Monte Carlo solvers for sparse linear systems

Michele Benzi, Massimiliano Lupo Pasini

Abstract

We consider linear systems of equations characterized by a sparsity pattern, focusing in particular on problems derived from the discretization of partial differential equations. We propose a treatise of the current stochastic techniques present in literature that might be employed to solve the problems at hand. New theoretical results are introduced as sufficient criteria for the guarantee of the convergence of the methods analyzed. An analysis about viable preconditioners is accomplished as well.

1 Introduction

For a long period the pursuit of a reliable solver for linear and nonlinear algebraic systems has been one of the most important issues in Numerical Analaysis. The mathematical properties of the model (e.g. likely sparsity pattern of the matrix) have urged scientists to look for efficient and pattern-driven solvers, capable to cope with curse of dimensionality and high scalability requests.

As scalability computing move towards exascale facilities (which stands for machines computing up to 10^{18} flops), new numerical schemes suitable for this kind of architectures are strongly demanded. In fact the purpose is to combine high fidelity of the results with an optimized use of hardware resources at hand.

Recently new algorithms that combine statistical and numerical techniques have been developed. Even though *numerical efficiency* has been sacrificed sometimes, the advantages in terms of *robustness* of these methods make it worth carrying on studies in this direction. A class of methods that may be employed for this purpose is represented by Monte Carlo linear solvers.

These algorithms are the main topic covered in the work presented here, starting from what was developed so far by some of the authors who in literature gave a contribution (see [Hal62], [Hal94], [DA98], [DAK01], [AAD$^+$05],[ESW13] and [EMSH14]).

As underlined in [DA98], Monte Carlo methods may be split into two classes: *direct methods* ([Hal62], [Hal94], [DA98], [DAK01]) and *iterative methods* ([ESW13] and [EMSH14]). The first are characterized by a merely stochastic scheme, therefore the provided error with respect to the exact solution is made of just a stochastic component. The second ones, instead, combine numerical and statistical schemes generating two types of error: a *stochastic* one and a *systematic* one. It does not mean that it will be always simple to recognize them separately. However it is important to be aware of this intrinsic structure, in order to target what is the part of the scheme that requires a refinement (e.g. increase of numerical iterations rather than random walks).

The paper is organized as follows. In Section 2 we describe all the viable stochastic settings for the definition of the Markovian process. In Section 3 we reinterpret probabilistic facts in terms of spectral properties of the linear system to be solved. Section 4 is dedicated to necessary conditions and sufficient conditions for the convergence of the algorithms. In Section 5 different

adaptive approaches are presented, in order to truncate the random walks and to decide how many samplings to consider. Section 6 regards the study of matrices for which the convergence of the stochastic schemes is facilitated. Then in Section 7 a hybrid scheme combining numerical and probabilistic techniques is described: the *Monte Carlo Synthetic Acceleration*. Numerical results are presented in Section 8, while Section 9 is dedicated to conclusive remarks and goals for the future.

2 Stochastic setting

We start our treatise by introducing the direct methods.

Assume to start from a linear system such as

$$A\mathbf{x} = \mathbf{b}, \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$, $\mathbf{x}$, $\mathbf{b} \in \mathbb{R}^n$.

By applying a left preconditioning, the system (1) gets the following shape:

$$P^{-1}A\mathbf{x} = P^{-1}\mathbf{b}, \tag{2}$$

(2) can be reinterpreted as a fixed point scheme

$$\mathbf{x} = H\mathbf{x} + \mathbf{f}. \tag{3}$$

where $H = I - P^{-1}A$ and $\mathbf{f} = P^{-1}\mathbf{b}$.

Assuming that the spectral radius $\rho(H) < 1$, the solution to (3) can be written in terms of a power series of H (Neumann series)

$$\mathbf{x} = \sum_{i=0}^{\infty} H^i\mathbf{f}.$$

Therefore the fixed point scheme generates a sequence of approximate solutions $\{\mathbf{x}^{(k)}\}_{k=0}^{\infty}$ which converges to the exact solution regardless of the initial guess $\mathbf{x}_0$.

By restricting the attention to a single component of $\mathbf{x}$ we have

$$x_i = \sum_{\ell=0}^{\infty}\sum_{k_1}^{n}\sum_{k_2}^{n}\cdots\sum_{k_\ell=1}^{n} H_{k_0,k_1}H_{k_1,k_2}\cdots H_{k_{\ell-1},k_\ell}f_{k_\ell}. \tag{4}$$

The last equation can be reinterpreted as the realization of an estimator defined on a random walk.

Let us start considering a random walk whose state space S is characterized by the set of indices of the forcing term $\mathbf{f}$:

$$S = \{1, 2, \cdots, n\} \subset \mathbb{N}.$$

Each i-th step of the random walk has a random variable k_i associated with it. The realization of k_i represents the index of the component of $\mathbf{f}$ which is visited in the current step of the random walk.

The way the transition probability is built and the way the initial state of the random walk is chosen give birth to two different approaches we are going to use.

2.1 Forward Method

The goal is to evaluate a functional such as

$$J(\mathbf{x}) = (\mathbf{h}, \mathbf{x}) = \sum_{i=1}^{n} h_i x_i.$$

where $\mathbf{h} \in \mathbb{R}^n$ is the Riesz representative in $\mathbb{R}^n$ of the functional J. We can use it to build the initial probability $\tilde{p} : S \to [0, 1]$ of the random walk such that

$$\tilde{p}(k_0 = i) = \tilde{p}_{k_0} = \frac{|h_i|}{\sum_{i=1}^{n} |h_i|}.$$

It is important to stress out that the role of vector $\mathbf{h}$ is restricted to the construction of the initial probability. Once that is done, it is not used anymore in the definition of the stochastic process. A possible choice for the transition probability P can be

$$p(k_i = j \mid k_{i-1} = i) = P_{i,j} = \frac{|H_{i,j}|}{\sum_{k=1}^{n} |H_{i,k}|}.$$

where $\tilde{p}(\cdot, i) : S \to [0, 1]$ $\forall i \in S$. A related sequence of random variables $w_{i,j}$ can be defined such that

$$w_{i,j} = \frac{H_{i,j}}{P_{i,j}}.$$

The probability distribution of the random variables $w_{i,j}$ is represented by the transition matrix that rules the stochastic process. The $w_{i,j}$'s just introduced can be used to build one more sequence of random variables. At first we introduce quantities W_j

$$W_0 = \frac{h_{k_0}}{\tilde{p}_{k_0}}, \quad W_j = W_{j-1} w_{i,j}, \quad j = 1, \cdots, i.$$

By defining

$$X(\nu) = \sum_{m=0}^{k} W_m f_{i_m}$$

as the random variable associated with a specific permutation ν, we can define the estimator θ_i such as

$$\theta = E[X] = \sum_{\nu} P_\nu X(\nu).$$

The integer n represents the size of the solution vector to (1) and the index i is referred to the component of the solution vector we want to compute. P_ν is the probability associated with a specific permutation of the random walk. It can be proved that

$$E[W_i f_{k_i}] = (\mathbf{h}, H^i \mathbf{f}), \quad i = 0, 1, 2, \cdots$$

and

$$\theta_i = E\left[\sum_{i=0}^{\infty} W_i f_{k_i} \right] = (\mathbf{h}, \mathbf{x}).$$

