relations $\Omega_e = \{n_1, n_2, \ldots\}$ among them which define each individual element in the corresponding discretization.

4.2 Fields

A field is a function which assigns a physical quantity to every point in space. This quantity may correspond to a scalar, a vector, or even a tensor. For any given differential equation problem, a finite element approximation to an unknown field $\phi(\vec{x})$ as a weighed sum over a fixed set of localized shape functions $\phi_n(\vec{x})$.

$$\phi(\vec{x}) = \sum_{n=1}^{N} d_n \phi_n(\vec{x})$$

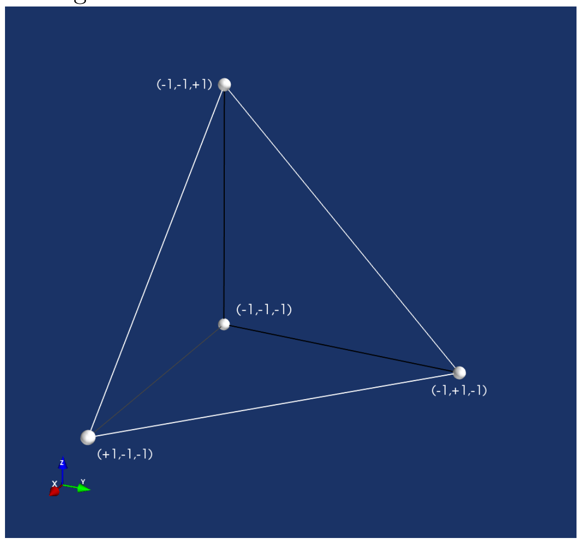
Naturally, once the weights d_n, or degrees of freedom as they are also called, are known to us, we can evaluate $\phi(\vec{x})$ at any point $\vec{x}$ we desire. Thus, in Cigma a field is represented simply by a list of degrees of freedom d_n, which may be a scalar, vector or tensor quantity, depending on the nature of ϕ.

4.3 Elements

This release of Cigma provides you with two built-in finite element spaces shown below. The location of each element is indexed into a spatial database in order to speed up the evaluation process.

Function Space tet4

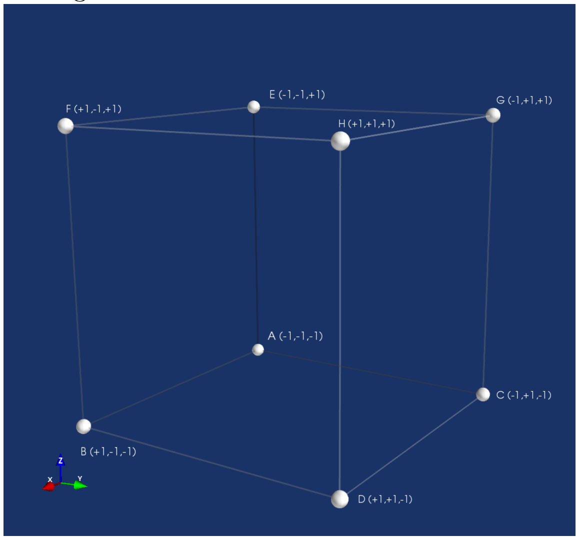
Figure 4.1: Reference tetrahedral element



$$
\begin{aligned}
TN_a &= \frac{1}{2}(-1 - x - y - z) \\
TN_b &= \frac{1}{2}(1 + x) \\
TN_c &= \frac{1}{2}(1 + y) \\
TN_d &= \frac{1}{2}(1 + z)
\end{aligned}
$$

Function Space hex8

Figure 4.2: Reference hexahedral element



$$
\begin{aligned}
HN_a &= \frac{1}{8}(1-x)(1-y)(1-z) \\
HN_b &= \frac{1}{8}(1+x)(1-y)(1-z) \\
HN_c &= \frac{1}{8}(1-x)(1+y)(1-z) \\
HN_d &= \frac{1}{8}(1+x)(1+y)(1-z) \\
HN_e &= \frac{1}{8}(1-x)(1-y)(1+z) \\
HN_f &= \frac{1}{8}(1+x)(1-y)(1+z) \\
HN_g &= \frac{1}{8}(1-x)(1+y)(1+z) \\
HN_h &= \frac{1}{8}(1+x)(1+y)(1+z)
\end{aligned}
$$

Chapter 5

Running Cigma

5.1 Comparing Two Fields

Comparing two arbitrary finite element fields can be accomplished with the `cigma-compare` command line utility, which takes a number of arguments in order to facilitate writing shell scripts. The comparisons always take place using the mesh of the first field. Note that the first and second arguments take the special form `filename:dataset`. Finally, the square of each of the local residual values are written to the specified VTK output file as cell-based scalars.

