MONTE CARLO METHODS TO SOLVE SPARSE LINEAR SYSTEMS
Report 2

Massimiliano Lupo Pasini

Introduction

In the previous report we have considered two different numerical schemes that resort to a Monte Carlo approach in order to solve a sparse linear system: the Sequential Monte Carlo and the Monte Carlo Synthetic Acceleration (MCSA). We focused in detail on different algebraic splitting and preconditioning to be applied on the iteration matrix of the fixed point scheme. This in order to improve the performances of the system solver, without compromising accuracy and high scalability.

In the current report we try to assert different approaches that can be used to modify the algorithm. At first, we keep moving on the same direction of the previous report by introducing new algebraic devices. In specific we refer to the *alternating methods* to solve the finite difference discretized Laplace problem, appealing to results described in [BS97]. After this, we will consider the possibility to exploit a Monte Carlo approach to explicitly approximate the inverse matrix associated with the system to be solved, accordingly to the work in [AAB$^+$05].

Then we will move to more statistics oriented issues. In fact, different ways to define the transition probability for the Monte Carlo scheme will be considered. For this we resort to [Sri00]. In conclusion, a method will be shown in order to provide statistical upper bounds and lower bounds to estimate the accuracy of the Monte Carlo system solver (*uncertainty quantification*).

Chapter 1

Alternating methods

Let us start by recalling some theoretical results, necessary to guarantee the convergence of the method we are going to use. For this we refer to results show in [BS97].

Take a linear system of the form

$$A\mathbf{x} = \mathbf{b}, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is a nonsingular matrix and $\mathbf{x}$, $\mathbf{b} \in \mathbb{R}^n$. The representation $A = B - C$ is called a splitting if B is nonsingular and it enables the introduction of a fixed-point iterative method

$$\mathbf{x}^{k+1} = T\mathbf{x}^k + \mathbf{c}, \quad k = 0, 1, \cdots$$

where $T = B^{-1}C \in \mathbb{R}^{n\times n}$ is the iteration matrix, $\mathbf{c} = B^{-1}\mathbf{b} \in \mathbb{R}^n$. The initial guess $\mathbf{x}^0$ is assumed to be chosen arbitrarily.

We now introduce consecutively some definitions and result to build up the theoretical setting required for the development of the algorithm.

Definition 1 *Consider a spitting $A = B - C$ and the corresponding iteration matrix $T = B^{-1}C$. The splitting is called P-regular if $B+C$ is positive definite, weak if $T \geq 0$, weak regular if $B^{-1} \geq 0$ and $T \geq 0$, regular if $B^{-1} \geq 0$ and $C \geq 0$.*

Definition 2 *A nonsingular matrix A is called monotone if $A^{-1} \geq 0$.*

Definition 3 *A matrix T is said to be convergent if the powers T^k converge to a limiting matrix as $k \to \infty$. If that limit is the zero matrix, then T is called zero-convergent.*

Theorem 1 *Let A and T be square matrices such that A and $I - T$ are nonsingular. Then, there exists a unique pair of matrices B, C, such that B is nonsingular, $T = B^{-1}C$ and $A = B - C$. The matrices are $B = A(I - T)^{-1}$ and $C = B - A$.*

The definitions and results introduced here above are preliminary to the following **alternating iteration** scheme. Consider an iterative method to solve 1.1 such as

$$\mathbf{x}^{k+\frac{1}{2}} = M^{-1}N\mathbf{x}^k + M^{-1}\mathbf{b}, \quad \mathbf{x}^{k+1} = P^{-1}Q\mathbf{x}^{k+\frac{1}{2}} + P^{-1}\mathbf{b}, \quad k = 0, 1, \cdots \tag{1.2}$$

The scheme 1.2 is composed of two fixed point schemes that alternate each other in providing refined approximations of the solution to 1.1. However the convergence of the single fixed point schemes in 1.2 (which is guaranteed if $\rho(M^{-1}N) < 1$ and $\rho(P^{-1}Q) < 1$) is not enough to ensure the convergence of the total scheme which can be reformulated in a single equation such as

$$\mathbf{x}^{k+1} = P^{-1}QM^{-1}N\mathbf{x}^k + P^{-1}(QM^{-1} + I)\mathbf{b}, \quad k = 0, 1, \cdots$$

This is the reason why the following theorem is needed to sustain our way of proceeding.

Theorem 2 *Let A be a symmetric positive definite matrix. If the splitting $A = M - N = P - Q$ are P-regular, then $T = P^{-1}QM^{-1}N$ is zero-convergent. Therefore, the sequence $\{\mathbf{x}^k\}$ generated by 1.2 converges to the unique solution of 1.1 for any choice of the initial guess $\mathbf{x}^0$. Moreover, the unique splitting induced by the iteration matrix is P-regular.*

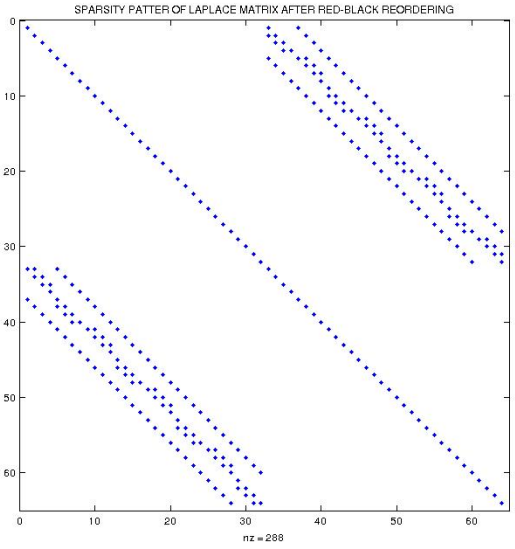

Figure 1.1: Red-black reordering of Laplace matrix $\tilde{A}$.

Now we put all this statements together for the case of our interest. Consider the Laplace problem: $\Omega = (0,1) \times (0,1)$

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

with $f = 1$ in Ω.

By introducing a finite difference discretization with a 5 point stencil, the system we have to solve is

$$A\mathbf{u} = \mathbf{f}. \tag{1.3}$$

$A \in \mathbf{R}^{(n-2)\times(n-2)}$, $\mathbf{u} \in \mathbb{R}^{(n-2)}$, $\mathbf{f} \in \mathbb{R}^{(n-2)}$ and n is the number of nodes used along each edge of the square domain to generate the mesh.

On the linear system 1.3 a red-black reordering is applied. The structure of matrix $\tilde{A}$ is such that

$$\tilde{A} = \begin{bmatrix} D_1 & B \\ C & D_2 \end{bmatrix}$$

. We report again the sparsity pattern of the matrix associated to the discretized Laplace operator after the reordering in Figure 1.1.

The obtained matrix is symmetric and positive definite. Two possible splittings are:

- Gauss-Seidel;
- lower triangular with main diagonal and upper triangular without main diagonal.

Gauss-Seidel splitting is such that

$$M = \begin{bmatrix} D_1 & 0 \\ C & D_2 \end{bmatrix}$$

and

$$N = \begin{bmatrix} 0 & -B \\ 0 & 0 \end{bmatrix}.$$

Asymmetric triangular splitting instead is such that

$$P = \begin{bmatrix} D_1 & 0 \\ C & D_2 \end{bmatrix}$$

and

$$Q = \begin{bmatrix} 0 & B \\ 0 & 0 \end{bmatrix}$$

.

Both the splittings con be proved to be P-regular, thus we can exploit what is stated by the last theorem. Therefore matrix $T = P^{-1}QM^{-1}N$ is zero-convergent. This implies that the scheme 1.2, applied to our case of interest, generates a convergent method.

