Machine-Learning-Accelerated Multigrid Setup for Lattice QCD Dirac Solves

Simon Pfahler Daniel Knüttel Riccardo Costantini Christoph Lehner Tilo Wettig

Department of Physics, University of Regensburg

Daniel Knüttel

Riccardo Costantini

Christoph Lehner

Tilo Wettig

Overview

- Gauge-field generation in Lattice QCD is expensive, mostly because of the required solves of the Dirac equation
- Algebraic multigrid[1] (MG) is the state-of-the-art preconditioner
- MG needs a costly setup → only limited benefit for gauge-field generation

→ Goal of this project:

Accelerate the MG setup by "moving the setup along the Markov-chain trajectory"

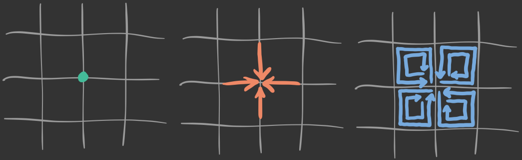
[1] Osborn et al., PoS Lattice 2010

Preconditioning

- Linear equation $D\psi = \varphi$ with $\varphi \in \mathbb{C}^{V\times 4\times 3}$
- Solve via iterative methods, e.g., GMRES
  $\longrightarrow$ Convergence depends on spectrum
- Preconditioners can change the spectrum to obtain better convergence:

$$D M \underbrace{M^{-1}\psi}_{y} = \varphi$$

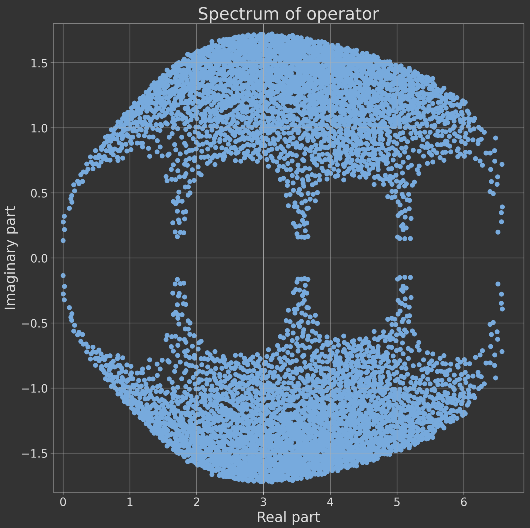
$$D = (m+4)\mathbb{1} - \sum_{\mu} \frac{\mathbb{1}-\gamma_\mu}{2} H_{-\mu} - \frac{\mathbb{1}+\gamma_\mu}{2} H_{+\mu} - \frac{c_{\text{sw}}}{4} \sum_{\mu,\nu} \sigma_{\mu\nu} F_{\mu\nu}$$



Preconditioning

- Linear equation $D\psi = \varphi$ with $\varphi \in \mathbb{C}^{V\times4\times3}$
- Solve via iterative methods, e.g., GMRES
  $\longrightarrow$ Convergence depends on spectrum
- Preconditioners can change the spectrum to obtain better convergence:

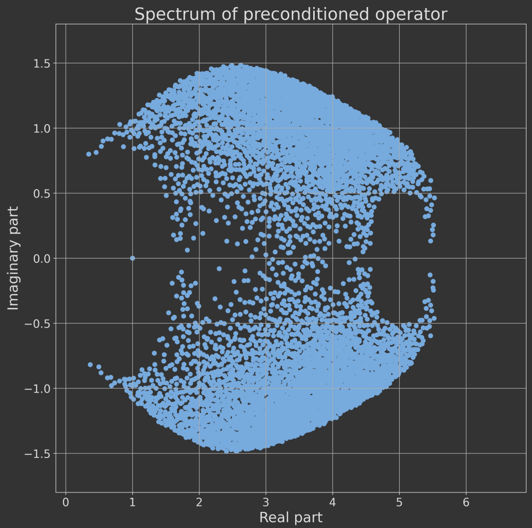
$$D M \underbrace{M^{-1}\psi}_{y} = \varphi$$



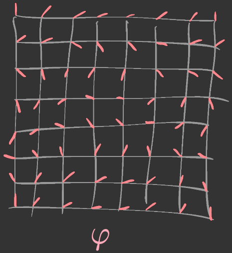
Preconditioning

- Linear equation $D\psi = \varphi$ with $\varphi \in \mathbb{C}^{V\times 4\times 3}$
- Solve via iterative methods, e.g., GMRES
  $\longrightarrow$ Convergence depends on spectrum
- Preconditioners can change the spectrum to obtain better convergence:

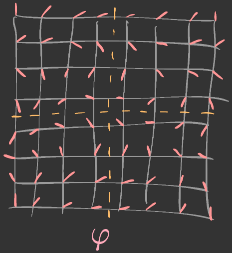
$$DM\underbrace{M^{-1}\psi}_{y} = \varphi$$



