Machine-Learned Preconditioners for Linear Solvers in Geophysical Fluid Flows

Jan Ackmann[1], Peter Dueben[2],
Piotr Smolarkiewicz[3], Tim Palmer[1]

[1] *University of Oxford, Oxford, UK*
[2] *ECMWF, Reading, UK*
[3] *NCAR*, Boulder, USA

UNIVERSITY OF OXFORD

European Research Council
Established by the European Commission

Semi-implicit Dynamical core

Background:

- Semi-implicit timestepping has become popular approach
- Part of the equations of motion performed in explicit fashion, fast waves solved implicitly
  → Large model timestep!
- But: Expensive Implicit height/pressure solve each timestep

Semi-implicit Dynamical Core

What does this implicit height/pressure solve look like?

$$\boldsymbol{A\Phi} \quad = \boldsymbol{b}$$

- Solved for height/pressure $\boldsymbol{\Phi}$
- Matrix $\boldsymbol{A}$ is square, pos. definite, non-symmetric

→ Problem has unique solution $\boldsymbol{\Phi}_*$

How do we solve this linear problem?

- Direct inversion of A impossible (dimension of $\boldsymbol{\Phi}$ is $O(10^9)$)
- Iterative linear solver is used (here, Krylov subspace method)
- Solver iteratively reduces the residual error $\boldsymbol{r_i} \coloneqq \boldsymbol{A\Phi_i} - \boldsymbol{b}$
- If residual error $\boldsymbol{r_i}$ ‘small enough’, set $\boldsymbol{\Phi^{n+1}} = \boldsymbol{\Phi_i}$

Motivation

Why focus on the Preconditioner for machine learning?

- Crucial component of a computationally efficient linear solver, but also computationally expensive
- Process that performs approximate inverse using incomplete information about the problem
  →uncertain by nature
- Machine learning application would be incorporated into the strong fundamentals of the linear solver
  →counters potential robustness issues coming with ML

Why is a preconditioner needed?

Problem for linear solvers:

- Condition number of matrix $\boldsymbol{A}$ determines convergence rate
- Given by squared ratio of largest to shortest wavelength

→ Very large condition numbers in NWP

→ Standard Iterative solver might not converge at all!

Why is a preconditioner needed?

Problem for iterative solvers:

- Condition number of matrix $\boldsymbol{A}$ determines convergence rate
- Given by squared ratio of largest to shortest wavelength

→ Very large condition numbers in NWP ~ O(10^{10})

→ Standard Iterative solvers would not converge at all!

Solution: Use a Preconditioner

- Find matrix $\widetilde{\boldsymbol{A}}$, such that the condition number of $\widetilde{\boldsymbol{A}}^{-1}\boldsymbol{A}$ is small
- Mathematically the problem becomes: $\widetilde{\boldsymbol{A}}^{-1}\boldsymbol{A}\boldsymbol{\Phi} \quad = \widetilde{\boldsymbol{A}}^{-1}\boldsymbol{b}$

→ Inversion needs to be done each solver iteration

→ Inversion needs to be cheap

Coming up with good Preconditioners is hard!

Idea: Machine-Learned Preconditioner?

What is a preconditioner supposed to do?

What would happen for the 'optimal' choice $\tilde{\boldsymbol{A}}$=$\boldsymbol{A}$…

$$\tilde{\boldsymbol{A}}^{-\mathbf{1}}(\boldsymbol{r}_{\mathbf{0}}) = \cdots = \boldsymbol{\Phi}^{\boldsymbol{n}} - \boldsymbol{\Phi}^{\boldsymbol{n}+\mathbf{1}} =: \Delta\boldsymbol{\Phi}$$

→ The preconditioner maps residuals $\boldsymbol{r}_{\mathbf{0}}$ to the increment in fluid thickness/pressure $\Delta\boldsymbol{\Phi}$

Idea: Machine-Learned Preconditioner?

What is a preconditioner supposed to do?

Let's see what happens for the 'optimal' choice $\widetilde{\boldsymbol{A}}=\boldsymbol{A}$…

$$\widetilde{\boldsymbol{A}}^{-1}(\boldsymbol{r}_0) = \cdots = \boldsymbol{\Phi}^n - \boldsymbol{\Phi}^{n+1} =: \Delta\boldsymbol{\Phi}$$

→ The preconditioner maps residuals $\boldsymbol{r}_0$ to the increment in fluid thickness/pressure $\Delta\boldsymbol{\Phi}$

How could a machine-learned preconditioner be set up?

