From Modeling to RTM and Inversion: interfacing SEAMX and TSOpt

William Symes

The Rice Inversion Project

February 20, 2009

Concept

Goal: Make SEAMX capabilities - large-scale, parallel, extensible FD modeling - available for research on inversion driven by time-domain simulation

Method: embed SEAMX in Rice Vector Library (RVL) `Operator` type, use RVL utilities (linear algebra, construction of least squares functions, etc.) and algorithms (CG, NLCG, LBFGS,...) to construct inversion applications.

RVL = latest incarnation of TRIP OO framework for linking complex simulation with optimization.

- originated with Hilbert Class Library (Mark Gockenbach, mid-90's)
- thesis work of Tony Padula, Shannon Scott, Hala Dajani
- design paper to appear in ACM TOMS March 09 (pdf of final draft on AR web page)

Structure of SEAMX

Common modules:

- `iwave` - defines internal details of simulation data structures, *given* modeling specs - *state* of system at one time
  - arrays, grids, utilities
  - MPI data exchange
  - "virtual functions" (pointers) defining specific models, called to organize internal details
- `term` - organizes sampling of traces, movie frames, input of source data - *time* of simulation, initializing structure.

Completed by selection of functions with specific interfaces defining *an individual time step* of a specific model, also initialization of arrays and other data structures - `init` function copies function pointers to `iwave` struct members ("concrete subclass").

Structure of TSOpt

- Based on `Algorithm` package - generic framework for iterative algorithms (A. Padula PhD 05)

  ```
  while ( term.query() ) step.run ();
  ```

  Standard *loop algorithm* - `term` is a `Terminator`, determines whether a stop condition has been attained; `step` is another `Algorithm`.
- `TSOpt::Sim` synthesizes `TimeStep` and `TimeTerm` into loop algorithm for time-domain modeler.
- `TSOpt::Jet` combines `TSOpt::Sim` for reference evolution ("source field" in RTM) with components for single steps of linearized (Born modeling) and adjoint evolution to create *single object* holding coherent implementations.
- `TSOpt::SimOp` and similar constructions combine `TSOpt::Jet` with RVL vector interface to create plug-in to RVL optimization code.

Linking SEAMX to TSOpt

Design of SEAMX consistent with TSOpt:

- time step code (`iwave`) separated from time control code (`term`)
- both "object oriented", with implementing constructor, destructor, core attributes

Wrapping of `iwave` as `TSOpt::TimeStep` and `term` as `Terminator` relatively straightforward.

RVL interface required construction of out-of-core `RVL::Space` classes - just what RVL was designed for!

Current status: basic wrappers finished and tested, space classes in testing.

Initial goal: acoustic RTM for use in Rami, Chao projects, plane wave operator for Dong's project.