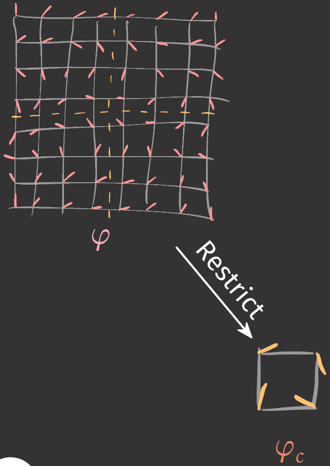
Multigrid - Overview



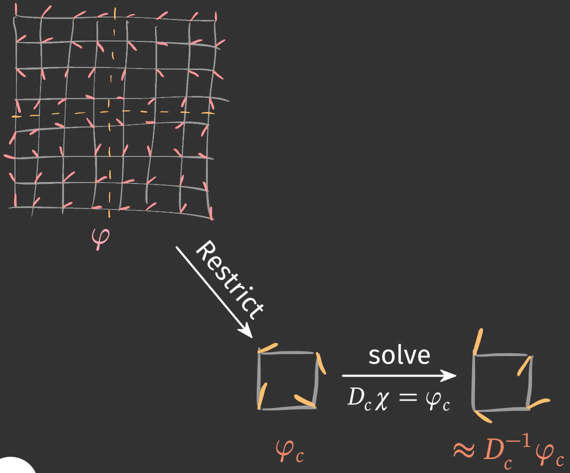
Multigrid - Overview



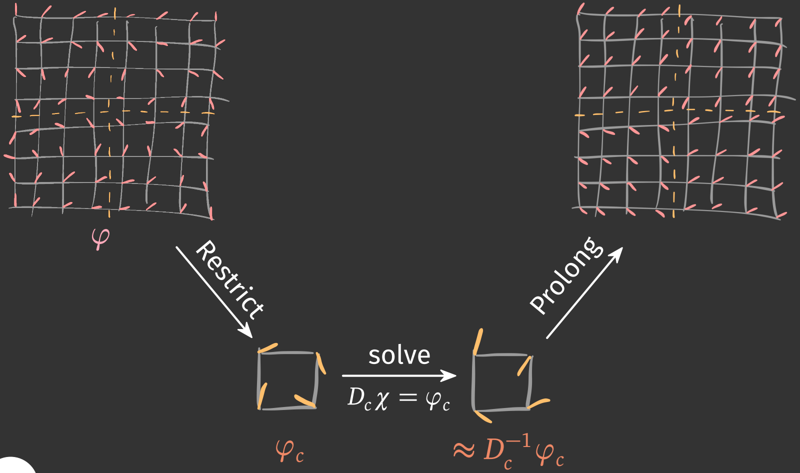
Multigrid - Overview



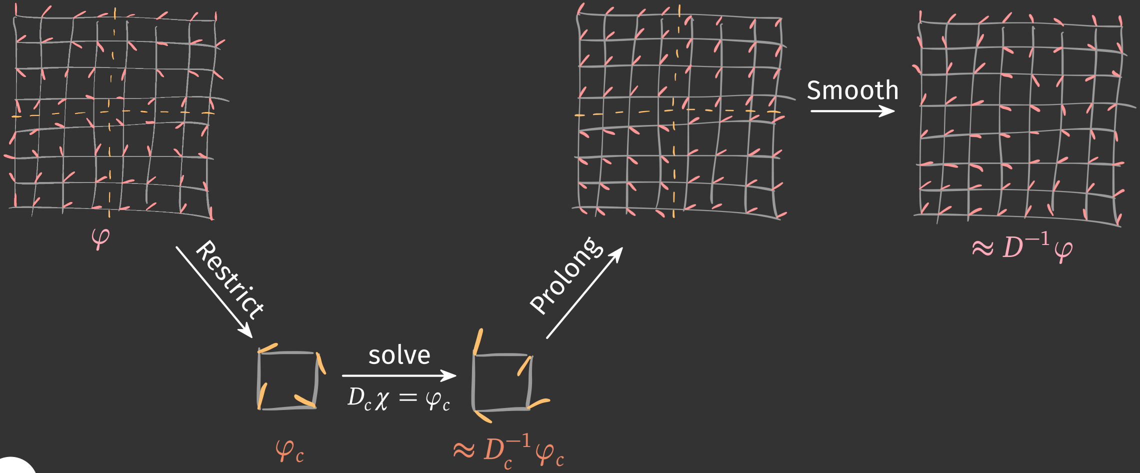
Multigrid - Overview



Multigrid - Overview



Multigrid - Overview



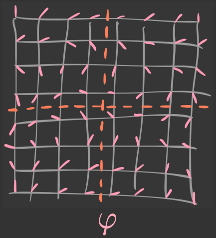
Multigrid - Restriction and Prolongation

How can we restrict φ to the coarse grid?[2,3]

- Take N low modes ψ_i on the fine grid (e.g., via $D\psi_i \approx 0$)
- Block-orthonormalize ψ_i to get $\hat{\psi}_i|_B$ on each block B
- Take scalar products

Prolongation = Restriction†
$\rightarrow$ Coarse-grid operator:

$$D_c = RDP$$



[2] Babich et al., PRL 105(2010)
[3] Lüscher, JHEP 07(2007)

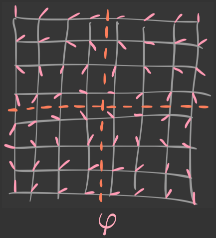
Multigrid - Restriction and Prolongation

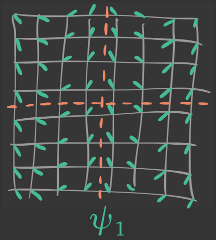
How can we restrict φ to the coarse grid?[2,3]

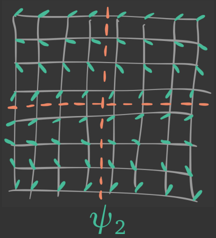
- Take N low modes ψ_i on the fine grid (e.g., via $D\psi_i \approx 0$)
- Block-orthonormalize ψ_i to get $\hat{\psi}_i|_B$ on each block B
- Take scalar products