A possible choice for $\mathbf{h}$ is a vector of the standard basis. This would correspond in setting manually the initial state of the random walk, which turns the related initial probability into a Kronecker delta

$$\tilde{p}(k_0 = i) = \delta_{i,j}.$$

By doing so, we have

$$\theta_i = E\Big[\sum_{\ell=0}^{\infty} W_\ell f_{k_\ell}\Big] = x_i = \sum_{l=0}^{\infty}\sum_{k_1=1}^{n}\sum_{k_2=1}^{n}\cdots\sum_{k_\ell=1}^{n} P_{k_0,k_1}P_{k_1,k_2}\cdots P_{k_{\ell-1},k_\ell} w_{k_0,k_1}w_{k_1,k_2}\cdots w_{k_{\ell-1},k_\ell} f_{k_\ell}. \tag{5}$$

As regards the variance, we remember that the following relation holds:

$$Var\Big[\sum_{\ell=0}^{\infty} W_\ell f_{k_\ell}\Big] = E\Big[\sum_{\ell=0}^{\infty} W_\ell^2 f_{k_\ell}^2\Big] - \Big(E\Big[\sum_{\ell=0}^{\infty} W_\ell f_{k_\ell}\Big]\Big)^2. \tag{6}$$

Hence the variance can be computed as the difference between the second moment of the random variable and the square of the first moment.

In order to apply the Central Limit Theorem (CLT) to the estimators defined above, we must require that the estimators have both finite expected value and variance. This is equivalent in checking the finiteness of the expected value and of the second moment. Therefore we have to impose the following conditions:

$$E\Big[\sum_{\ell=0}^{\infty} W_\ell f_{k_\ell}\Big] < \infty \tag{7}$$

$$E\Big[\sum_{\ell=0}^{\infty} W_\ell^2 f_{k_\ell}^2\Big] < \infty \tag{8}$$

The method proposed above, however, has the drawback that a set of permutations can be employed just to estimate a single entry of the solution at a time.

2.2 Adjoint Method

A second Monte Carlo method can be derived by considering the linear system adjoint to (2)

$$(P^{-1}A)^T\mathbf{y} = \mathbf{d}, \tag{9}$$

where $\mathbf{y}$ and $\mathbf{d}$ are the adjoint solution and source term.

By reformulating the fixed point scheme, introducing initial probability, transition probability and weights sequences in the same fashion as done before, the expected value for the estimator becomes

$$\theta_j = E\Big[\sum_{\ell=0}^{\infty} W_\ell \delta_{k_\ell,j}\Big] = \sum_{\ell=0}^{\infty}\sum_{k_1}^{n}\sum_{k_2}^{n}\cdots\sum_{k_\ell}^{n} b_{k_0} P_{k_0,k_1}P_{k_1,k_2}\cdots P_{k_{\ell-1},K_\ell} w_{k_0,k_1}\cdots w_{k_{\ell-1},k_\ell}\delta_{k_\ell,j}. \tag{10}$$

We can notice now that the component of the source vector, in each permutation of the random walk, is associated with the initial step. The Kronecker delta at the end of the series stands for a filter. It means that if the goal is to estimate the j-th component of the solution vector, just a subset of all the permutations gives a contribution to this. The only permutations considered are those ones that currently reside on the index associated with the entry to be estimated.

The variance is

$$Var\Big[\sum_{\ell=0}^{\infty} W_\ell f_{k_0}\delta_{k_\ell,j}\Big] = E\Big[\sum_{\ell=0}^{\infty} W_\ell^2 f_{k_0}^2\delta_{k_\ell,j}\Big] - \Big(E\Big[\sum_{\ell=0}^{\infty} W_\ell f_{k_0}\delta_{k_\ell,j}\Big]\Big)^2 \quad j = 1,\cdots,n. \tag{11}$$

On the same lead of what we did for the Forward method, we have to impose finiteness of the expected value and second moment. Therefore we require conditions:

$$E\left[\sum_{\ell=0}^{\infty} W_\ell \delta_{k_\ell,j}\right] < \infty \quad j = 1, \cdots, n \tag{12}$$

and

$$E\left[\sum_{\ell=0}^{\infty} W_\ell^2 f_{k_0}^2 \delta_{k_\ell,j}\right] < \infty \quad j = 1, \cdots, n. \tag{13}$$

The advantage of this method, compared to the Forward method, consists in the fact that a single set of permutation is used to estimate the entire solution vector. Therefore, in general, this algorithm has to be preferred to the previous one since it enables to reduce dramatically the computational burden.

3 Numerical interpretation of the statistical requirements

The requests imposed on the expected value and second moment of the estimators can be reformulated in a purely deterministic setting by looking at the spectral radii of some properly defined matrices.

For instance the condition of finiteness of the expected value can be reformulated by requiring

$$\rho(H) < 1, \tag{14}$$

where H is the iteration matrix of the fixed point scheme. In fact both equations (5) and (10) can be considered as power series in terms of H and the spectral radius of H smaller than 1 is a necessary and sufficient condition for the Neumann series to converge.

By analogy we aim at reproducing a similar reasoning for the second moment of the estimator, since we want it to be finite as well. By looking at the equations of the variance (6) and (11) for the Forward and the Adjoint method respectively, we notice that the second moment can be reinterpreted as a power series with respect to the matrices defined as follows:

$$H^*_{i,j} = \frac{H^2_{i,j}}{P_{i,j}} \quad \text{- Forward Method.}$$

or

$$H^*_{i,j} = \frac{H^2_{j,i}}{P_{i,j}} \quad \text{- Adjoint Method.}$$

Accordingly to the necessary and sufficient condition for the power series to converge, we require

$$\rho(H^*) < 1. \tag{15}$$

Condition (14) must be required for a generic fixed point scheme to reach convergence, whereas the extra condition (15) is typical of the stochastic schemes presented in this work. Having two conditions to verify makes the numerical treatment of the linear system via MC very difficult. We will discuss this in details later in the treatise.

4 Necessary conditions, sufficient conditions and choice of the transition matrix

Here below some results presented in [HMY13] about necessary conditions and sufficient conditions for convergence are enunciated. In particular a suitable choice for constructing a transition probability is discussed to guarantee and speed-up convergence.

We remember that the construction of the transition probability must satisfy some constraints such as

$$\begin{cases} P_{i,j} \geq 0 \\ \sum_{j=1}^{N} P_{i,j} = 1 \end{cases}$$

plus one more restriction which varies depending on the particular Monte Carlo approach chosen

$$\begin{aligned} &\text{Forward Method:} \quad H_{i,j} \neq 0 \Rightarrow P_{i,j} \neq 0 \\ &\text{Adjoint Method:} \quad H_{j,i} \neq 0 \Rightarrow P_{i,j} \neq 0 \end{aligned}$$

.

The previous restrictions on the attainable values for the entries of the transition probability are called *transition conditions*.

At first an introductory theoretical result is shown, necessary to motivate the choices made afterwards for the choice of the transition probability.

Theorem 1 *Consider a vector* $a = (a_1, a_2, \cdots, a_N)^T$ *where at least one element is nonzero,* $a_k \neq 0$ *for some* $k \in \{1, \cdots, N\}$.

