Rational hybrid Monte Carlo with block solvers and multiple pseudofermions

Philippe de Forcrand* and Liam Keegan†
Institut für Theoretische Physik, ETH Zürich, CH-8093 Zürich, Switzerland

(Received 14 August 2018; published 18 October 2018)

The dominant cost of most lattice QCD simulations is the inversion of the Dirac operator required to calculate the force term in the rational hybrid Monte Carlo (RHMC) update. One way to improve this situation is to use multiple pseudofermions, which reduces the size and variance of this force and hence allows a larger integration step size to be used. This means fewer force term calculations are required, but at the cost of having to invert the Dirac operator for each pseudofermion field. This bottleneck can be addressed: recently there has been renewed interest in the use of block Krylov solvers, which can solve multiple right-hand-side vectors with significantly fewer iterations than are required if each vector is solved using a separate Krylov solver. We combine these two ideas, achieving a significant speed-up of RHMC lattice QCD simulations.

DOI: 10.1103/PhysRevE.98.043306

I. INTRODUCTION

The main difficulty in lattice simulations of QCD is calculating the determinant of the Dirac operator, a very large and badly conditioned matrix. In the rational hybrid Monte Carlo [1–3] (RHMC) approach, this determinant is stochastically estimated by inverting the Dirac operator acting on a bosonic field of "pseudofermions" using an iterative Krylov solver. The RHMC evolution requires the numerical integration of the pseudofermion force term, and when this term is large or has a large variance a small integrator step size must be used, resulting in many costly pseudofermion force calculations.

Many different approaches have been proposed to reduce the computing cost of RHMC. They range from preconditioning the solver (e.g., even-odd [4,5], domain decomposition [6,7], deflation [8], multigrid [9,10]) to preconditioning the action (ILU [11], UV-filtering [12]) to tuning the integrator ([13–15]).

In particular, a popular strategy which reduces the RHMC fermionic force term is the "Hasenbusch trick" or "mass splitting," and its generalizations [16,17]. One replaces the Dirac matrix M by $(MH^{-1})H$, where H is associated with a heavy fermion, and represents each of the two determinants by a pseudofermion integral. The value of the heavy mass can be tuned to minimize the computer cost per accepted hybrid Monte Carlo (HMC) trajectory. This tuning becomes more challenging in the case of multiple mass splittings; an empirical rule consists of adjusting the magnitude of the pseudofermion forces to be the same for each factor. A simple way to obtain a similar effect is to replace M with $[M^{1/n_{\rm pf}}]^{n_{\rm pf}}$ [18] and represent each of the $n_{\rm pf}$ determinants by a pseudofermion integral. The resulting force magnitude is automatically the same for all factors, and only one parameter, the number $n_{\rm pf}$ of pseudofermions, needs to be adjusted. The cost is that the Dirac operator must be inverted on $n_{\rm pf}$ pseudofermion vectors for each force term calculation.

Recently there has been renewed interest [19–25] in the use of block Krylov solvers [26], which invert the same matrix on multiple vectors simultaneously, and, thanks to the enlarged Krylov basis from which solutions are constructed, can converge with significantly fewer iterations than are required to solve each vector separately.

Here we combine these two ideas to speed up the RHMC algorithm.

II. MULTIPLE PSEUDOFERMIONS

The partition function we want to sample, for $N_{\rm f}$ degenerate-mass quarks, is given by

$$\mathcal{Z} = \int dU\, e^{-S_{\rm g}} \det[M^\dagger M]^{N_{\rm f}/2} = \int dU\, e^{-S_{\rm g}-S_{\rm f}^{\rm eff}}, \qquad (1)$$

where $S_{\rm g}$ is the gauge action and M the Dirac operator, and both are functions of the gauge field U. To sample this using HMC requires the calculation of the fermionic force term,

$$F^a_{x\mu} = -\frac{\partial S_{\rm f}^{\rm eff}}{\partial U^a_{x\mu}} = \mathrm{Tr}\left[(M^\dagger M)^{-\frac{N_{\rm f}}{2}} \frac{\partial (M^\dagger M)^{\frac{N_{\rm f}}{2}}}{\partial U^a_{x\mu}}\right], \qquad (2)$$

where a is the color index, x the site index, and μ the direction index. This would require the entire Dirac operator to be diagonalized. To avoid doing this, the determinant can be written as an integral over bosonic pseudofermion fields ϕ, which gives (up to an overall constant) the equivalent partition function

$$\mathcal{Z} = \int dU\, d\phi\, d\phi^\dagger e^{-S_{\rm g}-\phi^\dagger [M^\dagger M]^{-N_{\rm f}/2}\phi}, \qquad (3)$$

where pseudofermions with the desired distribution can be generated by first sampling η from a normal distribution, then constructing $\phi = [M^\dagger M]^{N_{\rm f}/4}\eta$. The fractional powers of $M^\dagger M$ acting on a vector can in all cases be approximated to

*forcrand@phys.ethz.ch
†keeganl@phys.ethz.ch

any desired accuracy by use of a suitable rational approximation of the form

$$[M^\dagger M]^r x \simeq \alpha_0 x + \sum_{j=1}^{N_{\text{shifts}}} \alpha_j (M^\dagger M + \beta_j)^{-1} x, \tag{4}$$

where the coefficients $\alpha_j, \beta_j > 0$ and the number of shifts N_{shifts} depend on the exponent r, the spectral range of the Dirac operator, and the desired accuracy of the approximation.

This approach can be extended to multiple pseudofermions; using the trivial identity

$$\det[M^\dagger M] = \det[(M^\dagger M)^{\frac{1}{n_{\text{pf}}}}]^{n_{\text{pf}}}, \tag{5}$$

the partition function can instead be written as

$$\mathcal{Z} = \int dU \prod_{i=1}^{n_{\text{pf}}} (d\phi_i d\phi_i^\dagger) \times \exp\left(-S_{\text{g}} - \sum_{i=1}^{n_{\text{pf}}} \phi_i^\dagger [M^\dagger M]^{-\frac{N_{\text{f}}}{2n_{\text{pf}}}} \phi_i\right), \tag{6}$$

where η_i are again sampled from a normal distribution, and $\phi_i = [M^\dagger M]^{N_{\text{f}}/4n_{\text{pf}}} \eta_i$.

The resulting pseudofermion force term for n_{pf} pseudofermions is given by

$$F^a_{x\mu}(\phi_i, U, n_{\text{pf}}) = \sum_{i=1}^{n_{\text{pf}}} \phi_i^\dagger \frac{\partial [M^\dagger M]^{-\frac{N_{\text{f}}}{2n_{\text{pf}}}}}{\partial U^a_{x\mu}} \phi_i. \tag{7}$$

For a given gauge field U, writing the ϕ_i fields in terms of the Gaussian η_i fields, then integrating over them in Eq. (7), we recover the correct expectation value of the force term, Eq. (2), which is independent of n_{pf},

$$\overline{F^a_{x\mu}(U, n_{\text{pf}})} \equiv \int \prod_{i=1}^{n_{\text{pf}}} [p(\eta_i) d\eta_i] F^a_{x\mu}\left([M^\dagger M]^{\frac{N_{\text{f}}}{4n_{\text{pf}}}} \eta_i, U, n_{\text{pf}}\right) = \text{Tr}\left[(M^\dagger M)^{-\frac{N_{\text{f}}}{2}} \frac{\partial (M^\dagger M)^{\frac{N_{\text{f}}}{2}}}{\partial U^a_{x\mu}}\right], \tag{8}$$

with a variance that is suppressed by n_{pf},

$$\left[\overline{F^a_{x\mu}(U, n_{\text{pf}})^2}\right] - \left[\overline{F^a_{x\mu}(U, n_{\text{pf}})}\right]^2 = \frac{c_1}{n_{\text{pf}}} + \mathcal{O}\left(n_{\text{pf}}^{-2}\right), \tag{9}$$

where c_1 does not depend on n_{pf}. In simulations we can easily measure the norm F^2 of this pseudofermion force,

$$F^2(n_{\text{pf}}) = \left\langle \sum_{ax\mu} \frac{1}{2} \left[F^a_{x\mu}(\phi_i, U, n_{\text{pf}})\right]^2 \right\rangle, \tag{10}$$

where $\langle \cdots \rangle$ represents an average over the gauge fields. Moreover, for the particular choice of the second-order Omelyan (OMF2) [13,27] integrator with $\lambda = 1/6$, the variance of this norm is related to the variance of the energy violation ΔH [28] over a trajectory of length τ with integrator step size $\delta\tau = \tau/n_{\text{steps}}$,[1]

$$\text{var}[\Delta H] = 8\left(\frac{\delta\tau}{12}\right)^4 \text{var}[F^2(n_{\text{pf}})] + \mathcal{O}(\delta\tau^6). \tag{11}$$