Prolongation = Restriction†
$\rightarrow$ Coarse-grid operator:

$$D_c = RDP$$







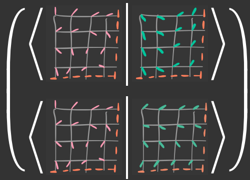
[2] Babich et al., PRL 105(2010)
[3] Lüscher, JHEP 07(2007)

Multigrid - Restriction and Prolongation

How can we restrict φ to the coarse grid?[2,3]

- Take N low modes ψ_i on the fine grid (e.g., via $D\psi_i \approx 0$)
- Block-orthonormalize ψ_i to get $\hat{\psi}_i|_B$ on each block B
- Take scalar products

$$\varphi_c(B) = \begin{pmatrix} \langle \varphi|_B | \hat{\psi}_1|_B \rangle \\ \langle \varphi|_B | \hat{\psi}_2|_B \rangle \end{pmatrix}$$

$=$ 

Prolongation = Restriction†

$\rightarrow$ Coarse-grid operator:

$$D_c = RDP$$

[2] Babich et al., PRL 105(2010)

[3] Lüscher, JHEP 07(2007)

Multigrid - Restriction and Prolongation

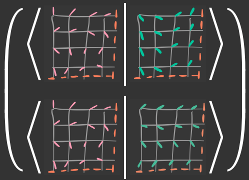
How can we restrict φ to the coarse grid?[2,3]

- Take N low modes ψ_i on the fine grid (e.g., via $D\psi_i \approx 0$)
- Block-orthonormalize ψ_i to get $\hat{\psi}_i|_B$ on each block B
- Take scalar products

Prolongation $=$ Restriction†
$\rightarrow$ Coarse-grid operator:

$$D_c = RDP$$

$$\varphi_c(B) = \begin{pmatrix} \left\langle \varphi|_B \middle| \hat{\psi}_1|_B \right\rangle \\ \left\langle \varphi|_B \middle| \hat{\psi}_2|_B \right\rangle \end{pmatrix}$$

$=$ 

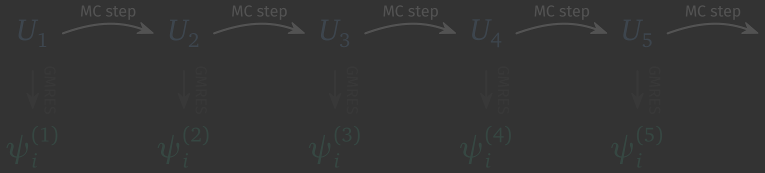
---

[2] Babich et al., PRL 105(2010)
[3] Lüscher, JHEP 07(2007)

MG setup transfer

- We can construct P and R for any set of approximate low modes ψ_i
- The solver speedup depends on the overlap of the chosen ψ_i with the low-mode space of D_U
- We typically generate configurations U in a Markov chain
- Generating the ψ_i anew for each configuration is expensive

$\rightarrow$ Move the ψ_i along the Markov chain!



MG setup transfer

- We can construct P and R for any set of approximate low modes ψ_i
- The solver speedup depends on the overlap of the chosen ψ_i with the low-mode space of D_U
- We typically generate configurations U in a Markov chain
- Generating the ψ_i anew for each configuration is expensive

$\rightarrow$ Move the ψ_i along the Markov chain!

$$\begin{tikzcd}
U_1 \arrow[r, "\text{MC step}"] \arrow[d, "\text{GMRES}"] & U_2 \arrow[r, "\text{MC step}"] \arrow[d, "\text{GMRES}"] & U_3 \arrow[r, "\text{MC step}"] \arrow[d, "\text{GMRES}"] & U_4 \arrow[r, "\text{MC step}"] \arrow[d, "\text{GMRES}"] & U_5 \arrow[r, "\text{MC step}"] \arrow[d, "\text{GMRES}"] & {} \\
\psi_i^{(1)} & \psi_i^{(2)} & \psi_i^{(3)} & \psi_i^{(4)} & \psi_i^{(5)} &
\end{tikzcd}$$

MG setup transfer

- We can construct P and R for any set of approximate low modes ψ_i
- The solver speedup depends on the overlap of the chosen ψ_i with the low-mode space of D_U
- We typically generate configurations U in a Markov chain
- Generating the ψ_i anew for each configuration is expensive

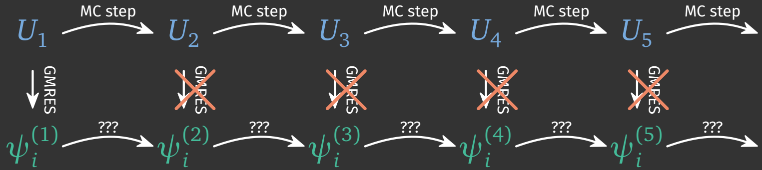
$\longrightarrow$ Move the ψ_i along the Markov chain!

$$
\begin{tikzcd}
U_1 \arrow[r, "\text{MC step}"] \arrow[d, "\text{GMRES}"] & U_2 \arrow[r, "\text{MC step}"] \arrow[d, "\text{GMRES}"] & U_3 \arrow[r, "\text{MC step}"] \arrow[d, "\text{GMRES}"] & U_4 \arrow[r, "\text{MC step}"] \arrow[d, "\text{GMRES}"] & U_5 \arrow[r, "\text{MC step}"] \arrow[d, "\text{GMRES}"] & {} \\
\psi_i^{(1)} & \psi_i^{(2)} & \psi_i^{(3)} & \psi_i^{(4)} & \psi_i^{(5)} &
\end{tikzcd}
$$

MG setup transfer

- We can construct P and R for any set of approximate low modes ψ_i
- The solver speedup depends on the overlap of the chosen ψ_i with the low-mode space of D_U
- We typically generate configurations U in a Markov chain
- Generating the ψ_i anew for each configuration is expensive

$\longrightarrow$ Move the ψ_i along the Markov chain!



