Devito Finite Differences - Important Quirks

Akhil Krishna Mohan

June 30, 2022

Abstract

This document intends to inform users of Devito that, depending on how a PDE is formulated in Devito, its propagation may be biased. This has potential to be an an issue every time we mix derivatives (e.g. $f(x, y).\frac{\partial}{\partial x}g(x, y)$. This is due to the fact that Devito sometimes shifts the computation of Finite Difference (FD) derivatives to ghost nodes between grid points. The disconnect in where things are defined (”at” grid points vs ”between” grid points), as well as the way Devito's inbuilt `grad` and `div` operations use the FD derivatives can lead to incorrect formulations of PDEs. This document examines the nature of Devito's FD derivatives and highlights the kinds of errors one can expect. In essence, correctly specifying a PDE using Devito's DSL can be a lot more difficult than it is made out to be.

Introduction

Devito is a powerful Domain-Specific Language that allows us to solve PDEs using finite difference stencils. It handles generating forward and adjoint propagators in C for efficient time-stepping. In particular, if we define a function u of space and time in our domain, u.dx defines the first derivative of u in the x-direction, and u.dy similarly in the y-direction. Problems arise when we need accurate stencils of higher orders. Typically, for a second derivative in the x-direction, we would use

$$u.\mathrm{dx2} \equiv \frac{\partial^2}{\partial x} u_{(x,y)} \approx \frac{u_{(x+1,y)} + u_{(x-1,y)} - 2u_{(x,y)}}{\Delta x^2} \tag{1}$$

This is the stencil generated when u.dx2 is called, and this is great. However, we get different behaviour if we use the notation (u.dx).dx, which instead approximates the above as

$$(u.\mathrm{dx}).\mathrm{dx} \equiv \frac{\partial^2}{\partial x} u_{(x,y)} \approx \frac{u_{(x+2,y)} + u_{(x,y)} - 2u_{(x+1,y)}}{\Delta x^2} \tag{2}$$

This discrepancy is slight, and for the most part has workarounds, but dealing with this issue has its nuances. We will build up to the behaviour of `grad` and `div`, but we choose to examine only x-derivatives (without loss of generality) until that point, for brevity.

Devito FD Mechanisms

The discrepancies in (1) and (2) are to be expected, since by default u.dx is equivalent to u.dxr, or a forward difference. This means that

$$u.\mathrm{dx} \equiv u.\mathrm{dxr} \equiv \frac{\partial}{\partial x} u_{(x,y)} \approx \frac{u_{(x+1,y)} - u_{(x,y)}}{\Delta x} \tag{3}$$

Accordingly, we have

$$u.\mathrm{dxl} \equiv \frac{\partial}{\partial x} u_{(x,y)} \approx \frac{u_{(x,y)} - u_{(x-1,y)}}{\Delta x} \tag{4}$$

and

$$u.\mathrm{dxc} \equiv \frac{\partial}{\partial x} u_{(x,y)} \approx \frac{u_{(x+1,y)} - u_{(x-1,y)}}{2\Delta x} \tag{5}$$

All of these are defined exactly this way when we set the spatial order of u to be 2 in Devito. Out of the above, only (5) seems to be a "true" second-order accurate scheme. However, one can argue that (3) and (4) are also second-order accurate. They are simply defined at $u_{\left(x+\frac{\Delta x}{2},y\right)}$ and $u_{\left(x-\frac{\Delta x}{2},y\right)}$ (ghost nodes on the grid). On the surface, this seems like a good workaround, but we have issues when we combine functions that are defined at different places. We consider the acoustic isotropic wave equation in 2D, but propagating only in x. The maths is identical for each dimension, but we look only at x-derivatives.