To use a quadrature rule for the field comparisons, you can specify arguments similar to the following

```
cigma-compare --output=squared-residuals.vtk   \
              --first=field1.h5:/field1/stepN  \
              --second=field2.h5:/field2/stepN \
              --rule=qr.h5:/path/to/rule
```

Alternatively, to perform a pointwise comparison at random sample points inside each element in the first mesh, you can use arguments similar to the following

```
cigma-compare --output=squared-residuals.vtk   \
              --first=field1.h5:/field1/stepN  \
              --second=field2.h5:/field2/stepN \
              --samples-per-element=1
```

5.2 Input Format

The Hierarchical Data Format (HDF) is a portable file format developed at the National Center for Supercomputing Applications (NCSA) (`hdf.ncsa.uiuc.edu/HDF5`). It is designed for storing, retrieving, analyzing, visualizing, and converting scientific data. The current and most popular version is HDF5, which stores multi-dimensional arrays together with ancillary data in a portable self-describing format. It uses a hierarchical structure that provides application programmers with a host of options for organizing how data is stored in HDF5 files.

HDF5 files are organized in a hierarchical structure, similar to a UNIX file system. Two types of primary objects, groups and datasets, are stored in this structure. A group contains instances of zero or more groups or datasets, while a dataset stores a multi-dimensional array of data elements. Both kinds of objects are accompanied by supporting metadata.

A dataset is physically stored in two parts: a header and a data array. The header contains miscellaneous metadata describing the dataset as well as information that is needed to interpret the array portion of the dataset. Essentially, it includes the name, datatype, dataspace, and storage layout of the dataset. The name is a text string identifying the dataset. The datatype describes the type of the data array elements. The dataspace defines the dimensionality of the dataset, i.e., the size and shape of the multi-dimensional array.

Using HDF5 datasets in Cigma allows us to avoid having to convert between too many distinct formats. Moreover, due to the amount of disk I/O, large finite element meshes can be handled more efficiently in binary format. A typical Cigma HDF5 file has the following structure:

```
model.h5
\__ model
    |__ mesh
    |   |__ coordinates    [nno x nsd]
    |   \__ connectivity   [nel x ndof]
    \__ variables
        \__ velocity
            \__ step00010  [nno x ndim]
```

You have a certain amount of flexibility in grouping your own data. Cigma will only require you to provide a small amount of metadata on your field and mesh datasets, described below:

MeshID a universally unique identifier (uuid) assigned to the mesh for easily distinguishing identical meshes.

MeshLocation points to the HDF5 group which contains the appropriate coordinates and connectivity datasets.

FunctionSpace string identifier to determine which shape functions to use for interpolating values inside the element.

5.2.1 Importing Data

In its source distribution, Cigma provides two examples for importing data from two different codes into its standard HDF5 format.

Mesh coordinates can be specified in the following format

```
nno nsd
1 x1 y1 z1
2 x2 y2 z2
3 x3 y3 z3
...
```

Mesh connectivity with

```
nel ndof
1 node_1 node_2 node_3 node_4 ...
2 node_1 node_2 node_3 node_4 ...
3 node_1 node_2 node_3 node_4 ...
...
```

A generic field with `ndim` components is specified by

```
nno ndim
1 f1 f2 f3 ..
2 f1 f2 f3 ..
...
```

Chapter 6

Visualization

As can be seen from the two images in Figure 6.1, simply visualizing two different solutions side by side does not give you enough insight into their actual differences. By using Cigma to calculate the residual field between them, you can get a better idea for how the local contributions to the global error are distributed both spatially and temporally.

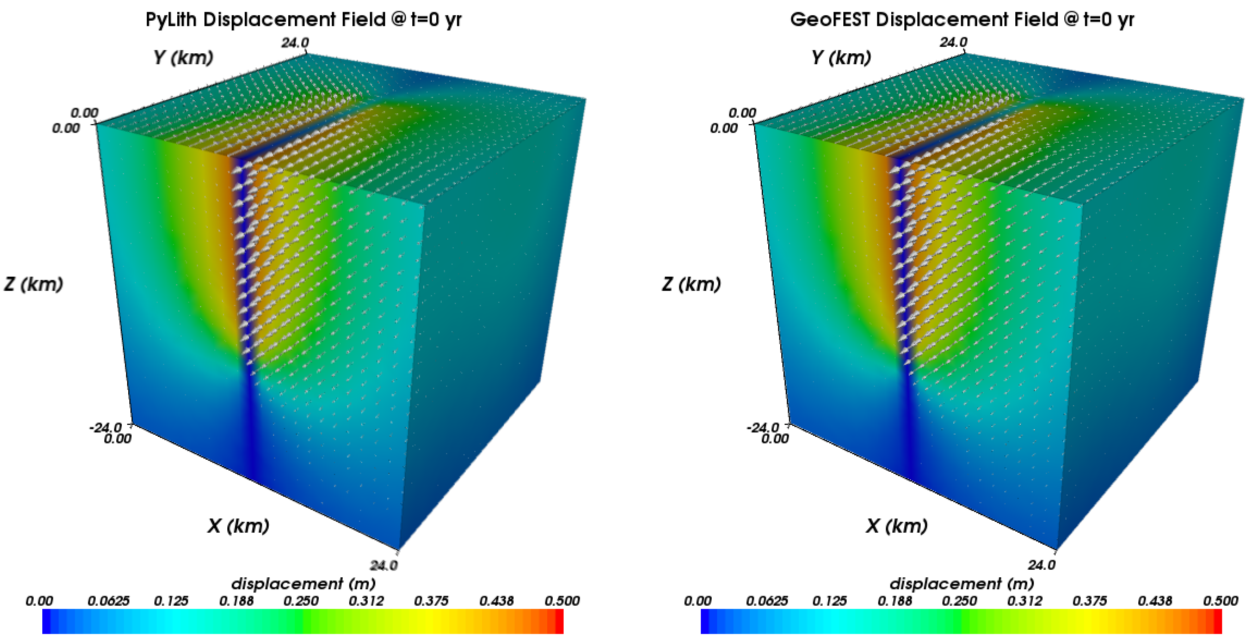


Figure 6.1: Solution of a viscoelastic problem with a fault using two finite element codes: PyLith solution (left) and GeoFEST solution (right).