Machine Learning - Building blocks

We use 2 types of layers in our model, taken from[4]

Parallel-transport (PT) layer:

$$\psi_a(x) \stackrel{PT}{=} T_{p_a}\varphi_a(x)$$

for $a = 1, \ldots, n$

Linear (L) layer:

$$\psi_b(x) \stackrel{L}{=} \sum_{a=1}^{n} W_{ba}\varphi_a(x)$$

for $a = 1, \ldots, n$ and $b = 1, \ldots, m$

Both are gauge-equivariant!

[4] Pfahler et al., PoS Lattice 2025

Machine Learning - Building blocks

We use 2 types of layers in our model, taken from[4]

Parallel-transport (PT) layer:

$$\psi_a(x) \stackrel{PT}{=} T_{p_a} \varphi_a(x)$$

for $a = 1, \ldots, n$

Linear (L) layer:

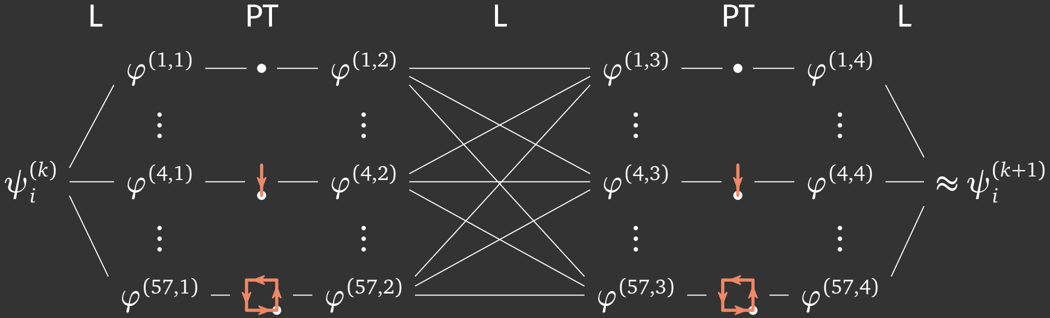
$$\psi_b(x) \stackrel{L}{=} \sum_{a=1}^{n} W_{ba} \varphi_a(x)$$

for $a = 1, \ldots, n$ and
$b = 1, \ldots, m$

Both are gauge-equivariant!

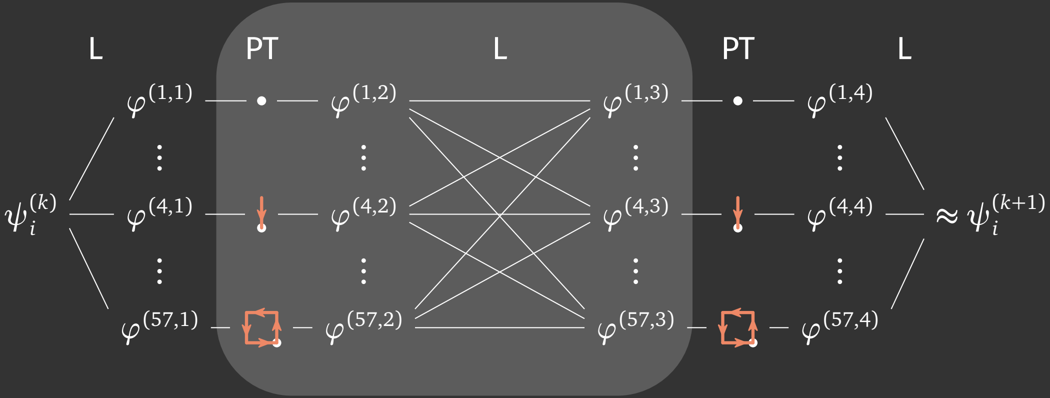
[4] Pfahler et al., PoS Lattice 2025

Network architecture



All PT layers have the paths

Network architecture



All PT layers have the paths 

Cost function for training

Aspects of our cost function:

- Average of a block-wise cost function
- On each block, calculate the overlap between the spaces spanned by the low modes of D_U and our model output

$$C = 1 - \frac{1}{\#\text{vectors} \cdot \#\text{blocks}} \sum_{\text{blocks } B} \sum_i \sum_j \left| \left\langle \hat{m}_i|_B \middle| \hat{\psi}_j^{(k+1)}|_B \right\rangle \right|^2$$

$$\hat{\psi}_j^{(k+1)}|_B = \text{standard MG setup vectors of } U_{k+1} \text{ on block } B$$

$$\hat{m}_i|_B = \text{orthonormalized model output on block } B\text{, with } \psi_i^{(k)} \text{ as model input}$$

Cost function for training

Aspects of our cost function:

- Average of a block-wise cost function
- On each block, calculate the overlap between the spaces spanned by the low modes of D_U and our model output

$$C = 1 - \frac{1}{\#\text{vectors} \cdot \#\text{blocks}} \sum_{\text{blocks } B} \sum_i \sum_j \left| \left\langle \hat{m}_i|_B \middle| \hat{\psi}_j^{(k+1)}|_B \right\rangle \right|^2$$