$$\frac{1}{v^2}\frac{\partial^2 u}{\partial t^2} - \frac{1}{b}\frac{\partial}{\partial x}\left(b\frac{\partial}{\partial x}u\right) - \frac{1}{b}\frac{\partial}{\partial y}\left(b\frac{\partial}{\partial y}u\right) = s \tag{6}$$

with s as a source term and b as buoyancy. Looking at the second term, we would aim to represent it with a stencil that is a function of $u_{(x-1,y)}, u_{(x,y)}$, and $u_{(x+1,y)}$, while also using $b_{(x-1,y)}, b_{(x,y)}$, and $b_{(x+1,y)}$.
We examine different ways of specifying the red term in (6) in Devito -

1.
```
pde = m * u.dt2 - 1/b * (b * u.dx).dx
```
gives us a stencil that is a function of $u_{(x,y)}, u_{(x+1,y)}$, and $u_{(x+2,y)}$, using $b_{(x,y)}$ and $b_{(x+1,y)}$. This is asymmetric (i.e. no $(x-1,y)$ terms), but also highlights a "bad" multiplication - b is defined at grid point (x,y) but u.dx is shifted, i.e. defined at $\left(x+\frac{\Delta x}{2},y\right)$.

2. We can set out to fix this by replacing u.dx with u.dxc. This will mean that b, defined "at" the grid points, will be multiplied by u.dxc, also defined at the grid points.
```
pde = m * u.dt2 - 1/b * (b * u.dxc).dxc
```
This stencil solves the problem of "bad" multiplications, but expands the stencil to be a function of $u_{(x-2,y)}, u_{(x,y)}$, and $u_{(x+2,y)}$, while using $b_{(x-1,y)}, b_{(x,y)}, b_{(x+1,y)}$. This is correctly defined, however, we can achieve second-order accuracy for this operation without using $u_{(x-2,y)}$ and $u_{(x+2,y)}$. In this stencil, we have an optimal stencil size, but we actually have 2 disjoint ("checkerboarded") grids (in 1D), since $u_{(x,y)}$ does not depend on $u_{(x+1,y)}$ and $u_{(x-1,y)}$.

3. Another approach would be to combine .dxr with .dxl. We could define
```
pde = m * u.dt2 - 1/b * (b * u.dxr).dxl
```
This has both of the previously mentioned drawbacks in that it uses the larger stencil, including $u_{(x-2,y)}, u_{(x-1,y)}, u_{(x,y)}, u_{(x+1,y)}$, and $u_{(x+2,y)}$, but uses shifted values of b, i.e. $b_{(x-2,y)}, b_{(x-1,y)}, b_{(x,y)}$. This stencil is then asymmetric in b and also seems to use the wrong kinds of multiplication.

4. We can also expand the derivatives in b algebraically before specifying the formulation in the PDE. This would look like
```
pde = m * u.dt2 - 1/b * (b * u.dx2 + b.dxc * u.dxc)
```
This gives us a stencil using $u_{(x-1,y)}, u_{(x,y)}, u_{(x+1,y)}, b_{(x-1,y)}, b_{(x,y)}$, and $b_{(x+1,y)}$, which we are after.

5. We can still do better. If we do not want to expand out the problem term ($\frac{\partial}{\partial x}\left(b\frac{\partial u}{\partial x}\right)$), we can replace it with some clever maths(1) to actually use Devito's inbuilt operators
```
pde = m * u.dt2 - 1/(2b) * ((b*u).laplace - u*(b.laplace) + b*(u.laplace))
```
This gives us the same stencil as approach 4, but generalizes better to higher dimensions as per (7).

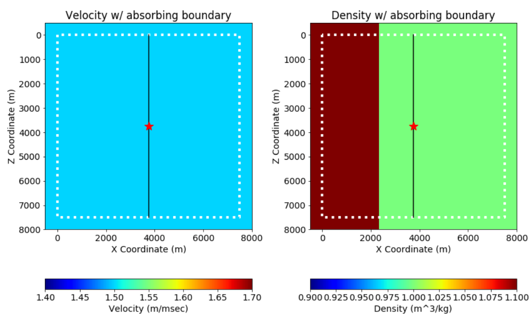
These FD models are examined in more detail in this notebook and results have been collated in this file.