- **Idea:** Train a neural network to perform the mapping $\widetilde{\boldsymbol{A}}^{-1}$.
  - **Inputs:** Local Stencils 'neighbouring values' of
    - Matrix coefficients of $\boldsymbol{A}$
    - Residual values $\boldsymbol{r}_0$
  - **Output:** Grid-point value of height/pressure increment $\Delta\boldsymbol{\Phi}$

Shallow-Water Model

Discretization:

- MPDATA advection scheme for Momenta (eulerian, 2nd Order)
- Generalized Conjugated Residual Method
- Lat-Lon Grid (study behaviour near grid singularities)

Semi-Implicit(SI) Richardson Preconditioner:

- Start with $\widetilde{\boldsymbol{A}} = \boldsymbol{A}$
- Omit all cross-derivative terms
- Split $\widetilde{\boldsymbol{A}}$ into Zonal, Helmholtz and Meridional part, $\widetilde{\boldsymbol{A}}^Z$, $\widetilde{\boldsymbol{A}}^H$, $\widetilde{\boldsymbol{A}}^M$

- Perform Richardson iteration to obtain $\Delta\boldsymbol{\Phi} \approx \widetilde{\boldsymbol{A}}^{-\mathbf{1}}(r)$:

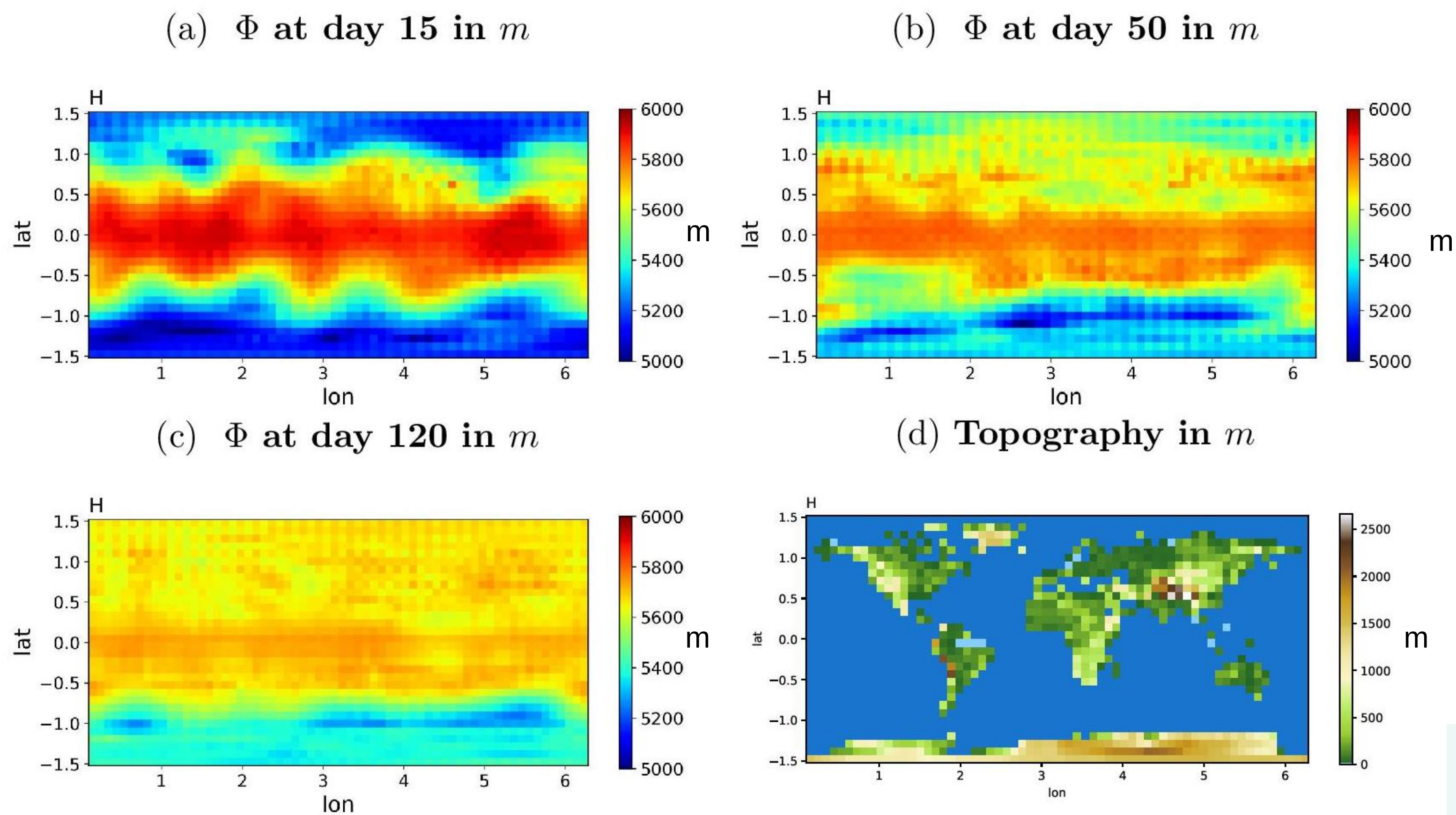
$$\left[\mathrm{I} - \delta t\widetilde{\boldsymbol{A}}^Z - \delta t\widetilde{\boldsymbol{A}}^H\right]\Delta\widetilde{\Phi}^{n+1} = \Delta\widetilde{\Phi}^n + \delta t\left[\widetilde{\boldsymbol{A}}^M\Delta\widetilde{\Phi}^n - r\right]$$