$\hat{\psi}_j^{(k+1)}|_B =$ standard MG setup vectors of U_{k+1} on block B

$\hat{m}_i|_B =$ orthonormalized model output on block B, with $\psi_i^{(k)}$ as model input

Results - Overview

$8^3 \times 16$ ensemble[5]:

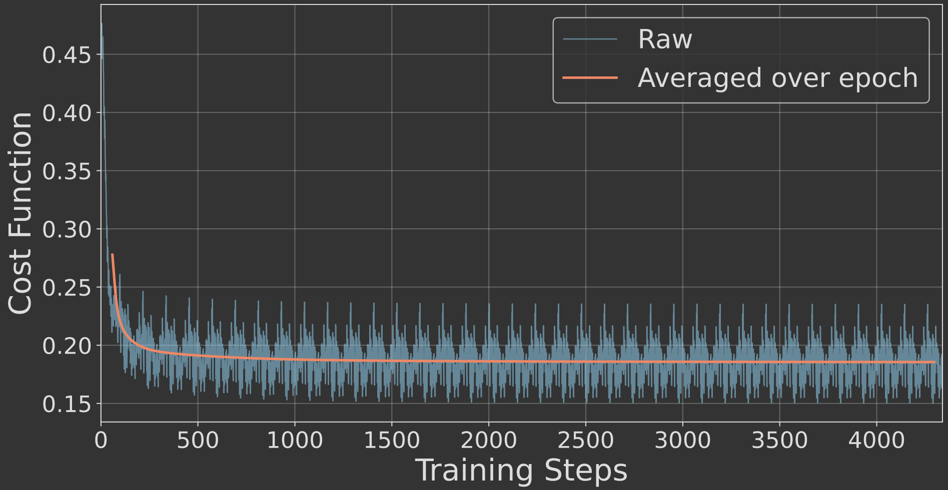
- Wilson action, $\beta = 6$
- quenched, heat bath
- Wilson-clover Dirac operator

$16^3 \times 32$ ensemble:

- Wilson action, $\beta = 6$
- quenched, HMC
- Wilson-clover Dirac operator

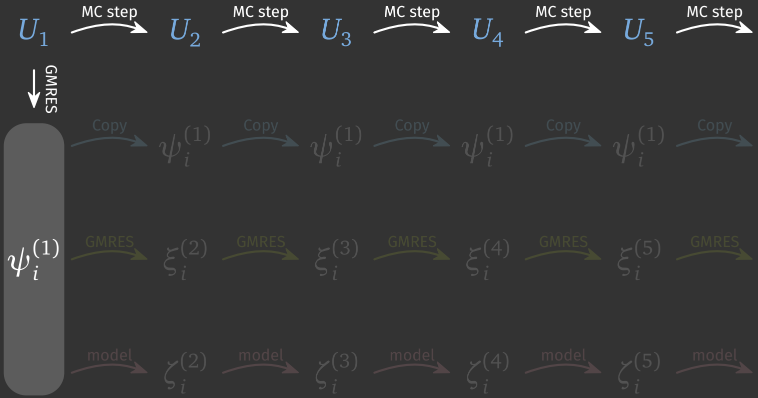
[5] Knüttel, doi.org/10.5281/zenodo.15018324

Training

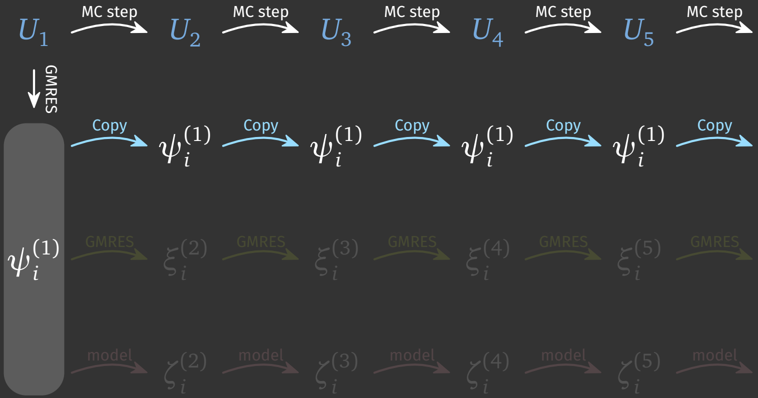


Raw
Averaged over epoch
Cost Function
0.45
0.40
0.35
0.30
0.25
0.20
0.15
0
500
1000
1500
2000
2500
3000
3500
4000
Training Steps

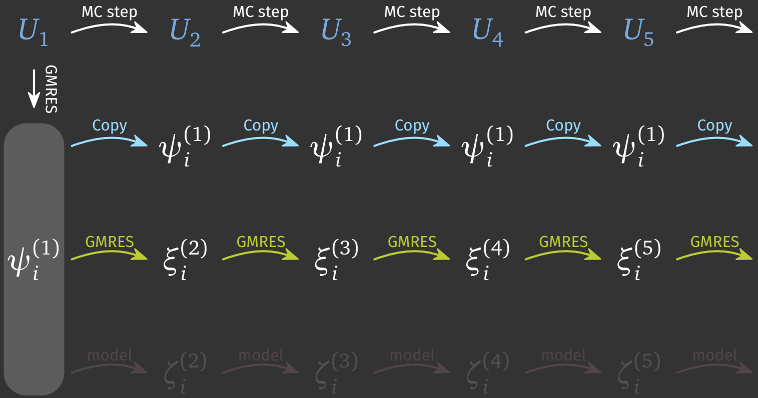
Evaluation


U_1
MC step
U_2
MC step
U_3
MC step
U_4
MC step
U_5
MC step
GMRES
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
Copy
GMRES
$\xi_i^{(2)}$
GMRES
$\xi_i^{(3)}$
GMRES
$\xi_i^{(4)}$
GMRES
$\xi_i^{(5)}$
GMRES
model
$\zeta_i^{(2)}$
model
$\zeta_i^{(3)}$
model
$\zeta_i^{(4)}$
model
$\zeta_i^{(5)}$
model

