SIAM J. SCI. COMPUT.
Vol. 41, No. 6, pp. A3449–A3475

MULTIWAY MONTE CARLO METHOD FOR LINEAR SYSTEMS[*]

TAO WU[†] AND DAVID F. GLEICH[†]

Abstract. We study a novel variation on the Ulam–von Neumann Monte Carlo method for solving a linear system. This is an old randomized procedure that results from using a random walk to stochastically evaluate terms in the Neumann series. In order to apply this procedure, the variance of the stochastic estimator needs to be bounded. The best known sufficient condition for bounding the variance is that the infinity norm of the matrix in the Neumann series is smaller than one, which greatly limits the usability of this method. We improve this condition by proposing a new stochastic estimator based on a different type of random walk. Our multiway walk and estimator is based on a time-inhomogeneous Markov process that iterates through a sequence of transition matrices built from the original linear system. For our new method, we prove that a necessary and sufficient condition for convergence is that the spectral radius of the elementwise absolute value of the matrix underlying the Neumann series is smaller than one. This is a strictly weaker condition than currently exists. In addition, our new method is often faster than the standard algorithm. Through experiments, we demonstrate the potential for our method to reduce the time needed to solve linear equations by incorporating it into an outer iterative method.

Key words. Markov chain Monte Carlo, linear solver, randomized algorithm

AMS subject classifications. 65C05, 65F50, 65Y20, 68Q25, 68W40

DOI. 10.1137/18M121527X

1. Introduction.

Monte Carlo algorithms for linear systems have a history going back to the dawn of computing, as described in Forsythe and Leibler (1950), and use ideas from von Neumann and Ulam. They are usually explained in terms of the following form of a linear system,

$$\mathbf{x} = \boldsymbol{H}\mathbf{x} + \mathbf{b}.$$

This linear system formulation is an alternative to the canonical form $\boldsymbol{A}\mathbf{x} = \mathbf{b}$ by defining $\boldsymbol{H} = \boldsymbol{I} - \boldsymbol{A}$. (We discuss ideas relating the two formulations in the context of our experiments in section 5.4.) The essence of the idea is to use random walks to construct a random variable X such the expected value $\mathrm{E}[X] = \mathbf{h}^T\mathbf{x}$ for some prescribed vector $\mathbf{h}$. Then the computation simply involves generating random instances of X and taking an average to stochastically estimate $\mathrm{E}[X]$. A standard use of the method would iterate through all choices $\mathbf{h} = \mathbf{e}_i$ for all n possible standard basis vectors to estimate each component of the solution. Alternatives include adjoint methods that directly estimate the entire solution (Halton, 1994). We return to a formal introduction to these ideas in section 2.

The class of Monte Carlo (MC) methods for linear systems is interesting because there are many different trade-offs and features from classical iterative and Krylov methods for solving linear systems. For instance, the methods parallelize exceptionally well because they involve independent samples of a random variable. The computational complexity or the convergence speed is not directly dependent on the size

[*] Submitted to the journal's Methods and Algorithms for Scientific Computing section September 20, 2018; accepted for publication (in revised form) June 11, 2019; published electronically November 5, 2019.
https://doi.org/10.1137/18M121527X
Funding: This work was supported by NSF IIS-1422918, CAREER award CCF-1149756, Center for Science of Information STC, CCF-0939370; DOE award DE-SC0014543; and the DARPA SIMPLEX program.

[†] Department of Computer Science, Purdue University, West Lafayette, IN 47906 (wu577@purdue.edu, dgleich@purdue.edu).

of the matrix and is determined by the variance of X instead. Thus, growing the size of the problem does not necessarily increase the complexity of the method. Also MC methods do not require access to all entries of the matrix $\boldsymbol{H}$. Rather, they need to simulate random walk steps, which usually only involves entries from a single row or column of $\boldsymbol{H}$. These scenarios are appealing for many recent complex and highly distributed computing environments such as cloud computing (Li and Mascagni, 2003) and sensor networks (Slattery, Evans, and Wilson, 2015). Moreover, MC methods have the ability to estimate only a single component or a linear combination of the solution vector when it is unnecessary to compute or store the whole solution (Wang et al., 2008).

One of the key theoretical weaknesses is that the applicability of these method is often limited due to its slow convergence or even its failure to converge (Ji, Mascagni, and Li, 2013). This has limited applicability of the method to scenarios where high accuracy is not required, such as PageRank computations (Avrachenkov et al., 2007) and graph partitioning (Srinivasan and Mascagni, 2002), or scenarios akin to preconditioning (Halton, 1994; Evans et al., 2014; Slattery, 2013) where MC provides an inner black box inside each iteration to accelerate the overall iterative procedure. This is often advantageous because the MC part is highly parallel (LeBeau, 1999; Dietrich and Boyd, 1996), making it suitable for methods which need to scale to extreme numbers of parallel processors.

1.1. Our contribution.

In this paper we contribute new theory and methods that improve the applicability of the MC method in terms of its convergence and address a class of cases where it previously would not converge. First, note that the central limit theorem underlying the relationship between the stochastic average $(1/N)(X_1 + X_2 + \cdots + X_N)$ and the expectation of $\mathrm{E}[X]$ (used in the analysis of MC methods above), requires that $\mathrm{Var}[X] < \infty$ for the stochastic average to converge. Existing research (Dimov, Maire, and Sellier, 2015; Srinivasan, 2010; Wang et al., 2008) assumes $\|\boldsymbol{H}\| < 1$ (for the infinity norm $\|\boldsymbol{H}\| = \max_i \sum_j |H_{i,j}|$), which suffices to show $\mathrm{Var}[X] < \infty$. The norm bound on $\boldsymbol{H}$, however, is a stronger condition than $\rho(\boldsymbol{H}) < 1$, which characterizes when the Neumann series underlying the method converges. Although it is possible to have $\mathrm{Var}[X] < \infty$ when $\|\boldsymbol{H}\| \geq 1$, there is no easy way to check.

To tackle this problem, in section 3 we propose a new multiway random walk that uses a time-inhomogeneous random walk built from the original linear system. This is a generalization of the standard MC method and its associated random walk. At each step of the random walk, the transition matrix is constructed in a way akin to the *Monte Carlo almost optimal* framework (Dimov, Maire, and Sellier, 2015; Ji, Mascagni, and Li, 2013). We prove an explicit form of the variance of this new type of MC method in section 3.3, and derive results on the convergence of this method. In section 4 we propose an efficient algorithm to construct the multiple transition matrices, which we assemble into a transition hypermatrix, and prove that it minimizes the upper bound of the norm of a matrix which directly determines the variance.

We further prove that, under this type of random walk, the new method always converges when $\rho(\boldsymbol{H}^+) < 1$, where $\boldsymbol{H}^+$ is the nonnegative matrix defined as $H^+_{ij} = |H_{ij}|$. This is a strictly weaker condition than $\|\boldsymbol{H}\| < 1$ and we can see that moving from the condition $\|\boldsymbol{H}\| < 1$ to $\rho(\boldsymbol{H}^+) < 1$ makes more problems solvable. An illustration of this advance is shown in Figure 1. Second, our new method can converge faster than the standard MC algorithm. We show that the standard algorithm is a non-optimal special case in our multiway random walk settings, whereas

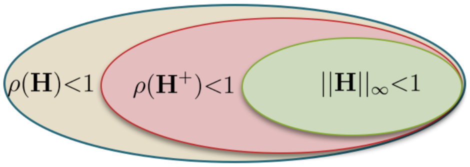

FIG. 1. *An illustration on the convergence regions for different methods: $\rho(\boldsymbol{H}) < 1$ is the requirement for the Neumann series underlying all MC methods to converge; $\rho(\boldsymbol{H}^+) < 1$ is the requirement for our multiway walk-based method to converge; $\|\boldsymbol{H}\|_\infty$ is the sufficient condition for the standard MC method to converge.*

our multiway random walks choose the optimal transition hypermatrices, which tend to minimize the variance.

In section 5, we conduct numerical experiments on both synthetic and real world matrices. We demonstrate the effectiveness of our new method by comparing the variances computed according to our theorem and by comparing the stochastic errors from the solvers as well. One downside to our approach is that is cannot be implemented in a purely local fashion akin to the standard MC method as it requires global work to build the multiway walk. Finally, we study an empirical performance model for a specific linear system and a hybrid Richardson-based solver (akin to those from (Halton, 1994; Evans et al., 2014; Slattery, 2013)), which shows that—as expected—our method can take advantage of more processors than traditional methods.

1.2. Notation. Throughout this paper, we use bold, uppercase letters such as $\boldsymbol{A}$ to denote matrices, and bold, lowercase letters such as $\mathbf{x}$ to denote vectors. We use letters with subscripts of indices to denote elements x_i of a vector, and similarly $A_{i,j}$ to denote elements of a matrix. Random variables are uppercase, nonbold and may also occur in sequences such as in $X_1, \ldots, X_N$. The vector $\mathbf{e}$ is the vector of all ones of appropriate size.

1.3. Related work. We briefly survey some of the recent work in the field and note that our techniques may be combined with many of these ideas for further improvements—although these are beyond the scope of our article. For instance, similar MC techniques can also handle eigenvalue problems (Dimov et al., 2008). The literature around MC for linear systems largely falls into three classes: direct methods, hybrid methods, and applied studies including applications.

Direct methods study improving the techniques of the MC simulation and random walks themselves. For instance Wasow (1952) constructs a slightly different estimator from Forsythe and Leibler (1950), which has smaller variance under certain conditions; Srinivasan and Aggarwal (2003) reformulates the MC method to evaluate intricate iterative schemes related to general matrix splittings. For estimating the full inverse, the results of Dimov, Dimov, and Gurov (1998) study the impact of column or row specific formulations of each system. There are also other classes of approaches that depart from the von Neumann–Ulam methods entirely, such as Sabelfeld and Mozartova (2009), who proposed the sparsified randomization MC method that uses samples of a random matrix estimator $\mathcal{X}$ where $\mathrm{E}[\mathcal{X}] = \boldsymbol{H}$ in an iterative scheme instead of the full matrix. These have different trade-offs.

Hybrid methods (Halton, 1994; Evans et al., 2014; Alexandrov et al., 2005) on the other hand, use a direct MC method as a blackbox combined with more standard iterative techniques that are amenable to stochastic variation in the iterates. The sequential Monte Carlo (SMC) method (Halton, 1994) is one such method, which approximates the solution of $\mathbf{r} = \boldsymbol{H}\mathbf{r} + \mathbf{b}$ at each iteration, where $\mathbf{r}$ is the residual after the previous approximate MC solves. Evans et al. (2014) and Slattery (2013) introduced the Monte Carlo synthetic acceleration method, where one step of Richardson iteration is added to each iteration. This is a competitive method compared to preconditioned conjugate gradients and GMRES on some systems.

Finally, there are also a variety of studies regarding other properties of the MC method, for instance, parallel implementation (Dimov, Alexandrov, and Karaivanova, 2001; Slattery, 2013; Alexandrov et al., 2005), real world application (Wang et al., 2008; Avrachenkov et al., 2007), convergence analysis (Ji, Mascagni, and Li, 2013; Benzi et al., 2017), and spectral analysis (Slattery, Evans, and Wilson, 2015).

Note that, although there are some similarities and relationships with the study of randomized methods, e.g., Drineas, Kannan, and Mahoney (2006a,b,c); Avron, Maymounkov, and Toledo (2010); Halko, Martinsson, and Tropp (2011), the analysis and types of techniques are rather different.

2. The standard Monte Carlo method.

Consider the linear system of the following form:

$$\mathbf{x} = \boldsymbol{H}\mathbf{x} + \mathbf{b}, \tag{1}$$

where $\boldsymbol{H} \in \mathbb{R}^{n\times n}$ and $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$. Assuming the spectral radius $\rho(\boldsymbol{H}) < 1$, then the following Neumann series converges to the solution

$$\mathbf{x} = \sum_{\ell=0}^{\infty} \boldsymbol{H}^{\ell}\mathbf{b}.$$

The MC method of estimating the solution vector is to simulate multiple Markov chains where each chain is an unbiased estimator of the above Neumann series. This can either be done with a *forward* technique to estimate any inner product with the solution or an *adjoint* technique to estimate the entire solution.

2.1. Forward method.

Consider the general goal of evaluating the functional of the solution

$$\mathbf{h}^T\mathbf{x} \quad \text{or as we now will use} \quad \langle \mathbf{h}, \mathbf{x}\rangle = \sum_{i=1}^{n} h_i x_i. \tag{2}$$

We could then use this primitive to compute the solution by evaluating the functional for each standard basis vector to get each single component of the solution. This is the idea of the forward method.

To evaluate the functional, the MC method creates the following Markov random walk $X_0, X_1, \dots,$ and the associated walk related weight W. The state space of the random walk are the indices of the linear system: $S = \{1, 2, \dots, n\}$. Both the initial probability distribution of the walk and the transition probabilities have a large degree of flexibility and hence are left underspecified. The method applies when the starting probability distribution for the walk involves $\mathbf{h}$ as follows:

$$\Pr(X_0 = i) = p_i \quad \text{such that} \quad h_i \neq 0 \Rightarrow p_i \neq 0$$

and the transition probabilities

$$\Pr(X_{\ell+1} = j \mid X_\ell = i) = P_{i,j} \quad \text{such that} \quad H_{i,j} \neq 0 \Rightarrow P_{i,j} \neq 0.$$

Let ν be a realization of the random walk:

$$k_0 \to k_1 \to k_2 \to \cdots \to k_\ell \to \cdots.$$

Let the walk related weight W and the random variable X be calculated as follows:

$$\begin{aligned} W_\ell(\nu) &= \frac{h_{k_0} H_{k_0,k_1} H_{k_1,k_2} \cdots H_{k_{\ell-1},k_\ell}}{p_{k_0} P_{k_0,k_1} P_{k_1,k_2} \cdots P_{k_{\ell-1},k_\ell}} \quad \text{for } \ell = 0, 1, 2, \ldots, \\ X(\nu) &= \sum_{\ell=0}^{\infty} W_\ell b_{k_\ell}. \end{aligned}$$

Then it can be shown (for instance Dimov, Dimov, and Gurov (1998)) that $\mathrm{E}[X] = \langle \mathbf{h}, \mathbf{x} \rangle$ and more specifically $\mathrm{E}[W_\ell f_{k_\ell}] = \langle \mathbf{h}, \boldsymbol{H}^\ell \mathbf{f} \rangle$. Since the random variable is an unbiased estimator of the solution, the MC method runs simulations of this random walk and uses the empirical mean $\bar{X}$ to approximate $\mathrm{E}[X]$.