The alternating method can be inserted in both the Sequential Monte Carlo and the Monte Carlo Synthetic Acceleration as shown in the following graphs.

Data: $A = M - N = P - Q$, $\mathbf{b}$, $\mathbf{x}_0$
Result: x_{num}
$\mathbf{x}_{old} = \mathbf{x}_0$;
while *not reached convergence* **do**
  $\mathbf{r} = M^{-1}\mathbf{b} - M^{-1}N\mathbf{x}_0$;
  $A\delta\mathbf{x} = \mathbf{r}$;
  $\mathbf{x}_{mid} = \mathbf{x}_{old} + \delta\mathbf{x}$;
  $\mathbf{r} = P^{-1}\mathbf{b} - P^{-1}Q\mathbf{x}_0$;
  $A\delta\mathbf{x} = \mathbf{r}$;
  $\mathbf{x}_{new} = \mathbf{x}_{mid} + \delta\mathbf{x}$;
end
$x_{num} = x_{new}$;

Algorithm 1: Sequential Monte Carlo with alternating method

Data: $A = M - N = P - Q$, $\mathbf{b}$, $\mathbf{x}_0$
Result: x_{num}
$\mathbf{x}^l = \mathbf{x}_0$;
$H_1 = M^{-1}N$;
$H_2 = P^{-1}Q$;
while *not reached convergence* **do**
  $\mathbf{x}^{mid_1} = H_1\mathbf{x}^{old} + \mathbf{b}$;
  $\mathbf{r}^{mid_1} = \mathbf{b} - A\mathbf{x}^{mid_1}$;
  $\delta\mathbf{x}^{mid_1} = (I - H_1)^{-1}\mathbf{r}^{mid_1}$;
  $\mathbf{x}^{prel} = \mathbf{x}^{mid_1} + \delta\mathbf{x}^{mid_1}$;
  $\mathbf{x}^{mid_2} = H_2\mathbf{x}^{prel} + \mathbf{b}$;
  $\mathbf{r}^{mid_2} = \mathbf{b} - A\mathbf{x}^{mid_2}$;
  $\delta\mathbf{x}^{mid_2} = (I - H_2)^{-1}\mathbf{r}^{mid_2}$;
  $\mathbf{x}^{new} = \mathbf{x}^{mid_2} + \delta\mathbf{x}^{mid_2}$;
end
$x_{num} = x^{new}$;

Algorithm 2: Monte Carlo Synthetic Acceleration with alternating method

1.1 Numerical results (with n=10)

By setting the discretization step $h = 0.\overline{1}$, we have a 10×10 grid which reduces to a 8×8 for the homogeneous Dirichlet boundary conditions. It means that $A \in \mathbf{R}^{64\times 64)}$, $\mathbf{u} \in \mathbb{R}^{64}$ and $\mathbf{f} \in \mathbb{R}^{64}$.

Both Sequential and MCSA have been run by resorting to the *Forward method* and to the *Adjoint* one. The maximal number of numerical iterations allowed is 50, the tolerance for the stop criterion is $\varepsilon = 10^{-3}$. For the Forward method 100 random walks have been run for each component of the solution vector, instead for the Adjoint method a total number of $64 \cdot 10^2$ is employed. Each random walk has been run for 20 steps. Results are presented in Table 1.1 and 1.2.

	SEQ - Forward method	SEQ - Adjoint method
Nb. random walks	$64 \cdot 10^2$	$64 \cdot 10^2$
Numerical. it.	4	(52)
Relative error	$4.84 \cdot 10^{-4}$	$8.97 \cdot 10^{-4}$
CPU time (s)	19.35	21.31

Table 1.1: Sequential Monte Carlo. Red-black reordering and alternating method for $n = 10$.

	MCSA - Forward method	MCSA - Adjoint method
Nb. random walks	$64 \cdot 10^2$	$64 \cdot 10^2$
Numerical. it.	4	14
Relative error	$3.15 \cdot 10^{-5}$	$7.7 \cdot 10^{-4}$
CPU time (s)	19.9	6.09

Table 1.2: MCSA. Red-black reordering and alternating method for $n = 10$.

As you can notice, for the Sequential Method there is a huge difference between the Forward and the Adjoint approach in terms of numerical iteration used to converge. However this is not reflected by the CPU time which is pretty similar. Moreover, an improvement in terms of performance is achieved with the Synthetic Acceleration. In fact the number of numerical iterations used by the Adjoint method is decreased significantly, with positive implications in terms of CPU time.

1.2 Numerical results (with n=14)

However, as the size of the linear system is slightly increased, such as $n = 14$, thing starts going worse for the Adjoint approach. The numerical setting is the same as for $n = 10$, except for the fact that the Forward method is run with 10^3 random walks for each component. The total number of random walks employed by the Adjoint method instead is $144 \cdot 10^3$.

What happens now is that the Sequential Monte Carlo resorting just to the triangular splitting does not converge anymore. It means that in the alternating method, the alternating iteration with the triangular splitting does not help convergence at all. Thus there is no point in using these alternating scheme anymore. In fact, comparing Forward and Adjoint approach with the Sequential Monte Carlo and just triangular splitting for the fixed point scheme, we have what shown in Table 1.3.

	SEQ - Forward method	SEQ - Adjoint method
Nb. random walks	$144 \cdot 10^3$	$144 \cdot 10^3$
Numerical. it.	6	50
Relative error	$5.23 \cdot 10^{-4}$	$4.2 \cdot 10^{+5}$
CPU time (s)	572.75	-

Table 1.3: Sequential Monte Carlo. Red-black reordering and triangular splitting for $n = 14$.

However, results are fixed by applying the Synthetic acceleration. In fact, as it can be notice in Table 1.4, the triangular splitting becomes convergent again both with the Forward and the Adjoint approach. Thus we are justified in analyzing the alternating scheme as well. This strengthen what already stated in [Sla13], that is the fact that MCSA accelerates the convergence of the numerical scheme with respect to the Sequential Monte Carlo.

	MCSA - Forward method	MCSA - Adjoint method
Nb. random walks	$144 \cdot 10^3$	$144 \cdot 10^3$
Numerical. it.	6	16
Relative error	$3.75 \cdot 10^{-4}$	$3.25 \cdot 10^{-4}$
CPU time (s)	581.31	2041.2

Table 1.4: MCSA. Red-black reordering and triangular splitting for $n = 14$.

It allows to apply alternating scheme as well and results are shown in Table 1.5.

In this case we can see that, equalizing the number of random walks employed in the Forward and the Adjoint methods, the former provides a better performance. This is true both for the number of numerical iterations and for the CPU seconds employed.

	MCSA - Forward method	MCSA - Adjoint method
Nb. random walks	$144 \cdot 10^3$	$144 \cdot 10^3$
Numerical. it.	6	18
Relative error	$4.34 \cdot 10^{-4}$	$7.51 \cdot 10^{-4}$
CPU time (s)	727.5	1185.1

Table 1.5: MCSA. Red-black reordering and alternating scheme for $n = 14$.

1.3 Numerical results (with n=22,32)

By further increasing the number of points to discretize the domain Ω, the adjoint approach compromises the convergence of the scheme both for Sequential Monte Carlo and for MCSA. The Forward Monte Carlo, instead, keeps working fine. Therefore, in the following tables, we present results reached by Forward Sequential Monte Carlo and Forward MCSA with alternating scheme as the size of the problem progresses.

	MCSA - Forward method
Nb. random walks	$400 \cdot 10^3$
Numerical. it.	14
Relative error	$9.45 \cdot 10^{-4}$
CPU time (s)	4126

Table 1.6: MCSA. Red-black reordering and triangular splitting for $n = 22$.