This relation is valid up to higher order corrections in the step size, and assumes that the trajectory length is long enough that the correlation between initial and final force terms can be neglected. Here we also assume that a multiscale integrator [29] is used such that the gauge force term's contribution to the integrator error is negligible. This variance in the trajectory energy violation can in turn be related to the acceptance P_{acc} using the Creutz acceptance formula [30,31]

$$P_{\text{acc}}(\Delta H) = \text{erfc}(\sqrt{\text{var}[\Delta H]/8}), \tag{12}$$

which is valid for high acceptances. Combining the two and expanding in $\delta\tau$ gives a simple prediction for the acceptance,

$$P_{\text{acc}} = 1 - \frac{1}{72\sqrt{\pi}} \delta\tau^2 \sqrt{\text{var}[F^2(n_{\text{pf}})]} + \mathcal{O}(\delta\tau^4), \tag{13}$$

and, assuming that the total trajectory cost is dominated by the force term inversions, the relative cost $C(n_{\text{pf}})$ of simulations at different n_{pf} can be estimated as the cost of a force term inversion ($\propto n_{\text{pf}}$) multiplied by the number of inversions ($\propto 1/\delta\tau$),

$$C(n_{\text{pf}}) \propto n_{\text{pf}}/\delta\tau \propto n_{\text{pf}} \{\text{var}[F^2(n_{\text{pf}})]\}^{1/4}, \tag{14}$$

which we can use to cheaply estimate the relative performance of simulations using different values of n_{pf} simply by measuring the variance of the force term for each n_{pf} on the same set of thermalized configurations. Another estimate for the cost is given in Ref. [18],

$$C(n_{\text{pf}}) \propto n_{\text{pf}}^2 \kappa^{\frac{1}{n_{\text{pf}}}}, \tag{15}$$

where κ is the condition number of the Dirac operator. We will compare these simple estimates with the actual cost of simulations for different n_{pf} in Sec. IV. For large values of n_{pf} Eq. (9) gives the n_{pf} dependence of the force norm as

$$F^2(n_{\text{pf}}) = c_0 + c_1 n_{\text{pf}}^{-1} + \mathcal{O}\left(n_{\text{pf}}^{-2}\right), \tag{16}$$

and similarly for the variance of this norm one finds

$$\text{var}[F^2(n_{\text{pf}})] = c_2 n_{\text{pf}}^{-1} + c_3 n_{\text{pf}}^{-2} + \mathcal{O}\left(n_{\text{pf}}^{-3}\right), \tag{17}$$

where the constants c_i are expectation values of traces involving the Dirac operator that do not depend on n_{pf}, and in particular $c_0 = F^2$ is the norm of the exact force term of Eq. (2).

We see that increasing n_{pf} reduces this variance, which according to Eq. (13) will allow a larger step size to be used in the integrator, resulting in fewer force term calculations. The lowest shift β_1 in the rational approximation of Eq. (4) also increases with n_{pf}, which makes the inversion of the Dirac operator converge faster. These gains are offset by the cost of inverting the Dirac operator n_{pf} times; however, empirical studies have shown that using intermediate values of $n_{\text{pf}} > 1$ results in a smaller total simulation cost than $n_{\text{pf}} = 1$ [18].

[1]Note that τ may need to be rescaled if the choice of normalization of the kinetic term in the HMC differs from that of Ref. [28].

In the next section we further improve on this idea, taking advantage of the presence of multiple pseudofermions to reduce the cost of these $n_{\rm pf}$ Dirac operator inversions, by combining the pseudofermion vectors at each site on the lattice to form a block matrix (or "pencil"). This has two benefits: applying the Dirac operator to the block matrix is more computationally efficient than applying it to each vector in turn, and the block structure allows the use of a block multishift conjugate gradient (CG) inverter, which requires fewer Dirac operator calls to converge.

III. BLOCK KRYLOV SOLVERS

A Krylov solver iteratively solves the system $Ax = b$ for the vector x given some vector b, where we take A to be a Hermitian positive definite matrix. Starting from some initial guess $x^{(0)}$ with residual $r = b - Ax^{(0)}$, it constructs a solution $x^{(k)}$ after k iterations from the Krylov basis $\mathcal{K}_k = \{r, Ar, A^2r, \ldots, A^{k-1}r\}$. The conjugate gradient (CG) solver is an example of such a Krylov solver; at each step it finds the solution that minimizes the error norm $|e_k|_A \equiv (x^{(k)} - x^*)^\dagger A(x^{(k)} - x^*)$, where x^* is the exact solution.

Since we want to solve for $n_{\rm pf}$ vectors b_j, where $j = 1, 2, \ldots, n_{\rm pf}$, with the *same* Dirac matrix for each vector, we can form a block matrix B whose jth column is b_j, and solve the system $AX = B$. The solution is now constructed from the much larger block-Krylov basis $\mathcal{K}_k = \{R, AR, A^2R, \ldots, A^{k-1}R\}$, where $R = B - AX^{(0)}$, which can potentially converge with significantly fewer iterations. Additionally there can be a performance gain from only having to read the matrix A once per $n_{\rm pf}$ vectors. Extending the CG solver in this way gives the block CG (BCG) algorithm [26], which minimizes $\mathrm{Tr}\,[(X^{(i)} - X^*)^\dagger A(X^{(i)} - X^*)]$ at each step, and is equivalent to CG for $n_{\rm pf} = 1$.

There is an upper bound on the relative error of the BCG solution after k steps [26],

$$\frac{|e_k|_A}{|e_0|_A} \leqslant c_1(n_{\rm pf})\left(\frac{1 - \sqrt{\lambda_{n_{\rm pf}}/\lambda_{\max}}}{1 + \sqrt{\lambda_{n_{\rm pf}}/\lambda_{\max}}}\right)^{2k}, \tag{18}$$

where the eigenvalues of A in ascending order are given by $\{\lambda_1, \lambda_2, \ldots, \lambda_{n_{\rm pf}}, \ldots, \lambda_{\max}\}$, and $c_1(n_{\rm pf})$, where $c_1(1) = 4$, is a function that we will approximate as constant here. Expanding in powers of $\sqrt{\lambda_{n_{\rm pf}}/\lambda_{\max}}$, this can be written as

$$\frac{|e_k|_A}{|e_0|_A} \leqslant c_1(n_{\rm pf})e^{-4k\sqrt{\lambda_{n_{\rm pf}}/\lambda_{\max}}} + \mathcal{O}(k(\lambda_{n_{\rm pf}}/\lambda_{\max})^{3/2}), \tag{19}$$

so we see that the rate of convergence for the block solver goes like $\sim\sqrt{\lambda_{n_{\rm pf}}}$ or, equivalently, the effective "condition number" that governs the convergence of the solver is reduced as $n_{\rm pf}$ is increased. Thus, if we keep the desired error constant, we expect the required number of iterations k to decrease as we increase $n_{\rm pf}$, as seen in Fig. 1.

This solver was proposed nearly 40 years ago [26], and perhaps one reason that it has not become more widely used is its numerical stability. In particular, if the matrix of residuals R becomes badly conditioned the BCG algorithm can fail to converge, while a separate CG solve for each vector for the same system would converge. Several solutions to this issue are proposed in Ref. [32]; we implemented and tested these numerically, reaching the same conclusion that the optimal choice in terms of stability and computational cost is to include a reorthogonalization via QR decomposition of the residual matrix at each iteration, known as the BCGrQ algorithm, as used in Ref. [25].

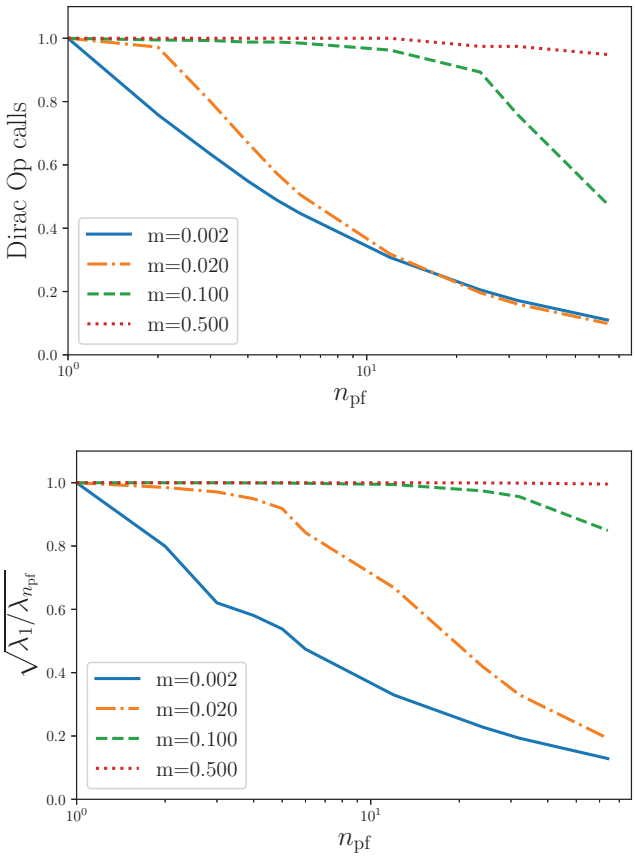

FIG. 1. *Top*: Number of Dirac operator calls for the block SBCGrQ solver to converge compared to the SCG solver, for a range of fermion masses. As the mass is made lighter, the block solver improvement increases. *Bottom*: The square root of the ratio of the lowest eigenvalue to the $n_{\rm pf}$th eigenvalue of the Dirac operator. The bound on the convergence rate of Eq. (19) is determined by this quantity, and qualitatively it also seems to describe the actual convergence of the block solver quite well.

