Breaking the memory barrier (for finite difference modeling)

Jon Marius Venstad

Norwegian University of Science and Technology (NTNU)
Department of Petroleum Engineering & Applied Geophysics
E-mail: venstad@gmail.com
Supervisor: Børge Arntsen

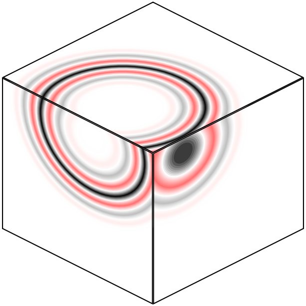
The ROSE meeting 2012
April 24th 2012

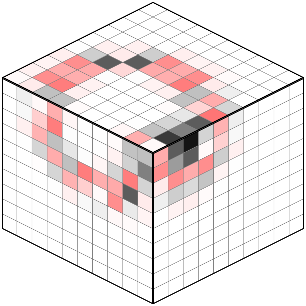
Overview

1. Finite Difference Modeling
2. The Memory Barrier
3. Breaking the Memory Barrier
4. Results
5. Conclusions

Finite Difference Modeling

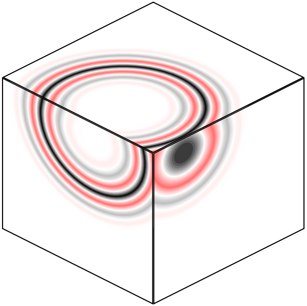
- Model physical phenomena, e.g. wave propagation.
- Bigger models and more calculations allow higher accuracy.
- Overwhelming memory requirements for 3D.
- Circumvent the memory limitations.

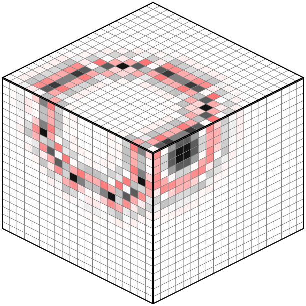




Finite Difference Modeling

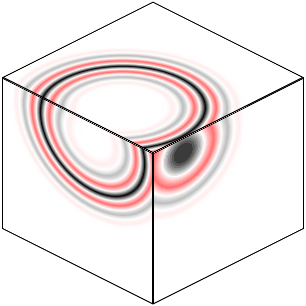
- Model physical phenomena, e.g. wave propagation.
- Bigger models and more calculations allow higher accuracy.
- Overwhelming memory requirements for 3D.
- Circumvent the memory limitations.

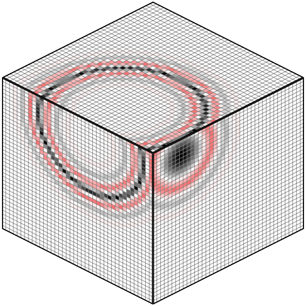




Finite Difference Modeling

- Model physical phenomena, e.g. wave propagation.
- Bigger models and more calculations allow higher accuracy.
- Overwhelming memory requirements for 3D.
- Circumvent the memory limitations.





Finite Difference Modeling

Model a given differential equation, e.g.:

$$\frac{\partial v_i}{\partial t} = \frac{1}{\rho}\left(\frac{\partial \tau_{ij}}{\partial j}\right) \tag{1}$$

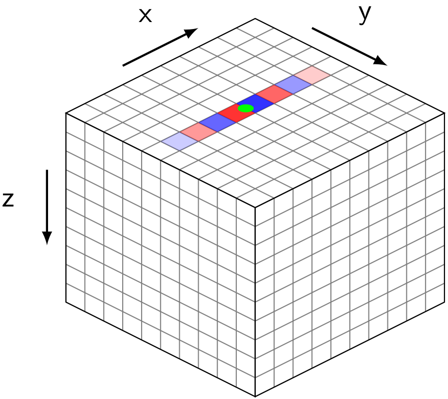
$$\frac{\partial \tau_{ij}}{\partial t} = \delta_{ij}\lambda\frac{\partial v_k}{\partial k} + \mu\left(\frac{\partial v_i}{\partial j} + \frac{\partial v_j}{\partial i}\right) \tag{2}$$

- 3-dimensional array for each variable.
- Approximate derivatives by weighted sums.
- Update each value across a small Δt.