	MCSA - Forward method
Nb. random walks	$900 \cdot 10^3$
Numerical. it.	??
Relative error	??
CPU time (s)	??

Table 1.7: MCSA. Red-black reordering and triangular splitting for $n = 32$.

	MCSA - Forward method
Nb. random walks	$400 \cdot 10^3$
Numerical. it.	16
Relative error	$4.16 \cdot 10^{-4}$
CPU time (s)	4766

Table 1.8: MCSA. Red-black reordering and alternating method for $n = 22$.

	MCSA - Forward method
Nb. random walks	$900 \cdot 10^3$
Numerical. it.	??
Relative error	??
CPU time (s)	??

Table 1.9: MCSA. Red-black reordering and alternating method for $n = 32$.

Chapter 2

Inverse matrix approximation

In Report 1 and in the the previous chapter we have used a Monte Carlo approach to compute the solution vector for a linear system. We will keep on these in the following chapters too. However I would like to introduce now a slight modification of the algorithm that enables to explicitly compute an approximation of A^{-1} in 1.1. The way of proceeding is very similar to the one used to compute the solution vector.

For these we refer to [AAB+05] and [Vaj07]. Assume to start from a linear system such as 1.1.

If $H = I - A \Rightarrow \rho(H) < 1$, then we know that Neumann series expansion is possible such that

$$A^{-1} = \sum_{i=0}^{\infty} A^i.$$

As we already know, the iteration matrix H can be used in order to define a Markovian process. A possible choice for the transition probability P in fact can be

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

Related random variables may be

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

and these can be exploited to build up a new sequence of random variables

$$W_0 = 1, \quad W_j = W_{j-1} w_{i,j}, \quad j = 1, \cdots, i.$$

It can be noticed that the setting is exactly the same used for the definition of the Monte Carlo algorithm to solve linear systems. In that case we had the following formula for the estimation of the j-th component of the solution vector $\mathbf{x}$

$$x_j = \sum_{k=0}^{n} H_{jk}^{-1} b_k$$

By replacing $\mathbf{b}$ with $\mathbf{f}_k = (\underbrace{0, \cdots, 0}_{k-1}, 1, 0, \cdots, 0)$ we have $x_j = A_{jk}^{-1}$. The corresponding unbiased estimator is

$$\theta(A_{jk}^{-1}) = \sum_{i_0 = j, i_l = k | l = 1}^{\infty} W_{i_0 \to i_l}.$$

This make possible for us to estimate each element of the inverse matrix by resorting to statistical estimators.

However the configuration $A = I - H$ with $\rho(H) < 1$ is feasible just in very rare cases. Assume that matrix A is diagonally dominant with $\rho(A) > 1$. If we consider a diagonal matrix $D \in \mathbb{R}^{n \times n}$ such that $D = diag(A)$, we can split matrix $A = D - (D - A)$. Therefore we can introduce

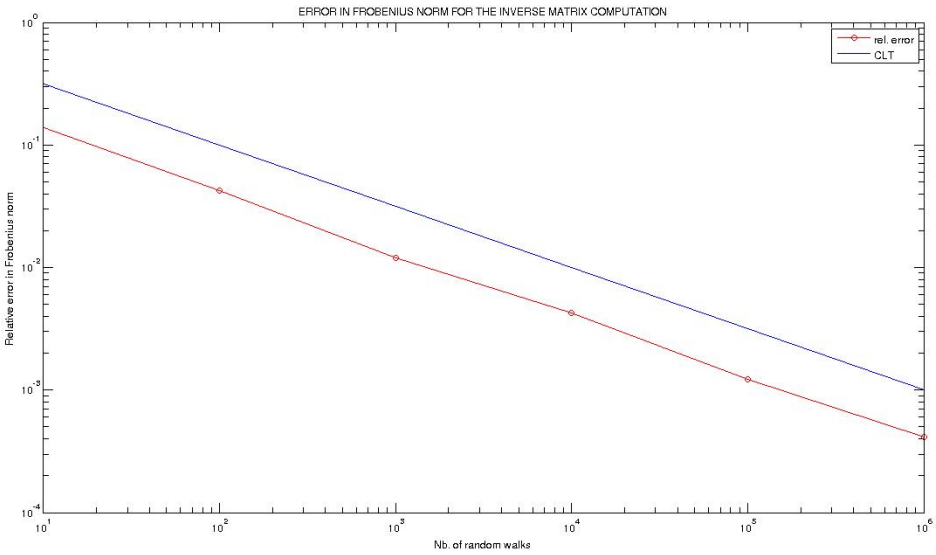

Figure 2.1: Relative error in Frobenius norm for the Monte Carlo inverse matrix A^{-1}.

$H = I - D^{-1}A$ such that $\rho(H) < 1$. Hence system (1.1) can be reinterpreted as a fixed point problem

$$\mathbf{x} = H\mathbf{x} + D^{-1}\mathbf{b}. \tag{2.1}$$

The requirement on H is to be nonsingular. By slightly modifying the scheme introduced before, we can exploit the Markovian process to estimate A^{-1} in such this case as well.

Here we report the pseudo code to do this.

Data: A
Result: x_{num}
Read matrix A;
Set $D_{ii} = A_{ii}$ and for $i \neq j$, $D_{ij} = 0$;
Compute $Y = I - H$ and its inverse by Monte Carlo method;
Compute D^{-1} by $D_{ii}^{-1} = \frac{1}{d_{ii}}$ and for $i \neq j$, $D_{ij} = 0$;
Compute $A^{-1} = Y^{-1}D$;
Return A^{-1};

Algorithm 3: Scheme for the computation of the inverse matrix

The above algorithm produce the Monte Carlo inversion for a given diagonally dominant matrix A.

2.1 Numerical test

For this test case consider a diagonally dominant matrix $A \in \mathbb{R}^{100\times 100}$ such that

$$A = \begin{bmatrix} +\frac{1}{2} & -\frac{1}{16} & 0 & \cdots & \cdots & 0 \\ -\frac{1}{16} & +\frac{1}{2} & -\frac{1}{16} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & \cdots\cdots & -\frac{1}{16} & +\frac{1}{2}. \end{bmatrix}$$

The spectral radius of the matrix is $\rho(A) = 0.624939535286498$. Therefore we can apply the simpler version of the algorithm for the inverse matrix computation without resorting to the diagonal preconditioning. The numerical setting is characterized by a varying number of random walks such that $N = 10^i$ for $i = 1, \cdots, 6$. For each random walk 20 steps are taken. Looking at the error curve as the number of random walks progresses (Figure 2.1), we can see that the theoretical trend imposed by the Central Limit Theorem is respected since the error is $O(\frac{1}{\sqrt{N}})$.

Chapter 3

Variants for the transition probability

As already hinted in the introduction, different ways of defining the transition probability are possible. This will affect the path of the random walks since the weight associated with each state will change. From the theoretical point of view, if each random walk were run without ever stopping it, asymptotic results wouldn't be affected by the particular choice of transition probability.