6.1 Benchmark Cases

Here we compare the output from two codes, PyLith-0.8 and GeoFEST-4.5, on two benchmark cases (`geodynamics.org/cig/workinggroups/short/workarea/benchmarks`) defined by the CIG Short-Term Tectonics working group. They are both defined on cube domain (Figure 6.2) with sides having a length of 24 km, consisting of two layers of different material types. The top layer is nearly elastic while the bottom layer follows viscoelastic relaxation of stresses. Bottom and side displacements are set to the elastic analytic solution. A symmetric boundary condition is also imposed on the y=0 plane.

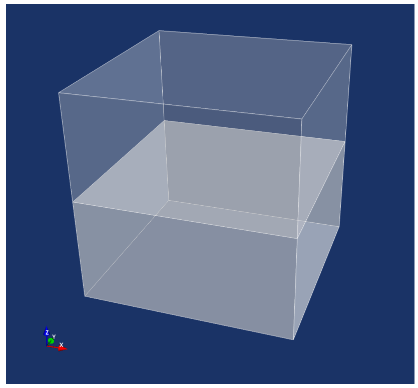


Figure 6.2: Cube domain of 24 km length consisting of two layers of different material types.

The first benchmark problem (left) consists of a vertical right-lateral strike-slip fault. The second benchmark problem consists of a 45-degree dipping reverse fault.

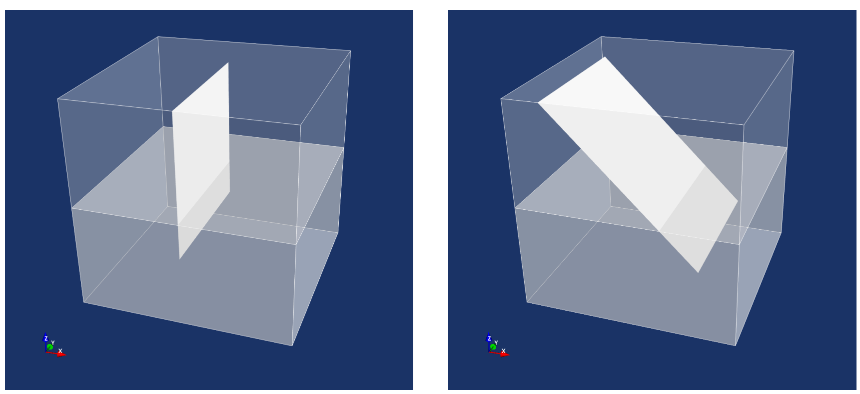


Figure 6.3: Two benchmark problems: (left) Benchmark problem consisting of a vertical right-lateral strike-slip fault; (right) Benchmark problem consisting of a 45-degree dipping reverse fault.

In both cases we solve for the displacement and vector fields at various time steps, namely 0, 1, 5, and 10 years. In the plots below, we show the distribution of the squared local residuals on each cell.

6.1.1 Strike-Slip with No Gravity

In this section, we show ten equally spaced isosurfaces of the displacement field residuals for the strike-slip benchmark (0 and 10 years shown). In Figure 6.1, we see that the differences are very localized at t=0 years. There is not much difference between time steps at t=1, 5 and 10 years, so we are only showing the last time step. Note that the maximum errors are localized to the interface between the two layers on the symmetric boundary.

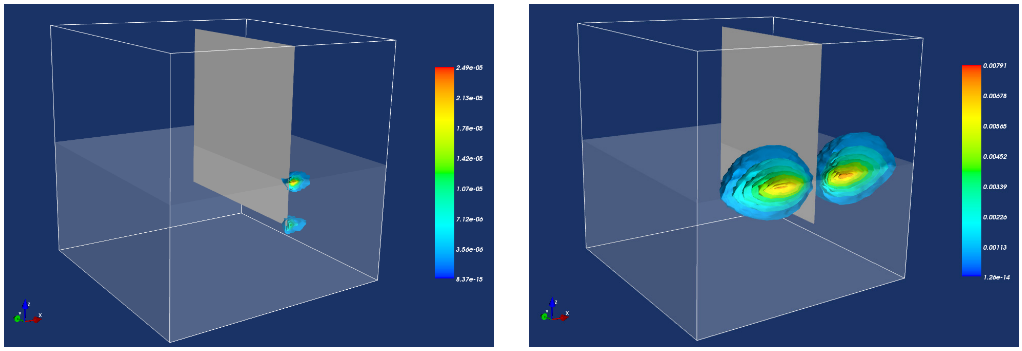


Figure 6.4: Strike-slip: PyLith-GeoFEST comparison of displacement residuals on a 500m resolution mesh (left: t=0 years, right: t=10 years).

