Frameworks for Modeling and Inversion

William W. Symes

The Rice Inversion Project Department of Computational and Applied Mathematics Rice University, Houston, TX

symes@caam.rice.edu

7 June 2012

Open Source E+P Software

Project Goal:

Aid rapid prototyping of new wave modeling, inversion apps – wave physics, modeling methods, optimization algorithms, hardware & software environments – without starting from scratch every time

Find the right abstractions and data structures, create a proof-of-concept open source implementation

Components

- Modeling IWAVE
- Optimization RVL
- Inversion IWAVE++

Framework for Wave Modeling [IWAVE]

Why a framework?

- FD, FE apps share many common tasks:
 - Grid allocation
 - Data exchange patterns (domain decomposition) depend on scheme
 - i/o w. common file structures (SEGY/SU, RSF,...)
- Many of these reusable across many apps, given interfaces and task def'ns
- Unstructured-mesh FEM frameworks (mostly CFDoriented): PETSc, DUNE, deal.II, FEniCS, Trilinos,...
- Regular grid FD/FE TD: restricted domain additional opportunities for re-use, efficient implementation

Basic Data Structures for FDTD

Regular grid methods: Arrays = fcns on rectangular subsets of infinite lattice, virtual sub-arrays

Every virtual sub-array carries reference to parent allocated array
Arrays located relative to common lattice – can compute overlaps, data exchanges,...

State of system = union of static, dynamic arrays, with functions: constructors, data exchange, time updates

Source, receiver data: arbitrary locations, position relative to reference lattice, sampling & i/o



Methods:

- Create/resize subarrays
- Output
- Etc.

Our Implementation: IWAVE

- ISO C99
- "Object oriented C": built around small set of C structs containing data and functions operating on them
- Core modules: services memory, comm, i/o, job control
- State repn = RDOM = array of arrays
- State type = IMODEL struct with RDOM and many virtual sub-RDOMS, functions eg.

tsf(RDOM * p, int * iv, void * pars)

- Application modules ("apps") choices of physics, scheme
- Parallelism via domain decomposition, MPI.
- Reads, writes data in standard exchange formats: initially SEGY/SU, RSF,...
- Open source, X11 license

IWAVE & SEAM Phase I

- Public domain QC code

 high accuracy FDTD,
 ABCs, SEAM-sized data,
 scale to 1000's of cores
- Couldn't find one, so we built it
- For the details: new SEG e-book by Fehler & Keliher - many insights into project management, QC – role of IWAVE



Visual Comparison: Shot 20433, E-W line at N 10.585 km Tierra



Visual Comparison: Shot 20433, E-W line at N 10.585 km IWAVE



IWAVE Documentation

Download	- Madagascar 🔅 🗧	Index of /d	ata/iwave ×	IWAVE: Altern	ate Use Cases	×	Valgrind	× +	
www.trip.caam.rice.edu/software/iwave/asg/doc/html/alternate.html							িল ল লে 🚼 ব valgrind 🔍 🔍		٩ 1
Most Visited 🔻	🍓 Getting Started	🔝 Latest Headlines 🔻	Apple 🛛 Yaho	o! 🥂 Google Maps	You YouTube	W Wikipedia	📄 News 🔻 📄 Popu	ılar -	🛃 Bookmarks

Alternate Use Cases

This section describes various alternative tasks implemented by asg.x, and corresponding parameter selection patterns.

- 1. Create/modify output data file: it is possible to take headers from one file of SEGY traces, yet write the output to another. To do this, specify the source of the trace headers as hdrfile, and the output data file as datafile (as in the typical use case):
 - o hdrfile = [string]

Main Page Classes Files Related Pages

o datafile = [string]

In this case, the output file is overwritten in its entirety (including any SEGY header information).

2. Movie output - uses IWAVE movie output facility, see IWAVE sampler package documentation for more on the structure of this facility.

Usage:

1. add "movie[n]=[code]" lines to par file to specify field(s) to be sampled. Legit codes for ASG model: p (pressure), v1, v2, v3 (components of velocity, in the internal axis ordering, i.e. v1=z-component, v2=x-component, v3=y-component. Any other code will be ignored. Thus for example

movie1=p movie2=v3

will produce two movies, one of pressure, one of the y-component of velocity.