In fact let us focus on the Forward method for a while. If we consider the analytical expression of the estimators used for the solution vector

$$w_{i,j} = \frac{H_{i,j}}{P_{i,j}}. \tag{3.1}$$

$$W_0 = \frac{h_{k_0}}{\tilde{p}_{k_0}}, \quad W_j = W_{j-1} w_{i,j}, \quad j = 1, \cdots, i. \tag{3.2}$$

It can be proved that

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

and

$$E\Big[\sum_{i=0}^{\infty} W_i b_{k_i}\Big] = (\mathbf{h}, \mathbf{x}).$$

This leads to the formula

$$E\Big[\sum_{l=0}^{\infty} W_l b_{k_l}\Big] = x_i = \sum_{l=0}^{\infty} \sum_{k_1=1}^{n} \sum_{k_2=1}^{n} \cdots \sum_{k_l=1}^{n} P_{k_0,k_1} P_{k_1,k_2} \cdots P_{k_{l-1},k_l} w_{k_0,k_1} w_{k_1,k_2} \cdots w_{k_{l-1},k_l} b_{k_l}. \tag{3.3}$$

By expanding the expression for variables w_{k_{j-1},k_j} in 3.3, we have

$$E\Big[\sum_{l=0}^{\infty} W_l b_{k_l}\Big] = x_i = \sum_{l=0}^{\infty} \sum_{k_1=1}^{n} \sum_{k_2=1}^{n} \cdots \sum_{k_l=1}^{n} P_{k_0,k_1} P_{k_1,k_2} \cdots P_{k_{l-1},k_l} \frac{h_{k_0,k_1}}{P_{k_0,k_1}} \frac{h_{k_1,k_2}}{P_{k_1,k_2}} \cdots \frac{h_{k_{l-1},k_l}}{P_{k_{l-1},k_l}} b_{k_l}. \tag{3.4}$$

It can be noticed that the elements P_{k_{j-1},k_j} appear twice, both in the numerator and in the denominator. Thus they cancel each other and the result is

$$E\Big[\sum_{l=0}^{\infty} W_l b_{k_l}\Big] = x_i = \sum_{l=0}^{\infty} \sum_{k_1=1}^{n} \sum_{k_2=1}^{n} \cdots \sum_{k_l=1}^{n} h_{k_0,k_1} h_{k_1,k_2} \cdots h_{k_{l-1},k_l} b_{k_l}. \tag{3.5}$$

In the last formula no stochastic terms appear, since the transition probability terms canceled each other. The remaining terms are just the elements of the iteration matrix and the elements of the right hand side. This means that the asymptotic result is not affected by the particular choice of transition probability.

From the practical point of view, the random walks are not run for infinitely many steps. However, if the stopping criterion is set properly, it will cut off the random walks when further steps are not relevant anymore. The choice of the transition probability may affect the number

of steps preserved by the stopping criterion. It means that the particular transition probability selected may affect the efficiency of the solver, accordingly to the number of steps used for each random walk.

In [AAB+05] and [Sri00] two different kinds of transition probability are taken into account.

Uniform probabilities

With this approach transition matrix P is such that all the non-zero elements in each row have equal probability of occurring.

$$P_{i,j} = \begin{cases} 0 & \text{if} \quad H_{i,j} = 0 \\ \frac{1}{\#(\text{non-zeros in the row})} & \text{if} \quad H_{i,j} \neq 0 \end{cases}$$

The Monte Carlo approach resorting to this definition of the transition matrix, in accordance to [AAB+05], is called *Uniform Monte Carlo* (UM).

Weighted probabilities

Another way of defining the transition matrix consists in attributing probabilities accordingly to the magnitude of the elements. This is the approach used so far.

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

This way of proceeding, still basing on the work [AAB+05], is called *Monte Carlo Almost Optimal* (MAO).

3.1 Numerical results

In this section we consider a linear system defined as follow. $A \in \mathbb{R}^{500 \times 500}$, $\mathbf{b} \in \mathbb{R}^{500}$. In particular, A is a tridiagonal matrix such that

$$A = \begin{bmatrix} 4 & -1 & 0 & \cdots & \cdots & 0 \\ -1 & 4 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & \cdots\cdots & -1 & +4. \end{bmatrix} \tag{3.6}$$

and $\mathbf{b}$ is chosen such that its components are increasing from 1 to 500 as the index progresses the same way

$$\mathbf{b} = [1, 2, 3, \cdots, 500]^T. \tag{3.7}$$

The goal is to compute the solution to the introduced linear system resorting to both Monte Carlo Forward Method and Monte Carlo Adjoint Method. Both UM and MAO transition matrices are used in order to compare the efficiency in terms of accuracy and employed time. Neither Sequential Monte Carlo nor MCSA scheme are used since we want to stress out the impact of the different ways to define the transition matrix on the convergence rate.

In Figure 3.1 and **??** we have the error behavior for Forward method by resorting to uniform and almost optimal transition matrix respectively. As expected, we can notice that the CLT trend is respected by both the probabilities. It has a sense, since we the asymptotic result is independent of the particular choice made for the distribution.

As it can be noticed, Adjoint method works with expected performance by resorting to Monte Carlo Almost Optimal distribution. In fact the error behavior is in accordance with Central Limit Theorem prediction. However it is not the same for the Uniform Monte Carlo distribution (Figure 3.3). In fact there is a stagnation of the error that sticks around 10^0 and it does not decrease any more as the number of employed random walks progresses. This behavior has been detected also in [Sri00].

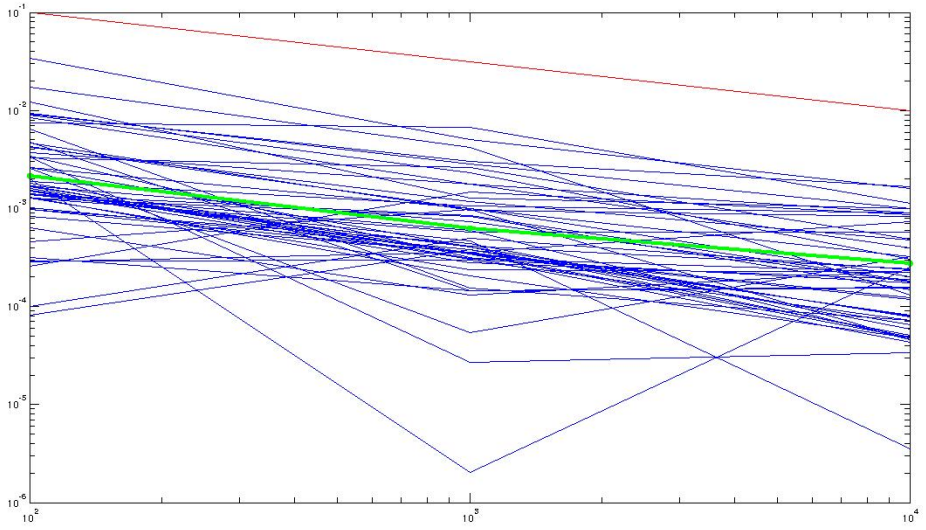

Figure 3.1: Relative error in Euclidean norm for Forward Method with uniform transition probability.

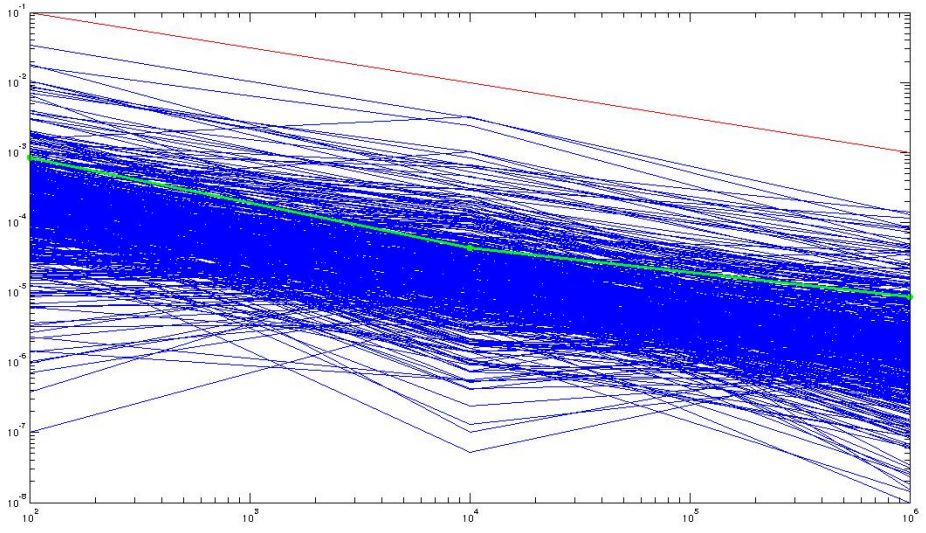

Figure 3.2: Relative error in Euclidean norm for Forward Method with almost optimal transition probability.