→Zonal part implicit because that is where the stiffness of $\boldsymbol{A}$ resides in

Model Setup 1

Shallow-water Test-case:

- Zonal geostrophic flow around scaled-down, Real-earth orography
- 5.2° model resolution
- 120 days of integration time



Neural Network Setups

Training the Neural networks:

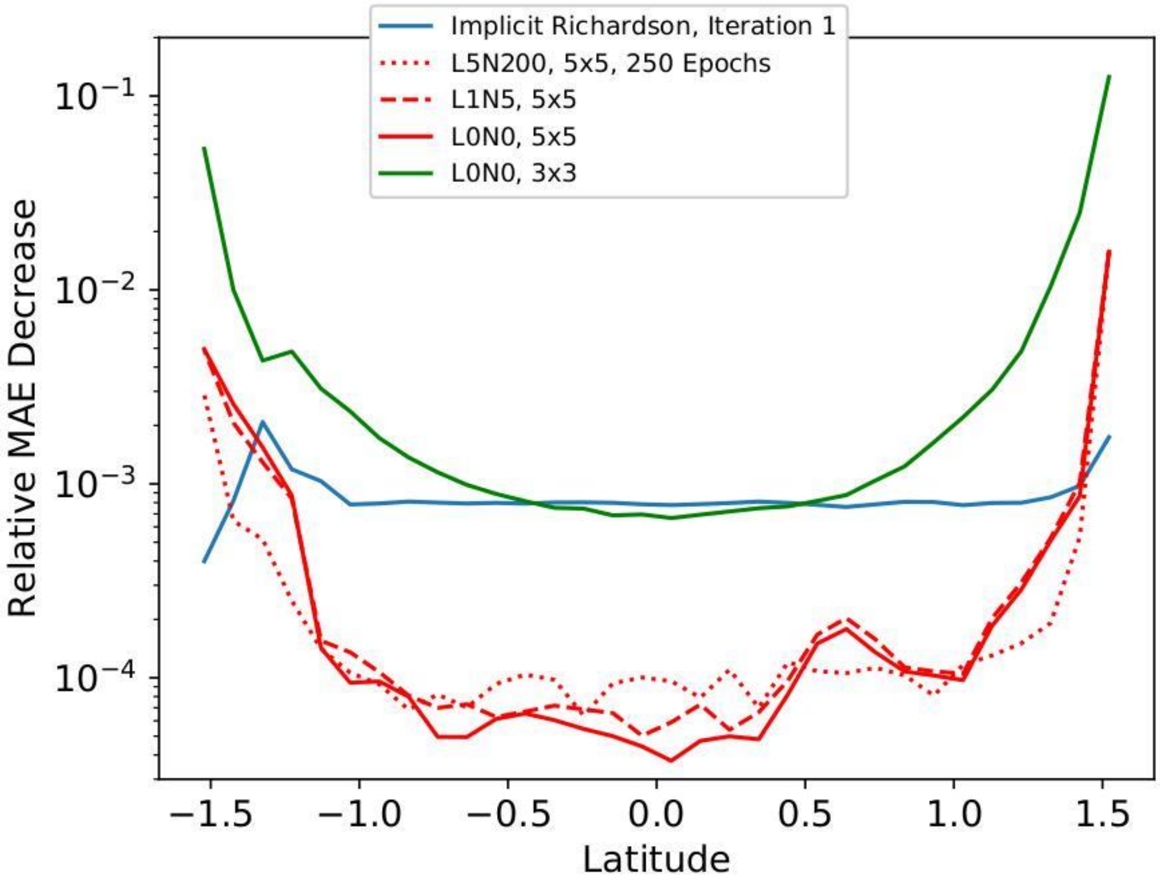
Architecture:

