# Exploiting the hopping parameter expansion in the HMC simulation of lattice QCD

Martin Hasenbusch

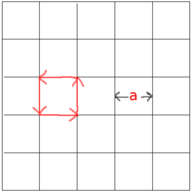
Humboldt-Universität zu Berlin

Frontiers in Lattice Quantum Field Theory, 1 June 2018
Instituto de Física Teórica UAM-CSIC, Madrid

# Overview

M. Hasenbusch, [arXiv:1805.03560]

- The model; Improved pseudo fermions
- UV-filtering, rooted polynomials
- Numerical Results
- Conclusions




Lattice QCD:

- 4 dimensional hyper-cubic lattice
- $x$ sites of the lattice, $\mu$ direction
- Gauge field $U_{x,\mu} \in SU(3)$ lives on the link $(x,\mu)$
- quark fields live on the sites

Wilson gauge action

$$S_G[U] = -\frac{\beta}{3} \sum_x \sum_{\mu>\nu} \text{Re Tr} \left( U_{x,\mu} U_{x+\hat{\mu},\nu} U^\dagger_{x+\hat{\nu},\mu} U^\dagger_{x,\nu} \right)$$

Wilson fermions

$$H[U] = \sum_\mu \left\{ (1-\gamma_\mu)\; U_{x,\mu}\; \delta_{x+\hat{\mu},y} + (1+\gamma_\mu)\; U^\dagger_{x-\hat{\mu},\mu}\; \delta_{x-\hat{\mu},y} \right\}$$

$$M[U] = 1 - \kappa H[U]$$

# QCD according to the particle data book 2017:

$$m_u = 2.2^{+0.6}_{-0.4}\ MeV \quad m_d = 4.7^{+0.5}_{-0.4}\ MeV \quad m_s = 96^{+8}_{-4}\ MeV$$

$$m_c = 1.28 \pm 0.03\ GeV \qquad m_b = 4.18^{+0.04}_{-0.03}\ GeV \quad m_t = 173.1 \pm 0.6\ GeV$$

We simulate: Two degenerate flavours of dynamical quarks

$m_u = m_d$ finite $\qquad m_s = m_c = m_b = m_t = \infty$

Fermions (Grassmann variables) can be integrated out:

$$Z = \int \mathrm{D}[U]\ \exp(-S_G[U])\ \det M[U]^2$$

# Hybrid Monte Carlo (HMC) Algorithm

determinant is too expensive ( $\propto$ *Volume*$^3$) $\rightarrow$ pseudo-fermions

$$\det M^2 = \det M M^\dagger \;\propto\; \int \mathrm{D}[\phi^\dagger] \int \mathrm{D}[\phi]\;\exp(-|M^{-1}\phi|^2)$$

Problem: $S_F = |M^{-1}\phi|^2$ is non-local

$\Longrightarrow$ molecular dynamics evolution of all gauge fields.
Introduce conjugate momenta $P$ for the gauge field

$\Longrightarrow$ Hamiltonian:

$$H(U,\phi,\phi^\dagger,\Pi) = S_G(U) + S_F(U,\phi,\phi^\dagger) + \frac{1}{2}(\Pi,\Pi)$$

where $(\Pi,\Pi) = -2\sum_{x,\mu} \mathrm{Tr}\Pi_{x,\mu}^2$

# What is the problem?

The simulation becomes more expensive as the quark mass becomes smaller, (Lattice 2001, Berlin "Berlin wall"):

$cost = m_{PS}^{-2.8(2)}$ (for $\beta = 5.6$, Lippert 2001)

- Condition number of $M$ increases
  $\Longrightarrow$ solver needs more iterations
- step size must be decreased to get constant acceptance

# Improved pseudo-fermions

Introduce $N$ matrices $W_i$ such that

$$M = \prod_{i=1}^{N} W_i$$

The $W_i$ should have a smaller condition number than $M$
Introduce pseudo-fermions for each $W_i$

$$\det M M^\dagger \quad \propto$$

$$\int D[\phi_1^\dagger] \int D[\phi_1] ... \int D[\phi_N^\dagger] \int D[\phi_N] \;\; \exp(-\sum_{i=1}^{N} |W_i^{-1} \phi_i|^2)$$