A FDM example

$$\frac{\partial}{\partial x} u_{i+\frac{1}{2},j,k} \approx$$

0.0038	$u_{i-3,j,k}$
-0.0211	$u_{i-2,j,k}$
$+0.1049$	$u_{i-1,j,k}$
-1.2327	$u_{i,j,k}$
$+1.2327$	$u_{i+1,j,k}$
-0.1049	$u_{i+2,j,k}$
$+0.0211$	$u_{i+3,j,k}$
-0.0038	$u_{i+4,j,k}$



$$\tau_{xy}^{n+\frac{1}{2}} = \tau_{xy}^{n-\frac{1}{2}} + \mu \Delta t \left(\frac{\partial}{\partial x} v_y^n + \frac{\partial}{\partial y} v_x^n \right)$$

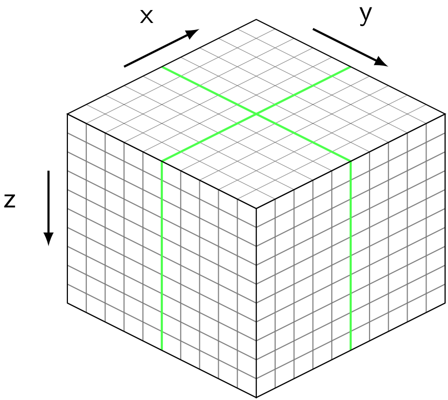
Seismic FDM

Model the full frequency spectrum of a seismic shot, in a fully anisotropic medium, in an area corresponding to a single shot:

- Wavelengths of 10m.
- 4m$\times$4m$\times$4m cells.
- $1000 \times 1000 \times 500$ grid cells.
- 24GB of data. (61GB anisotropic.)

The Memory Barrier – Fitting data into RAM

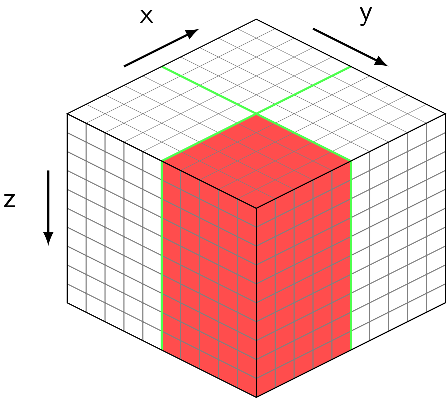
Split the model in smaller parts:



Only the blue cells are correctly updated.

The Memory Barrier – Fitting data into RAM

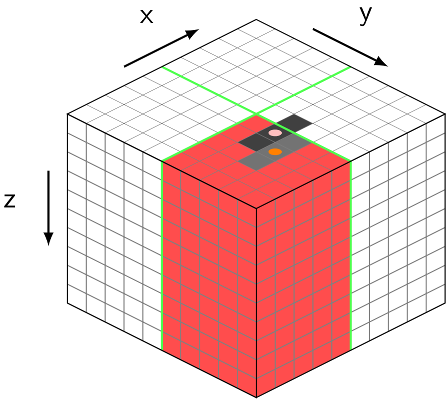
Split the model in smaller parts:



Only the blue cells are correctly updated.

The Memory Barrier – Fitting data into RAM

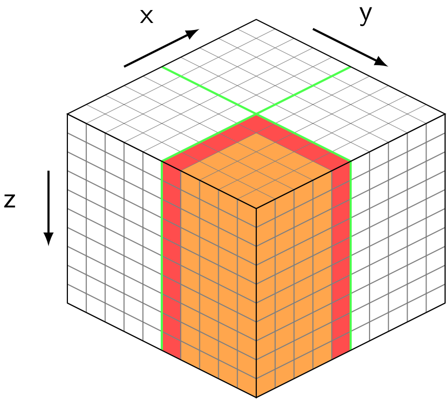
Split the model in smaller parts:



Only the blue cells are correctly updated.

The Memory Barrier – Fitting data into RAM

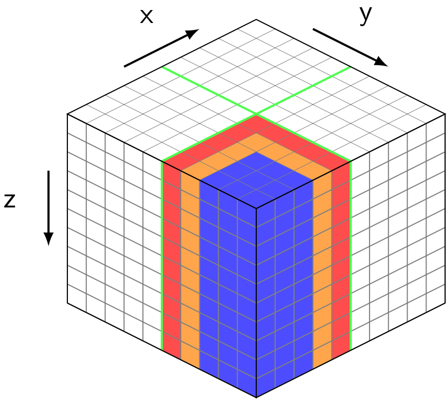
Split the model in smaller parts:



Only the blue cells are correctly updated.

The Memory Barrier – Fitting data into RAM

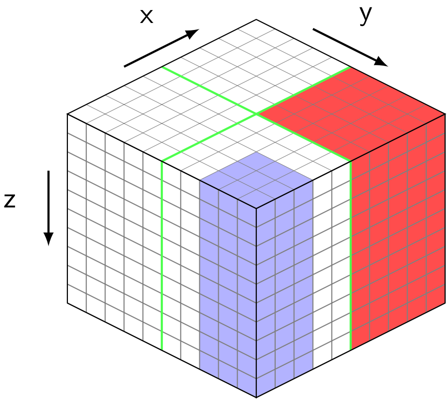
Split the model in smaller parts:



Only the blue cells are correctly updated.

