

DD-αAMG vs QUDA-MG

a comparison of multigrid performance using Twisted Mass fermions

Simone Bacchio[a,b], Kate Clark[c], Jacob Finkerath[d],
Dean Howarth[e], Bartosz Kostrzewa[f], Alejandro Vaquero[g]

[a] University of Cyprus
[b] University of Wuppertal
[c] NVIDIA
[d] The Cyprus Institute
[e] Temple University
[f] University of Bonn
[g] University of Utah

Lattice2017
35th annual International Symposium on Lattice Field Theory

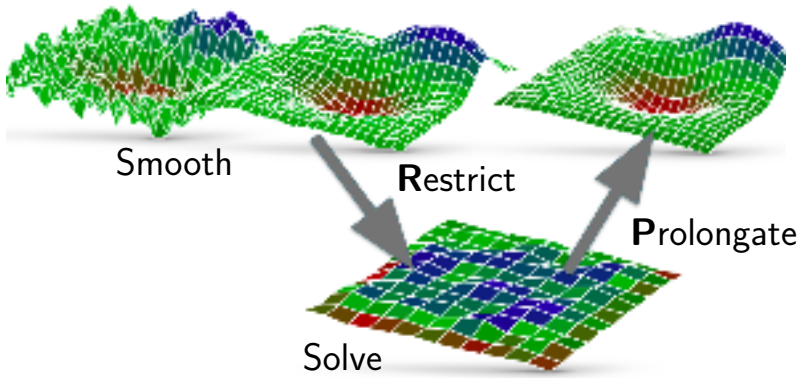
Overview

Multigrid methods and Softwares:

- DD-αAMG for TM [C. Alexandrou et al. 2016] [github.com/sbacchio/DDalphaAMG]
- QUDA-MG [M. A. Clark et al. 2016] [github.com/lattice/quda]

Covered topics:

- setup procedure
- smoother, SAP vs MR
- use of odd-even reduction



Multigrid methods for LQCD

Multigrid methods are used in LQCD to precondition flexible iterative solvers. They have proved to be efficient against the critical slowing down of the solvers approaching the physical point.

A two-level multigrid has the following error propagation

$$E_{2g}\epsilon = \underbrace{(\mathcal{I} - MD)^{\nu}}_{\text{post-smoothing}} \underbrace{\left(\mathcal{I} - PD_c^{-1}P^{\dagger}D\right)}_{\text{coarse grid correction}} \underbrace{(\mathcal{I} - MD)^{\mu}}_{\text{pre-smoothing}} \epsilon$$

where

- M is the smoother ($\sim$ standard preconditioner)
- $D_c = P^{\dagger}DP$ is the coarse grid operator
- P is the prolongation operator constructed such that
  - preserves the sparsity of D
  - projects the IR modes of D into the coarse grid
  - preserves the Γ_5-hermiticity of D (optional)

how? $\Gamma_5 P = P\Gamma_{5,c} \quad \Rightarrow \quad \Gamma_{5,c}\, D_c = \Gamma_{5,c}\, P^{\dagger} DP = P^{\dagger} D^{\dagger} P\, \Gamma_{5,c} = D_c^{\dagger}\, \Gamma_{5,c}.$

Multigrid for TM fermions

The $N_f = 2$ TM operator is given by

$$D(\mu) = D_W \pm i\Gamma_5\mu.$$

Thanks to the Γ_5-compatibility of P we obtain a similar coarse operator

$$D_c(\mu) = P^\dagger D_W P \pm i\Gamma_{5,c}\mu.$$

Thus MG machinery for Wilson fermions can be applied to TM fermions.

But

Multigrid for TM fermions

The $N_f = 2$ TM operator is given by

$$D(\mu) = D_W \pm i\Gamma_5\mu.$$

Thanks to the Γ_5-compatibility of P we obtain a similar coarse operator

$$D_c(\mu) = P^\dagger D_W P \pm i\Gamma_{5,c}\mu.$$

Thus MG machinery for Wilson fermions can be applied to TM fermions.

But a well-working set of parameters for D_W does not work (converges badly) for $D(\mu)$ at the physical point.

Multigrid for TM fermions

The $N_f = 2$ TM operator is given by

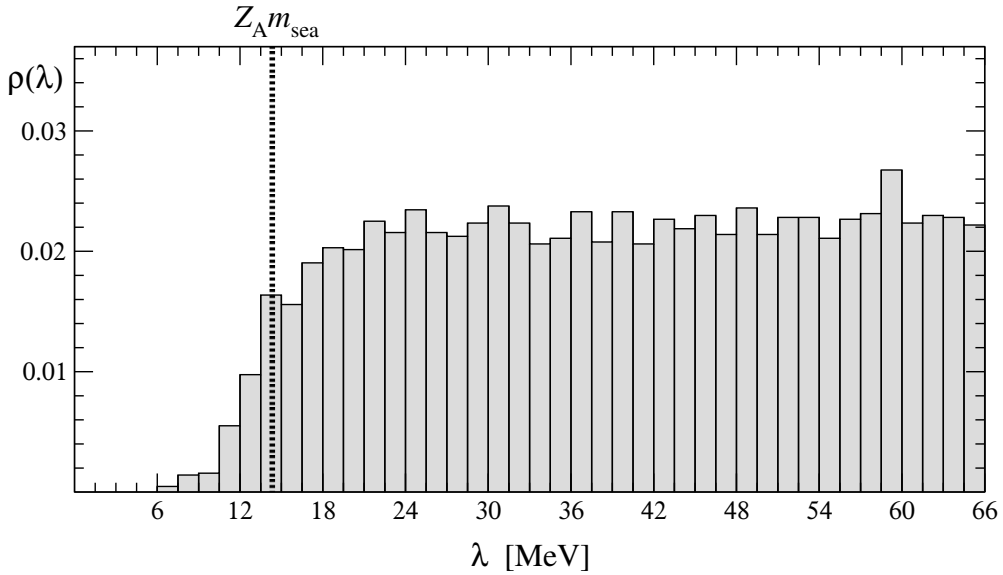
$$D(\mu) = D_W \pm i\Gamma_5\mu.$$

Thanks to the Γ_5-compatibility of P we obtain a similar coarse operator

$$D_c(\mu) = P^\dagger D_W P \pm i\Gamma_{5,c}\mu.$$

Thus MG machinery for Wilson fermions can be applied to TM fermions.