Why is this a problem?

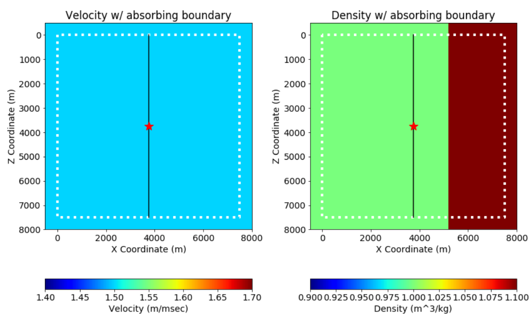
Apart from "bad" multiplications and larger-than-necessary stencils, there are more concerning problems with Devito.

- The checkerboarding pattern described above can be exploited to do asynchronous grid-updates and save memory, but by default Devito does not do this. For time order k, Devito keeps $k+1$ copies of the function and updates them completely. Thus, we are unable to exploit this for performance, and also end up with essentially 2^n disjoint grids within our computational grid, where n is the number of dimensions we formulate in.

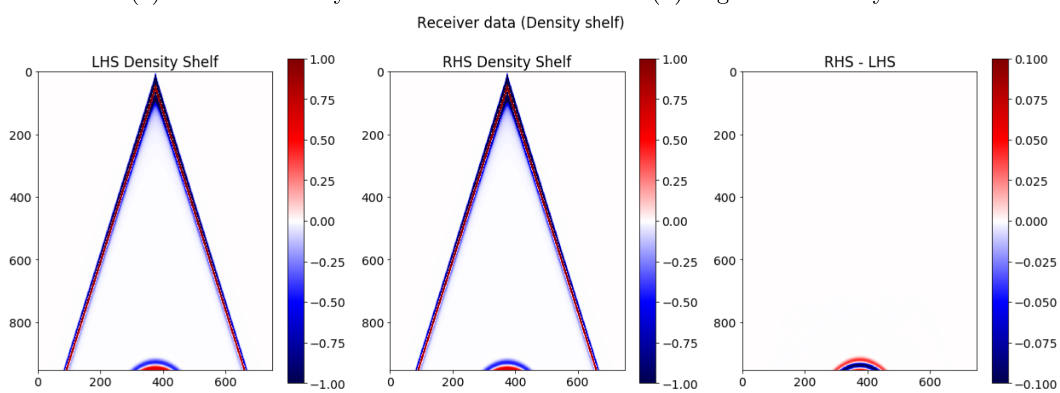
- Consider the code example provided in Devito's examples package. It defines a self-adjoint operator. However, as per a testing notebook I created, I was able to analyze if the propagation was symmetric in both directions.