If the desired output is the functional $\langle \mathbf{h}, \mathbf{x} \rangle$ itself, this method is a good idea. However, as previously mentioned, in order to estimate the entire solution vector, the method has to be applied multiple times to get a single entry of the solution each time. This limitation can be better handled by the following adjoint method.

2.2. Adjoint method. Note that, in the forward method, the right-hand side $\mathbf{b}$ only arises in the final path-dependent random variable $X(\nu)$. And so, we could easily change $\mathbf{b}$ to estimate the solution of slightly different linear systems. However, we would need to control the start of the walks to vary the vector $\mathbf{h}$, which is what we need to vary to get multiple components of the solution $\mathbf{x}$. The idea with the adjoint method is that we'd like to (and can!) invert the role of these two features. We can create a method that *starts* depending on $\mathbf{b}$ and estimates solution entries with path-dependent weights. To do so, consider the following linear system

$$\mathbf{y} = \boldsymbol{H}^T \mathbf{y} + \mathbf{d}.$$

If $\mathbf{x}$ solves (1), then we have the following inner product equivalence:

$$\langle \mathbf{x}, \mathbf{d} \rangle = \langle (\boldsymbol{I} - \boldsymbol{H})^T \mathbf{y}, \mathbf{x} \rangle = \langle \mathbf{y}, (\boldsymbol{I} - \boldsymbol{H}) \mathbf{x} \rangle \quad \Longrightarrow \quad \langle \mathbf{x}, \mathbf{d} \rangle = \langle \mathbf{y}, \mathbf{b} \rangle.$$

We can apply the technique from the forward method to compute the functional $\langle \mathbf{y}, \mathbf{b} \rangle$. Notice this requires that walks start depending on $\mathbf{b}$. Then we can vary $\mathbf{d}$ though $\mathbf{e}_1, \ldots, \mathbf{e}_n$ depending on which state the random walk lands at each step. This way, by a single random walk, each solution element $x_1, \ldots, x_n$ can be updated by different random walk steps.

Putting the pieces together, now to estimate the whole original solution vector $\mathbf{x}$ from the linear system (1), the adjoint method will construct the following Markov chain with the initial probability

$$\Pr(X_0 = i) = p_i \quad \text{such that} \quad b_i \neq 0 \Rightarrow p_i \neq 0$$

and the transition probability

$$\Pr(X_{\ell+1} = j \mid X_\ell = i) = P_{i,j} \quad \text{such that} \quad H^T_{i,j} \neq 0 \Rightarrow P_{i,j} \neq 0.$$

And similarly, given a realization ν, we define the weight for each step as

$$W_\ell = \frac{h_{k_0} H^T_{k_0,k_1} H^T_{k_1,k_2} \cdots H^T_{k_{\ell-1},k_\ell}}{p_{k_0} P_{k_0,k_1} P_{k_1,k_2} \cdots P_{k_{\ell-1},k_\ell}} \quad \text{for } \ell = 0, 1, 2, \ldots.$$

Algorithm 1 Sequential Monte Carlo.

```
Require: matrix $\boldsymbol{H}$, vector $\mathbf{b}$, and initial point $\mathbf{x}_0$
Ensure: solution vector $\mathbf{x}^*$ of $\mathbf{x} = \boldsymbol{H}\mathbf{x} + \mathbf{b}$
  $\mathbf{x}^{old} = \mathbf{x}_0$
  while convergence requirement not met do
    compute residual $\mathbf{r} = \mathbf{b} + \boldsymbol{H}\mathbf{x}^{old} - \mathbf{x}^{old}$
    apply direct MC to approximate $\Delta\mathbf{x} \simeq (\boldsymbol{I} - \boldsymbol{H})^{-1}\mathbf{r}$
    update solution $\mathbf{x}^{new} = \mathbf{x}^{old} + \Delta\mathbf{x}$
    $\mathbf{x}^{old} = \mathbf{x}^{new}$
  end
```

As a result we can estimate the solution vector $\mathbf{x}$ for all components through the expression

$$x_i = \mathrm{E}\left[\sum_{\ell=0}^{\infty} W_\ell \delta_{k_\ell, i}\right], \quad \text{where } \delta_{i,j} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

The above formula (3) shows a single chain can estimate every entry of the solution vector $\mathbf{x}$. To summarize, at each step the random walk will add a contribution to the associated component i if it lands on state i.

In the following sections of this paper, we present our theoretical findings and new techniques based on the forward method. Many of the conclusions hold also for the adjoint method as the technique is fundamentally the same as described here.

2.3. A key and fundamental limitation. The overall algorithms for the MC approach rely on the central limit theorem to estimate $\mathrm{E}[X]$ as the average of many realizations of the random variable X. As discussed by Ji, Mascagni, and Li (2013) and Wasow (1952), using MC simulation does not guarantee convergence because a necessary and sufficient condition to estimate $\mathrm{E}[X]$ using the empirical mean value of X is $\mathrm{Var}[X] < \infty$. Empirical studies (Ji, Mascagni, and Li, 2013; Benzi et al., 2017) show that it is common to have $\mathrm{Var}[X] = \infty$ even when the Neumann series converges (i.e., $\rho(\boldsymbol{H}) < 1$). Additionally, the variance itself affects the convergence speed of the MC simulation. So having a smaller variance is always an improvement of the algorithm. It is easy to see that $\mathrm{Var}[X]$ depends on the transition matrix $\boldsymbol{P}$ and the initial probability vector $\mathbf{p}$; we will analyze the relations between variance and transitions after we introduce our multiway random walk, as it generalizes the standard procedure.

2.4. Sequential Monte Carlo. A downside of these direct methods is that the convergence of directly applying the MC simulation is slow. The accuracy of each estimate follows the conclusions from the central limit theorem, which yields that the convergence rate of the direct method has a term of $1/\sqrt{N}$, where N is the number of simulations. This can require prohibitively large numbers of simulations, each of which could be quite long, if we require high accuracy.

A more practical approach is to sequentially apply the standard MC scheme to iteratively improve the accuracy of the solution in the fashion of iterative refinement. This method is called sequential Monte Carlo (SMC) (Halton, 1994). Algorithm 1 shows the general procedure of such an approach.

As we note from the algorithm, each time the direct MC method is applied to the linear system defined by the residual vector. For instance, if the relative error of the residual can be decreased to 0.1 on average for each iteration, then it only requires around 8 iterations to reach an overall relative error at 10^{-8}. The advantage of SMC over the direct method is faster convergence in terms of total number of simulation steps. The sequential method only seeks to approximate the solution at each iteration so the total number of simulations required is much smaller. We will see this trade-off in practice in the study in section 5.4.

3. Multiway Markov random walks.

In this section we generalize the idea of using a random walk for estimating the functional $\langle \mathbf{h}, \mathbf{x} \rangle$ to using a hypermatrix of transitions to compute the estimate. Then we analyze the convergence of the simulations based on the variance of the relevant random variable. We show that the variance of this multiway random walk is a power series in a specific matrix, thus the convergence of the MC simulation is determined by the spectral radius of this matrix. Our notation for hypermatrices as in $\underline{\boldsymbol{P}}$ are bold, underlined, uppercase letters. For a mode-3 hypermatrix $\underline{\boldsymbol{P}}$, its elements are denoted by $\underline{P}^{(\ell)}_{i,j}$.

3.1. Hypermatrix transitions.

Instead of using a fixed transition matrix $\boldsymbol{P}$ as in the classic MC method described in section 2, we extend this framework by allowing the random walk to vary transition matrices along each step. To put a limit on the number of different transition matrices, an m-way random walk is defined as walking via m different transition matrices periodically in a round-robin way. Formally, we define an m-way Markov random walk Z_t on the state space, $S = \{1, 2, \ldots, n\}$, where the initial probability follows $\mathbf{p}$, and the transition probability follows a hypermatrix $\underline{\boldsymbol{P}}$ with m slices:

$$\begin{aligned} \Pr(Z_0 = i) &= p_i, \\ \Pr(Z_{\ell+1} = j \mid Z_\ell = i) &= \underline{P}^{(\mathrm{mod}(\ell, m)+1)}_{i,j}. \end{aligned} \tag{4}$$

Here $\mathrm{mod}(\ell, m)$ denotes the remainder after dividing ℓ by m. For notation simplicity, we are going to pretend that $\underline{\boldsymbol{P}}$ has an unlimited number of slices by using: $\underline{P}^{(\ell)}_{i,j}$ to denote $\underline{P}^{(\mathrm{mod}(\ell-1, m)+1)}_{i,j}$ for $\ell = 1, 2, \ldots$. So under this notation, we have $\underline{\boldsymbol{P}}^{(\ell)} = \underline{\boldsymbol{P}}^{(\ell+m)}$ for $\ell = 1, 2, \ldots$, and also $\Pr(Z_{\ell+1} = j \mid Z_\ell = i) = \underline{P}^{\ell+1}_{i,j}$.

The above definition of multiway random walk is an instance of inhomogeneous chains, which have been applied in various areas (Davis and Principe, 1993; Brémaud, 2013). However this idea has never been applied to the MC method on linear systems. In this paper we show that under this new m-way random walk, several improvements can be made into the standard algorithm. The reason we define the m-way random walk in a round-robin manner is that we later find out the variance is directly related to the matrix by grouping these m transition matrices together, and there exists an efficient algorithm to construct these m transition matrices.

3.2. Estimating with multiway random walks.

Our goal is to compute the functional $\langle \mathbf{h}, \mathbf{x} \rangle$, where $\mathbf{x}$ is the solution of linear system $\mathbf{x} = \boldsymbol{H}\mathbf{x} + \mathbf{b}$. Throughout the paper we have the basic assumption $\rho(\boldsymbol{H}) < 1$. We also exclude the corner cases where $\mathbf{h}$ is a zero vector, or $\boldsymbol{H}$ has zero rows/columns, otherwise the zero rows/columns of $\boldsymbol{H}$ can be easily removed by some basic linear algebra transformation.

If we construct the initial probability $\mathbf{p}$ and the transition hypermatrix $\underline{\boldsymbol{P}}$ such that

$$h_i \neq 0 \Rightarrow p_i \neq 0 \quad \text{and} \quad H_{i,j} \neq 0 \Rightarrow \underline{P}^{(\ell)}_{i,j} \neq 0,$$

then for a random walk realization,

$$k_0 \to k_1 \to k_2 \to \cdots \to k_\ell \to \cdots,$$

we can define the related weights W_ℓ and the variable Z in a similar way with section 2. Formally,

$$\begin{aligned} W_\ell &= \frac{h_{k_0} H_{k_0,k_1} H_{k_1,k_2} \cdots H_{k_{\ell-1},k_\ell}}{p_{k_0} \underline{P}^{(1)}_{k_0,k_1} \underline{P}^{(2)}_{k_1,k_2} \cdots \underline{P}^{(\ell)}_{k_{\ell-1},k_\ell}} \quad \text{for } \ell = 0,1,2,\ldots, \\ Z &= \sum_{\ell=0}^{\infty} W_\ell b_{k_\ell}. \end{aligned} \tag{5}$$

It is worth noting the above definition of multiway Markov random walk is a generalization of the standard Markov chain, which is the special case with $m = 1$.

The following theorem shows that this random walk generalization has the same expectation. Thus it also can be used to approximate the solution by MC simulation.

THEOREM 1. *For the linear system* $\mathbf{x} = \boldsymbol{H}\mathbf{x} + \mathbf{b}$ *with* $\rho(\boldsymbol{H}) < 1$*, the random variable* Z *defined from* (5) *has the expected value* $\mathrm{E}[Z] = \langle \mathbf{h}, \mathbf{x} \rangle$*.*

Proof. We first prove that $\mathrm{E}[W_\ell b_{k_\ell}] = \langle \mathbf{h}, \boldsymbol{H}^\ell \mathbf{b} \rangle$ for all $\ell = 0,1,2,\ldots$. Then the convergence of the Neumann series will give us $\mathrm{E}[Z] = \langle \mathbf{h}, \mathbf{x} \rangle$.