Slightly different behavior can be observed in the velocity field residuals. Shown here are ten equally spaced isosurfaces at each time step, where each isosurface is displayed as a point distribution to reveal the inner structure. Note that after 10 years, most of the disagreement occurs inside the bottom viscoelastic layer, centered around the fault's interior sharp corner.

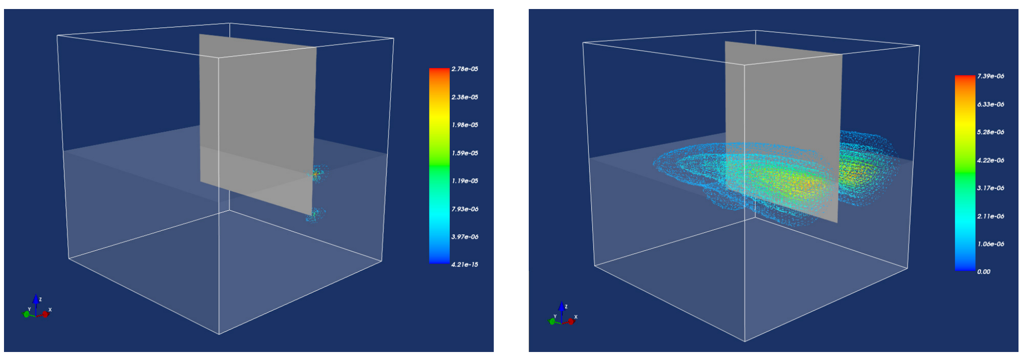


Figure 6.5: Strike-slip: PyLith-GeoFEST comparison of velocity residuals on a 500m resolution mesh (left: t=0 years, right: t=1 year).

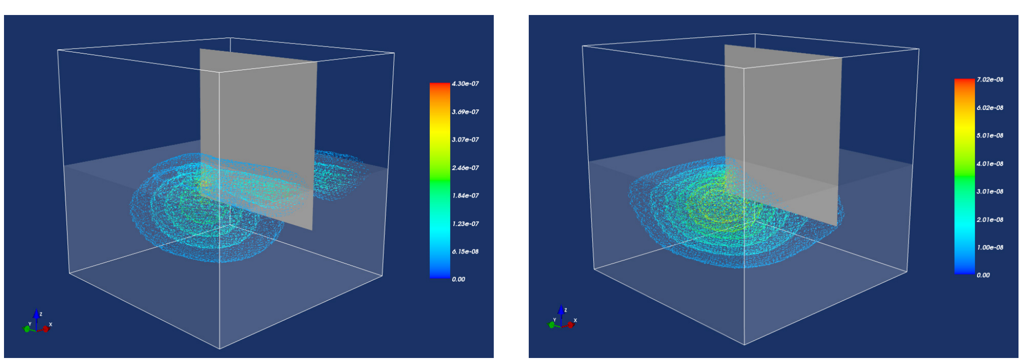


Figure 6.6: Strike-slip: PyLith-GeoFEST comparison of velocity residuals on a 500m resolution mesh (left: t=5 years, right: t=10 years).

Finally, here is a higher resolution comparison of the displacement residuals at t=0 years, sampled over a 250m resolution mesh. Displayed here are ten equally spaced isosurfaces, nine of which are very near the fault. In this case, the linear taper over the internal edges of the fault is clearly visible.

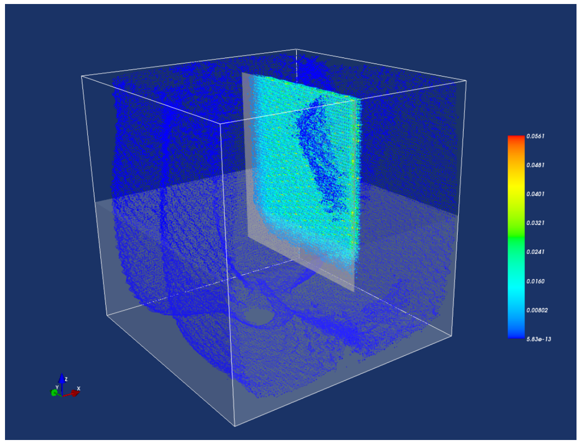


Figure 6.7: Strike-slip: PyLith-GeoFEST comparison of displacement residuals on a 250m resolution mesh (t=0 years).