(a) Left side density shelf



(b) Right side density shelf

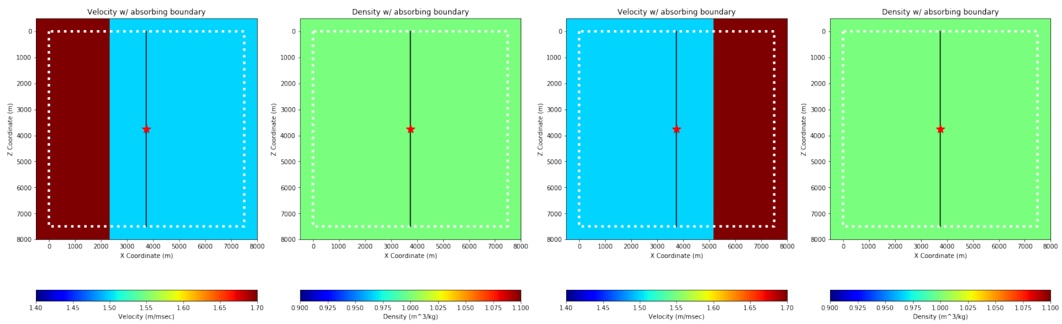


(c) Receiver data

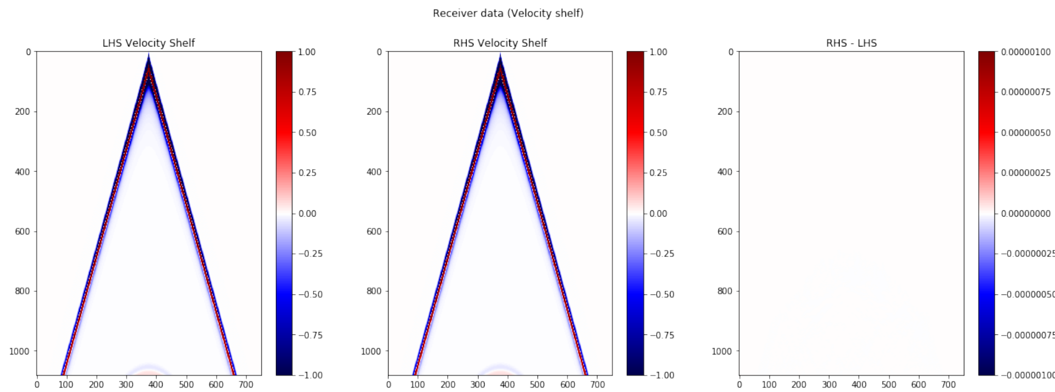
Figure 1: L-R Density Propagation

I found that when the velocity field is asymmetric, we see no difference in wave propagation, but when the buoyancy is asymmetric, the resulting waveform is shifted by a small amount, i.e. the propagator had a bias to the right side of the domain. This happens because of the above described FD models, and the in-built Devito implementation of `grad` and `div`. We never take a derivative of the velocity in the example, therefore the behaviour is unbiased. We do take derivatives of the buoyancy, and this leads to a biased propagation. The fact that propagation is not the same in the $+x$ and $-x$ directions should be critical enough to warrant our attention. Figures 1 and 2 demonstrate this. Figure 3 flips and overlays the wavelets of the left and right cases to demonstrate the asymmetric propagation.

- Since the implementations of `grad` and `div` inherently rely on the above Finite Difference(FD) models, they will suffer from similar shifting problems, stopping us from specifying PDEs in a convenient format using Devito's inbuilt operators. This means we need to be careful with how we specify our equations. Being unable to use Devito's inbuilt `grad` and `div` functions adds much