- *For a probability vector* $p = (p_1, p_2, \cdots, p_N)^T$ *satisfying the transition conditions, the lower bound of* $\sum_{k=0}^{N} \frac{a_k^2}{p_k}$ *is* $\left(\sum_{k=1}^{N} |a_k| \right)^2$
- *There always exists a probability vector such that* $\sum_{k=0}^{N} \frac{a_k^2}{p_k} \geq c \geq 1$ *for all* $c > 1$.

It might be observed that the minimum of the quantity $\sum_{k=0}^{N} \frac{a_k^2}{p_k}$is attained when the probability vector is defined such as $p_k = \frac{|a_k|}{\sum_{k=1}^{N} |a_k|}$. It implies that the MAO (Monte Carlo Almost Optimal) probability is the one that reduces as much as possible the ∞-norm of H^*.

For the sake of simplicity let us introduce now a generic random walk truncated at a certain k-th step

$$\gamma_k : \; r_0 \to r_1 \to r_2 \to \cdots \to r_k$$

and the statistical estimator

$$X(\gamma_k) = \frac{H_{r_0,r_1} H_{r_1,r_2} \cdots H_{r_{k-1},r_k}}{P_{r_0,r_1} P_{r_1,r_2} \cdots P_{r_{k-1},r_k}} f_{r_k}$$

which is associated with the Forward Method.

Then the following theorem holds

Theorem 2 *(Suited to the Forward Method)*

Let $H \in \mathbb{R}^{n\times n}$ such that $\|H\|_\infty < 1$. Consider ν_k as the realization of a random walk γ truncated at the k-th step. Then, there always exists a transition matrix P such that $Var\Big(X(\nu_k)\Big) \to 0$ and $Var\Big(\sum_\nu X(\nu_k)\Big)$ is bounded as $k \to \infty$.

If we introduce the estimator

$$X(\gamma_k) = \frac{H^T_{r_0,r_1} H^T_{r_1,r_2} \cdots H^T_{r_{k-1},r_k}}{P_{r_0,r_1} P_{r_1,r_2} \cdots P_{r_{k-1},r_k}} sign(f_{r_0}) \|\mathbf{f}\|_1$$

which is associated with the Adjoint Method, then we can state a theorem specular to 2.

Theorem 3 *(Suited to the Adjoint Method)*

Let $H \in \mathbb{R}^{n\times n}$ with $\|H\|_1 < 1$. Consider ν_k as the realization of a random walk γ truncated at the k-th step. Then, there always exists a transition matrix P such that $Var\Big(X(\nu_k)\Big) \to 0$ and $Var\Big(\sum_\nu X(\nu_k)\Big)$ is bounded as $k \to \infty$.

In particular, relying on what guaranteed by Theorem 1, in both Theorems 2 and 3 the thesis holds by picking the MAO transition probability.

These theoretical results represent sufficient conditions for the convergence of the Forward and Adjoint Monte Carlo and they can be easily checked. However requiring $\|H\|_\infty < 1$ or $\|H\|_1 < 1$ may be too demanding. Indeed, even if there exist some preconditioners able to guarantee this condition (e.g. approximate inverse preconditioners with a proper value of dropping tolerance), it may entail to compromise the sparsity of the iteration matrix.

Thusly, in situations where the hypotheses of Theorems 2 and 3 are too demanding to be verified, the necessary and sufficient condition on the spectral radius of H^* is the only criterion to take as reference. Nonetheless this constraint consists in computing the spectral radii of two matrices and the calculation might be very costly. Moreover the computation of the spectral radii is affected by resilience issues as well as a deterministic solver for the resolution of a linear system. Therefore the original difficulty is transferred without being actually tackled.

5 Adaptive approaches

In formulas (5) and (10) the estimation of the solution to the linear system (2) involves infinite sums. Of course this is not viable for computational issues, therefore it is necessary to resort to truncation of the sums through proper criteria. In details, the truncation implies the employment of a finite number of permutations and a finite number of steps for each one of the permutations computed.

5.1 An adaptive cut-off for the length of random walks

At first we focus on the definition of a weight cut-off in order to decide where to terminate the histories. It means that we are looking for a quantity that guides us in deciding how many terms of the Neumann series to consider.

The goal is to find an automatic way for the cutoff of the histories. Therefore we are looking for an integer m such that

$$\tilde{\theta}_i = E\Big[\sum_{\ell=0}^{m} W_\ell f_{k_\ell}\Big] = x_i = \sum_{\ell=0}^{m}\sum_{k_1=1}^{n}\sum_{k_2=1}^{n}\cdots\sum_{k_\ell=1}^{n} P_{k_0,k_1}P_{k_1,k_2}\cdots P_{k_{\ell-1},k_\ell} w_{k_0,k_1}w_{k_1,k_2}\cdots w_{k_{\ell-1},k_\ell} f_{k_\ell}$$

and

$$\tilde{\theta}_j = E\Big[\sum_{\ell=0}^{m} W_\ell \delta_{k_\ell,j}\Big] = \sum_{\ell=0}^{m}\sum_{k_1}^{n}\sum_{k_2}^{n}\cdots\sum_{k_\ell}^{n} f_{k_0}P_{k_0,k_1}P_{k_1,k_2}\cdots P_{k_{\ell-1},K_\ell} w_{k_0,k_1}\cdots w_{k_{\ell-1},k_\ell}\delta_{k_\ell,j}$$

are good approximations of (5) and (10) respectively.

In [Sla13] there is a criterion for the cutoff of the random walk applicable to both the Forward and the Adjoint method. It requires the set up of a relative cutoff threshold W_c and to look for a step m such that

$$W_m \leq W_c W_0. \tag{16}$$

W_0 is the value of the weight at the initial step of the random walk and W_m is the value of the weight after m steps.

5.2 An a posteriori variance-based adaptivity

As regards the Forward method, we know that the expression for the variance is defined in formula (6).

In this context, a reasonable criterion to determine the number $\tilde{N}_i$ of random walks to be run is setting a threshold ε_1 and determine

$$\tilde{N}_i \quad s.t. \quad \frac{\sqrt{Var[\theta_i]}}{|E[\theta_i]|} < \varepsilon_1, \quad i = 1, \cdots, n. \tag{17}$$

The dependence of $Var[\theta_i]$ and $E[\theta_i]$ on $\tilde{N}_i$, which seems to be absent in the previous formula, is highlighted by the fact that θ_i is estimated by fixing a finite number of histories. Therefore we are controlling the relative standard deviation requiring it not to be too large. In other words we are pursuing a statistical setting where the uncertainty factor is not dominating over the expected value. This simple adaptive approach can be applied for the estimation of each component x_i. Hence different number of histories may be employed to compute different entries of the solution vector.

As concerns the Adjoint method, the estimation of the variance for each entry is reproduced by Formula (11).

A possible adaptive selection of $\tilde{N}$, in this situation, is

$$\tilde{N} \quad s.t. \quad \frac{\|\boldsymbol{\sigma}_{\tilde{N}}\|_1}{\|\mathbf{x}\|_1} < \varepsilon_1, \tag{18}$$

where $\boldsymbol{\sigma}$ is a vector whose entries are $\boldsymbol{\sigma}_{\tilde{N},i} = Var[\theta_i]$.

What introduced above can be exploited in order to build an a posteriori adaptive algorithm, capable to identify the minimal value of $\tilde{N}$ that verifies (17) or (18) respectively.

Below there are the pseudo-codes associated with both Forward and Adjoint Monte Carlo.