Indeed the eigenvalues density for $(D_W^\dagger D_W)^{1/2}$ is [M. Lüscher, 2007]



Multigrid for TM fermions

The $N_f = 2$ TM operator is given by

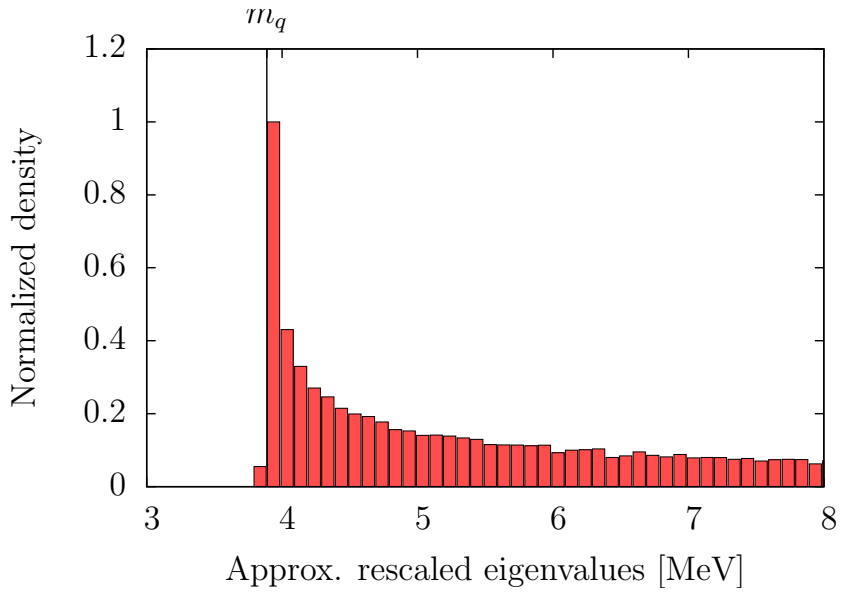
$$D(\mu) = D_W \pm i\Gamma_5\mu.$$

Thanks to the Γ_5-compatibility of P we obtain a similar coarse operator

$$D_c(\mu) = P^\dagger D_W P \pm i\Gamma_{5,c}\mu.$$

Thus MG machinery for Wilson fermions can be applied to TM fermions.

While for $(D^\dagger(\mu)D(\mu))^{1/2} = (D_W^\dagger D_W + \mu^2)^{1/2}$ is



Multigrid for TM fermions

The $N_f = 2$ TM operator is given by

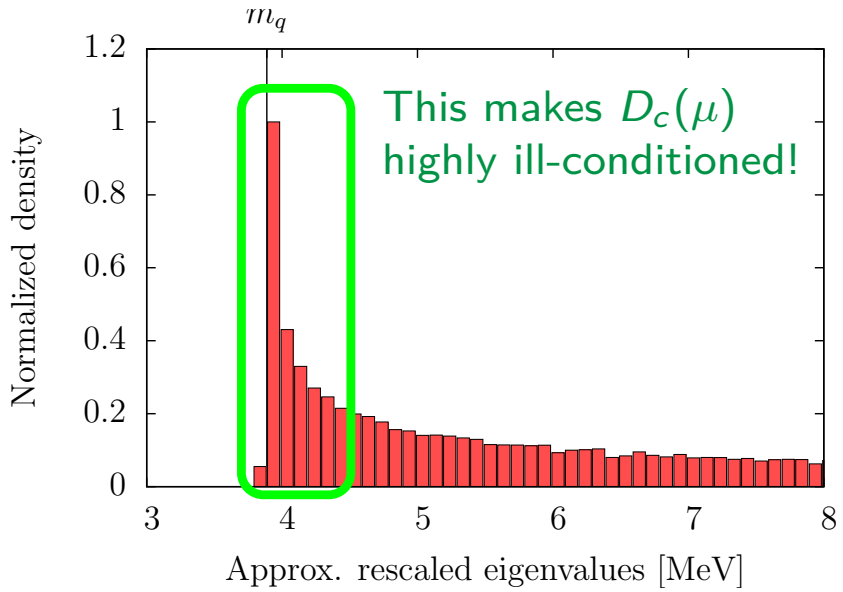
$$D(\mu) = D_W \pm i\Gamma_5\mu.$$

Thanks to the Γ_5-compatibility of P we obtain a similar coarse operator

$$D_c(\mu) = P^\dagger D_W P \pm i\Gamma_{5,c}\mu.$$

Thus MG machinery for Wilson fermions can be applied to TM fermions.