The Memory Barrier – Fitting data into RAM

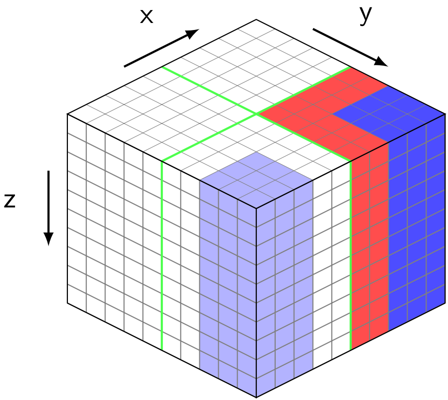
Split the model in smaller parts:



Only the blue cells are correctly updated.

The Memory Barrier – Fitting data into RAM

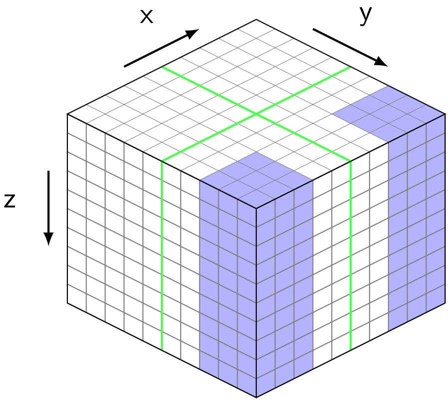
Split the model in smaller parts:



Only the blue cells are correctly updated.

The Memory Barrier – Fitting data into RAM

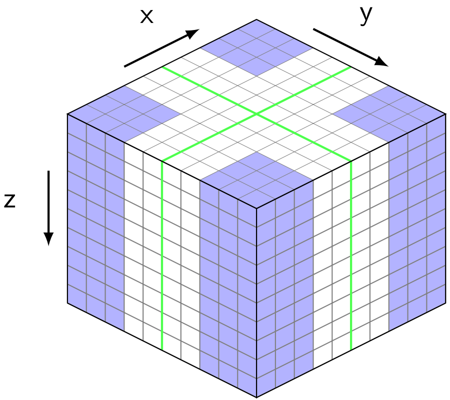
Split the model in smaller parts:



Only the blue cells are correctly updated.

The Memory Barrier – Fitting data into RAM

Split the model in smaller parts:



Only the blue cells are correctly updated.

The Memory Barrier – Data Transfer Slowdown

Moving the data to/from RAM becomes a bottleneck:

Work	Speed	Time
24×2GB	50MB/s	1000s
$1.5 \cdot 10^{11}$flop	10^9flop/s	150s

- An 8-fold time increase is not acceptable.
- We want IO time $\leq$ CPU time.

Breaking the Memory Barrier

1. Correctly calculate all cell updates.
2. Increase the CPU vs IO ratio.

Breaking the Memory Barrier

1. Correctly calculate all cell updates.

   Use overlapping model blocks.

2. Increase the CPU vs IO ratio.

Breaking the Memory Barrier

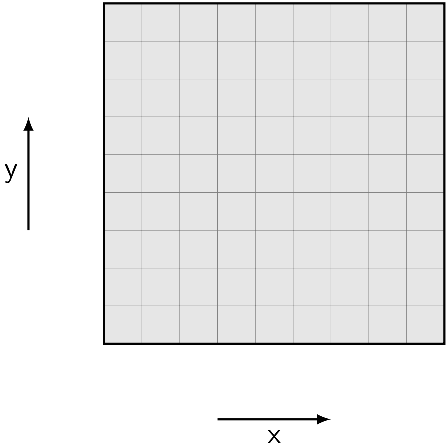
1. Correctly calculate all cell updates.

   Use overlapping model blocks.

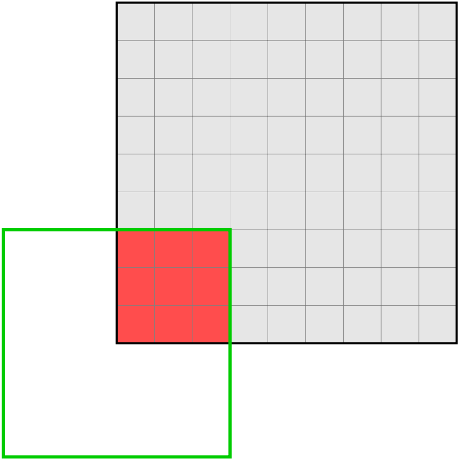
2. Increase the CPU vs IO ratio.

   Calculate several time steps per sweep.