Data: N, ε_1
Result: $\tilde{N}_i$, σ_i, x_i
For each entry of the solution vector (for $i = 1, \cdots, n$) ;
$\tilde{N}_i = N$;
compute θ_i;
while $\frac{\sigma_i}{|E[\theta_i]|} < \varepsilon_1$ **do**
  $\tilde{N} = \tilde{N} + N$;
  compute $E[\theta_i] = x_i$;
end
return θ_i, x_i, σ_i;

Algorithm 1: A posteriori adaptive Forward Monte Carlo

Data: N, ε_1
Result: $\tilde{N}$, $\boldsymbol{\sigma_i}$, $\mathbf{x}$
$\tilde{N} = N$;
compute θ;
while $\frac{\|\boldsymbol{\sigma}\|}{\|E[\theta]\|} < \varepsilon_1$ **do**
  $\tilde{N} = \tilde{N} + N$;
  compute $E[\theta] = \mathbf{x}$;
end
return θ, $\mathbf{x}$, $\boldsymbol{\sigma}$;

Algorithm 2: A posteriori adaptive Adjoint Monte Carlo

6 Sets of matrices for which the convergence of the MC linear solver is guaranteed

As we have already seen, sufficient conditions for the convergence of the Monte Carlo linear solvers are very restrictive (see [HMY13]), whereas the necessary and sufficient condition requires the computation of $\rho(H)$and $\rho(H^*)$, which is very expensive in general. Once a proper preconditioner is picked (e.g. Approximate Inverse), $\rho(H) < 1$ may be always guaranteed, whilst $\rho(H^*) < 1$ is more problematic to be verified.

In order to bypass the cost of these checks, it is useful to pursue for sets of matrices for which the Monte Carlo linear solver always converges, maybe with the use of an ad-hoc preconditioner.

6.1 Strictly diagonally dominant matrices

One of the these sets is represented by strictly diagonally dominant matrices.

Definition 1 *A matrix $A \in \mathbb{R}^{n\times n}$is strictly diagonally dominant (s.d.d) by rows if*

$$|a_{ij}| > \sum_{\substack{i=1 \\ i\neq j}}^{i=n} |a_{ij}| \tag{19}$$

Definition 2 *A matrix $A \in \mathbb{R}^{n\times n}$is strictly diagonally dominant (s.d.d) by columns if*

$$|a_{ij}| > \sum_{\substack{j=1 \\ j\neq i}}^{j=n} |a_{ij}| \tag{20}$$

When (19) holds we can resort to a left diagonal preconditioning and we get an iteration matrix $H = I - D^{-1}A$ such that $\|H\|_\infty < 1$.

Introducing a MAO transition probability for the Forward Method we get

$$P_{ij} = \frac{|H_{ij}|}{\sum_{k=1}^{n} |H_{ik}|}$$

therefore the entries of H^* are defined as follows

$$H^*_{ij} = \frac{H^2_{ij}}{P_{ij}} = |H_{ij}| \Big(\sum_{k=1}^{n} |H_{ik}| \Big).$$

Therefore

$$\sum_{j=1}^{n} |H^*_{ij}| = \sum_{j=1}^{n} H^*_{ij} = \Big(\sum_{j=1}^{n} |H_{ij}| \Big) \Big(\sum_{k=1}^{n} |H_{ik}| \Big) = \Big(\sum_{j=1}^{n} |H_{ij}| \Big)^2 < 1 \quad \forall i = 1, \cdots, n.$$

This automatically implies that $\rho(H^*) <= \|H^*\|_\infty < 1$. Thus the Forward Monte Carlo converges. However nothing a priori can be stated about the behavior of the Adjoint method.

Instead if (20) holds we can resort to a right diagonal preconditioning and we get an iteration matrix $H = I - AD^{-1}$ such that $\|H\|_1 < 1$. In this case, by following a similar reasoning to the one made before, we conclude that the Adjoint Method convergences thanks to $\|H^*\|_1 < 1$, whereas nothing can be said in advance as concerns the Forward method.

6.2 Generalized diagonally dominant matrices (GDDM)

Another set of matrices for which the convergence of the MC solvers is guaranteed is made of *generalized diagonally dominant matrices.*

Definition 3 *A square matrix $A \in \mathbb{R}^{n \times n}$ is said to be generalized diagonally dominant if*

$$|a_i i| x_i \geq \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| x_j, \quad i = 1, \cdots, n$$

for some positive vector $\mathbf{x} = (x_1, \cdots, x_n)^T$.

A proper subset of the generalized diagonally dominant matrices is represented by M- matrices (see [Axe96]). There are many definitions of M-matrices that may be found on books and all of them are equivalent. The one presented in [Saa03] states the following

Definition 4 *A matrix $A \in \mathbb{R}^{n \times n}$ is said to be an M-matrix if it satisfies the following four properties:*

- $a_{i,i} > 0$ *for* $i = 1, \cdots, n$
- $a_{i,j} \leq 0$ *for* $i \neq j$, $i, j = 1, cdots, n$
- *A is nonsingular*
- $A^{-1} \geq 0$

A very important result about M-matrices enables to come up with a convergent splitting and, therefore, a convergent fixed point scheme.

In particular Jacobi, Block Jacobi and Gauss Seidel are convergent splitting for an M-matrix. However, the construction of a splitting like this is not enough to ensure the convergence of the Monte Carlo linear solver. In fact none of these conditions is sufficient to guarantee $\rho(H^*) < 1$. Because of this obstacle, other options must be considered.

For this purpose we appeal to a result in [Li02]. In this paper the author presents an algorithm to transform a generalized diagonally dominant matrix with nonzero diagonal entries into strictly diagonally dominant. The algorithm works for a generic complex matrix $A \in \mathbb{C}^{n\times n}$.

Here below we report the algorithm at hand

For a given complex matrix A, $a_{ii} \neq 0$, $i = 1, \cdots, n$;

1. Compute $S_i = \sum_{\substack{j=1^n \\ j\neq i}} |a_{ij}|$, $i = 1, 2, \cdots, n$
2. Set $t = 0$. For $i = 1, 2, \cdots, n$, if $|a_{ii}| > S_i$, then set $t = t + 1$
3. If $t = 0$, then print "A is not a GDDM": END
4. If $t = n$, then print "A is a GDDM": END
5. **for** *i=1,n* **do**
   $d_i = \frac{S_i+\varepsilon}{|a_{ii}|+\varepsilon} \quad \varepsilon > 0, \quad j = 1, 2, \cdots, n;$
   $a_{ji} = a_{ji} \cdot d_i$
   end
6. Go to step 1.

Algorithm 3: Algorithm to turn a GDDM matrix into a s.d.d by rows.

This approach turns a generalized diagonally dominant matrix into a strictly diagonally dominant matrix by rows. By substituting $S_i = \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|$ at step 1 with $S_i = \sum_{\substack{i=1 \\ i\neq j}}^{n} |a_{ij}|$ and by replacing $a_{ji} = a_{ji} \cdot d_i$ with $a_{ji} = a_{ji} \cdot d_j$ we obtain the algorithm that turns a GDDM matrix into a s.d.d. by columns.

Once we have applied this transformation to the matrix at hand into a s.d.d., we can use the Monte Carlo linear solver which is ensured to converge.