6.1.2 Reverse-Slip with No Gravity

You can also visualize the distribution of errors by plotting the residual field values over the surface of each cell and applying a threshold filter which eliminates cells containing values outside a threshold interval. Here we show how velocity field residuals in the reverse-slip benchmark are distributed temporally by throwing away all squared residuals lower than $10^{-7}(\mathrm{m/s})^2$

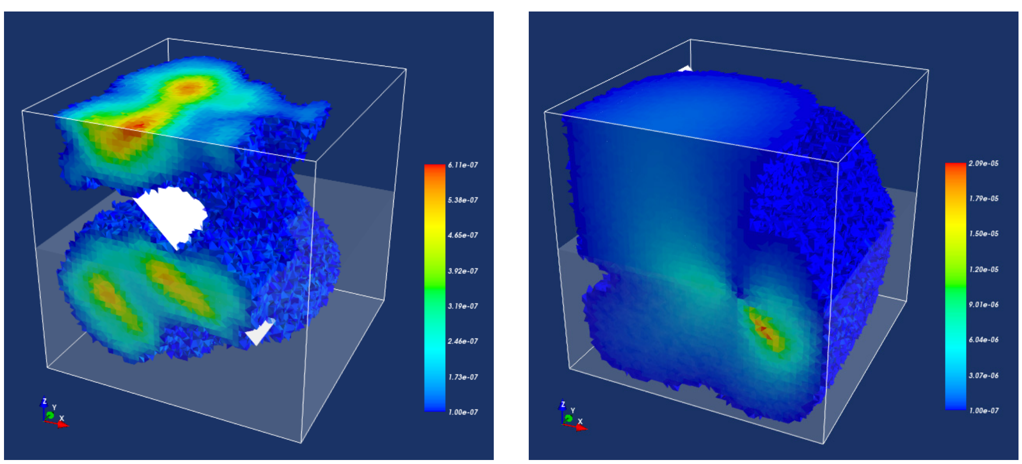


Figure 6.8: Reverse-slip: PyLith-GeoFEST comparison of velocity residuals on a 500m resolution mesh (left: t=0 years, right: t=1 year).

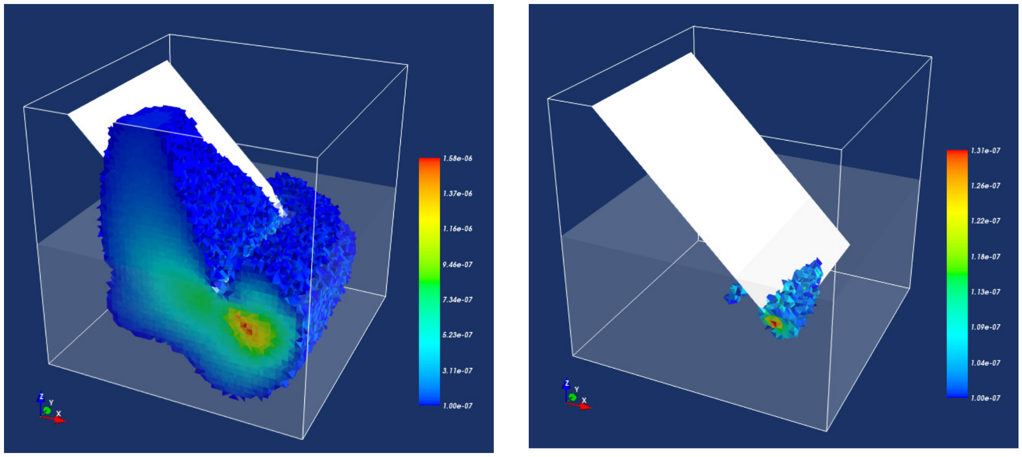


Figure 6.9: Reverse-slip: PyLith-GeoFEST comparison of velocity residuals on a 500m resolution mesh (left: t=5 years, right: t=10 years).

Finally, below we display ten equally spaced isosurfaces over the displacement field residuals of the reverse-slip benchmark. Note that after 1 year, most of the disagreement occurs in the bottom viscoelastic layer.

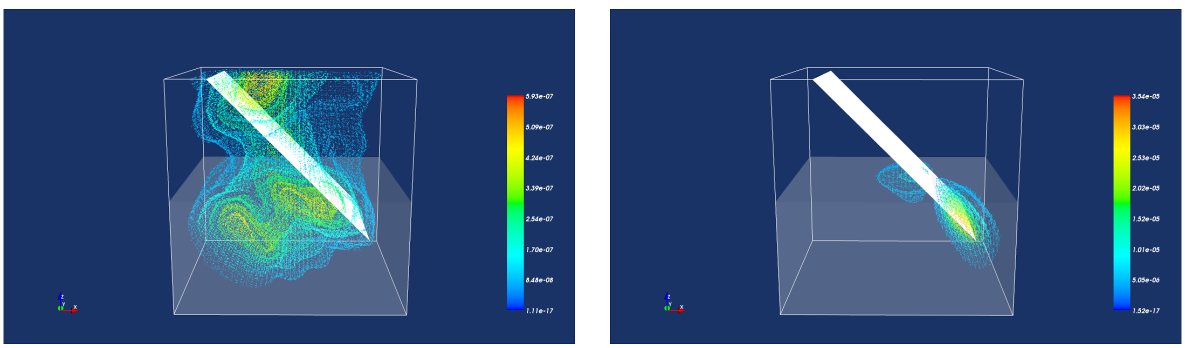


Figure 6.10: Reverse-slip: PyLith-GeoFEST comparison of displacement residuals on a 500m resolution mesh (left: t=0 years, right: t=1 year).