- Mass-preconditioning (Hasenbusch 2001):

$$
\begin{aligned}
W_1 &= M + \rho_1 \\
W_i &= (M + \rho_{i-1})^{-1}(M + \rho_i) \quad \text{for } 1 < i < N \\
W_N &= (M + \rho_{N-1})^{-1} M
\end{aligned}
$$

Alternative (R. Sommer; Hasenbusch and Jansen 2003): add $\rho_i \gamma_5$

- Polynomial splitting (Peardon 2001)
formally the same splitting as in PHMC, however low order polynomial

$$W_1^{-1} = P(M) \approx M^{-1}$$

Easy to combine with “UV-filtering” or PHMC.

- RHMC (Clark, Kennedy 2004): take the $n^{th}$ root of $M$, introduce a pseudo-fermion field for each of the roots. Technically done with a rational approximation. Requires a multi-mass solve for each of the roots.

- Schwarz preconditioned HMC (Lüscher 2004)

# The hopping parameter expansion

$$\det M^\dagger M = \exp(\mathrm{Tr}\ln M^\dagger + \mathrm{Tr}\ln M)$$

one expands

$$\ln M = \ln(1-\kappa H) = -\sum_{n=1}^{\infty} \frac{1}{n}\kappa^n H^n \; .$$

For small $n$, $\mathrm{Tr} H^n$ can be evaluated analytically.
$n = 4$: shift in the gauge-coupling $\beta$.

Clover-improvement: already non-trivial contribution from $n = 2$
K.-I. Ishikawa *et al.*,
arXiv:hep-lat/0610037, PoS LAT **2006**, 027 (2006).

- M.H., hep-lat/9807031, Phys.Rev. D 59 (1999) 054505;
- Ph. de Forcrand, *UV-filtered fermionic Monte Carlo*, hep-lat/9809145, Nucl.Phys.Proc.Suppl. 73 (1999) 822.

$$\tilde{M} = M \, \exp\left(\sum_{n=1}^{k} \frac{1}{n} \kappa^n H^n\right)$$

The inverse $\tilde{M}^{-1}$ can be represented by a polynomial in $M$.

$\rightarrow$ Multiboson algorithm

C. Alexandrou, Ph. de Forcrand, M. D'Elia, and H. Panagopoulos, *Efficiency of the UV-filtered Multiboson algorithm*, [arXiv:hep-lat/9906029], Phys. Rev. D **61**, 074503 (2000).

*... reduces the number of bosonic fields by a factor 3 or more ...*

→ Polynomial Hybrid Monte Carlo (PHMC)

K.-I. Ishikawa *et al.* [PACS-CS Collaboration], *An Application of the UV-filtering preconditioner to the polynomial hybrid Monte Carlo algorithm*, [arXiv:hep-lat/0610037] PoS LAT **2006**, 027 (2006).

... *UV-filtering reduces the magnitude of the molecular dynamics force from the pseudo fermion by a factor 3 by tuning the UV-filter parameter. Combining with the multi-time scale molecular dynamics integrator we achieve a factor 2 improvement*.

Here, simplest example:

$$\tilde{M}^{-1} = \exp(-\kappa H)(1-\kappa H)^{-1} = \sum_{n=0}^{\infty} a_n \kappa^n H^n$$

$$a_n = \sum_{i=0}^{n} (-1)^i \frac{1}{i!} \qquad , \qquad \lim_{n\to\infty} a_n = \exp(-1)$$

Hence we can write

$$\tilde{M}^{-1} = \sum_{n=0}^{\infty} b_n \kappa^n H^n + \alpha M^{-1}$$

where $\alpha = \exp(-1)$ and $b_n = -\sum_{i=n+1}^{\infty} (-1)^i \frac{1}{i!}$

$\Longrightarrow \sum_{n=0}^{\infty} b_n \kappa^n H^n$ can be truncated at low order