6.3 "Block-diagonally dominant"-like matrices

In this section we wonder in which situations a block diagonal preconditioning might come up with a convergent Monte Carlo linear solver. By mimicking the computations we already showed previously, the iteration matrix $H \in \mathbb{R}^{n\times n}$ resulting from a block diagonal preconditioning takes the form

$$H = I - D^{-1}A = \begin{bmatrix} 0_{n_1\times n_1} & -[A_{11}]^{-1}A_{12} & \cdots & \cdots & -[A_{11}]^{-1}A_{1p} \\ -[A_{22}]^{-1}A_{21} & 0_{n_2\times n_2} & -[A_{22}]^{-1}A_{23} & \cdots & -[A_{22}]^{-1}A_{2p} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -[A_{pp}]^{-1}A_{p1} & \cdots & \cdots & -[A_{pp}]^{-1}A_{p,p-1} & 0_{n_p\times n_p} \end{bmatrix}.$$

For the sake of clarity, we notify that the symbol "%" used in the following subsections represents the modulus function between two integer numbers, while the floor function is represented with pretty standard symbol "$\lfloor\cdot\rfloor$".

Forward method

By assuming that all n_i's have the same value we may define

$$m = n_i = \text{size of a block} = \frac{n}{p}.$$

The MAO transition probability matrix becomes:

$$P_{i,j} = \frac{|H_{i,j}|}{\sum_{k=1}^{n}|H_{i,k}|} = \frac{\left|\left([A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{i}{m}\rfloor}]^{-1}A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{j}{m}\rfloor}\right)_{i\%m,j\%m}\right|}{\sum_{\substack{k=1\\k\neq i}}^{n}\left|\left([A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{i}{m}\rfloor}]^{-1}A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{k}{m}\rfloor}\right)_{i\%m,k\%m}\right|}$$

Consequently, the H^* matrix is defined such that

$$H^*_{i,j} = |H_{i,j}|\left(\sum_{k=1}^{n}|H_{ik}|\right) = \left|\left([A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{i}{m}\rfloor}]^{-1}A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{j}{m}\rfloor}\right)_{i\%m,j\%m}\right|\sum_{\substack{k=1\\k\neq i}}^{n}\left|\left([A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{i}{m}\rfloor}]^{-1}A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{k}{m}\rfloor}\right)_{i\%m,k\%m}\right|$$

By computing the sum over a generic row of H^* we get:

$$\sum_{j=1}^{n}|H^*_{i,j}| = \sum_{j=1}^{n}H^*_{i,j} = \left(\sum_{\substack{j=1\\j\neq i}}^{n}\left|\left([A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{i}{m}\rfloor}]^{-1}A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{j}{m}\rfloor}\right)_{i\%m,j\%m}\right|\right)^2.$$

If we focus on the norm $\|H^*\|_\infty$, then the following equivalence condition holds:

$$\|H^*\|_\infty < 1 \Leftrightarrow \sum_{\substack{j=1\\j\neq i}}^{n}\left|\left([A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{i}{m}\rfloor}]^{-1}A_{\lfloor\frac{i}{m}\rfloor\lfloor\frac{j}{m}\rfloor}\right)_{i\%m,j\%m}\right| < 1 \quad \forall i = 1,\cdots,n$$

A sufficient condition for this to happen is

$$\sum_{\substack{j=1\\j\neq i}}^{p}\|[A_{ii}]^{-1}A_{ij}\|_\infty < 1. \quad \forall i = 1,\ldots,p. \tag{21}$$

By defining a matrix $\tilde{H} \in \mathbb{R}^{p \times p}$ such that

$$\tilde{H} = \begin{bmatrix} 0 & \|[A_{11}]^{-1}A_{12}\|_\infty & \cdots & \cdots & \|[A_{11}]^{-1}A_{1p}\|_\infty \\ \|[A_{22}]^{-1}A_{21}\|_\infty & 0 & \|[A_{22}]^{-1}A_{23}\|_\infty & \cdots & \|[A_{22}]^{-1}A_{2p}\|_\infty \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \|[A_{pp}]^{-1}A_{p1}\|_\infty & \cdots & \cdots & \|[A_{pp}]^{-1}A_{p,p-1}\|_\infty & 0 \end{bmatrix}$$

We can use the $\tilde{H}$ matrix just defined in order to introduce a sufficient condition for the convergence of the Forward Monet Carlo method with a Block Diagonal preconditioning.

$$\|\tilde{H}\|_\infty < 1 \Rightarrow \|H^*\|_\infty < 1. \tag{22}$$

Adjoint method

Analogously to the Forward method, if we define

$$(H^*)^T_{i,j} = |H^T_{i,j}|\left(\sum_{k=1}^{n}|H^T_{ik}|\right)$$

we can formulate a sufficient condition for the convergence of the Adjoint Monte Carlo method by introducing a matrix $\tilde{H}$ which in this case is such that

$$\tilde{H} = \begin{bmatrix} 0 & \|[A_{11}]^{-1}A_{12}\|_1 & \cdots & \cdots & \|[A_{11}]^{-1}A_{1p}\|_1 \\ \|[A_{22}]^{-1}A_{21}\|_1 & 0 & \|[A_{22}]^{-1}A_{23}\|_1 & \cdots & \|[A_{22}]^{-1}A_{2p}\|_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \|[A_{pp}]^{-1}A_{p1}\|_1 & \cdots & \cdots & \|[A_{pp}]^{-1}A_{p,p-1}\|_1 & 0 \end{bmatrix}.$$

The sufficient condition assumes the form

$$\|\tilde{H}\|_1 < 1 \Rightarrow \|H^*\|_1 < 1. \tag{23}$$

7 Linear solvers with hybrid schemes

Even if the standard Monte Carlo, in both its Forward and Adjoint formulation, has the advantage of being embarrassingly parallelizable, it has a drawback as well. In fact its slow rate of convergence, associated with the $\frac{1}{\sqrt{N}}$ behavior predicted but the CLT, prevents its use for problems of actual interest in applied linear algebra.

Moreover, when the spectral radius of the iteration matrix is near the unit value, Monte Carlo direct methods reach convergence after an excessive number of steps for each permutation. Therefore the standard Mote Carlo paradigm is most of the time unpractical for solving linear systems associated with realistic applications.

To cope with this, in the early 2000's Evans et al. introduced a new method: the *Monte Carlo Synthetic Acceleration (MCSA)* ([ESW13] and [EMSH14]). If we think about a fixed point formulation of the problem such as $\mathbf{x} = H\mathbf{x} + \mathbf{b}$, then the Monte Carlo Synthetic Acceleration assumes this form

Data: H, $\mathbf{b}$, $\mathbf{x}_0$
Result: x_{num}
$\mathbf{x}^l = \mathbf{x}_0$;
while *not reached convergence* **do**
  $\mathbf{x}^{l+\frac{1}{2}} = H\mathbf{x}^l + \mathbf{b}$;
  $\mathbf{r}^{l+\frac{1}{2}} = \mathbf{b} - A\mathbf{x}^{l+\frac{1}{2}}$;
  $\delta\mathbf{x}^{l+\frac{1}{2}} = (I - H)^{-1}\mathbf{r}^{l+\frac{1}{2}}$;
  $\mathbf{x}^{l+1} = \mathbf{x}^{l+\frac{1}{2}} + \delta\mathbf{x}^{l+\frac{1}{2}}$;
end
$x_{num} = x^{l+1}$;

Algorithm 4: Monte Carlo Synthetic Acceleration

The Monte Carlo method is used to compute the updating contribution $\delta\mathbf{x}^{l+\frac{1}{2}}$.

