A Conflict Between Chip Architecture and a Direction of Ocean Model Development

Alistair Adcroft
Princeton University

Atmosphere, Oceans, and Computational Infrastructure
California Institute of Technology
May 16-18, 2018

When will my models become non-competitive?

Alistair Adcroft
Princeton University

Atmosphere, Oceans, and Computational Infrastructure
California Institute of Technology
May 16-18, 2018

Theme of talk

- General vertical coordinate algorithms, and new [physical] parameterizations of sub-grid scale processes, look more and more like column-wise code and some are non-vectorizable
  - This is non-optimal on existing architectures but worse moving forward
- GPUs offer significant speedups only on parts of the code
  - Effective speedup of the whole model is moderate, if even positive
- The general coordinate algorithms allow comparably very large time-steps which [currently] far out pace the benefits of GPUs
  - Are there ways to get the new algorithms to work on GPUs or TPUs?

Time-step and efficiency limitations

- Sound waves are filtered out
- Surface gravity waves
  - $c_{bt} \sim \sqrt{gH} \sim$ 200 m/s
- Internal waves
  - $c_{iw} \sim NH \sim$ 2 m/s
- Currents
  - $U \sim$ 1-3 m/s
  - $W_{eddy} \sim$ 10 m/day
  - $W_{IG\ waves} \sim$ 100 m/day
  - $W_{overturns} \sim U \sim$1 cm/s

	100 km	25 km	1 km	100 m
$\Delta x / c_{bt}$	8 minutes	2 minutes	5 seconds	½ second
$\Delta x / c_{iw}$	12 hours	3 hours	8 minutes	4 minutes
$\Delta x / U$	1 day	2 hours	5 minutes	2 minutes
f^{-1}	1.9 hours	1.9 hours	1.9 hours	1.9 hours
	50 m	10 m	1 m	1 cm
$\Delta z / W$	1 days	2 hours	15 minutes	10 seconds

- At coarse resolutions, numerics are dominated by considerations of rotational dynamics
- At fine resolutions, codes/methods should look more towards CFD
- **External mode continues to be a critical barrier**

Barotropic solver

- Equations are not up for debate!
  - Need to be solved with particular methods for consistency
  - **Discrepancies give tsunamis instead of tides**
- Small amount of computation
  - Almost linear
- Frequent communication
- Latency sensitive

$$\partial_t u_k = -\frac{1}{\rho_0}\nabla p + \cdots$$

$$\partial_t h_k = \nabla \cdot (h_k u_k)$$

Solve with baroclinic Δt_{bc}

SSH (constraint):

$$\partial_t \eta = \sum_k \partial_t h_k = \nabla \cdot \sum_k h_k u_k = \nabla \cdot (HU)$$

Reconcile with:

$$\partial_t \eta = \nabla \cdot (HU)$$

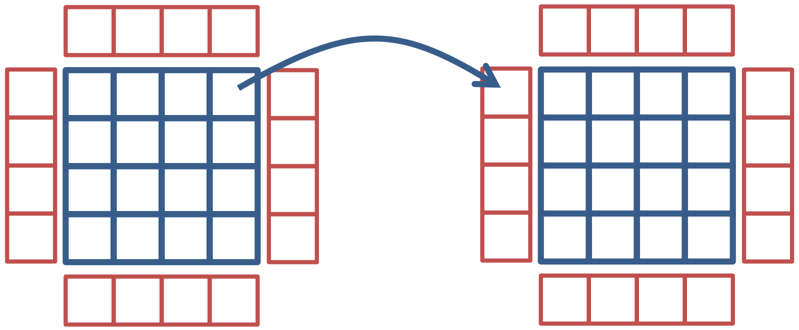
$$\partial_t U = -g\nabla \eta + \cdots$$

