SIAM Review
Vol. 59, No. 3, pp. 547–587

Fast Randomized Iteration: Diffusion Monte Carlo through the Lens of Numerical Linear Algebra*

Lek-Heng Lim†
Jonathan Weare‡

Abstract. We review the basic outline of the highly successful diffusion Monte Carlo technique commonly used in contexts ranging from electronic structure calculations to rare event simulation and data assimilation, and propose a new class of randomized iterative algorithms based on similar principles to address a variety of common tasks in numerical linear algebra. From the point of view of numerical linear algebra, the main novelty of the fast randomized iteration schemes described in this article is that they have dramatically reduced operations and storage cost per iteration (as low as constant under appropriate conditions) and are rather versatile: we will show how they apply to the solution of linear systems, eigenvalue problems, and matrix exponentiation, in dimensions far beyond the present limits of numerical linear algebra. While traditional iterative methods in numerical linear algebra were created in part to deal with instances where a matrix (of size $\mathcal{O}(n^2)$) is too big to store, the algorithms that we propose are effective even in instances where the solution vector itself (of size $\mathcal{O}(n)$) may be too big to store or manipulate. In fact, our work is motivated by recent diffusion Monte Carlo based quantum Monte Carlo schemes that have been applied to matrices as large as $10^{108} \times 10^{108}$. We provide basic convergence results, discuss the dependence of these results on the dimension of the system, and demonstrate dramatic cost savings on a range of test problems.

Key words. dimension reduction, randomized algorithm, eigenvalue problem, matrix exponentiation, diffusion Monte Carlo, quantum Monte Carlo

AMS subject classifications. 65C05, 65F10, 65F15, 65F60, 68W20

DOI. 10.1137/15M1040827

Contents

*Received by the editors September 22, 2015; accepted for publication (in revised form) September 23, 2016; published electronically August 7, 2017.
http://www.siam.org/journals/sirev/59-3/M104082.html

Funding: LHL's work was generously supported by DARPA D15AP00109, AFOSR FA9550-13-1-0133, NSF IIS-1546413, DMS-1209136, and DMS-1057064. JQW's effort was supported by the Advance Scientific Computing Research program within the DOE Office of Science through award DE-SC0014205 as well as through a contract from Argonne, a U.S. Department of Energy Office of Science laboratory.

†Department of Statistics, University of Chicago, Chicago, IL 60637 (lekheng@galton.uchicago.edu).

‡James Franck Institute, Department of Statistics, University of Chicago, Chicago, IL 60637 (weare@uchicago.edu).

1. Introduction. Numerical linear algebra has been the cornerstone of scientific computing from its earliest days and randomized approaches to solving problems in linear algebra have a history almost as long as numerical linear algebra itself (see, e.g., [1, 17, 18, 25, 34, 35, 36, 38, 39, 62]).[1] As the size of matrices encountered in typical applications has increased (e.g., as we have sought greater and greater accuracy in numerical solution of partial differential equations (PDEs)), so has the attention paid to the performance of linear algebra routines on very large matrices in terms of both memory usage and operations count. Today, in applications ranging from numerical solution of PDEs to data analysis, we are frequently faced with the need to solve linear algebraic problems at and beyond the boundary of applicability of classical techniques. In response, randomized numerical linear algebra algorithms are receiving renewed attention and, over the last decade, have become an immensely popular subject of study within the applied mathematics and computer science communities (see, e.g., [14, 20, 21, 22, 24, 28, 44, 46, 54, 55, 61, 64]).

The goal of this article, after providing a brief introduction to the highly successful *diffusion Monte Carlo* (DMC) algorithm, is to suggest a new class of algorithms inspired by DMC for problems in numerical linear algebra. DMC is used in applications including electronic structure calculations, rare event simulation, and data assimilation to efficiently approximate expectations of the type appearing in Feynman–Kac formulae, i.e., for weighted expectations of Markov processes typically associated with parabolic PDEs (see, e.g., [15]). While based on principles underlying DMC, the fast randomized iteration (FRI) schemes that we study in this article are designed to address arguably the most classical and common tasks in matrix computations: linear systems, eigenvector problems, and matrix exponentiation, i.e., solving for v in

$$Av = b, \qquad Av = \lambda v, \qquad v = \exp(A)b \tag{1}$$

for matrices A that might not have any natural association with a Markov process.

[1] As pointed out in [30] many classical iterative techniques in numerical linear algebra are intimately related to Markov chain Monte Carlo (MCMC) schemes.

FRI schemes rely on basic principles similar to those at the core of other randomized methods that have appeared recently in the numerical linear algebra literature, but they differ substantially in their detail and in the problems they address. These differences will be remarked on again later, but roughly, while many recent randomized linear algebra techniques rely on a single sampling step, FRI methods randomize repeatedly and, as a consequence, are more sensitive to errors in the constructed randomizations. FRI schemes are not, however, the first to employ randomization within iterative schemes (see, in particular, [1] and [34]). In fact, the strategy of replacing expensive integrals or sums (without immediate stochastic interpretation) appearing in iterative protocols has a long history in a diverse array of fields. For example, it was used in schemes for the numerical solution of hyperbolic systems of PDEs in [11]. That strategy is represented today in applications ranging from density functional calculations in physics and chemistry (see, e.g., [3]) to maximum likelihood estimation in statistics and machine learning (see, e.g., [8]). Though related in that they rely on repeated randomization within an iterative procedure, these schemes differ from the methods we consider in that they do not use a stochastic representation of the solution vector itself. In contrast to these and other randomized methods that have been used in linear algebra applications, our focus is on problems for which the solution vector is extremely large so they can only be treated by linear or constant cost algorithms. In fact, the scheme that is our primary focus is ideally suited to problems so large that the solution vector itself is too large to store, so that no traditional iterative method (even for sparse matrices) is appropriate. This is possible because our scheme computes only low-dimensional projections of the solution vector and not the solution vector itself. The full solution is replaced by a sequence of sparse random vectors whose expectations are close to the true solution and whose variances are small.

DMC (see, e.g., [2, 7, 10, 26, 32, 33, 40, 42, 45]) is a central component in the quantum Monte Carlo (QMC) approach to computing the electronic ground state energy of the Schrödinger–Hamiltonian operator[2]

$$\mathcal{H}v = -\frac{1}{2}\Delta v + U\,v. \tag{2}$$

We are motivated, in particular, by the work in [7] in which the authors apply a version of the DMC procedure to a finite- (but huge-) dimensional projection of $\mathcal{H}$ onto a discrete basis respecting an antisymmetry property of the desired eigenfunction. The approach in [7] and subsequent papers have yielded remarkable results in situations where the projected Hamiltonian is an extremely large matrix (e.g., $10^{108} \times 10^{108}$; see [59]) and standard approaches to finite-dimensional eigenproblems are far from reasonable (see [4, 5, 6, 12, 13, 59]).

The basic DMC approach is also at the core of schemes developed for a number of applications beyond electronic structure. In fact, early incarnations of DMC were used in the simulation of small probability events [37, 56] for statistical physics models. Also in statistical physics, the transfer matrix Monte Carlo (TMMC) method was developed to compute the partition functions of certain lattice models by exploiting the observation that the partition function can all be represented as the dominant eigenvalue of the so-called transfer matrix, a real, positive, and sparse matrix (see

[2]The symbol Δ is used here to denote the usual Laplacian operator on functions of $\mathbb{R}^d$, $\Delta u = \sum_{i=1}^{d} \partial_{x_i}^2 u$. U is a potential function that acts on v by pointwise multiplication. Though this operator is symmetric, the FRI schemes we introduce are not restricted to symmetric eigenproblems.

[48]). TMMC may be regarded as an application of DMC to discrete eigenproblems. DMC has also become a popular method for many data assimilation problems and the notion of a "compression" operation introduced below is very closely related to the "resampling" strategies developed for those problems (see, e.g., [31, 41, 19]).

One can view the basic DMC, or for that matter Markov chain Monte Carlo (MCMC), procedure as a combination of two steps: In one step an integral operator (a Green's function) is applied to an approximate solution consisting of a weighted finite sum of delta functions; in the second step the resulting function (which is no longer a finite mixture of delta functions) is again approximated by a finite mixture of delta functions. The more delta functions allowed in the mixture, the higher the accuracy and cost of the method. A key to understanding the success of these methods is the observation that not *all* delta functions (i.e., at all positions in space) need appear in the mixture. A similar observation holds for the methods we introduce: FRI schemes need not access all entries in the matrix of interest to yield an accurate solution. In fact, we prove that the cost to achieve a fixed level of accuracy with our methods can be bounded independently of the size of the matrix, though in general applications one should expect some dependence on dimension. As with other randomized schemes, when an effective deterministic method is available it will very likely outperform the methods we propose; our focus is on problems for which satisfactory deterministic alternatives are not available (e.g., when the size of the intermediate iterates or final result are so large as to prohibit any conceivable deterministic method). Moreover, the schemes that we propose are a supplement and not a replacement for traditional dimensional reduction strategies (e.g., intelligent choice of basis). Indeed, successful application of DMC within practical QMC applications relies heavily on a change of variables based on approximations extracted by other methods (see the discussion of importance sampling in [26]).

The theoretical results that we provide are somewhat atypical of results commonly presented in the numerical linear algebra literature. In the context of linear algebra applications, both DMC and FRI schemes are most naturally viewed as randomizations of standard iterative procedures and their performance is largely determined by the structure of the particular deterministic iteration being randomized. For this reason, as well as to avoid obscuring the essential issues with the details of individual cases, we choose to frame our results in terms of the difference between the iterates v_t produced by a general iterative scheme and the iterates generated by the corresponding randomized scheme V_t (rather than considering the difference between V_t and $\lim_{t\to\infty} v_t$). In ideal situations (see Corollary 1) our bounds are of the form

$$|||V_t - v_t||| := \sup_{f\in\mathbb{C}^n,\ \|f\|_\infty\le 1} \sqrt{\mathbf{E}\left[|f\cdot V_t - f\cdot v_t|^2\right]} \le \frac{C}{\sqrt{m}}, \tag{3}$$

where the (in general t-dependent) constant C is independent of the dimension n of the problem and $m \le n$ controls the cost per iteration of the randomized scheme (one iteration of the randomized scheme is roughly a factor of n/m less costly than its deterministic counterpart and the two schemes are identical when $m = n$). The norm in (3) measures the root mean squared deviation in low-dimensional projections of the iterates. This choice is important, as is described in more detail in sections 3 and 4. For more general applications, one can expect the constant C, which incorporates the stability properties of the randomized iteration, to depend on dimension. In the worst case scenario, the randomized scheme is no more efficient than its deterministic counterpart (in other words, reasonable performance may require $m \sim n$). Our

numerical simulations, in which n/m ranges roughly between 10^7 and 10^{14}, strongly suggest that this scenario may be rare.

We will begin our development in section 2 with a description of the basic DMC procedure. Next, in section 3 we describe how ideas borrowed from DMC can be applied to general iterative procedures in numerical linear algebra. As a prototypical example, we describe how randomization can be used to dramatically decrease the cost of finding the dominant eigenvalue and (projections of) the dominant eigenvector. Also in that section, we consider the specific case in which the iteration mapping is an ε-perturbation of the identity, relevant to a wide range of applications involving evolutionary differential equations. In this case a poorly chosen randomization scheme can result in an unstable algorithm while a well-chosen randomization can result in an error that decreases with ε (over ε^{-1} iterations). Next, in sections 4 and 5, we establish several simple bounds regarding the stability and error of our schemes. Finally, in section 6 we provide three computational examples to demonstrate the performance of our approach. A simple, educational implementation of FRI applied to our first computational example is available online (see [63]). In the interest of reducing the length of this article, we list the proofs of all of our results separately in a supplemental document.

Remark 1. In several places we have included remarks that clarify or emphasize concepts that may otherwise be unclear to readers more familiar with classical, deterministic, numerical linear algebra methods. We anticipate that some of these remarks will be useful to this article's broader audience as well.

2. Diffusion Monte Carlo within Quantum Monte Carlo. The ground state energy, λ_*, of a quantum mechanical system governed by the Hamiltonian in (2) is the smallest eigenvalue (with corresponding eigenfunction v_*) of the Hermitian operator $\mathcal{H}$. The starting point for a DMC calculation is the imaginary-time Schrödinger equation[3,4]

$$\partial_t v = -\mathcal{H} v \tag{4}$$

(for a review of QMC, see [26]). One can, in principle, use power iteration to find λ_*: beginning from an initial guess v_0 (and assuming a gap between λ_* and the rest of the spectrum of $\mathcal{H}$), the iteration

$$\lambda_t = -\frac{1}{\varepsilon} \log \int e^{-\varepsilon \mathcal{H}} v_{t-1}(x)\, dx \quad \text{and} \quad v_t = \frac{e^{-\varepsilon \mathcal{H}} v_{t-1}}{\int e^{-\varepsilon \mathcal{H}} v_{t-1}(x)\, dx} \tag{5}$$

will converge to the pair (λ_*, v_*), where v_* is the eigenfunction corresponding to λ_*. Here the integral is over $x \in \mathbb{R}^d$.

Remark 2. For readers who are more familiar with the power method in numerical linear algebra, this may seem a bit odd, but a discrete analogue of (5) is just $v_t =$

[3]The reader familiar with quantum mechanics but unfamiliar with QMC may wonder why we begin with the imaginary-time Schrödinger equation and not the usual Schrödinger equation $i\partial_t v = -\mathcal{H} v$. The reason is that while the solutions to both equations can be expanded in terms of the eigenfunctions of $\mathcal{H}$, for the usual Schrödinger equation the contributions to the solution from eigenfunctions with larger eigenvalues do not decay relative to the ground state. By approximating the solution to the imaginary-time equation for large times we can approximate the ground state eigenfunction of $\mathcal{H}$.

[4]In practical QMC applications one solves for $\rho = v_* \tilde{v}$, where $\tilde{v}$ is an approximate solution found in advance by other methods. The new function ρ is the ground state eigenfunction of a Hamiltonian of the form $\tilde{\mathcal{H}} v = -\frac{1}{2}\Delta v + \operatorname{div}(bv) + \tilde{U} v$. The implications for the discussion in this section are minor.

$Av_{t-1}/\|Av_{t-1}\|_1$ applied to a positive definite matrix $A = \exp(-\varepsilon H) \in \mathbb{C}^{n\times n}$, where H is Hermitian.[5] The slight departure from the usual power method lies only in (i) normalizing by a 1-norm (or rather, by the sum of entries $\mathbb{1}^{\mathrm{T}} Av_{t-1}$ since both v and A are nonnegative) and (ii) iterating on $\exp(-\varepsilon H)$ instead of on H directly (the goal in this context is to find the smallest eigenvalue of H, not the magnitude-dominant eigenvalue of H). The iteration on the eigenvalue is then $\lambda_t = -\epsilon^{-1} \log\|Av_{t-1}\|_1$ since $\lambda_*(H) = -\epsilon^{-1} \log \lambda_*(A)$.

The first step in any (deterministic or stochastic) practical implementation of (5) is discretization of the operator $e^{-\varepsilon\mathcal{H}}$. DMC often uses the second order time discretization

$$e^{-\varepsilon\mathcal{H}} \approx K_\varepsilon = e^{-\frac{\varepsilon}{2}U} e^{\frac{\varepsilon}{2}\Delta} e^{-\frac{\varepsilon}{2}U}.$$

A standard deterministic approach would then proceed by discretizing the operator K_ε in space and replacing $e^{-\varepsilon\mathcal{H}}$ in (5) with the space and time discretized approximate operator. The number of spatial discretization points required by such a scheme to achieve a fixed accuracy will, in general, grow exponentially in the dimension d of x.

DMC uses two randomization steps to avoid this explosion in cost as d increases. These randomizations have the effect of ensuring that the random approximations V_t^m of the iterates v_t are always of the form

$$V_t^m(x) = \sum_{j=1}^{N_t} W_t^{(j)} \delta_{X_t^{(j)}}(x),$$

where $\delta_y(x)$ is the Dirac delta function centered at $y \in \mathbb{R}^d$, the $W_t^{(j)}$ are real, nonnegative numbers with $\mathbf{E}\big[\sum_{j=1}^{N_t} W_t^{(j)}\big] = 1$, and, for each $j \le N_t$, $X_t^{(j)} \in \mathbb{R}^d$. As will be made clear in a moment, the integer m superscripted in our notation controls the number of delta functions, N_t, included in the above expression for V_t^m.

The fact that the function V_t^m is nonzero at only N_t values is crucial to the efficiency of DMC. Starting with $N_0 = m$ and from an initial condition of the form

$$V_0^m = \frac{1}{m} \sum_{j=1}^{m} \delta_{X_0^{(j)}},$$

the first factor of $e^{-\frac{\varepsilon}{2}U}$ applied to V_0^m results in

$$\frac{1}{m} \sum_{j=1}^{m} e^{-\frac{\varepsilon}{2}U(X_0^{(j)})} \delta_{X_0^{(j)}},$$

which can be assembled in $\mathcal{O}(m)$ operations. The first of the randomization steps used in DMC relies on the well-known relationship

$$\int f(x)[e^{\frac{\varepsilon}{2}\Delta}\delta_y](x)\,dx = \mathbf{E}_y\left[f(B_\varepsilon)\right], \tag{6}$$

where f is a test function, B_s is a standard Brownian motion evaluated at time $s \ge 0$, and the subscript on the expectation is meant to indicate that $B_0 = y$ (i.e.,

[5] Note that the matrix H is not the same as the operator $\mathcal{H}$ on a function space, and it is only introduced for the purposes of relating the expression in (5) to the usual power method for matrices.

in the expectation in (6) B_ε is a Gaussian random variable with mean y and variance ε). In fact, this representation is a special case of the Feynman–Kac formula $\int f(x)[e^{-\varepsilon\mathcal{H}}\delta_y](x)\,dx = \mathbf{E}_y\big[f(B_\varepsilon)e^{-\int_0^\varepsilon U(B_s)\,ds}\big]$. Representation (6) suggests the approximation

$$K_\varepsilon V_0^m \approx \tilde{V}_1^m = \frac{1}{m}\sum_{j=1}^m e^{-\frac{\varepsilon}{2}\left(U(\xi_1^{(j)})+U(X_0^{(j)})\right)}\delta_{\xi_1^{(j)}},$$

where, conditioned on the $X_0^{(j)}$, the $\xi_1^{(j)}$ are independent and $\xi_1^{(j)}$ is normally distributed with mean $X_0^{(j)}$ and covariance εI (here I is the $d\times d$ identity matrix). This first randomization has allowed an approximation of $K_\varepsilon V_0^m$ by a distribution, $\tilde{V}_1^m$, which is again supported on only m points in $\mathbb{R}^d$. One might, therefore, attempt to define a sequence V_t^m iteratively by the recursion

$$V_{t+1}^m = \frac{\tilde{V}_{t+1}^m}{\int \tilde{V}_{t+1}^m(x)\,dx} = \sum_{j=1}^m W_{t+1}^{(j)}\,\delta_{\xi_{t+1}^{(j)}},$$

where we have recursively defined the weights

$$W_{t+1}^{(j)} = \frac{e^{-\frac{\varepsilon}{2}\left(U(\xi_{t+1}^{(j)})+U(X_t^{(j)})\right)}W_t^{(j)}}{\sum_{\ell=1}^m e^{-\frac{\varepsilon}{2}\left(U(\xi_{t+1}^{(\ell)})+U(X_t^{(\ell)})\right)}W_t^{(\ell)}}$$

with $W_0^{(j)} = 1/m$ for each j. The cost of this randomization procedure is $\mathcal{O}(dm)$ so that the total cost of a single iteration is $\mathcal{O}(dm)$.