The role of the first line of the iterative process has the goal to compress the residual of the previous iteration. Once this done, the corrected residual is employed to compute an update $\delta x^{1+\frac{1}{2}}$ to the solution via one of the Monte Carlo techniques introduced before. In particular the intent is to designate most of the time employed by the computation to the accomplishment of the stochastic part of the algorithm. This is useful to decrease the probability of fault occurrence in the non-resilient steps.

7.1 Preconditioning techniques

In this section we focus on the pursuit of preconditioners that facilitate the convergence of the stochastic schemes introduced so far. A priori, all of the following techniques are good candidates to improve the numerical setting. However issues about the fill-in effect and the verification of convergence conditions limit their employment for our purposes.

Block Preconditioning

Consider a matrix $A \in \mathbb{R}^{n\times n}$ of the form

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1p} \\ A_{21} & A_{22} & \cdots & A_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ A_{p1} & \cdots & \cdots & A_{pp} \end{bmatrix}$$

where p is a divisor of n and such that $A_{ii} \in \mathbb{R}^{n_i\times n_i}$ are nonsingular square matrices. By defining a block diagonal matrix such as

$$D = \begin{bmatrix} A_{11} & 0_{12} & \cdots & 0_{1p} \\ 0_{21} & A_{22} & \cdots & 0_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 0_{p1} & \cdots & \cdots & A_{pp} \end{bmatrix}$$

it is possible to compute its inverse. D^{-1} can be used as a preconditioner for the original matrix A. In particular

$$D^{-1}A = \begin{bmatrix} I_{n_1 \times n_1} & A_{11}^{-1}A_{12} & \cdots & \cdots & A_{11}^{-1}A_{1p} \\ A_{22}^{-1}A_{21} & I_{n_2 \times n_2} & A_{22}^{-1}A_{23} & \cdots & A_{22}^{-1}A_{2p} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A_{pp}^{-1}A_{p1} & \cdots & \cdots & A_{pp}^{-1}A_{p,p-1} & I_{n_p \times n_p} \end{bmatrix}.$$

The ultimate goal is to resort to this preconditioning technique in order to achieve the condition $\rho(D^{-1}A) < 1$. In fact this is necessary in order to reformulate 1 into 3.

Convergence of 3 with block diagonal preconditioning, from the deterministic point of view, is guaranteed when A is an irreducibly diagonally dominant matrix or when A is an M-matrix. In fact in situation such as these, theoretical results guarantee that $A = D - N$ is a regular splitting (see [Axe96]).

Approximate Inverse of a Sparse Matrix (AINV)

This stimulated our interests into other types of preconditioners. At first we concentrated on Approximate Inverse preconditioners (see [Saa03] and [Ben02]).

There are many practical techniques that may be applied in order to build the AINV preconditioner. One of these is described in [BMM96] and [BT98]. This is the actual approach we applied to the construction of the preconditioner. The technique constructs the preconditioner via a bi-conjugation fashion. The idea is to compute an approximate factorization of the form $W^TAZ = D$. W and Z are unit upper triangular matrices and D is a diagonal matrix.

In order to compute the AINV preconditioner, a `FORTRAN` code provided by Miroslav Tuma (Academy of Sciences, Prague) is used. We report as follows the algorithm employed for the construction of the so called *Right-looking Factored AINV*. The algorithm provide the user with a dropping tolerance τ which can be used in order to tune the fill-in effect of the factored preconditioner.

Data: $p = q = (0, \cdots, 0) \in \mathbb{R}^n$, $Z = [z_1, \cdots, z_n] = I_n$, $W = [w_1, \cdots, w_n] = I_n$
Result: Z, W
for $k = 1, \cdots, n$ **do**
  $p_k = w_k^T A e_k$, $q_k = e_k^T A z_k$;
  for $i = k+1, \cdots, n$ **do**
    $p_i = (w_k^T A e_k)/p_k$, $q_i = (e_k^T A z_i)/q_k$;
    Apply a dropping rule to p_i, q_i;
    $w_i = w_i - w_k p_i$, $z_i = z_i - z_k q_i$
    Apply a dropping rule to $w_{j,i}$ and $z_{j,i}$ for $j = 1, \cdots, i$;
  end
  ;
end
;
Choose diagonal entries of D as the component of p or q;
return Z, W;

Algorithm 5: Right-looking Factored AINV

Incomplete LU Factorization Preconditioners (ILU)

This set of preconditioners has the goal of computing a sparse lower triangular matrix L and a sparse upper triangular matrix U such that the residual matrix $R = LU - A$ respects some constraints. One of these constraints may be the fact that some entries must be set to zero. For instance we can introduce a zero pattern set

$$P \subset \{(i,j)|i \neq j; 1 \leq i,j \leq n\}$$

and use this set in order to impose some constraints on the shape of the preconditioner built.

Different variants of the Incomplete LU Factorization algorithm are available (see [**?**]). However it is possible to prove that if all the formulation are well defined, then they are all equivalent. One of these variants is the following

Data: P
Result: A
For each $(i,j) \in P$ set $a_{ij} = 0$;
for $k = 1, \cdot, n-1$ **do**
  for $i = k+1, n$ **do**
    if $(i,k) \notin P$ **then**
      $a_{ik} = \frac{a_{ik}}{a_{kk}}$ **for** $j = k+1, \cdots, n$ *and for* $(i,j) \notin P$ **do**
        $a_{ij} = a_{ij} - a_{ik} * a_{kj}$
      end
    end
  end
end
return Z, W;

Algorithm 6: General Static Pattern ILU

This kind of algorithm is guaranteed to terminate without any breakdown just for M-matrices. Assuming that the zero pattern P coincides with the zero pattern of A leads to the $ILU(0)$.

Data: P
Result: A
for $i = 2, \cdot, n$ **do**
  for $k = i, i-1$ *and for* $(i,k) \in NZ(A)$ **do**
    $a_{ik} = \frac{a_{ik}}{a_{kk}}$ **for** $j = k+1, \cdots, n$ *and for* $(i,j) \in NZ(A)$ **do**
      $a_{ij} = a_{ij} - a_{ik}a_{kj}$
    end
  end
end

Algorithm 7: ILU(0).

More sophisticated variants of the algorithm enable to enlighten the constraint of the sparsity pattern, finding a compromise between sparsity and accuracy in approximating the inverse of A. We refer to [Saa03] and [Ben02] for further information about it.

8 Numerical results

All the numerical results shown in the following sections are computed with `Matlab` on a standard laptop. The simulations are accomplished in a serial mode.

In the first subsection the standard Monte Carlo solver is employed, while in the subsequents ones results are obtained with the MCSA scheme.

8.1 Check on the effectiveness of the adaptive criteria

In this subsection we show some numerical tests to validate the utility of the adaptive approaches for the selection of the number of permutations to employ. The standard Monte Carlo linear is employed for this purpose.

The adaptive threshold associated with formulas (17) and (18) is set to $\varepsilon_1 = 0.01$ and to $\varepsilon_1 = 10^{-4}$.