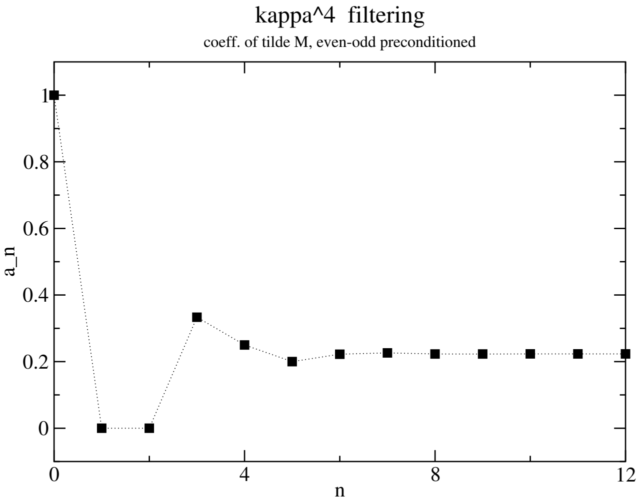
# Even-odd preconditioning

$$M_{oo} = 1 - \kappa^2 H_{oe} H_{eo}$$

Reduces the condition number of the fermion matrix
Larger step size (de Forcrand, Takaishi 1996)

Order of polynomials in the following: powers of $\kappa^2$

$M_{oo} = 1 - \kappa^2 H_{oe} H_{eo}$. Coefficient $n$ refers to powers of $\kappa^2$



# Comparison with mass preconditioning

$$W_1 = M + \rho$$
$$W_2 = (M + \rho)^{-1} M$$

Taking the inverse

$$W_2^{-1} = 1 + \rho M^{-1}$$

Hence $\alpha$ plays a similar role as $\rho$

Generalization by using factorized rooted polynomials
Idea: Noise reduction by rooting, similar to RHMC

$M_1 = \tilde{M}$, define recursively

$$M_{j+1} = W_j^{-N_j} M_j$$

where

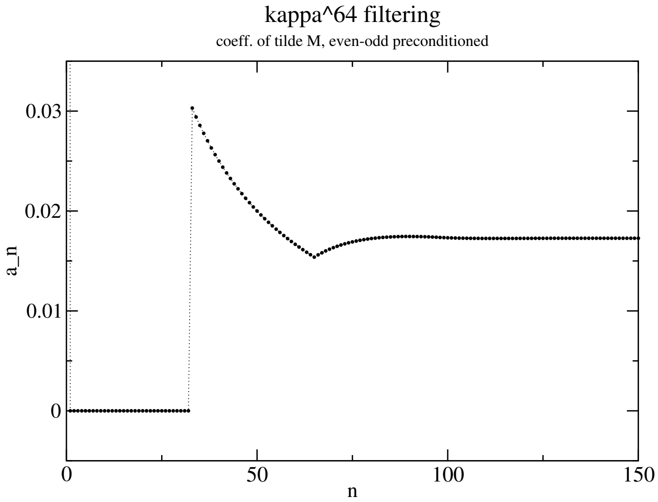
$$W_j^{-1} = \sum_{i=0}^{n_j} a_{j,i} \kappa^i H^i = M_j^{-1/N_j} + \mathrm{O}\left(\kappa^{n_j+1}\right)$$

The remainder

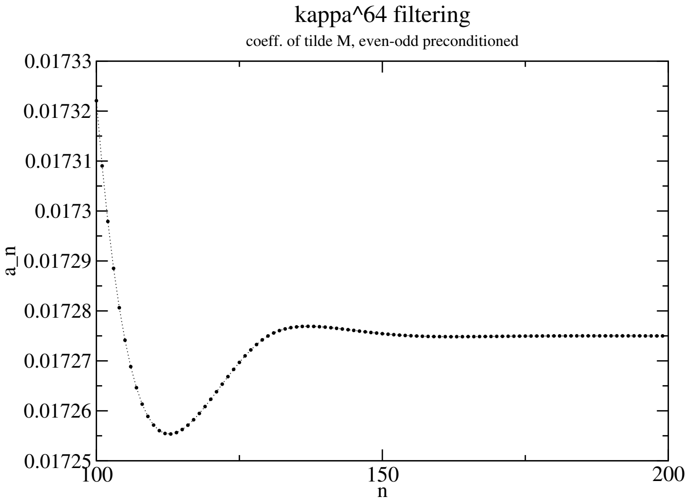
$$M_{j_{max}+1}^{-1} = \sum_{n=0}^{\infty} b_n \kappa^n H^n + \alpha M^{-1}$$