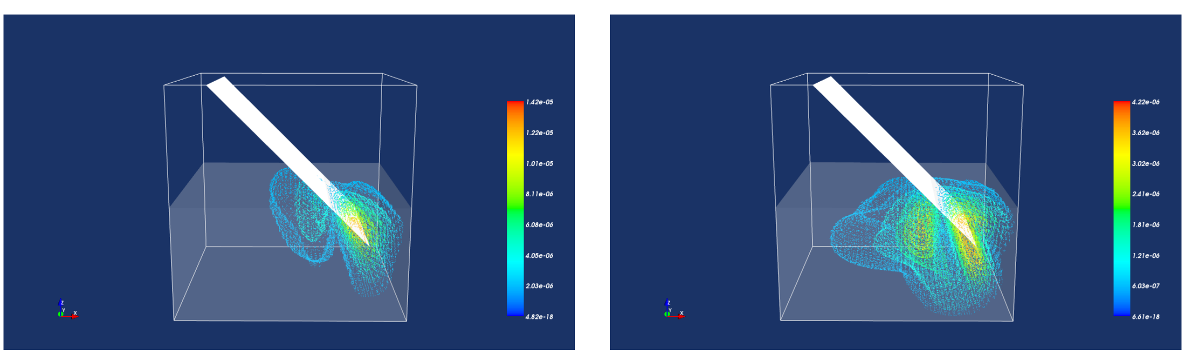
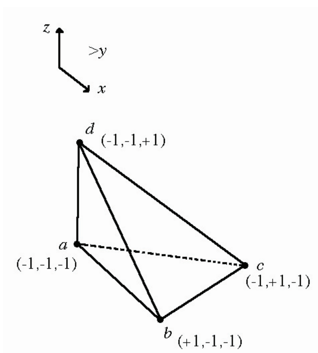


Figure 6.11: Reverse-slip: PyLith-GeoFEST comparison of displacement residuals on a 500m resolution mesh (left: t=5 years, right: t=10 years).

Appendix A

Shape Functions

A.1 Linear Tetrahedral Element (tet4)



$$
\begin{array}{cccccc}
 & & & & \phi & \\
(-1, & -1, & -1) & \mapsto & a \\
(+1, & -1, & -1) & \mapsto & b \\
(-1, & +1, & -1) & \mapsto & c \\
(-1, & -1, & +1) & \mapsto & d
\end{array}
$$

A field $\vec{\varphi} = \varphi_x\hat{x} + \varphi_y\hat{y} + \varphi_z\hat{z}$ defined over a linear tetrahedral element with vertices at $A(x_0, y_0, z_0)$, $B(x_1, y_1, z_1)$, $C(x_2, y_2, z_2)$, $D(x_3, y_3, z_3)$, has the following functional form inside the reference tetrahedral element shown in Figure 6.2.

$$
\begin{aligned}
\varphi_x(\vec{\xi}) &= \alpha_0 + \alpha_1\xi + \alpha_2\eta + \alpha_3\zeta \\
\varphi_y(\vec{\xi}) &= \beta_0 + \beta_1\xi + \beta_2\eta + \beta_3\zeta \\
\varphi_z(\vec{\xi}) &= \gamma_0 + \gamma_1\xi + \gamma_2\eta + \gamma_3\zeta
\end{aligned}
$$

In particular, the map from the reference coordinates into the regular coordinates vector $\vec{x}(\vec{\xi}) = x(\hat{\vec{\xi}})x + y(\vec{\xi})\hat{y} + y(\vec{\xi})\hat{z}$ looks like

$$
\begin{aligned}
x(\vec{\xi}) &= \alpha_0 + \alpha_1\xi + \alpha_2\eta + \alpha_3\zeta \\
y(\vec{\xi}) &= \beta_0 + \beta_1\xi + \beta_2\eta + \beta_3\zeta \\
z(\vec{\xi}) &= \gamma_0 + \gamma_1\xi + \gamma_2\eta + \gamma_3\zeta
\end{aligned}
$$

The following natural mappings uniquely determine the coefficients α_k, β_k, γ_k

$$
\begin{array}{lcl}
\vec{x}(-1,-1,-1) & \mapsto & (x_0, y_0, z_0) \\
\vec{x}(+1,-1,-1) & \mapsto & (x_1, y_1, z_1) \\
\vec{x}(-1,+1,-1) & \mapsto & (x_2, y_2, z_2) \\
\vec{x}(-1,-1,+1) & \mapsto & (x_3, y_3, z_3)
\end{array}
$$

Considering only the first component $x(\vec{\xi})$, we can obtain the matrix equation

$$\begin{array}{c} a \\ b \\ c \\ d \end{array} \begin{bmatrix} 1 & (-1) & (-1) & (-1) \\ 1 & (+1) & (-1) & (-1) \\ 1 & (-1) & (+1) & (-1) \\ 1 & (-1) & (-1) & (+1) \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Applying our desired map to the other two functions yields two other identical systems. Thus we may augment the system as follows,

$$\begin{bmatrix} 1 & -1 & -1 & -1 \\ 1 & +1 & -1 & -1 \\ 1 & -1 & +1 & -1 \\ 1 & -1 & -1 & +1 \end{bmatrix} \begin{bmatrix} \alpha_0 & \beta_0 & \gamma_0 \\ \alpha_1 & \beta_1 & \gamma_1 \\ \alpha_2 & \beta_2 & \gamma_2 \\ \alpha_3 & \beta_3 & \gamma_3 \end{bmatrix} = \begin{bmatrix} x_0 & y_0 & z_0 \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}$$

