zedhat: an EIT tool library

Alistair Boyle
boyle@sce.carleton.ca

Carleton UNIVERSITY

A preliminary release of `zedhat`, an open source C library and command line interface for EIT image reconstruction. The goals and rationale for the development of this code are described.

Introduction

The new software `zedhat` ($\hat{\mathbf{z}}$) is a dynamically and statically linkable C library of EIT algorithms which provides a stable and tested infrastructure for building further EIT-based tools or for incorporating EIT into higher-level software which combines functionality from a variety of disciplines or modalities. The key idea is that EIT is not the end point but a building block that can and should be integrated with other techniques and tools. The software described here aims to follow that idea to a concrete implementation as a scalable and flexible library.

Zedhat initially aims to provide minimal, core EIT functionality sufficient to calculate an iterative solution. With a solid base and stable interface, the software will be expanded to incorporate greater functionality as it evolves. The software is open-source (BSD licensed) and available at zedhat.org and github.com/boyle/zedhat.

Purpose

The vision is to provide a building block (Figure 1) with minimal dependencies which can be used to develop solutions that incorporate EIT techniques.

One such realization is a SPICE circuit simulator where re-implementing SPICE in a monolithic SPICE+EIDORS solution would be an undertaking of large complexity. Instead, our approach is to build this EIT core and link to an open source simulator (ngspice) to build an integrated solution with minimal redundancy.

References

[1] A. Adler and W. R. B. Lionheart. Uses and abuses of EIDORS: an extensible software base for EIT. 27(5):S25–S42, May 2006.

[2] A. Adler, A. Boyle, F. Braun, M. G. Crabb, B. Grychtol, W. R. B. Lionheart, H. F. J. Tregidgo, and R. Yerworth. EIDORS 3.9. In *EIT2017*, Dartmouth, USA, June 2017.

Preliminary Release

This preliminary release offers a minimal set of numerical tools and interfaces. The release supports mesh loading for Netgen initially, with FEM assembly and CEM boundary conditions. Forward model conductivities may be fixed values, time varying, or frequency dependent Cole-Cole models. Initial support for storage and loading of measurement data $\mathbf{d}$ uses a simple text format. The library supplies an efficient implementation for calculating the forward solution, the Jacobian $\mathbf{J}$, and a time difference conductivity reconstruction $\hat{\mathbf{z}}$ from difference measurements $\mathbf{b} = \Delta\mathbf{d}$ using an $\ell 2$-norm Tikhonov regularized ($\mathbf{Q} = \mathbf{L}^\mathsf{T}\mathbf{L} = \mathbf{I}$) Gauss-Newton single-step method

$$\hat{\mathbf{z}} = \arg\min_{\mathbf{z}} ||\mathbf{Jb} - \mathbf{z}||_2^2 + ||\lambda\mathbf{Lz}||_2^2$$

$$\hat{\mathbf{z}} = (\mathbf{J}^\mathsf{T}\mathbf{J} + \lambda^2\mathbf{Q})^{-1}\mathbf{J}^\mathsf{T}\mathbf{b}$$

for hyperparameter λ. A command line program offers a minimal working example for use of the library.