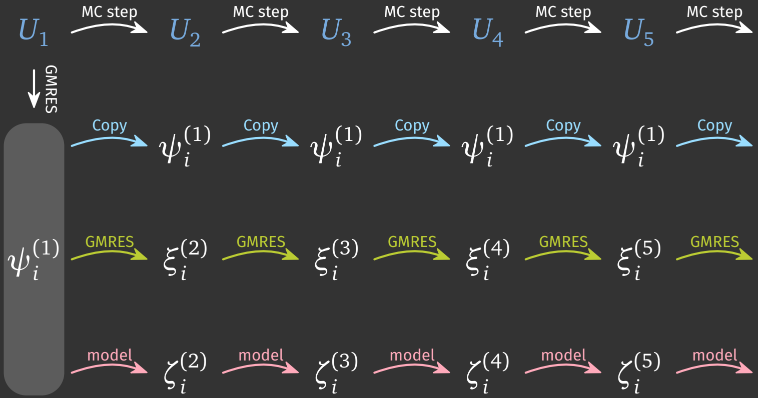
Evaluation


U_1
MC step
U_2
MC step
U_3
MC step
U_4
MC step
U_5
MC step
GMRES
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
GMRES
$\xi_i^{(2)}$
GMRES
$\xi_i^{(3)}$
GMRES
$\xi_i^{(4)}$
GMRES
$\xi_i^{(5)}$
GMRES
model
$\zeta_i^{(2)}$
model
$\zeta_i^{(3)}$
model
$\zeta_i^{(4)}$
model
$\zeta_i^{(5)}$
model

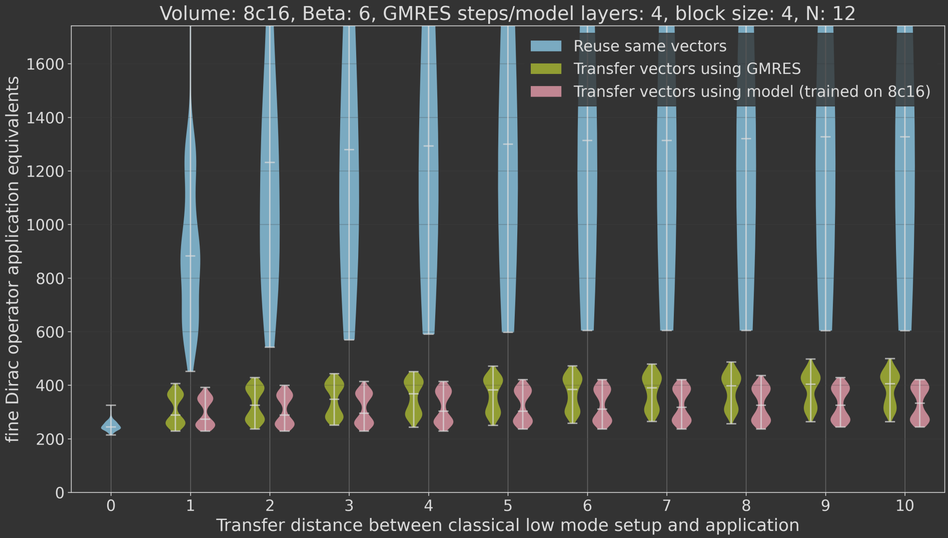
Evaluation


U_1 MC step U_2 MC step U_3 MC step U_4 MC step U_5 MC step
GMRES
Copy $\psi_i^{(1)}$ Copy $\psi_i^{(1)}$ Copy $\psi_i^{(1)}$ Copy $\psi_i^{(1)}$ Copy
$\psi_i^{(1)}$ GMRES $\xi_i^{(2)}$ GMRES $\xi_i^{(3)}$ GMRES $\xi_i^{(4)}$ GMRES $\xi_i^{(5)}$ GMRES
model $\zeta_i^{(2)}$ model $\zeta_i^{(3)}$ model $\zeta_i^{(4)}$ model $\zeta_i^{(5)}$ model