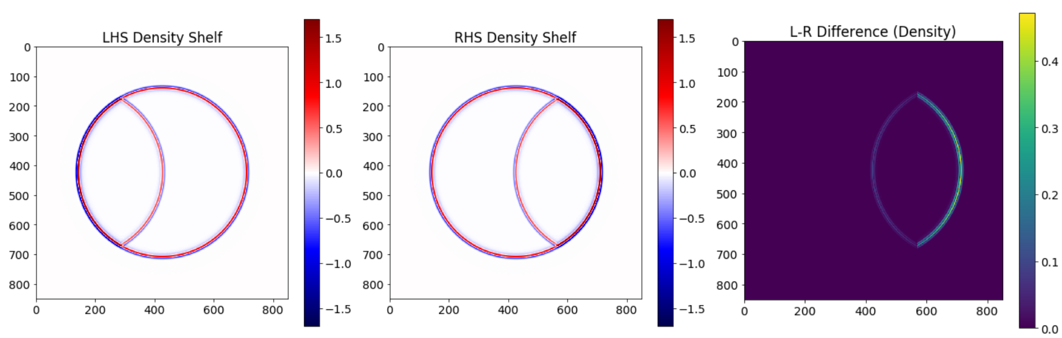


(a) Left side velocity shelf (b) Right side velocity shelf

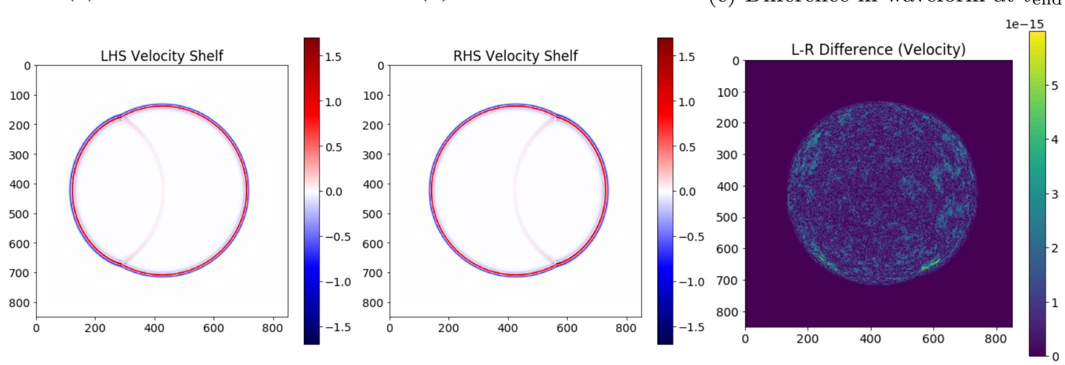


(c) Receiver data

Figure 2: L-R Velocity Propagation



(a) Waveform at t_{end} (b) Waveform at t_{end} (c) Difference in waveform at t_{end}



(d) Waveform at t_{end} (e) Waveform at t_{end} (f) Difference in waveform at t_{end}

Figure 3: Waveform Differences at t_{end}

clutter to the PDE formulation, and we would likely need to look into automatic code generation to make sure schemes are defined correctly.

Appendix

1. **Clever maths**

   We can identify the problem term as mentioned above as $\frac{\partial}{\partial x}\left(b\frac{\partial u}{\partial x}\right)$. Looking at it and $\frac{\partial^2}{\partial x^2}(bu)$ gives us -

$$\frac{\partial^2}{\partial x^2}(bu) = \frac{\partial}{\partial x}\left(b\frac{\partial u}{\partial x} + u\frac{\partial b}{\partial x}\right)$$

$$\equiv \frac{\partial^2}{\partial x^2}(bu) = b\frac{\partial^2 u}{\partial x^2} + u\frac{\partial^2 b}{\partial x^2} + 2\frac{\partial b}{\partial x}\frac{\partial u}{\partial x}$$

$$\frac{\partial}{\partial x}\left(b\frac{\partial u}{\partial x}\right) = b\frac{\partial^2 u}{\partial x^2} + \frac{\partial b}{\partial x}\frac{\partial u}{\partial x}$$

$$\implies \frac{\partial b}{\partial x}\frac{\partial u}{\partial x} = \frac{\partial}{\partial x}\left(b\frac{\partial u}{\partial x}\right) - b\frac{\partial^2 u}{\partial x^2}$$

$$\frac{\partial^2}{\partial x^2}(bu) = b\frac{\partial^2 u}{\partial x^2} + u\frac{\partial^2 b}{\partial x^2} + 2\left(\frac{\partial}{\partial x}\left(b\frac{\partial u}{\partial x}\right) - b\frac{\partial^2 u}{\partial x^2}\right)$$

$$\implies \frac{\partial}{\partial x}\left(b\frac{\partial u}{\partial x}\right) = \frac{\frac{\partial^2}{\partial x^2}(bu) - u\frac{\partial^2 b}{\partial x^2} + b\frac{\partial^2 u}{\partial x^2}}{2}$$

   Or, in higher dimensions

$$\nabla.(b\nabla u) = \frac{\nabla^2(bu) - u\nabla^2 b + b\nabla^2 u}{2} \tag{7}$$