Solve with barotropic Δt_{bt}

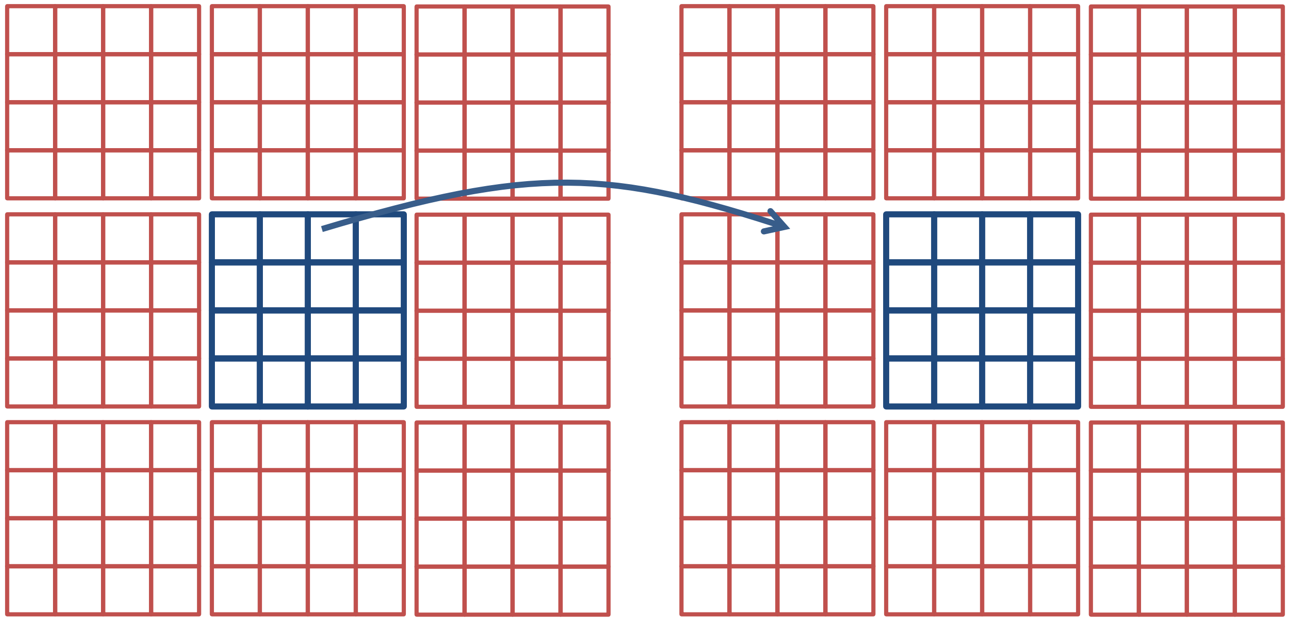
$$\Delta t_{bt} \ll \Delta t_{bc}$$

Adjustable strategies for barotropic solver

- Low latency/low bandwidth
  - Send small packets frequently

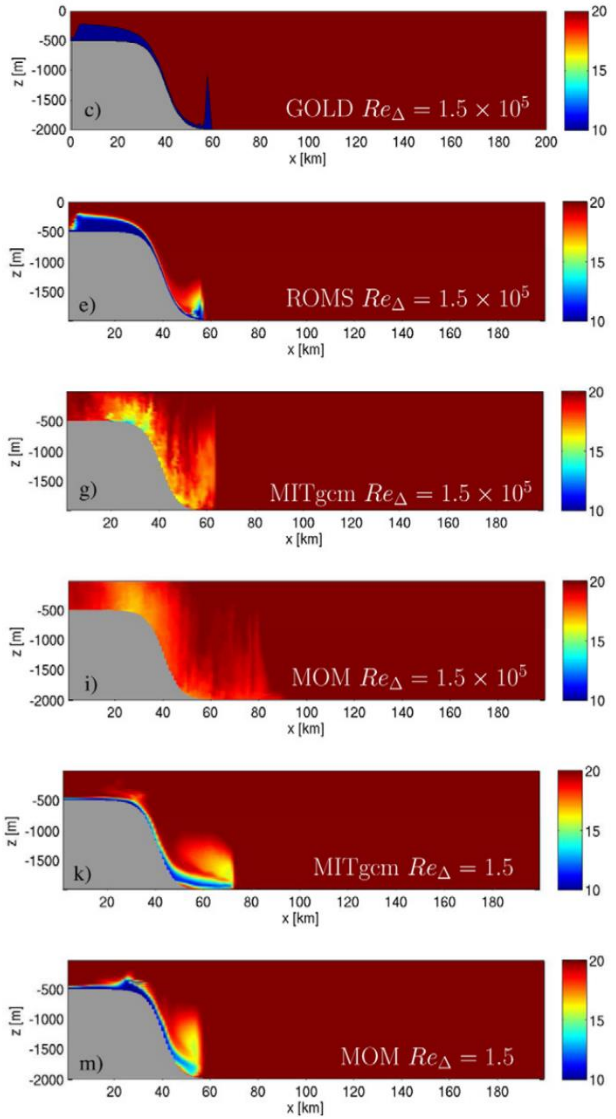


- High bandwidth/high latency
  - Send large packets infrequently



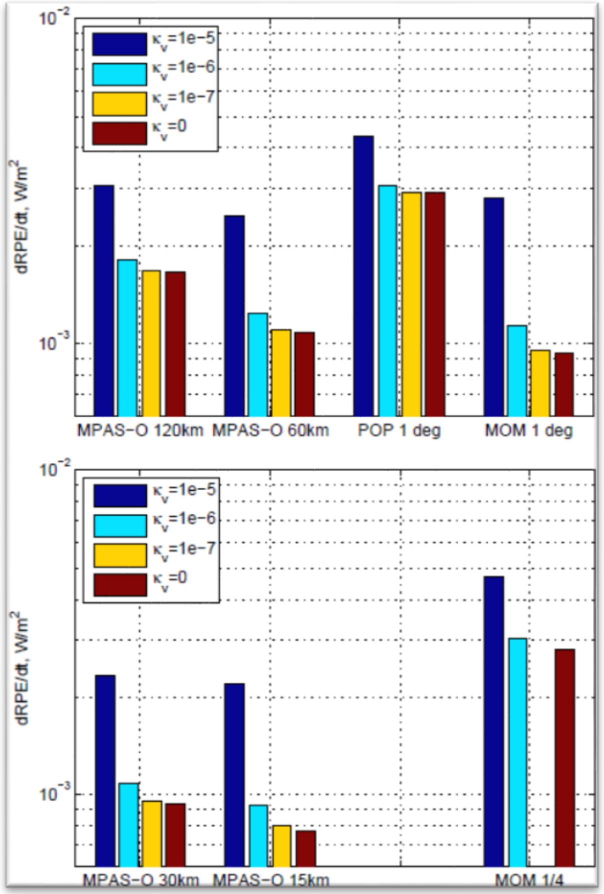
- Signal propagates across many cores in a single baroclinic time step