While for $(D^\dagger(\mu)D(\mu))^{1/2} = (D_W^\dagger D_W + \mu^2)^{1/2}$ is



Multigrid for TM fermions

The $N_f = 2$ TM operator is given by

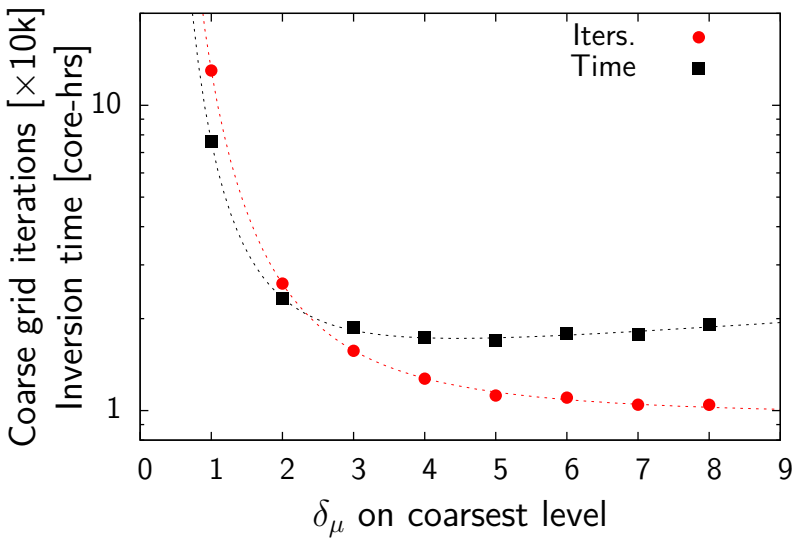
$$D(\mu) = D_W \pm i\Gamma_5\mu.$$

Thanks to the Γ_5-compatibility of P we obtain a similar coarse operator

$$D_c(\mu, \delta_\mu) = P^\dagger D_W P \pm i\Gamma_{5,c}\delta_\mu\mu.$$

Thus MG machinery for Wilson fermions can be applied to TM fermions.

Solution: improve the condition number of D_c increasing the TM parameter.



Status of the softwares

- DDalphaAMG
  - CPU code, written in C
  - supports Wilson operator of the kind $D(m, c_{sw}, \mu)$
  - implementation of DD-αAMG method
- QUDA
  - GPU code, written in C++
  - supports a long list of operators, but MG works for $D(m, c_{sw}, \mu)$
  - implementation of features from MG-GCR, DD-αAMG and new ideas

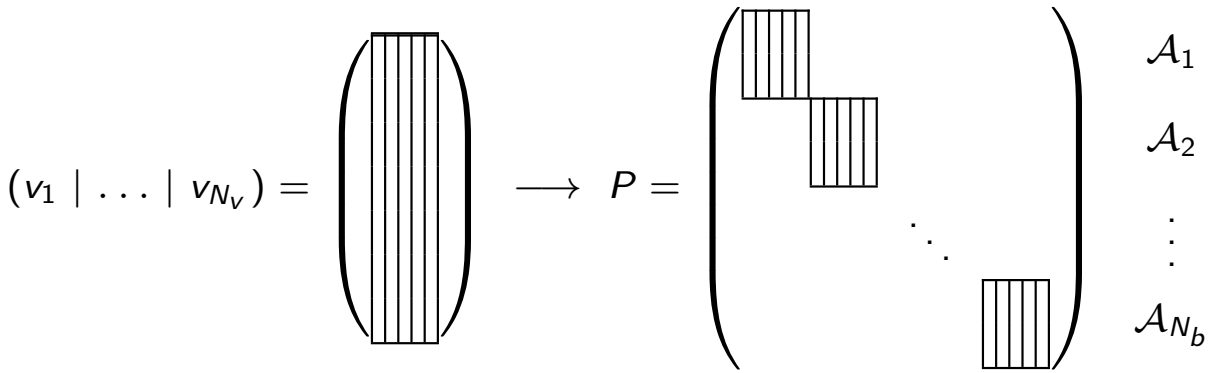
Main differences of the MG

Component	DDalphaAMG	QUDA-MG
setup	test vector	null vector
smoother	SAP	MR, ∼SAP, any solver
oe-reduction	on coarsest level	can be every where
top solver	FGMRES	GCR
bottom solver	GMRES	GCR, any solver

Setup procedure - test vectors (DD-αAMG)

- **Idea:** test vectors capture the low eigenmodes of D by inverting several times D on a random rhs.

