

**МЕТОД МОНТЕ-КАРЛО И АСИНХРОННЫЕ ИТЕРАЦИИ\****А. В. Дмитриев, С. М. Ермаков*

Санкт-Петербургский государственный университет,  
Российская Федерация, 199034, Санкт-Петербург, Университетская наб., 7-9

Асинхронные методы, так же как и методы Монте-Карло, играют важную роль при многопроцессорных вычислениях, и исследование связей между ними, а в особенности расширенные области их применимости, являются важными задачами. Рассматривается система вида  $x = Ax + f$ , где  $x$  — неизвестный вектор-столбец длины  $n$ ,  $A$  — матрица  $n \times n$ ,  $f$  — вектор правых частей длины  $n$ . Случай, когда первое собственное число матрицы  $|A|$  по модулю меньше единицы, хорошо изучен. В этом случае установлена тесная связь между асинхронными итерациями и методом Монте-Карло. Если же первое собственное число матрицы  $|A|$  по модулю больше единицы, в то время как первое собственное число матрицы  $A$  по модулю меньше единицы, то схожие методы оказываются неприменимыми. Предложены модификации методов Монте-Карло и асинхронных итераций с частичной синхронизацией для решения задачи в указанном случае. Получены оценки погрешностей предложенных алгоритмов и условия сходимости предложенных методов. Проведены численные эксперименты. Библиогр. 6 назв. Ил. 4.

*Ключевые слова:* Метод Монте-Карло, асинхронные методы, асинхронные итерации, параллельные алгоритмы, статистическое моделирование.

**Введение.** Необходимость синхронизации операций, как известно, приводит к простому отдельным элементам оборудования многопроцессорных вычислительных комплексов. Это обстоятельство привлекло внимание исследователей к алгоритмам, не требующим синхронизации, так называемым асинхронным алгоритмам. В частности, были предложены асинхронные варианты методов итераций для решения систем уравнений (см. [1, 2]). Естественными свойствами асинхронности обладают также многие разновидности метода Монте-Карло для решения систем уравнений, и представляет интерес проследить связи и общность свойств этих разновидностей и детерминированных асинхронных методов, упомянутых выше.

Заметим также, что требование асинхронности налагает определенные ограничения на класс решаемых задач. Мы покажем ниже, как эти ограничения можно ослабить за счёт частичной синхронизации. Оказывается, что во всех случаях методика частичной синхронизации является общей, хотя метод Монте-Карло позволяет, вообще говоря, использовать больший арсенал приемов конструирования асинхронных алгоритмов.

Ограничимся рассмотрением методов итераций для решения систем линейных алгебраических уравнений вида

$$x = Ax + f, \quad (1)$$

где  $x = (x_1, x_2, \dots, x_n)^T$  — вектор-столбец неизвестных,  $f = (f_1, f_2, \dots, f_n)^T$  — вектор правых частей и  $A = \|a_{ij}\|_{i,j=1}^n$  — матрица системы.

Введем следующие обозначения: компоненты вектора  $x$  из  $\mathbb{R}^n$  будем обозначать  $x_i$ ,  $i = 1, \dots, n$ , а последовательность векторов из  $\mathbb{R}^n$  будем обозначать  $x(j)$ ,  $j = 0, 1, \dots$

Будем использовать следующие нормы:

---

\*Работа выполнена при финансовой поддержке РФФИ в рамках научного проекта (грант № 14-01-00271-а).

- для вектора  $x = (x_1, \dots, x_n)^T$

$$\|x\| = \max_{1 \leq i \leq n} |x_i|, \quad (2)$$

- для матрицы  $A = \|a_{ij}\|_{i,j=1}^n$

$$\|A\| = \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}|. \quad (3)$$

**1. Асинхронные итерации.** В [2] Боде ввел определение асинхронных итераций для решения систем (в том числе и нелинейных) уравнений. Перепишем это определение для линейного случая.