At each step the weights in the expression for the iterates V_t^m are multiplied by additional random factors. These factors are determined by the potential U and the positions of the $\xi_t^{(j)}$. On the other hand, the $\xi_t^{(j)}$, evolve without reference to the potential function U (they are discretely sampled points from m independent Brownian motions). As a consequence, over many iterations one can expect extreme relative variations in the $W_t^{(j)}$ and, therefore, poor accuracy in V_t^m as an approximation of the functions v_t produced by (5).

The second randomization employed by the DMC algorithm is the key to controlling the growth in variance, and generalizations of the idea will be key to designing fast randomized iteration schemes in the next section. In order to control the variation in weights, at step t, DMC randomly removes points $\xi_t^{(j)}$ corresponding to small weights $W_t^{(j)}$ and duplicates points corresponding to large weights. The resulting number of points stored at each iteration, N_t, is close to, or exactly, m. At step t, a new distribution Y_t^m is generated from V_t^m so that

$$\mathbf{E}\,[Y_t^m \mid V_t^m] = V_t^m$$

by "resampling" a new collection of N_{t+1} points from the N_t points $\xi_t^{(j)}$ with associated probabilities $W_t^{(j)}$. The resulting points are labeled $X_t^{(j)}$ and the new distribution Y_t^m takes the form

$$Y_t^m = \frac{1}{N_t}\sum_{j=1}^{N_{t+1}}\delta_{X_t^{(j)}}.$$

The next iterate V_{t+1}^m is then built exactly as before but with V_t^m replaced by Y_t^m. All methods to select the $X_t^{(j)}$ generate, for each j, a nonnegative integer $N_t^{(j)}$ with

$$\mathbf{E}\big[N_t^{(j)} \mid \{W_t^{(\ell)}\}_{\ell=1}^m\big] = mW_t^{(j)}$$

and then sets $N_t^{(j)}$ of the elements in the collection $\{X_t^{(j)}\}_{j=1}^{N_{t+1}}$ equal to $\xi_t^{(j)}$ so that $N_{t+1} = \sum_{j=1}^{N_t} N_t^{(j)}$. For example, one popular strategy in DMC generates the $N_t^{(j)}$ independently with

$$\begin{aligned} \mathbf{P}\big[N_t^{(j)} = \lfloor mW_t^{(j)} \rfloor\big] &= \lceil mW_t^{(j)} \rceil - mW_t^{(j)}, \\ \mathbf{P}\big[N_t^{(j)} = \lceil mW_t^{(j)} \rceil\big] &= mW_t^{(j)} - \lfloor mW_t^{(j)} \rfloor. \end{aligned} \tag{7}$$

The above steps define a randomized iterative algorithm to generate approximations V_t^m of v_t. The second randomization procedure (generating Y_t^m from V_t^m) will typically require $\mathcal{O}(m)$ operations, preserving the overall $\mathcal{O}(dm)$ per iteration cost of DMC (as we have described it). The memory requirements of the scheme are also $\mathcal{O}(dm)$. The eigenvalue λ_* can be approximated, for example, by

$$-\frac{1}{\varepsilon} \log \left(\frac{1}{T} \sum_{t=1}^{T} \frac{1}{N_t} \sum_{j=1}^{N_t} e^{-\frac{\varepsilon}{2}\left(U(\xi_{t+1}^{(j)}) + U(X_t^{(j)})\right)} \right)$$

for T large.

Before moving on to more general problems, notice that the scheme outlined above applies just as easily to space-time discretizations of $e^{-\varepsilon \mathcal{H}}$. For example, if we set $h = \sqrt{(1+2d)\varepsilon}$ and denote by $\mathcal{E}_h^d \subset \mathbb{R}^d$ the uniform rectangular grid with resolution h in each direction, the operator $e^{-\varepsilon \mathcal{H}}$ can be discretized using

$$e^{-\varepsilon \mathcal{H}} \approx K_{\varepsilon,h} = e^{-\frac{\varepsilon}{2}U} e^{\frac{\varepsilon}{2}\Delta_h} e^{-\frac{\varepsilon}{2}U},$$

where, for any vector $g \in \mathcal{E}_h^d$,

$$\Delta_h g(x) = \frac{1}{\varepsilon} \left(-g(x) + \frac{1}{1+2d} \sum_{y \in \mathcal{E}_h^d,\ \|y-x\|_2 \le h} g(y) \right)$$

(here we find it convenient to identify functions $g : \mathcal{E}_h^d \to \mathbb{R}$ and vectors in $\mathbb{R}^{\mathcal{E}_h^d}$). The operator $e^{-\varepsilon \Delta_h}$ again has a stochastic representation; now the representation is in terms of a jump Markov process with jumps from a point in $\mathcal{E}_h^d$ to one of its nearest neighbors on the grid (for an interesting approach to discretizing stochastic differential equations taking advantage of a similar observation, see [9]).

Remark 3. The reader should notice that not only will we be unable to store the matrix $K_{\varepsilon,h}$ (which is exponentially large in d) or afford to compute $K_{\varepsilon,h}v$ for a general vector v, but we will not even be able to store the iterates v_t generated by the power method. Even the sparse matrix routines developed in numerical linear algebra to deal with large matrices are not reasonable for this problem.

In this discrete context, a direct application of the DMC approach (as in [48]) would represent the solution vector as a superposition of standard basis elements[6] and replace calculation of $K_{\varepsilon,h}v$ by a random approximation whose cost is (for this particular problem) free from any direct dependence on the size of $K_{\varepsilon,h}$ (though its cost can depend on structural properties of $K_{\varepsilon,h}$ which may be related to its size), whose

[6]The delta functions represent the indices of v_t that we are keeping track of; δ_ξ is more commonly denoted $\mathbf{e}_\xi$ in numerical linear algebra—the standard basis vector with 1 in the ξth coordinate and zero elsewhere.

expectation is exactly $K_{\varepsilon,h}v$, and whose variance is small. The approach in [7] is also an application of these same basic DMC steps to a discrete problem, though in that case the desired eigenvector has entries of a priori unknown sign, requiring that the solution vector be represented by a superposition of signed standard basis elements.

In this article we take a different approach to adapting DMC to discrete problems. Instead of exactly reproducing in the discrete setting the steps comprising DMC, consider a slightly modified scheme that omits direct randomization of an approximation to $e^{\varepsilon\Delta_h}$, and instead relies solely on a general random mapping Φ_t^m very similar to the map from V_t^m to Y_t^m, but which takes a vector $V_t^m \in \mathbb{R}^{\mathcal{E}_h^d}$ with nonnegative entries and $\|V_t^m\|_1 = 1$ (i.e., a probability measure on $\{1, 2, \ldots, |\mathcal{E}_h^d|\}$) and produces a new random vector Y_t^m with m, or nearly m, nonzero components that satisfies $\mathbf{E}\left[Y_t^m \mid V_t^m\right] = V_t^m$ as above. Starting from a nonnegative initial vector $V_0^m \in \mathcal{E}_h^d$ with $\|V_0^m\|_1 = 1$ and with at most $\mathcal{O}(md)$ nonzero entries, V_{t+1}^m is generated from V_t^m as follows:

Step 1. Generate $Y_t^m = \Phi_t^m\left(V_t^m\right)$ with approximately or exactly m nonzero entries.

Step 2. Set $V_{t+1}^m = \frac{K_{\varepsilon,h}Y_t^m}{\|K_{\varepsilon,h}Y_t^m\|_1}$.

Just as at iteration t, DMC produces a random approximation of the result of t iterations of power iteration for the infinite-dimensional integral operator $e^{-\varepsilon\mathcal{H}}$, the above steps produce a random approximation of the result of t iterations of the power iteration for the matrix $K_{\varepsilon,h}$. The improved efficiency of DMC is due to the application of the integral operator to a finite sum of delta functions in place of a more general function. Similarly, the efficiency of the finite-dimensional method in the above two step procedure is a consequence of the replacement of a general vector v in the product $K_{\varepsilon,h}v$ by a sparse approximation, $\Phi_t^m(v)$.

For the random mapping $\Phi_t^m(v)$ we might, for example, adapt the popular resampling choice mentioned above and choose the entries of Y_t^m independently with

$$\begin{aligned}\mathbf{P}\left[(Y_t^m)_j = \left\lfloor (V_t^m)_j\, m \right\rfloor / m\right] &= \left\lceil (V_t^m)_j\, m \right\rceil - (V_t^m)_j\, m,\\ \mathbf{P}\left[(Y_t^m)_j = \left\lceil (V_t^m)_j\, m \right\rceil / m\right] &= (V_t^m)_j\, m - \left\lfloor (V_t^m)_j\, m \right\rfloor.\end{aligned} \tag{8}$$

Note that this rule results in a vector Y_t^m with expectation exactly equal to V_t^m. On the other hand, when the number of nonzero entries in V_t^m is large, many of those entries must be less than $1/m$ (because $\|V_t^m\|_1 = 1$) and will have some probability of being set equal to zero in Y_t^m. In fact, the number of nonzero entries in Y_t^m has expectation and variance bounded by m. The details of the mappings Φ_t^m, which we call compression mappings, will be described later in section 5 where, for example, we will find that the cost of applying the mapping Φ_t^m will typically be $\mathcal{O}(n)$ when its argument has n nonzero entries (in this setting, $n = \mathcal{O}(md)$). And while the cost of applying $K_{\varepsilon,h}$ to an arbitrary vector in $\mathbb{R}^{\mathcal{E}_h^d}$ is $|\mathcal{E}_h^d|$, the cost of applying $K_{\varepsilon,h}$ to a vector with m nonzero entries is only $\mathcal{O}(md)$. The total cost of the scheme per iteration is therefore $\mathcal{O}(md)$ in storage and operations. These cost requirements are dependent on the particular spatial discretization of $e^{\varepsilon\Delta}$; if we had chosen a discretization corresponding to a dense matrix $K_{\varepsilon,h}$, then the cost reduction would be much less extreme. Nonetheless, as we will see in section 3, the scheme just described can be easily generalized and, as we will see in sections 4 and 6, will often result in methods that are significantly faster than their deterministic counterparts.

Remark 4. Rather than focusing on sampling indices of entries in a vector v, as is typical of some of the literature on randomized numerical linear algebra, we focus

on constructing an accurate sparse representation of v. This is primarily a difference in perspective, but has consequences for the accuracy of our randomizations. For example, the techniques in [17, 18] and in [20, 21, 22] would correspond, in our notation and context, to setting for $v \in \mathbb{R}^n$

$$\Phi_t^m(v) = \frac{\|v\|_1}{m} \sum_{j=1}^{n} \frac{v_j}{|v_j|} N_t^{(j)} \mathbf{e}_j, \tag{9}$$

where $\mathbf{e}_j$ is the jth standard basis element and the random vector

$$\left(N_t^{(1)}, N_t^{(2)}, \ldots, N_t^{(n)}\right) \sim \textsc{Multinomial}(m, p_1, \ldots, p_n)$$

with $p_j = |v_j|/\|v\|_1$. As for all Monte Carlo methods, the sparse characteristic of this representation is responsible for gains in efficiency. Also, when error is measured by the norm in (3), only a random sparse representation can be accurate for general v. However, random index selection yields only one of many possible random sparse representations of v and not one that is particularly accurate. In fact, *effective FRI schemes of the type introduced in this article cannot be based solely on random index selection as in* (9). In the setting of this section, if we were to use (9) in place of the rule in (8) the result would be an unstable scheme (the error would become uncontrollable as ε is decreased). As we will see in section 5, much more accurate sparse representations are possible.

Even restricting ourselves to the QMC context, there is ample motivation to generalize the DMC scheme. Often one wishes to approximate not the smallest eigenvalue of $\mathcal{H}$ but instead the smallest eigenvalue corresponding to an antisymmetric (in exchange of particle positions) eigenfunction. DMC as described in this section cannot be applied directly to computing this value, a difficulty commonly referred to as the Fermion sign problem. Several authors have attempted to address this issue with various modifications of DMC. In particular, Alavi and coworkers recently developed a version of DMC for a particular spatial discretization of the Hamiltonian (in the configuration interaction basis) that exactly preserves antisymmetry (unlike the finite difference discretization we just described). Run to convergence, their method provides the same approximation as the so-called full configuration interaction method but can be applied with a much larger basis (e.g., in experiments by Alavi and coworkers reported in [59] up to 10^{108} total functions in the expansion of the solution). Though the generalizations of DMC represented by the two step procedure in the last paragraph and by the developments in the next section are motivated by the scheme proposed in [7], they differ substantially in their details and can be applied to a wider range of problems (including different discretizations of $\mathcal{H}$). Finally, we remark that while we have considered DMC in the particular context of computing the ground state energy of a Hamiltonian, the method can be used for a much wider variety of tasks with only minor modification to its basic structure. For example, particle filters (see, e.g., [19]) are an application of DMC to on-line data assimilation and substantive differences lie mostly in the interpretation of the operator to which DMC is applied (and the fact that one is typically interested in the solution after finitely many iterations).

3. A General Framework. Consider the general iterative procedure

$$v_{t+1} = \mathcal{M}(v_t) \tag{10}$$

for $v_t \in \mathbb{C}^n$. Eigenproblems, linear systems, and matrix exponentiation can all be accomplished by versions of this iteration. In each of those settings the cost of eval-

uating $\mathcal{M}(v)$ is dominated by a matrix-vector multiplication. We assume that the cost (in terms of floating point operations and storage) of performing the required matrix-vector multiplication makes it impossible to carry out recursion (10) to the desired precision. As in the steps described at the end of the last section, we will consider the error resulting from replacement of (10) by

$$V_{t+1}^m = \mathcal{M}\left(\Phi_t^m(V_t^m)\right), \tag{11}$$

where the compression maps $\Phi_t^m : \mathbb{C}^n \to \mathbb{C}^n$ are independent, inexpensive to evaluate, and enforce sparsity in the V_t^m iterates (the number of nonzero entries in V_t^m will be $\mathcal{O}(m)$) so that $\mathcal{M}$ can be evaluated at much less expense. When $\mathcal{M}$ is a perturbation of identity *and* an $\mathcal{O}(n)$ scheme is appropriate (see sections 3.2 and 4 below) we will also consider the scheme,

$$V_{t+1}^m = V_t^m + \mathcal{M}\left(\Phi_t^m(V_t^m)\right) - \Phi_t^m(V_t^m). \tag{12}$$

The compressions Φ_t^m will satisfy (or very nearly satisfy) the statistical consistency criterion

$$\mathbf{E}\left[\Phi_t^m(v)\right] = v$$

and will have to be carefully constructed to avoid instabilities and yield effective methods. For definiteness one can imagine that Φ_t^m is defined by a natural extension of (8),

$$\begin{aligned}
\mathbf{P}\left[\left(\Phi_t^m(v)\right)_j = \frac{v_j}{m|v_j|}\left\lfloor \frac{m|v_j|}{\|v\|_1}\right\rfloor\right] &= \left\lceil \frac{m|v_j|}{\|v\|_1}\right\rceil - \frac{m|v_j|}{\|v\|_1}, \\
\mathbf{P}\left[\left(\Phi_t^m(v)\right)_j = \frac{v_j}{m|v_j|}\left\lceil \frac{m|v_j|}{\|v\|_1}\right\rceil\right] &= \frac{m|v_j|}{\|v\|_1} - \left\lfloor \frac{m|v_j|}{\|v\|_1}\right\rfloor,
\end{aligned} \tag{13}$$

to accept arguments $v \in \mathbb{C}^n$ (V_t^m is no longer a nonnegative real number). This choice has several drawbacks, not least of which is its cost, and *we do not use it in our numerical experiments.* Alternative compression schemes, including the one used in our numerical simulations, are considered in detail in section 5. There we will learn that one can expect that, for any pair $f, v \in \mathbb{C}^n$,

$$\sqrt{\mathbf{E}\left[|f^{\mathrm{H}}\Phi_t^m(v) - f^{\mathrm{H}}v|^2\right]} \leq \frac{2}{\sqrt{m}}\|f\|_\infty\|v\|_1 \tag{14}$$

(the superscript $^{\mathrm{H}}$ is used throughout this article to denote the conjugate transpose of a vector with complex entries). These errors are introduced at each iteration and need to be removed to obtain an accurate estimate. Depending on the setting, we may rely on averaging over long trajectories, averaging over parallel simulations (replicas), or dynamical self-averaging (see sections 3.2 and 4), to remove the noise introduced by our randomization procedure. Because the specific choice of $\mathcal{M}$ and the form of averaging used to remove noise can differ substantially by setting, we will describe the schemes within the context of specific (and common) iterative procedures.