Vertical coordinates and spurious mixing

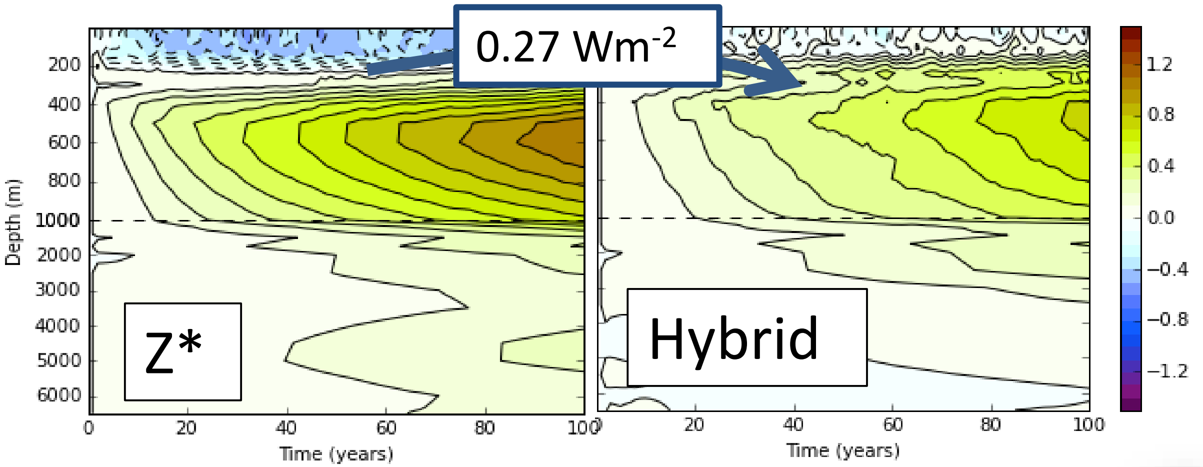


Coordinate MIP

Ilicak et al., 2012; Petersen et al., 2014

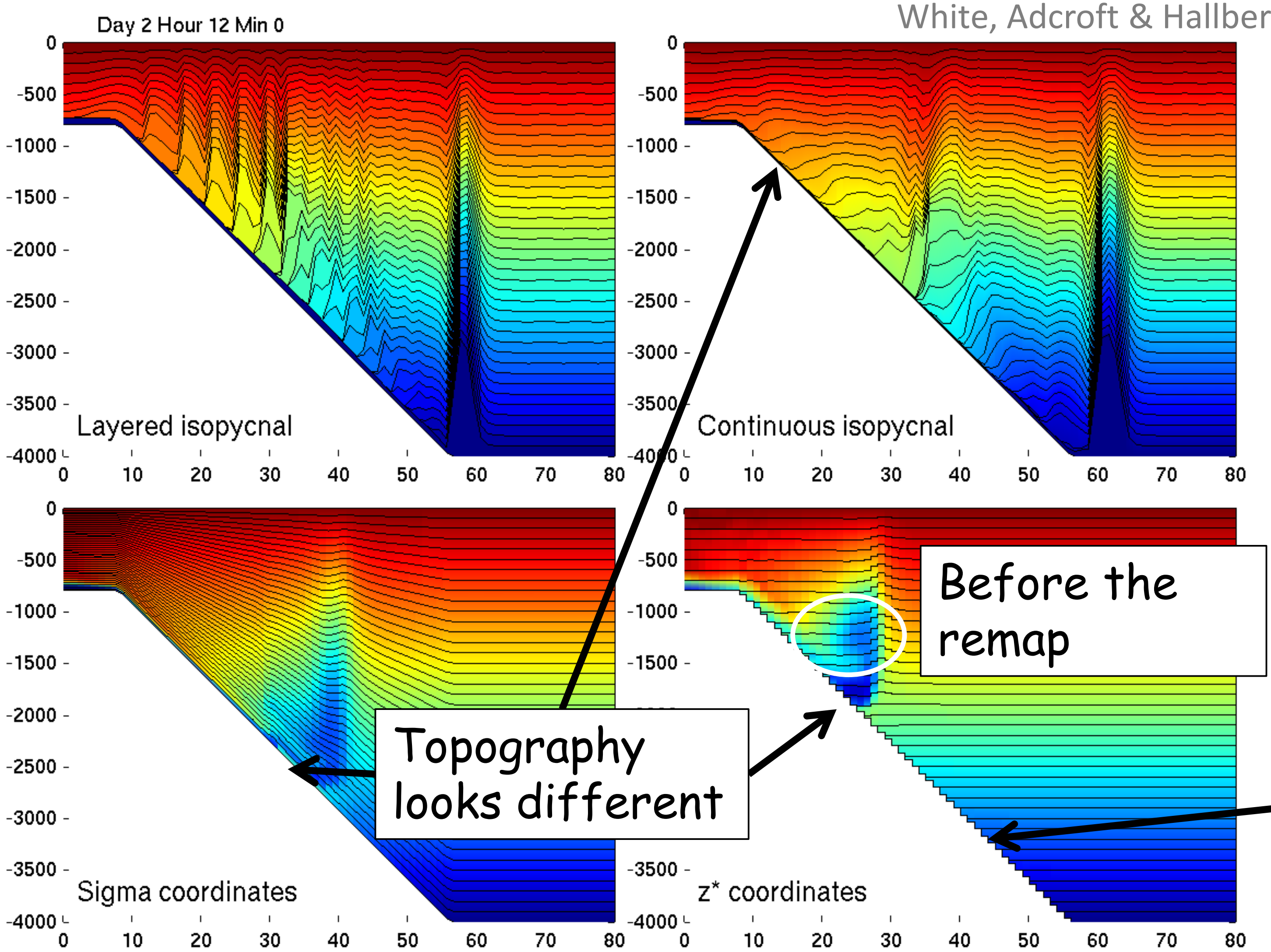


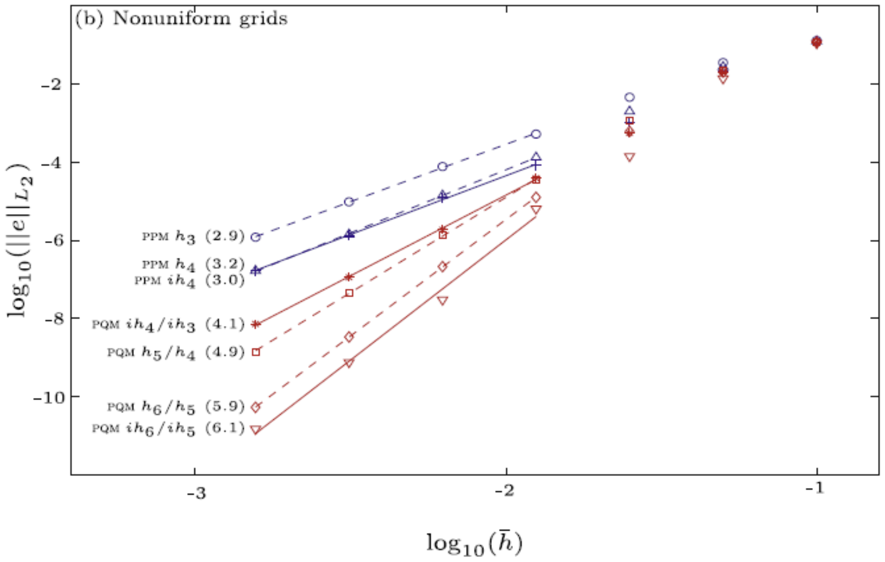
- Long debate about appropriate vertical coordinate
  - Now fairly well understood
- Spurious mixing:
  - grid Reynolds number + …
  - long time scale questions



A.L.E. allows you to work with any coordinate

White, Adcroft & Hallberg, JCP 2009





Very thin layers

Hirt et al. , 1974

Two basic algorithms

Eulerian

$$\partial_z p = -g\rho(z, S^n, \theta^n)$$

$$v_h^{n+1} = v_h^n + \Delta t\left(-\frac{1}{\rho_o}\nabla_z p + \cdots\right)$$

$$\partial_z w = -\nabla \cdot v_h^{n+1}$$

