Scalable seismic modeling with Azure Batch

George Iordanescu[1], Wee Hyong Tok[1], Philipp A. Witte[2],
Mathias Louboutin[2] and Felix J. Herrmann[2]
[1] Microsoft Corporation
[2] Georgia Institute of Technology, School of Computational Science and Engineering

1 Contents

2 Overview

Numerical seismic modeling lies at the core of many geoscience applications, such as subsurface imaging or CO2 monitoring, and involves modeling acoustic or elastic wave propagation in the subsurface. This physical process is modeled numerially by solving partial differential equations on large-scale two- and three-dimensional domains using finite-difference time-stepping. Typical seismic surveys, as used for resource exploration in the oil and gas industry, involve modeling seismic data for thousands of individual experiments. Solving wave equations for real-world problem sizes is computationally expensive, but as individual experiments are independent of each other, this process can be executed as an embarassingly parallel workload.

This guide provides a walk through of how to deploy a parallel seismic modeling workload to Azure using Azure Batch and Batch Shipyard. For discretizing and solving the underlying wave equations the example uses Devito, a domain-specific language compiler for finite-differnce modeling. Devito allows implementing wave equations as high-level symbolic Python expressions and automatically generates and compiles optimized C code during runtime. The Python script provided in the accompanying software models seismic data for a given seismic source location and stores the modeled data in Blob storage. To model seismic data for a large number of individual source experiments, the workload is deployed as a parallel job to a pool of workers using Batch Shipyard, a tool for provisioning and monitoring workloads with Azure Batch. Each task of the batch job corresponds to modeling seismic data for a given seismic source location and can be executed on

a single compute node or on a cluster of multiple nodes using message passing (Figure 1). By leveraging the scalabilty of Azure Batch, the modeling workflow can be scaled to thousands of tasks, while automatic scaling enables cost efficient provisioning of computational resources.

The following sections provide a brief overview of seismic modeling with Devito and step-by-step instructions how the bundle the software into a docker container and deploy it as a parallel workload to Azure. This process involves uploading the necessary user data, such as the seismic model and acquisition geometry to Blob storage and setting up the batch environment. This guide demonstrates how to submit and monitor your job and discusses some possible extensions and applications of this software.

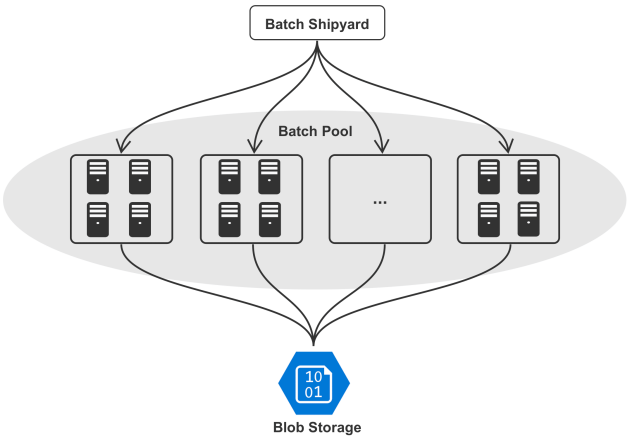


Figure 1: Parallel batch job for modeling seismic data. Jobs are submitted to a pool using Batch Shipyard and individual tasks either run on separate nodes or as distributed workloads on small clusters. Each task is responsible for modeling the data of a given source location and stores the generated data in Blob storage upon completion.