3.1. The Eigenproblem Revisited. Consider, for example, a more general eigenproblem than the one we considered in section 2. Given $K \in \mathbb{C}^{n\times n}$ the goal is to determine $\lambda_* \in \mathbb{C}$ and $v_* \in \mathbb{C}^n$ such that

$$Kv_* = \lambda_* v_* \tag{15}$$

and such that, for any other solution pair (λ, v), $|\lambda| < |\lambda_*|$. In what follows in this section we will assume that this problem has a unique solution. The standard methods of approximate solution of (15) are variants of the power method, a simple version of which performs

$$v_{t+1} = \frac{Kv_t}{\|Kv_t\|_1}, \qquad \lambda_{t+1} = \frac{u^{\mathrm{H}} K v_t}{u^{\mathrm{H}} v_t}, \tag{16}$$

where $u \in \mathbb{C}^n$ is chosen by the user. Under generic conditions, these converge to the correct (λ_*, v_*) starting from an appropriate initial vector v_0 (see, e.g., [16]). The scheme in (16) requires $\mathcal{O}(n^2)$ work per iteration and at least $\mathcal{O}(n)$ storage. In this article, we are interested in situations in which these cost and storage requirements are unreasonable.

From $\mathcal{O}(n^2)$ to $\mathcal{O}(nm)$. For the iteration in (16) the randomized scheme (11) (along with an approximation of λ_*) becomes

$$\begin{aligned} V_{t+1}^m &= \frac{K\Phi_t^m(V_t^m)}{\|K\Phi_t^m(V_t^m)\|_1}, \quad \Lambda_{t+1}^m = \frac{u^{\mathrm{H}} K\Phi_t^m(V_t^m)}{u^{\mathrm{H}} V_t^m}, \\ \overline{V}_t^m &= \frac{1}{t}\sum_{s=1}^{t} V_s^m, \qquad\qquad \overline{\Lambda}_t^m, = \frac{1}{t}\sum_{s=1}^{t} \Lambda_t^m, \end{aligned} \tag{17}$$

where $\overline{V}_t^m$ and $\overline{\Lambda}_t^m$ are trajectory averages estimating v_* and λ_*. In (17), the compressions Φ_t^m are independent of one another. Using the rules defining Φ_t^m in section 5, construction of $\Phi_t^m(V_t^m)$ at each step will require $\mathcal{O}(n)$ operations. Since multiplication of the vector $\Phi_t^m(V_t^m)$ by a dense matrix K requires $\mathcal{O}(nm)$ operations, this scheme has $\mathcal{O}(nm)$ cost and $\mathcal{O}(n)$ storage per iteration requirement.

Iteration (12), on the other hand, replaces (16) with

$$V_{t+1}^m = V_t^m + \left(\frac{K\Phi_t^m(V_t^m)}{\|K\Phi_t^m(V_t^m)\|_1} - \Phi_t^m(V_t^m)\right), \quad \Lambda_{t+1}^m = \frac{u^{\mathrm{H}} K\Phi_t^m(V_t^m)}{u^{\mathrm{H}} V_t^m}. \tag{18}$$

By the same arguments as above, this iteration will also have cost and storage requirements of $\mathcal{O}(nm)$ and $\mathcal{O}(n)$, respectively. When $K = I + \varepsilon A$ for some matrix A and small parameter $\varepsilon > 0$, the iteration in (18) bears strong resemblance to the Robbins–Monro stochastic approximation algorithm [53, 43]. In fact, as we will see in section 4, when the mapping $\mathcal{M}$ is of the form $v + \varepsilon b(v)$ the convergence of methods of the form in (11) and (12) is reliant on the self-averaging phenomenon also at the heart of stochastic approximation. We will also learn that for $\mathcal{M}$ of this form one can expect the error corresponding to (12) to be smaller than the error corresponding to (11).

From $\mathcal{O}(nm)$ to $\mathcal{O}(m)$. For many problems even $\mathcal{O}(nm)$ cost and storage requirements are unacceptable. This is the case, for example, when n is so large that a vector of length n cannot be stored. But now suppose that K is sparse with at most q nonzero entries per column. Because $\Phi_{t-1}^m(V_{t-1}^m)$ has $\mathcal{O}(m)$ nonzero entries, the product $K\Phi_{t-1}^m(V_{t-1}^m)$ (and hence also V_t^m) has at most $\mathcal{O}(qm)$ entries and requires $\mathcal{O}(qm)$ operations to assemble. On the other hand, if V_t^m has at most $\mathcal{O}(qm)$ nonzero entries, then application of Φ_t^m to V_t^m requires only $\mathcal{O}(qm)$ operations. Consequently, as long as V_0^m has at most $\mathcal{O}(qm)$ nonzero entries, the total number of floating point operations required by (17) reduces to $\mathcal{O}(qm)$ per iteration. This observation does

not hold for methods of the form (12) which will typically result in dense iterates V_t^m and a cost of $\mathcal{O}(n)$ even when K is sparse.

As we have mentioned (and as was true in section 2), in many settings even storing the full solution vector is impossible. Overcoming this impediment requires a departure from the usual perspective of numerical linear algebra. Instead of trying to approximate all entries of v_*, our goal becomes to compute

$$f_* = f^{\mathrm{H}} v_*$$

for some vector (or small number of vectors) f. This change in perspective is reflected in the form of our compression rule error estimate in (14) and in the form of our convergence results in section 4 that measure error in terms of dot products with test vectors as in (3) above. As discussed in more detail in section 4, the choice of error norm in (3) is essential to our eventual error estimates. Indeed, were we to estimate a more standard quantity such as

$$\mathbf{E}\left[\|V_t^m - v_t\|_1\right],$$

we would find that the error decreases proportional to $(n-m)/n$, requiring that m increase with n to achieve fixed accuracy. The algorithmic consequence of our focus on computing low-dimensional projections of v_* is simply the removal in (17) of the equation defining $\overline{V}_t^m$ and insertion of

$$F_t^m = f^{\mathrm{H}} V_t^m \quad \text{and} \quad \overline{F}_t^m = \frac{1}{t}\sum_{s=1}^{t} F_s^m, \tag{19}$$

which produces an estimate F_t^m of f_*.

Remark 5. While estimation of f_* may seem an unusual goal in the context of classical iterative schemes, it is completely in line with the goals of any MCMC scheme which typically seek only to compute averages with respect to the invariant measure of a Markov chain and not to completely characterize that measure.

Schemes with $\mathcal{O}(m)$ storage and operations requirements per iteration can easily be designed for any general matrix. Accomplishing this for a dense matrix requires an additional randomization in which columns of K (or of some factor of K) are randomly set to zero independently at each iteration; e.g., again in the context of power iteration, assuming that V_{t-1}^m has at most $\mathcal{O}(m)$ nonzero entries, one can use

$$V_{t+1}^m = \frac{Y_{t+1}^m}{\|Y_{t+1}^m\|_1} \quad \text{with} \quad Y_{t+1}^m = \sum_{j=1}^{n} \left(\Phi_t^m(V_t^m)\right)_j \Phi_t^{m_t^j, j}(K_j) \tag{20}$$

in place of (17), where K_j is used to denote the jth column of K and each $\Phi_t^{m_t^j,j}$ is an independent copy of $\Phi_t^{m_t^j}$, which are assumed independent of Φ_t^m. The number of entries retained in each column is controlled by m_t^j which can, for example, be set to

$$m_t^j = \left\lceil \frac{\|K_j\|_1 |(V_t^m)_j|}{\sum_{\ell=1}^n \|K_\ell\|_1 |(V_t^m)_\ell|}\, m \right\rceil \quad \text{or} \quad m_t^j = \left\lceil |(V_t^m)_j|\, m \right\rceil$$

at each iteration and the resulting vector can then be compressed so that it has exactly or approximately m nonzero entries. Use of (20) in place of (17) will result in a scheme

whose cost per iteration is independent of n if the compressions of the columns have cost independent of n. This may be possible without introducing significant error, for example, when the entries in the columns of K can take only a small number of distinct values. Notice that one obtains the update in (17) from (20) by removing the compression of the columns. Consequently, given V_t^m, the conditional variance of V_{t+1}^m generated by (20) will typically exceed the conditional variance resulting from (17).

3.2. Perturbations of Identity. We now consider the case that $\mathcal{M}$ is a perturbation of the identity, i.e., that

$$\mathcal{M}(v) - v = \varepsilon\, b(v) + o(\varepsilon), \tag{21}$$

where ε is a small positive parameter. This case is of particular importance because, when the goal is to solve an ordinary differential equation (ODE) initial value problem

$$\frac{d}{dt}y = b(y), \qquad y(0) = y_0, \tag{22}$$

discrete-in-time approximations take the form (10) with $\mathcal{M}$ of the form in (21). As is the case in several of our numerical examples, the solution to (22) may represent, for example, a semidiscretization (a discretization in space) of a PDE.

Several common tasks in numerical linear algebra, not necessarily directly related to ODEs or PDEs, can also be addressed by considering (22). For example, suppose that we solve the ODE (22) with $b(y) = Ay - r$ for some $r \in \mathbb{C}^n$ and any $n \times n$ complex valued matrix A. The solution to (22) in this case is

$$y(t) = e^{tA}y_0 + A^{-1}\left(I - e^{tA}\right) r.$$

Setting $r = 0$ in the last expression we find that any method to approximate ODE (22) for $t = 1$ can be used to approximate the product of a given vector and the exponential of the matrix A. On the other hand, if $r \neq 0$ and all eigenvalues of A have negative real part, then, for very large t, the solution to (22) converges to $A^{-1}r$. In fact, in this case we obtain the continuous-time variant of Jacobi iteration for the equation $Ax = r$. Like Jacobi iteration, it can be extended to somewhat more general matrices. Discretizing (22) in time with $b(y) = Ay - r$ and a small time step allows for the treatment of matrices with a wider range of eigenvalues than would be possible with standard Jacobi iteration.

Some important eigenproblems are also solved using an $\mathcal{M}$ satisfying (21). For example, this is the case when the goal is to compute the eigenvalue/vector pair corresponding to the eigenvalue of largest real part (rather than of largest magnitude) of a differential operator, e.g., the Schrödinger operator discussed in section 2. While the power method applied directly to a matrix A converges to the eigenvector of A corresponding to the eigenvalue of largest magnitude, the angle between the vector $\exp(tA)y_0$ and the eigenvector corresponding to the eigenvalue of A with largest real part converges to zero (assuming y_0 is not orthogonal to that eigenvector). If we discretize (22) in time with $b(v) = Av$ and renormalize the solution at each time step (to have unit norm), then the iteration will converge to the desired eigenvector (or an ε-dependent approximation of that eigenvector).

As we will learn in the next section, designing effective FRI schemes for these problems requires that the error in the stochastic representation $\mathcal{M} \circ \Phi_t^m$ of $\mathcal{M}$ decrease sufficiently rapidly with ε. In particular, in order for our schemes to accurately

approximate solutions to (22) over intervals of $\mathcal{O}(1)$ units of time (i.e., over $\mathcal{O}(1/\varepsilon)$ iterations of the discrete scheme), we will need, and will verify in certain cases, that

$$\sqrt{\mathbf{E}\left[|f^{\mathrm{H}}\mathcal{M}\left(\Phi_t^m(v)\right) - f^{\mathrm{H}}\mathcal{M}(v)|^2\right]} \sim \sqrt{\varepsilon}.$$

Obtaining a bound of this type will require that we use a carefully constructed random compression Φ_t^m such as those described in section 5. In fact, when a scheme with $\mathcal{O}(n)$ cost per iteration is acceptable, iteration (11) can be replaced by (12), i.e., by

$$V_{t+1}^m = V_t^m + \varepsilon\, b\left(\Phi_t^m(V_t^m)\right) + o(\varepsilon), \tag{23}$$

in which case we can expect errors over $\mathcal{O}\left(\varepsilon^{-1}\right)$ iterations that vanish with ε (rather than merely remaining stable). As we will see in more detail in the next section, the principle of dynamic self-averaging is essential to the convergence of either (11) or (12) when $\mathcal{M}$ is a perturbation of identity. The same principle is invoked in the contexts of, for example, multiscale simulation (see, e.g., [52] and [23] and the many references therein) and stochastic approximation (see, e.g., [43] and the many references therein).

4. Convergence. Many randomized linear algebra schemes referenced in the opening paragraph of this article rely at their core on an approximation of a product such as AB, where, for example, A and B are $n \times n$ matrices, by a product of the form $A\Theta B$, where Θ is an $n \times n$ random matrix with $\mathbf{E}[\Theta] = I$, and such that $A\Theta B$ can be assembled at much less expense than AB. For example, one might choose Θ to be a diagonal matrix with only $m \ll n$ nonzero entries on the diagonal so that ΘB has only m nonzero rows and $A\Theta B$ can be assembled in $\mathcal{O}(n^2 m)$ operations instead of $\mathcal{O}(n^3)$ operations. Alternatively, one might choose $\Theta = \xi\xi^{\mathrm{T}}$, where ξ is an $n \times m$ random matrix with independent entries, each having mean 0 and variance $1/m$. With this choice one can again construct $A\Theta B$ in $\mathcal{O}(n^2 m)$ operations. Typically, this randomization is carried out once in the course of the algorithm. The error made in such an approximation can be expected to be of size $\mathcal{O}(1/\sqrt{m})$, where the prefactor depends (very roughly) on a norm of the matrices (and other structural properties) but does not depend directly on n (see, e.g., [38, equation 30] or [20, Theorem 1]).

In the schemes that we consider, we apply a similar randomization to speed matrix vector multiplication at each iteration of the algorithm (though our compression rules vary in distribution from iteration to iteration). As is explored below, the consequence is that any stability property of the original, deterministic iteration responsible for its convergence will be weakened by the randomization and that effect might introduce an additional n dependence into the cost of the algorithm to achieve a fixed accuracy. The compression rule must therefore be carefully constructed to minimize error. Compression rules are discussed in detail in section 5. In this section, we consider the error resulting from (11) and (12) for an unspecified compression rule satisfying the generic error properties established (with caveats) in section 5. Because it provides a dramatic illustration of the need to construct accurate compression rules, and because of its importance in practical applications, we pay particular attention to the case in which $\mathcal{M}$ is an ε-perturbation of the identity. Our results rely on classical techniques in the numerical analysis of deterministic and stochastic dynamical systems and, in particular, are typical of basic results concerning the convergence of stochastic approximation (see, e.g., [43] for a general introduction and [47] for results in the context of machine learning) and sequential Monte Carlo methods (see, e.g., [15] for a general introduction and [57] for results in the context of QMC). They concern the mean squared size of the difference between the output, V_t^m, of the randomized scheme and

the output, v_t, of its deterministic counterpart and are chosen to efficiently highlight important issues such as the role of stability properties of the deterministic iteration (10), the dependence of the error on the size of the solution vector, n, and the role of dynamic self-averaging. More sophisticated results (such as central limit theorems and exponential bounds on deviation probabilities) are possible following developments in, for example, [43] and [15, 57]. In the interest of reducing the length of this article we list the proofs of all of our results separately in a supplemental document.

Our notion of error will be important. It will not be possible to prove, for example, that $\mathbf{E}\left[\|V_t^m - v_t\|_1\right]$ remains small without a strong dependence on n. It is not even the case that $\mathbf{E}\left[\|\Phi_t^m(v) - v\|_1\right]$ is small when n is large and $\|v\|_1 = 1$. Take, for example, the case that $v_i = 1/n$. In this case any scheme that sets $n - m$ entries to zero will result in an error $\|\Phi_t^m(v) - v\|_1 \geq (n-m)/n$. On the other hand, we need to choose a measure of error sufficiently stringent that our eventual error bounds imply that our methods accurately approximate observables of the form $f^{\mathrm{H}} v_t$. For example, analogues of all of the results below using the error metric $(\mathbf{E}[\|V_t^m - v_t\|_2^2])^{1/2}$ could be established. However, error bounds of this form are not, in themselves, sufficient to imply dimension-independent bounds on the error in $f^{\mathrm{H}} v_t$ because they ignore correlations between the components of V_t^m. Indeed, in general one can only expect that $(\mathbf{E}[|f^{\mathrm{H}} V_t^m - f^{\mathrm{H}} v_t|])^{1/2} \leq \sqrt{n}(\mathbf{E}\|V_t^m - v_t\|_2^2])^{1/2}$ when $\|f\|_\infty \leq 1$.

Remark 6. It is perhaps more typical in numerical linear algebra to state error bounds in terms of the quantity one ultimately hopes to approximate and not in terms of the distance to another approximation of that quantity. For example, one might wonder why our results are not stated in terms of the total work required to achieve (say, with high probability) an error of a specified size in an approximation of the dominate eigenvalue of a matrix. Our choice to consider the difference between V_t^m and v_t is motivated by the fact that the essential characteristics contributing to errors due to randomization are most naturally described in terms of the map defining the deterministic iteration. More traditional characterizations of the accuracy of the schemes can be inferred from the bounds provided below and error bounds for the corresponding deterministic iterative schemes.

Motivated by our stated goal, as described in section 3, of estimating quantities of the form $f^{\mathrm{H}} v_t$, we measure the size of the (random) errors produced by our scheme using the norm

$$|||X||| = \sup_{\|f\|_\infty \leq 1} \sqrt{\mathbf{E}\left[|f^{\mathrm{H}} X|^2\right]}, \tag{24}$$

where X is a random variable with values in $\mathbb{C}^n$ (all random variables referenced are assumed to be functions on a single probability space that will be left unspecified). This norm is the $(\infty, 2)$-norm [27, section 7] of the square root of the second moment matrix of X, i.e.,

$$|||X||| = \|B\|_{\infty,2} = \sup_{\|f\|_\infty \leq 1} \|Bf\|_2,$$

where

$$B^{\mathrm{H}} B = \mathbf{E}\left[X X^{\mathrm{H}}\right].$$

It is not difficult to see that the particular square root chosen does not affect the value of the norm. It will become apparent that our choice of the norm in (24) is a natural one for our convergence results in this and the next section.

The following alternate characterization of $|||\cdot|||$ will be useful later.