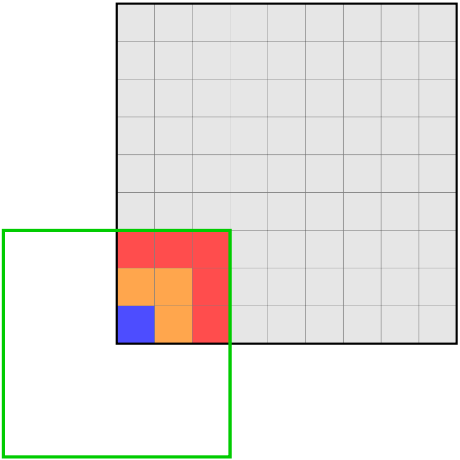
Breaking the Memory Barrier



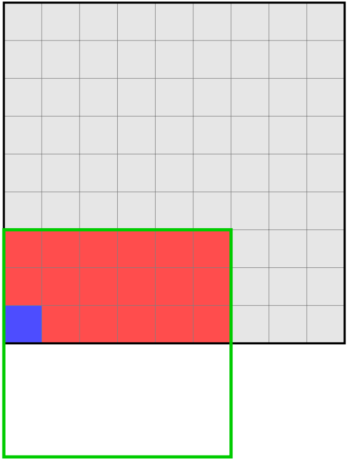
Breaking the Memory Barrier



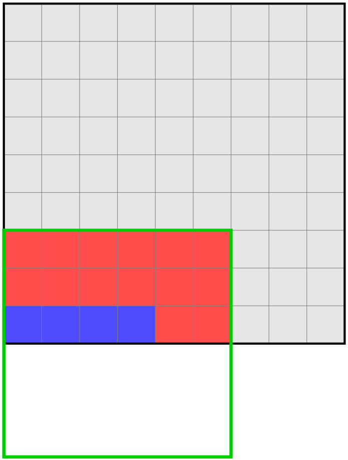
Breaking the Memory Barrier



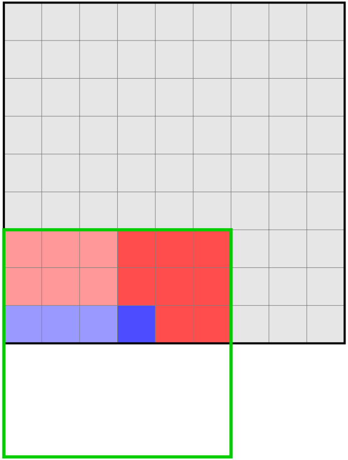
Breaking the Memory Barrier



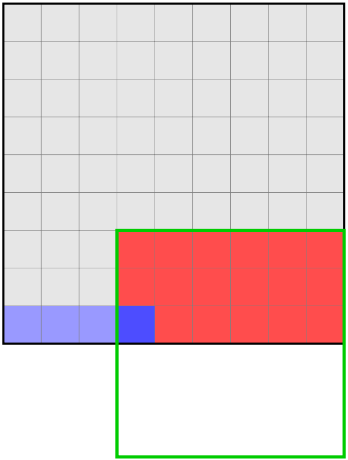
Breaking the Memory Barrier



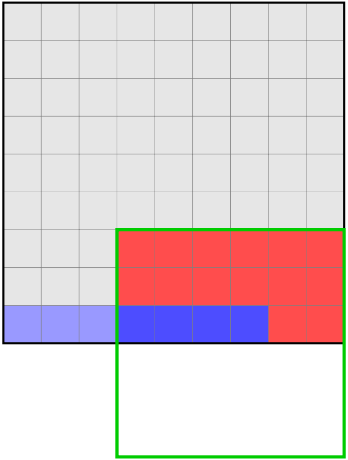
Breaking the Memory Barrier



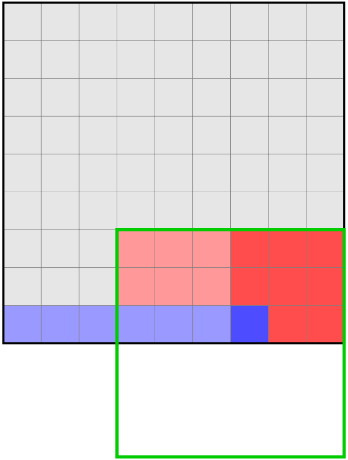
Breaking the Memory Barrier



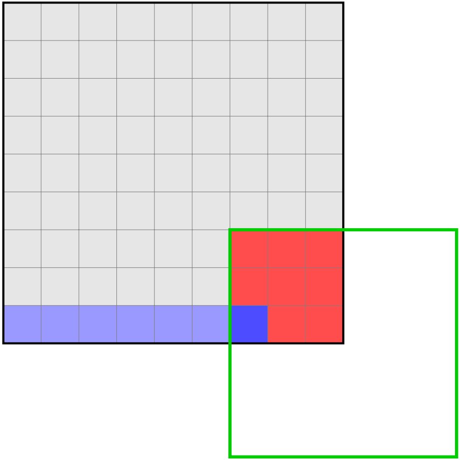