For the RHMC we need a multishift variant of this solver. For CG the shift invariance of the Krylov basis allows the residuals of the shifted systems to be related to the residuals of the unshifted one, leading to the multishift CG (SCG) algorithm [33,34]. The same can be done for the BCGrQ algorithm, which leads to the SBCGrQ [35] multishift block solver. The main difference to the multishift CG solver is that in the block case the relations between shifted and unshifted systems involve $n_{\rm pf} \times n_{\rm pf}$ matrices instead of scalars.

It is instructive to consider how the bound on the error, Eq. (19), changes for the shifted matrix $A + \sigma$, in particular for the case where $\sigma \gg \lambda_{n_{\rm pf}}$,

$$\begin{aligned}\frac{|e_k|_{A+\sigma}}{|e_0|_{A+\sigma}} &\lesssim c_1(n_{\rm pf})e^{-4k\sqrt{(\sigma+\lambda_{n_{\rm pf}})/(\sigma+\lambda_{\max})}} \\ &\lesssim c_1(n_{\rm pf})e^{-4k\sqrt{\sigma/(\sigma+\lambda_{\max})}[1+\mathcal{O}(\lambda_{n_{\rm pf}}/\sigma)]}.\end{aligned} \tag{20}$$

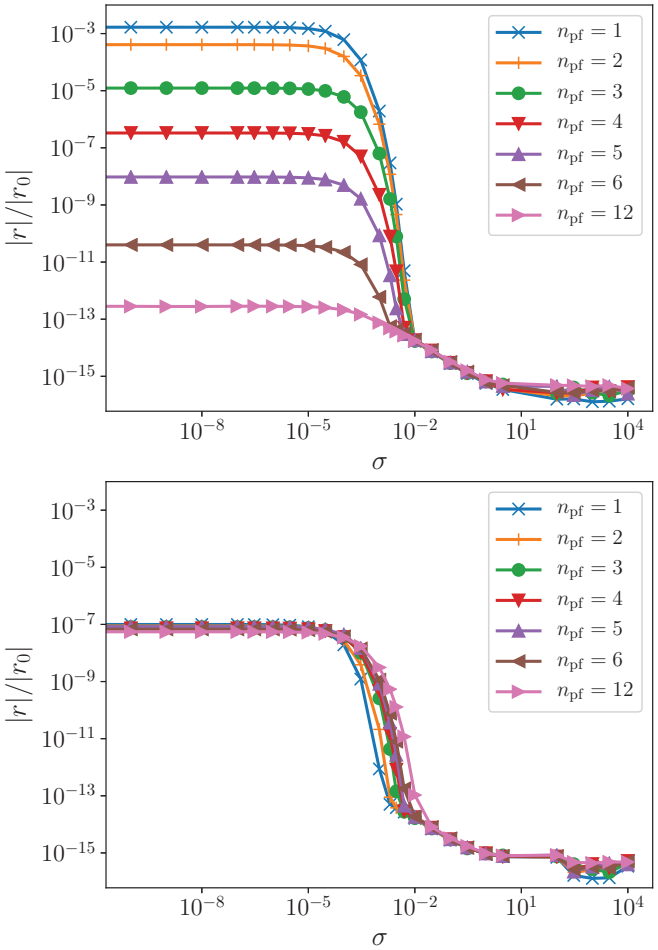

FIG. 2. *Top*: Residual of shifted solution versus shift σ, for fixed $k = 400$ solver iterations. We see a dramatic decrease in the residual for small shifts as $n_{\rm pf}$ is increased. *Bottom*: Same quantity but keeping fixed the unshifted residual $|r|/|r_0| = 10^{-7}$. We see that the reduction in iterations leads to a relative increase in the residuals of the larger shifts with $n_{\rm pf}$.

Here we see that to leading order the convergence rate does not depend on $\lambda_{n_{\rm pf}}$, but only on the size of the shift σ and the number of steps k. From Eq. (19) we expect that the number of steps k required for a given error on the unshifted solution decreases with $n_{\rm pf}$. Equation (20) suggests that, as a side effect, the error on shifted solutions with large shifts will increase with $n_{\rm pf}$, as shown in Fig. 2.

The formulation of SBCGrQ used here is described in Algorithm 1. It is numerically equivalent to Ref. [35], but we use a pair of two-term coupled recursion relations instead of a single three-term recursion relation to calculate the shift matrices, which we find improves the numerical accuracy of the shifted solutions for very badly conditioned systems [36]. The updating of a shifted solution can be stopped once the relative norm of its residual, $\sqrt{\sum_j \delta_k^{(s)}(i,j)/\sum_j \delta_0(i,j)}$, is less than machine precision, where $\delta_k^{(s)} = \rho_k \alpha_k^{-1} \alpha_k^{(s)}$; see Algorithm 1 [37]. Compared to BCGrQ, each shifted solution requires two additional block vectors to be stored, and two additional multiply-add operations (lines 12 and 13 of Algorithm 1) involving these block vectors at each iteration. There are also some extra $n_{\rm pf} \times n_{\rm pf}$ matrix operations (lines 10 and 11 of Algorithm 1) that have negligible storage and computational impact. The expression $\{Q, R\} = \mathrm{qr}(B)$ in Algorithm 1 refers to a thin QR decomposition of the matrix B into an orthogonal matrix Q and an upper-triangular matrix R such that $QR = B$, as described for example in Ref. [25].

Algorithm 1 SBCGrQ: Solve $(A + \sigma_s)X^{(s)} = B$ for $s = 0, 1, \ldots, N_{\rm shifts} - 1$.

1: $X^{(s)}, P^{(s)}, Q, \in \mathcal{C}^{L \times n_{\rm pf}}; \alpha, \rho, \delta, \alpha^{(s)}, \beta^{(s)} \in \mathcal{C}^{n_{\rm pf} \times n_{\rm pf}}$
2: $X_0^{(s)} = 0, \{Q_0, \delta_0\} = \mathrm{qr}(B), P_0^{(s)} = Q_0;$
$\rho_0 = \delta_0, \alpha_0 = \alpha_0^{(s)} = \beta_0^{(s)} = 1$
3: **for** $k = 1, 2, \ldots$ until $\sqrt{\sum_j \delta_k(i,j)/\sum_j \delta_0(i,j)} < \epsilon \; \forall i$ **do**
4: $\alpha_k \leftarrow (P_{k-1}^{(0)\dagger}(A + \sigma_0)P_{k-1}^{(0)})^{-1}$
5: $\{Q_k, \rho_k\} \leftarrow \mathrm{qr}\,(Q_{k-1} - (A + \sigma_0)P_{k-1}^{(0)}\alpha_k)$
6: $X_k^{(0)} \leftarrow X_{k-1}^{(0)} + P_{k-1}^{(0)}\alpha_k\delta_{k-1}$
7: $P_k^{(0)} \leftarrow Q_k + P_{k-1}^{(0)}\rho_k^\dagger$
8: $\delta_k \leftarrow \rho_k\delta_{k-1}$
9: **for** $s = 1, \ldots, N_{\rm shifts} - 1$ **do**
10: $\beta_k^{(s)} \leftarrow (1 + (\sigma_s - \sigma_0)\alpha_k + \alpha_k\rho_{k-1}\alpha_{k-1}^{-1}(1 - \beta_{k-1}^{(s)})\rho_{k-1}^\dagger)^{-1}$
11: $\alpha_k^{(s)} \leftarrow \beta_k^{(s)}\alpha_k\rho_{k-1}\alpha_{k-1}^{-1}\alpha_{k-1}^{(s)}$
12: $X_k^{(s)} \leftarrow X_{k-1}^{(s)} + P_{k-1}^{(s)}\alpha_k^{(s)}$
13: $P_k^{(s)} \leftarrow Q_k + P_{k-1}^{(s)}\beta_k^{(s)}\rho_k^\dagger$
14: **end for**
15: **end for**

IV. RESULTS

As an initial numerical study of the method we simulate $N_{\rm f} = 4$ QCD using unimproved staggered fermions with even-odd preconditioning and the Wilson gauge action, on lattices of size 8^4, with gauge coupling $\beta = 5.12$ and fermion mass $am = 0.002$. These parameters are chosen to have a small mass while remaining in the confined phase of this theory [38], and the choice $N_{\rm f} = 4$ allows a direct comparison to HMC for the case $n_{\rm pf} = 1$ while avoiding any issues related to rooting. These small-scale simulations allow us to perform many simulations with different parameters and investigate a wide range of values of $n_{\rm pf}$ and integrator step sizes, as well as to perform very long simulations to study the integrated autocorrelation times of measured observables.