LEMMA 1. *The norm in* (24) *may also be expressed as*

$$|||X||| = \sup_{\|G\|_{\infty,*}\leq 1} \sqrt{\mathbf{E}\left[\|GX\|_1^2\right]}, \tag{25}$$

where

$$\|G\|_{\infty,*} := \sum_{i=1}^{n} \max_{j=1,\ldots,n} |G_{ij}| \tag{26}$$

is the dual norm[7] *of the* ∞*-norm of* $G \in \mathbb{C}^{n\times n}$,

$$\|G\|_\infty = \max_{\|f\|_\infty\leq 1} \|Gf\|_\infty = \max_{i=1,\ldots,n} \sum_{j=1}^{n} |G_{ij}|.$$

Note that if the variable X is not random, then one can choose $f_i = X_i/|X_i|$ in (24) and find that $|||X||| = \|X\|_1$. When X is random we have the upper bound $|||X|||^2 \leq \mathbf{E}\left[\|X\|_1^2\right]$. If, on the other hand, X is random but has mean zero and independent components, then $|||X|||^2 = \mathbf{E}\left[\|X\|_2^2\right]$. We rely on the following lemma to establish a general relationship between these two norms.

LEMMA 2. *Let* A *be any* $n \times n$ *Hermitian matrix with entries in* $\mathbb{C}$. *Then*

$$\sup_{\|f\|_\infty\leq 1} f^{\mathrm{H}} A f \geq \operatorname{trace} A.$$

Lemma 2, applied to the second moment matrix of X, implies that $|||X|||^2 \geq \mathbf{E}\left[\|X\|_2^2\right]$. Summarizing these relationships we have

$$\mathbf{E}\left[\|X\|_2^2\right] \leq |||X|||^2 \leq \mathbf{E}\left[\|X\|_1^2\right]. \tag{27}$$

The norms appearing in (27) are all equivalent. What is important about the inequalities in (27) for our purposes is that they are independent of dimension.

Basic conditions. Consistent with results in the next section we will assume that the typical error from our compression rule is

$$|||\Phi_t^m(v) - v||| \leq \frac{\gamma}{\sqrt{m}} \|v\|_1 \tag{28}$$

for $v \in \mathbb{C}^n$, where γ is a constant that is independent of m and n. We will also assume that

$$\mathbf{E}\left[\|\Phi_t^m(v)\|_1^2\right] \leq C_b \|v\|_1^2 \tag{29}$$

for some constant C_b independent of m and n (for the compression scheme used in section 6, (29) is an equality with $C_b = 1$). For all of the compression methods detailed

[7] See [27, Proposition 7.2].

in section 5 (including the one used in our numerical experiments in section 6), the statistical consistency condition

$$\mathbf{E}\left[\Phi_t^m(v)\right] = v \tag{30}$$

is satisfied exactly, and we will assume that it holds exactly in this section. Modification of the results of this section to accommodate a bias $\|\mathbf{E}\left[\Phi_t^m(v)\right] - v\|_1 \neq 0$ is straightforward.

As a result of the appearance of $\|v\|_1$ in (28), in our eventual error bounds it will be impossible to avoid dependence on $\|V_t^m\|_1$. The growth of these quantities is controllable by increasing m, but the value of m required will often depend on the n. The next theorem concerns the size of $\mathbf{E}\left[\|V_t^m\|_1^2\right]$. After the statement and proof of the theorem we discuss how the various quantities appearing there can be expected to depend on n. In this theorem and in the rest of our results it will be convenient to recognize that, in many applications, the iterates V_t^m and $Y_t^m = \Phi_t^m(V_t^m)$ are confined within some subset of $\mathbb{C}^n$. For example, the iterates may all have a fixed norm or may have all nonnegative entries. We use the symbol $\mathcal{X}$ to identify this subset (which differs depending on the problem). Until Theorem 6 at the end of this section, our focus will be on iteration (11) though all of our results have analogues when (11) is replaced by iteration (12).

THEOREM 1. *Assume that V_t^m is generated by* (11) *with a compression rule satisfying* (28) *and* (30). *Suppose that $\mathcal{U}$ is a twice continuously differentiable function from $\mathcal{X}$ to $\mathbb{R}$, satisfying*

$$\mathcal{U}(\mathcal{M}(v)) \leq \alpha\,\mathcal{U}(v) + R \quad \textit{for all} \quad v \in \mathcal{X}$$

for some constants α and R, and

$$\|v\|_1^2 \leq \beta\,\mathcal{U}(v) \quad \textit{for all} \quad v \in \mathcal{X}$$

for some constant β. Assume that there are a constant σ and a matrix $G \in \mathbb{C}^{n\times n}$ satisfying $\|G\|_{\infty,*} \leq 1$, so that, for $z \in \mathbb{C}^n$,

$$\sup_{v\in\mathbb{C}^n} z^{\mathrm{H}}\left(D^2\mathcal{U}(v)\right)z \leq \sigma\|Gz\|_1^2,$$

where $D^2\mathcal{U}$ is the matrix of second derivatives of $\mathcal{U}$. Then

$$\mathbf{E}\left[\|V_t^m\|_1^2\right] \leq \beta R\left[\frac{1-\alpha^t\left(1+\frac{\beta\gamma^2\sigma}{2m}\right)^t}{1-\alpha\left(1+\frac{\beta\gamma^2\sigma}{2m}\right)}\right] + \beta\alpha^t\left(1+\frac{\beta\gamma^2\sigma}{2m}\right)^t\mathcal{U}(V_0^m).$$

First, the reader should notice that setting $\gamma = 0$ in Theorem 1 shows that the deterministic iteration (10) is stable whenever $\alpha < 1$. However, even for an $\mathcal{M}$ corresponding to a stable iteration, the randomized iteration (11) may not be stable (and will, in general, be less stable). If the goal is to estimate, e.g., a fixed point of $\mathcal{M}$, the user will first have to choose m large enough that the randomized iteration is stable.

Though it is not explicit in the statement of Theorem 1, in general the requirements for stability will depend on n. Consider, for example, the case of a linear iteration, $\mathcal{M}(v) = Kv$. This iteration is stable if the largest eigenvalue (in magnitude) is less than 1. If we choose $\mathcal{U}(v) = \|v\|_2^2$, then we can take α to be the largest eigenvalue

of $K^{\mathrm{H}}K$ and $R = 0$ in the statement of Theorem 1. The bound $\|\cdot\|_1 \leq \sqrt{n}\|\cdot\|_2$ (and the fact that it is sharp) suggests that we will have to take $\beta = n$ in Theorem 1 (note that we can take $\sigma = 1$ in the eventual bound). This scaling suggests that to guarantee stability we need to choose $m \sim n$.

Fortunately this prediction is often (but not always) pessimistic. For example, if K is a matrix with nonnegative real entries and V_0^m has nonnegative real entries, then the iterates V_t^m will have real, nonnegative entries (i.e., $v \in \mathcal{X}$ implies $v_i \geq 0$). We can therefore use $\mathcal{U}(v) = (\mathbb{1}^{\mathrm{T}}v)^2 = \|v\|_1^2$ for $v \in \mathcal{X}$ and find that we can take $\alpha = \|K\|_1^2$, $R = 0$, and $\beta = 1$ in the statement of Theorem 1. With this choice of $\mathcal{U}$ we can again choose $\sigma = 1$ so that n does not appear directly in the stability bound. We anticipate that most applications will fall somewhere between these two extremes; maintaining stability will require increasing m as n is increased, but not in proportion to the increase in n.

Having characterized the stability of our schemes we now move on to bounding their error. We have crafted the theorem below to address both situations in which one is interested in the error after a finite number of iterations and situations that require error bounds independent of the number of iterations. In general, achieving error bounds independent of the number of iterations requires that $\mathcal{M}$ satisfy stronger stability properties than those implied by the conditions in Theorem 1. While the requirements in Theorem 1 could be modified to imply the appropriate properties for most applications, we opt instead for a more direct approach and modify our stability assumptions on $\mathcal{M}$ to (31) and (32) below. In this theorem and below we will use the notation $\mathcal{M}_s^t$ to denote $\mathcal{M}$ composed with itself $t - s$ times. In our proof of the bound in Theorem 2 below we divide the error into two terms, one of which is a sum of individual terms with vanishing conditional expectations. Much like sums of independent, mean zero, random variables with finite variance, the size (measured by the square root of the second moment) of their sum can be expected to grow less than linearly with the number of iterations (see the proof of Theorem 2). This general observation is called dynamic self-averaging and results in an improved error bound. The improvement is essential in the context of perturbations of the identity and we will mention it again below when we focus on that case.

THEOREM 2. *Suppose that the iterates V_t^m of (11) remain in $\mathcal{X} \subset \mathbb{C}^n$. Fix a positive integer T. Assume that there are constants $\alpha \geq 0$, L_1, and L_2 such that for every pair of integers $s \leq r \leq T$ and for every vector $f \in \mathbb{C}^n$ with $\|f\|_\infty \leq 1$ there are matrices G and G' in $\mathbb{C}^{n\times n}$ satisfying $\|G\|_{\infty,*} \leq 1$ and a bounded, measurable $\mathbb{C}^{n\times n}$ valued function, A, such that*

$$\sup_{v,\tilde{v}\in\mathcal{X}} \frac{|f^{\mathrm{H}}\mathcal{M}_s^r(v) - f^{\mathrm{H}}\mathcal{M}_s^r(\tilde{v})|}{\|Gv - G\tilde{v}\|_1} \leq L_1\alpha^{r-s} \tag{31}$$

and

$$\sup_{v,\tilde{v}\in\mathcal{X}} \frac{|f^{\mathrm{H}}\mathcal{M}_s^r(v) - f^{\mathrm{H}}\mathcal{M}_s^r(\tilde{v}) - f^{\mathrm{H}}A(\tilde{v})(v-\tilde{v})|}{\|G'v - G'\tilde{v}\|_1^2} \leq L_2\alpha^{r-s}. \tag{32}$$

Then the error at step $t \leq T$ satisfies the bound

$$|||V_t^m - v_t||| \leq \alpha\left[\gamma\frac{1}{\sqrt{m}}(L_1 + L_2)\left(\frac{1-\alpha^{2t}}{1-\alpha^2}\right)^{1/2} M_t + \gamma^2\frac{1}{m}L_2\frac{1-\alpha^t}{1-\alpha}M_t^2\right],$$

where $M_t^2 = \sup_{r<t}\mathbf{E}\left[\|V_r^m\|_1^2\right]$.

Conditions (31) and (32) are easily verified for general linear maps $\mathcal{M} = K$ with $\alpha = \|K\|_1$. The conditions are more difficult to verify for the power iteration map

$$\mathcal{M}(v) = \frac{Kv}{\|Kv\|_1}.$$

A condition close to (31) holds for all $v, \tilde{v}$ with $\|v\|_1 = \|\tilde{v}\|_1 = 1$, but the parameter α in general depends on the proximity of v and $\tilde{v}$ to the space spanned by all of the nondominant eigenvectors (see, e.g., [60, Theorem 1.1]). For large enough m we can ensure that all iterates V_t^m remain at least some fixed distance from the space spanned by the nondominant eigenvectors, but this will often require that m grows with n.

As the following corollary establishes, when the matrix K is real and nonnegative we expect both matrix multiplication and power iteration to have errors independent of dimension.

COROLLARY 1. *Suppose that K is a real, entrywise nonnegative, irreducible, $n \times n$ matrix and that V_0^m is real and nonnegative with at most m nonzero entries. Then for both $\mathcal{M}(v) = Kv$ and $\mathcal{M}(v) = Kv/\|Kv\|_1$, the bound on $|||V_t^m - v_t|||$ in Theorem 2 is independent of dimension n. Let v_L and v_R be the unique dominant left and right eigenvectors of K with corresponding eigenvalue λ_*. If S is the stochastic matrix with entries $S_{ij} = \lambda_*^{-1} (v_L)_i K_{ij}/(v_L)_j$ and*

$$\alpha = \sup_{\substack{\|v\|_1 = 1 \\ \mathbb{1}^{\mathrm{T}} v = 0}} \|Sv\|_1,$$

then $\alpha < 1$ and the total error for the randomized iteration (11) with $\mathcal{M}(v) = Kv/\|Kv\|_1$ (i.e., for randomized power iteration) as an approximation of v_R is bounded by

$$|||V_t^m - v_R||| \leq C_1 \frac{\alpha}{1-\alpha} \frac{1}{\sqrt{m}} + C_2 \alpha^t |||V_0^m - v_R||| \tag{33}$$

for some constants C_1 and C_2 that depend on $\max_j\{(v_L)_j\}/\min_j\{(v_L)_j\}$, but do not otherwise depend on the iteration index, dimension, α, or K.

The total error bound on randomized power iteration in Corollary 1 is a finite-dimensional analogue of similar results concerning convergence of DMC and related schemes (see [15] and the references therein). In fact, the transformation from K to S in Corollary 1 is a finite-dimensional analogue of a transformation that is essential to the efficiency of QMC in practical applications (see the discussion of importance sampling in [26]) and that was used in [57] to establish error bounds for a QMC scheme by an argument similar to the proof of Corollary 1.

As discussed in section 3, when K is a dense matrix, the cost per iteration (measured in terms of floating point operations) of computing $\Phi_t^m(V_t^m)$ is $\mathcal{O}(n)$, while the cost of assembling the product $K\Phi_t^m(V_t^m)$ is $\mathcal{O}(mn)$. On the other hand, when K has at most q nonzero entries per column, the number of nonzero entries in V_t^m will be at most km so that the cost of computing $\Phi_t^m(V_t^m)$ is only $\mathcal{O}(km)$ and the cost of assembling $K\Phi_t^m(V_t^m)$ is only $\mathcal{O}(km)$. As a consequence of these observations and the bound in (33) we see that within any family of sparse (with a uniformly bounded number of nonzero entries per column) entrywise nonnegative matrices among which the parameter α is uniformly bounded below 1 and the ratio $\max_j\{(v_L)_j\}/\min_j\{(v_L)_j\}$ is uniformly bounded, the total cost to achieve a fixed accuracy is completely independent of dimension.

For more general problems one can expect the speedup over the standard deterministic power method to be roughly between a factor of n and no speedup at all (it is clear that the randomized scheme can be worse than its deterministic counterpart when that method is a reasonable alternative). Identification of more general conditions under which one should expect sublinear scaling for FRI in the particular context of power iteration seems to be a very interesting problem, but is not pursued here.

4.1. Bias. Even for a very general iteration the effect of randomization is evident when one considers the size of the expected error (rather than the expected size of the error). When $\mathcal{M}(v) = Kv$ and V_t^m is generated by (11), one can easily check that $z_t = \mathbf{E}\,[V_t^m]$ satisfies the iteration $z_{t+1} = Kz_t$, i.e., $z_t = v_t$. Even when the mapping $\mathcal{M}$ is nonlinear, the expected error, $\mathbf{E}\,[V_t^m] - v_t$, is often much smaller than the error, $V_t^m - v_t$, itself. The following is one simple result in this direction and demonstrates that one can often expect the bias to be $\mathcal{O}(m^{-1})$ (which should be contrasted with the expected error of $\mathcal{O}(m^{-1/2})$). The proof is very similar to the proof of Theorem 2 and is omitted from our list of proofs in the supplemental document.

THEOREM 3. *Under the same assumptions as in Theorem* 2 *and using the same notation, the bias at step $t \le T$ satisfies the bound*

$$\|\mathbf{E}\,[V_t^m] - v_t\|_1 \le \frac{\gamma^2 L_2}{m} \frac{\alpha(1-\alpha^t)}{1-\alpha} M_t^2.$$

4.2. Perturbations of Identity. When the goal is to solve ODEs or PDEs, stronger assumptions on the structure of $\mathcal{M}$ are appropriate. We now consider the case in which $\mathcal{M}$ is a perturbation of the identity. More precisely, we will assume that

$$\mathcal{M}(v) = v + \varepsilon\, b(v). \tag{34}$$

Though we will not write it explicitly, further dependence of b on ε is allowed as long as the assumptions on b below hold uniformly in ε. In the differential equations setting ε can be thought of as a time discretization parameter as in section 2.

An additional condition. When $\mathcal{M}$ is a perturbation of identity, it is reasonable to strengthen our assumptions on the error made at each compression step. The improvement stems from the fact that the mapping $\mathcal{M}$ nearly preserves the sparsity of its argument. As we will explain in detail in the next section, if $v \in \mathcal{S}_m$, where

$$\mathcal{S}_m = \{z \in \mathbb{C}^n : |\{j :\, z_j \neq 0\}| \le m\}$$

and $w \in \mathbb{C}^n$, then it is reasonable to assume that, for example,

$$|||\Phi_t^m(v+w) - v - w||| \le \frac{\gamma_p}{\sqrt{m}} \|w\|_1^{\frac{1}{2}} \|v+w\|_1^{\frac{1}{2}} \tag{35}$$

for some constant γ_p independent of m and n.

The following lemma illustrates how such a bound on the compression rule can translate into small compression errors when $\mathcal{M}$ is a perturbation of the identity.

LEMMA 3. *Suppose that the iterates V_t^m of* (11) *remain in $\mathcal{X} \subset \mathbb{C}^n$ and that the compression rule satisfies* (35) *and* (29). *Suppose that $\mathcal{M}(v) = v + \varepsilon b(v)$ with $\|b(v)\|_1 \le L(1 + \|v\|_1)$ for all $v \in \mathcal{X}$. Then for some constant $\tilde{\gamma}$,*

$$|||\Phi_t^m(V_t^m) - V_t^m|||^2 \le \tilde{\gamma}^2 \frac{\varepsilon}{m} \sqrt{\mathbf{E}\,[\|V_t^m\|_1^2]} \sqrt{1 + \mathbf{E}\left[\|V_{t-1}^m\|_1^2\right]}. \tag{36}$$

We now provide versions of Theorems 1 and 2 appropriate when $\mathcal{M}$ is a perturbation of identity. The proofs of both of these theorems are very similar to the proofs of Theorems 1 and 2 and are, at least in part, omitted. First we address stability in the perturbation of identity case.