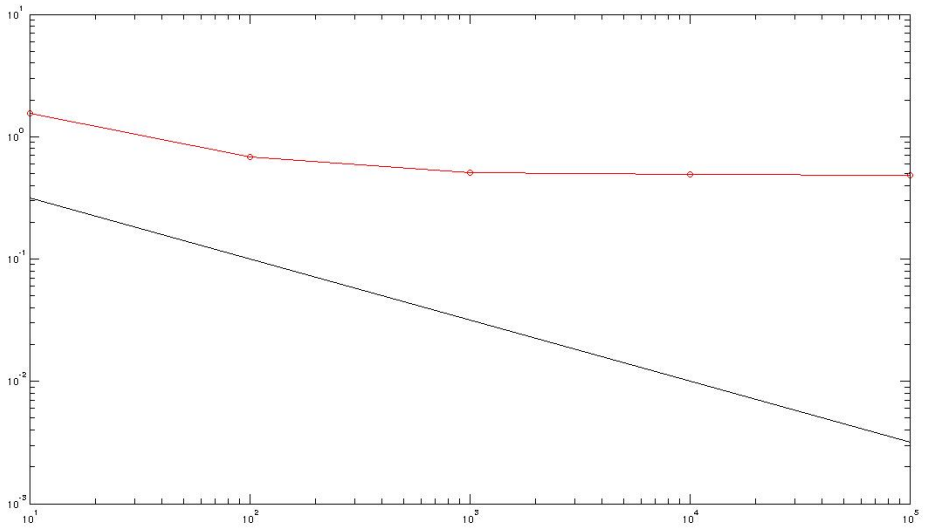

Figure 3.3: Relative error in Euclidean norm for Adjoint Method with uniform transition probability.

Chapter 4

Uncertainty quantification

The last chapter of this report deals with the estimation of the inaccuracy characterizing the Monte Carlo solution to a linear system. Since the system solver is based on a stochastic approach, the error bound estimation will be affected by this stochastic counterpart as well.

At first we provide an estimation of the error as concerns the Monte Carlo Forward Method and Adjoint Method. Then we will move to analyze how this estimations may affect the accuracy of the hybrid schemes such as the Sequential Monte Carlo and the Monte Carlo Synthetic Acceleration.

In order to proceed in this direction, it is necessary to appeal to some theoretical results that will be reminded in the following lines.

Theorem 3 *Central Limit Theorem (CLT) for independent sequences*

Suppose $X_1, X_2, ...$ is a sequence of independent identically distributed (i.i.d.) random variables with $E[Xi] = \mu$ and $Var[X_i] = \sigma^2 < \infty$. Then as $n \to \infty$, the random variables $\sqrt{n}(S_n - \mu)$ converge in distribution to a normal $\mathcal{N}(0, \sigma^2)$:

$$\sqrt{n}\left(\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) - \mu\right) \xrightarrow{\mathcal{L}} \mathcal{N}(0, \sigma^2).$$

Definition 4 *A type I error (or error of the first kind) is the incorrect rejection of a true null hypothesis. With respect to the non-null hypothesis, it represents a false positive.*

Asymptotic Confidence Intervals (ACI)

Assume all the hypotheses required for the CLT hold. Given α the error of the first kind, we can build a bilateral asymptotic confidence interval (ACI) with confidence level $(1 - \alpha)$ as follows

$$ACI_{(1-\alpha)} = \left\{ x \in \mathbb{R} \quad s.t. \quad |x - \overline{x}| < \frac{\sigma}{\sqrt{n}}\, q_{(1-\frac{\alpha}{2})} \right\} \tag{4.1}$$

where $q_{(1-\frac{\alpha}{2})}$ is the quantile of the standard normal of order $(1 - \frac{\alpha}{2})$. $\overline{x}$ instead represents the sample mean defined as

$$\overline{x} = \frac{S_n}{n}.$$

The dividing factor on α by 2 in 4.1 is necessary, in order to split the type I error both on the left and right hand side of the sample mean.

The number n represents the size of the sample data. In our cases of interest it will be replaced by the number of random walks N.

Since the theoretical variance of the estimator θ for the solution vector is not known, we replace it with the sample variance.

Definition 5 *Suppose X is a random variable with finite first and second moment. Assuming $x_1, x_2, ...$ is a sample dataset consisting in n realization of X, the sample variance is defined as*

$$\overline{s} = \frac{1}{(n-1)} \sum_{i=1}^{n} (x_i - \overline{x})^2$$

where $\overline{x}$ is the sample mean.

In the definition of $\overline{s}$ we have divided by $(n-1)$ instead of n in order to build an unbiased estimator for the variance.

In Appendix B of [Sla13] there are formulas of the variance of the solution estimator for both the Forward and the Adjoint methods.

4.1 Forward Estimator Variance

In the Forward method the estimator built on the Markovian process computes each component of the solution vector $\mathbf{x}$ at a time. Therefore it is possible to estimate the variance associated with each component x_i. By definition of the variance for a random variable

$$\sigma_i^2 = Var[x_i] = E[x_i^2] - (E[x_i])^2. \tag{4.2}$$

In our case the estimator θ_i is conceived as the expected value of random variables X_i defined on random walks

$$\theta_i = E[X_j] = \sum_{\nu} P_\nu X_j(\nu), \quad j = 1, \cdots, n$$

where n represents the size of the solution vector to be computed, ν is the single permutation of a random walk and P_ν is the probability associated with the specific permutation.

We remind that the random variable X_i associated with a permutation of the first k steps of the random walk is

$$X_i = \sum_{m=0}^{k} W_m b_{i_m}. \tag{4.3}$$

By expanding the square of the summation in 4.2 plugging in 4.3, we have

$$Var[\theta_i] = \sum_{\nu} P_\nu \Bigg(\sum_{m=0}^{k} \Big[W_m^2 b_{i_m}^2 + 2 \sum_{j=0}^{m-1} W_m W_j b_{i_m} b_{i_j} \Big] \Bigg) - x_i^2.$$

Reminding 3.1 and 3.2, we have

$$\begin{aligned} Var[\theta_i] = \sum_{k=0}^{\infty} \sum_{i_1}^{N} \sum_{i_2}^{N} \cdots \sum_{i_k}^{N} p_{i,i_1} p_{i_1,i_2} \cdots p_{i_{k-1},i_k} \Big[& w_{i,i_1}^2 w_{i_1,i_2}^2 \cdots w_{i_{k-1},i_k}^2 b_{i_k}^2 + \\ & + 2 \sum_{j=0}^{k-1} w_{i,i_1} w_{i_1,i_2} \cdots w_{i_{k-1},i_k} b_{i_k} w_{j,j_1} w_{j_1,j_2} \cdots w_{j_{k-2},j_{k-1}} b_{j_{k-1}} \Big] - x_i^2. \end{aligned}$$