For the molecular dynamics force term we use a stopping criterion $|r|/|r_0| < 10^{-7}$ for the solver, and a rational approximation with relative error $<10^{-7}$ and $N_{\rm shifts} \simeq 15$, while for the heat-bath and accept/reject steps the stopping criterion is 10^{-14}, and the rational approximation has relative error $<10^{-15}$ and $N_{\rm shifts} \simeq 30$. We use a two-level OMF2 integrator, setting $\lambda = 1/6$ in order to compare with the predicted acceptance rates of Eq. (13). For each pseudofermion integration step the gauge force is integrated with at least three steps, such that its contribution to the integrator error is negligible. For $n_{\rm pf} = 1$–6 we ran 5000 $\tau = 1$ trajectories for a wide range of integrator step sizes, whose acceptance rates are shown in Fig. 3, along with the predicted acceptance rates using Eq. (13). For high acceptance rates and small integrator step size $\delta\tau$, where Eq. (13) is valid, the measured values are in

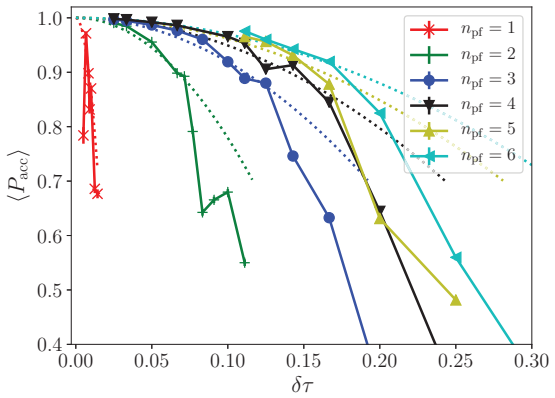

FIG. 3. Measured expectation values of acceptance rate (solid lines), compared with the predicted acceptance rate from measured force variances using Eq. (13) (dotted lines). For high acceptance and small $\delta\tau$ the agreement is reasonable; it turns out the difference between the prediction and the measured values is largely due to the neglected correlation between initial and final force terms not being negligible in these data, so increasing the trajectory length would improve the agreement.

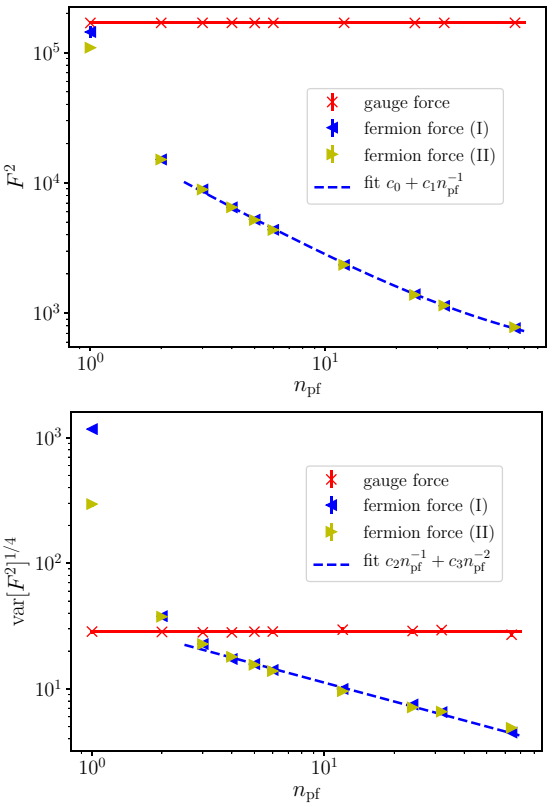

FIG. 4. Gauge and fermion force norms versus $n_{\rm pf}$, with large-$n_{\rm pf}$ scaling predictions. *Top*: Force norms with a fit to Eq. (16). *Bottom*: Fourth root of variance of force norms, approximately proportional to the number of integration steps required for the OMF2 integrator, along with a fit to Eq. (17). Force (I) is measured at every integration step along the trajectory, while Force (II) is measured on the same set of 2000 thermalized configurations.

reasonable agreement with the prediction; the main source of the difference between the two in this case is the neglected contribution from the correlation between initial and final force terms in a trajectory, which is not negligible in our simulations. Increasing the trajectory length would suppress this contribution and improve the agreement between the predicted and measured acceptance rates. We also performed some additional shorter runs at larger $n_{\rm pf}$ up to $n_{\rm pf} = 64$.

To study the $n_{\rm pf}$ dependence of the distribution of ΔH and of various observables and their autocorrelation times, we performed a single long run for each $n_{\rm pf} \leqslant 6$ as described in Table I, using the OMF2 integrator setting $\lambda = 0.20$. The expectation value of the plaquette is consistent within errors for all $n_{\rm pf}$. Its integrated autocorrelation time also exhibits no clear dependence on $n_{\rm pf}$, nor did the various other smeared and unsmeared gauge observables that we measured.

A. Multiple pseudofermions

Increasing $n_{\rm pf}$ reduces both the size and the variance of the norm of the pseudofermion force term. Fig. 4 shows these quantities for both gauge and pseudofermion fields as a function of $n_{\rm pf}$. The large variance of the fermionic force comes from the poor accuracy of this pseudofermion estimate; for small $n_{\rm pf}$ it is orders of magnitude larger than the exact (large-$n_{\rm pf}$ limit) value: $c_0/c_1 \sim 10^{-3}$ in Eq. (16). The blue left-facing triangles with error bars are measured for every force term calculation during the simulation, while the yellow right-facing triangles with error bars are measured on a set of 2000 thermalized configurations. For $n_{\rm pf} > 1$, the two measurements agree within errors, but for $n_{\rm pf} = 1$ they differ significantly. This is caused by infrequent but very large spikes in the force for $n_{\rm pf} = 1$, which means that many more than 2000 measurements would be required to reliably estimate the variance of the force in this case. Also shown is a fit to the large-$n_{\rm pf}$ form predicted by Eqs. (16) and (17), which seems to provide a good description of the data for $n_{\rm pf} \gtrsim 3$.

A histogram of the values of the pseudofermion rms force is shown in the top panel of Fig. 5, where for $n_{\rm pf} = 1$ the distribution is clearly non-Gaussian, with a long tail of large values. As $n_{\rm pf}$ is increased, the mean and variance of the distribution of force norms decrease, as already seen in Fig. 4, and in addition the form of the distribution becomes closer to a Gaussian, without a long tail of values much larger than the mean. Since empirically we find $c_0 \ll c_1$ and $c_2 \ll c_3$ in Eqs. (16) and (17), we can expect the quantity $n_{\rm pf} F^2(n_{\rm pf})$ to have approximately $n_{\rm pf}$-independent mean and variance for some intermediate range of values of $n_{\rm pf}$. This quantity

TABLE I. Run parameters for the longer simulations, with $n_{\rm steps}$ tuned such that $\langle P_{\rm acc}\rangle \simeq 0.96$. The integrated autocorrelation time of the plaquette does not appear to depend on $n_{\rm pf}$.