THEOREM 4. *Suppose that the iterates V_t^m of (11) remain in $\mathcal{X} \subset \mathbb{C}^n$ and that the compression rule satisfies (35), (29), and (30). Suppose that $\mathcal{M}(v) = v + \varepsilon b(v)$ with $\|b(v)\|_1 \leq L(1 + \|v\|_1)$ for all $v \in \mathcal{X}$. Suppose further that $\mathcal{U}$ satisfies the conditions in the statement of Theorem 1 with the exception that*

$$\mathcal{U}(\mathcal{M}(v)) \leq e^{\varepsilon\alpha}\,\mathcal{U}(v) + \varepsilon R.$$

Then

$$\sup_{t<T/\varepsilon} \mathbf{E}\left[\|V_t^m\|_1^2\right] \leq \beta R\left[\varepsilon + \frac{\exp\left[T\left(\alpha + \frac{\beta\tilde{\gamma}^2\sigma}{2m}\right)\right] - 1}{\alpha + \frac{\beta\tilde{\gamma}^2\sigma}{2m}}\right] + \beta \exp\left[T\left(\alpha + \frac{\beta\tilde{\gamma}^2\sigma}{2m}\right)\right]\mathcal{U}(V_0^m),$$

where $\tilde{\gamma}$ is the constant appearing in (36) and β and σ are defined in the statement of Theorem 1.

What is important about the statement of Theorem 4 is that the bound remains stable as ε decreases, despite the fact that the set being supremized over is increasing. Under the assumptions in the theorem (which are only reasonable when $\mathcal{M}$ is a perturbation of the identity) one can expect that the iterates can be bounded over $\mathcal{O}\left(\varepsilon^{-1}\right)$ iterations uniformly in ε.

The following theorem interprets the result of Theorem 2 when $\mathcal{M}$ is a perturbation of identity. One might expect that, over $\mathcal{O}(\varepsilon^{-1})$ iterations, $\mathcal{O}\left(\sqrt{\varepsilon}\right)$ errors made during the compression step would accumulate and lead to an error of $\mathcal{O}(\varepsilon^{-1/2})$. Indeed, this is exactly what would happen if the errors made in the compression step were systematic (i.e., if the compression bias was $\mathcal{O}\left(\sqrt{\varepsilon}\right)$). Fortunately, when the compression rule satisfies the consistency criterion (30) the errors self-average and their effect on the overall error of the scheme is reduced. As mentioned above, this phenomenon played a role in the structure of the result in Theorem 2 and its proof, but its role is more crucial in Theorem 5 which provides uniform in ε bounds on the error of (11) over $\mathcal{O}(\varepsilon^{-1})$ iterations. Without the reduction in the growth of the error with t provided by self-averaging it would not be possible to achieve an error bound over $\mathcal{O}\left(\varepsilon^{-1}\right)$ iterations that is stable as ε decreases.

THEOREM 5. *Suppose that the iterates V_t^m of (11) remain in $\mathcal{X} \subset \mathbb{C}^n$ and that the compression rule satisfies (35), (29), and (30). Suppose that $\mathcal{M}(v) = v + \varepsilon b(v)$ with $\|b(v)\|_1 \leq L(1 + \|v\|_1)$ for all $v \in \mathcal{X}$. Fix a real number $T > 0$ and assume that, for some real number α and some constants L_1 and L_2 and for every pair of integers $s \leq r \leq T/\varepsilon$, for every vector $f \in \mathbb{C}^n$ with $\|f\|_\infty \leq 1$, there are matrices G and G' in $\mathbb{C}^{n\times n}$ satisfying $\|G\|_{\infty,*} \leq 1$ and a bounded, measurable $\mathbb{C}^{n\times n}$ valued function A such that*

$$\sup_{v,\tilde{v}\in\mathcal{X}} \frac{|f^{\mathrm{H}}\mathcal{M}_s^r(v) - f^{\mathrm{H}}\mathcal{M}_s^r(\tilde{v})|}{\|Gv - G\tilde{v}\|_1} \leq L_1 e^{-\varepsilon\alpha(r-s)} \tag{37}$$

and

$$\sup_{v,\tilde{v}\in\mathcal{X}} \frac{|f^{\mathrm{H}}\mathcal{M}_s^r(v) - f^{\mathrm{H}}\mathcal{M}_s^r(\tilde{v}) - f^{\mathrm{H}}A(\tilde{v})(v-\tilde{v})|}{\|Gv - G\tilde{v}\|_1^2} \leq L_2 e^{-\varepsilon\alpha(r-s)}. \tag{38}$$

Then the error at step $t \leq T/\varepsilon$ satisfies the bound

$$|||V_t^m - v_t||| \leq \frac{\tilde{\gamma}(L_1+L_2)}{\sqrt{m}} \left(e^{-2\alpha T} \mathbf{E}\left[\|V_0^m\|_1^2\right] + \frac{1-e^{-2\alpha T}}{2\alpha} M_T \sqrt{1+M_T^2} \right)^{1/2} + \frac{\tilde{\gamma}^2 L_2}{m} \left(e^{-2\alpha T} \mathbf{E}\left[\|V_0^m\|_1^2\right] + \frac{1-e^{-\alpha T}}{\alpha} M_T \sqrt{1+M_T^2} \right),$$

where $M_T^2 = \sup_{r<T/\varepsilon} \mathbf{E}\left[\|V_r^m\|_1^2\right]$.

Though the error established in the last claim is stable as ε decreases, we have mentioned in section 3.2 that when $\mathcal{M}$ is a perturbation of identity, by using iteration (12) instead of (11), one might be able to obtain errors that vanish as ε decreases (keeping m fixed). This is the subject of Theorem 6 below, which, like Theorem 5, relies crucially on self-averaging of the compression errors. Note that iteration (12) typically requires $\mathcal{O}(n)$ operations per iteration and storage of length n vectors. We have the following theorem demonstrating the decrease in error with ε in this setting.

THEOREM 6. *Suppose that the iterates V_t^m of* (12) *remain in $\mathcal{X} \subset \mathbb{C}^n$ and that* (28) *holds. Under the same assumptions on $\mathcal{M}$ as in Theorem* 5*, the error at step $t \leq T/\varepsilon$ satisfies the bound*

$$\sup_{t \leq T/\varepsilon} |||V_t^m - v_t||| \leq \frac{\sqrt{\varepsilon}\gamma}{\sqrt{m}}(L_1+L_2)L_1 \left(\frac{1-e^{-2\alpha T}}{2\alpha}\right)^{\frac{1}{2}} e^{\alpha\varepsilon} M_T + \frac{\varepsilon\gamma^2 L_2 L_1^2}{m} \left(\frac{1-e^{-\alpha T}}{\alpha}\right) M_T^2,$$

where $M_T^2 = \sup_{r<T/\varepsilon} \mathbf{E}\left[\|V_r^m\|_1^2\right]$.

5. Compression Rules. In this section we give a detailed description of the compression rule used in our numerical simulations as well as several others, and an analysis of the accuracy of those schemes. Programmed efficiently, and assuming that v has exactly n nonzero entries, all of the schemes we discuss in this section will require at most $\mathcal{O}(n)$ floating point operations including the generation of as few as one uniform random variate and $\mathcal{O}(n + m \log n)$ floating point comparisons. It is likely that better compression schemes are possible, for example, by incorporation of ideas from [33]. The reader should note that in this section n represents the number of nonzero entries in the input vector v of the compression rule and not the dimension associated with a particular problem (which may be much larger). In our implementation of (11), when the underlying matrix is sparse (so that an $\mathcal{O}(m)$ work/storage per iteration method is possible) we store only the indices and values of the nonzero entries in any vector (including matrix columns).

We begin by discussing the simple choice

$$(\Phi_t^m(v))_j = \begin{cases} N_j \frac{\|v\|_1}{m} \frac{v_j}{|v_j|} & \text{if } |v_j| > 0, \\ 0 & \text{if } |v_j| = 0, \end{cases} \tag{39}$$

where each N_j is a random, nonnegative, integer with

$$\mathbf{E}\left[N_j \mid v\right] = \frac{m|v_j|}{\|v\|_1}, \tag{40}$$

so that $\mathbf{E}\left[\Phi_t^m(v)\right] = v$ and the consistency condition (30) is satisfied. Notice that if we define a collection of $N = \sum_{j=1}^n N_j$ integers $\{X^{(j)}\}$ so that exactly N_j elements

of the collection are equal to j, then the output of a compression scheme of this type can be written

$$\Phi_t^m(v) = \frac{\|v\|_1}{m} \sum_{j=1}^{N} \frac{v_{X^{(j)}}}{|v_{X^{(j)}}|} \mathbf{e}_{X^{(j)}},$$

where $\mathbf{e}_j$ is the jth standard basis vector in $\mathbb{R}^n$. When the input vector v is real, $\Phi_t^m(v)$ is a finite-dimensional analogue of the DMC resampling step described in section 2. In the infinite-dimensional setting the efficiency of DMC is due to the application of an integral operator, $e^{-\varepsilon\mathcal{H}}$, to a finite sum of delta functions in place of a more general function. Likewise, the gain in efficiency of an FRI scheme over deterministic methods is a consequence of the replacement of a general vector v in the product Kv by a sparse approximation, $\Phi_t^m(v)$. Though we will deviate somewhat from the form in (13) to arrive at the compression scheme used in the numerical simulations reported on in section 6, essential elements of (13) will be retained.

Notice that the consistency condition (40) leaves substantial freedom in the specification of the joint distribution of the N_j. For example, one simple choice might be to select the vector of N_j from the multinomial distribution with parameters m and $(|v_1|, |v_2|, \ldots, |v_n|)/\|v\|_1$. This choice would result in a compression scheme satisfying (28), (29), and (30), as required in Theorems 1 and 2 in section 4. However, it would not satisfy (35) and would be a particularly poor choice when $\mathcal{M}$ is a perturbation of the identity. In fact, this choice would lead to unstable schemes as the size of the perturbation decreases. An alternative, much more accurate choice that will lead below (in Lemma 5) to a compression scheme satisfying (35) is to select the N_j independently with

$$\mathbf{P}\left(N_j = \left\lceil \frac{m|v_j|}{\|v\|_1} \right\rceil\right) = 1 - \mathbf{P}\left(N_j = \left\lfloor \frac{m|v_j|}{\|v\|_1} \right\rfloor\right) = \frac{m|v_j|}{\|v\|_1} - \left\lfloor \frac{m|v_j|}{\|v\|_1} \right\rfloor. \tag{41}$$

Note that this rule randomly rounds $m|v_j|/\|v\|_1$ to a nearby integer and satisfies (40). The compression rule (39) with (41) has already appeared in (13). When v has exactly n nonzero entries, the corresponding cost to assemble Φ_t^m by this rule is $\mathcal{O}(n)$ operations.

However, we have emphasized repeatedly in this article that the cost saving at each iteration of an FRI scheme is entirely due to sparsity introduced by our compressions. In addition, the results of the last section reveal that compression schemes with large variance will typically give rise to FRI schemes with large error. In this regard the compression rule in (39) is clearly suboptimal. In particular, for any entry j for which $m|v_j|/\|v\|_1 > 1$, the jth component of $\Phi_t^m(v)$ is nonzero with probability 1, so that the error $(\Phi_t^m(v))_j - v_j$ is not compensated for by an increase in sparsity. To improve the scheme we can introduce a rule for exactly preserving sufficiently large entries of v. To that end, let σ be a permutation of $\{1, 2, \ldots, n\}$ such that the elements of v_σ have decreasing magnitude (i.e., v_σ is a rearrangement of the entries of v so that, for each j, $|v_{\sigma_j}| \geq |v_{\sigma_{j+1}}|$) and let

$$\tau_v^m = \min\left\{ 0 \leq \ell \leq m : \sum_{j=\ell+1}^{n} |v_{\sigma_j}| \geq (m-\ell)|v_{\sigma_{\ell+1}}| \right\}. \tag{42}$$

All of the compression schemes we consider will preserve entries $v_{\sigma_1}, v_{\sigma_2}, \ldots, v_{\sigma_{\tau_v^m}}$ exactly. In fact, they will have the basic structure as given in Algorithm 1.

To justify preservation of the τ_v^m largest entries in our compression schemes, we need the following lemma.

Algorithm 1 A simple compression rule.

Data: $v \in \mathbb{C}^n$ with all nonzero entries, $m \in \mathbb{N}$.
Result: $V = \Phi^m(v) \in \mathbb{C}^n$ with at most m nonzero entries.
$\tau_v^m = 0$;
$V = 0$;
$r = \|v\|_1/m$;
$\sigma_1 = \arg\max_i\{|v_i|\}$;
while $|v_{\sigma_{\tau_v^m+1}}| \geq r$ **do**
  $\tau_v^m = \tau_v^m + 1$;
  $V_{\sigma_{\tau_v^m}} = v_{\sigma_{\tau_v^m}}$;
  $v_{\sigma_{\tau_v^m}} = 0$;
  $r = \|v\|_1/(m - \tau_v^m)$;
  $\sigma_{\tau_v^m+1} = \arg\max_i\{|v_i|\}$;
end
For each j let N_j be a nonnegative random integer with $\mathbf{E}[N_j \mid v] = (m-\tau_v^m)|v_j|/\|v\|_1$. Finally, for $j \in \{1, 2, \ldots, n\} \setminus \{\sigma_1, \sigma_2, \ldots, \sigma_{\tau_v^m}\}$, set

$$V_j = N_j \frac{v_j \|v\|_1}{|v_j|(m - \tau_v^m)}.$$

(Note that v here may fewer nonzero entries than it did upon input.)

LEMMA 4. *τ_v^m satisfies*

$$\sum_{j=\tau_v^m+1}^{n} |v_{\sigma_j}| \leq \frac{m - \tau_v^m}{m} \|v\|_1.$$

Note that for any compression scheme satisfying an error bound of the form (28) for a general vector $v \in \mathbb{C}^n$, the error resulting from application of the compression scheme after exact preservation of the largest τ_v^m entries is bounded by

$$\frac{\gamma}{\sqrt{m - \tau_v^m}} \sum_{j=\tau_v^m+1}^{n} |v_{\sigma_j}|,$$

which, by Lemma 4, is itself bounded by

$$\gamma \frac{\sqrt{m - \tau_v^m}}{m} \|v\|_1$$

and is always an improvement over (28).

Lemma 5 summarizes the properties of the compression scheme resulting from preserving the largest τ_v^m entries of an input vector $v \in \mathbb{C}^n$ exactly and applying (39) with (41) with m replaced by $m - \tau_v^m$ to the remaining entries. In particular, Lemma 5 implies that the compression scheme satisfies conditions (28), (29), and (35).

LEMMA 5. *Let $v, w \in \mathbb{C}^n$ and assume that v has at most m nonzero entries. For Φ_t^m defined by Algorithm 1 with (41),*

$$\|\|\Phi_t(v + w) - v - w\|\| \leq \sqrt{2} \frac{\|w\|_1^{\frac{1}{2}} \|v + w\|_1^{\frac{1}{2}}}{\sqrt{m}}. \tag{43}$$

Concerning the size of the resampled vector we have the bound

$$\mathbf{E}\left[\|\Phi_t^m(v+w)\|_1^2\right] \leq \|v+w\|_1^2 + 2\frac{\|v+w\|_1\|w\|_1}{m}.$$

Finally, if $\tau_{v+w}^m > 0$, *then* $\mathbf{P}\left[\Phi_t(v+w)=0\right]=0$. *If* $\tau_{v+w}^m = 0$, *then*

$$\mathbf{P}\left[\Phi_t(v)=0\right] \leq \left(\min\left\{\frac{\|w\|_1}{\|v+w\|_1}, \frac{1}{e}\right\}\right)^m.$$

In practice this compression scheme would need to be modified to avoid the possibility that $\Phi_t^m(v)=0$. As Lemma 5 demonstrates, the probability of this event is extremely small. The issue can be avoided by simply sampling $\Phi_t^m(v)$ until $\Phi_t^m(v)\neq 0$, i.e., sampling $\Phi_t^m(v)$ conditioned on the event $\{\Phi_t^m(v)\neq 0\}$ and multiplying each entry of the resulting vector by $\mathbf{P}\left[\Phi_t^m(v)\neq 0\right]$, which can be computed exactly. A more significant issue is that while Lemma 5 does guarantee that the compression scheme just described satisfies (35), the scheme does not guarantee that the number of nonzero entries in $\Phi_t^m(v)$ does not exceed m as required by Lemma 3 in the last section. The results of that section can be modified accordingly or the compression scheme can be modified so that $\Phi_t^m(v)$ has no more than m nonzero entries (by randomly selecting additional entries to set to zero). Instead of pursuing these modifications here we move on to describe the compression scheme used to generate the results reported in the next section.

Like the compression scheme considered in Lemma 5, the compression scheme used to generate the results in section 6 begins with an application of Algorithm 1. To fully specify the scheme we need to specify the rule used to generate the random variables N_j for $j \in \{\sigma_k : k > \tau_v^m\}$. For $k = 1, 2, \ldots, m-\tau_v^m$, define the random variables

$$U^{(k)} = \frac{1}{m-\tau_v^m}\left(k-1+U\right), \tag{44}$$

where U is a single uniformly chosen random variable on the interval (0,1). We then set

$$N_{\sigma_j} = \left|\left\{k : U^{(k)} \sum_{j=\tau_v^m+1}^{n} |v_{\sigma_j}| \in I_j\right\}\right|, \tag{45}$$

where we have defined the intervals $I_{\tau_v^m+1} = \left[0, |v_{\sigma_{\tau_v^m+1}}|\right)$ and, for $i = \tau_v^m+2, \ldots, n$,

$$I_i = \left[\sum_{j=\tau_v^m+1}^{i-1} |v_{\sigma_j}|, \sum_{j=\tau_v^m+1}^{i} |v_{\sigma_j}|\right). \tag{46}$$

As for the rule in (41), the variables N_j generated according to (45) satisfy

$$\mathbf{E}\left[N_j \mid v\right] = \frac{(m-\tau_v^m)|v_j|}{\sum_{i=\tau_v^m+1}^n |v_{\sigma_i}|},$$

so that the compression mapping Φ_t^m resulting from use of (45) with Algorithm 1 satisfies (30). From the definition of τ_v^m, we know that for $j > \tau_v^m$, $(m-\tau_v^m)|v_j| \leq \sum_{i=\tau_v^m+1}^n |v_{\sigma_i}|$, which implies by (45) that $N_j \in \{0,1\}$.