$$\theta^{n+1} = \theta^n - \Delta t\begin{bmatrix}\nabla \cdot (v_h^{n+1}\theta^n) + \\ \partial_z(w\theta^n) + \cdots\end{bmatrix}$$

$$\boxed{\frac{\Delta t w}{\Delta z} < 1}$$

A.L.E. (flavor 1)

$$\partial_z p = -g\rho(z, S^n, \theta^n)$$

$$v_h^{n+1} = v_h^n + \Delta t\left(-\frac{1}{\rho_o}\nabla_z p + \cdots\right)$$

$$\delta_k(w^* + w_g) = -\nabla \cdot h^n v_h^{n+1}$$

$$h^{n+1} = h^n + \Delta t \delta_k(w_g)$$

$$h^{n+1}\theta^{n+1} = h^n\theta^n - \Delta t\begin{bmatrix}\nabla \cdot (h^n v_h^{n+1}\theta^n) \\ +\delta_k(w^*\theta^n) + \cdots\end{bmatrix}$$

$$\boxed{\frac{\Delta t w^*}{\Delta z} < 1}$$

$$w^* = w - w_g$$

Leclair & Madec, 2011, use this form

A.L.E. (flavor 2)

$$\partial_z p = -g\rho(z, S^n, \theta^n)$$

$$v_h^\dagger = v_h^n + \Delta t\left(-\frac{1}{\rho_o}\nabla_z p + \cdots\right)$$