A set of matrices associated with problems of various nature has been collected All the problems treated have a small number of d.o.f's to control the massive computational cost of MC solvers on a standard laptop. One of these matrices is produced by the finite volume discretization of a thermal equation (Marshak problem). In addition three more matrices are derived from the discretization of other differential problems. In particular they are a 1D shifted Laplace operator discretized with finite differences, a 2D laplacian discretized with finite differences as well and an advection diffusion problem discretized with quadrilateral linear finite elements (the `ifiss` package is used to generate the matrix). For all the problems at hand a fixed point scheme is formulated after a left diagonal preconditioner is applied. Details about all these matrices are gathered in Table 1.

Type of problem	**d.o.f.'s**	$\rho(H)$	Forward $\rho(H^*)$	Adjoint $\rho(H^*)$
1d shifted Laplacian	50	0.4991	0.9827	0.9827
2d Laplacian	900	0.9949	0.994	0.9945
ifiss	1089	0.9835	0.9827	0.9827
Marshak Equation	1600	0.6009	0.3758	0.3815

Table 1: Properties of the matrices employed for the checks on adaptivity.

For the numerical tests we present results just for the Adjoint Monte Carlo approach, since it is the scheme preferred in general for the aforementioned reasons.

At each adaptive check the number of random walks employed is increased by two.

No restraints are put for a maximal number of random walks admitted. Therefore the algorithm is set free to run until the constraint in formula (18) is verified.

Here below Tables 2 and 3 show results for the different test cases considered using $\varepsilon_1 = 0.01$ and $\varepsilon_1 = 10^{-4}$ respectively. By comparing the two tables, it is noticed that a decrease of the threshold induces a reduction of the relative error too. Therefore this validates the effectiveness of the adaptive selection of histories with an error reduction goal.

Type of problem	**Relative Error**	**Nb. Histories**
1d shifted Laplacian	0.139	730
2d Laplacian	0.136	570
ifiss advection diffusion	0.376	630
Marshak equation	0.288	880

Table 2: Adjoint Monte Carlo. Adaptive selection of histories. $\varepsilon_1 = 0.01$

Type of problem	Relative Error	Nb. Histories
1d shifted Laplacian	0.0198	54720
2d Laplacian	0.0215	30950
ifiss advection diffusion	0.0649	21090
Marshak equation	0.0312	72210

Table 3: Adjoint Monte Carlo. Adaptive selection of histories. $\varepsilon_1 = 10^{-4}$

8.2 The Poisson equation solved with MCSA

In the current subsection a Poisson problem is taken into account:

$$\begin{cases} -\Delta u = f \quad \text{in} \quad \Omega \\ u = 0 \quad \text{on} \quad \partial\Omega \end{cases} \tag{24}$$

where $\Omega = (0,1) \times (0,1)$, $\sigma = 1$. A finite difference scheme is employed to discretize the problem. As a right hand side, a sinusoidal distribution in both x and y directions is considered, so that

$$f(x_i, y_i) = sin\left(\frac{\pi x}{N-1}\right) sin\left(\frac{\pi y}{N-1}\right).$$

N is the number of node considered on each direction, in our case $N = 32$. The resulting linear system has 900 d.o.f.'s and $h = \frac{1}{29}$. By applying a left diagonal preconditioning on the matrix, it is possible to reformulate the problem as a fixed point scheme. The iteration matrix H is characterized by the fact that $\|H\|_1 = 1$, which is not sufficient to guarantee the convergence of any stochastic scheme analyzed in this work. However, by theoretical results, it is well known that applying a left diagonal preconditioning to the problem at hand produces an iteration matrix H such that

$$\lim_{h \to 0} \rho(H) = 1^-$$

where h is the spatial discretization step. This is enough to have the guarantee that the deterministic fixed point scheme converges without the explicit computation of the spectral radius $\rho(H)$. However, the smaller is the discretization step h, the higher is the number of iterations required to reach a prescribed convergence criterion. For our discretization setting, the iteration matrix H has a spectral radius of $\rho(H) \approx 0.9949$.

In order to have the guarantee of the convergence of the Adjoint Monte Carlo method, it is necessary to look at the spectral radius of H^* too. If an almost optimal probability is used, this implies that the Adjoint Monte Carlo method has a H^* matrix such that

$$\|H^*\|_1 \leq (\|H\|_1)^2 = 1.$$

This condition by itself is not enough to guarantee that $\rho(H^*) < 1$. However it must be recalled that, for the way the almost optimal probability is defined, in this particular situation we have

$$H \equiv H^* \quad \Rightarrow \quad \rho(H) \equiv \rho(H^*).$$

Therefore the diagonal preconditioning always guarantees a convergent setting for the Monte Carlo linear solvers, once it is applied on an n-dimensional Laplace operator discretized with finite differences .

The computation of the solution is realized both with the deterministic Richardson and with Adjoint MCSA to compare the effectiveness of the stochastic technique. Results are reported in Table 4. An evident discrepancy in terms of numerical iterations is discovered with the MCSA outperforming the preconditioned fixed point.

algorithm	relative err.	# iterations
Richardson	$9.8081 \cdot 10^{-8}$	3131
Adjoint MCSA	$7.6562 \cdot 10^{-8}$	485

Table 4: Numerical results for the Poisson problem.

8.3 A diffusion reaction problem solved with MCSA

By modifying the equation of the previous test case, we consider now an advection reaction problem

$$\begin{cases} -\Delta u + \sigma u = f \quad \text{in} \quad \Omega \\ u = 0 \quad \text{on} \quad \partial\Omega \end{cases} \tag{25}$$

where $\Omega = (0,1) \times (0,1)$, $\sigma = 1$ and $f = 1$. A finite difference scheme is applied to discretize the problem. The number of nodes on each direction of the domain is 100, so that $h \approx 0.01$. The discretized problem has 9604 d.o.f's. A left diagonal preconditioning is still applied to the stiffness matrix obtained from the discretization. The 1-norm of the iteration matrix is $\|H\|_1 \approx 0.8$. This automatically guarantees the convergence of the Adjoint Monte Carlo linear solver. The computation of the solution to the partial differential problem is still accomplished with the MCSA algorithm. A threshold of $\varepsilon = 10^{-8}$ is used as a stop criterion for the relative residual. The threshold for the adaptive selection of the random walks instead is set to $\varepsilon_1 = 0.01$. In Table 5 a comparison between the deterministic Richardson scheme and the MCSA is provided. The employment of the MCSA provides a significant reduction of the number of iterations necessary to satisfy the convergence criterion.

algorithm	relative err.	# iterations
Richardson	$9.073 \cdot 10^{-8}$	69
Adjoint MCSA	$4.9497 \cdot 10^{-8}$	39

Table 5: Numerical results for the diffusion reaction problem.

8.4 A parabolic problem solved with MCSA

In this subsection we restrict our treatise to the analysis of parabolic partial differential problems of the form

$$\begin{cases} \frac{\partial u}{\partial t} + \mathcal{L}(u) = f \quad \text{in}\Omega \\ \text{B.C.} \quad \text{on} \quad \delta\Omega. \end{cases} \tag{26}$$

The partial derivative in time is discretized with finite differences, whereas quadrilateral linear finite elements are employed for the space discretization. Naming N_h the number of d.o.f.'s associated with the finite element discretization, the problem 26 turn into a fully discretized problem such as:

$$\left(\frac{1}{\Delta t}M + A\right)(\mathbf{u}^{k+1} - \mathbf{u}^k) + A[\theta\mathbf{u}^{k+1} + (1-\theta)\mathbf{u}^k] = \theta\mathbf{f}^{k+1} + (1-\theta)\mathbf{f}^k,$$

where k is an index that goes with the time dimension.