Unlike (41), (45) results in N_j that are correlated and satisfy $\sum_{j=\tau_v^m+1}^{n} N_j = m - \tau_v^m$ exactly (not just in expectation). The corresponding compression scheme exactly preserves the ℓ_1-norm of v and results in a vector $\Phi_t^m(v)$ with at most m nonzero entries. Note that this compression scheme, like the one considered in Lemma 5, only requires knowledge of the set $\{\sigma_1, \sigma_2, \ldots, \sigma_{\tau_v^m}\}$ and does not require sorting of the entire input vector v. Perhaps the most obvious advantage of this scheme over the one that generates the N_j according to (41) is that the compression scheme using (45) only requires a single random variate per iteration (compared to up to n for (41)). Depending on the cost of evaluating $\mathcal{M}(V_t^m)$, this advantage could be substantial.

Notice that if we replaced the $U^{(k)}$ in (44) by independent random variables uniformly chosen in $(0, 1)$, then the N_j would be distributed multinomially, which we have already mentioned is a poor choice. Relative to multinomial N_j, the increased correlation between the $U^{(k)}$ defined in (44) results in substantially decreased variance for the N_j, but also increased covariance. An unfortunate consequence of this increased covariance is that the analogue of the error bound (43) from Lemma 5 does not hold for N_j generated according to (45). In fact, the rule in (45) is very closely related to the systematic resampling scheme used frequently in the context of sequential Monte Carlo (see, e.g., [19]), which is well known to fail to converge for certain sequences of input vectors.[8] Nonetheless, in unreported numerical comparisons we found that the rule (45) results in FRI schemes with significantly lower error than for (41).

6. Numerical Tests. In this section we describe the application of the framework above to particular matrices arising in (i) the computation of the per-spin partition function of the two-dimensional Ising model, (ii) the spectral gap of a diffusion process governing the evolution of a system of up to five two-dimensional particles (i.e., up to ten spatial dimensions), and (iii) a free energy landscape for that process. The corresponding numerical linear algebra problems are, respectively, (i) computing the dominant eigenvalue/eigenvector of matrices up to size $10^{15} \times 10^{15}$, (ii) computing the second largest eigenvalue/eigenvector of matrices up to size $10^{20} \times 10^{20}$, and (iii) solving a linear system involving exponentiation of matrices up to size $10^{20} \times 10^{20}$. Aside from sparsity, these matrices have no known readily exploitable structure for computations.

All but the first test problem involve matrices with entries of any sign. As we learned in section 4, we can often expect much better error scaling with dimension when applying FRI to problems involving matrices with all nonnegative entries. The numerical results in this section suggest that dramatic speedups are possible even for more general matrices.

The reader might wonder why we consider random compressions instead of simple thresholding, i.e., a compression rule in which, if σ_j is the index of the jth largest (in magnitude) of v, v_{σ_j} is simply set to zero for all $j > m$ (the resulting vector can be normalized to preserve ℓ^1-norm, or not). In the rest of this paper we will refer to methods using such a compression rule as truncation-by-size (TbS) schemes. TbS schemes have been considered by many authors (see, e.g., [29, 58, 49, 50]) and are a natural approach. Note, however, that the error (if the compression is not normalized),

$$\left| \sum_{j>m} \bar{f}_{\sigma_j} v_{\sigma_j} \right|,$$

[8] If applied to the same vector v as m increases, the scheme does converge.

for the thresholding compression can be as large as $\|f\|_\infty \|v\|_1 (1 - m/n)$, which only vanishes if m is increased faster than n. In contrast, the random compressions above can have vanishing error even when n is infinite. This observation is key to understanding the substantial reduction in error we find for our fast randomized scheme over TbS in numerical results presented in this section. In our first test example the TbS scheme converges to a value far from a high quality estimate of the true value (a relative error of 98% compared to 8% for FRI). In the subsequent examples the TbS iteration appears to converge (in the iteration index t) to substantially different values for each fixed choice of m, whereas FRI shows much more consistent behavior in m. Moreover, in practice we observe no cost savings per iteration for TbS over FRI.

Finally we comment that, in order for the FRI approach to yield significant performance improvements, one must use an efficient implementation of matrix by sparse vector multiplication. In the examples below we list the i, j pairs for which the product $K_{ij} v_j$ is nonzero, then sort the products according to the i-index and, finally, add the products with common i. This is a simple and suboptimal solution. More details can be found in the example code available in [63].

6.1. A Transfer Matrix Eigenproblem. In this example we find the dominant eigenvalue λ_* of the transfer matrix K of the two-dimensional ℓ-spin Ising model. This eigenvalue is the per-spin partition function of the infinite-spin Ising model, i.e.,

$$\lambda_*(T, B) = \lim_{\ell \to \infty} \left(\sum_{\sigma} e^{\frac{1}{T} \sum_{|(i,j)-(i',j')|=1} \sigma_{ij} \sigma_{i'j'} + B\sigma_{ij}} \right)^{1/\ell},$$

where $\sigma_{ij} \in \{-1, 1\}$ and the sum in the exponent is over pairs of indices on a square two-dimensional, periodic lattice with ℓ sites. The outer sum is over all 2^ℓ possible values of σ, and for larger ℓ, one cannot possibly compute it directly. The matrix K is $2^\ell \times 2^\ell$. For example, in the case $\ell = 3$,

$$K = \begin{bmatrix} a & & & a^{-1} & & & & \\ b & & & b^{-1} & & & & \\ & a & & & a^{-1} & & & \\ & b & & & b^{-1} & & & \\ & & b & & & b^{-1} & & \\ & & c & & & c^{-1} & & \\ & & & b & & & b^{-1} & \\ & & & c & & & c^{-1} & \end{bmatrix}, \tag{47}$$

where

$$a = e^{(2-B)/T}, \quad b = e^{-B/T}, \quad c = e^{-(2+B)/T}.$$

We therefore cannot hope to apply the power method (or its relatives) directly to K when ℓ is large. In our experiments we set $T = 2.2$, $B = 0.01$, and $\ell = 50$ so that $n = 2^\ell > 10^{15}$. We apply both the FRI and TbS, $\mathcal{O}(1)$ schemes to computing λ_* as well as to computing the sum of all components of the corresponding eigenvector, v_* (normalized to have sum equal to 1), with index greater than or equal to 2^{49}, i.e.,

$$f_* = \sum_{j \geq n/2} (v_*)_j.$$

Knowledge of the partition function λ_* as a function of temperature T and field strength B allows one to determine useful quantities such as the average magnetization (sum of spin values) and to diagnose phase transitions [29]. Our choice to estimate

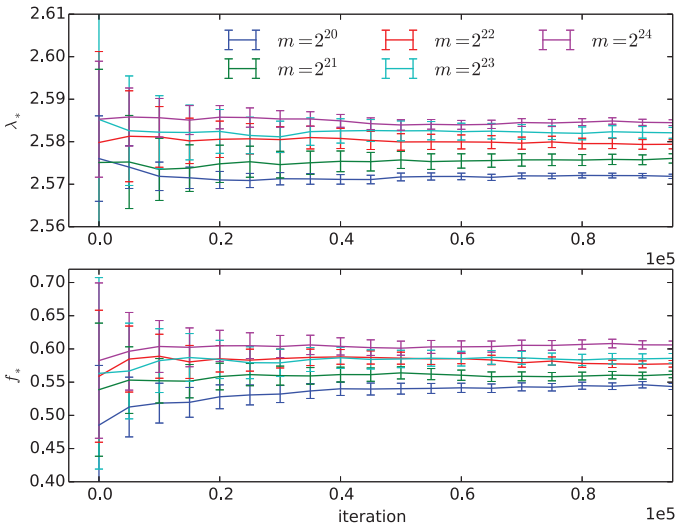

Fig. 1 Top: *Trajectory averages of the approximation,* Λ_t^m*, of the partition function for the* 50*-spin Ising model with* $B = 0.01$ *and* $T = 2.2$*, with* 95% *confidence intervals*[9] *as computed by the FRI method with* $m = 2^k$ *for* $k = 20, 21, 22, 23,$ *and* 24*. The best (highest* m*) estimate for* λ_* *is* 2.584*, a difference of roughly* 0.5% *from the value for the* 24*-spin Ising model.* Bottom: *Corresponding trajectory averages of the approximation,* F_t^m*, of the total weight of all components of* v_* *with index greater than or equal to* 2^{49} *with* 95% *confidence intervals for the FRI method. The best (highest* m*) estimate for* f_* *is* 0.606*, a difference of roughly* 8% *from the value for the* 24*-spin model.*

λ_* and f_* is motivated in part by the fact that these quantities can be approximated accurately by the corresponding values for smaller Ising systems. We will compare our results to those for the 24-spin Ising model which we can solve by standard power iteration. For an effective specialized method for this problem, see [51]. A simple, educational implementation of FRI applied to this problem can be found in [63].

In Figure 1 we report the trajectory averages of the approximations Λ_t^m and F_t^m generated by the FRI scheme (iteration (17) using Algorithm 1) with $m = 2^{20}$, 2^{21}, 2^{22}, 2^{23}, and 2^{24} and 10^5 total iterations. The best (highest m) approximation of λ_* is 2.584 and the best approximation of f_* is 0.606. The results for the 24-spin Ising problem are $\lambda_* \approx 2.596$ and $f_* \approx 0.658$, a difference of roughly 0.5% and 8% from the respective approximations generated by the FRI method. In Figure 2 we plot the corresponding trajectories of Λ_t^m and F_t^m. These plots strongly suggest that the iteration equilibrates rapidly (relative to the total number of iterations). Indeed, we estimate the integrated autocorrelation times[10] of Λ_t^m and F_t^m to be 20.5 and 274, respectively. This in turn suggests that one could achieve a dramatic speedup by running many parallel and independent copies (replicas) of the simulation and averaging the resulting estimates of λ_* and f_*, though we have not taken advantage of this here.

[9]We use the term "confidence intervals" loosely. We plot confidence intervals for the values $\lim_{t\to\infty} \mathbf{E}\,[\Lambda_t^m]$ and $\lim_{t\to\infty} \mathbf{E}\,[F_t^m]$ (i.e., for finite m) and not for λ_* and f_*. In other words, our confidence intervals do not account for bias resulting from a finite choice of m.

[10]According to the central limit theorem for Markov processes (assuming it holds), for large t the variance of the trajectory average of Λ_t^m should be $\sigma^2\tau/t$, where σ^2 is the infinite t limit of the variance of Λ_t and τ is the integrated autocorrelation time of Λ_t^m. Roughly, it measures the number of iterations between independent Λ_t^m.

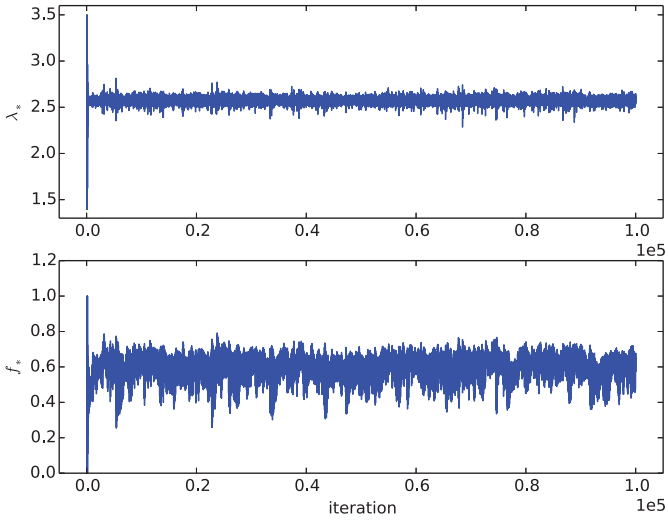

Fig. 2 Top: *Trajectory of the approximation,* Λ_t^m*, of the partition function for the* 50*-spin Ising model, for the FRI method with* $m = 2^{24}$*. The approximate integrated autocorrelation time for* Λ_t^m *is* 20.5 *iterations.* Bottom: *Corresponding trajectory of* F_t^m *as computed by the FRI method. The approximate integrated autocorrelation time for* F_t^m *is* 274 *iterations.*

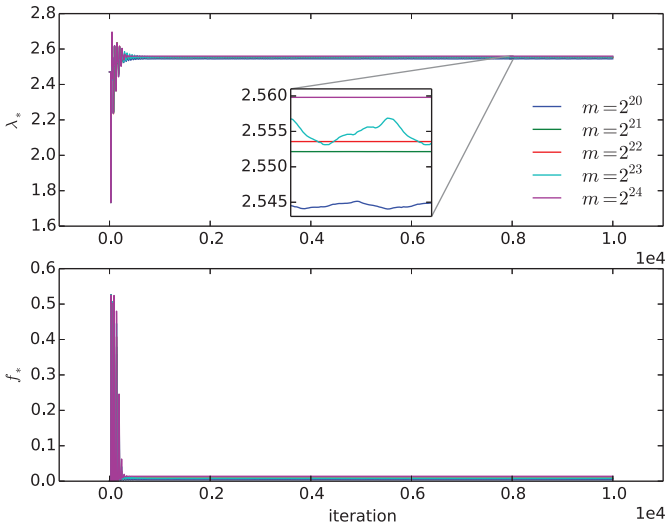

Fig. 3 Top: *Trajectory of the approximation,* Λ_t^m*, of the partition function for the* 50*-spin Ising model, for the TbS method with* $m = 2^{24}$*. The best (highest* m*) approximation is* $\lambda_* \approx 2.545$*, a difference of about* 2% *from the value for the* 24*-spin Ising model.* Bottom: *Corresponding trajectory of* F_t^m *as computed by the TbS method. The best (highest* m*) approximation is* $f_* \approx 0.014$*, a difference of almost* 98% *from the value for the* 24*-spin model.*

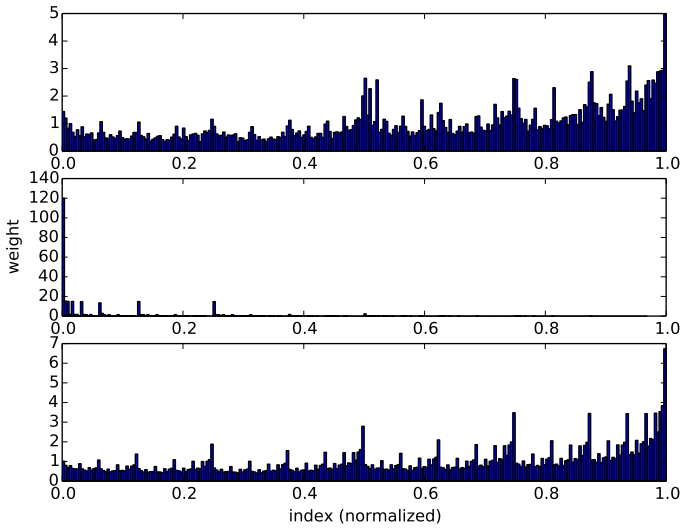

Fig. 4 Top: *Sums of the values of the approximation, V_t^m, of the dominant eigenvector of the* 50*-spin Ising transfer matrix with $B = 0.01$ and $T = 2.2$, at $t = 10^5$ over* 256 *intervals of equal size out of the 2^{50} total indices for the FRI method with $m = 2^{24}$.* Middle: *Corresponding sums for the TbS method.* Bottom: *Exact eigenvector for the* 20*-spin Ising model for qualitative comparison.*

Figure 3 reports the analogous trajectories (see (48) below) of Λ_t^m and F_t^m as generated by iteration (17) with the TbS scheme (iteration (17) using truncation-by-size) and the same values of m. The best (highest m) TbS approximation of λ_* is 2.545 and the best TbS approximation of f_* is 0.014, a difference of almost 2% and 98%, respectively. In Figure 4 we plot the sums of the values of the approximation, V_t^m, of the dominant eigenvector of the Ising transfer matrix at $t = 10^5$ over 256 intervals of equal size out of the 2^{50} total indices. The top plot represents V_t^m as generated by the FRI method and the middle plot represents V_t^m as generated by the TbS approach. The TbS iteration has converged to a vector with nearly all of weight concentrated on very low indices. The bottom plot in Figure 4 represents the dominant eigenvector for the 24-spin Ising transfer matrix. The qualitative agreement with the realization of V_t^m as generated by the FRI method is much stronger than agreement with the result of the TbS method.

Remark 7. In this problem we compute the dominant eigenvalue λ_* and a projection f_* of the dominant eigenvector v_* of the matrix K defined in (47) using the FRI in conjunction with the power method. Using the trajectory averages

$$\bar{\Lambda}_t^m = \frac{1}{t}\sum_{s=1}^{t} \Lambda_t^m \qquad \text{and} \qquad \bar{F}_t^m = \frac{1}{t}\sum_{s=1}^{t} F_t^m \tag{48}$$

to estimate λ_* and f_* would seem strange had the iterates Λ_t^m and F_t^m been generated by the deterministic power method (we have not reported trajectory averages for the deterministic TbS approach). However, for finite m we do not expect Λ_t^m or F_t^m to converge to λ_* and f_* as t increases. Rather we expect that the distribution of Λ_t^m and F_t^m will converge to some distribution roughly centered around λ_* and f_*,

respectively. Though in our convergence results we have not addressed the ergodicity of the Markov process V_t^m, one would expect that reasonable functions of V_t^m such as Λ_t^m and F_t^m satisfy a law of large numbers so that, for very large t, the trajectory averages $\bar{\Lambda}_t^m$ and $\bar{F}_t^m$ differ from λ_* and f_* only by a systematic error (i.e., they converge to the limit of the expectations of Λ_t and F_t^m, respectively).

6.2. A PDE Eigenproblem. For given functions $b(x)$ and $\sigma(x)$ with values $\mathbb{R}^n$ and $\mathbb{R}^n \times \mathbb{R}^r$, the backwards Kolmogorov operator

$$Lf = b^{\mathrm{T}} Df + \frac{1}{2} \operatorname{trace}\left(\sigma\sigma^{\mathrm{T}} D^2 f\right) \tag{49}$$

is the generator of the diffusion process

$$dX(t) = b(X(t))\, dt + \sigma(X(t))\, dW(t), \tag{50}$$

where $W(t)$ is an r-dimensional Brownian motion, Df is the vector of first order derivatives of f, and $D^2 f$ is the matrix of its second order derivatives. The operator L governs the evolution of moments of $X(t)$ in the sense that

$$\frac{d}{dt} \mathbf{E}_x \big[f(X(t)) \big] \Big|_{t=0} = Lf(x)$$

(the subscript on the expectation indicates that $X_0 = x$). Note that constant functions are in the kernel of L. The nontrivial eigenfunctions of L all correspond to negative eigenvalues. The magnitude of the greatest of these negative eigenvalues is the spectral gap and characterizes the rate of convergence of expectations such as $\mathbf{E}_x \big[f(X(t)) \big]$ to their equilibrium (large t) values.