Computed by using an algebra program

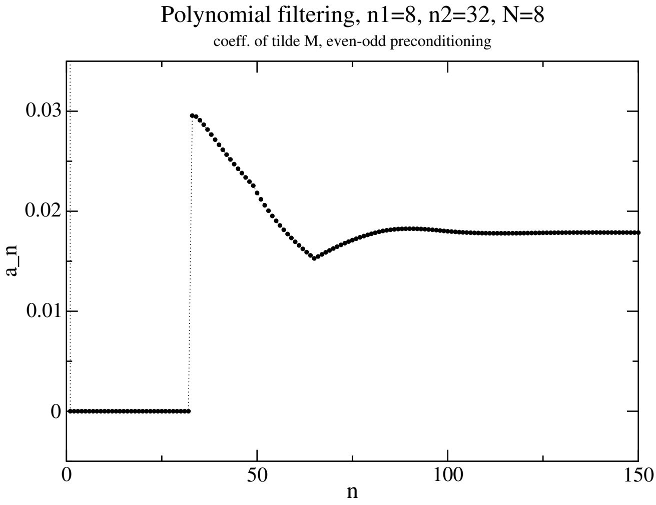
$M_{oo} = 1 - \kappa^2 H_{oe} H_{eo}$. Coefficient $n$ refers to powers of $\kappa^2$



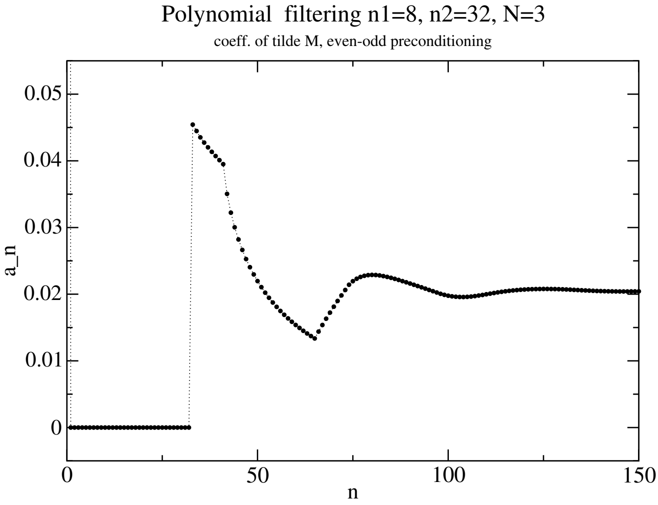
$M_{oo} = 1 - \kappa^2 H_{oe} H_{eo}$. Coefficient $n$ refers to powers of $\kappa^2$



$M_{oo} = 1 - \kappa^2 H_{oe} H_{eo}$. Coefficient $n$ refers to powers of $\kappa^2$



$M_{oo} = 1 - \kappa^2 H_{oe} H_{eo}$. Coefficient $n$ refers to powers of $\kappa^2$



# Computing forces

Rooted Polynomials:
PHMC: Horner scheme; we need to store $n$ vectors, where $n$ is the order of the polynomial

Remainder $M_{j_{max}+1}^{-1}$:
extra effort due to the additive polynomial

$$\sum_{n=0}^{\infty} b_n \kappa^n H^n$$

That can be (hopefully) truncated at low order.