4.2 Adjoint Estimator Variance

In [Sla13] it is called *Collision Estimator Variance*. By proceeding with an approach similar to the one used for the direct method, we can compute the variance associated with the Adjoint Monte Carlo estimator. Still referring to [Sla13] we have

$$Var[\theta_i] = \sum_{\nu} P_\nu \sum_{m=0}^{k} W_m^2 \delta_{i_m,i} + 2 \sum_{\nu} P_\nu \sum_{m=0}^{k} \sum_{l=0}^{m-1} W_m W_l \delta_{i_m,i} \delta_{i_l,i} - x_i^2. \tag{4.4}$$

The delta functions $\delta_{i_m,i}$ and $\delta_{i_l,i}$ will be nonzero just for random walks that are in the current solution state ($i_m = i_l = i$). All the other states visited by the random walks during their history do not count to determine the variance of the current state. This is the substantial difference between

the Forward and the Adjoint method. Therefore variances associated with different components of the solution vector are not correlated one to the other. The formula for the variance of the estimator θ_i associated with the i-th component of the solution vector is

$$Var[\theta_i] = \sum_{k=0}^{\infty} \sum_{i_1}^{N} \sum_{i_2}^{N} \cdots \sum_{i_k}^{N} p_{i_k,i_{k-1}} \cdots p_{i_2,i_1} p_{i_1,i_0} \Big[w_{i_k,i_{k-1}}^2 \cdots w_{i_2,i_1}^2 w_{i_1,i_0}^2 b_{i_0}^2 \delta_{i_k,i} + \\ + 2 \sum_{l=0}^{k-1} w_{i_k,i_{k-1}} \cdots w_{i_2,i_1} w_{i_1,i_0} w_{l_{k-1},l_{k-2}} \cdots w_{l_1,l_0} \delta_{i_k,i} \delta_{l_{k-1},i} \Big] - x_i^2. \tag{4.5}$$

Both Forward Estimation Variance and Adjoint Estimation Variance supply us with a method to compute the variance associated with each component of the solution vector. In fact the variance of the estimator θ_i, from a practical, point of view, can be computed such as

$$Var[\theta_i] = \frac{1}{N} Var[X_i]$$

where N is the total number of random walks employed.

For the sake of clarity we will refer to σ_i as the variances associated with variables X_i

$$\sigma_i = Var[X_i].$$

Since, from a practical point of view, the random walks will be stopped after a finite number of steps, we will deal with an approximation of estimators θ_i defined as

$$\tilde{\theta}_i = E\Big[\sum_{l=0}^{m} W_l b_{k_l} \Big] = x_i = \sum_{l=0}^{m} \sum_{k_1=1}^{n} \sum_{k_2=1}^{n} \cdots \sum_{k_l=1}^{n} P_{k_0,k_1} P_{k_1,k_2} \cdots P_{k_{l-1},k_l} w_{k_0,k_1} w_{k_1,k_2} \cdots w_{k_{l-1},k_l} b_{k_l}.$$

By employing $\tilde{\theta}_i$ for the computation of the asymptotic confidence intervals we have

$$ACI_{(1-\alpha)}(\theta_i) = \Big\{ x \in \mathbb{R} \quad s.t. \quad |x - \tilde{\theta}_i| < \frac{\sigma_i}{\sqrt{N}} \, q_{(1-\frac{\alpha}{2})} \Big\}. \tag{4.6}$$

Now that the estimations of the variance for the Forward and Adjoint methods are provided, we can exploit them in order to analyze the uncertainty at each numerical iteration for the *Sequential Monte Carlo* and *Monte Carlo Synthetic Acceleration*.

4.3 Sequential Monte Carlo Variance

Consider e linear system such as 1.1. It can be reviewed in a fixed point scheme

$$\mathbf{x} = H\mathbf{x} + \mathbf{f}. \tag{4.7}$$

We report here the algorithm

Data: H, $\mathbf{f}$, $\mathbf{x}_0$
Result: x_{num}
$B = I - H$;
$\mathbf{x}_{old} = \mathbf{x}_0$;
while *not reached convergence* **do**
  $\mathbf{r} = \mathbf{f} - B\mathbf{x}_{old}$;
  $B\delta\mathbf{x} = \mathbf{r}$;
  $\mathbf{x}_{new} = \mathbf{x}_{old} + \delta\mathbf{x}$;
end
$x_{num} = x_{new}$;

Algorithm 4: Sequential Monte Carlo

In the following lines we provide a proof bu induction to compute the variance associated with the k-th numerical iteration of the Sequential Monte Carlo method.

Proof - Base case

Let us introduce a vector $\sigma \in \mathbb{R}^n$, where n is the size of the solution vector to the linear system, such that

$$\boldsymbol{\sigma}_i = \sigma_i.$$

At the first iteration of the Sequential algorithm we have

$$\mathbf{r}^0 = \mathbf{f} - B\mathbf{x}^0.$$

Thus the updating of the solution is

$$\delta\mathbf{x}^0 = (I-H)^{-1}\mathbf{r}^0 \pm \frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})},$$

where $\boldsymbol{\sigma}^i$ is the variance vector associated with the solution at the i-th iteration of the numerical scheme. Therefore, at the end of the first step, we have

$$\mathbf{x}^1 = \mathbf{x}^0 + \delta\mathbf{x} = \mathbf{x}^0 + (I-H)^{-1}\mathbf{r}^0 \pm \frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}.$$

At the second step the residual attains the value

$$\mathbf{r}^1 = \mathbf{f} - B\mathbf{x}^1 = \mathbf{f} - B(\mathbf{x}^0 + (I-H)^{-1}\mathbf{r}^0 \pm \frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}).$$

By defining $\tilde{\mathbf{r}}^1 = \mathbf{f} - B(\mathbf{x}^0 + (I-H)^{-1}\mathbf{r}^0$ as the deterministic part of the residual we can write

$$\mathbf{r}^1 = \tilde{\mathbf{r}}^1 \mp B\frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}.$$

This yield the updating at the second step

$$\begin{aligned}\delta\mathbf{x}^1 =& (I-H)^{-1}\tilde{\mathbf{r}}^1 \mp (I-H)^{-1}B\frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})} \pm \frac{\boldsymbol{\sigma}^1}{\sqrt{N}} q_{(1-\frac{\alpha}{2})} = \\ &= (I-H)^{-1}\tilde{\mathbf{r}}^1 \mp \frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})} \pm \frac{\boldsymbol{\sigma}^1}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}\end{aligned}$$

and therefore

$$\mathbf{x}^2 = \mathbf{x}^0 + (I-H)^{-1}\tilde{\mathbf{r}}^1 \mp \frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})} \pm \frac{\boldsymbol{\sigma}^1}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}$$

Proof - Induction step

Assume that at the $(n-1)$-th step the approximation to $\mathbf{x}$ is

$$\mathbf{x}^{n-1} = (I-H)^{-1}\tilde{\mathbf{r}}^{n-2} \pm \sum_{i=1}^{n-1}(-1)^{i-1}\frac{\boldsymbol{\sigma}^{n-1-i}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}$$

where $\tilde{\mathbf{r}}^{n-2}$ contains all the updating contributions associated with the previous numerical iterations. We do not write this terms explicitly because we want just to focus on the behavior of the stochastic term, which is the rightmost in the right term. The computation of the residual is

$$\mathbf{r}^{n-1} = \mathbf{f} - B(I-H)^{-1}\tilde{\mathbf{r}}^{n-2} \mp \sum_{i=1}^{n-1} B\frac{\boldsymbol{\sigma}^{n-1-i}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}$$

and it takes to the new updating terms

$$\delta\mathbf{x} = (I-H)^{-1}\mathbf{f} - (I-H)^{-1}B(I-H)^{-1}\mathbf{r}^{n-2} \mp \sum_{i=1}^{n-1}(-1)^{i-1}(I-H)^{-1}B\frac{\boldsymbol{\sigma}^{n-1-i}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})} \pm \frac{\boldsymbol{\sigma}^{n-1}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}.$$

Therefore

$$\mathbf{x}^n = (I-H)^{-1}\tilde{\mathbf{r}}^{n-1} \pm \sum_{i=1}^{n}(-1)^{i-1}\frac{\boldsymbol{\sigma}^{n-i}}{\sqrt{N}}q_{(1-\frac{\alpha}{2})}$$

which terminates the proof by induction.