- Several neural network sizes: linear (L0N0), L1N5, L5N200
- ReLU activation function

Data:

- Trained on data from the first solver iteration (days 15-120)
- One neural network per Latitude
- different input stencil sizes (3x3 or 5x5)
- **Input:** **7** input Stencils = **6** Matrix coefficient fields define $\boldsymbol{A}$
  + **1** residual error field $\boldsymbol{r_0}$
- **Output:** single grid-point value of increment $\Delta\boldsymbol{\Phi}$
- →$\mathbf{1,5 * 10^6}$ training examples, $\mathbf{5 * 10^5}$ validation examples

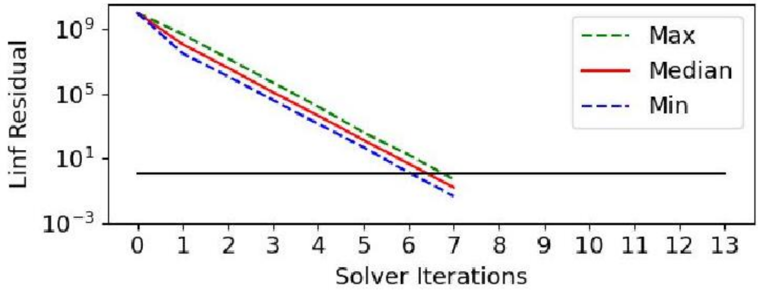
Mean Absolute Value for $\frac{\Delta\widetilde{\Phi}-\Delta\Phi}{\Delta\Phi}$



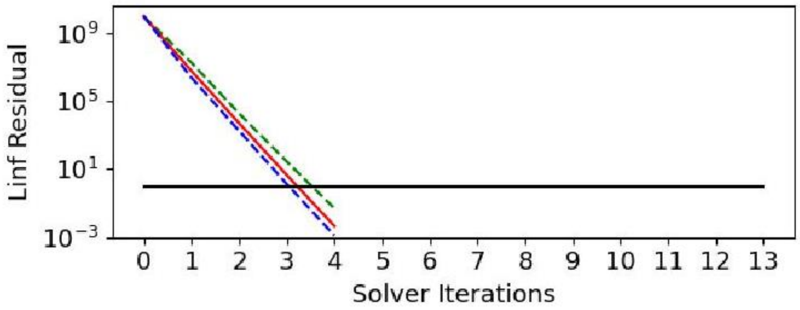
- Different complexity preconditioners on first solver iteration
- Implicit Richardson powerful preconditioner for this type of SW model
- Linear regression model (L0N0) with 5x5 stencil performs best

Convergence Results (Days 15-120)