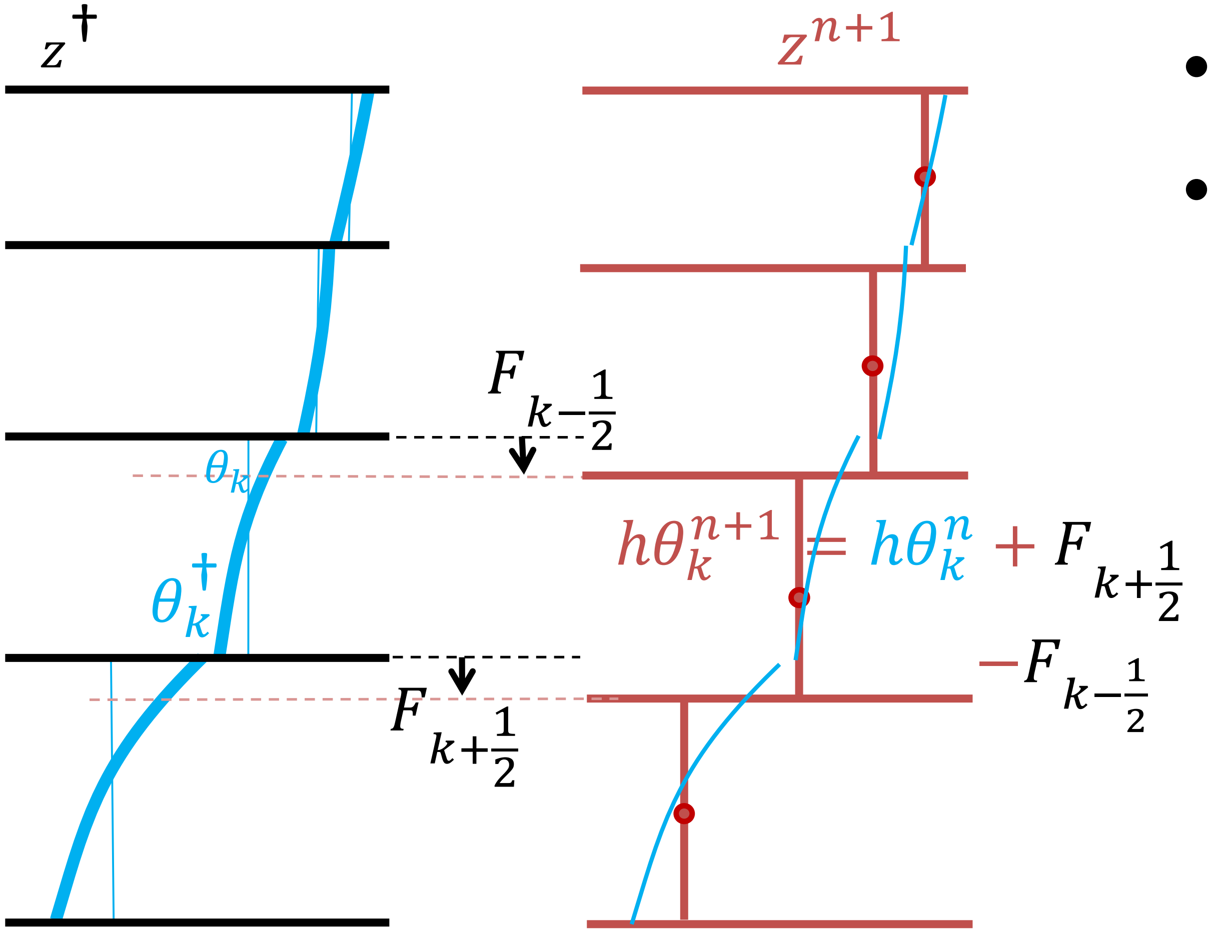
$$h^\dagger = h^n - \Delta t\nabla \cdot (h^n v_h^\dagger)$$

$$h^{n+1} \leftarrow \delta_k Z\left(z^\dagger\right)$$

$$\theta^{n+1} = \theta^\dagger\left(Z(z^\dagger)\right)$$

Bleck, 2002

Remapping styles: flux form small CFL



- Flux-form → conservative
- CFL<1 local stencil
- Stencil is "known"

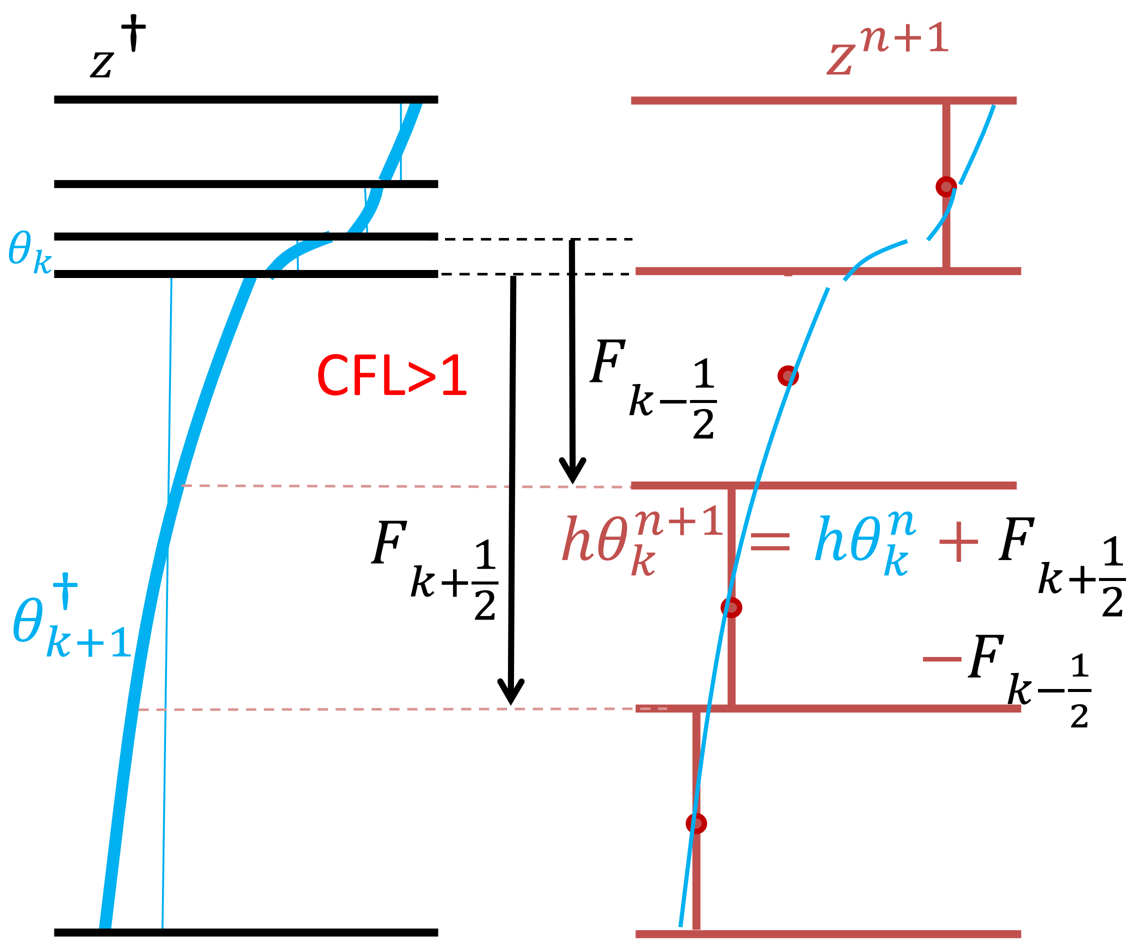
e.g. The kernel

$$F_{k+\frac{1}{2}} = \max\left(0, \Delta t w_{k-\frac{1}{2}}\right)\theta_{k+1} + \min\left(0, \Delta t w_{k-\frac{1}{2}}\right)\theta_k$$