We have $\mathrm{E}[W_0 b_{k_0}] = \sum_{p_{k_0} \neq 0} \frac{h_{k_0}}{p_{k_0}} b_{k_0} p_{k_0} = \sum_{h_{k_0} \neq 0} h_{k_0} b_{k_0} = \langle \mathbf{h}, \mathbf{b} \rangle$. Similarly for the case of $\ell \geq 1$,

$$\begin{aligned} \mathrm{E}[W_\ell b_{k_\ell}] &= \sum_{k_0} \sum_{k_1} \cdots \sum_{k_\ell} \frac{h_{k_0} H_{k_0,k_1} H_{k_1,k_2} \cdots H_{k_{\ell-1},k_\ell}}{p_{k_0} \underline{P}^{(1)}_{k_0,k_1} \underline{P}^{(2)}_{k_1,k_2} \cdots \underline{P}^{(\ell)}_{k_{\ell-1},k_\ell}} b_{k_\ell} p_{k_0} \underline{P}^{(1)}_{k_0,k_1} \underline{P}^{(2)}_{k_1,k_2} \cdots \underline{P}^{(\ell)}_{k_{\ell-1},k_\ell} \\ &= \sum_{k_0} \sum_{k_1} \cdots \sum_{k_\ell} h_{k_0} H_{k_0,k_1} H_{k_1,k_2} \cdots H_{k_{\ell-1},k_\ell} b_{k_\ell} \\ &= \langle \mathbf{h}, \boldsymbol{H}^\ell \mathbf{b} \rangle. \end{aligned}$$

So $\mathrm{E}[Z] = \sum_{\ell=0}^{\infty} \mathrm{E}[W_\ell b_{k_\ell}] = \langle \mathbf{h}, \sum_{\ell=0}^{\infty} \boldsymbol{H}^\ell \mathbf{b} \rangle = \langle \mathbf{h}, \mathbf{x} \rangle$. □

3.3. Convergence analysis.

In order to statistically estimate $\mathrm{E}[Z]$, we need to ensure $\mathrm{Var}[Z] < \infty$. The following theorem reveals the explicit form of $\mathrm{Var}[Z]$ determined by $\mathbf{h}, \mathbf{b}, \boldsymbol{H}$ and the m-way random walk transition probabilities $(\mathbf{p}, \underline{\boldsymbol{P}})$.

THEOREM 2. *For the linear system* $\mathbf{x} = \boldsymbol{H}\mathbf{x} + \mathbf{b}$ *with* $\rho(\boldsymbol{H}) < 1$*, if* $\boldsymbol{H}$ *and* $\mathbf{b}$ *are nonnegative,* Z *defined from* (5) *has variance*

$$\mathrm{Var}[Z] = \left\langle \hat{\mathbf{h}}, \sum_{i=0}^{\infty} \tilde{\boldsymbol{H}}^i \boldsymbol{G} \,\mathrm{Diag}(\mathbf{b})(2\boldsymbol{H}\mathbf{x} + \mathbf{b}) \right\rangle - \langle \mathbf{h}, \mathbf{x} \rangle^2, \tag{6}$$

where $\mathrm{Diag}(\mathbf{b})$ *is a diagonal matrix with diagonal entries equal to* $\mathbf{b}$*, and* $\hat{\mathbf{h}}, \tilde{\boldsymbol{H}}, \boldsymbol{G}$ *are defined as*

$$\hat{h}_i = \begin{cases} h_i^2/p_i & \text{if } h_i \neq 0, \\ 0 & \text{if } h_i = 0, \end{cases} \qquad \hat{H}^{(\ell)}_{i,j} = \begin{cases} H_{i,j}^2/\underline{P}^{(\ell)}_{i,j} & \text{if } H_{i,j} \neq 0, \\ 0 & \text{if } H_{i,j} = 0, \end{cases}$$

$$\tilde{\boldsymbol{H}} = \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(m)}, \qquad \boldsymbol{G} = \boldsymbol{I} + \hat{\boldsymbol{H}}^{(1)} + \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} + \cdots + \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(m-1)}.$$

Proof. Since $\mathrm{Var}[Z] = \mathrm{E}[Z^2] - (\mathrm{E}[Z])^2 = \mathrm{E}[Z^2] - \langle \mathbf{h}, \mathbf{x} \rangle^2$, we will focus on computing $\mathrm{E}[Z^2]$:

$$\mathrm{E}[Z^2] = \mathrm{E}\left[\sum_{\ell=0}^{\infty} W_\ell^2 b_{k_\ell}^2 + 2\sum_{r>\ell} W_\ell W_r b_{k_\ell} b_{k_r}\right].$$

Since all the intermediate terms are nonnegative, by Tonelli's theorem we can analyze the sum in pieces and the summation of the above terms are interchangeable.

For the first part of $\mathrm{E}[Z^2]$, when $\ell = 0$, we have $\mathrm{E}[W_0^2 b_{k_0}^2] = \sum_{p_{k_0} \neq 0} \frac{h_{k_0}^2}{p_{k_0}^2} b_{k_0}^2 p_{k_0} = \sum_{\hat{h}_{k_0} \neq 0} \hat{h}_{k_0} b_{k_0}^2 = \langle \hat{\mathbf{h}}, \mathrm{Diag}(\mathbf{b})\mathbf{b} \rangle$, and when $\ell \geq 1$,

$$\begin{aligned}
\mathrm{E}[W_\ell^2 b_{k_\ell}^2] &= \sum_{k_0}\sum_{k_1}\cdots\sum_{k_\ell}\left(\frac{h_{k_0} H_{k_0,k_1} H_{k_1,k_2} \cdots H_{k_{\ell-1},k_\ell}}{p_{k_0} \underline{P}^{(1)}_{k_0,k_1} \underline{P}^{(2)}_{k_1,k_2} \cdots \underline{P}^{(\ell)}_{k_{\ell-1},k_\ell}}\right)^2 b_{k_\ell}^2 p_{k_0} \underline{P}^{(1)}_{k_0,k_1} \underline{P}^{(2)}_{k_1,k_2} \cdots \underline{P}^{(\ell)}_{k_{\ell-1},k_\ell} \\
&= \sum_{k_0}\sum_{k_1}\cdots\sum_{k_\ell} \hat{h}_{k_0} \hat{H}^{(1)}_{k_0,k_1} \hat{H}^{(2)}_{k_1,k_2} \cdots \hat{H}^{(\ell)}_{k_{\ell-1},k_\ell} b_{k_\ell}^2 \qquad (7) \\
&= \langle \hat{\mathbf{h}}, \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(\ell)} \mathrm{Diag}(\mathbf{b})\mathbf{b} \rangle.
\end{aligned}$$

Applying the above result from (7), we have

$$E\left[\sum_{\ell=0}^{\infty} W_\ell^2 b_{k_\ell}^2\right] = \left\langle \hat{\mathbf{h}}, \left(\boldsymbol{I} + \sum_{\ell=1}^{\infty} \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(\ell)}\right) \mathrm{Diag}(\mathbf{b})\mathbf{b} \right\rangle.$$

Denote $\boldsymbol{F} = \boldsymbol{I} + \sum_{\ell=1}^{\infty} \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(\ell)}$. Then we have

$$\begin{aligned}
\boldsymbol{F} &= \boldsymbol{I} + \hat{\boldsymbol{H}}^{(1)} + \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} + \cdots + \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(m-1)} + \sum_{\ell=m}^{\infty} \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(\ell)} \\
&= \boldsymbol{G} + \sum_{\ell=m}^{\infty} \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(\ell)} \\
&= \boldsymbol{G} + \tilde{\boldsymbol{H}} \left(\boldsymbol{I} + \sum_{\ell=1}^{\infty} \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(\ell)}\right) \\
&\qquad \text{(here we extract the shared term } \tilde{\boldsymbol{H}} = \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(m)} \text{ as } \hat{\boldsymbol{H}}^{(\ell)} \text{is periodic)} \\
&= \boldsymbol{G} + \tilde{\boldsymbol{H}} \boldsymbol{F} = \boldsymbol{G} + \tilde{\boldsymbol{H}}(\boldsymbol{G} + \tilde{\boldsymbol{H}} \boldsymbol{F}) \\
&= \sum_{i=0}^{\infty} \tilde{\boldsymbol{H}}^i \boldsymbol{G}.
\end{aligned}$$

Then after inserting the result of $\boldsymbol{F}$, we have

$$\mathrm{E}\left[\sum_{\ell=0}^{\infty} W_\ell^2 b_{k_\ell}^2\right] = \left\langle \hat{\mathbf{h}}, \boldsymbol{F}\, \mathrm{Diag}(\mathbf{b})\mathbf{b} \right\rangle = \left\langle \hat{\mathbf{h}}, \sum_{i=0}^{\infty} \tilde{\boldsymbol{H}}^i \boldsymbol{G}\, \mathrm{Diag}(\mathbf{b})\mathbf{b} \right\rangle. \qquad (8)$$

Next we compute the second part of $\mathrm{E}[Z^2]$:

$$
\begin{aligned}
&(9)\\
&\mathrm{E}\left[\sum_{r>\ell} W_\ell W_r b_{k_\ell} b_{k_r}\right] = \mathrm{E}\left[\sum_{\ell=0}^{\infty} W_\ell b_{k_\ell}\left(\sum_{r=\ell+1}^{\infty} W_r b_{k_r}\right)\right]\\
&= \sum_{\ell=0}^{\infty}\sum_{k_0}\cdots\sum_{k_\ell}\left(\frac{h_{k_0}H_{k_0,k_1}H_{k_1,k_2}\cdots H_{k_{\ell-1},k_\ell}}{p_{k_0}\underline{P}^{(1)}_{k_0,k_1}\underline{P}^{(2)}_{k_1,k_2}\cdots \underline{P}^{(\ell)}_{k_{\ell-1},k_\ell}}\right)^2\\
&\qquad \times\left(\sum_{r=\ell+1}^{\infty}\sum_{k_{\ell+1}}\cdots\sum_{k_r}\frac{H_{k_\ell,k_{\ell+1}}\cdots H_{k_{r-1},k_r}}{\underline{P}^{(\ell+1)}_{k_\ell,k_{\ell+1}}\cdots \underline{P}^{(r)}_{k_{r-1},k_r}}\right.\\
&\qquad \left.\times\ p_{k_0}\underline{P}^{(1)}_{k_0,k_1}\underline{P}^{(2)}_{k_1,k_2}\cdots \underline{P}^{(\ell)}_{k_{\ell-1},k_\ell}\underline{P}^{(\ell+1)}_{k_\ell,k_{\ell+1}}\cdots \underline{P}^{(r)}_{k_{r-1},k_r} b_{k_\ell} b_{k_r}\right)
\end{aligned}
$$

(here, we have extracted all the $\ell+1$ prefix terms in W_ℓ and W_r that are the same because W_r covers W_ℓ)

$$
\begin{aligned}
&= \sum_{\ell=0}^{\infty}\sum_{k_0}\cdots\sum_{k_\ell}\left(\hat{h}_{k_0}\hat{H}^{(1)}_{k_0,k_1}\hat{H}^{(2)}_{k_1,k_2}\cdots\hat{H}^{(\ell)}_{k_{\ell-1},k_\ell}\right)b_{k_\ell}\left(\sum_{r=\ell+1}^{\infty}\sum_{k_{\ell+1}}\cdots\sum_{k_r} H_{k_\ell,k_{\ell+1}}\cdots H_{k_{r-1},k_r}\right)b_{k_r}\\
&= \left\langle \hat{\mathbf{h}}, \sum_{\ell=0}^{\infty}(\hat{\boldsymbol{H}}^{(1)}\cdots\hat{\boldsymbol{H}}^{(\ell)})\,\mathrm{Diag}(\mathbf{b})\left(\sum_{r=\ell+1}^{\infty}\boldsymbol{H}^{r-\ell}\mathbf{b}\right)\right\rangle\\
&= \left\langle \hat{\mathbf{h}}, \sum_{\ell=0}^{\infty}(\hat{\boldsymbol{H}}^{(1)}\cdots\hat{\boldsymbol{H}}^{(\ell)})\,\mathrm{Diag}(\mathbf{b})\boldsymbol{H}\mathbf{x}\right\rangle\\
&= \left\langle \hat{\mathbf{h}}, \sum_{i=0}^{\infty}\tilde{\boldsymbol{H}}^{i}\boldsymbol{G}\,\mathrm{Diag}(\mathbf{b})\boldsymbol{H}\mathbf{x}\right\rangle.
\end{aligned}
$$

For these final steps, we used the Neumann series to move from the power series $\sum_{r=\ell+1}^{\infty}\boldsymbol{H}^{r-\ell}\mathbf{b}$ to $\boldsymbol{H}\mathbf{x}$ and then used the periodicity again to rewrite the expressions in terms of $\boldsymbol{G}$. Now, combining the results from (8) and (9) we have the result. □

For the general cases of $\boldsymbol{H}, \mathbf{b}$ without the assumption of nonnegativity, if $\rho(\tilde{\boldsymbol{H}}) < 1$, the above conclusion (i.e., (6)) still holds according to Fubini's theorem. To see that, when $\rho(\tilde{\boldsymbol{H}}) < 1$, the variance in (6) is bounded, which is the value if $\boldsymbol{H}$ and $\mathbf{b}$ are changed into a positive matrix or vector. Then the original summation is interchangeable as the summation for the terms in absolute value is bounded.

Combining both of these results, the following corollary is straightforward from the conclusion of Theorem 2.

COROLLARY 3. *For the linear system* $\mathbf{x} = \boldsymbol{H}\mathbf{x} + \mathbf{b}$ *with* $\rho(\boldsymbol{H}) < 1$, *if the spectral radius* $\rho(\tilde{\boldsymbol{H}}) < 1$, *then* $\mathrm{Var}[Z] = \langle \hat{\mathbf{h}}, (\boldsymbol{I} - \tilde{\boldsymbol{H}})^{-1}\boldsymbol{G}\,\mathrm{Diag}(\mathbf{b})(2\boldsymbol{H}\mathbf{x} + \mathbf{b})\rangle - \langle \mathbf{h}, \mathbf{x}\rangle^2 < \infty$.

The above analysis of $\mathrm{Var}[Z]$ shows that with the condition $\rho(\tilde{\boldsymbol{H}}) < 1$ and, by the law of large numbers, we can estimate the value of $\langle \mathbf{h}, \mathbf{x}\rangle$ from the variable Z. For the cases when $\rho(\tilde{\boldsymbol{H}}) \geq 1$, the following corollary shows that it is possible to have $\mathrm{Var}[Z] = \infty$. The essence of the idea and proof is just that we can construct a vector to touch the dominant eigenspace associated with eigenvalues ≥ 1.