Breaking the Memory Barrier



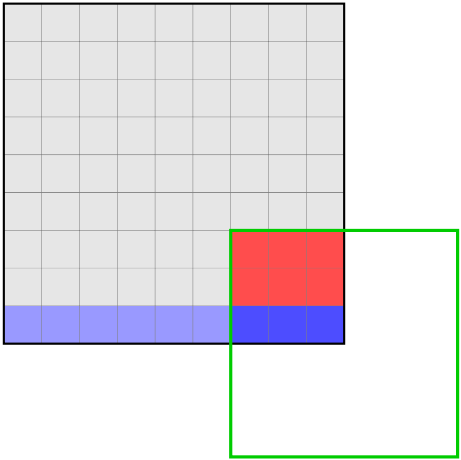
Breaking the Memory Barrier



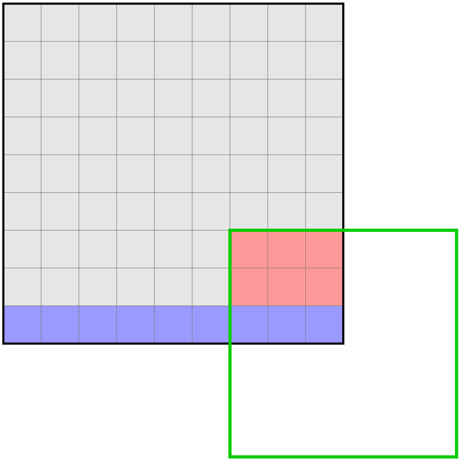
Breaking the Memory Barrier



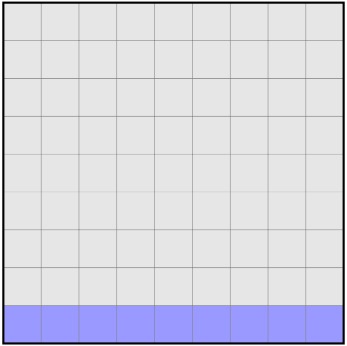
Breaking the Memory Barrier



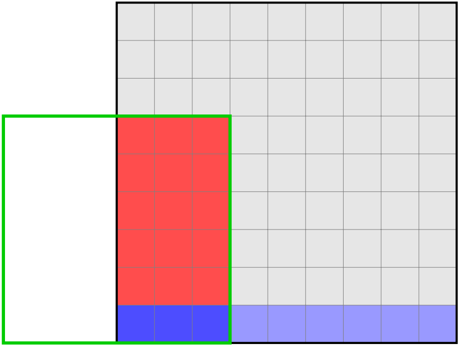
Breaking the Memory Barrier



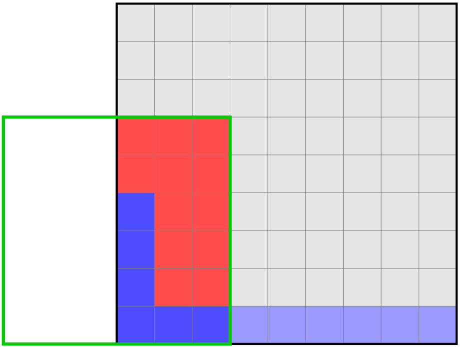
Breaking the Memory Barrier



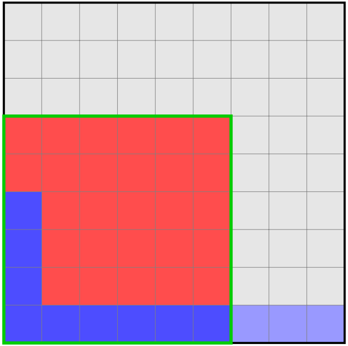
Breaking the Memory Barrier



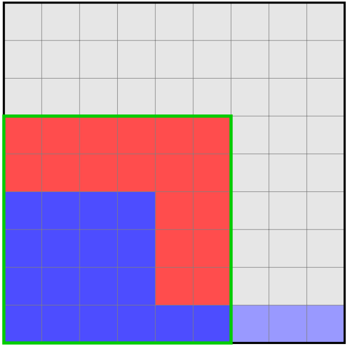
Breaking the Memory Barrier