In this section we consider estimation of the largest negative eigenvalue of L for $b = -DV$ with

$$V(x^{(1)}, \ldots, x^{(\ell)}) = \frac{1}{2} \sum_{j=1}^{\ell} \cos(2\pi x_1^{(i)}) \cos(2\pi x_2^{(j)}) + 2 \sum_{j=1}^{\ell} \sum_{k=j+1}^{\ell} \cos(\pi(x_1^{(j)} - x_1^{(k)})) \cos(\pi(x_2^{(j)} - x_2^{(k)}))$$

for $x^{(j)} = (x_1^{(j)}, x_2^{(j)}) \in [-1, 1) \times [-1, 1)$. The diffusion coefficient, $\sigma(x)$, is fixed as $\sqrt{2}$. The function V is the potential energy for a periodic system of ℓ, two-dimensional particles, each subject to both an external force as well as a nonlinear spring coupling the particles together. Equation (50) is a model of the dynamics of that system of particles (in a high friction limit).

The equation $Lg_* = \lambda_* g_*$ is first projected onto a Fourier spectral basis, i.e., we assume that

$$g_*(\vec{x}) = \sum_{\vec{\alpha} \in \mathbb{Z}_N^{2\ell}} v(\vec{\alpha})\, e^{i\pi \vec{\alpha}^{\mathrm{H}} \vec{x}},$$

where $\vec{\alpha} = (\alpha^{(1)}, \ldots, \alpha^{(\ell)})$ with $\alpha^{(j)} = (\alpha_1^{(j)}, \alpha_1^{(j)})$, and the symbol $\mathbb{Z}_N^{2\ell}$ is used to indicate that both $\alpha_1^{(j)}$ and $\alpha_1^{(j)}$ are integers with magnitude less than N.

Suppose that $\hat{L}$ is the corresponding spectral projection of L (which, in this case, is real). The matrix $\hat{L}$ can be decomposed into a sum of a diagonal (corresponding to

the second order term in L) and a nondiagonal (corresponding to first order term in L) term, i.e.,

$$\hat{L} = A + D.$$

In this problem the eigenvalues are real and we are trying to find the largest nonzero eigenvalue instead of the eigenvalue with largest magnitude. We must first transform $\hat{L}$ so that the largest eigenvalues of $\hat{L}$ correspond to the magnitude dominant eigenvalues of the transformed matrix. As we mentioned in section 3.2 this can be accomplished using the matrix obtained from a discrete-in-time approximation of the ODE

$$\frac{d}{dt}y = \hat{L}y,$$

i.e., by exponentiating the matrix $t\hat{L}$ for very large t. For example, for sufficiently small $\varepsilon > 0$, the eigenvalue, μ, of largest magnitude of the matrix

$$K = e^{\frac{1}{2}\varepsilon D}(I + \varepsilon A)e^{\frac{1}{2}\varepsilon D} \tag{51}$$

is, to within an error of order ε^2, $1+\varepsilon\lambda_*$, where λ_* is the eigenvalue of $\hat{L}$ of largest real part (in our case the eigenvalues are real and nonpositive). We will apply our iteration schemes to K. By fixing $v_t(\vec{0}) = 0$ we can guarantee that the approximate solutions all have vanishing integral over $[-1, 1)^{2\ell}$, ensuring that the iteration converges to an approximation of the desired eigenvector/value pair (instead of to $v(\vec{0}) = 1$, $v(\vec{\alpha}) = 0$ if $\vec{\alpha} \neq \vec{0}$).

Remark 8. In this problem, rather than estimating the dominant eigenvector of K, our goal is estimate the second largest (in magnitude) eigenvalue of K. Given that we know the largest eigenvalue of K is 1 with an eigenvector that has value 1 in the component corresponding to $\vec{\alpha} = \vec{0}$ and zeros in all other components, we can therefore exactly orthogonalize the iterates V_t^m with respect to the dominant eigenvalue at each iteration (by fixing $V_t^m(\vec{0}) = 0$). We may view this as using FRI in conjunction with a simple case of orthogonal iteration.

We compare the FRI and TbS approaches with $N = 51$ for the four- and five-particle systems ($\ell = 4, 5$). The corresponding total count of real numbers needed to represent the solution in the five-particle case is $101^{10} \approx 10^{20}$, so only the $\mathcal{O}(1)$ scheme is reasonable. For ε we choose a value of 10^{-3}. Our potential V is chosen so that the resulting matrix $\hat{L}$ (and therefore also K) is sparse and its entries are computed by hand. For a more complicated V, the entries of $\hat{L}$ might have to be computed numerically on the fly or might not be computable at all. Our ability to efficiently compute the entries of $\hat{L}$ will be strongly affected by the choice of basis. For example, if we use a finite difference approximation of L, then the entries of $\hat{L}$ can be computed easily. On the other hand, if the solution is reasonably smooth, the finite difference approximation will converge much more slowly than an approximation (like the spectral approximation) that more directly incorporates properties of the solution (regularity in this case).

Figure 5 plots the trajectory averages over 10^5 iterations for Λ_t^m in the $\ell = 4$ case generated by the FRI method (iteration (17) using Algorithm 1) along with corresponding trajectories of Λ_t^m as generated by the TbS approach (iteration (17) using TbS). We present results for both methods with $m = 1$, 2, 3, and 4×10^4. Observe that the results from the FRI method appear to have converged on one another, while the results generated by the TbS approach show no signs of convergence. The best

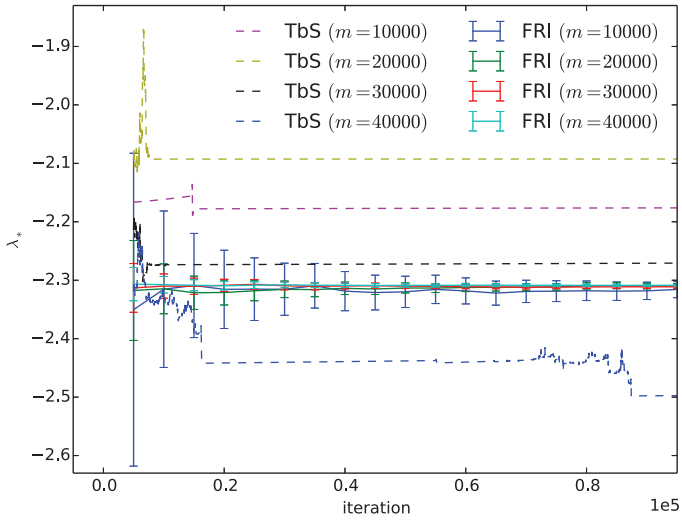

Fig. 5 *Trajectory averages of the approximation, Λ_t^m, of the largest negative eigenvalue of a backwards Kolmogorov operator for a four two-dimensional particle (eight-dimensional) system, with 95% confidence intervals for the FRI method with $m = 1, 2, 3$, and 4×10^4. The operator is discretized using a Fourier basis with 101 modes per dimension for a total of more than 10^{16} basis elements (half that after taking advantage of the fact that the desired eigenvector is real). The step-size parameter ε is set to 10^{-3}. Also on this graph are shown trajectories of Λ_t^m for the TbS method for the same values of m.*

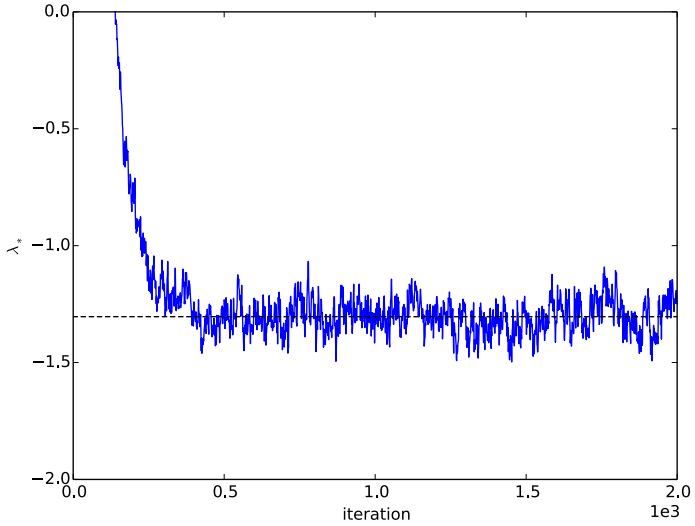

Fig. 6 *Trajectory of the approximation, Λ_t^m (solid line), of the largest negative eigenvalue of a backwards Kolmogorov operator for the five-particle system as computed by the FRI method with $m = 10^6$ over 2×10^3 iterations. The total dimension of the discretized system is more than 10^{20}. The average value of Λ_t^m (ignoring the first 500 iterations) is -1.3 and is shown by a horizontal dotted line.*

(highest m) estimate of the eigenvalue generated by FRI is -2.31 and the best estimate generated by TbS is -2.49. Figure 6 plots the trajectory of Λ_t^m in the five particle ($\ell = 5$) case as generated by the FRI method with $m = 10^6$ along with its trajectory average (neglecting the first 500 iterations) of about -1.3. Note that Λ_t^m appears to reach its statistical equilibrium rapidly relative to the 2×10^3 total iterations. Again, the rapid equilibration suggests that statistical error could be removed by averaging over many shorter trajectories evolved in parallel.

6.3. A PDE Steady State Problem. The adjoint L^* of the operator defined in (49) with respect to the standard inner product is called the Fokker–Planck operator. The operator determines the evolution of the density of the process $X(t)$ defined in (50) in the sense that if μ is that density, then

$$\partial_t \mu = L^* \mu.$$

An element in the kernel of L^* (a steady state solution of the Fokker–Planck equation) is a density left invariant by $X(t)$.

For the choice of b and σ given in the previous subsection, the steady state solutions are easily seen to be constant multiples of the function

$$\mu_*(\vec{x}) = \frac{e^{-V(\vec{x})}}{\int e^{-V(\vec{x})}}.$$

In most applications, the goal is to compute averages of observables with respect to μ_*. For example, one might hope to find (up to an additive constant) the effective potential (or free energy) experienced by particle 1,

$$\mathcal{F}_1(x^{(1)}) = -\log \int \mu_*(\vec{x})\, dx^{(2)} \cdots dx^{(\ell)}.$$

For that purpose, explicit knowledge of μ_* is of little value since one cannot hope to compute integrals with respect to a function of so many variables (up to 101^8 in our tests). One instead hopes to find a more digestible expression for μ_*. Notice that if a Fourier expansion

$$\mu_*(\vec{x}) = \sum_{\vec{\alpha} \in \mathbb{Z}_N^{2\ell}} v(\vec{j})\, e^{i\pi \vec{\alpha}^{\mathrm{H}} \vec{x}}$$

was available, then we could compute

$$\mathcal{F}_1(x^{(1)}) = -\log \sum_{\alpha^{(1)} \in \mathbb{Z}_N^2} v(\alpha^{(1)}, 0, \ldots, 0)\, e^{i\pi \alpha^{(1)T} x^{(1)}}.$$

As in the previous section[11] we discretize the Fokker–Planck operator in a Fourier basis resulting in a finite-dimensional linear root finding problem

$$(K - I)\, v_* = 0,$$

where K is now defined just as in (51) but with $A = \hat{L}^{\mathrm{H}} - D$. We choose to normalize the solution so that $v(\vec{0}) = 1$, which then results in a linear system

$$\left(\bar{K} - I\right) \bar{v}_* = r,$$

[11] Note that the matrix obtained by L^2-projection of the adjoint of a differential operator with real coefficients is the conjugate transpose of the matrix obtained by L^2-projection of the differential operator.

where $r(\vec{\alpha}) = -K_{\vec{\alpha}\vec{0}}$, $\bar{K}$ has the row and column corresponding to the index $\vec{0}$ removed, and $\bar{v}_*$ has the component corresponding to index $\vec{0}$ removed.

Remark 9. Note that the linear system $(\bar{K} - I)\bar{v}_* = r$ is solved for v_* here using FRI in conjunction with Jacobi iteration. With the normalization $v_t(\vec{0}) = 1$, this is equivalent to using the power iteration to find the eigenvector corresponding to the largest eigenvalue of K (which is 1). Recalling that here $K \approx I + \varepsilon \hat{L}^{\mathrm{H}}$, observe that we are (when ε is small) approximately computing $\lim_{t\to\infty} \exp(t\hat{L}^{\mathrm{H}})v_0$ which, since the largest eigenvalue of $\hat{L}^{\mathrm{H}}$ is 0, is the desired eigenvector. We repeat that though we know the dominant eigenvalue of K and we have a formula for μ_*, the dominant eigenvector of L^*, our goal is to compute projections of μ_* that cannot be computed by deterministic means.

In Figure 7 we present approximations of the function $\mathcal{F}_1$ generated by the $\mathcal{O}(1)$ scheme

$$\begin{aligned} V_{t+1}^m &= \Phi_t^m\left(KV_t^m\right), \\ F_{t+1}^m(\alpha^{(1)}) &= \left(KV_t^m\right)(\alpha^{(1)}, 0, \ldots, 0), \\ \overline{F}_{t+1}^m(\alpha^{(1)}) &= (1-\varepsilon_t)\overline{F}_t(\alpha^{(1)}) + \varepsilon_t F_{t+1}^m(\alpha^{(1)}) \end{aligned}$$

for all $\alpha^{(1)} \in \mathbb{Z}_N^2$ with $\varepsilon_t = (t+1)^{-1}$, where the independent mappings Φ_t^m are generated according to Algorithm 1. The single particle free energy[12] as generated by the FRI approach is plotted for $\ell = 2$, 3, 4, and 5 with $m = 10$, 200, 10^4, and 10^6, respectively. In the two-, three-, and four-particle simulations we use 10^5 iterations. Again we choose $N = 51$ and $\varepsilon = 10^{-3}$. The high cost per iteration in the five-particle case restricts our simulations to 2×10^3 iterations. In the four-particle case, for which we have validated the FRI solution by simulations with higher values of m ($m = 4 \times 10^4$), the free energy profile produced by the TbS approach differs from the FRI result by as much as 100%. We take slight advantage of the particle exchange symmetry and, at each iteration, replace $\left(KV_t^m\right)(\alpha^{(1)}, 0, \ldots, 0)$ in the above equation for F_{t+1}^m by the average of all ℓ components of the form $\left(KV_t^m\right)(0, \ldots, \alpha^{(k)}, 0, \ldots, 0)$. Note that in the expansion of μ_*, we know that $v(\vec{\alpha})$ is unchanged when we swap the indices $\alpha^{(j)}$ and $\alpha^{(k)}$ corresponding to any two particles. This fact could be leveraged to greatly reduce the number of basis functions required to accurately represent the solution. We have not exploited this possibility.

Though it is not accurate, the TbS scheme is substantially more stable on this problem. We assume that the relative stability of the TbS scheme is a manifestation of the fact that TbS is not actually representing the high wave number modes that are responsible for stability constraints. Nonetheless, simulating higher-dimensional systems would require modifications in our approach. In particular, it might be necessary to identify a small number of components of the solution that should always be resolved (never set to zero). For this problem, for example, one might choose to resolve some number of basis functions, each dependent only on the position of a single particle.

7. Discussion. We have introduced a family of FRI schemes for eigenproblems, linear systems, and matrix exponentiation. Traditional iterative methods for numerical linear algebra were created in part to deal with instances where the coefficient

[12]Note that we only approximate $\mathcal{F}_1$ up to the additive constant $-\log \int e^{-V(\vec{x})}$. In fact, the free energy is typically only defined up to that constant because it is not uniquely specified (one can add a constant to V without changing μ_*).

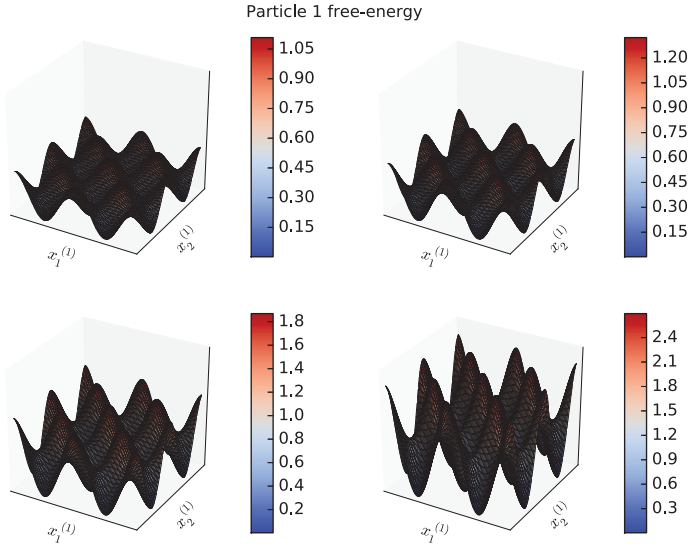

Fig. 7 *Free energy landscape experienced by a single particle for the (clockwise from top left) two-, three-, four-, and five- two-dimensional particle systems. The surfaces were generated using the FRI method with* $m = 10, 200, 10^4$, *and* 10^6, *respectively, and* $\varepsilon = 10^{-3}$. *The two-, three-, and four-particle simulations were run for* 10^5 *iterations. The five-particle simulation was substantially more expensive per iteration and was run for only* 2×10^3 *iterations. The number of Fourier modes used to represent the solution in all simulations is* 101 *per dimension for a total of more than* 10^8, 10^{12}, 10^{16}, *and* 10^{20} *basis functions (half that after taking advantage of the fact that the solution is real). As expected, the free energy basins deepen as the number of particles grows. In the four-particle case (for which we have high confidence in the estimate produced by FRI), the error from the TbS approach (the results of which we do not plot) is roughly* 100% *at peaks of the free energy landscape.*

matrix A (of size $\mathcal{O}(n^2)$) is too big to store but where the operation $x \mapsto Ax$ can nonetheless be carried out. The iterative methods in this article are intended for instances in which the ultimate goal is to compute $f \cdot x$ for some predetermined vector f, but the cost of assembling the product Ax ($\mathcal{O}(n^2)$) is too high, or even for cases in which the solution vector x (of size $\mathcal{O}(n)$) is too big to store. We provide basic theoretical results justifying the approach and illustrating, in particular, that the cost of the schemes can be independent of dimension for some problems. Generally we expect sublinear scaling with dimension of both cost and storage requirements as observed in our numerical experiments. The identification of general conditions guaranteeing sublinear scaling for FRI schemes is not addressed in this article, but seems a very interesting direction for future research.