COROLLARY 4. *Under the same assumptions as Theorem* 2, *if the spectral radius* $\rho(\tilde{\boldsymbol{H}}) \geq 1$, *and if* $\boldsymbol{G}$ *is full rank, then there always exist some* $\mathbf{b}, \mathbf{h} \in \mathbb{R}^n$ *such that* $\text{Var}[Z] = \infty$. *(Note that for the standard MC method (i.e.,* $m = 1$*), since* $\boldsymbol{G} = \boldsymbol{I}$*, the method diverges for certain* $\mathbf{b}, \mathbf{h}$*.)*

Proof. For simplicity, let us write $\text{Var}[Z] = \hat{\mathbf{h}}^T \sum_{i=0}^{\infty} \tilde{\boldsymbol{H}}^i \boldsymbol{G}\mathbf{q} - \mathbf{h}^T\mathbf{x}$, where the vector $\mathbf{q} = \text{Diag}(\mathbf{b})(2\boldsymbol{H}\mathbf{x} + \mathbf{b})$. Since our focus is to show that $\text{Var}[Z] = \infty$ we can focus on the first term alone. Now, because $\rho(\tilde{\boldsymbol{H}}) \geq 1$, there must be a nontrivial subspace associated with the eigenvalues with magnitude at least 1. This subspace also must include a nonnegative vector because $\tilde{\boldsymbol{H}}$ is nonnegative and the Perron–Frobenius theory guarantees a nonnegative matrix has a nonnegative eigenvector with the spectral radius (Horn and Johnson, 2012, Theorem 8.3.1). Consequently, we can let $\mathbf{h}$ be any vector so that $\hat{\mathbf{h}}$ satisfies $\|\hat{\mathbf{h}}^T \tilde{\boldsymbol{H}}\| \geq \|\hat{\mathbf{h}}\|$, where it suffices to set $\hat{\mathbf{h}}$ to the nonnegative dominant eigenvector in which case $\mathbf{h}$ and p_i can be set to give this $\hat{\mathbf{h}}$ under the conditions of Theorem 2. Now, note that $\text{Var}[Z]$ will equal infinity unless $\boldsymbol{G}\mathbf{q}$ will necessarily touch a subspace where powers decay (that is, if $\|\tilde{\boldsymbol{H}}^i \boldsymbol{G}\mathbf{q}\| < \|\boldsymbol{G}\mathbf{q}\|$). Again, we know that there is some right eigenvector $\mathbf{v}$ of $\tilde{\boldsymbol{H}}$ associated with the eigenvalue with magnitude at least 1. Consequently, we need to ensure we can choose $\mathbf{b}$, which in turn determines $\mathbf{q}$, such that $\mathbf{v}^T\boldsymbol{G}\mathbf{q} \neq 0$ to make sure that we can always touch the growing subspace. For simplicity of notation, let $\mathbf{y} = \boldsymbol{G}^T\mathbf{v}$. We know that $\mathbf{y} \neq 0$ because $\boldsymbol{G}$ is full rank and $\mathbf{v}$ is an eigenvector.

Note the following, $\mathbf{q}$ can be analyzed as a function of either $\mathbf{b}$ or $\mathbf{x}$ because $(\boldsymbol{I} - \boldsymbol{H})\mathbf{x} = \mathbf{b}$ is nonsingular (as in the original problem). As a function of $\mathbf{x}$, then $\mathbf{q}(\mathbf{x}) = \text{Diag}((\boldsymbol{I} - \boldsymbol{H})\mathbf{x})(\boldsymbol{I} + \boldsymbol{H})\mathbf{x} = \text{Diag}(\mathbf{x})\mathbf{x} - \text{Diag}(\boldsymbol{H}\mathbf{x})\boldsymbol{H}\mathbf{x}$. (This follows by canceling the terms $\text{Diag}(\mathbf{x})\boldsymbol{H}\mathbf{x} - \text{Diag}(\boldsymbol{H}\mathbf{x})\mathbf{x}$, which are elementwise equal.) Hence, we wish to establish that for a specific $\mathbf{y}$, we can always choose $\mathbf{x}$ such that $\mathbf{y}^T[\text{Diag}(\mathbf{x})\mathbf{x} - \text{Diag}(\boldsymbol{H}\mathbf{x})\boldsymbol{H}\mathbf{x}] \neq 0$.

Assume by way of contradiction that $\mathbf{y}^T\mathbf{q}(\mathbf{x}) = 0$ for all $\mathbf{x}$. This implies

$$\begin{aligned}\mathbf{y}^T\mathbf{q}(\mathbf{x}) &= \mathbf{e}^T \text{Diag}(\mathbf{y}) \text{Diag}(\mathbf{x})\mathbf{x} - \mathbf{e}^T \text{Diag}(\mathbf{y}) \text{Diag}(\boldsymbol{H}\mathbf{x})\mathbf{x} \\ &= \mathbf{x}^T \underbrace{[\text{Diag}(\mathbf{y}) - \boldsymbol{H}^T \text{Diag}(\mathbf{y})\boldsymbol{H}]}_{=\boldsymbol{C}} \mathbf{x} = 0,\end{aligned}$$

because diagonal matrices commute. Since we have $\mathbf{x}^T\boldsymbol{C}\mathbf{x} = 0$ for all $\mathbf{x}$, then $\boldsymbol{C} = 0$ will be zero. Thus, we have $\text{Diag}(\mathbf{y}) = \boldsymbol{H}^T \text{Diag}(\mathbf{y})\boldsymbol{H}$. Now, this equality yields the bound $\rho(\text{Diag}(\mathbf{y})) \leq \rho(\boldsymbol{H}^T)\rho(\text{Diag}(\mathbf{y}))\rho(\boldsymbol{H})$, and this is our contradiction because we are given that $\rho(\boldsymbol{H}) < 1$, which makes the previous expression read $\rho(\text{Diag}(\mathbf{y})) < \rho(\text{Diag}(\mathbf{y}))$, where $\mathbf{y} \neq 0$. □

Remark. Although Corollary 4 states that for the case of $\rho(\tilde{\boldsymbol{H}}) \geq 1$, there exists some $\mathbf{b}, \mathbf{h} \in \mathbb{R}^n$ to make the variance infinite. It does *not* guarantee the variance is bounded for other values of $\mathbf{b}$ and $\mathbf{h}$. In fact, from the proof details we can see that as long as there is a nonzero part projecting to the leading eigenvector with eigenvalue ≥ 1, then the variance goes to infinity. And this is likely to occur for some general class of $\mathbf{b}$ and $\mathbf{h}$.

4. Multiway Monte Carlo method.

In this section we discuss two aspects of practically applying the MC method based on the multiway Markov random walk introduced in section 3. First, we detail the construction of the transition hypermatrix $\underline{\boldsymbol{P}}$. Second, we give the error analysis regarding the truncation of the random walk, as well as the probable error from the stochastic estimation.

Algorithm 2 Compute transition hypermatrix.

```
Require: matrix $\boldsymbol{H}$
Ensure: transition hypermatrix $\underline{\boldsymbol{P}}$
initialization $\omega_i = 1$ for $i = 1, 2, \ldots, n$
for $k = m$ to $1$ do
    $\eta_i = \sum_{\ell=1}^{n} \omega_\ell |H_{i,\ell}|$    for $i = 1, 2, \ldots, n$
    $\underline{P}_{i,j}^{(k)} = \omega_j |H_{i,j}| / \eta_i$
    $\omega_i = \eta_i$    for $i = 1, 2, \ldots, n$
end
```

4.1. Transition hypermatrix. In section 3, Corollaries 3 and 4 indicate that the spectral radius of matrix $\tilde{\boldsymbol{H}}$ is crucial to the variance Var$[Z]$. The matrix $\tilde{\boldsymbol{H}}$ is determined by the transition hypermatrix $\underline{\boldsymbol{P}}$. Since it is usually computationally inefficient to directly compute the spectral radius of a matrix, the common practice is to find an upper bound of $\rho(\tilde{\boldsymbol{H}})$. The spectral radius of a matrix is bounded by any submultiplicative matrix norm. As is standard in the literature, we use the infinity norm $\| \cdot \| \overset{\text{def}}{=} \| \cdot \|_\infty$ for this section.

We first consider the case for the standard Markov random walk (i.e., $m = 1$), where $\tilde{H}_{i,j} = \hat{H}_{i,j}^{(1)} = H_{i,j}^2 / \underline{P}_{i,j}^{(1)}$. The following lemma (Ji, Mascagni, and Li, 2013) provides insight on how to assign the values of the transition matrix $\underline{\boldsymbol{P}}^{(1)}$ in order to minimize the norm.

LEMMA 5 (Ji, Mascagni, and Li (2013)). *Let* $\mathbf{h} = (h_1, h_2, \ldots, h_n)^T$ *be a vector, where at least one of its elements is nonzero:* $h_k \neq 0$ *for some* $k \in \{1, 2, \ldots, n\}$. *Let* $\mathbf{p} = (p_1, p_2, \ldots, p_n)^T$ *be a probability distribution vector. Then* $\sum_{i=1}^{n} h_i^2 / p_i \geq (\sum_{i=1}^{n} |h_i|)^2$, *and the lower bound is attained when* $p_i = |h_i| / \sum_{r=1}^{n} |h_r|$.

According to Lemma 5, the infinity norm

$$\|\tilde{\boldsymbol{H}}\| = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |\tilde{H}_{i,j}| = \max_{1 \leq i \leq n} \sum_{j=1}^{n} \frac{H_{i,j}^2}{\underline{P}_{i,j}^{(1)}} \geq \max_{1 \leq i \leq n} \left(\sum_{j=1}^{n} |H_{i,j}| \right)^2 = (\|\boldsymbol{H}\|)^2.$$

And when $\underline{\boldsymbol{P}}^{(1)}$ is defined as

$$\underline{P}_{i,j}^{(1)} = \frac{|H_{i,j}|}{\sum_{k=1}^{n} |H_{i,k}|} \text{ for all } i, j = 1, 2, \ldots, n,$$

the above lower bound $(\|\boldsymbol{H}\|)^2$ is reached, making this choice in some sense optimal.

However, for a variety of problems, this choice is unlikely to result in a method that will have $\rho(\tilde{\boldsymbol{H}}) < 1$. For a linear system $\boldsymbol{A}\mathbf{x} = \mathbf{b}$, we can rewrite it into $\mathbf{x} = \boldsymbol{H}\mathbf{x} + \mathbf{b}$ as $\boldsymbol{H} = \boldsymbol{I} - \boldsymbol{A}$. It is common to have $\rho(\boldsymbol{H})$ be very close to 1 even with the help of preconditioners (Benzi et al., 2017). Since the infinity norm is generally a loose upper bound for the spectral radius, $\|\boldsymbol{H}\| > 1$ is likely (Benzi et al., 2017) to be true. We will show we can tighten these results with our multiway walks.

We describe the method for computing $\underline{\boldsymbol{P}}$ for an m-way Markov random walk in Algorithm 2, then we prove in Theorem 6 that it minimizes $\|\tilde{\boldsymbol{H}}\|$.

It is worth noting that Algorithm 2 only takes linear time in the number of nonzeros of the matrix $\boldsymbol{H}$ at each iteration. Also the output result of the transition hypermatrix is compatible with different values of m, which means that m does not need to be predefined to run the algorithm. In other words, we can stop the iteration anytime we want and still get the output hypermatrix for some smaller m. This is useful when we later discuss how to choose the value of m, as it turns out that we can set a criterion to stop the iteration. Last we see that the output transition hypermatrix $\underline{\boldsymbol{P}}$ is only determined by $\boldsymbol{H}$. So the procedure of computing $\underline{\boldsymbol{P}}$ is similar to loading the matrix into the memory as they both only need to be done once for different problems (i.e., different $\mathbf{h}$ and $\mathbf{b}$). On the other hand, the operations in Algorithm 2 work akin to matrix-vector products. This means that we need global computation to compute this sequence and this choice prohibits a purely local algorithm.

Another property to note is that the multiway procedure walk created by Algorithm 2 will have no multiway effects when all the row sums of $\boldsymbol{H}^+$ are the same. When $\boldsymbol{H}^+$ has the same row sums, then the output transition matrices $\underline{\boldsymbol{P}}^{(1)}, \underline{\boldsymbol{P}}^{(2)}, \dots, \underline{\boldsymbol{P}}^{(m)}$ will be the same, where $\underline{\boldsymbol{P}}^{(i)}$ for $i = 1, 2, \dots, m$ denotes the matrix slice from the hypermatrix $\underline{\boldsymbol{P}}$.

The following theorem proves that the output from Algorithm 2 reaches optimality.

Theorem 6. *Let $\underline{\boldsymbol{P}}$ be the output of Algorithm* 2, *then $\tilde{\boldsymbol{H}}$ defined in Theorem* 2 *has reached its minimal infinity norm.*

Proof. Denote $\eta_i^{(k)}$ as the value of η_i at the kth iteration.