$n_{\rm pf}$	$n_{\rm steps}$	$\langle P_{\rm acc}\rangle$	$\langle e^{-\Delta H}\rangle$	$\langle$plaq$\rangle$	$\tau_{\rm int}$	$n_{\rm trajectories}$
1	250	0.961(11)	0.9701(100)	0.52268(14)	5	5×10^3
2	16	0.942(5)	0.9920(28)	0.52283(6)	4	28×10^3
3	11	0.965(1)	0.9998(6)	0.52288(8)	5	33×10^3
4	9	0.966(1)	1.0005(5)	0.52297(6)	4	26×10^3
5	8	0.960(1)	0.9994(7)	0.52272(8)	5	25×10^3
6	7	0.954(2)	1.0006(8)	0.52277(10)	6	21×10^3

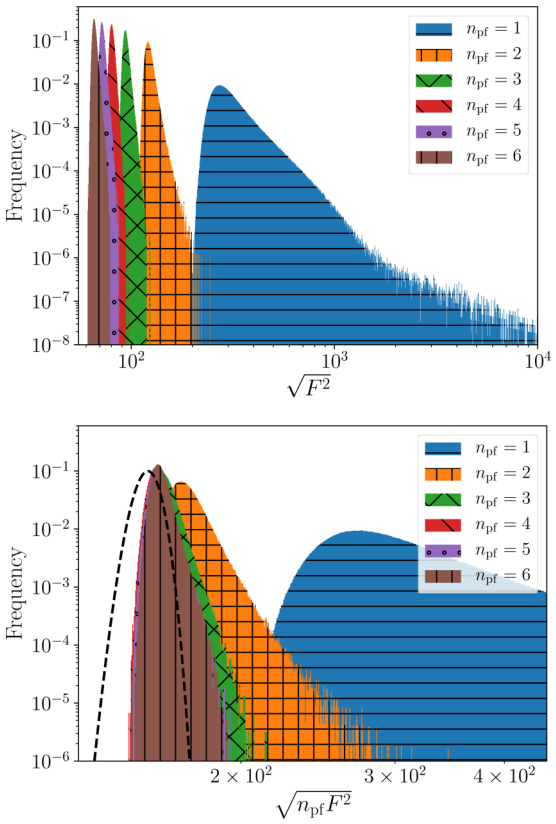

FIG. 5. *Top*: Histogram of the rms pseudofermion force norm, $\sqrt{F^2}$, for different $n_{\rm pf}$. For $n_{\rm pf} = 1$ the distribution is very non-Gaussian, with a long tail of large values. *Bottom*: Histogram of $\sqrt{n_{\rm pf} F^2}$ wich shows an approximate $n_{\rm pf}$ invariance for intermediate values of $n_{\rm pf}$, due to the c_1 and c_3 terms dominating Eqs. (16) and (17) for these values of $n_{\rm pf}$.

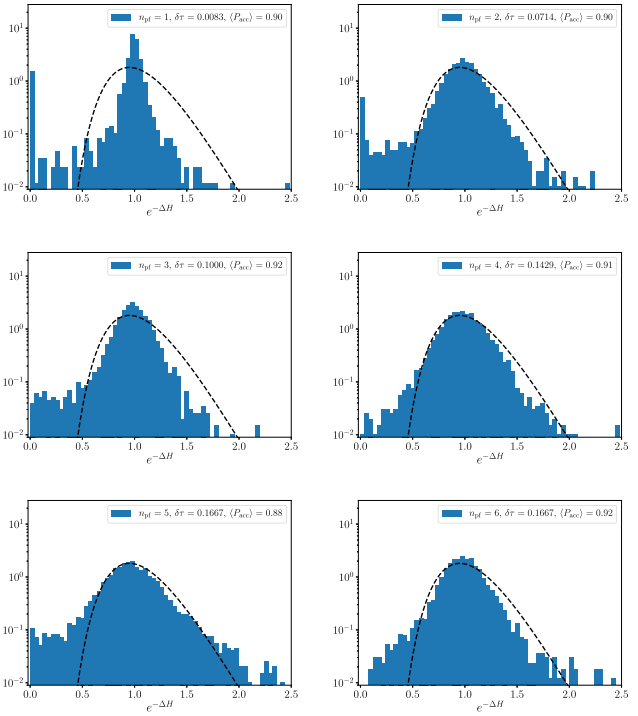

FIG. 6. Histogram of $e^{-\Delta H}$ for $n_{\rm pf} = 1$ to 6 with the acceptance rate tuned to $\simeq$90%. The black dotted line shows the prediction for a Gaussian distribution of ΔH with the same acceptance rate. For $n_{\rm pf} = 1$ (top left) the distribution is very far from gaussian, with an excess of very small values, but as $n_{\rm pf}$ is increased the distribution approaches the Gaussian one.

is shown in the bottom panel of Fig. 5, which shows this approximate scaling for intermediate $n_{\rm pf}$, along with a dotted black line showing a Gaussian distribution with the same mean and variance.

Another way to see the improvement from using multiple pseudofermions is to look at the distribution of $e^{-\Delta H}$, where ΔH is the energy violation of the trajectory. Figure 6 shows the distribution of this quantity for $n_{\rm pf} = 1$ to 6, with the integrator step size tuned such that the acceptance is $\simeq$90% for each. The distribution expected for this acceptance rate assuming a Gaussian distribution for ΔH is also shown, and as $n_{\rm pf}$ is increased the measured distribution becomes closer to the Gaussian one. For the case $n_{\rm pf} = 1$, the distribution of ΔH is very far from Gaussian, with an excess of tiny values of $e^{-\Delta H}$ which reflect the large fluctuations in the force term. Such "exceptional configurations" can trigger an instability of the integrator, which makes the Monte Carlo error analysis more delicate and may introduce long autocorrelation times.

Using Eq. (14) we can use the variance of the pseudofermion force norm to predict the approximate cost of generating an RHMC trajectory as a function of $n_{\rm pf}$. Another prediction of the cost using the condition number of the Dirac operator is given by Eq. (15). These predictions are compared to the measured cost of actual simulations using the multishift CG solver, with the integrator step size tuned to make the acceptance rate $\simeq$90%. The results are shown in Fig. 7, where all costs are normalized to 1 for the case $n_{\rm pf} = 1$. There is a large reduction in the cost for $n_{\rm pf} = 2$ compared to $n_{\rm pf} = 1$, followed by a gradual increase in the cost with $n_{\rm pf}$.

In this section we have shown that using multiple pseudofermions with the usual multishift CG solver significantly reduces the mean and variance of the pseudofermion force term, which both speeds up RHMC simulations and results in a much more Gaussian distribution of ΔH. In the next section we take advantage of having multiple pseudofermions to store them in block form, which allows us to make use of a more efficient, block version of the multishift CG solver and also increases the computational efficiency of the Dirac operator.

B. Block solvers

Block solvers have been shown to provide large speed-ups in two recent lattice QCD studies of inverting the Dirac operator with multiple right-hand-side (RHS) vectors [21,25]. There are two sources of this speed-up: one is that as the number of RHS vectors ($n_{\rm pf}$ in our case) is increased the number of iterations required for the solver to converge decreases, the other is that applying the Dirac operator to a block of vectors is significantly faster, since the cost of loading the gauge links is amortized over the many RHS vectors, and these data are contiguous, allowing better use of the CPU cache.

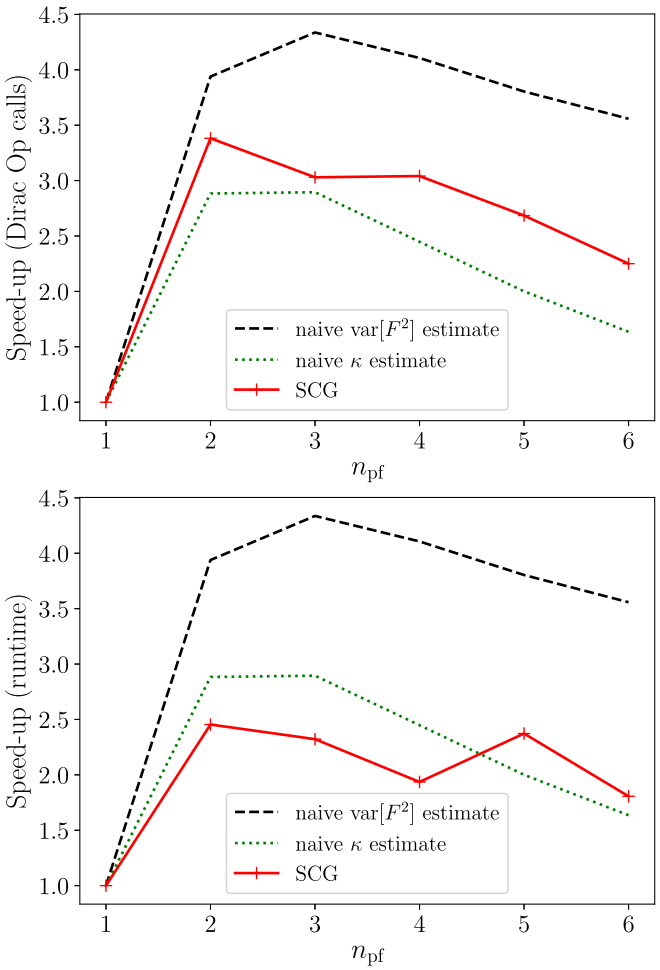

FIG. 7. Trajectory speed-up versus $n_{\rm pf}$, normalized to 1 for $n_{\rm pf} = 1$. At each step the multishift CG (SCG) solver is used $n_{\rm pf}$ times. The black dashed line is the simple prediction from the variance of the pseudofermion force norm using Eq. (14), and the green dotted line is the simple prediction from the condition number of the Dirac operator using Eq. (15). Going from $n_{\rm pf} = 1$ to $n_{\rm pf} = 2$ gives a significant cost reduction, but increasing $n_{\rm pf}$ further results in a larger cost per trajectory.