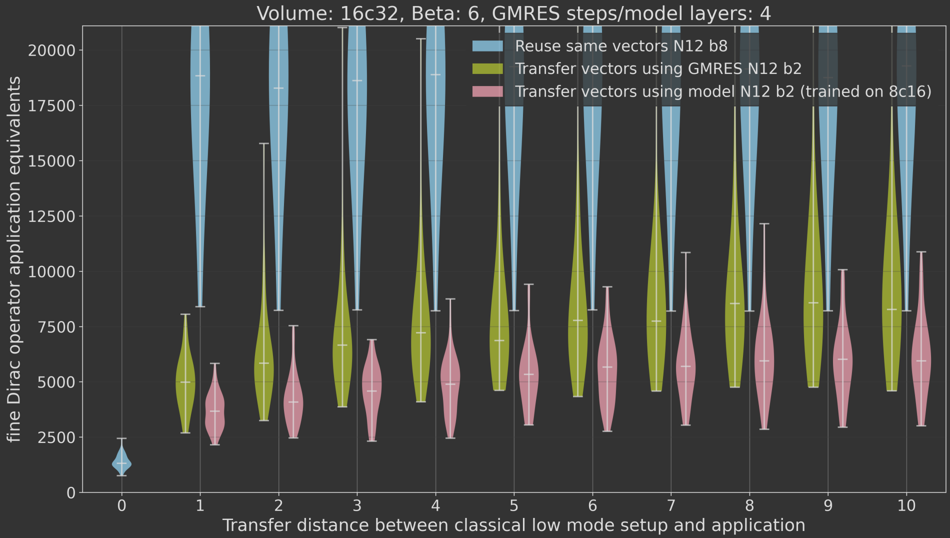
Evaluation


U_1
MC step
U_2
MC step
U_3
MC step
U_4
MC step
U_5
MC step
GMRES
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
Copy
$\psi_i^{(1)}$
GMRES
$\xi_i^{(2)}$
GMRES
$\xi_i^{(3)}$
GMRES
$\xi_i^{(4)}$
GMRES
$\xi_i^{(5)}$
GMRES
model
$\zeta_i^{(2)}$
model
$\zeta_i^{(3)}$
model
$\zeta_i^{(4)}$
model
$\zeta_i^{(5)}$
model

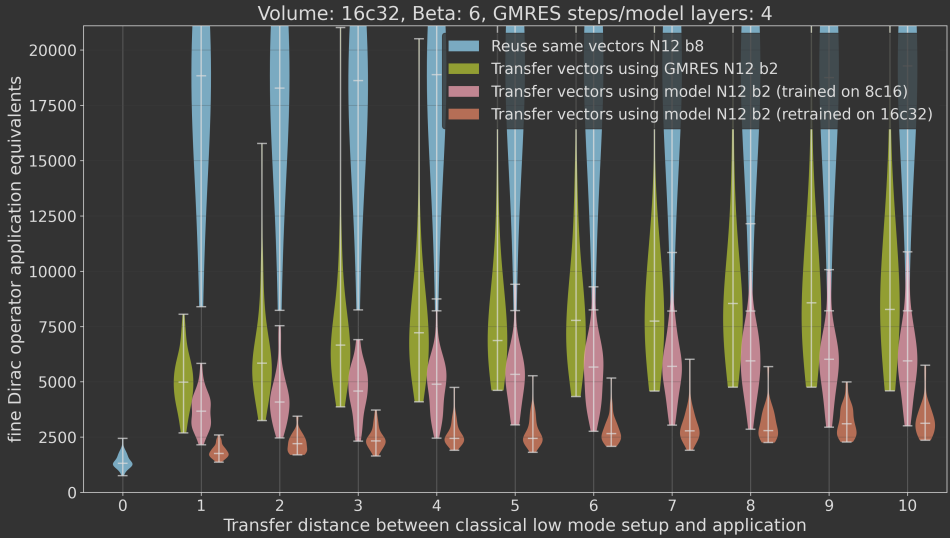
$8^3 \times 16$ volume



Transfer to $16^3 \times 32$ volume

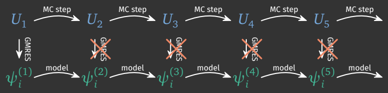


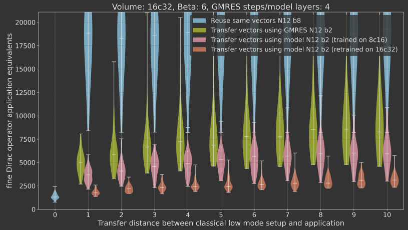
Retraining on $16^3 \times 32$ volume



Summary

- MG setups can be transferred from one configuration to the next in a Markov chain
- Using Machine Learning, we can outperform standard methods like GMRES



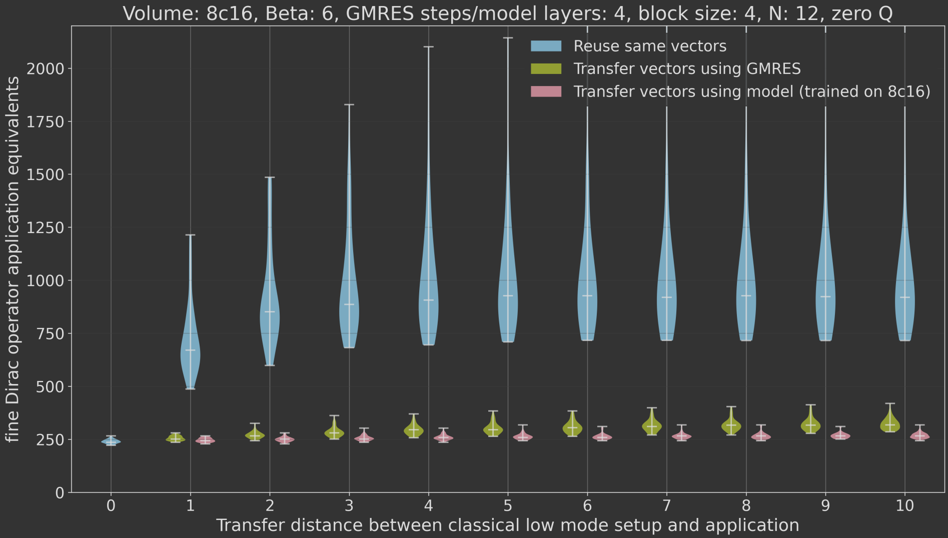


- Outlook
  - Implement this for unquenched simulations
  - Investigate analytically what the models learn