We first prove that the value of $\|\tilde{\boldsymbol{H}}\|$ cannot be further decreased by changing $\underline{\boldsymbol{P}}^{(m)}$. Since $\tilde{\boldsymbol{H}}$ is a nonnegative matrix, $\|\tilde{\boldsymbol{H}}\| = \|\tilde{\boldsymbol{H}}\mathbf{e}\|$ holds. The kth element of $\hat{\boldsymbol{H}}^{(m)}\mathbf{e}$ is $\sum_{j=1}^n H_{k,j}^2 / \underline{P}_{k,j}^{(m)}$ and, according to Lemma 5, this value is minimized when $\underline{P}_{k,i}^{(m)} = |H_{k,i}| / \sum_{j=1}^n |H_{k,j}|$, which is exactly the kth row of $\underline{\boldsymbol{P}}^{(m)}$ from the algorithm. Thus by changing $\underline{\boldsymbol{P}}^{(m)}$, we cannot decrease any elements of vector $\hat{\boldsymbol{H}}^{(m)}\mathbf{e}$, and thus

$$\|\tilde{\boldsymbol{H}}\| = \|\tilde{\boldsymbol{H}}\mathbf{e}\| = \|(\hat{\boldsymbol{H}}^{(1)} \cdots \hat{\boldsymbol{H}}^{(m-1)})\hat{\boldsymbol{H}}^{(m)}\mathbf{e}\|$$

will not decrease because all the matrices involved are nonnegative and $\boldsymbol{A}\mathbf{x} \le \boldsymbol{A}\mathbf{y}$ (elementwise) for a nonnegative matrix when $\mathbf{x} \le \mathbf{y}$ (elementwise).

Second we note that

$$\big((\eta_1^{(m)})^2, (\eta_2^{(m)})^2, \dots, (\eta_n^{(m)})^2\big)^T = \hat{\boldsymbol{H}}^{(m)}\mathbf{e}.$$

This follows directly from the algorithm because $\underline{\boldsymbol{P}}^{(m)}$ is constructed as $\underline{P}_{k,i}^{(m)} = |H_{k,i}| / \sum_{j=1}^n |H_{k,j}|$, which means the kth element of $\hat{\boldsymbol{H}}^{(m)}\mathbf{e}$ is $(\sum_{j=1}^n |H_{k,j}|)^2 = (\eta_k^{(m)})^2$.

Last we use mathematical induction by assuming that

- we cannot decrease $\|\tilde{\boldsymbol{H}}\|$ by changing $\underline{\boldsymbol{P}}^{(\ell)}$ for $r + 1 \le \ell \le m$;
- $\big((\eta_1^{(\ell)})^2, (\eta_2^{(\ell)})^2, \dots, (\eta_n^{(\ell)})^2\big)^T = \hat{\boldsymbol{H}}^{(\ell)} \cdots \hat{\boldsymbol{H}}^{(m)}\mathbf{e}$ for $r + 1 \le \ell \le m$.

Since we already proved the statement holds for the case of $\ell = m$, then similarly we prove that the statement holds for $\underline{\boldsymbol{P}}^{(r)}$. We notice that

$$\text{the } k\text{th element of } \hat{\boldsymbol{H}}^{(r)}\hat{\boldsymbol{H}}^{(r+1)} \cdots \hat{\boldsymbol{H}}^{(m)}\mathbf{e} \text{ is } \sum_{j=1}^n (H_{k,j}\eta_j^{(r+1)})^2 / \underline{P}_{k,j}^r$$

and this value is minimized because $\underline{\boldsymbol{P}}^{(r)}$ is constructed as

$$\underline{P}_{i,j}^{(r)} = \eta_j^{(r+1)} |H_{i,j}| / \sum_{k=1}^n \eta_k^{(r+1)} |H_{i,k}|. \tag{10}$$

So no elements of the vector $\hat{\boldsymbol{H}}^{(r)}\hat{\boldsymbol{H}}^{(r+1)}\cdots\hat{\boldsymbol{H}}^{(m)}\mathbf{e}$ will decrease in value and neither will norm $\|\tilde{\boldsymbol{H}}\|$ if we change $\underline{\boldsymbol{P}}^{(r)}$. From formula (10) we can compute the kth element of $\hat{\boldsymbol{H}}^{(r)}\hat{\boldsymbol{H}}^{(r+1)}\cdots\hat{\boldsymbol{H}}^{(m)}\mathbf{e}$ as $(\sum_{j=1}^{n}\eta_j^{(r+1)}|H_{k,j}|)^2 = (\eta_k^{(r)})^2$. So we have proved that this induction statement also holds for $\ell = r$ and we are done. □

Remark. The standard 1-way method can also be viewed as a special case of the m-way random walk setting, with the m transition matrices being the same. However the 1-way method generally does not minimize $\|\tilde{\boldsymbol{H}}\|$ in the m-way setting according to Algorithm 2. Formula (6) from Theorem 2 indicates the connection between the variance and the power series of $\tilde{\boldsymbol{H}}$. Since $\|\tilde{\boldsymbol{H}}\|$ is an upper bound of $\rho(\tilde{\boldsymbol{H}})$ and $\rho(\tilde{\boldsymbol{H}})$ affects how big this power series will grow, we can see that the m-way random walk with transition hypermatrix defined from Algorithm 2 has the tendency to decrease the variance compared to the standard 1-way method. Although the above analysis does not ensure a smaller variance for the m-way method, numerical experiments in both synthetic matrices and matrices in real applications support this conjecture (see section 5).

Example. We now illustrate a simple, concrete, example of how the multiway walks improve the simulation procedure. Consider a problem where

$$\boldsymbol{H} = \begin{bmatrix} 0.85 & 0.4 \\ 0.2 & 0 \end{bmatrix}.$$

The transition matrix generated by the standard method is

$$\boldsymbol{P} = \begin{bmatrix} 0.68 & 0.32 \\ 1 & 0 \end{bmatrix}.$$

The two transition matrices generated by the 2-way method are

$$\underline{\boldsymbol{P}}^{(1)} = \begin{bmatrix} 0.93 & 0.07 \\ 1 & 0 \end{bmatrix}, \qquad \underline{\boldsymbol{P}}^{(2)} = \begin{bmatrix} 0.68 & 0.32 \\ 1 & 0 \end{bmatrix}.$$

The resulting $\tilde{\boldsymbol{H}}$ are

$$\text{the standard method } \begin{bmatrix} 1.04 & 0.50 \\ 0.04 & 0 \end{bmatrix} \text{ with spectral radius 1.060;}$$

$$\text{the 2-way method } \begin{bmatrix} 0.88 & 0.38 \\ 0.04 & 0.02 \end{bmatrix} \text{ with spectral radius 0.899.}$$

Since the probability values appear as divisors when computing $\tilde{\boldsymbol{H}}$, without being able to assign a high enough value to the first entry in the transition matrix makes the entry in $\tilde{\boldsymbol{H}}$ grow to 1.04, which causes the spectral radius to be bigger than one. In other words, this effect accumulates and amplifies over steps causing the variance of the weights goes to infinity.

The transition matrices from the standard MC are row independent (they don't utilize the information among rows), but transition matrices in multiway MC are row dependent (the values in one row can affect the transitions in another row). In this way the multiway MC provides a more global view on how to assign the probabilities of the transition matrices. In this example, the standard method assigns the probability values of each entry purely based on the corresponding row of $\boldsymbol{H}$ (i.e., proportional), while the multiway method also considers the information from other rows. For example, both rows have the highest value in index 1, so it means that we should assign high probabilities to index 1 for both rows. However the second row indicates that the probability for index 1 should be much higher than index 2. Since the standard method cannot utilize such a hint, it only assigns 0.68 to the corresponding entry. The 2-way method adjusts it to 0.93 once that information appears.

4.2. Guaranteeing convergence. Recall that in order to have $\rho(\tilde{\boldsymbol{H}}) < 1$, a sufficient condition for the standard 1-way method is to have $\|\boldsymbol{H}\| < 1$. As we previously mentioned, others have found this is not always easy to accomplish.

For a multiway random walk, to bound the spectral radius, a sufficient condition is to have $\|\tilde{\boldsymbol{H}}\| < 1$. We now show that this can be guaranteed in a more general setting than for the standard 1-way method.

THEOREM 7. *Let $\boldsymbol{H}^+$ denote the nonnegative matrix, where $H^+_{i,j} = |H_{i,j}|$. There exists an m-way Markov random walk transition hypermatrix $\underline{\boldsymbol{P}}$ such that $\|\tilde{\boldsymbol{H}}\| < 1$ if and only if $\rho(\boldsymbol{H}) \le \rho(\boldsymbol{H}^+) < 1$.*

Proof. First, note that $\rho(\boldsymbol{H}) \le \rho(\boldsymbol{H}^+)$. (We were unable to rapidly find a reference for this fact, but the proof follows quickly when $\mathbf{v}$ is the eigenvector corresponding to the spectral radius of $\boldsymbol{H}$, so $\rho(\boldsymbol{H}) \le \min_{i \text{ with } v_i \neq 0} \sum_j |H_{ij}||v_j|/|v_i|$ from simple absolute value properties. Then $|\mathbf{v}|$ is a feasible vector for the Collatz–Wielandt variational characterization of the spectral radius $\rho(\boldsymbol{H}^+)$ from Horn and Johnson (2012, Corollary 8.3.3).) If there exists an m-way Markov random walk transition hypermatrix $\underline{\boldsymbol{P}}$ such that $\|\tilde{\boldsymbol{H}}\| < 1$, without a loss of generality we assume $\underline{\boldsymbol{P}}$ is the output from Algorithm 2 since Theorem 6 states that it minimizes $\|\tilde{\boldsymbol{H}}\|$. From the proof of Theorem 6 we have

$$\|\tilde{\boldsymbol{H}}\| = \|\tilde{\boldsymbol{H}}\mathbf{e}\| = \|\hat{\boldsymbol{H}}^{(1)}\hat{\boldsymbol{H}}^{(2)}\cdots\hat{\boldsymbol{H}}^{(m)}\mathbf{e}\| = \|\big((\eta_1^{(1)})^2, (\eta_2^{(1)})^2, \dots, (\eta_n^{(1)})^2\big)^T\|. \tag{11}$$

According to the constructing procedure of Algorithm 2 we have

$$(\eta_1^{(\ell)}, \eta_2^{(\ell)}, \dots, \eta_n^{(\ell)})^T = (\boldsymbol{H}^+)^m\mathbf{e},$$

so that

$$\|\tilde{\boldsymbol{H}}\| < 1 \Longrightarrow \|(\boldsymbol{H}^+)^m\mathbf{e}\| < 1 \Longrightarrow \|(\boldsymbol{H}^+)^m\| < 1 \Longrightarrow \rho(\boldsymbol{H}^+) < 1.$$

If we have $\rho(\boldsymbol{H}^+) < 1$, from Gelfand's formula, we have

$$\rho(\boldsymbol{H}^+) = \lim_{k\to\infty} \|(\boldsymbol{H}^+)^k\|^{1/k}.$$

Then we can find a sufficiently large number m such that for any $k \ge m$ the inequality $\|(\boldsymbol{H}^+)^k\|^{1/k} < 1$ holds. Let $\tilde{\boldsymbol{H}}$ be the matrix based on the transition hypermatrix output from Algorithm 2. Based on the observation of (11), we have

$$\rho(\boldsymbol{H}^+) < 1 \Longrightarrow \|(\boldsymbol{H}^+)^m\| < 1 \Longrightarrow \eta_i^{(1)} < 1, i = 1, 2, \dots, n \Longrightarrow \|\tilde{\boldsymbol{H}}\| < 1. \qquad \square$$

Theorem 7 creates an equivalent link between $\rho(\boldsymbol{H}^+) < 1$ and the existence of an m-way Markov random walk such that $\|\tilde{\boldsymbol{H}}\| < 1$. We note that $\rho(\boldsymbol{H}^+) < 1$ is a strictly weaker condition than $\|\boldsymbol{H}\| < 1$, which means it is one step closer to the condition $\rho(\boldsymbol{H}) < 1$ that Neumann series converges to given the conclusion of $\rho(\boldsymbol{H}) \le \rho(\boldsymbol{H}^+)$. Also it is worth noting Theorem 7 does not guarantee the size of m. In other words, one can always cook up some example matrix $\boldsymbol{H}$ with $\rho(\boldsymbol{H}^+) < 1$ but that will require m to be arbitrarily large. Although these extreme cases are not our primary focus in this paper, we point it out for the discussion of the practical implementation of Algorithm 2. In order to find the transition hypermatrix $\underline{\boldsymbol{P}}$ with $\|\tilde{\boldsymbol{H}}\| < 1$, we can set a threshold number $\phi_{\max}$, and let m grow until we have $\eta_i < 1$ for all $i = 1, 2, \dots, n$ or $m = \phi_{\max}$. Note that the condition of $\eta_i < 1$ for all $i = 1, 2, \dots, n$ directly reveals that $\|\tilde{\boldsymbol{H}}\| < 1$. If this condition has not been satisfied when $m = \phi_{\max}$, then this problem cannot have an m-way random walk with m up to $\phi_{\max}$ to ensure the convergence of the MC simulation. The value of $\phi_{\max}$ is a trade-off between how much computational effort to spend before and during the random walk simulation. As stated before, we do not need to rerun the algorithm for different values of m, because the way Algorithm 2 computes the transition hypermatrix is compatible with different values of m.

4.3. Random walk error analysis. To practically estimate the value $\langle \mathbf{h}, \mathbf{x} \rangle$ from simulating the random variable Z, we need to truncate the multiway Markov random walk in order for it to end after some large number of steps N. To do this, we wish to show that the estimator weight W_N will go to zero as the walk continues, in which case, a practical truncation procedure (Dimov, Dimov, and Gurov, 1998; Benzi et al., 2017) to determine N is through the criterion: $|W_N| \le \varepsilon |W_0|$, where $\varepsilon > 0$ denotes some small number. An alternative for solving a linear system is to fix N upfront and use an SMC procedure, as we do in the next section.

The value $|W_N|$ is a random variable and, through a similar analysis in Theorem 1, we have the following result regarding its expectation and variance.