3 Seismic modeling with Devito

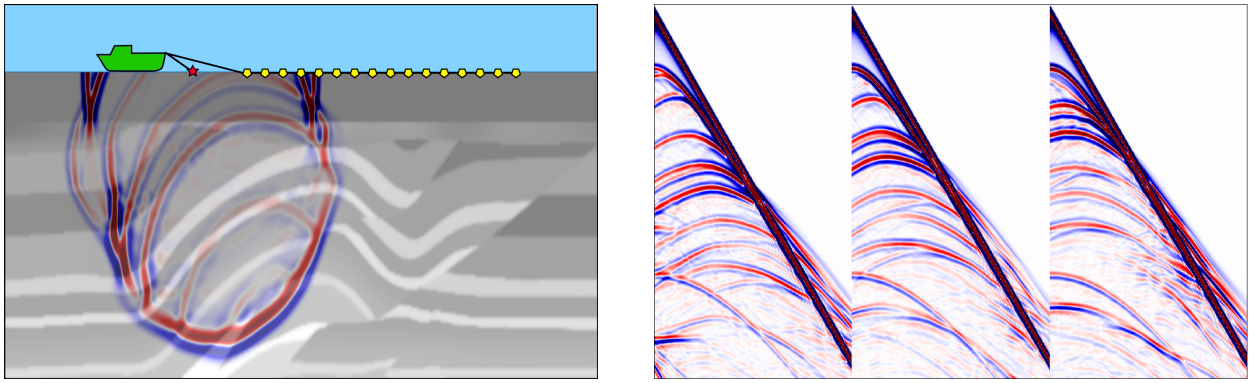
Seismic modeling is used in geophysical exploration and monitoring to numerically predict data that is recorded in seismic surveys. Such surveys involve a seismic source being repeateldy fired within the target area, which causes waves to propagate through the subsurface. Where physical properties of the subsurface such as wave speed or density change, waves are reflected and travel back to the surface, where they are recorded by an array of seismic sensors (Figure 2a). The sensors record relative pressure changes in the water as a function of time, sensor number and the source location (Figure 2b). Realistic surveys involve thousands of seismic source locations and data has to be modeled for each location individually by numerically solving the corresponding wave equation.

In this workflow, wave equations are discretized and solved with Devito, a Python package for finite-difference modeling and inversion. Devito's application programming interface allows implementing wave equations as symbolic Python expressions that closely resemble the mathematical notation:

```
pde = model.m * u.dt2 - u.laplace + model.damp*u.dt
```

where `model.m` is a symbolic expression for the acoustic wave speed, `u` is the discretized acoustic wavefield and `u.dt2` and `u.lapace` are short-hand expressions for finite difference stencils of second temporal and spatial derivatives. The last expression implements absorbing boundaries to mimic wave propagation in an unbounded domain. During runtime, Devito automatically generates optimized C code for solving this equation from the symbolic expression, using its internal compiler.

This tutorial demonstrates how use Azure Batch to model seismic data for a large number of source locations as an embarassingly parallel workload, making it possible to easily scale to relevant problem sizes as encountered in real-world scenarios. The software is deployed to a pool of parallel workers as a Docker container, which contains Devito implementations of the tilted transverse-isotropic (TTI) wave equation. This tutorial covers how to build the required Docker containers from scratch and how to manage the parallel pool and job submissions with Batch Shipyard.



(a) (b)

Figure 2: A marine seismic survey, in which a vessel tows a source and generates seismic waves that travel though the subsurface. At geological interfaces, wave are reflected back to the surface, where they are recorded by a set of seismic sensors (a). The sensors record pressure changes in the water as a function of time (vertical axis) and sensor number (horizontal axis) and the experiment is repeated for many source locations (b).

4 Prerequisites

- Install Azure CLI, Batch Shipyard
- Optional: Batch Explorer
- (Or run everything from George's docker container. Will it be made public?)

5 Set up

5.1 Docker container

- Optional: Build Docker containers and upload to Dockerhub
- Alternatively: Use pre-built containers

5.2 Upload user files

- Upload model + geometry
- Upload Python script

5.3 Batch shipyard configuration

- Fill out config files (`credentials.yaml`, `config.yaml`, `pool.yaml`, `jobs.yaml`)

6 Submit and monitor a job

- Start small pool: `./shipyard pool add -v` (then resize)
- Submit job: `./shipyard jobs add --tail stdout.txt -v`
- Check `stderr.txt` for Devito output
- Connect to compute nodes and run `top`
- Monitor CPU usage + memory in Batch Explorer

7 Performance analysis and results

- Show scalability
- Plot performance, FLOPs
- Download + plot seismic data

8 Clean up

- Kill job: `./shipyard jobs del -v`
- Shut down pool: `./shipyard pool del -v`

9 Next steps

Numerical seismic modeling functions as a key ingredient to a broad variety of applications, including subsurface imaging, parameter estimation or monitoring of geohazards. As such, the seismic modeling workflow demonstrated in this tutorial can function as a building block for more sophisticated workflows, which rely on forward modeling as the underlying workhorse. For example, instead of forward propagating a seismic source to model seismic data, recorded data from a seismic survey can be backpropagated in time by solving an adjoint wave equation, which results in a seismic image of the subsurface. An example of seismic imaging using a combination of Azure Batch and Azure Functions can be found here. Furthermore, the forward modeling module from this tutorial can function as a building block for iterative workflows, such as full-waveform inversion or least-squares imaging.