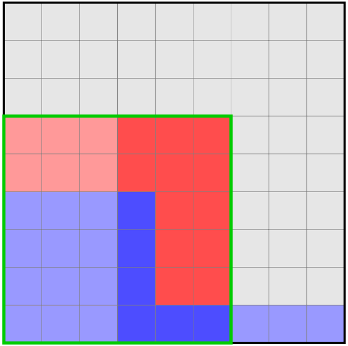
Breaking the Memory Barrier



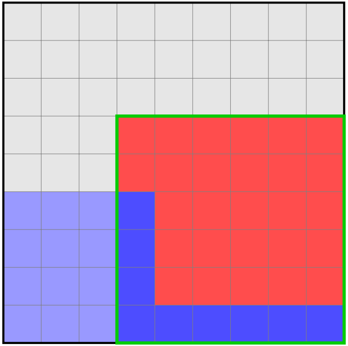
Breaking the Memory Barrier



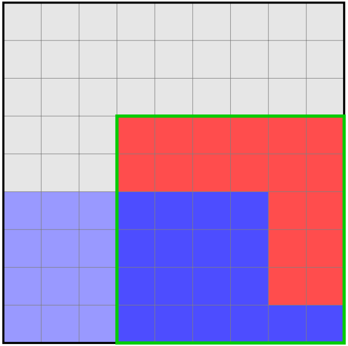
Breaking the Memory Barrier



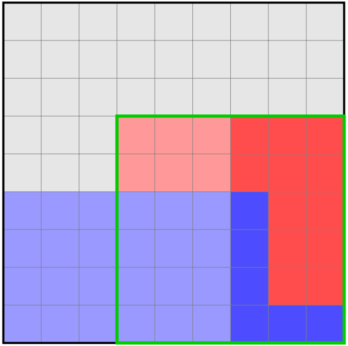
Breaking the Memory Barrier



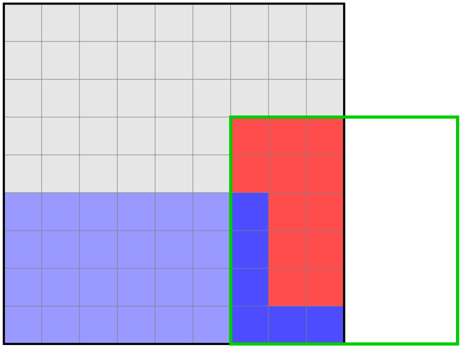
Breaking the Memory Barrier



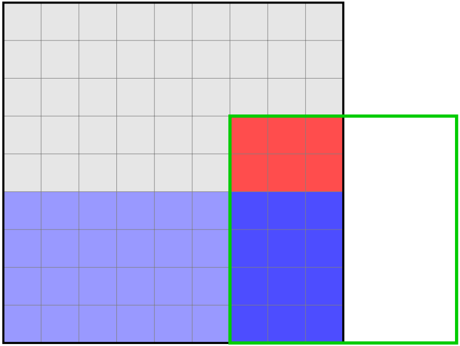
Breaking the Memory Barrier



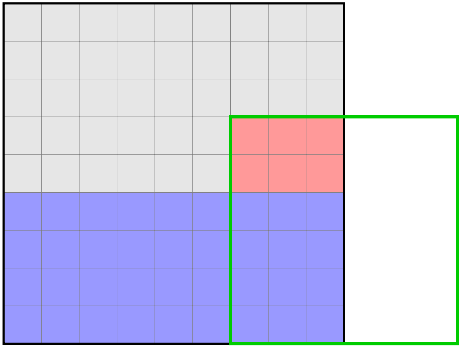
Breaking the Memory Barrier



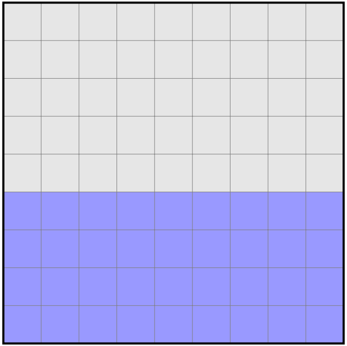
Breaking the Memory Barrier