COROLLARY 8. *Let W_ℓ denote the weight of the m-way random walk after ℓ steps. Then* $\mathrm{E}[|W_\ell|] = \langle \mathbf{h}^+, (\boldsymbol{H}^+)^\ell \mathbf{e} \rangle$ *and* $\mathrm{Var}[|W_\ell|] = \langle \hat{\mathbf{h}}, \tilde{\boldsymbol{H}}^{\ell/m} \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(\ell\% m)} \mathbf{e} \rangle - (\langle \mathbf{h}^+, (\boldsymbol{H}^+)^\ell \mathbf{e} \rangle)^2$*, where we use ℓ/m and $\ell\%m$ to denote the quotient and the remainder.*

Proof. First it is easy to see the expectation $\mathrm{E}[|W_\ell|] = \langle \mathbf{h}^+, (\boldsymbol{H}^+)^\ell \mathbf{e} \rangle$ with a similar analysis from the proof of Theorem 1. We then compute the variance: $\mathrm{Var}[|W_\ell|] = \mathrm{E}[W_\ell^2] - (\mathrm{E}[|W_\ell|])^2$.

For the first part of the variance,

$$
\begin{aligned}
\mathrm{E}[W_\ell^2] &= \sum_{k_0} \sum_{k_1} \cdots \sum_{k_\ell} \left(\frac{h_{k_0} H_{k_0,k_1} H_{k_1,k_2} \cdots H_{k_{\ell-1},k_\ell}}{p_{k_0} \underline{P}^{(1)}_{k_0,k_1} \underline{P}^{(2)}_{k_1,k_2} \cdots \underline{P}^{(\ell)}_{k_{\ell-1},k_\ell}} \right)^2 p_{k_0} \underline{P}^{(1)}_{k_0,k_1} \underline{P}^{(2)}_{k_1,k_2} \cdots \underline{P}^{(\ell)}_{k_{\ell-1},k_\ell} \\
&= \sum_{k_0} \sum_{k_1} \cdots \sum_{k_\ell} \hat{h}_{k_0} \hat{H}^{(1)}_{k_0,k_1} \hat{H}^{(2)}_{k_1,k_2} \cdots \hat{H}^{(\ell)}_{k_{\ell-1},k_\ell} \\
&= \langle \hat{\mathbf{h}}, \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(\ell)} \mathbf{e} \rangle \\
&= \langle \hat{\mathbf{h}}, \tilde{\boldsymbol{H}}^{\ell/m} \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(\ell\%m)} \mathbf{e} \rangle.
\end{aligned}
$$

The last step of the derivation uses the periodicity of the transition matrices. So the variance $\mathrm{Var}[|W_\ell|] = \langle \hat{\mathbf{h}}, \tilde{\boldsymbol{H}}^{\ell/m} \hat{\boldsymbol{H}}^{(1)} \hat{\boldsymbol{H}}^{(2)} \cdots \hat{\boldsymbol{H}}^{(\ell\%m)} \mathbf{e} \rangle - (\langle \mathbf{h}^+, (\boldsymbol{H}^+)^\ell \mathbf{e} \rangle)^2$. □

This result shows that the expectation of $|W_\ell|$ grows with the matrix $\boldsymbol{H}^+$, whereas the variance grows with the matrix $\tilde{\boldsymbol{H}}$. And they can both either shrink to zero or expand to infinity. Here, we find that $\rho(\boldsymbol{H}^+) < 1$ is a necessary condition to have a finite truncation number N, in which case the weight goes to zero in expectation. In fact, this is a necessary requirement for any MC procedure (even $m = 1$) to be able to truncate the walk with a finite N. This setting nicely matches our m-way walk as $\rho(\boldsymbol{H}^+) < 1$ was a sufficient condition for that procedure to have a finite variance (see Theorem 7).

5. Numerical experiments.

In this section, we conduct several experiments to study the behaviors of the MC methods on linear systems. Codes to reproduce these experiments are available at https://github.com/wutao27/multi-way-MC.

- In section 5.1, we compute the variance of the MC methods derived from Theorem 2, to see how our multiway methods would speed up computation.
- In section 5.2, we run the actual MC simulations and show that the variance obtained behaves as Theorem 2 indicates.
- In section 5.3, we apply both the forward method and the adjoint method to solve linear systems. And we compare the error versus number of simulations for the standard method and our multiway methods.
- In section 5.4, we conduct a case study on one problem to demonstrate that a multiway MC method and SMC can, in principle, out-perform existing solvers in parallel environments.

Throughout the experiment section, we use n to denote the size of the solution vector, m to denote the number of ways in the multiway setting, and N to denote the total number of simulations.

5.1. Analysis of variance based on theory.

In this section we analyze the methods in terms of their variances computed from Theorem 2. We look into the two aspects of the variance:

- The MC simulation will only work when the variance is bounded. We check whether the sufficient condition $\|\tilde{\boldsymbol{H}}\| < 1$ holds for specific values of m in a large number of synthetic problems to determine the fraction of solvable problems at a specific m.
- Smaller variance should indicate faster convergence for the MC simulation. We compare the speed improvement from the standard 1-way method to our multiway methods based on the theoretical variances.

Solvable problems. A problem is solvable when the condition $\|\tilde{\boldsymbol{H}}\| < 1$ is satisfied in light of the value of m, which is a sufficient condition that guarantees the convergence of the MC simulations. We use synthetic data where each $\boldsymbol{H}$ is generated as a 1000 by 1000 sparse random matrix with 20% of its entries being nonzeros, and each nonzero is a random number following uniform distribution between $(0, 1)$. These synthetic matrices are rescaled so that the spectral radius $\rho(\boldsymbol{H}^+)$ is as prescribed during the experiments. Formally to get a spectral radius $0 < r < 1$ of $\rho(\boldsymbol{H}^+)$, we scale $\boldsymbol{H} \leftarrow r\boldsymbol{H}/\rho(\boldsymbol{H}^+)$. Given our theory, there exists a value of m such that all the problems we study are solvable. Figure 2 shows the ratio of solvable problems, which is defined as the percentage of synthetic random problems that are solvable at a particular value of m. Each result is the average over 100 trials.

As we can see in Figure 2 when $\rho(\boldsymbol{H}^+)$ increases, the problems become harder to solve and the solvable ratio drops for each method. However our multiway methods are much more robust and can handle the cases of large $\rho(\boldsymbol{H}^+)$. To see that, the standard method can hardly guarantee any solution convergence when $\rho(\boldsymbol{H}^+) \geq 0.85$. As m increases, our method can guarantee to solve more problems, and there is a big improvement even when switching from the standard method to the 2-way method.

Speedup times. Here we consider the same type of synthetic problems as well as real world matrices. For real world matrices, we focus on the Harwell-Boeing (HB) sparse matrix collection (Duff, Grimes, and Lewis, 1992; Davis and Hu, 2011). The

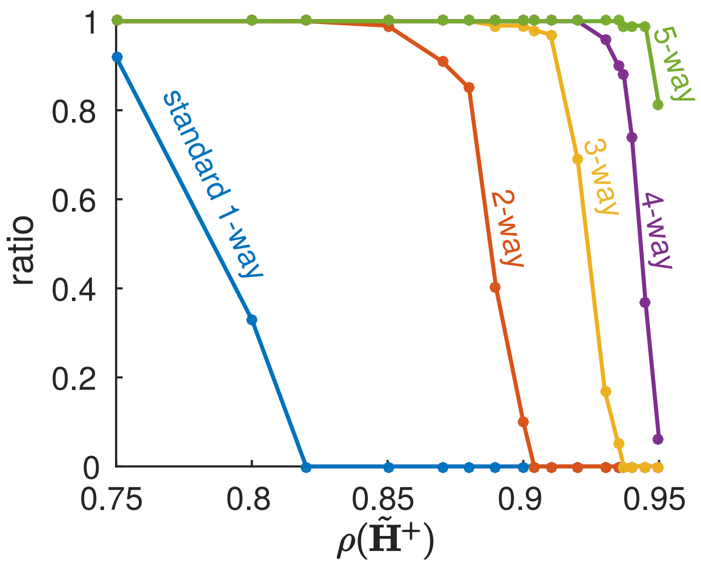

FIG. 2. *The results of the standard* 1*-way method and our multiway methods with* $m = 2, 3, 4, 5$ *for the ratio of solvable synthetic problems versus the spectral radius* $\rho(\boldsymbol{H}^{+})$*.*

matrix $\boldsymbol{H}$ is constructed by a simple left diagonal precondition operation on the original matrix $\boldsymbol{A}$ from the collection: $\boldsymbol{H} = \boldsymbol{I} - \mathrm{Diag}(\boldsymbol{A})^{-1}\boldsymbol{A}$. And for the test problems we only consider the matrices that have $\rho(\boldsymbol{H}^{+}) < 1$. In the interest of simplicity, we only use problems with fewer than 5000 dimensions. We exclude the problems where MATLAB gives a warning about the matrix being nearly singular, as therefore it cannot accurately calculate the variance. Also we do not consider problems with equal row sums in $\boldsymbol{H}$ since the multiway method would be the same as the standard method. There are 20 matrices from the Harwell-Boeing collection in this experiment that meet this criteria. There are 6 problems where the sufficient condition for the standard method is violated but the condition for our multiway methods is satisfied, and 3 of them can be confirmed to converge with $m \leq 5$. They are matrices *fs_*760_1*, jpwh_*991, and *nos*7.

In Table 1, we study theoretical speedup times based on variance. In this case, speedup is defined as $\mathrm{Var}[X]/\,\mathrm{Var}[Z]$, where X and Z denote the random variables from the standard 1-way method and our method, respectively. It is an indicator of how many times faster our multiway methods could be compared to the standard 1-way method. We apply the conclusion in Theorem 2 to compute $\mathrm{Var}[X]$ and $\mathrm{Var}[Z]$ for all the testing methods. We can see that the speedup times increase as m increases; therefore, allowing the random walk to transition among multiple matrices helps with the performance. We also notice that as the problem becomes harder (i.e., $\rho(\boldsymbol{H}^{+})$ gets closer to 1), the improvements become more significant.

Among our testing problems, there is only one outlier (http://www.cise.ufl.edu/research/sparse/matrices/HB/fs_760_1.html) where our multiway methods have a larger variance than that from the standard method. For this problem, the matrix $\boldsymbol{H}$ is almost outside our assumptions in this paper. We assume that $\boldsymbol{H}$ does not have zero rows in order to assign transition probabilities for each state. For this testing problem $\boldsymbol{H}$, the row sums of $\boldsymbol{H}^{+}$ distribute in a drastic way. Over half of the rows have sum values between 10^{-17} to 10^{-6}, and quite a few "big" rows have sums larger

Table 1
The speedup times by our multiway methods with $m = 2, 3, 4, 5$ compared to the standard 1-way method on synthetic problems and the Harwell-Boeing collection. r denotes $\rho(\boldsymbol{H}^+)$.

	2-way	3-way	4-way	5-way
$\rho(\boldsymbol{H}^+)$	Synthetic matrices			
0.8	1.09	1.13	1.14	1.15
0.9	1.38	1.58	1.69	1.77
0.95	1.75	2.30	2.73	3.06
0.99	2.40	3.77	5.10	6.39
	Harwell-Boeing collection			
	1.20	1.44	1.59	1.74

Table 2
The variances computed from Theorem 2 for problems $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$.

	1-way	2-way	3-way	4-way	5-way
$\boldsymbol{H}_1$	1.645	0.6526	0.4654	0.3960	0.3599
$\boldsymbol{H}_2$	∞	3.771	1.446	0.9764	0.7768

than 10^3. So this matrix has many rows that are nearly zero. We did not spend time trying to customize our methods to handle this particular case. Note also, that this case fails the sufficient condition for the standard method to converge as discussed above, but when we compute the variance explicitly, we find it is finite. For all the other testing problems, our multiway methods can achieve smaller variances than the standard method, and the speedup times are shown in Table 1.

5.2. Analysis of variance based on simulations.

In this section we implement the actual MC simulations based on the standard and the multiway random walk procedures. Since the theoretical conclusion regarding the variance (i.e., Theorem 2) gives a computable expression, we want to validate it by comparing the variances obtained from theory and simulations.

Consider the following two problems with matrix $\boldsymbol{H}$ defined as

$$\boldsymbol{H}_1 = \begin{bmatrix} 0.75 & 0.4 \\ 0.2 & 0 \end{bmatrix}, \qquad \boldsymbol{H}_2 = \begin{bmatrix} 0.85 & 0.4 \\ 0.2 & 0 \end{bmatrix}.$$

The problem H_2 was also studied in section 4.1.

These two problems look very similar; however, according to Theorem 2 the standard method only manages to solve the problem with $\boldsymbol{H}_1$, and the variance for the problem $\boldsymbol{H}_2$ is infinity (see Table 2). Note that, for simplicity, we use the all ones vector $\mathbf{e}$ for the value of $\mathbf{b}$ and $\mathbf{h}$. And we scale $\mathbf{h}$ to make $E[Z] = 1$, which helps provide a unified scaling in terms of variances.