# Integration schemes

$$\begin{aligned} P(\delta\tau): \; \Pi_{x,\mu} \to \Pi'_{x,\mu} &= \Pi_{x,\mu} + \delta\tau \mathcal{F}_{x,\mu} \;, \\ T(\delta\tau): U_{x,\mu} \to U'_{x,\mu} &= \exp(i\delta\tau\Pi_{x,\mu}) \; U_{x,\mu} \;. \end{aligned}$$

Second order Omelyan integrator

$$T_O = P(\lambda\delta\tau) \; T(\delta\tau/2) \; P([1-2\lambda]\delta\tau) \; T(\delta\tau/2) \; P(\lambda\delta\tau)$$

$\lambda = 1/6$ the scheme proposed by Sexton and Weingarten

$\lambda = 1/2$ leapfrog scheme

$$T_L = P(\delta\tau/2) \; T(\delta\tau) \; P(\delta\tau/2)$$

Multi-time scale following Sexton and Weingarten

At the end of the trajectory the new gauge field is accepted with the probability

$$P_{acc} = \min[1, \exp(-\Delta H)]$$

where

$$\Delta H = H(U', \phi, \phi^\dagger, \Pi') - H(U, \phi, \phi^\dagger, \Pi)$$

Various authors $\approx$ 1990

$$P_{acc} = \mathrm{erfc}\left(\sqrt{\mathrm{Var}(\Delta H)/8}\right) = 1 - \frac{1}{\sqrt{2\pi}}\sqrt{\mathrm{Var}(\Delta H)} + \ldots$$

Bussone et al. 2018: For the second order Omelyan with $\lambda = 1/6$:

$$\mathrm{Var}(\Delta H) = \frac{2\delta\tau^4}{72^2}\left[\mathrm{Var}(|\mathcal{F}_{i_{max}}|^2) + \frac{\mathrm{Var}(|\mathcal{F}_{i_{max}-1}|^2)}{(4m^2_{i_{max}-1})^2} + \ldots\right]$$

# Numerical test

Numerical studies at $\beta = 5.6$, $\kappa = 0.156$ and $0.1575$. Extensively studied by SESAM and Gral collaboration (See B. Orth, T. Lippert, and K. Schilling, [arXiv:hep-lat/0503016], Phys. Rev. D **72**, 014503 (2005).)
Used in algorithmic studies e.g. Lüscher 2004, Urbach, Jansen, Shindler and Wenger 2005

$\kappa = 0.156$: $a = 0.09796(56)$ fm, $m_{PS} = 0.9002(69)$ GeV
$\kappa = 0.1575$: $a = 0.0839(11)$ fm, $m_{PS} = 0.6524(86)$ GeV

Real world: $m_{\pi^0} \approx 135$ MeV.

Trajectory length: $\tau = \sqrt{2}$ throughout

$\kappa^k$-filtering, $12^3 \times 24$ lattice, $\beta = 5.6$, $\kappa = 0.156$. Leapfrog scheme.

| k | $m$ | stat | $\langle P \rangle$ | $P_{acc}$ | Var($\Delta H$) |
|---|---|---|---|---|---|
| 0 | 42 | 2770 | 0.56982(7) | 0.8006(43) | 0.2673(54) |
| 2 | 21 | 7050 | 0.56991(6) | 0.7981(26) | 0.2643(43) |
| 4 | 16 | 7610 | 0.56995(4) | 0.8106(24) | 0.2264(40) |

| k | Var($\vert\mathcal{F}_{PF}\vert^2$) |
|---|---|
| 0 | 11400000(200000) |
| 2 | 344000(4000) |
| 4 | 57500(1000) |

$\kappa^4$-filtering, $16^3 \times 32$, $\kappa = 0.1575$: Speed-up by roughly a factor of 3 compared with SESAM, Gral

$\kappa^4$-filtering; Trunction of $\sum_{n=0}^{\infty} b_n \kappa^n H^n$ , force calculation

| $n_t$ | stat | $P_{acc}$ | Var($\Delta H$) |
|---|---|---|---|
| 3 | 200 | 0.22(3) | 4.98(59) |
| 4 | 1030 | 0.177(8) | 8.12(24) |
| 5 | 6400 | 0.8631(22) | 0.1180(31) |
| 6 | 2200 | 0.8506(45) | 0.1512(63) |
| 7 | 2200 | 0.8868(32) | 0.0920(36) |
| 8 | 2000 | 0.8845(31) | 0.0848(29) |
| 9 | 2200 | 0.8851(36) | 0.0904(33) |
| 15 | 24500 | 0.8830(15) | 0.0886(15) |