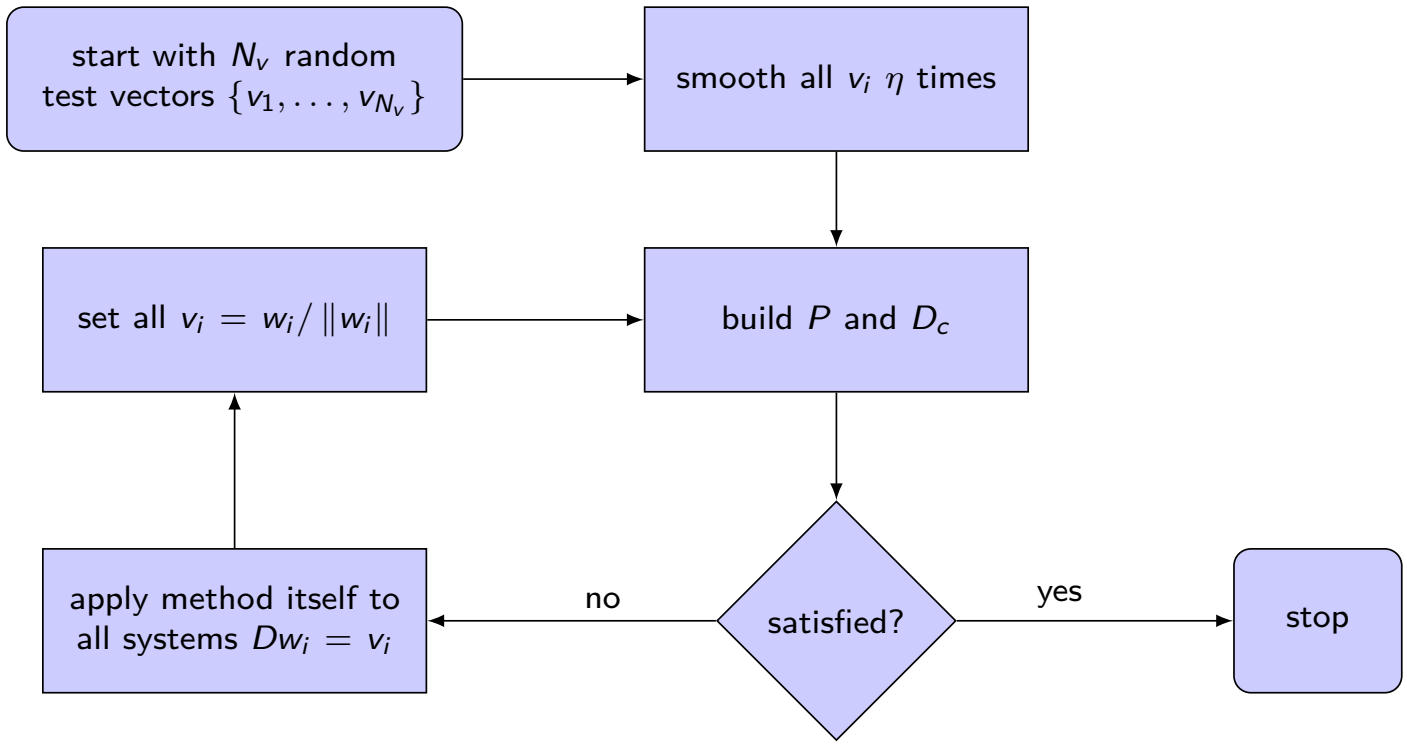
$\Rightarrow$ Indeed $D^{-n} = U\lambda^{-n}V$ is dominated by the low-modes of D.



Setup procedure - test vectors (DD-αAMG)

- **Idea:** test vectors capture the low eigenmodes of D by inverting several times D on a random rhs.

$\Rightarrow$ Indeed $D^{-n} = U\lambda^{-n}V$ is dominated by the low-modes of D.



Setup procedure - test vectors (DD-αAMG)

- **Idea:** test vectors capture the low eigenmodes of D by inverting several times D on a random rhs.

$\Rightarrow$ Indeed $D^{-n} = U\lambda^{-n}V$ is dominated by the low-modes of D.

Setup iters	Setup time [core-hrs]	Setup iter cost [core-hrs]	Inversion time [core-hrs]	Inversion iters
2	5.5	–	4.6	48
3	11.2	5.7	1.9	20
4	17.8	6.6	1.7	18
5	24.4	6.6	1.6	17
6	30.8	6.4	1.7	18
7	37.2	6.4	1.7	18

- 3 setup iterations are good for few inversions (up to $\sim$20).
- 5 setup iterations are good for a stable optimal time to solution.
- The setup cost increases because the coarse operator turns more ill-conditioned.

*results for a 96×48^3 lattice, $a = 0.0931(10)$ fm and $m_\pi = 131(1)$ MeV from *cA*2.09.48.
[Abdel-Rehim et al, 2015]

Setup procedure - null vectors (QUDA-MG)

- **Idea:** null vectors capture the low eigenmodes of D by solving the homogeneous equation starting from a random initial guess.

$\Rightarrow$ Indeed $Dx = 0$ returns the error of the solver, $x = \epsilon$, which is dominated by the modes where the solver does not converge well
$\Rightarrow$ low-modes for standard Krylov solver.

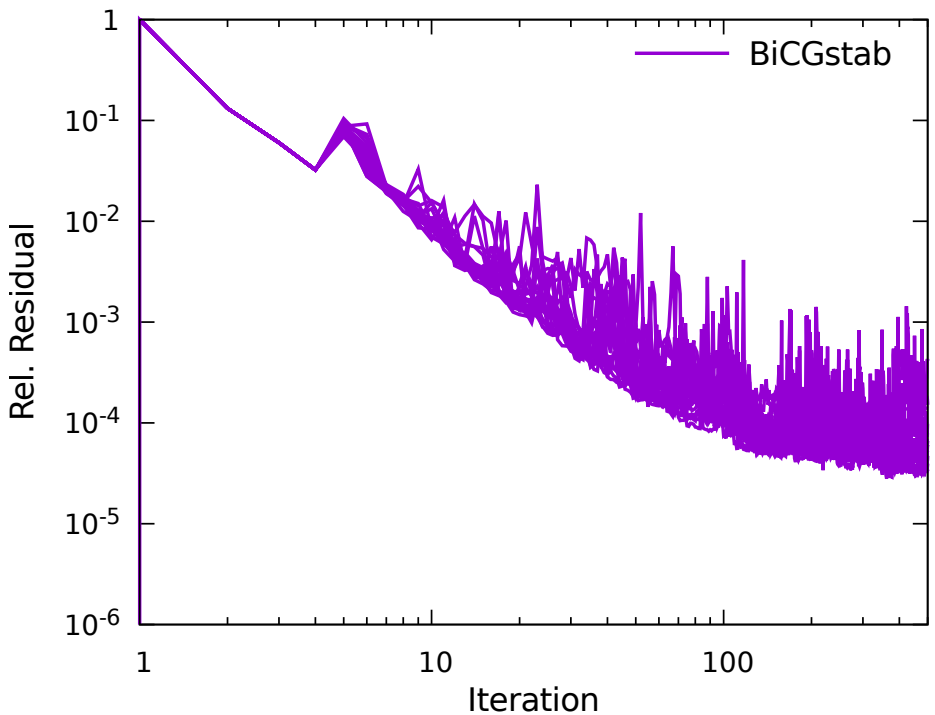
Setup procedure - null vectors (QUDA-MG)

- **Idea:** null vectors capture the low eigenmodes of D by solving the homogeneous equation starting from a random initial guess.