Then we run the MC simulations and calculate the empirical variances. The simulation does 10^8 random walks with the truncation value set to $N = 100$ for problem $\boldsymbol{H}_1$ and $N = 200$ for problem $\boldsymbol{H}_2$. Figure 3 shows the variance results for problems $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$. As we can see, the variances increase then converge to the same value as theoretically computed. For $\boldsymbol{H}_2$, we plot the empirical variance of the 1-way walk in Figure 4. The variance for the standard method surges up to 1600, which is far larger than the variances (bounded by 5) from the multiway methods. We can conclude that the variance for the standard method diverges, even though the variance looks like it con-

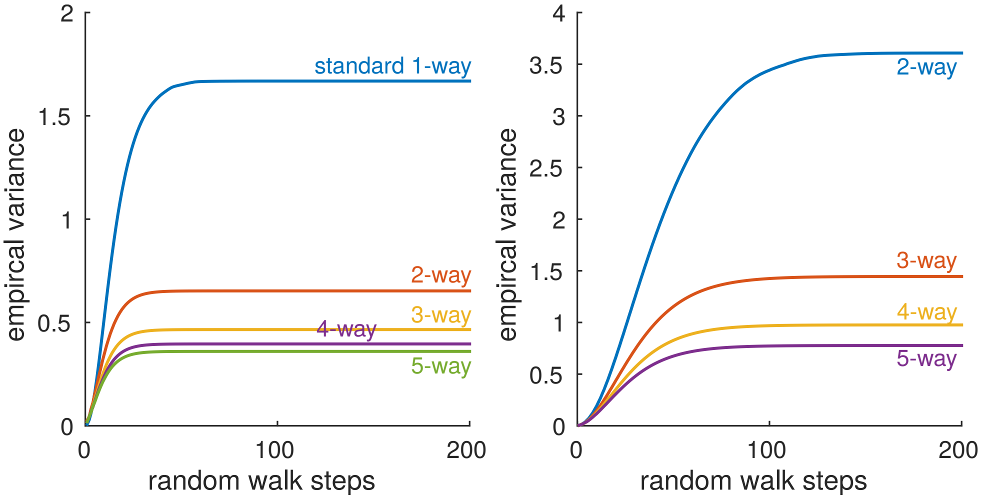

FIG. 3. *The empirical variances obtained from the MC simulations for different methods on problems* $\boldsymbol{H}_1$ *(left) and* $\boldsymbol{H}_2$ *(right).*

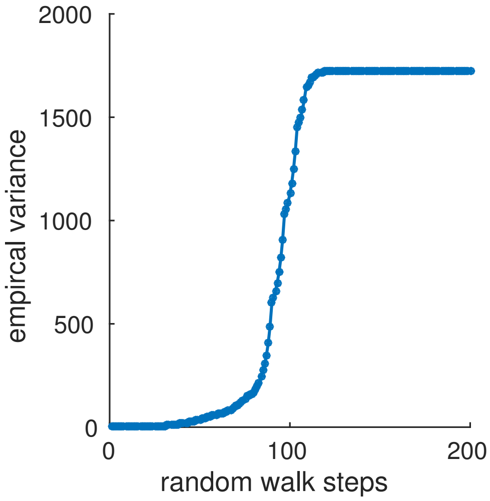

FIG. 4. *The empirical variance obtained from the MC simulations for the standard* 1*-way method on problem* $\boldsymbol{H}_2$*.*

verges to some value around 1700. The reason is that we cannot accurately estimate an infinite value by simulations. In fact when we run multiple trials, we get different variances (from several hundred to several thousand) for this case. On the other hand, the multiway methods shown in Figure 3, converge nicely and reliably as the number of steps grows. These experiments support and illustrate our theoretical findings.

5.3. Analysis of solvers based on simulations. In this section, we run the MC simulations on solving the linear system with both the forward method and the adjoint method. We compare the performances of the standard 1-way method and the multiway (i.e., $m = 5$) method by evaluating the relative errors. Then we also calculate the speedup times from the simulations to check if they are consistent with

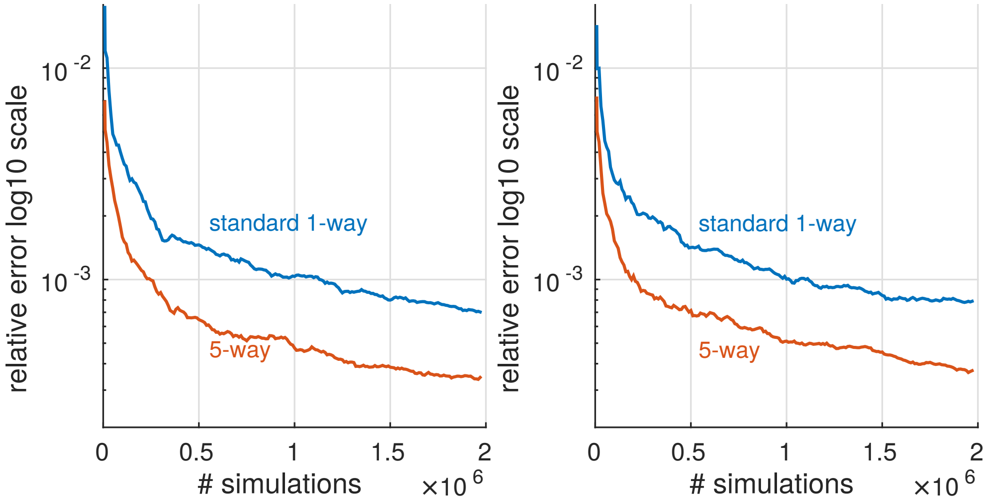

Fig. 5. *The relative error (average over* 100 *trials) obtained from the MC simulations for the standard* 1*-way method and the multiway method with* $m = 5$ *on problem* $\boldsymbol{H}_1$*. The result for the forward method is on the left, and the result for the adjoint method is on the right.*

the ones we compute from variances. For the forward method, the target problem is $\langle \mathbf{h}, \mathbf{x} \rangle$ such that $\mathbf{x} = \boldsymbol{H}\mathbf{x} + \mathbf{b}$, where both $\mathbf{h}$ and $\mathbf{b}$ are the vector of all ones. For the adjoint method, the target problem is to solve $\mathbf{x}$ from $\mathbf{x} = \boldsymbol{H}^T\mathbf{x} + \mathbf{b}$. Note that we use $\boldsymbol{H}^T$ instead of $\boldsymbol{H}$ to make the adjoint method consistent with the forward method in terms of the transition matrix.

In order to show that our multiway methods can speed up the convergences the same way as theoretically predicted, we first consider the problem $\boldsymbol{H}_1$ from section 5.2. The speedup times are calculated as

$$\frac{\mathrm{Var}[X]}{\mathrm{Var}[Z]} = \frac{1.645}{0.3599} = 4.57$$

when we compare the multiway ($m = 5$) method against the standard one. For the experiment setting, we apply a truncation value of $N = 100$ for the length of each random walk. We study the convergence of the relative error up to 2 million simulations. Since the MC simulations can produce stochastic volatility, in order to make the results more robust and smooth, we report the average results over 100 trials.

Figure 5 shows both the relative errors drop when the number of simulations increases. However the convergence from the multiway method is much faster, and this observation holds for both the forward method and the adjoint method with similar scale. For instance, when we fix the relative error as 10^{-3}, at the forward method setting, it takes around 240,000 simulations for the multiway method and 1,140,000 simulations for the standard method. Therefore the speedup times are around 4.75 which matches our expectation. A similar analysis holds for the adjoint method as well.

Next we choose a matrix (`jpwh_991`) from the Harwell-Boeing collection to conduct the same error analysis. It is a 991 by 991 matrix with 6027 nonzeros. Note that we do not choose the matrix with the best speedup performance. We choose this matrix as it represents the average speedup performance of the Harwell-Boeing collection. The theoretical speedup times calculated from the variances is 1.88 for the

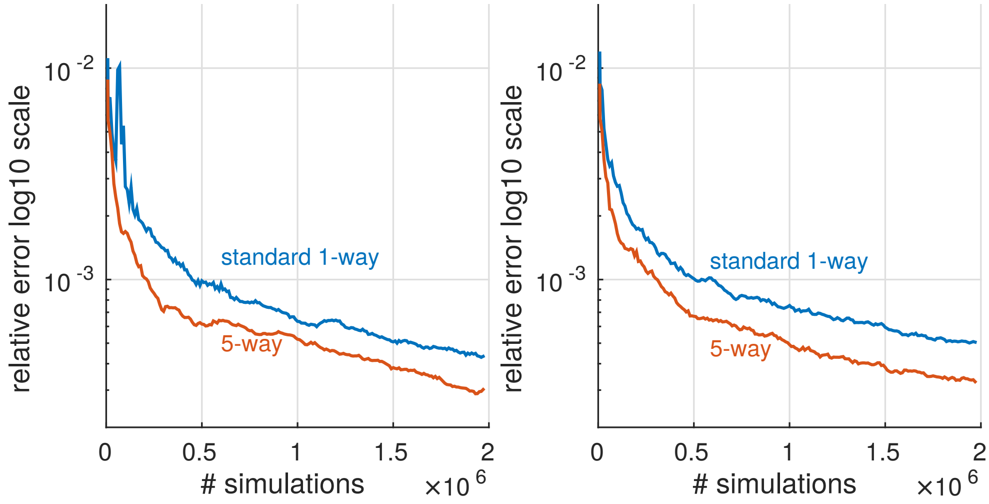

Fig. 6. *The relative error (average over* 100 *trials) obtained from the MC simulations for the standard* 1*-way method and the multiway method with* $m = 5$ *on the problem from the Harwell-Boeing collection. The result for the forward method is on the left, and the result for the adjoint method is on the right.*

multiway ($m = 5$) method. In this experiment, the truncation value is $N = 1000$, and all the other settings remain the same. The results are shown in Figure 6. As we can see from the figures, it takes approximately 2 times the number of simulations for the standard method in order to reach the same relative error with the multiway method.

5.4. Solving to high accuracy, a case study on parallelism in `jpwh_991`. We now wish to understand how our results might be applied to solving a linear system to a higher accuracy. This involves choosing a forward or adjoint solution method as a low-accuracy solver and then using the SMC framework to iteratively achieve high accuracy. Our particular goal here is to understand if our solver could, in principle, be parallelized such that it is faster than a Richardson or GMRES solver on the same system. Thus, we use the early synthetic results as data for a parallel performance model that shows we should be able to outperform a parallel GMRES and Richardson implementation with enough processors.

We use the same real world matrix from section 5.3, `jpwh_991` from Harwell-Boeing. The test problem is $\boldsymbol{A}\mathbf{x} = \mathbf{b}$, where we set $\mathbf{b} = \mathbf{e}$ as the vector of all ones. The problems studied in these experiments are as follows:

$$\begin{aligned} &\text{left diagonal preconditioner:} \quad \boldsymbol{H} = \boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{A}, \\ &\text{right diagonal preconditioner:} \quad \boldsymbol{H} = \boldsymbol{I} - \boldsymbol{A}\boldsymbol{D}^{-1}. \end{aligned}$$

The matrix is 846 by 846 with 4716 nonzeros after preprocessing of deleting empty rows and columns induced by the transformation (there are some rows with only diagonal entries).

Forward versus adjoint solvers. From the introduction in section 2, we know that the forward method can also be applied to get the entire solution vector $\mathbf{x}$ from the linear system $\mathbf{x} = \boldsymbol{H}\mathbf{x} + \mathbf{b}$ if we run the simulation independently for each individual basis vector $\mathbf{e}_i$. Alternatively we can apply the adjoint method, where each random walk simulation will be able to update every element of the solution vector. We

Table 3

The number of simulations needed to reach a relative error of around 0.05*. Each experiment is repeated* 100 *times, and the average relative error is reported. The matrix is* 846 *by* 846 *with* 4716 *nonzeros after preprocessing of deleting empty rows and columns induced by the transformation (there are some rows with only diagonal entries).*

Method	# simulations	Relative error
Forward and left precondition	126,900 = 150 × 846	0.0493
Adjoint and left precondition	465,000	0.0515
Forward and right precondition	16,074,000 = 19,000 × 846	0.0511
Adjoint and right precondition	7000	0.0500

first conduct an experiment to compare these two different approaches to understand which one to study further on this problem.

We notice that there are mainly two differences between these two approaches:

- The weight W for each random walk is defined based on $\boldsymbol{H}$ for the forward method and $\boldsymbol{H}^T$ for the adjoint method. Thus they can have different variances.
- Each random walk simulation for the forward method only updates one element from the solution vector $\mathbf{x}$, while the adjoint method allows each random walk to update all elements of the vector. So intuitively the adjoint approach is a more efficient approach. However we cannot simply conclude that the adjoint method should be n times more efficient than the forward method, because allowing each random walk updating multiple elements introduces covariance into the solution.

We compare the total number of simulations required in order to reach a relative error of 0.05 on average for the forward method with a left diagonal preconditioner, the forward method with a right diagonal preconditioner, the adjoint method with a left diagonal preconditioner, and the adjoint method with a right diagonal preconditioner. We conduct experiments for all the above methods with the standard 1-way setting. The truncation size is set at $N = 200$.

Table 3 shows the experimental results. We find that when applying the left preconditioner, the adjoint method is actually more expensive. We believe the reason is that left diagonal preconditioning has the effect of normalizing rows of $\boldsymbol{H}$, which works well when the forward method constructs the transition matrix based on the rows of $\boldsymbol{H}$. The adjoint method constructs the transition probabilities based on the row of $\boldsymbol{H}^T$ (i.e., columns of $\boldsymbol{H}$), therefore, it does not work very well given the left diagonal preconditioner. When applying the right diagonal preconditioner, we find the adjoint method is much more efficient, while the forward method becomes quite expensive. To summarize, by comparing the adjoint method (with the right precondition) to the forward method (with the left precondition), we find the adjoint method is around 18 times faster, which is not as trivial as $n = 846$ times faster.

SMC analysis. Next, we study how the SMC method works in practice compared to the Richardson iteration and GMRES. We apply the 5-way MC simulation for each inner iteration. Table 4 shows the results as we vary the random walk length and the number of simulations.