- add "moviestep=[real]" line to par file to specify time step between movie frames, in units of ms. Thus to sample ten frames per second, include "moviestep=100.0". Default = DEFSTEP*dt, where dt is the *internal* time step of the simulartor, and DEFSTEP is defined in <u>sample/include/movie.h</u> (so this default is not very useful, since it depends on dt which one doesn't know a priori - better to specify moviestep).
- 3. only 2D movies are produced. Therefore no 1D movies at all, and only a single plane may be sampled in the 3D case. The sampling plane is perpindicular to one of the coordinate axes; which axis it's perp to is the value of the "movieaxis3d=[int]", legit values 0,1,2, default = 2 (this choice refers to IWAVE's internal axis ordering, so in the standard (z x y) ordering it's the (z x) plane, but in the other common ordering (x y z) it's the (x y) plane depends on the

Abstract Optimization and Linear Algebra [Rice Vector Library]

Types for Abstract Optimization

Essential step: define types for high level abstractions – Space, Vector, (linear or nonlinear) Operator, Functional

Evaluation objects – organize the value of function & derivatives at a point, enforce coherency

With these, can express coordinate free algorithms of linear algebra and optimization – Krylov (CG etc.), Lanczos, Landweber, quasi-Newton,...

Our Implementation: the Rice Vector Library ("RVL")

C++ classes expressing calculus in Hilbert Space

Design Paper: Padula, Scott & S, ACM TOMS 2009

Standard interfaces to concrete data types – in-core, out-of-core, distributed,... - and operations on them:

DataContainer – data abstraction, forms Visitor pattern with
FunctionObject – encapsulates all actions on data

Abstract Optimization with RVL

Typical use: RTM looks like

```
MyKindaDataSpace dsp(...);
MyKindaModelSpace msp(...);
Vector m(msp); Vector g(msp); Vector d(dsp);
...
MyKindaModelingOp op(....);
OperatorEvaluation opeval(op,m);
opeval.getDeriv().applyAdj(d,g);
```

```
"THE MATH IS THE API"
```

Abstract Optimization with RVL

Built on RVL:

Optimization Library UMin: LBFGS, trust region G-N-K, CG, Arnoldi...

Abstract time-stepping library TSOpt, including universal implementation of optimal checkpointing (Griewank 92)

Plans: additional algorithms (L1, TV, TR-SQP, ...)

Inversion [RVL + IWAVE = IWAVE++]

Requirements for inversion implementations

- modeling:
 - forward modeling
 - linearized ("Born") modeling
 - adjoint (transposed) linearized modeling
- optimization algorithm, implementation
- interface between modeling and optimization

Natural type mismatch between modeling & optimization – concrete (grids, traces) vs. abstract (vectors, operators)

Internal state of simulator need not be a Vector object – only input and output types

Our Solution: IWAVE++

Middleware layer – C++ classes encapsulating

- input & output types for IWAVE as RVL vector classes
- out-of-core design
- Born & adjoint extensions of IWAVE function interfaces, eg. tsfa(RDOM * p, RDOM * r, int iv, void * pars)
- delegates time loop control, checkpointing to TSOpt

Defines abstract RVL Operator class combining fwd, lin, and adj modeling

```
[WWS, Enriquez & Sun, Geophys. Prosp. 11]
```

Our Implementation: IWAVE++

Simple expl – adjoint ("dot product") test

```
IWaveOp<(list of policy types)> op(dom, rng, pars, stream,
minit);
Vector<float> m(op.getDomain());
... (initialize m)...
OperatorEvaluation<float> opeval(op,m);
AdjointTest(opeval.getDeriv(), rnd, stream);
```

Output, Marmousi 4m grid, 960 cores, 4x1 dom decomp:

```
adjoint relation holds:|<Ax,y>-<x,A'y>|/|Ax||y| = 4.91893895e-12
< 100*macheps = 1.19209290e-05
<Ax,y> = -1.63633438e+05
<x,A'y> = -1.63633469e+05
|Ax||y| = 6.35299635e+09
7 June 2012 Open Source E+P Software
```

Releases and Availability



Open Source E+P Software

Releases, plans

- IWAVE:
 - initial release at SEG 09
 - initial through current (1.5) at TRIP web site <u>www.trip.caam.rice.edu</u>
 - beginning with next release (2.0, 12 Q3) at SourceForge svn rsf.svn.sourceforge.net/svnroot/rsf/trunk
 - download and install either
 - stand-alone, or
 - integrated with Madagascar
- RVL:
 - initial release 11 Q3
 - r1.2 coming 12 Q3
- IWAVE++:
 - beta
 - Initial release planned for early 13

Thanks to

- IWAVE team Igor Terentyev, Tanya Vdovina, Xin Wang
- RVL team Mark Gockenbach, Shannon Scott, Tony Padula, Hala Dajani
- IWAVE++ team Dong Sun, Marco Enriquez
- Max Deschantsreiter, John Anderson, Scott Morton, Christof Stork, Ted Baragy, Murtaza Ali, John Mellor-Crummey, John Washbourne
- SEAM Project, NSF, sponsors of TRIP

www.trip.caam.rice.edu/software