By exploiting the triangle inequality we can say that at the n-th iteration of the Sequential method

$$ACI_{(1-\frac{\alpha}{2})}(\mathbf{x}) \subset \left\{\mathbf{y}\in\mathbb{R}^n \quad s.t. \quad \left|y_j - x^n_{det,j}\right| < \sum_{i=1}^{n}\frac{\sigma_j^{n-i}}{\sqrt{N}}q_{(1-\frac{\alpha}{2})}, \quad j=1,\cdots,n\right\}$$

where $\mathbf{x}^n_{det}$ represents the deterministic part of the updating term of the numerical scheme.

4.4 MCSA variance

Starting again from the fixed point formulation 4.7, we remind the MCSA scheme

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 5: Monte Carlo Synthetic Acceleration

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

The proof of the variance associated with the solution vector $\mathbf{x}^k$ at the k-th step of the numerical scheme is very similar to the one used for the Sequential method. We refer to the Appending as concerns th details of the proof. Here we report just the final result

$$\mathbf{x}^n = (I-H)^{-1}\tilde{\mathbf{r}}^{n-1} \pm \sum_{i=1}^{n}(-1)^{i-1}H^{i-1}\frac{\boldsymbol{\sigma}^{n-i}}{\sqrt{N}}q_{(1-\frac{\alpha}{2})}.$$

By resorting to the triangle inequality again we have

$$\mathbf{x}^n = (I-H)^{-1}\tilde{\mathbf{r}}^{n-1} \pm \sum_{i=1}^{n}H^{i-1}\frac{\boldsymbol{\sigma}^{n-i}}{\sqrt{N}}q_{(1-\frac{\alpha}{2})}.$$

It is very important to notice the new multiplicative term H^{i-1} in front of the variance $\frac{\boldsymbol{\sigma}^{n-i}}{\sqrt{N}}q_{(1-\frac{\alpha}{2})}$. Since the iteration matrix H is supposed to be a contractive in order for the fixed point scheme to converge, it means that the variance associated with the MCSA is always expected to be smaller than the one associated with the Sequential method.

4.5 Numerical Results

4.5.1 Test case 1

In this section we still consider the linear system made of matrix and right hand side as in 3.6 and 3.7. This problem is solved both with Forward and Adjoint Monte Carlo without any accelerating scheme. In order to compute the confidence band, the error of the first kind is set such that $\alpha = 0.05$.

In order to quantify the effectiveness of the confidence band, we can compute its relative width as

$$width_{rel} = \frac{2q_{1-\frac{\alpha}{2}}\sum_{i=1}^{n}\sigma_i}{\|\mathbf{x}\|_2} \tag{4.8}$$

where **x** is the solution computed with the back-slash command.

Confidence intervals is compared with the values attained by the solution too the linear system computed with the back-slash command in `Matlab`.

In the case of the Forward method all the components of the solution vector computed with back-slash command sits inside the confidence interval. The relative width of the band is 0.0655.

In the case of the Adjoint method 40 components out of 500 computed with back-slash sits out the confidence band. In this case not just the committed error but also the uncertainty is higher, since the relative bandwidth is 0.1052.

4.5.2 Test case 2

In this section we still consider Laplace problem with finite differences discretization. The goal is to analyze the trend of the variance with respect to the Sequential Monte Carlo and the Monte Carlo Synthetic acceleration. Different splitting of the iteration matrix are analyzed in order to detect a likely optimal choice in terms of uncertainty reduction. The size $(n-2) \times (n-2)$ of the discretized Laplace operator is set such that $n = 14$. This value is chosen because is the biggest one for which both Forward and Adjoint method provide a solution at a reasonable time.

All the different splitting are tested. Thus we can analyze how they affect not just the deterministic component of the algorithm, but also the stochastic one in terms of uncertainty quantification.

For all the cases presented the type I error is set such that $\alpha = 0.025$. It means that we want to take a smaller risk than before of giving a wrong estimation. All the times we check that the numerical solution computed with the back-slash command sits in or not in the confidence interval.

Diagonal splitting

As concerns the Sequential Method with Adjoint estimation, all the components of the numerical solution are inside of the confidence band. The relative amplitude of the confidence band is 0.1323.

By resorting to the MCSA with Adjoint estimation we still have all the components of the solution vector inside of the confidence band which has a relative width equal to 0.087.

Triangular splitting

By resorting to the Sequential Method with Adjoint scheme we know that the algorithm does not converge. It has a statistical counterpart looking at the variance. In Figure 4.1 we have the plot of the reference solution (red dots) and the lower and upper bounds (green dots). You can see that the variance is not able to provide a useful estimation of the band where the true solution is bound. It looks like the divergence of the numerical scheme is reflected by a divergence of the variance associated with the statistical component of the algorithm. The relative amplitude of the confidence band now is $1.07 \cdot 10^{+7}$.

As already said previously, things work better by replacing Sequential Monte Carlo scheme with the MCSA one. We have already shown details about the convergence. As concerns the relative amplitude of the statistical bound we have a decrease to 5.005.

Gauss Seidel splitting

Applying the Adjoint Sequential Method for the resolution of the linear system after having employed a Gauss-Seidel splitting, the method converges and the relative amplitude of the confidence interval is 13.59.

By resorting to MCSA the width of the confidence band decrease remarkably to 1.699. It has a sense because MCSA always implied a noticeable decrease of the number of iterations required in comparison to Sequential Monte Carlo.

In general, the fact that the relative width of the band is pretty high with this splitting fr the iteration matrix is reflected by the fact that the scheme converges very slowly. Indeed in Report 1 we stressed out this fact. It is partially related to the pattern of the preconditioned iteration matrix. The fact that a lot of columns are full of zeros causes the sudden death for a huge number of random walks, without giving a contribution for the estimation of the solution vector.

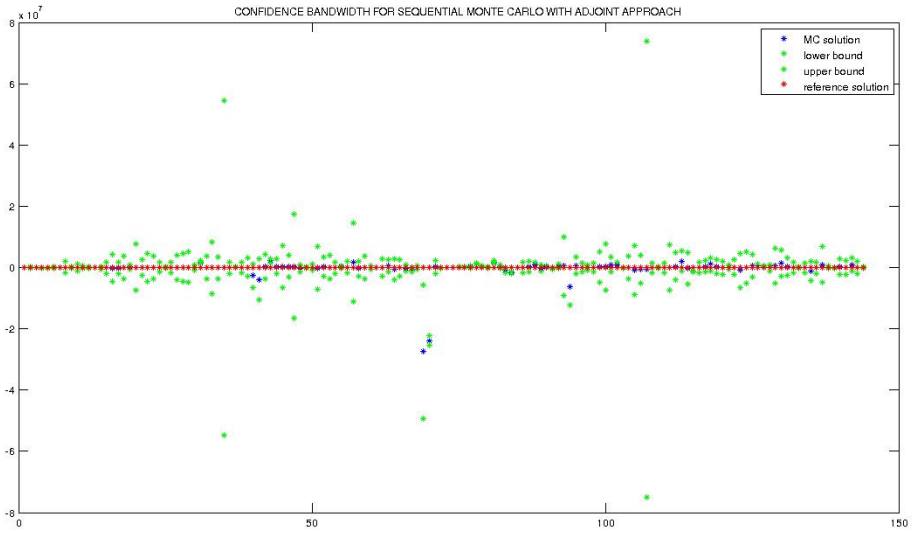

Figure 4.1: Confidence band for the solution to Laplace problem with Adjoint Sequential Monte Carlo.