$\Rightarrow$ Indeed $Dx = 0$ returns the error of the solver, $x = \epsilon$, which is dominated by the modes where the solver does not converge well
$\Rightarrow$ low-modes for standard Krylov solver.

- Some solvers do not converge well for TM fermions.

Results: BiCGstab (55 iters, 26.5 secs)



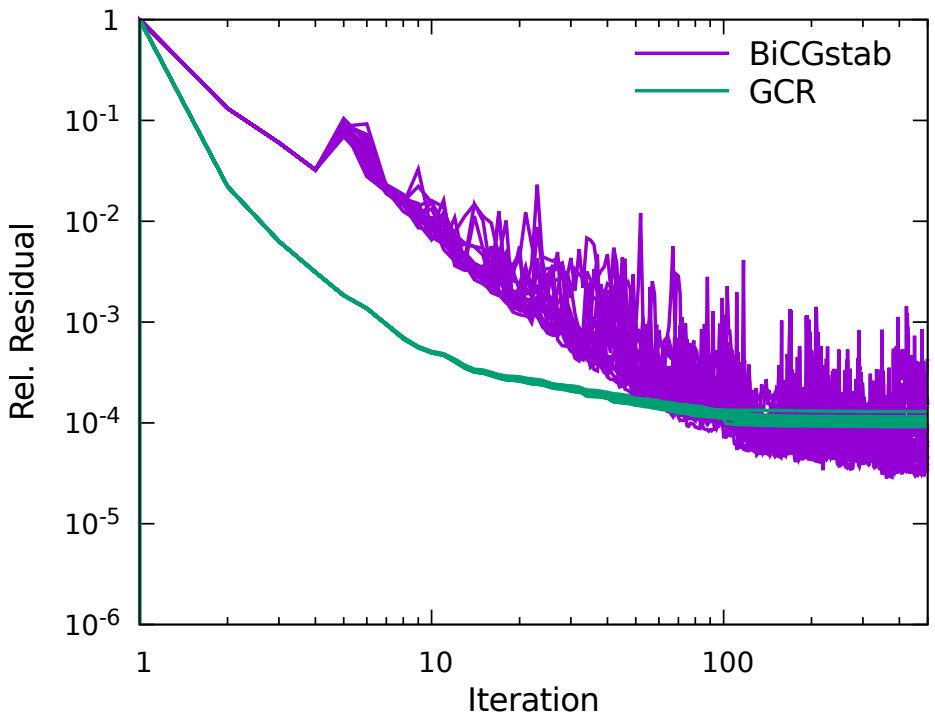
Setup procedure - null vectors (QUDA-MG)

- **Idea:** null vectors capture the low eigenmodes of D by solving the homogeneous equation starting from a random initial guess.

$\Rightarrow$ Indeed $Dx = 0$ returns the error of the solver, $x = \epsilon$, which is dominated by the modes where the solver does not converge well
$\Rightarrow$ low-modes for standard Krylov solver.

- Some solvers do not converge well for TM fermions.

Results: BiCGstab (55 iters, 26.5 secs), GCR (69 iters, 29 secs)



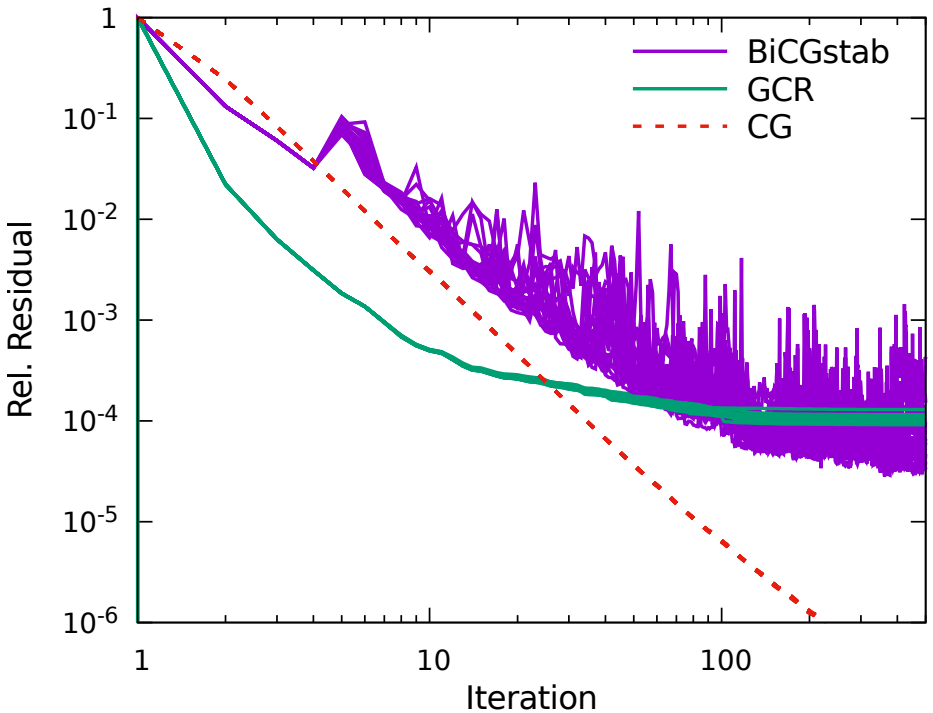
Setup procedure - null vectors (QUDA-MG)

- **Idea:** null vectors capture the low eigenmodes of D by solving the homogeneous equation starting from a random initial guess.