is vectorizable in horizontal

$$F[:,K] = \max(0, w[:,K])\,\theta[:,k+1] + \min(0, w[:,K])\,\theta[:,k]$$

Remapping styles: flux form for arbitrary CFL

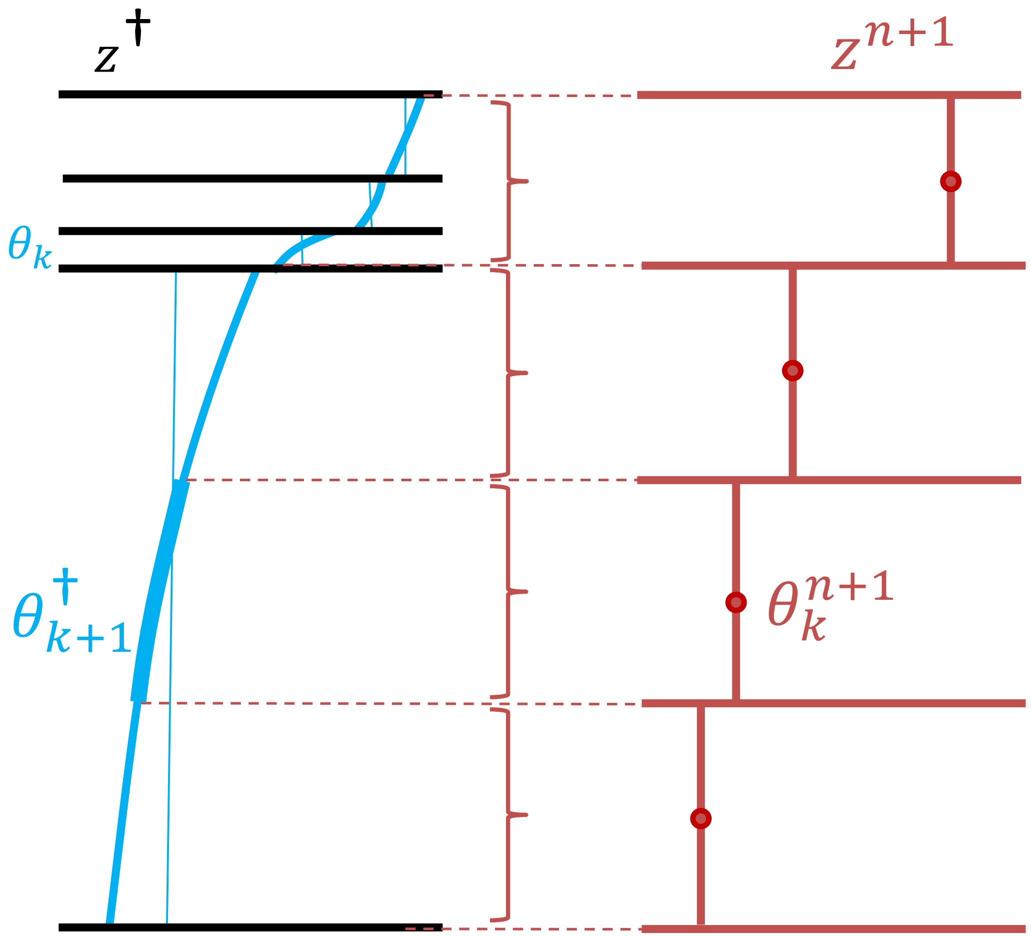


- Flux-form → conservative
- CFL>1 often evaluates **residual difference of large numbers**
  - Potentially inaccurate
- Stencil can be full column:

$$F_{k+\frac{1}{2}} = \begin{cases} \sum_{k}^{m-1} h_l \theta_l + \Delta t w_{k-\frac{1}{2}} \theta_m & w_{k-\frac{1}{2}} > 0 \\ \sum_{m+1}^{k-1} h_l \theta_l + \Delta t w_{k-\frac{1}{2}} \theta_m & w_{k-\frac{1}{2}} < 0 \end{cases}$$

  - Nested loops – eek!

Remapping styles: "projection"