**Определение 1.** *Асинхронными итерациями для решения системы уравнений (1) с начальным вектором  $x(0)$  называется последовательность  $x(j)$ ,  $j = 0, 1, \dots$ , векторов из  $\mathbb{R}^n$ , определяемых соотношением*

$$x_i(j) = \begin{cases} x_i(j-1), & \text{если } i \notin J_j, \\ \sum_{k=1}^n a_{ik} x_k(s_k(j)) + f_i, & \text{если } i \in J_j, \end{cases} \quad (4)$$

где  $\mathcal{J} = \{J_j | j = 1, 2, \dots\}$  — последовательность непустых подмножеств  $\{1, 2, \dots, n\}$  и  $\mathcal{S} = \{(s_1(j), s_2(j), \dots, s_n(j)) | j = 1, 2, \dots\}$  — последовательность из  $\mathbb{N}^n$ . Причем  $\mathcal{J}$  и  $\mathcal{S}$  удовлетворяют следующим условиям:

- $s_i(j) \leq j - 1$  для  $j = 1, 2, \dots$ ,
- $s_i(j)$  является функцией от  $j$  и стремится к бесконечности, когда  $j$  стремится к бесконечности,
- $i$  появляется бесконечно число раз в в множествах  $J_j$ ,  $j = 1, 2, \dots$

Асинхронные итерации для задачи (1), начинающиеся с  $x(0)$ , для определенных  $\mathcal{J}$ ,  $\mathcal{S}$  обозначим пятеркой  $(A, f, x(0), \mathcal{J}, \mathcal{S})$ .

Определению асинхронных итераций можно дать следующую интерпретацию. Пусть вычисления осуществляются на многопроцессорной системе и  $t_1 < t_2 < t_3 < \dots < t_j < \dots$  — моменты времени, в которые система производит вычисление очередной итерации. В момент времени  $t_j$  один из процессоров вычисляет значение итерации  $x(j)$ , которая отличается от  $x(j-1)$  набором компонент  $\{x_i | i \in J_j\}$ . Эти компоненты вычисляются на основе компонент, полученных на предыдущих итерациях. Последовательность  $\mathcal{S}$  определяет номера предыдущих итераций и компоненты, используемые для вычисления  $x(j)$ . Выбор компонент может быть обусловлен любым критерием, в том числе самым естественным — всегда использовать значения недавно вычисленных компонент.

Такая схема подразумевает отсутствие синхронизации между процессорами, однако для сходимости асинхронных итераций к решению задачи (1) при любом выборе  $x(0)$ ,  $\mathcal{J}$ ,  $\mathcal{S}$  необходимо, чтобы матрица  $A$  удовлетворяла определенным условиям (см. [1, 2]).

**Теорема 1.** *Произвольные асинхронные итерации  $(A, f, x(0), \mathcal{J}, \mathcal{S})$  сходятся к решению задачи (1) тогда и только тогда, когда*

$$\lambda_1(|A|) < 1, \quad (5)$$

где  $|A|$  — матрица, составленная из модулей элементов матрицы  $A$ , а  $\lambda_1(|A|)$  — первое собственное число матрицы  $|A|$ .

Таким образом, если обычный (синхронный) процесс сходится —  $|\lambda_1(A)| < 1$ , но  $\lambda_1(|A|) > 1$ , то по крайней мере асинхронные итерации некоторого вида обязательно расходятся. Можно попытаться исправить положение, осуществляя после некоторой группы асинхронных итераций определенное количество синхронных, которые уменьшают ошибку (частичная синхронизация).

Будем далее рассматривать алгоритм, который после каждых  $m$  асинхронных итераций использует  $l$  обычных. Очевидно, существует такое  $m$ , при котором этот комбинированный итерационный процесс будет сходиться, но медленнее, вообще говоря, чем процесс, полностью синхронизированный. Поэтому наш подход имеет смысл, если асинхронные итерации существенно дешевле, чем синхронные.

Оценка возможной получаемой выгоды существенно зависит от вида матрицы  $A$  и в общем случае может быть достаточно грубой. По-видимому, наиболее эффективным здесь может быть численный эксперимент. Тем не менее мы докажем лемму, которая указывает границы роста ошибки в асинхронном случае при  $\lambda_1(|A|) > 1$ .

Пусть  $x(j), j = 0, 1, 2, \dots$  — последовательность асинхронных итераций, определенная набором  $(A, f, x(0), \mathcal{J}, \mathcal{S})$ , а  $\tilde{x}$  — решение системы (1). Тогда  $x(j)$  можно представить в виде  $x(j) = \tilde{x} + \Delta x(j)$ , где  $\Delta x(j)$  — последовательность асинхронных итераций, заданная пятеркой  $(A, 0, \Delta x(0), \mathcal{J}, \mathcal{S})$ .