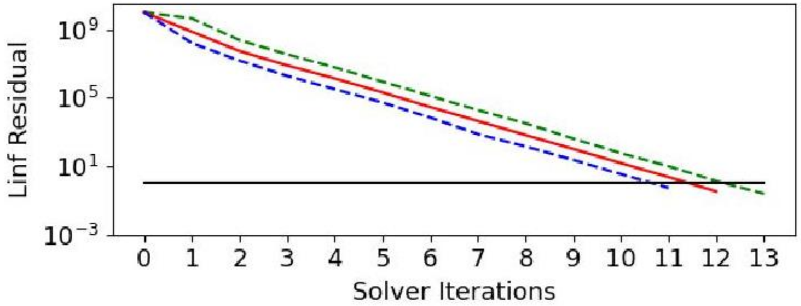
Linear (L0N0) with 5x5 stencil



SI-Richardson Preconditioner



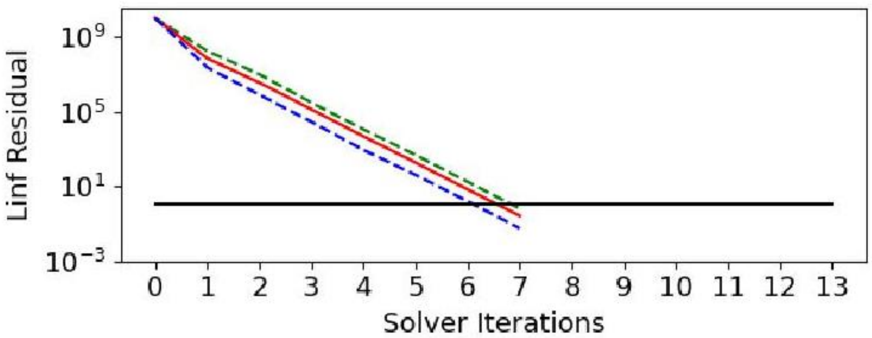
No Preconditioner



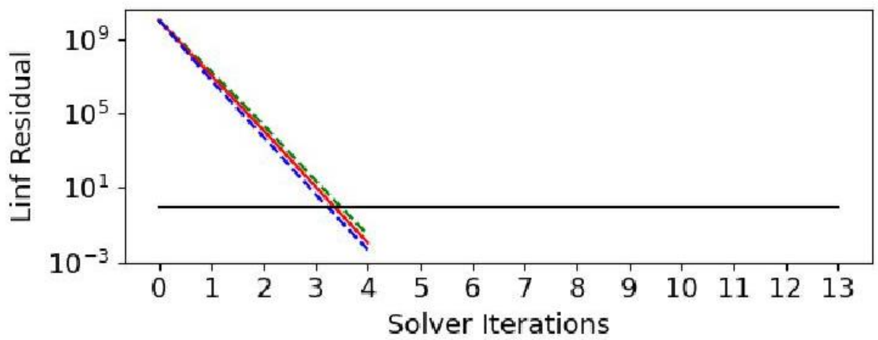
- Simulations run from perturbed initial conditions
- Robust convergence with the machine-learned Preconditioner
- Convergence rate consistently high for all solver iterations
- At no point during training was data from later iterations seen!

Convergence Results (Days 1-14)

Linear (L0N0) with 5x5 stencil



SI-Richardson Preconditioner



Spin-up phase with shocks from model initialization:

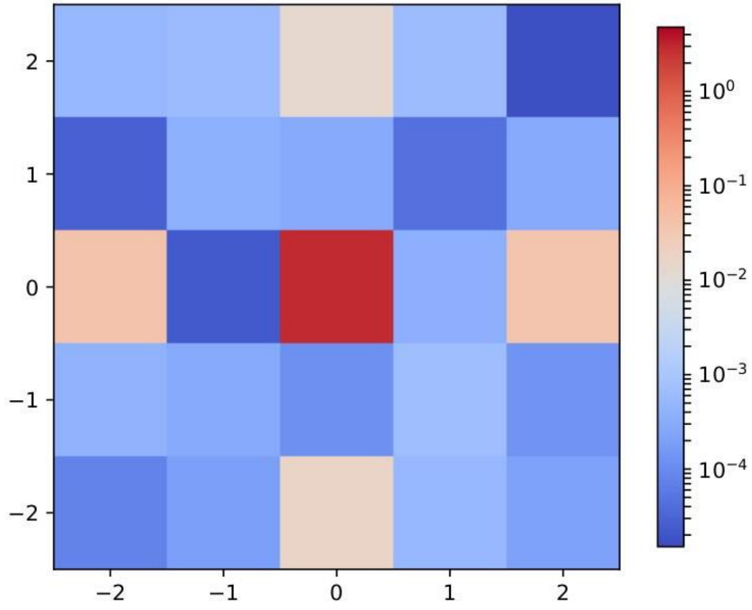
- The initial 14 days of the simulation were not part of the training set
- Yet, Machine-learned Preconditioner performs robustly

In summary:

→ The machine-learned preconditioner has learned some general rule about predicting fluid thickness increments $\Delta\boldsymbol{\Phi}$

Interpretability of the ML Preconditioner

Relative Contribution of r_i



- Relative Contribution of the 6 Matrix Coefficient fields negligible
- r_i shows imprint of A, i.e. stencil of spatial discretization
- Only r_i contribution needs to be computed each solver iteration

  → ML preconditioner is cheaper per application than SI-Richardson!