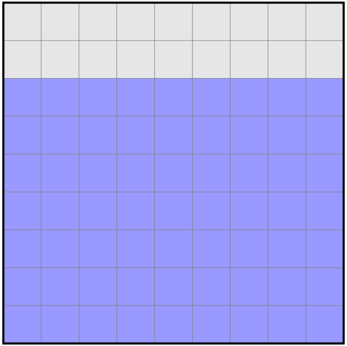
Breaking the Memory Barrier



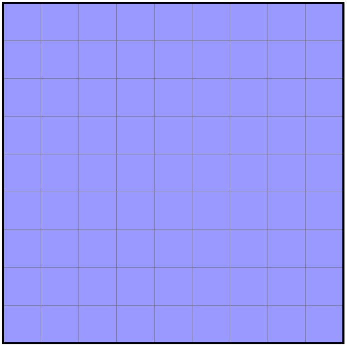
Breaking the Memory Barrier



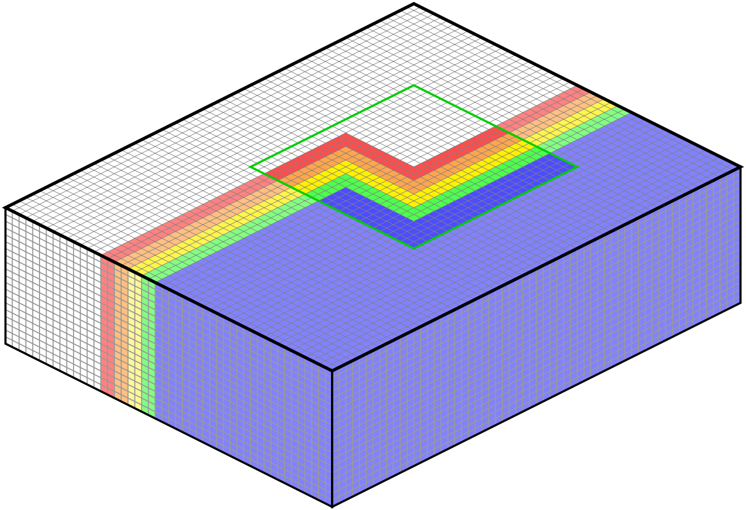
Breaking the Memory Barrier



Breaking the Memory Barrier

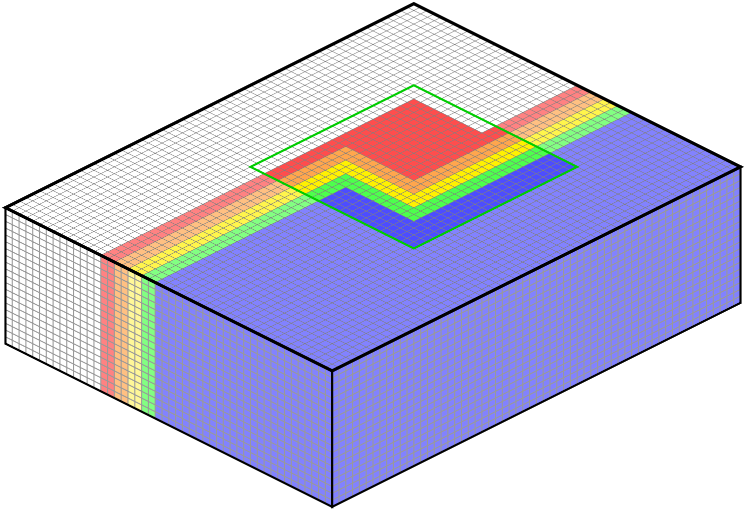


Breaking the Memory Barrier



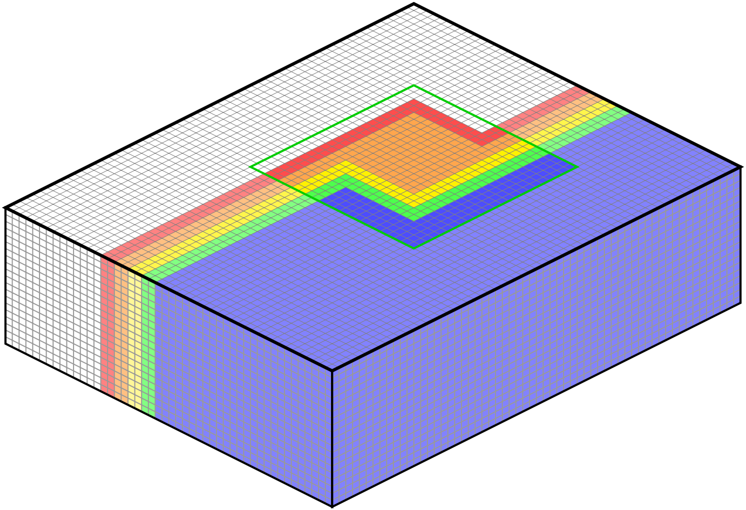
Number of time steps per sweep: $\frac{C-2L}{4L}$

Breaking the Memory Barrier



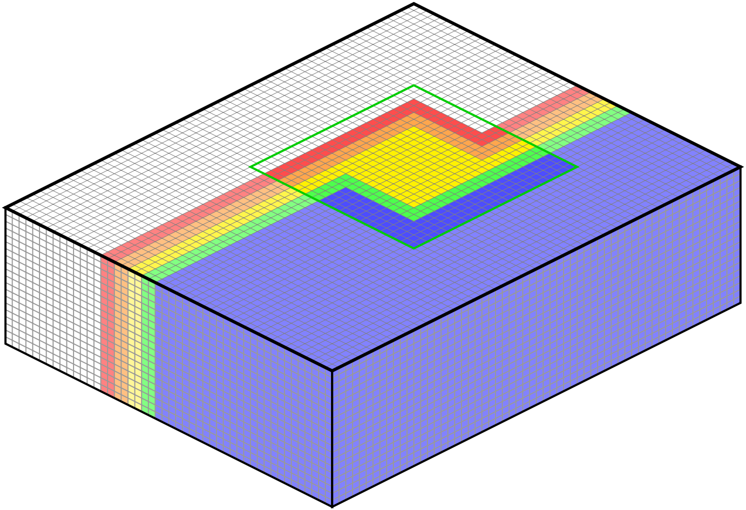
Number of time steps per sweep: $\frac{C-2L}{4L}$