$\Rightarrow$ Indeed $Dx = 0$ returns the error of the solver, $x = \epsilon$, which is dominated by the modes where the solver does not converge well
$\Rightarrow$ low-modes for standard Krylov solver.

- Some solvers do not converge well for TM fermions.

Results: BiCGstab (55 iters, 26.5 secs), GCR (69 iters, 29 secs), CG (35 iters, 9.2 secs)

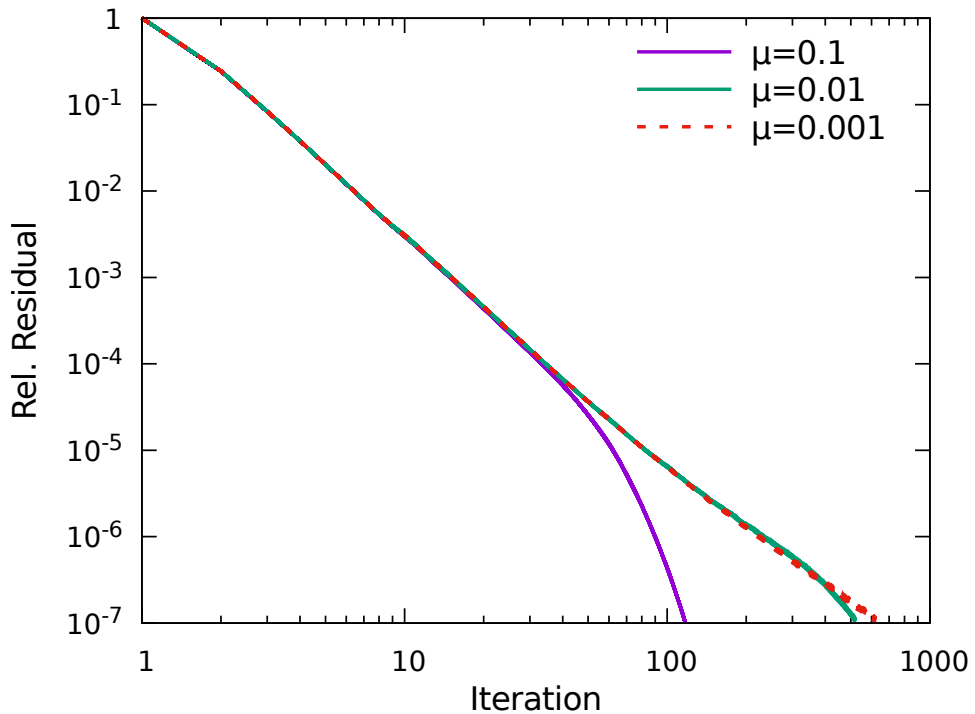


Setup procedure - null vectors (QUDA-MG)

- **Idea:** null vectors capture the low eigenmodes of D by solving the homogeneous equation starting from a random initial guess.

$\Rightarrow$ Indeed $Dx = 0$ returns the error of the solver, $x = \epsilon$, which is dominated by the modes where the solver does not converge well
$\Rightarrow$ low-modes for standard Krylov solver.

- A good setup for TM fermions is obtained with CG.
- The convergence is not affected by the critical slowing down.

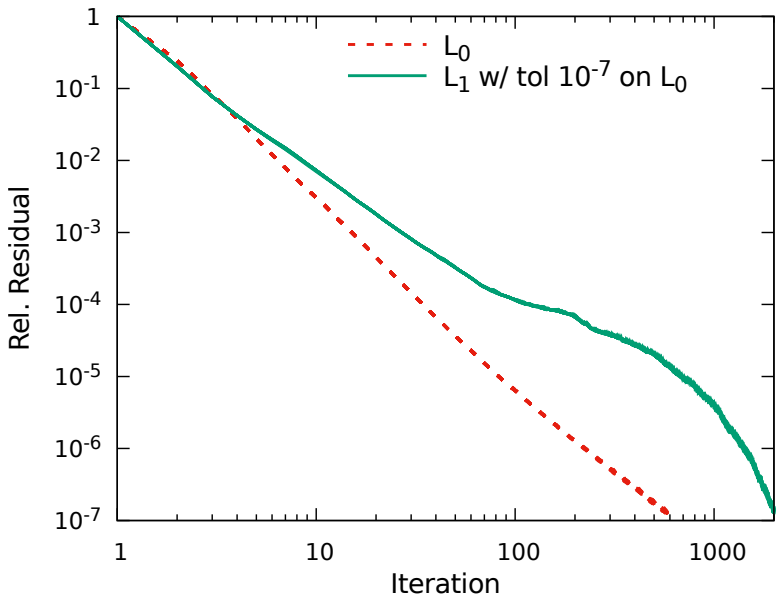


Setup procedure - null vectors (QUDA-MG)

- **Idea:** null vectors capture the low eigenmodes of D by solving the homogeneous equation starting from a random initial guess.

$\Rightarrow$ Indeed $Dx = 0$ returns the error of the solver, $x = \epsilon$, which is dominated by the modes where the solver does not converge well
$\Rightarrow$ low-modes for standard Krylov solver.

- A good setup for TM fermions is obtained with CG.
- The convergence is not affected by the critical slowing down.
- A good performance is obtained with a tolerance $> 5 \cdot 10^{-6}$.