Model Setup 2

Increase Complexity…

- Increase model resolution to 0.7°

  → zonal grid spacing down to 400m at the Poles

- Real-Earth Orography (steeper orography gradients and smaller fluid thicknesses)

  →lead to more asymmetric $\boldsymbol{A}$ and more non-linear flow

- Flow reinitialized using anomalies every 14 simulation days

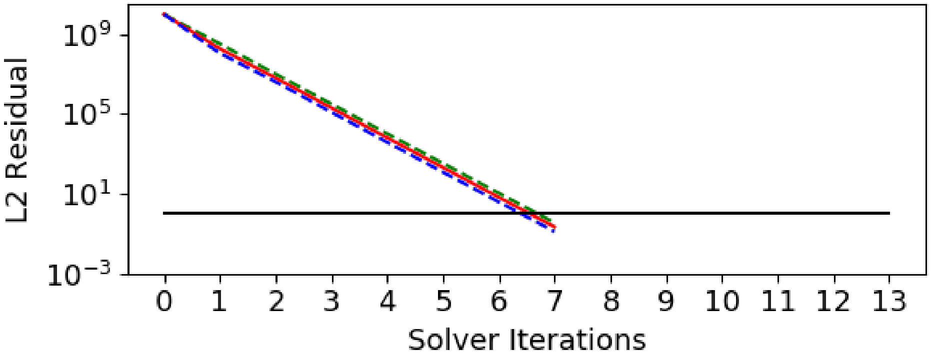
  →Potential energy restoring; Flow does not decay

- Introduce polar absorbers

  →enables dt=200s; advective Courant Number close to numerical limit of 1

Reference Solver Convergence Results

SI-Richardson Preconditioner

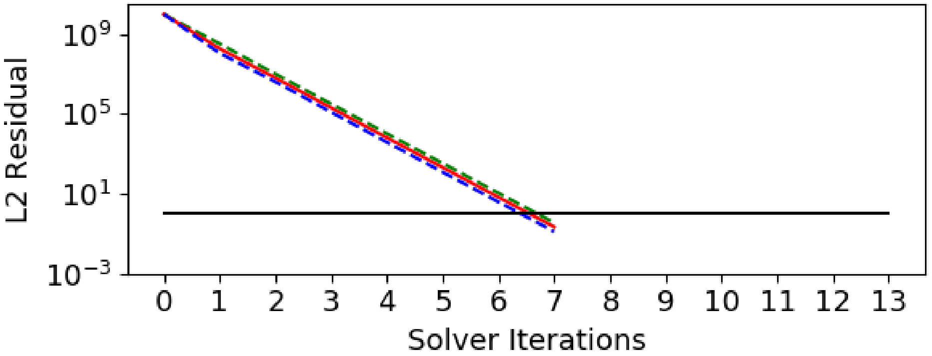


Much harder linear problem…

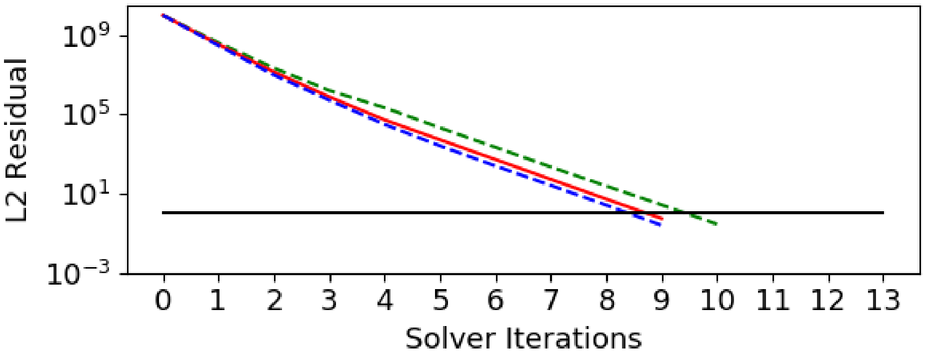
- Convergence rate for days 1-120
- For comparison:
  - No preconditioener ~540 iterations
  - Diagonal preconditioner ~490 iterations
  - explicit Richardson ~300 iterations
- SI-Richardson Preconditioner yields enormous Speed-up=35