A completely deterministic approach to iterative problems related to the methods proposed in this article is the simple TbS in which, at each iteration, the smallest entries in the approximation are set to zero. An adaptive version of the TbS approach has recently been advocated for a wide range of applications (see [58, 49, 50]). Like TbS our randomized schemes also rely on the enforcement of sparsity and also tend to set small entries in the approximate solution to zero. While the TbS approach can be effective on some problems with sparse solutions, their error in general will be strongly dependent on system size and we find that it performs very poorly on our test problems relative to FRI.

The core concept behind the FRI schemes introduced in this article is the notion that, by randomly setting entries in vectors to zero, while maintaining a statistical consistency property, we can dramatically reduce the cost and storage of standard iterative schemes. One can view our FRI schemes as an attempt to blur the line separating MCMC, which is effective in extremely high-dimensional settings but is limited to a relatively narrow class of problems and often does not allow the user to take full advantage of known properties of the solution (e.g., smoothness or symmetry properties as in [7]), and traditional deterministic schemes, which are effective on a very general set of relatively low-dimensional problems. As for MCMC approaches, if one settles for computing low-dimensional projections of the full solution, then not every element of the state space need be visited and effective FRI schemes with per iteration cost and storage requirements independent of system size can be derived (as for MCMC the validity of this statement depends on the particular sequence of problems considered). Also, as for MCMC we expect that when deterministic alternatives are available, they will outperform our randomized schemes. For matrices of the size considered in all of our numerical tests, deterministic alternatives are *not* available.

Experience with DMC in the context of QMC simulations suggests that our randomized schemes will be most useful if applied after considerable effort has been expended on finding changes of variables that either make the desired solution as sparse as possible (reducing both bias and variance) or reduce bias by some other means. In many cases this will mean applying our randomized schemes only after one has obtained an estimate of the solution by some deterministic method applied to a reduced-dimensional version of the target problem.

Acknowledgments. JQW would like to thank Eric Cances, Tony Lelievre, and Matthias Rousset for their hospitality and helpful discussions during a visit to ENPC that coincided with the early stages of this work. Both authors would like to thank Mihai Anitescu, Alexandre Chorin, Petros Drineas, Risi Kondor, Jianfeng Lu, Omiros Papaspiliopoulos, Panos Stinis, and the anonymous referees, who all made comments that strongly affected this article's structure and content.

REFERENCES

[1] V. Alexandrov and S. Lakka, *Comparison of three Monte Carlo methods for matrix inversion*, in Euro-Par'96 Parallel Processing, L. Bougé, P. Fraigniaud, A. Mignotte, and Y. Robert, eds., Lecture Notes in Comput. Sci. 1124, Springer, Berlin, Heidelberg, 1996, pp. 72–80, https://doi.org/10.1007/BFb0024687. (Cited on pp. 548, 549)

[2] J. Anderson, *A random-walk simulation of the Schrödinger equation: H_3^+*, J. Chem. Phys., 63 (1975), pp. 1499–1503. (Cited on p. 549)

[3] R. Baer, D. Neuhauser, and E. Rabani, *Self-averaging stochastic Kohn-Sham density-functional theory*, Phys. Rev. Lett., 111 (2013), 106402, https://doi.org/10.1103/PhysRevLett.111.106402. (Cited on p. 549)

[4] G. H. Booth and A. Alavi, *Approaching chemical accuracy using full configuration interaction quantum Monte Carlo: A study of ionization potentials*, J. Chem. Phys., 132 (2010), 174104. (Cited on p. 549)

[5] G. H. Booth, D. Cleland, A. J. W. Thom, and A. Alavi, *Breaking the carbon dimer: The challenges of multiple bond dissociation with full configuration interaction quantum Monte Carlo methods*, J. Chem. Phys., 135 (2011), 084104. (Cited on p. 549)

[6] G. H. Booth, A. Grüneis, G. Kresse, and A. Alavi, *Towards an exact description of electronic wavefunctions in real solids*, Nature, 493 (2013), pp. 365–370, https://doi.org/10.1038/nature11770. (Cited on p. 549)

[7] G. H. Booth, A. J. W. Thom, and A. Alavi, *Fermion Monte Carlo without fixed nodes: A game of life, death, and annihilation in Slater determinant space*, J. Chem. Phys., 131 (2009), 054106, https://doi.org/10.1063/1.3193710. (Cited on pp. 549, 555, 556, 584)

[8] L. Bottou, *Stochastic learning*, in Advanced Lectures on Machine Learning, O. Bousquet and U. von Luxburg, eds., Lecture Notes in Artificial Intelligence, 3176, Springer Verlag, Berlin, 2004, pp. 146–168. (Cited on p. 549)

[9] N. Bou-Rabee and E. Vanden-Eijnden, *Continuous-Time Random Walks for the Numerical Solution of Stochastic Differential Equations*, preprint, https://arxiv.org/abs/1502.05034, 2015. (Cited on p. 554)

[10] D. Ceperley and B. Alder, *Ground state of electron gas by a stochastic method*, Phys. Rev. Lett., 45 (1980), pp. 566–569. (Cited on p. 549)

[11] A. Chorin, *Random choice solution of hyperbolic systems*, J. Comput. Phys., 22 (1976), pp. 517–536. (Cited on p. 549)

[12] D. Cleland, G. H. Booth, and A. Alavi, *Survival of the fittest: Accelerating convergence in full configuration-interaction quantum Monte Carlo*, J. Chem. Phys., 132 (2010), 041103. (Cited on p. 549)

[13] D. Cleland, G. H. Booth, and A. Alavi, *A study of electron affinities using the initiator approach to full configuration interaction quantum Monte Carlo*, J. Chem. Phys., 134 (2011), 024112. (Cited on p. 549)

[14] E. S. Coakley, V. Rokhlin, and M. Tygert, *A fast randomized algorithm for orthogonal projection*, SIAM J. Sci. Comput., 33 (2011), pp. 849–868, https://doi.org/10.1137/090779656. (Cited on p. 548)

[15] P. Del Moral, *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications*, Probab. Appl. (N.Y.), Springer-Verlag, New York, 2004, https://doi.org/10.1007/978-1-4684-9393-1. (Cited on pp. 548, 561, 562, 566)

[16] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997, https://doi.org/10.1137/1.9781611971446. (Cited on p. 558)

[17] I. Dimov, T. Dimov, and T. Gurov, *A new iterative Monte Carlo approach for inverse matrix problem*, J. Comput. Appl. Math., 92 (1998), pp. 15–35, https://doi.org/10.1016/S0377-0427(98)00043-0. (Cited on pp. 548, 556)

[18] I. Dimov, A. Karaivanova, and P. Yordanova, *Monte Carlo algorithms for calculating eigenvalues*, in Monte Carlo and Quasi-Monte Carlo Methods 1996, H. Niederreiter, P. Hellekalek, G. Larcher, and P. Zinterhof, eds., Lecture Notes in Statist. 127, Springer, New York, 1998, pp. 205–220, https://doi.org/10.1007/978-1-4612-1690-2_12. (Cited on pp. 548, 556)

[19] A. Doucet, N. de Freitas, and N. Gordon, eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, 2005. (Cited on pp. 550, 556, 573)

[20] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices* I: *Approximating matrix multiplication*, SIAM J. Comput., 36 (2006), pp. 132–157, https://doi.org/10.1137/S0097539704442684. (Cited on pp. 548, 556, 561)

[21] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices* II: *Computing a low-rank approximation to a matrix*, SIAM J. Comput., 36 (2006), pp. 158–183, https://doi.org/10.1137/S0097539704442696. (Cited on pp. 548, 556)

[22] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices* III: *Computing a compressed approximate matrix decomposition*, SIAM J. Comput., 36 (2006), pp. 184–206, https://doi.org/10.1137/S0097539704442702. (Cited on pp. 548, 556)

[23] W. E, B. Engquist, X. Li, W. Ren, and E. Vanden-Eijnden, *The heterogeneous multiscale methods: A review*, Commun. Comput. Phys., 2 (2007), pp. 367–450. (Cited on p. 561)

[24] S. Eriksson-Bique, M. Solbrig, M. Stefanelli, S. Warkentin, R. Abbey, and I. C. F. Ipsen, *Importance sampling for a Monte Carlo matrix multiplication algorithm, with application to information retrieval*, SIAM J. Sci. Comput., 33 (2011), pp. 1689–1706, https://doi.org/10.1137/10080659X. (Cited on p. 548)

[25] G. E. Forsythe and R. A. Leibler, *Matrix inversion by a Monte Carlo method*, Math. Tables and Other Aids to Computation, 4 (1950), pp. 127–129. (Cited on p. 548)

[26] W. Foulkes, L. Mitas, R. Needs, and G. Rajagopal, *Quantum Monte Carlo simulations of solids*, Rev. Modern Phys., 73 (2001), pp. 33–79. (Cited on pp. 549, 550, 551, 566)

[27] S. Friedland and L.-H. Lim, *Nuclear norm of higher-order tensors*, Math. Comp., to appear, https://doi.org/10.109D/mcom/3239. (Cited on pp. 562, 563)

[28] A. Frieze, R. Kannan, and S. Vempala, *Fast Monte-Carlo algorithms for finding low-rank approximations*, J. ACM, 51 (2004), pp. 1025–1041, https://doi.org/10.1145/1039488.1039494 (Cited on p. 548)

[29] N. H. Fuchs, *Approximate solutions for large transfer matrix problems*, J. Comput. Phys., 83 (1989), pp. 201–211, https://doi.org/10.1016/0021-9991(89)90228-3. (Cited on pp. 573, 574)

[30] J. Goodman and N. Madras, *Random-walk interpretations of classical iteration methods*, Linear Algebra Appl., 216 (1995), pp. 61–79. (Cited on p. 548)
[31] N. Gordon, D. Salmond, and A. Smith, *Novel approach to nonlinear non-Gaussian Bayesian state estimation*, IEE Proc. F, 51 (1993), pp. 107–113. (Cited on p. 550)
[32] R. Grimm and R. Storer, *Monte-Carlo solution of Schrödinger's equation*, J. Comput. Phys., 7 (1971), pp. 134–156. (Cited on p. 549)
[33] M. Hairer and J. Weare, *Improved diffusion Monte Carlo*, Commun. Pure Appl. Math., 67 (2014), pp. 1995–2021, https://doi.org/10.1002/cpa.21526. (Cited on pp. 549, 569)
[34] J. H. Halton, *Sequential Monte Carlo*, Proc. Cambridge Philos. Soc., 58 (1962), pp. 57–78. (Cited on pp. 548, 549)
[35] J. H. Halton, *A retrospective and prospective survey of the Monte Carlo method*, SIAM Rev., 12 (1970), pp. 1–63, https://doi.org/10.1137/1012001. (Cited on p. 548)
[36] J. H. Halton, *Sequential Monte Carlo techniques for the solution of linear systems*, J. Sci. Comput., 9 (1994), pp. 213–257, https://doi.org/10.1007/BF01578388. (Cited on p. 548)
[37] J. Hammersley and K. Morton, *Poor man's Monte Carlo*, J. R. Stat. Soc. B Stat. Methodol., 16 (1954), pp. 23–38. (Cited on p. 549)
[38] J. M. Hammersley, *Monte Carlo methods for solving multivariable problems*, Ann. New York Acad. Sci., 86 (1960), pp. 844–874, https://doi.org/10.1111/j.1749-6632.1960.tb42846.x. (Cited on pp. 548, 561)
[39] J. M. Hammersley and D. C. Handscomb, *Monte Carlo Methods*, Methuen and Co., London, John Wiley and Sons, New York, 1964. (Cited on p. 548)
[40] M. Kalos, *Monte Carlo calculations of the ground state of three- and four-body nuclei*, Phys. Rev., 128 (1962), pp. 1791–1795. (Cited on p. 549)
[41] G. Kitagawa, *Monte Carlo filter and smoother for non-Gaussian nonlinear state space models*, J. Comput. Graphical Statist., 5 (1996), pp. 1–25. (Cited on p. 550)
[42] J. Kolorenč and L. Mitas, *Applications of quantum Monte Carlo methods in condensed systems*, Rep. Progr. Phys., 74 (2011), pp. 1–28. (Cited on p. 549)
[43] H. Kushner and G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications*, 2nd ed., Appl. Math. (N.Y.) 35, Springer-Verlag, New York, 2003. (Cited on pp. 558, 561, 562)
[44] E. Liberty, F. Woolfe, P.-G. Martinsson, V. Rokhlin, and M. Tygert, *Randomized algorithms for the low-rank approximation of matrices*, Proc. Natl. Acad. Sci. USA, 104 (2007), pp. 20167–20172, https://doi.org/10.1073/pnas.0709640104. (Cited on p. 548)
[45] P. López Ríos, A. Ma, N. D. Drummond, M. D. Towler, and R. J. Needs, *Inhomogeneous backflow transformations in quantum Monte Carlo calculations*, Phys. Rev. E, 74 (2006), 066701, https://doi.org/10.1103/PhysRevE.74.066701. (Cited on p. 549)
[46] P.-G. Martinsson, V. Rokhlin, and M. Tygert, *A randomized algorithm for the decomposition of matrices*, Appl. Comput. Harmon. Anal., 30 (2011), pp. 47–68, https://doi.org/10.1016/j.acha.2010.02.003. (Cited on p. 548)
[47] E. Moulines and F. R. Bach, *Non-asymptotic analysis of stochastic approximation algorithms for machine learning*, in Proceedings of the 24th International Conference on Neural Information Processing Systems, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, eds., Curran Associates, 2011, pp. 451–459, http://papers.nips.cc/paper/4316-non-asymptotic-analysis-of-stochastic-approximation-algorithms-for-machine-learning.pdf. (Cited on p. 561)
[48] M. P. Nightingale and H. W. J. Blöte, *Gap of the linear spin-1 Heisenberg antiferromagnet: A Monte Carlo calculation*, Phys. Rev. B, 33 (1986), pp. 659–661, https://doi.org/10.1103/PhysRevB.33.659. (Cited on pp. 550, 554)
[49] V. Ozolins, R. Lai, R. Caflisch, and S. Osher, *Compressed modes for variational problems in mathematics and physics*, Proc. Natl. Acad. Sci. USA, 110 (2013), pp. 18368–18373, https://doi.org/10.1073/pnas.1318679110. (Cited on pp. 573, 583)
[50] V. Ozolins, R. Lai, R. Caflisch, and S. Osher, *Compressed plane waves yield a compactly supported multiresolution basis for the Laplace operator*, Proc. Natl. Acad. Sci. USA, 111 (2014), pp. 1691–1696, https://doi.org/10.1073/pnas.1323260111. (Cited on pp. 573, 583)
[51] B. Parlett and W.-L. Heng, *The method of minimal representations in 2D Ising model calculations*, J. Comput. Phys., 114 (1994), pp. 257–264. (Cited on p. 575)
[52] G. Pavliotis and A. Stuart, *Multiscale Methods: Averaging and Homogenization*, Texts Appl. Math. 53, Springer, New York, 2008. (Cited on p. 561)
[53] H. Robbins and S. Monro, *A stochastic approximation method*, Ann. Math. Statist., 22 (1951), pp. 400–407. (Cited on p. 558)
[54] V. Rokhlin, A. Szlam, and M. Tygert, *A randomized algorithm for principal component analysis*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1100–1124, https://doi.org/10.1137/080736417. (Cited on p. 548)

[55] V. Rokhlin and M. Tygert, *A fast randomized algorithm for overdetermined linear least-squares regression*, Proc. Natl. Acad. Sci. USA, 105 (2008), pp. 13212–13217, https://doi.org/10.1073/pnas.0804869105. (Cited on p. 548)

[56] M. Rosenbluth and A. Rosenbluth, *Monte Carlo calculation of the average extension of molecular chains*, J. Chem. Phys., 23 (1955), pp. 356–359. (Cited on p. 549)

[57] M. Rousset, *On the control of an interacting particle estimation of Schrödinger ground states*, SIAM J. Math. Anal., 38 (2006), pp. 824–844, https://doi.org/10.1137/050640667. (Cited on pp. 561, 562, 566)

[58] H. Schaeffer, R. Caflisch, C. D. Hauck, and S. Osher, *Sparse dynamics for partial differential equations*, Proc. Natl. Acad. Sci. USA, 110 (2013), pp. 6634–6639, https://doi.org/10.1073/pnas.1302752110. (Cited on pp. 573, 583)

[59] J. Sheperd, G. H. Booth, A. Grüneis, and A. Alavi, *Full configuration interaction perspective on the homogeneous electron gas*, Phys. Rev. B, 85 (2012), 081103. (Cited on pp. 549, 556)

[60] G. W. Stewart, *Matrix Algorithms* II: *Eigensystems*, SIAM, Philadelphia, 2001. (Cited on p. 566)

[61] T. Strohmer and R. Vershynin, *A randomized Kaczmarz algorithm with exponential convergence*, J. Fourier Anal. Appl., 15 (2008), pp. 262–278, https://doi.org/10.1007/s00041-008-9030-4. (Cited on p. 548)

[62] W. R. Wasow, *A note on the inversion of matrices by random walk*, Math. Tables and Other Aids to Computation, 6 (1952), pp. 78–81. (Cited on p. 548)

[63] J. Weare, *A simple example in C++ of FRI applied to computing the dominant eigenvalue of a matrix*, https://doi.org/10.5281/zenodo.31208, 2015. (Cited on pp. 551, 574, 575)

[64] F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert, *A fast randomized algorithm for the approximation of matrices*, Appl. Comput. Harmon. Anal., 25 (2008), pp. 335–366, https://doi.org/10.1016/j.acha.2007.12.002. (Cited on p. 548)