First we notice that for the SMC method, when the number of simulations per iteration increases, the total number of iterations decreases as a result of more accurate approximation of the direct MC within the iteration. And it becomes much more expensive to further decrease the number of iterations. Another interesting result

TABLE 4

Results of the three methods: Richardson iteration, GMRES, and SMC on the linear system with a target relative residual of 10^{-8}. *We use the* 5*-way MC simulation for each inner iteration. We run multiple versions of SMC varying the total number of iterations and the number of simulations per iteration. The random walk length is chosen as the best length given the number of simulations per iteration. We note that when the number of simulations is small, the random walk length also tends to be small as it cannot accurately handle the larger length.*

Methods	Random walk length	# simulations per iteration	# iterations	Relative residual
Richardson	-	-	891	9.92×10^{-9}
GMRES	-	-	55	6.41×10^{-9}
SMC	2	800	462	9.53×10^{-9}
SMC	6	2500	159	9.83×10^{-9}
SMC	10	5000	95	9.83×10^{-9}
SMC	30	25,000	33	5.54×10^{-9}
SMC	50	50,000	23	9.51×10^{-9}
SMC	120	500,000	11	2.02×10^{-9}

is that we notice for the first two versions of SMC, the number of simulations per iteration is even far smaller than the total number of nonzeros of the matrix, which means that the MC simulation can approximate the solution with a moderate accuracy without the need of accessing all the elements of the matrix, although, we used all that information to build the preconditioners and the transition probabilities.

Evaluation of parallelization potential. We now seek to approximate how well an idealized parallel computation would perform. In terms of effort, a Richardson iteration uses the number of nonzeros (i.e., nnz) floating point operations (FLOPs) per iteration, as it basically computes a sparse mat-vec product. For GMRES, at the kth iteration, it takes a sparse mat-vec product $O(\text{nnz})$ as well as an Arnoldi iteration which costs $O(kn)$ FLOPs. For SMC, each random sample costs constant time (Walker, 1974) (note, this is based on a simple pseudorandom number generator), therefore each iteration costs $O(NL{+}\text{nnz})$, where N is the total number of simulations and L is the length of the random walk. We compute the total FLOPs for each method, and split them to parallelizable and nonparallelizable. Then we can theoretically model the runtime as we vary the number of processors and communication costs. The performance model we use is to

- assume the cost of a single FLOP is 1 and
- assume the number of processors is p and the cost of the communications and synchronizations is σ, then the cost of one parallel computation for t FLOPs is $t/p + \sigma$ for $p > 1$ and t for $p = 1$.

First we describe a set of operations we consider parallelizable. The mat-vec production is parallelizable for all three methods. The Arnoldi iteration in GMRES is parallelizable. And MC simulation in SMC is parallelizable. In theory, vector operations such as addition, multiplication, or computing the norm are also parallelizable, however, we do not count the work involved in vector operations as it can often be combined with other parallel work or be implemented highly efficiently. Consequently, we only consider matrix operations. Let `niter` denote the total number of iterations (note that it is different for each method).

- Runtime of GMRES: $\sum_{k=1}^{\texttt{niter}} \big(2{*}\text{nnz}/p{+}\sigma{+}4{*}n{*}\text{ceil}(k/\min(k,p)){+}\sigma\big){+}k^2{+}2nk/p$, where ceil is the function that outputs the greatest integer less than or equal to the input number. Among these values, $2 * \text{nnz}/p + \sigma$ denotes the cost of a parallel mat-vec product. The summation in the middle denotes the

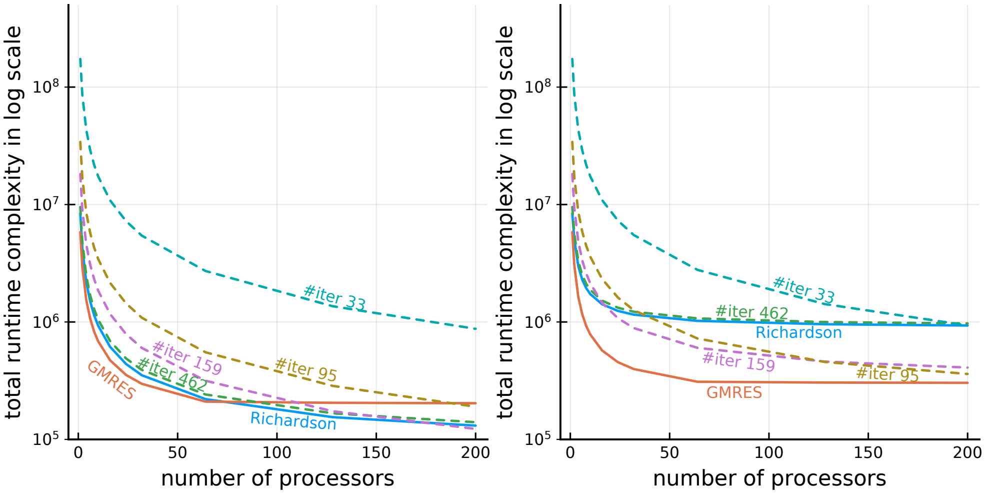

FIG. 7. *Runtime complexity comparison for different methods with different numbers of processors in the parallel setting. The SMC results are the dashed lines reflect the random walk lengths from Table* 4. *The runtime complexity is in log scale. The communications and synchronizations costs are assumed to be* 100 *times a FLOP for the figure in the* ***left*** *and* 1000 *times a FLOP for the figure in the* ***right****.*

cost of the Arnoldi function at iteration k, where the ceil function arises by parallelizing over columns which cannot exceed k Arnoldi iterations. Finally, $k^2 + 2nk/p$ is the cost of recovering the solution vector. Again, we drop the cost of Givens rotation for simplicity.

- Runtime of Richardson: $\texttt{niter} * (2 * \mathrm{nnz}/p + \sigma)$.
- Runtime of SMC: $\texttt{niter}*(2*\mathrm{nnz}/p+7*N*L/p+2*\sigma)$. Among these values, $2 * \mathrm{nnz}/p + \sigma$ denotes the cost of a parallel mat-vec product. $7 * N * L/p + \sigma$ denotes the cost for one iteration of a full MC (i.e., with N simulations of random walks of length L). The number 7 is a constant representing the number of FLOPs needed for one random walk step (i.e., random sampling, computing weight, updating the random variable).

We show the cost comparisons among the three methods for $\sigma = 100$ and $\sigma = 1000$ in Figure 7. In the setting of a single processor or a few processors, GMRES is faster than SMC and the Richardson iterations. However when the number of processors increases, GMRES has less parallelizable work than the SMC methods. This trend is consistent with different values of σ. Compared with trade-offs between different SMC methods, we would favor the SMC method with fewer iterations when there are more processors. The reason is that as there are more processors for each parallel iteration, the communication cost becomes more dominant, so by decreasing the total number of outer iterations, it also decreases the total communication cost. This observation is also consistent with the fact that by increasing σ from 100 to 1000, the SMC methods with fewer iterations are affected less in terms of runtime complexity. This shows that, in principle, these methods are able to outperform a Krylov subspace method.

It is also worth noting that besides the benefit of being easily parallelized, the SMC method, due to its stochastic nature, is more robust when faults or hardware failures occur.

6. Conclusion and future work. In this paper we studied a generalization of MC methods for linear systems. The generalization allows the Markov random walk to transition using a set of matrices. We derived the variance of the resulting estimator and construct the matrices in a way to attempt to produce a finite variance. The advantages of this new random walk procedures are twofold. First it can solve more problems that the standard method fails to solve. Second our new method has the tendency to decrease the variance and thus decrease the computation needed to estimate the solution. Numerical experiments on both synthetic and real world matrices confirm the superiority of our method in the above two aspects when compared to the standard MC method. An open problem suggested by our work is to get a purely local method that avoids the global work in building the sequence of transition matrices. For future work we would like to explore the possible solutions to this problem. Another direction for future work is to study the robustness of the multiway method.

REFERENCES

V. Alexandrov, E. Atanassov, I. Dimov, S. Branford, A. Thandavan, and C. Weihrauch (2005), *Parallel hybrid Monte Carlo algorithms for matrix computations*, in International Conference on Computational Science, Part II, Lecture Notes in Comput. Sci. 3516, Springer, Berlin,. pp. 752–759.

K. Avrachenkov, N. Litvak, D. Nemirovsky, and N. Osipova (2007), *Monte Carlo methods in PageRank computation: When one iteration is sufficient*, SIAM J. Numer. Anal., 45, pp. 890–904.

H. Avron, P. Maymounkov, and S. Toledo (2010), *Blendenpik: Supercharging LAPACKS's least-squares solver*, SIAM J. Sci. Comput., 32, pp. 1217–1236, http://dx.doi.org/10.1137/090767911.

M. Benzi, T. M. Evans, S. P. Hamilton, M. Lupo Pasini, and S. R. Slattery (2017), *Analysis of Monte Carlo accelerated iterative methods for sparse linear systems*, Numer. Linear Algebra Appl., 24, e2088.

P. Brémaud (2011), *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*, Springer, New York.

T. A. Davis and Y. Hu (2011), *The University of Florida sparse matrix collection*, ACM Trans. Math. Software, 38, 1.

T. E. Davis and J. C. Principe (1993), *A Markov chain framework for the simple genetic algorithm*, Evol. Comput., 1, pp. 269–288.

S. Dietrich and I. D. Boyd (1996), *Scalar and parallel optimized implementation of the direct simulation Monte Carlo method*, J. Comput. Phys., 126, pp. 328–342.

I. Dimov, V. Alexandrov, and A. Karaivanova (2001), *Parallel resolvent Monte Carlo algorithms for linear algebra problems*, Math. Comput. Simulation, 55, pp. 25–35.

I. Dimov, T. Dimov, and T. Gurov (1998), *A new iterative Monte Carlo approach for inverse matrix problem*, J. Comput. Appl. Math., 92, pp. 15–35.

I. Dimov, S. Maire, and J. M. Sellier (2015), *A new Walk on Equations Monte Carlo method for solving systems of linear algebraic equations*, Appl. Math. Model., 39, pp. 4494–4510.

I. Dimov, B. Philippe, A. Karaivanova, and C. Weihrauch (2008), *Robustness and applicability of Markov chain Monte Carlo algorithms for eigenvalue problems*, Appl. Math. Model., 32, pp. 1511–1529.

P. Drineas, R. Kannan, and M. W. Mahoney (2006a), *Fast Monte Carlo algorithms for matrices* I: *Approximating matrix multiplication*, SIAM J. Comput., 36, pp. 132–157.

P. Drineas, R. Kannan, and M. W. Mahoney (2006b), *Fast Monte Carlo algorithms for matrices* II: *Computing a low-rank approximation to a matrix*, SIAM J. Comput., 36, pp. 158–183.

P. Drineas, R. Kannan, and M. W. Mahoney (2006c), *Fast Monte Carlo algorithms for matrices* III: *Computing a compressed approximate matrix decomposition*, SIAM J. Comput., 36, pp. 184–206.

I. S. Duff, R. G. Grimes, and J. G. Lewis (1992), *Users' Guide for the Harwell-Boeing Sparse Matrix Collection (release* 1*)*, Technical report RAL-92-086, Rutherford Appleton Laboratory, England.

T. M. Evans, S. W. Mosher, S. R. Slattery, and S. P. Hamilton (2014), *A Monte Carlo synthetic-acceleration method for solving the thermal radiation diffusion equation*, J. Comput. Phys., 258, pp. 338–358.

G. E. Forsythe and R. A. Leibler (1950), *Matrix inversion by a Monte Carlo method*, Math. Comp., 4, pp. 127–129.

N. Halko, P. G. Martinsson, and J. A. Tropp (2011), *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Rev., 53, pp. 217–288, http://dx.doi.org/10.1137/090771806.

J. H. Halton (1994), *Sequential Monte Carlo techniques for the solution of linear systems*, J. Sci. Comput., 9, pp. 213–257.

R. A. Horn and C. R. Johnson (2012), *Matrix Analysis*, Cambridge University Press, Cambridge.

H. Ji, M. Mascagni, and Y. Li (2013), *Convergence analysis of Markov chain Monte Carlo linear solvers using Ulam-von Neumann algorithm*, SIAM J. Numer. Anal., 51, pp. 2107–2122.

G. LeBeau (1999), *A parallel implementation of the direct simulation Monte Carlo method*, Comput. Methods Appl. Mech. Engrg., 174, pp. 319–337.

Y. Li and M. Mascagni (2003), *Analysis of large-scale grid-based Monte Carlo applications*, Internat. J. High Perf. Comput. Appl., 17, pp. 369–382.

K. Sabelfeld and N. Mozartova (2009), *Sparsified randomization algorithms for large systems of linear equations and a new version of the Random Walk on Boundary method*, Monte Carlo Methods Appl., 15, pp. 257–284.

S. R. Slattery (2013), *Parallel Monte Carlo Synthetic Acceleration Methods for Discrete Transport Problems*, Ph.D. thesis, University of Wisconsin Madison, Madison, WI.

S. R. Slattery, T. M. Evans, and P. P. Wilson (2015), *A spectral analysis of the domain decomposed Monte Carlo method for linear systems*, Nucl. Eng. Des., 295, pp. 632–638.

A. Srinivasan (2010), *Monte Carlo linear solvers with non-diagonal splitting*, Math. Comput. Simulation, 80, pp. 1133–1143.

A. Srinivasan and V. Aggarwal (2003), *Improved Monte Carlo linear solvers through non-diagonal splitting*, in International Conference on Computational Science and Its Applications Part III, Springer, Berlin, pp. 168–177.

A. Srinivasan and M. Mascagni (2002), *Monte Carlo techniques for estimating the Fiedler vector in graph applications*, in International Conference on Computational Science, Part II, Springer, Berlin, pp. 635–645.

A. J. Walker (1974), *New fast method for generating discrete random numbers with arbitrary frequency distributions*, Electron. Lett., 10, pp. 127–128.

Q. Wang, D. Gleich, A. Saberi, N. Etemadi, and P. Moin (2008), *A Monte Carlo method for solving unsteady adjoint equations*, J. Comput. Phys., 227, pp. 6184–6205.

W. Wasow (1952), *A note on the inversion of matrices by random walks*, Math. Tables Other Aids Comput., 6, pp. 78–81.