Solver Convergence Results (I)

SI-Richardson Preconditioner



Hybrid Preconditioner



Hybrid Preconditioner

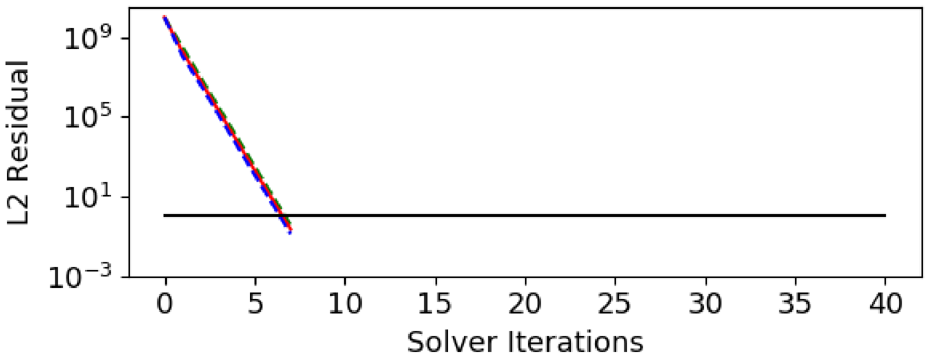
- 70S to 70N: Linear with 5x5 stencil
- Remaining: SI-Richardson Preconditioner

- Expected Solver Speed-up of factor 2 compared to pure SI-Richardson Preconditioner

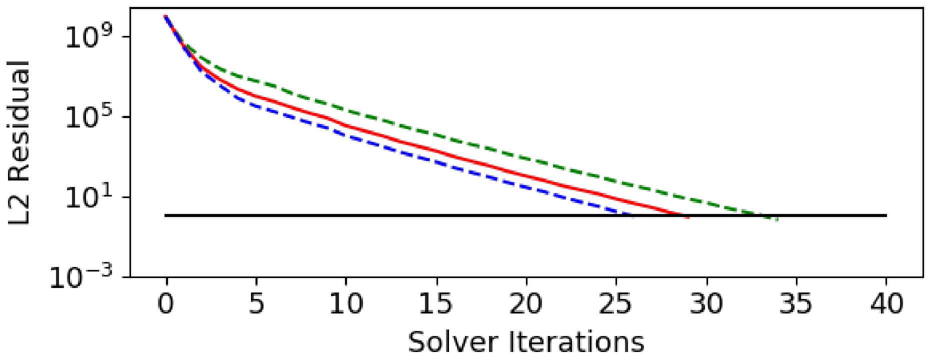
- Linear with 5x5 stencil unusable near singularities
- Also, no improvement found when using larger neural networks

Solver Convergence Results (II)

SI-Richardson Preconditioner



Hybrid Preconditioner



Larger input stencils?

- Hybrid preconditioner setup:
  - 70S - 70N: Linear with 5x5 stencil
  - 86S-70S and 70N-86N: Linear with 60x5 stencil
  - Remaining: SI-Richardson Preconditioner
- Training and Validation: 10x larger data sets; 1st-5th solver iteration
- Not perfect yet, but convergence rate for first iterations indicates further improvements might be possible.

Conclusions

The preconditioning step in linear solvers of weather and climate models can be performed using machine learning.

For our test-case, performance is comparable to reference preconditioner:

- Hybrid preconditioner more computationally efficient
- Area near grid singularities challenging, work is ongoing

Due to the low complexity, the preconditioner can be interpreted. Possibly, improve analytical preconditioners.

https://arxiv.org/pdf/2010.02866.pdf

Next Step: Investigate machine-learned preconditioners for ECMWF’s IFS-FVM