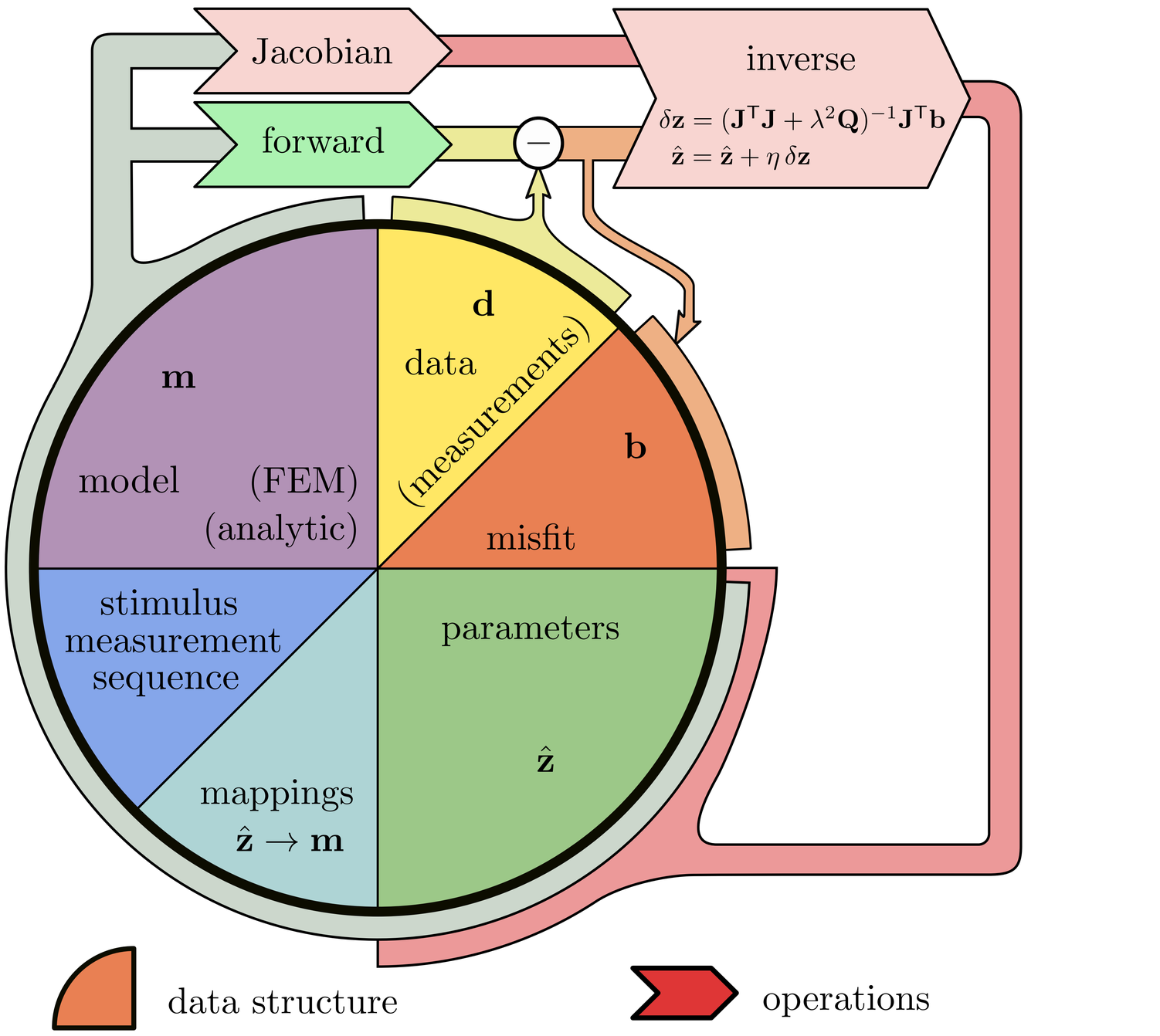

Figure 1: Gauss-Newton (GN) solver; operations define the library interface; core data structures are generic; the forward solutions cancel for time-difference 1-step GN ($\mathbf{b} = \Delta\mathbf{d}$)

Near Future

In the near future, we plan to support Python and Matlab language bindings. This enables a path to validating implementations against EIDORS/Matlab solutions, while isolating the library itself from the challenges of supporting these environments directly. Python, due to its continuing popularity and open source status, is currently the language of choice for many scientific domains. A binding to Python enables low-overhead integration with other tools, the ability to script functionality, and programming language features that many users have come to expect. The code is initially available on Linux platforms and does not provide a graphical user interface (GUI). Without GUI dependencies, we anticipate support for Windows and OSX in the short-term. We aim to achieve and maintain 100% line coverage for testing our code in a continuous integration framework enabling rapid evolution of the codebase and frequent releases without the accumulation of technical debt which can cripple rapid innovation.

We explicitly avoid, at this point, dealing with the complexities of mesh design, device specific file formats, and output visualization. EIDORS [1, 2] offers an exceptional array of application specific mesh generation tools and instrument file format support. EIDORS also enables rapid and efficient visualization of 2D and 3D simulation models and reconstructions. This library, particularly with a Matlab language binding, should eventually be complementary to EIDORS' comprehensive collection of inverse problems "ingredients" and exceptional selection of data handling and model generation tools.

Quick Facts

Language	C	✓
Linkable Library	C/C++	
Language Bindings	Matlab	
	Python	
Command Line Interface	text	✓
Graphical Interface	QT	
Supported File Formats	Netgen	✓
	Gmsh	
Open Source	BSD	✓
Revision Controlled	git	✓
Continuous Integration	Linux	✓
compiles on...	MacOS	✓
	Windows	✓
Tested (Line Coverage)	100%	✓
Lines of Code	`src`	2761
	`tests`	6656
	build	141
Citations	0	...

Dependencies: BLAS, LAPACKE, CHOLMOD (dense and sparse numerical linear algebra kernels)