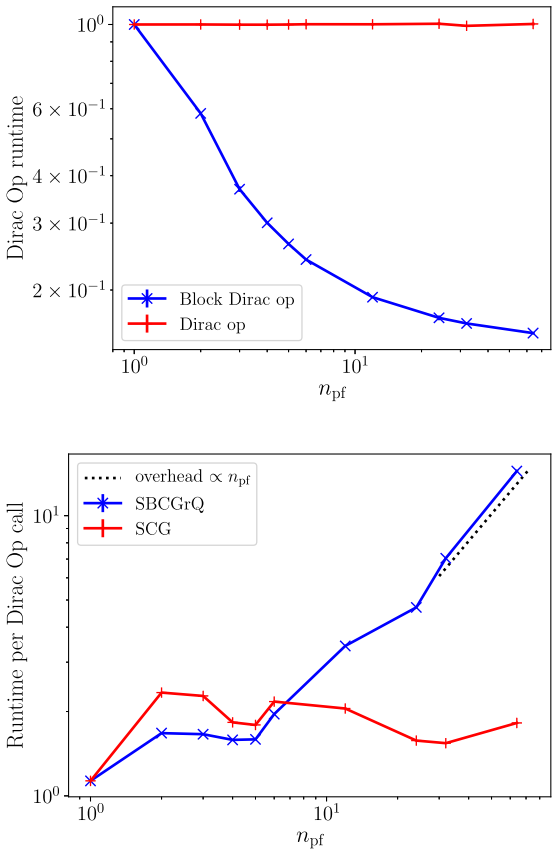

FIG. 8. *Top*: Runtime of Dirac operator acting on vectors in block form, normalized to the nonblock form. *Bottom*: Solver runtime per Dirac operator call versus $n_{\rm pf}$. For small $n_{\rm pf}$ one iteration of block multishift SBCGrQ is much faster than multishift SCG since the block Dirac operator is faster. For large enough $n_{\rm pf}$, however, the SBCGrQ solver overhead that grows $\propto n_{\rm pf} N_{\rm shifts}$ eventually dominates the cost.

However, there is a cost that comes with these benefits, which is that all pseudofermion vector operations in the solver are promoted to matrix operations in the block solver, and this overhead grows with a factor $n_{\rm pf}$ compared to the cost of applying the Dirac operator. Figure 8 compares the runtime of block and nonblock versions of a single Dirac operator call and a single iteration of the two multishift solvers used in this work: multishift CG (SCG) and block multishift CG (SBCGrQ). The top panel shows that the block Dirac operator is significantly faster than the nonblock version. In the bottom panel, for $n_{\rm pf} \leqslant 6$ one iteration of the block multishift solver SBCGrQ is also faster than multishift CG for the same reason, because the cost is dominated by the Dirac operator. For very large $n_{\rm pf}$ the overhead becomes significant however, and can be seen to dominate the cost of a single SBCGrQ iteration for $n_{\rm pf} \gtrsim 20$.

Figure 9 compares the cost of calculating the pseudofermion force term using the block multishift CG (SBCGrQ) solver with pseudofermions in block form against the previous results using the multishift CG (SCG) solver. We see a large reduction in both the number of Dirac operator calls and the overall runtime. The overhead of the SBCGrQ algorithm will eventually dominate the cost at large $n_{\rm pf}$, but, as we already saw in Fig. 8, for the region of interest, $n_{\rm pf} \lesssim 6$, this overhead is not prohibitive. It is also possible when using the block solver to take the stopping criterion for the force solves to be very small without a significant increase in cost, which reduces the potential reversibility violations caused by finite precision, which may be a concern for badly conditioned systems or if the RHMC trajectory length τ is increased [39].

At the start and end of a trajectory, a high precision inversion must also be done, and Fig. 10 compares the cost of this step between the original and block methods, and we again see a large improvement from the block version.

So far we have compared solvers for different $n_{\rm pf}$ while keeping the residual of the lowest shift the same, but from Eq. (20) we can also expect the residuals of the shifted solutions to depend on $n_{\rm pf}$. Figure 2 shows the residual of shifted solutions using the SBCGrQ solver (for $n_{\rm pf} = 1$ this reduces to the SCG solver), for a wide range of shifts σ.

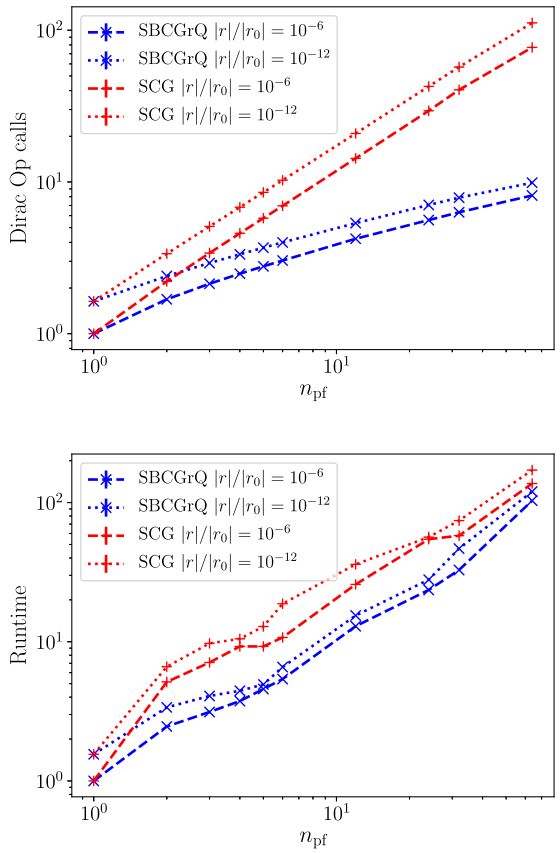

FIG. 9. Cost of calculating the pseudofermion force term versus $n_{\rm pf}$, using either multishift (SCG) or block multishift (SBCGrQ) solvers with stopping criterion 10^{-6} or 10^{-12}. The block solver is a significant improvement, moreover it allows the use of a very tight stopping criterion without significant extra cost, which reduces possible reversibility violations.

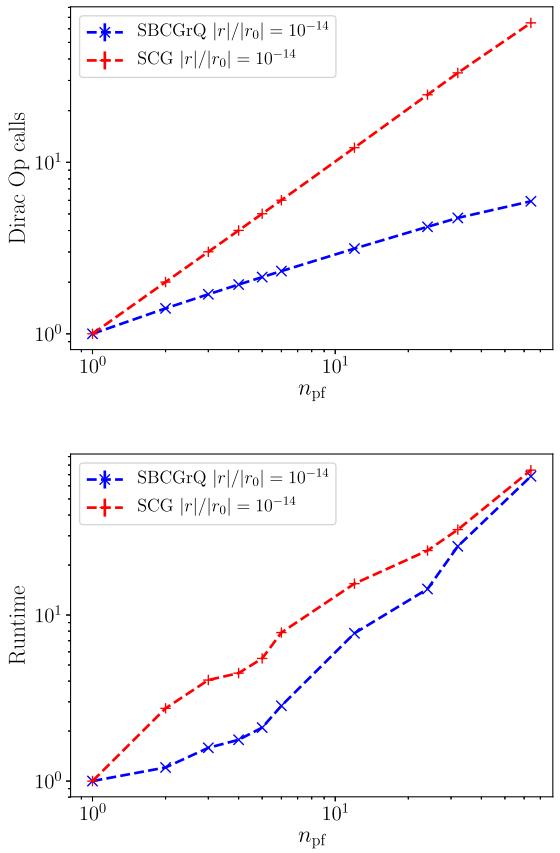

FIG. 10. Cost of calculating the pseudofermion action versus $n_{\rm pf}$, using either multishift (SCG) or block multishift (SBCGrQ) solvers with stopping criterion 10^{-14}. This is done twice per trajectory: at the start for the heat bath and at the end for the accept/reject step. The block solver significantly reduces the cost of this step.