Now we can easily invert the square matrix on the left, yielding the values of the unknown coefficients,

$$\begin{bmatrix} \alpha_0 & \beta_0 & \gamma_0 \\ \alpha_1 & \beta_1 & \gamma_1 \\ \alpha_2 & \beta_2 & \gamma_2 \\ \alpha_3 & \beta_3 & \gamma_3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} -1 & 1 & 1 & 1 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 & y_0 & z_0 \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}$$

Using the values forα_k, we find that $x(\vec{\xi})$ becomes

$$\begin{aligned} x(\vec{\xi}) &= \alpha_0 + \alpha_1\xi + \alpha_2\eta + \alpha_3\zeta \\ &= \left[\frac{1}{2}(-x_0 + x_1 + x_2 + x_3)\right] \\ &\quad + \left[\frac{1}{2}(-x_0 + x_1)\right]\xi \\ &\quad + \left[\frac{1}{2}(-x_0 + x_2)\right]\eta \end{aligned}$$

Rearranging terms, we get

$$\begin{aligned} x(\vec{\xi}) &= \left[\frac{1}{2}(-1 - \xi - \eta - \zeta)\right] x_0 \\ &\quad + \left[\frac{1}{2}(1 + \xi)\right] x_1 \\ &\quad + \left[\frac{1}{2}(1 + \eta)\right] x_2 \\ &\quad + \left[\frac{1}{2}(1 + \zeta)\right] x_3 \end{aligned}$$

$$\begin{aligned} x(\vec{\xi}) &= N_0(\vec{\xi})x_0 + N_1(\vec{\xi})x_1 + N_2(\vec{\xi})x_2 + N_3(\vec{\xi})x_3 \\ y(\vec{\xi}) &= N_0(\vec{\xi})y_0 + N_1(\vec{\xi})y_1 + N_2(\vec{\xi})y_2 + N_3(\vec{\xi})y_3 \\ z(\vec{\xi}) &= N_0(\vec{\xi})z_0 + N_1(\vec{\xi})z_1 + N_2(\vec{\xi})z_2 + N_3(\vec{\xi})z_3 \end{aligned}$$

where

$$
\begin{aligned}
N_0(\vec{\xi}) &= \frac{1}{2}(-1-\xi-\eta-\zeta) \\
N_1(\vec{\xi}) &= \frac{1}{2}(1+\xi) \\
N_2(\vec{\xi}) &= \frac{1}{2}(1+\eta) \\
N_3(\vec{\xi}) &= \frac{1}{2}(1+\zeta)
\end{aligned}
$$

As a final note, observe that we can streamline the evaluation process over any number of points, $\vec{\xi}_0, \vec{\xi}_1, \vec{\xi}_2, \ldots$, inside the same element via the following matrix multiplication

$$
\begin{bmatrix} x(\vec{\xi}_0) & y(\vec{\xi}_0) & z(\vec{\xi}_0) \\ x(\vec{\xi}_1) & y(\vec{\xi}_1) & z(\vec{\xi}_1) \\ x(\vec{\xi}_2) & y(\vec{\xi}_2) & z(\vec{\xi}_2) \\ \vdots & \vdots & \vdots \end{bmatrix} = \begin{bmatrix} N_0(\vec{\xi}_0) & N_1(\vec{\xi}_0) & N_2(\vec{\xi}_0) & N_3(\vec{\xi}_0) \\ N_0(\vec{\xi}_1) & N_1(\vec{\xi}_1) & N_2(\vec{\xi}_1) & N_3(\vec{\xi}_1) \\ N_0(\vec{\xi}_2) & N_1(\vec{\xi}_2) & N_2(\vec{\xi}_2) & N_3(\vec{\xi}_2) \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} x_0 & y_0 & z_0 \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}
$$

A.1.1 Jacobian Matrix

Recall the definition of the Jacobian matrix:

$$
\frac{\partial \vec{x}}{\partial \vec{\xi}} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \zeta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} & \frac{\partial y}{\partial \zeta} \\ \frac{\partial z}{\partial \xi} & \frac{\partial z}{\partial \eta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} \qquad \begin{aligned} x\left(\vec{\xi}\right) &= \textstyle\sum_i N_i\left(\vec{\xi}\right) x_i \\ y\left(\vec{\xi}\right) &= \textstyle\sum_i N_i\left(\vec{\xi}\right) y_i \\ z\left(\vec{\xi}\right) &= \textstyle\sum_i N_i\left(\vec{\xi}\right) z_i \end{aligned}
$$

$$
\frac{\partial x}{\partial \xi} = \sum_i \frac{\partial N_i}{\partial \xi} x_i \quad , \quad \frac{\partial x}{\partial \eta} = \sum_i \frac{\partial N_i}{\partial \eta} x_i \quad , \quad \frac{\partial x}{\partial \zeta} = \sum_i \frac{\partial N_i}{\partial \zeta} x_i
$$

$$
\frac{\partial y}{\partial \xi} = \sum_i \frac{\partial N_i}{\partial \xi} y_i \quad , \quad \frac{\partial y}{\partial \eta} = \sum_i \frac{\partial N_i}{\partial \eta} y_i \quad , \quad \frac{\partial y}{\partial \zeta} = \sum_i \frac{\partial N_i}{\partial \zeta} y_i
$$

$$
\frac{\partial z}{\partial \xi} = \sum_i \frac{\partial N_i}{\partial \xi} z_i \quad , \quad \frac{\partial z}{\partial \eta} = \sum_i \frac{\partial N_i}{\partial \eta} z_i \quad , \quad \frac{\partial z}{\partial \zeta} = \sum_i \frac{\partial N_i}{\partial \zeta} z_i
$$