Block triangular preconditioning

This is the nicest approach that improves considerably the performance of the linear system solver. Therefore we expect it to be reflected by a small variance.

In the case of the Forward Sequential scheme the estimation is very precise and the confidence interval is characterized by a relative width equal to 0.086. By resorting to the Synthetic Acceleration it goes down to $5.028 \cdot 10^{-8}$.

For the Adjoint Sequential approach we have a relative width of the band which is equal to 1.19. By employing the MCSA instead we have a remarkable improvement, since the relative amplitude drops down to $4.5 \cdot 10^{-17}$.

We still notice a breakthrough replacing the Sequential Method with the Synthetic Acceleration. Moreover, considering the reasoning introduced previously, the way the variance decreased is justified by the fact that the spectral radius of the preconditioned iteration matrix is zero.

Appendix

4.6 MCSA variance - Proof by induction

Proof - Base case

At the first step of the numerical scheme we start from the initial guess $\mathbf{x}^0$ and

$$\mathbf{x}^{\frac{1}{2}} = H\mathbf{x}^0 + \mathbf{f}$$

$$\mathbf{r}^{\frac{1}{2}} = \mathbf{f} - B(H\mathbf{x}^0 + \mathbf{f}) = \mathbf{f} - BH\mathbf{x}^0 - B\mathbf{f}$$

Thus the increment to be added to the initial guess is

$$\delta\mathbf{x}^{\frac{1}{2}} = (I - H)^{-1}(\mathbf{f} - BH\mathbf{x}^0 - B\mathbf{f}) \pm \frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}.$$

Hence

$$\mathbf{x}^1 = \mathbf{x}^{\frac{1}{2}} + (I - H)^{-1}(\mathbf{b} - BH\mathbf{x}^0 - B\mathbf{f}) \pm \frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}.$$

For the sake of simplicity we gather all the deterministic terms such that

$$\tilde{\mathbf{x}}^1 = \mathbf{x}^{\frac{1}{2}} + (I - H)^{-1}(\mathbf{b} - BH\mathbf{x}^0 - B\mathbf{f})$$

and

$$\mathbf{x}^1 = \tilde{\mathbf{x}}^1 \pm \frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}.$$

At the second step we have

$$\mathbf{x}^{1+\frac{1}{2}} = H\tilde{\mathbf{x}}^1 \pm H\frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}$$

and the residual is

$$\mathbf{r}^{1+\frac{1}{2}} = \mathbf{f} - B\left(H\tilde{\mathbf{x}}^1 \pm H\frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}\right)$$

$$\mathbf{r}^{1+\frac{1}{2}} = \mathbf{f} - BH\tilde{\mathbf{x}}^1 \mp BH\frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}.$$

The increment to the solution at the second step is

$$\delta\mathbf{x}^{1+\frac{1}{2}} = (I - H)^{-1}\left[\mathbf{f} - BH\tilde{\mathbf{x}}^1 \mp BH\frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}\right] \pm \frac{\boldsymbol{\sigma}^1}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}.$$

This implies that the approximated solution at the second step is

$$\mathbf{x}^2 = \tilde{\mathbf{x}}^2 \mp H\frac{\boldsymbol{\sigma}^0}{\sqrt{N}} q_{(1-\frac{\alpha}{2})} \pm \frac{\boldsymbol{\sigma}^1}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}.$$

4.6.1 Proof - Induction step

Assume that at the $(n-1)$-th numerical step the computed solution is

$$\mathbf{x}^{n-1} = \tilde{\mathbf{x}}^{n-1} \pm \sum_{i=1}^{n-1} (-1)^{i-1} (H^{i-1}) \frac{\boldsymbol{\sigma}^{n-1-i}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}$$

Then

$$\mathbf{x}^{n-1+\frac{1}{2}} = H\mathbf{x}^{n-1} + \mathbf{f} = H\tilde{\mathbf{x}}^{n-1} \pm \sum_{i=1}^{n-1} (-1)^{i-1} H^i \frac{\boldsymbol{\sigma}^{n-1-i}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})} + \mathbf{f}.$$

The residual is

$$\begin{aligned}\mathbf{r}^{n-1+\frac{1}{2}} =& \mathbf{f} - B\mathbf{x}^{n-1+\frac{1}{2}} = \mathbf{f} - B\left(H\tilde{\mathbf{x}}^{n-1} \pm \sum_{i=1}^{n-1} (-1)^{i-1} H^i \frac{\boldsymbol{\sigma}^{n-1-i}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})} + \mathbf{f} \right) = \\ &= \mathbf{f} - B(H\tilde{\mathbf{x}}^{n-1} + \mathbf{f}) \pm \sum_{i=1}^{n-1} (-1)^{i} B H^i \frac{\boldsymbol{\sigma}^{n-1-i}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}\end{aligned}$$

The increment to be added to the solution vector is

$$\begin{aligned}\delta\mathbf{x}^{n-1+\frac{1}{2}} =& (I-H)^{-1}(\mathbf{f} - B(H\tilde{\mathbf{x}}^{n-1} + \mathbf{f})) \pm \sum_{i=1}^{n-1} (-1)^{i} (I-H)^{-1} B H^i \frac{\boldsymbol{\sigma}^{n-1-i}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})} + \frac{\boldsymbol{\sigma}^{n}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})} = \\ &= (I-H)^{-1}(\mathbf{f} - B(H\tilde{\mathbf{x}}^{n-1} + \mathbf{f})) \pm \sum_{i=1}^{n} (-1)^{i} H^i \frac{\boldsymbol{\sigma}^{n-i}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}.\end{aligned}$$

Therefore

$$\mathbf{x}^{n} = \tilde{\mathbf{x}}^{n} \pm \sum_{i=1}^{n} (-1)^{i-1} (H^{i-1}) \frac{\boldsymbol{\sigma}^{n-i}}{\sqrt{N}} q_{(1-\frac{\alpha}{2})}$$

which terminates the proof.

Bibliography

[AAB+05] V. Alessandrov, I. Atanassov, E. amd 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.

[BS97] M. Benzi and B. Szyld. Existence and uniqueness of splittings for stationary iterative methods with applications to alternating methods. *Numerische Matematik*, 76:309–321, 1997.

[BU] M. Benzi and B. Ucar. Product preconditioning for Markov chain problems.

[BU07] M. Benzi and B. Ucar. Block trianguar preconditioners for M-matrices and Markov chains. *Electronic Transaction on Numerical Analysis*, 26:209–227, 2007.

[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.

[ESW02] T.M. Evans, S.R. Slattery, and P.P.H Wilson. Mixed Monte Carlo parallel algorithms for matrix computation. *International Conference on Computational Science 2002*, 2002.

[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.

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

[Sri00] A. Srinivasan. Monte Carlo linear solvers with non-diagonal splitting. *Mathematics and Computer in Simulation*, 80:1133–1143, 2000.

[Vaj07] B. F. Vajargah. Different stochastic algorithms to obtain matrix inversion. *Applied Mathematics and Computation*, 189:1841–1846, 2007.