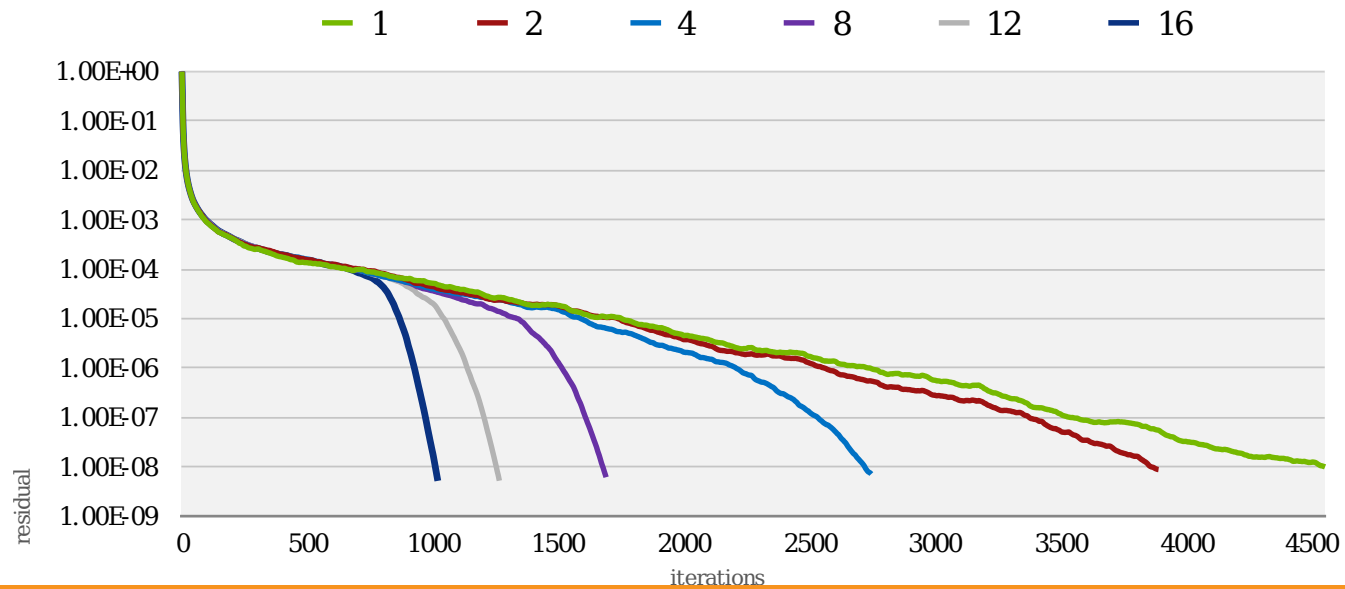
Setup tol.	Inv. iters.	Inv. time
10^{-4}	652	67.2 s
$10^{-5.5}$	307	48.4
10^{-5}	62	11.1
$10^{-6.5}$	42	8.2
10^{-6}	35	7.2
$10^{-7.5}$	32	7.1

Setup procedure - null vectors (QUDA-MG)

- **Idea:** null vectors capture the low eigenmodes of D by solving the homogeneous equation starting from a random initial guess.

$\Rightarrow$ Indeed $Dx = 0$ returns the error of the solver, $x = \epsilon$, which is dominated by the modes where the solver does not converge well
$\Rightarrow$ low-modes for standard Krylov solver.

- A good setup for TM fermions is obtained with CG.
- The convergence is not affected by the critical slowing down.
- A good performance is obtained with a tolerance $> 5 \cdot 10^{-6}$.
- Next to try: block-CG solver [M. Wagner, Tue]

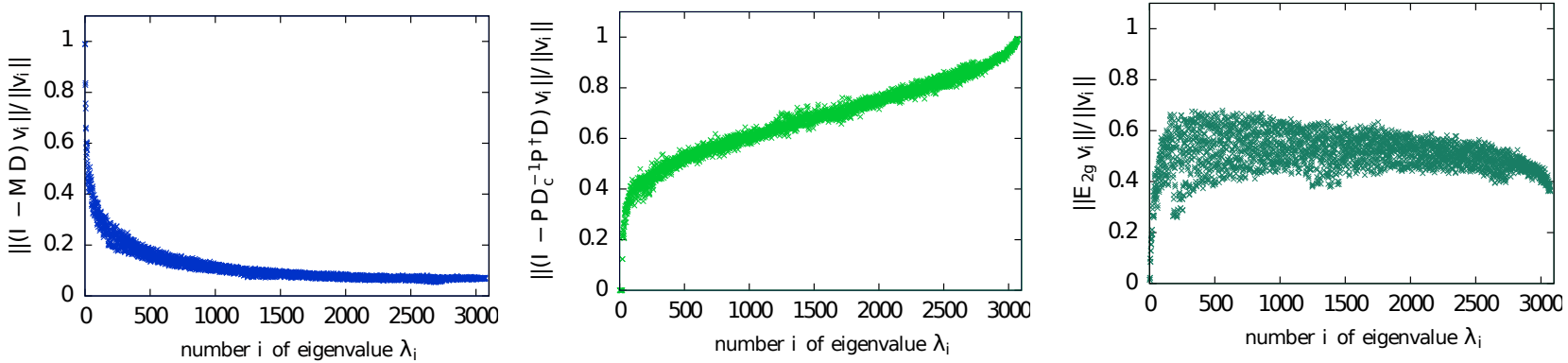


SAP vs MR

In DD-αAMG the Schwarz Alternating Procedure (SAP) is used as a smoother.

- The SAP is known to work well on the UV mode. [M. Lüscher, 2003]
- It has been chosen in order to be complementary to the coarse grid correction.