Applying these to our shape functions on a tetrahedron, we obtain

i	TN_i	$\partial TN_i/\partial\xi$	$\partial TN_i/\partial\eta$	$\partial TN_i/\partial\zeta$
0	$\frac{1}{2}(-1-\xi-\eta-\zeta)$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$
1	$\frac{1}{2}(1+\xi)$	$+\frac{1}{2}$	0	0
2	$\frac{1}{2}(1+\eta)$	0	$+\frac{1}{2}$	0
3	$\frac{1}{2}(1+\zeta)$	0	0	$+\frac{1}{2}$

$$\frac{\partial x}{\partial \xi} = \left(-\frac{1}{2}\right) x_0 + \left(\frac{1}{2}\right) x_1 \ , \ \frac{\partial x}{\partial \eta} = \left(-\frac{1}{2}\right) x_0 + \left(\frac{1}{2}\right) x_2 \ , \ \frac{\partial x}{\partial \zeta} = \left(-\frac{1}{2}\right) x_0 + \left(\frac{1}{2}\right) x_3$$

$$\frac{\partial y}{\partial \xi} = \left(-\frac{1}{2}\right) y_0 + \left(\frac{1}{2}\right) y_1 \ , \ \frac{\partial y}{\partial \eta} = \left(-\frac{1}{2}\right) y_0 + \left(\frac{1}{2}\right) y_2 \ , \ \frac{\partial y}{\partial \zeta} = \left(-\frac{1}{2}\right) y_0 + \left(\frac{1}{2}\right) y_3$$

$$\frac{\partial z}{\partial \xi} = \left(-\frac{1}{2}\right) z_0 + \left(\frac{1}{2}\right) z_1 \ , \ \frac{\partial z}{\partial \eta} = \left(-\frac{1}{2}\right) z_0 + \left(\frac{1}{2}\right) z_2 \ , \ \frac{\partial z}{\partial \zeta} = \left(-\frac{1}{2}\right) z_0 + \left(\frac{1}{2}\right) z_3$$

$$\mathrm{J} = \left|\frac{\partial \vec{x}}{\partial \vec{\xi}}\right| = \begin{vmatrix} \frac{1}{2}(x_1 - x_0) & \frac{1}{2}(x_2 - x_0) & \frac{1}{2}(x_3 - x_0) \\ \frac{1}{2}(y_1 - y_0) & \frac{1}{2}(y_2 - y_0) & \frac{1}{2}(y_3 - y_0) \\ \frac{1}{2}(z_1 - z_0) & \frac{1}{2}(z_2 - z_0) & \frac{1}{2}(z_3 - z_0) \end{vmatrix}$$

$$= \frac{1}{8} \begin{vmatrix} 1 & x_0 & y_0 & z_0 \\ 1 & x_1 & y_1 & z_1 \\ 1 & x_2 & y_2 & z_2 \\ 1 & x_3 & y_3 & z_3 \end{vmatrix}$$

Appendix B

License

GNU GENERAL PUBLIC LICENSE Version 2, June 1991. Copyright (C) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software – to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps:

1. Copyright the software, and

2. Offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program" below refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification.") Each licensee is addressed as "you."

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

   Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

   In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version," you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found. For example:

> One line to give the program's name and a brief idea of what it does. Copyright © (year) (name of author)
>
> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

> Gnomovision version 69, Copyright © year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than 'show w' and 'show c'; they could even be mouse-clicks or menu items – whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

> Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.
>
> (signature of Ty Coon)
1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Bibliography

[1] Akin, J.E. (2005), *Finite Element Analysis with Error Estimators,* Butterworth-Heinemann, Oxford, 447 pp.

[2] Encyclopaedia of Cubature Formulas (1998-2005), Katholieke Universiteit Leuven, Dept. Computerwetenschappen, www.cs.kuleuven.be/~nines/research/ecf/

[3] Hughes, Thomas J.R. (2000), *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, Dover, 672 pp.

[4] Karniadakis, George E. and Spencer J. Sherwin (2005), *Spectral/*hp *Element Methods for Computational Fluid Dynamics, Second Edition*, Numerical Mathematics and Scientific Computation, Oxford, 657 pp.

[5] Uesu, D., L. Bavoil, S. Fleishman, J. Shepherd, and C. Silva (2005), Simplification of Unstructured Tetrahedral Meshes by Point-Sampling, *Proceedings of the 2005 International Workshop on Volume Graphics,* 157-165, doi: 10.2312/VG/VG05/157-165.