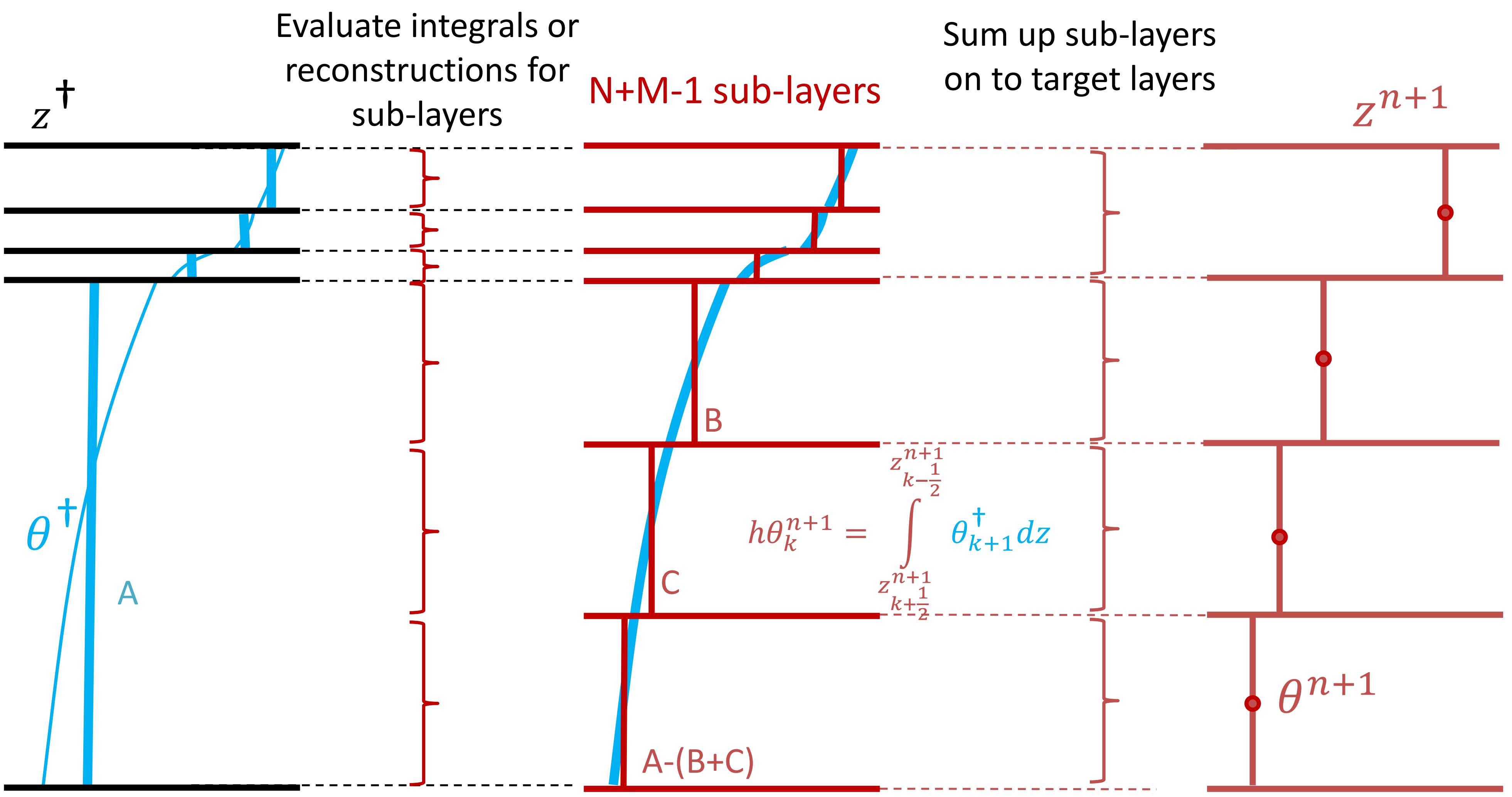


- Accuracy independent of CFL
  - No residual of differences
- Stencil can still be full column

$$h\theta_k^{n+1} = \int_{z_{k+\frac{1}{2}}^{n+1}}^{z_{k-\frac{1}{2}}^{n+1}} \theta_{k+1}^{\dagger}\, dz$$

- Conservation is less accurate
  - No equal/opposite flux terms

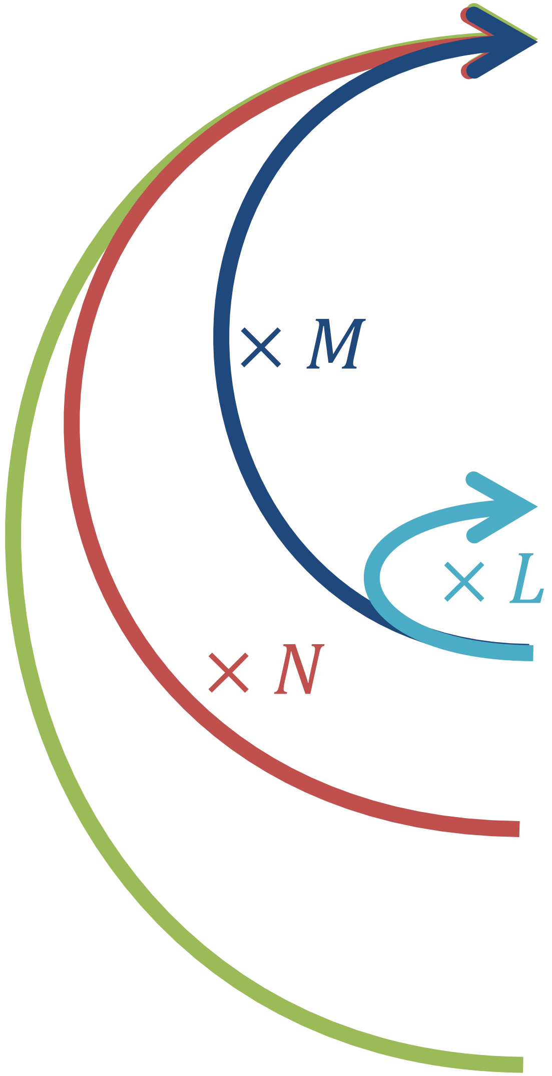
How we implement remapping in MOM6



- **No nested loops but there are now extra branches**

Hallberg and Adcroft, in prep.