Smoother (SAP) + Coarse grid correction = DD-αAMG



SAP vs MR

In DD-αAMG the Schwarz Alternating Procedure (SAP) is used as a smoother.

- The SAP is known to work well on the UV mode. [M. Lüscher, 2003]
- It has been chosen in order to be complementary to the coarse grid correction.

In MG-GCR few iterations of relaxed MR were suggested as a smoother.

This method is implemented in QUDA.

SAP vs MR

In DD-αAMG the Schwarz Alternating Procedure (SAP) is used as a smoother.

- The SAP is known to work well on the UV mode. [M. Lüscher, 2003]
- It has been chosen in order to be complementary to the coarse grid correction.

In MG-GCR few iterations of relaxed MR were suggested as a smoother.

This method is implemented in QUDA.

SAP, DDalphaAMG*, 54 nodes†

n SAP	Inv. iters	Inv. time
1	37	14.7 s
2	23	7.0 s
3	20	7.0 s
4	19	6.7 s
5	18	7.0 s
6	18	7.1 s

MR, $\omega = 0.85$, QUDA*, 54 GPUs‡

n iter	Inv. iters	Inv. time
1	55	15.1 s
2	42	7.4 s
3	39	6.4 s
4	38	6.2 s
5	36	6.4 s
6	34	6.9 s

* using a similar set of parameters (main difference: setup phase)

† Supermuc (2015) ‡ Titan (2012)

SAP vs MR

In DD-αAMG the Schwarz Alternating Procedure (SAP) is used as a smoother.

- The SAP is known to work well on the UV mode. [M. Lüscher, 2003]
- It has been chosen in order to be complementary to the coarse grid correction.

In MG-GCR few iterations of relaxed MR were suggested as a smoother.

This method is implemented in QUDA.

We started the implementation of SAP in QUDA and at the moment a communication-*killing* version is available – i.e. the Schwarz block has the same size of the local volume.

This depends on the parallelization and a speed-up of 10%÷40% can be obtained in inversion time compared to MR. [K. Clark, Tue]

Use of oe-reduction

In DD-αAMG the oe-reduction is used in the smoother and in the bottom solver where D_c^{-1} is computed.

- This does not change the definition of the MG algorithm.

$\Rightarrow$ The speed-up is around 20%.

Use of oe-reduction

In DD-αAMG the oe-reduction is used in the smoother and in the bottom solver where D_c^{-1} is computed.

- This does not change the definition of the MG algorithm.

$\Rightarrow$ The speed-up is around 20%.

In QUDA-MG the oe-reduction can be used on all the levels, up to precondition the inversion of the Schur complement with MG.

- In the latter case the MG algorithm changes and the speed-up depends on the application.

$\Rightarrow$ For Wilson fermions the speed-up is around 50%.

Use of oe-reduction

In DD-αAMG the oe-reduction is used in the smoother and in the bottom solver where D_c^{-1} is computed.

- This does not change the definition of the MG algorithm.

$\Rightarrow$ The speed-up is around 20%.

In QUDA-MG the oe-reduction can be used on all the levels, up to precondition the inversion of the Schur complement with MG.

- In the latter case the MG algorithm changes and the speed-up depends on the application.

$\Rightarrow$ For TM fermions the speed-up depends on the μ-shift.

no-oe			**oe on D_c, $\sim$20%**			**oe on D, $\sim$40%**		
δ_μ	Inv. iters	Inv. time	δ_μ	Inv. iters	Inv. time	δ_μ	Inv. iters	Inv. time
2	26	47.1 s	2	26	31.7 s	5	29	66.8 s
3	28	12.8 s	3	29	9.9 s	10	30	9.7 s
4	30	10.8 s	4	30	8.5 s	15	32	7.0 s
5	31	10.7 s	5	31	8.2 s	20	33	6.7 s
6	31	10.5 s	6	31	8.1 s	25	36	7.1 s
7	32	11.2 s	7	32	8.4 s	30	37	7.4 s
8	33	11.4 s	8	33	8.6 s	35	37	6.7 s

Conclusions

What has been done in QUDA-MG for TM fermions

- μ-shifts can be used on all levels,
- CG enabled in the setup phase $\Rightarrow$ needed $D_c^\dagger$,
- implementation of SAP is ongoing,
- many other trials that didn't work,
- tuning of the parameters.

Other remarks

- TM fermions are more sensible to the change of parameters

$\Rightarrow$ require an accurate tuning for a good MG,

$\Rightarrow$ finding a good approach can help to improve MG for Wilson fermions.

Thank you for you attention!

and I kindly thank

My supervisors	Prof. C. Alexandrou (UCY) and Prof. A. Frommer (BUW)
Horizon 2020	for founding under the Marie Sklodowska-Curie grant agreement No 642069.
QUDA developers	for all the contributions to the software.
JSC	for hosting Eurohack 2017, the GPU hackathon in Jülich.
Quantum-tornados	team of Eurohack 2017, additionally to the authors K. Hadjiyiannakou and B. Kostrzewa.
Facilities	SuperMUC at the Munich, Titan at the Oak Ridge and Piz Daint at the ETH Zürich Supercomputing Centers.

DD-αAMG vs QUDA-MG

a comparison of multigrid performance using Twisted Mass fermions

Simone Bacchio[a,b], Kate Clark[c], Jacob Finkerath[d], Dean Howarth[e], Bartosz Kostrzewa[f], Alejandro Vaquero[g]

[a] University of Cyprus
[b] University of Wuppertal
[c] NVIDIA
[d] The Cyprus Institute
[e] Temple University
[f] University of Bonn
[g] University of Utah

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No' 642069.