In the top panel, the number of solver iterations k is kept constant, and we see the residuals for small shifts decrease dramatically as $n_{\rm pf}$ is increased, which is consistent with the expectation from Eq. (19). In the bottom panel, the number of solver iterations is adjusted such that the unshifted relative residual is $|r|/|r_0| \simeq 10^{-7}$. Here we see a relative increase in the shifted residuals for intermediate shifts, as predicted by Eq. (20), since fewer iterations are required as $n_{\rm pf}$ is increased. For large values of $n_{\rm pf}$ this might mean that a tighter residual for the force term inversions will be required to maintain the accuracy of the force term, but we saw no such issues in our runs for $n_{\rm pf} \leqslant 6$ where we use the same stopping criterion for all $n_{\rm pf}$.

C. Combined results

Combining our results from the previous two sections, we can measure the cost of generating an accepted RHMC trajectory in two ways. One is in terms of Dirac operator calls per trajectory divided by the acceptance rate, which is implementation independent but does not take into account the acceleration of the Dirac operator or the overhead of the multishift block solver. The second measure of the cost is simply the CPU time required by our reference implementation (running on a single thread of a CPU) to generate a trajectory, divided by the acceptance rate. This takes all the costs into account, but the results are now heavily implementation dependent, and, as our implementation is not parallelized and prioritises flexibility over performance, the results may be significantly different on a fully optimized production lattice QCD code. Moreover, GPU-based hardware with a higher ratio of compute performance to memory bandwidth should benefit more from the increased arithmetic intensity of the block Dirac operator.

Both measures of the cost are shown in Fig. 11 as a function of the integrator step size for $n_{\rm pf} = 1$ to 6, using the SBCGrQ inverter and block Dirac operator. For both cost measures there is a clear benefit from increasing $n_{\rm pf}$ to 3 or 4. The optimal integrator step size for each $n_{\rm pf}$ in this plot corresponds to a $\simeq$90% acceptance rate. Taking these optimal integrator step sizes, we can compare the overall improvement the block method offers compared to the previous nonblock results of Sec. IV A, which is shown in Fig. 12. We see a $\sim 6\times$ speed-up using $n_{\rm pf} = 4$ compared to HMC, while the nonblock multishift CG solver gave a $\sim 3\times$ speed-up using $n_{\rm pf} = 2$.

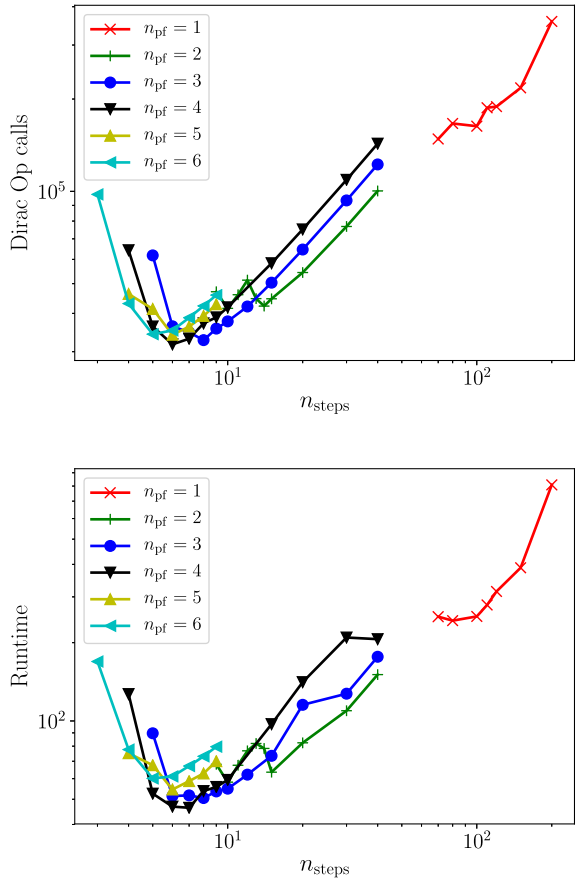

FIG. 11. The cost of generating an accepted $\tau = 1$ trajectory using the block multishift (SBCGrQ) inverter for different $n_{\rm pf}$ versus the number of integrator steps $n_{\rm steps} = \tau/\delta\tau$. *Top*: cost in Dirac operator calls; *bottom*: computer runtime cost.

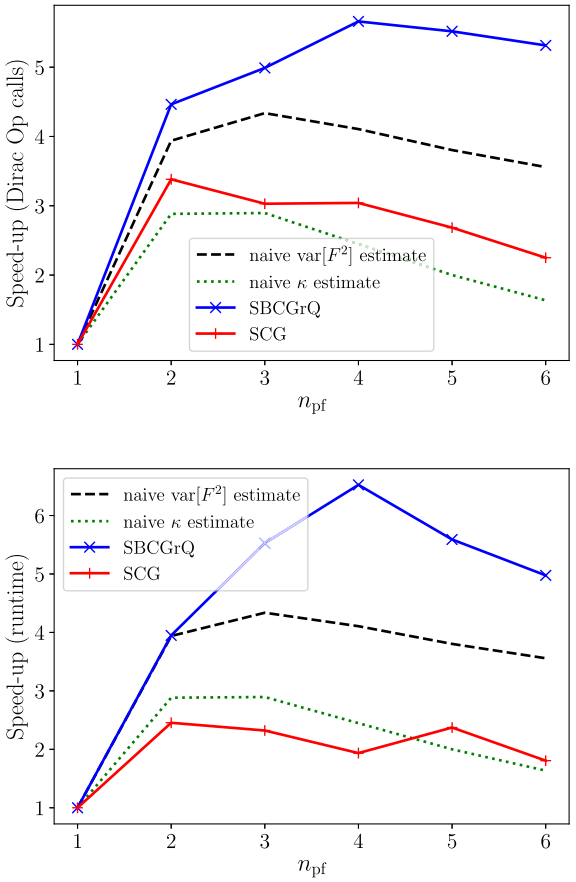

FIG. 12. Trajectory speed-up versus $n_{\rm pf}$ at $\simeq$90% acceptance, normalised to 1 for $n_{\rm pf} = 1$, using either multishift (SCG) or block multishift (SBCGrQ) solvers. *Top*: speed-up in Dirac operator calls; *bottom*: speed-up in computer runtime. The black dashed line is the simple prediction using the force norm variance of Eq. (14), and the green dotted line is the simple prediction from the condition number of the Dirac operator using Eq. (15). The optimal $n_{\rm pf}$ and the overall gain are both significantly increased by the use of block methods.

V. CONCLUSIONS

Let us summarize our study. We find that using multiple, $n_{\rm pf} > 1$, pseudofermions in RHMC simulations of lattice QCD offers three *cumulative* advantages:

(1) The magnitude of the fermionic force is reduced, which allows an increase of the integrator step size. Fewer steps are required per trajectory.

(2) The computation of the pseudofermionic force at each step now involves solving $n_{\rm pf}$ linear systems with different right-hand sides, all with the same Dirac matrix. Such systems are advantageously solved by *block* Krylov solvers, which converge with *fewer* Dirac matrix-vector operations, because the dimension of the search Krylov space increases by $n_{\rm pf}$ at each iteration.

(3) The computing time for a Dirac matrix-vector operation decreases, because the gauge field entering the Dirac matrix needs only to be loaded once for $n_{\rm pf}$ vectors to be multiplied, and cache locality is improved.

In addition, one may speculate that a smaller fermionic force, as obtained by multiple pseudofermions, indicates a smoother energy landscape, which might be explored faster by RHMC dynamics. We looked for a possible reduction of autocorrelation time under an increase of $n_{\rm pf}$, but found no clear indication of such (see Table I).

The solver that we use, described in Algorithm 1, is a multishift block version of the conjugate gradient, constructed in Ref. [35]. The problem of numerical instability seen in previous block solvers is handled by reorthogonalization of the search matrix, as recommended in Ref. [32] and recently used in Refs. [21,25].

Our simulations, albeit on a small lattice, show that 3 or 4 pseudofermions allow for a gain $\mathcal{O}(6)$ in CPU time. Let us discuss what to expect in a more realistic setup.

An improved, less local Dirac operator of staggered type would probably lead to further CPU gains because the assembly of the Dirac matrix elements from memory could be amortized even better. Similarly, a GPU-type architecture would benefit more, since its memory bandwidth is typically more limited compared to its FLOP performance. Reference [25] has shown significant gains from a block solver on a GPU machine. The multishift version thereof should yield similar benefits.

The reduction in solver iterations is strongly dependent on the ratio of the $n_{\rm pf}$th eigenvalue of the Dirac operator to the smallest one; the larger this ratio, the greater the reduction in

the number of iterations, as predicted from the convergence bound of Eq. (19) and also as seen empirically in Fig. 1. This observation can guide our expectations for how the gain from the block solver should depend on the mass, volume, and lattice spacing. In general, reducing the mass, going to coarser lattice spacing, or reducing the physical volume should all increase the gain of the block solver. Conversely, increasing the mass, going to finer lattice spacing, or increasing the physical volumes would presumably reduce the benefits of the block solver, so one scenario where this method may be particularly advantageous would be simulations done in the ϵ regime.