pfahler.online

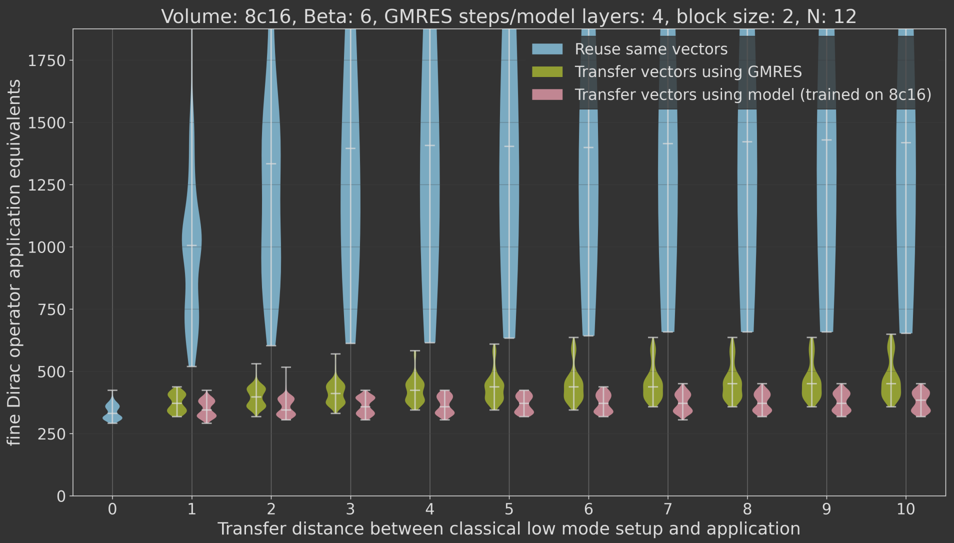
Bonus slides

Q-dependence



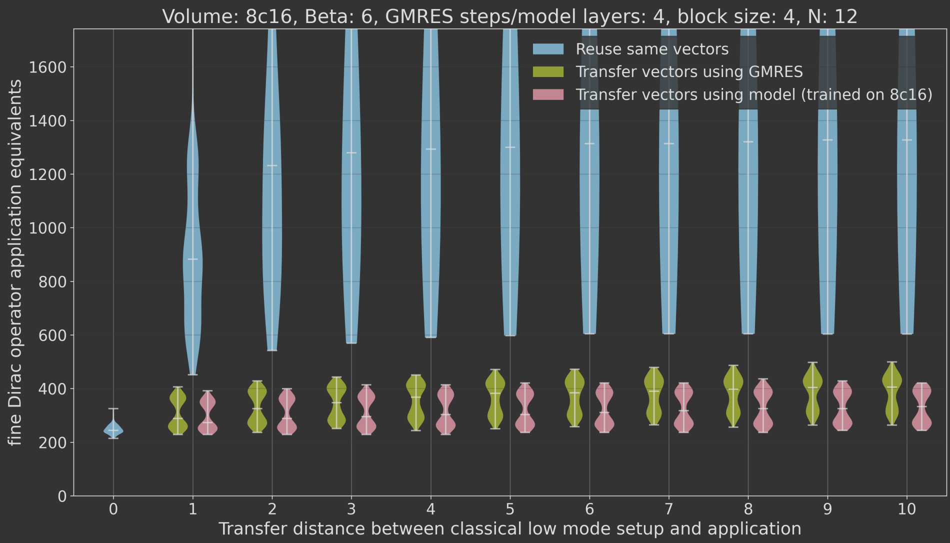
Q-dependence

Dependence on block size - $8^3 \times 16$

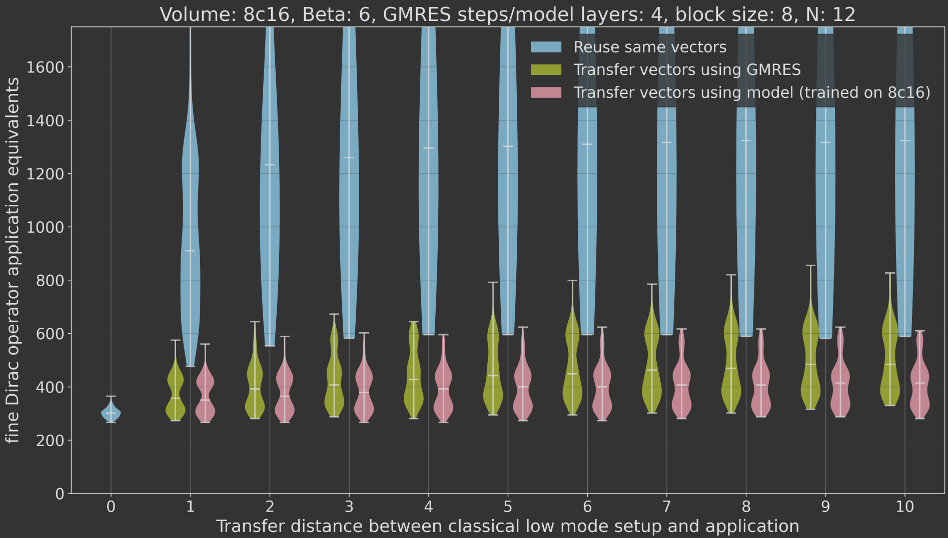


UR

Dependence on block size - $8^3 \times 16$



Dependence on block size - $8^3 \times 16$



Dependence on block size - $16^3 \times 32$

Dependence on block size - $16^3 \times 32$

Dependence on block size - $16^3 \times 32$