By tuning the value of the parameter θ different numerical schemes can be obtained with different properties of numerical stability. However our discussion is not involved in this kind of

issues. In particular we restrict our attention to a single generic time step associated with an Implicit Euler time discretization (corresponding to $\theta = 1$). The vector of the right hand side is chosen so that the exact solution to the linear system for the specific time step chosen is a unit vector $\tilde{\mathbf{u}} = \mathbb{1}^{N_h}$.

By referring to h as the space discretization step, we pick

$$\Delta t \leq Ch.$$

Give an open and bounded set $\Omega = (0,1) \times (0,1)$, we consider the following problem

$$\begin{cases} \frac{\partial u}{\partial t} - \mu \Delta u + \boldsymbol{\beta}(\mathbf{x}) \cdot \nabla u = 0, & \mathbf{x} \in \Omega, \quad t \in (0,T] \\ u(\mathbf{x},0) = u_0, & \mathbf{x} \in \Omega \\ u(\mathbf{x},t) = u_D(\mathbf{x}), & \mathbf{x} \in \partial\Omega, \quad t \in (0,T], \end{cases} \tag{27}$$

where $\mu = \frac{3}{200}$, $\boldsymbol{\beta}(\mathbf{x}) = [2y(1-x^2),\ -2x(1-y^2)]^T$,
$u_D = 0$ on $\{\{x = 0\} \times (0,1)\}$, $\{(0,1) \times \{y = 0\}\}$, $\{(0,1) \times \{y = 1\}\}$.

The discretization of the problem is realized with the `IFISS` toolbox. The value of the discretization step is $h = 2^{-8}$, generating a discretized linear system with 66,049 d.o.f.'s. As concerns the time discretization, the time step is set to $\Delta t = 10h$.

The sparse approximate inverse is employed for the linear system at hand as a right preconditioner, with a drop tolerance of $\tau = 0.05$ for both the factors.

In this situation it is not possible to resort to any of the sufficient conditions to verify the convergence. Therefore the only viable option is the explicit computation of the spectral radii of H and H^*. The spectral radius of the iteration matrix is $\rho(H) \approx 0.9218$ and the spectral radius of H^* for the Adjoint Monte Carlo is $\rho(H^*) \approx 0.9148$.

The fill-in effect is

$$\frac{nnz(H)}{nnz(A)} = 4.26,$$

therefore the relative number of nonzero elements in H is still acceptable in terms of sparsity pattern and memory storage.

The threshold for the check on the relative residual is set to $\varepsilon = 10^{-8}$ is, whilst the threshold for the adaptive selection of the random walks is set to $\varepsilon_1 = 0.01$. In order to test the effectiveness of employing the MCSA with respect to a purely deterministic scheme, a comparison with the Richardson algorithm is accomplished. The results are shown in Table 6. As you can see, the adoption of the MCSA halves the number of iterations with respect to the fixed point method. Of course this advantage has the increase of the computational time as a payoff. Nevertheless we put off this discussion which is of interest for a parallelized implementation of the method.

algorithm	relative err.	# iterations
Richardson	$6.277 \cdot 10^{-7}$	178
Adjoint MCSA	$9.885 \cdot 10^{-7}$	90

Table 6: Numerical results for the parabolic problem.

Besides the sparse approximate inverse preconditioner other preconditioners have been testes as well, such as block diagonal and incomplete LU factorization. However the fill-in effect introduced by the application of these preconditioners compromises the sparsity of the linear system.

9 Conclusions and future developments

The work presented in this paper has been accomplished to provide an overview about MC linear solvers. In particular the goal was to combine the main theoretical results found in literature with additional novel contributions on both the theoretical and empirical sides. This made it possible to understand the actual limitations of the presented algorithms.

Necessary and sufficient conditions to ensure convergence are too costly for most of the applicative problems coming fro engineering and Computational Physics. On the other hand, sufficient criteria are currently viable for a restricted set of matrices (generalized diagonally dominant).

An empirical analysis has been carries out on parabolic problems and it has been shown that a proper selection of the time discretization step enables the employment of stochastic solvers.

We reserve for future works the analysis of the plausible connections between $\rho(H)$, $\rho(H^*)$ and the number of numerical iterations necessary to reach convergence. Moreover a study about the resilience of the problem will be accomplished, resorting to fault injection techniques.

References

[AAD+05] V. Alessandrov, E. Atanassov, I. Dimov, S. Branford, A. Thandavan, and C. Weihrauch. Parallel hybrid Monte Carlo algorithms for matrix computations problems. *Lecture Notes in Computer Science*, 3516:752–759, 2005.

[Axe96] O. Axelsson. *Iterative Solution Methods*. Cambridge University Press, 1996.

[Ben02] M. Benzi. Preconditioning Techniques for Large Linear Systems: A Survey. *Journal of Computational Physics*, 182:418–477, 2002.

[BMM96] M. Benzi, C.D. Meyer, and Tuma M. A sparse approximate inverse preconditioner for the conjugate gradient method. *SIAM Journal on Scientific Computing*, 17:1135–1149, 1996.

[BT98] M. Benzi and M. Tuma. A sparse approximate inverse preconditioner for nonsymetric linear systems. *SIAM Journal on Scientific Computing*, 19:968–994, 1998.

[DA98] I.T. Dimov and V.N. Alexandrov. A new highly convergent Monte Carlo method for matrix computations. *Mathematics and Computers in Simulation*, (47):165–181, 1998.

[DAK01] I. Dimov, V. Alexandrov, and A. Karaivanova. Parallel resolvent Monte Carlo algorithms for linear algebra problems. *Mathematics and Computers in Simulation*, 55(55):25–35, 2001.

[EMSH14] T.M Evans, S.W. Mosher, S.R. Slattery, and S.P. Hamilton. A Monte Carlo synthetic-acceleration method for solving the thermal radiation diffusion equation. *Journal of Computational Physics*, 258(November 2013):338–358, 2014.

[ESW13] T.M. Evans, S.R. Slattery, and P.P.H Wilson. A spectral analysis of the domain decomposed Monte Carlo method for linear systems. *International Conference on Mathematics and Computational Methods Applied to Nuclear Science & Engineering*, 2013.

[Hal62] J.H. Halton. Sequential Monte Carlo. *Mathematical Proceeding of the Cambridge Philosophical Society*, 58(1):57–58, 1962.

[Hal94] J.H. Halton. Sequential Monte Carlo techniques for the solution of linear systems. *Journal of Scientific Computing*, 9(2):213–257, 1994.

[HMY13] J. Hao, M. Mascagni, and L. Yaohang. Convergence analysis of Markov chain Monte Carlo linear solvers using Ulam-von Neumann algorithm. *SIAM Journal on Numerical Analysis*, 51(4):2107–2122, 2013.

[Li02] L. Li. On the iterative criterion for generalized diagonally dominant matrices. *SIAM Journal in Matrix Analysis and Applications*, 24(1):17–24, 2002.

[Saa03] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, 2003.

[Sla13] S. Slattery. *Parallel Monte Carlo Synthetic Acceleration Methods For Discrete Transport Problems*. PhD thesis, University of Wisconsin-Madison, 2013.