# Rooted polynomials

$16^3 \times 32$, $\kappa = 0.1575$

Order of the polynomials $n_1 = 8$, $n_2 = 32$
(ad hoc choices to get a first idea)

Roots: $N = 2, 3, 4, 6, 8$, and $16$

Leap-frog integration scheme with multiple time scales

| $N$ | stat | $m_0$ | $m_1$ | $m_2$ | $m$ | $\alpha$ | $\langle P \rangle$ | $P_{acc}$ |
|---|---|---|---|---|---|---|---|---|
| 2 | 290 | 6 | 6 | 4 | 8 | 0.022110... | 0.57279(6) | 0.870(11) |
| 3 | 500 | 10 | 6 | 3 | 6 | 0.020504... | 0.57255(6) | 0.788(9) |
| 4 | 910 | 6 | 5 | 4 | 5 | 0.019687... | 0.57258(3) | 0.793(9) |
| 6 | 600 | 10 | 5 | 2 | 5 | 0.018872... | 0.57256(5) | 0.773(11) |
| 8 | 500 | 6 | 5 | 2 | 5 | 0.018467... | 0.57254(5) | 0.792(12) |
| 16 | 200 | 6 | 5 | 2 | 5 | 0.017866... | 0.57259(6) | 0.806(17) |

Variances of the force:

| $N$ | $G$ | $PF,1$ | $PF,2$ | $PF,3$ |
|---|---|---|---|---|
| 2 | 85000000(4500000) | 1110000(60000) | 11000(900) | 1300(120) |
| 3 | 82000000(4700000) | 290000(15000) | 2020(180) | 540(60) |
| 4 | 84000000(3500000) | 114000(4000) | 710(100) | 360(60) |
| 6 | 77000000(4000000) | 42400(2000) | 197(17) | 156(15) |
| 8 | 79000000(4000000) | 20200(1000) | 81(8) | 123(14) |
| 16 | 83000000(6000000) | 4860(400) | 16.5(3.0) | 156(40) |

Cost index related to the terms $S_{PF,1}$ and $S_{PF,2}$

| $N$ | $8N\ \mathrm{Var}(\lvert\mathcal{F}_{PF,1}\rvert^2)^{1/4}$ | $32N\ \mathrm{Var}(\lvert\mathcal{F}_{PF,2}\rvert^2)^{1/4}$ |
|---|---|---|
| 2 | 519(7) | 655(13) |
| 3 | 557(7) | 644(14) |
| 4 | 588(5) | 661(22) |
| 6 | 689(8) | 719(15) |
| 8 | 763(9) | 768(18) |
| 16 | 1069(21) | 1032(44) |

# Using $\kappa^4$-filtering

$n_1 = 12$, $n_2 = 42$, $N = 8$

$\alpha = 0.01390254...$; For comparison $N = \infty$: $\alpha = 0.01321050...$

$m = 4$, $m_2 = 2$, $m_1 = 3$, $m_0 = 40$, and $n_t = 160$.
The acceptance rate is $P_{acc} = 0.790(10)$ and $\mathrm{Var}(\Delta H) = 0.249(22)$.

$\mathrm{Var}(|\mathcal{F}_{PF,1}|^2) = 2370(160)$
$\mathrm{Var}(|\mathcal{F}_{PF,2}|^2) = 23.8(2.5)$
$\mathrm{Var}(|\mathcal{F}_{PF,3}|^2) = 42(5)$

# Conclusions and Outlook

- Speed-up of factor of 2 or 3 by using $\kappa^2$ and $\kappa^4$ filtering
- Solver can be used to compute the remainder
- Extension to higher orders by using rooted polynomials
- Promising results; strong reduction of forces and their variances
- Many parameters in the game; need more insight to fix them
- New chance for local finite step updates?