The benefit from using multiple pseudofermions in the molecular dynamics also grows as the mass is reduced, moreover the reduced variance of the force term would allow the use of higher order (but less stable) integrators whose costs grow more slowly with the volume [18].

A more quantitative statement about the scaling of the method with these parameters and how it compares to other recent algorithmic improvements such as multigrid [9,10] and deflation [8] would be highly desirable, but would require large scale simulations that are beyond the scope of this work.

Finally, we emphasize that our approach is algorithmically simple; more realistic tests involve rather small amounts of programming, and a single parameter to optimize: the number $n_{\rm pf}$ of pseudofermions.

ACKNOWLEDGMENTS

This work is supported by the Swiss National Science Foundation under Grant No. 200020-162515. Numerical simulations were performed on the Euler cluster at ETH Zürich. The authors thank the CERN Theoretical Physics Department for its hospitality.

[1] A. D. Kennedy, I. Horvath, and S. Sint, A New exact method for dynamical fermion computations with nonlocal actions, Nucl. Phys. Proc. Suppl. **73**, 834 (1999).

[2] M. A. Clark and A. D. Kennedy, The RHMC algorithm for two flavors of dynamical staggered fermions, Nucl. Phys. Proc. Suppl. **129**, 850 (2004).

[3] M. A. Clark, The rational hybrid Monte Carlo algorithm, PoS **LAT2006**, 004 (2006).

[4] T. A. DeGrand and P. Rossi, Conditioning techniques for dynamical fermions, Comput. Phys. Commun. **60**, 211 (1990).

[5] T. Lippert, Parallel SSOR preconditioning for lattice QCD, Parallel Comput. **25**, 1357 (1999).

[6] M. Luscher, Solution of the Dirac equation in lattice QCD using a domain decomposition method, Comput. Phys. Commun. **156**, 209 (2004).

[7] A. Frommer, A. Nobile, and P. Zingler, Deflation and Flexible SAP-Preconditioning of GMRES in Lattice QCD Simulation, arXiv:1204.5463.

[8] M. Luscher, Local coherence and deflation of the low quark modes in lattice QCD, J. High Energy Phys. 07 (2007) 081.

[9] A. Frommer, K. Kahl, S. Krieg, B. Leder, and M. Rottmann, Adaptive aggregation-based domain decomposition multigrid for the lattice Wilson-Dirac operator, SIAM J. Sci. Comput. **36**, A1581 (2014).

[10] R. C. Brower, M. A. Clark, A. Strelchenko, and E. Weinberg, Multigrid algorithm for staggered lattice fermions, Phys. Rev. D **97**, 114513 (2018).

[11] P. de Forcrand and T. Takaishi, Fast fermion Monte Carlo, Nucl. Phys. Proc. Suppl. **53**, 968 (1997).

[12] P. de Forcrand, UV filtered fermionic Monte Carlo, Nucl. Phys. Proc. Suppl. **73**, 822 (1999).

[13] T. Takaishi and P. de Forcrand, Testing and tuning new symplectic integrators for hybrid Monte Carlo algorithm in lattice QCD, Phys. Rev. E **73**, 036706 (2006).

[14] M. A. Clark, A. D. Kennedy, and P. J. Silva, Tuning HMC using Poisson brackets, PoS **LATTICE2008**, 041 (2008).

[15] A. D. Kennedy, P. J. Silva, and M. A. Clark, Shadow Hamiltonians, Poisson Brackets, and Gauge Theories, Phys. Rev. D **87**, 034511 (2013).

[16] M. Hasenbusch, Speeding up the hybrid Monte Carlo algorithm for dynamical fermions, Phys. Lett. B **519**, 177 (2001).

[17] M. Hasenbusch and K. Jansen, Speeding up lattice QCD simulations with clover improved Wilson fermions, Nucl. Phys. B **659**, 299 (2003).

[18] M. A. Clark and A. D. Kennedy, Accelerating Dynamical Fermion Computations Using the Rational Hybrid Monte Carlo (RHMC) Algorithm with Multiple Pseudofermion Fields, Phys. Rev. Lett. **98**, 051601 (2007).

[19] T. Sakurai, H. Tadano, and Y. Kuramashi, Application of block Krylov subspace algorithms to the Wilson-Dirac equation with multiple right-hand sides in lattice QCD, Comput. Phys. Commun. **181**, 113 (2010).

[20] H. Tadano, Y. Kuramashi, and T. Sakurai, Application of preconditioned block BiCGGR to the Wilson-Dirac equation with multiple right-hand sides in lattice QCD, Comput. Phys. Commun. **181**, 883 (2010).

[21] Y. Nakamura, K.-I. Ishikawa, Y. Kuramashi, T. Sakurai, and H. Tadano, Modified block BiCGSTAB for lattice QCD, Comput. Phys. Commun. **183**, 34 (2012).

[22] S. Birk and A. Frommer, A CG method for multiple right hand sides and multiple shifts in lattice QCD calculations, PoS **LATTICE2011**, 027 (2011).

[23] S. Birk and A. Frommer, A deflated conjugate gradient method for multiple right hand sides and multiple shifts, Numer. Algorithms **67**, 507 (2014).

[24] S. Birk, Deflated Shifted Block Krylov Subspace Methods for Hermitian Positive Definite Matrices, Ph.D. Thesis, Fachbereich Mathematik und Naturwissenschaften der Bergischen Universität Wuppertal, 2015, http://elpub.bib.uni-wuppertal.de/edocs/dokumente/fbc/mathematik/diss2015/birk/dc1505.pdf

[25] M. A. Clark, A. Strelchenko, A. Vaquero, M. Wagner, and E. Weinberg, Pushing memory bandwidth limitations through efficient implementations of Block-Krylov space solvers on GPUs, Comp. Phys. Comm. **233**, 29 (2018).

[26] D. P. O'Leary, The block conjugate gradient algorithm and related methods, Linear Algebra Appl. **29**, 293 (1980), Special Volume Dedicated to Alson S. Householder.

[27] I. Omelyan, I. Mryglod, and R. Folk, Symplectic analytically integrable decomposition algorithms: Classification, derivation, and application to molecular dynamics, quantum and celestial mechanics simulations, Comput. Phys. Commun. **151**, 272 (2003).

[28] A. Bussone, M. Della Morte, V. Drach, and C. Pica, Tuning the Hybrid Monte Carlo algorithm using molecular dynamics forces' variances, Comput. Phys. Commun., doi:10.1016/j.cpc.2018.07.012.

[29] J. C. Sexton and D. H. Weingarten, Hamiltonian evolution for the hybrid Monte Carlo algorithm, Nucl. Phys. B **380**, 665 (1992).

[30] M. Creutz, Global Monte Carlo algorithms for many-fermion systems, Phys. Rev. D **38**, 1228 (1988).

[31] S. Gupta, A. Irback, F. Karsch, and B. Petersson, The acceptance probability in the hybrid Monte Carlo method, Phys. Lett. B **242**, 437 (1990).

[32] A. A. Dubrulle, Retooling the method of block conjugate gradients, Electron. Trans. Numer. Anal. **12**, 216 (2001).

[33] A. Frommer, B. Nockel, S. Gusken, T. Lippert, and K. Schilling, Many masses on one stroke: Economic computation of quark propagators, Int. J. Mod. Phys. C **6**, 627 (1995).

[34] B. Jegerlehner, Krylov space solvers for shifted linear systems, arXiv:hep-lat/9612014.

[35] Y. Futamura, T. Sakurai, S. Furuya, and J.-I. Iwata, 'Efficient algorithm for linear systems arising in solutions of eigenproblems and its application to electronic-structure calculations, in *High Performance Computing for Computational Science–VECPAR 2012* (Springer, Berlin, 2013), pp. 226–235.

[36] M. Gutknecht and Z. Strakos, Accuracy of two three-term and three two-term recurrences for krylov space solvers, SIAM J. Matrix Anal. Appl. **22**, 213 (2000).

[37] A reference C++ implementation of the algorithm is available at https://github.com/lkeegan/blockCG

[38] P. de Forcrand and M. D'Elia, Continuum limit and universality of the Columbia plot, PoS **LATTICE2016**, 081 (2017).

[39] H. B. Meyer, H. Simma, R. Sommer, M. Della Morte, O. Witzel, and U. Wolff, Exploring the HMC trajectory-length dependence of autocorrelation times in lattice QCD, Comput. Phys. Commun. **176**, 91 (2007).