Subcycling to minimize compute per step



$$\delta_k p = -\rho(z, S^n, \theta^n)\delta_k \Phi$$

$$v_h^{m+1} = v_h^m + \frac{1}{M}\frac{\Delta t}{\rho_o}\left(-\nabla_r p - \rho \nabla_r \Phi + \cdots\right)$$

$$h^{m+1} = h^m - \frac{1}{M}\Delta t \nabla_r \cdot \left(h^m v_h^{m+1}\right)$$

Internal gravity waves $\frac{\Delta t c_g}{\Delta x} < 1$

$$U^{l+1} = U^l + \frac{1}{L}\Delta t\left(-\nabla \eta^l + \cdots\right)$$

$$\eta^{l+1} = \eta^m - \frac{1}{L}\Delta t \nabla_r \cdot \left(H U^{l+1}\right)$$

Barotropic gravity waves

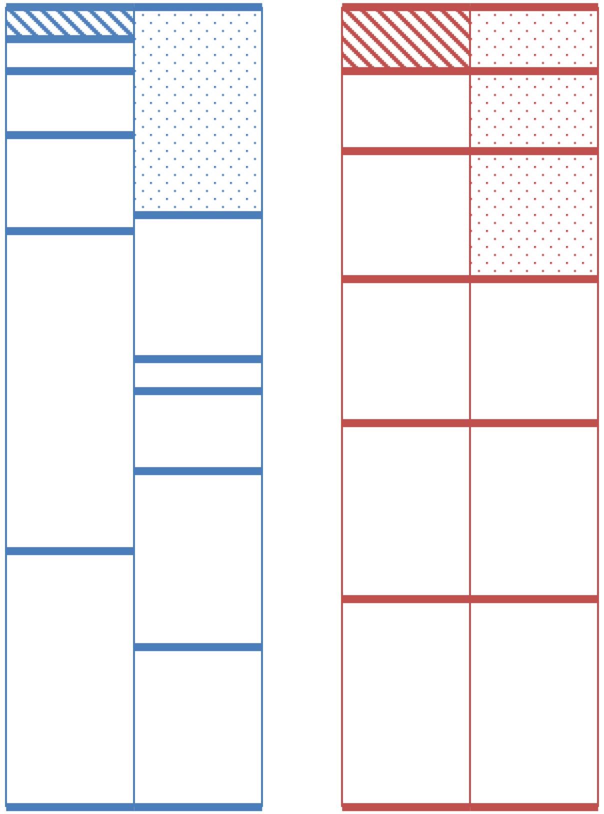
$$h^* \theta^* = h^n \theta^n - M\Delta t \left[\nabla \cdot \left(\sum_{m=1}^{M} h^m v_h^{m+1} \theta^n\right)\right]$$

$\frac{\mathrm{M}\Delta t u_h}{\Delta x} < 1$

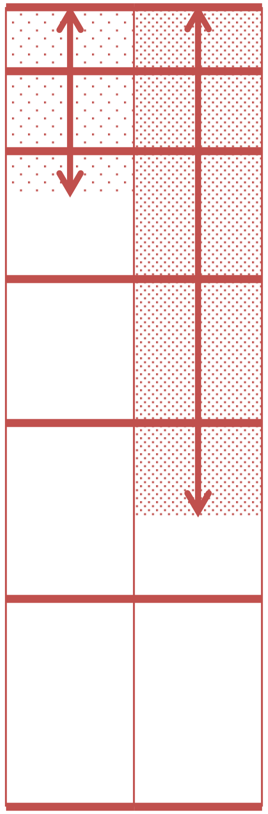
$$h^{n+1} \leftarrow \delta_k Z(z^*)\,;\, \theta^{n+1} = \theta^*(Z(z^*))\,;\, \ldots$$

Non SIMD aspect of GC + column physics

- Column-to-column variations in source/target grid



- Some parameterizations have built-in non-linear non-local solvers



What about TPUs (ML)?

- All discretizations ultimately look like a [sparse] matrix multiply
  - Challenge for our algorithm is construction of the matrix in each column is very “branchy”
    - Different pattern of addressing in each column
    - Changes every step
- Could we delegate to ML?
  - I have not tried this but willing to entertain
  - However, ML is inexact

Accuracy

- Sea-level calculation:

$$\boldsymbol{\eta} = -D + \sum_k h_k$$

$$\mathrm{err}(\boldsymbol{\eta}) \sim \boldsymbol{\varepsilon}\, D$$

- $\boldsymbol{\varepsilon}_{16\text{-bit}}=2^{-11}$, $\mathrm{err}(\boldsymbol{\eta})_{16\text{-bit}} \sim 3.4$ m
- $\boldsymbol{\varepsilon}_{32\text{-bit}}=2^{-24}$, $\mathrm{err}(\boldsymbol{\eta})_{32\text{-bit}} \sim 4\times10^{-4}$ m (0.4 mm)
- $\boldsymbol{\varepsilon}_{64\text{-bit}}=2^{-53}$, $\mathrm{err}(\boldsymbol{\eta})_{64\text{-bit}} \sim 8\times10^{-13}$ m (.008 Å)
- Context of evolving sea-level:
  - Sea-level rise ~ .3 m/Cy or 1×10^{-5} m/day
  - Tides ~ 10 m / day

My concerns

- Algorithm developments have yielded great gains
  - Challenging to implement on existing SIMD tech
- New hardware still doesn't address old problems
  - Barotropic solve likely to always be a performance barrier unless model fits on one chip
- Inexact math of ML is a reality
  - Potential advantages where stochasticity is appropriate
  - Major problem for algorithms that rely on consistency