Breaking the Memory Barrier



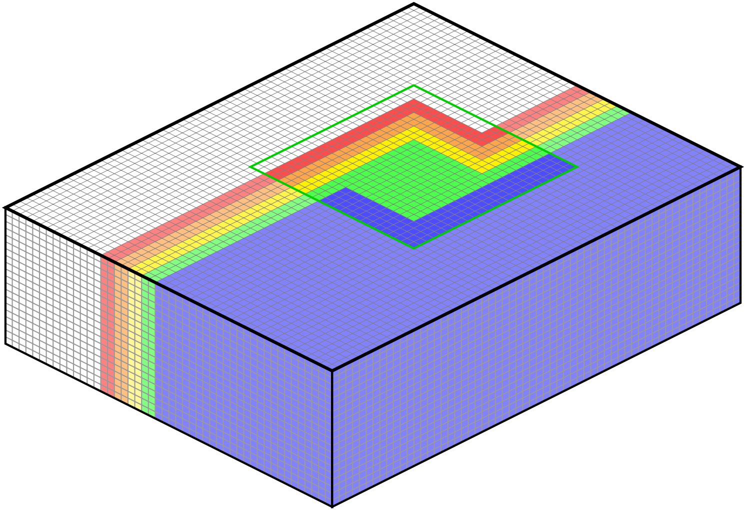
Number of time steps per sweep: $\frac{C-2L}{4L}$

Breaking the Memory Barrier



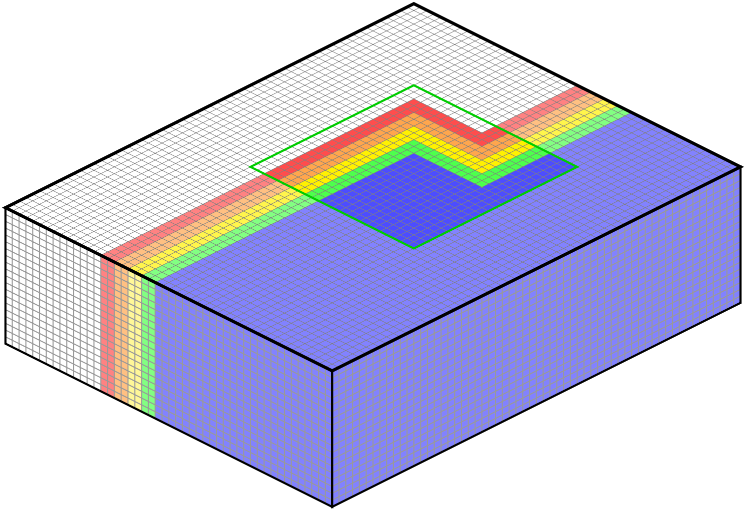
Number of time steps per sweep: $\frac{C-2L}{4L}$

Breaking the Memory Barrier



Number of time steps per sweep: $\frac{C-2L}{4L}$

Breaking the Memory Barrier



Number of time steps per sweep: $\frac{C-2L}{4L}$

Breaking the Memory Barrier

- $C = 200$ and $L = 4$ gives 12 time steps per sweep, or 6 time steps per read/write.
- Almost the same as the IO / CPU ratio.
- Only $200 \times 200 \times 500 \times 12 \times 4 \approx 1$GB.

Results

Tests on a laptop computer, using the 24GB model:

- 1.25GB memory.
- Cycles of 7s CPU – 8s IO.
- 150s per time step.
- 20 000 time steps: 5 weeks on the laptop.

Results

Tests on a laptop computer, using the 24GB model:

- 1.25GB memory.
- Cycles of 7s CPU – 8s IO.
- 150s per time step.
- 20 000 time steps: 5 weeks on the laptop.
    - Reduced 50% by asynchronous IO.

Results

Tests on a laptop computer, using the 24GB model:

- 1.25GB memory.
- Cycles of 7s CPU – 8s IO.
- 150s per time step.
- 20 000 time steps: 5 weeks on the laptop.
  - Reduced 50% by asynchronous IO.
  - Time usage proportional to f^4 and v_{min}^{-4}.

Results

Tests on a laptop computer, using the 24GB model:

- 1.25GB memory.
- Cycles of 7s CPU – 8s IO.
- 150s per time step.
- 20 000 time steps: 5 weeks on the laptop.
  - Reduced 50% by asynchronous IO.
  - Time usage proportional to f^4 and v_{min}^{-4}.
  - Corresponds to 3h on a single GPU!

Conclusions

- Large scale FDM possible without large memory requirements.
- Computational speed is still an issue.
- Need to maintain the CPU vs IO ratio. Factors:
  - Faster CPU.
  - Faster IO.
  - More memory.
- Robust alternative to parallellisation when several modelings are needed.
- The only cost is code complexity!

Considerations

- Smaller steps halve the memory requirements.
- Eliminate the IO wait with asynchronous IO.
- Share static data within nodes.
- Use GPUs with data streamed from RAM.

Acknowledgements

I wish to acknowledge Statoil for funding this research, the sponsors and organisers of the ROSE Meeting for making the conference happen and the Department of Petroleum Engineering & Applied Geophysics for being a great work place.

venstad@gmail.com