**Лемма 1.** Если для матрицы  $A$  выполнено  $|\lambda_1(A)| < 1$  и  $\lambda_1(|A|) > 1$ , то

$$\|\Delta x(k)\| \leq \|A\|^k \|\Delta x(0)\| \quad (6)$$

для  $k = 0, 1, 2, \dots$

**ДОКАЗАТЕЛЬСТВО.** Проведем доказательство по индукции. Для  $k = 0$  имеем

$$\|\Delta x(0)\| = \|A\|^0 \|\Delta x(0)\|,$$

и, следовательно, база индукции доказана. Пусть теперь (6) выполнено для всех  $k < m$ . Покажем, что (6) выполняется при  $k = m$ . Согласно определению асинхронных итераций если  $j \in J_m$ , то

$$\Delta x_j(m) = \sum_{r=1}^n a_{jr} \Delta x_r(s_r(m)),$$

а если  $j \notin J_m$ , то

$$\Delta x_j(m) = \Delta x_j(m-1).$$

Пусть  $\Delta \hat{x}$  — вектор длины  $2n$ , такой что  $\Delta \hat{x}_j = \Delta x_j(m)$ ,  $\Delta \hat{x}_{n+j} = 0$  при  $j \in J_m$  и  $\Delta \hat{x}_j = 0$ ,  $\Delta \hat{x}_{n+j} = \Delta x_j(m-1)$  при  $j \notin J_m$ . Обозначим вектор  $(\Delta x_1(s_1(m)), \dots, \Delta x_n(s_n(m)))^T$  за  $\tilde{\Delta x}$ . Для  $\Delta \hat{x}$ ,  $\tilde{\Delta x}$  и  $\Delta x(m-1)$  справедливо равенство

$$\Delta \hat{x} = \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix} \begin{pmatrix} \tilde{\Delta x} \\ \Delta x(m-1) \end{pmatrix},$$

где  $A_1, A_2$  — матрицы  $n \times n$ . При  $j \in J_m$   $j$ -я строка матрицы  $A_1$  равняется  $j$ -й строке матрицы  $A$ , а при  $j \notin J_m$   $j$ -я строка матрицы  $A_1$  состоит из нулей.  $A_2$  — диагональная матрица, для которой  $j$ -й элемент диагонали равен 0, если  $j \in J_m$ , и равен 1 в противном случае.

Заметим, что  $\|\Delta\hat{x}\| = \|\Delta x(m)\|$ ,  $\|A_1\| \leq \|A\|$ , и в силу того, что  $\|A\| \geq |\lambda_1(|A|)| > 1$ , выполнено неравенство  $\|A_2\| \leq \|A\|$ . Тогда справедливо неравенство

$$\|\Delta x(m)\| = \left\| \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix} \begin{pmatrix} \widetilde{\Delta x} \\ \Delta x(m-1) \end{pmatrix} \right\| \leq \|A\| \left\| \begin{pmatrix} \widetilde{\Delta x} \\ \Delta x(m-1) \end{pmatrix} \right\|.$$

Пусть  $\|(\widetilde{\Delta x}^T, \Delta x(m-1)^T)\| = |\Delta x_{j'}(s')|$  для некоторого натурального  $s' < m$  и  $1 \leq j' \leq n$ . Так как  $s' < m$ , в силу предположения индукции будет выполнено

$$\left\| \begin{pmatrix} \widetilde{\Delta x} \\ \Delta x(m-1) \end{pmatrix} \right\| \leq \|\Delta x(s')\| \leq \|A\|^{s'} \|\Delta x(0)\| \leq \|A\|^{m-1} \|\Delta x(0)\|,$$

а следовательно

$$\|\Delta x(m)\| \leq \|A\|^m \|\Delta x(0)\|,$$

и лемма доказана.

Теперь легко доказать следующую теорему.

**Теорема 2.** *Если итерационный процесс состоит из последовательности групп  $m$  асинхронных итераций и затем  $l$  синхронных, то при достаточно большом  $l$  он сходится. При этом сходится не медленнее, чем  $\lambda_\varepsilon \left(\frac{\|A\|}{\lambda_\varepsilon}\right)^{\frac{m}{m+l}}$ , где  $\lambda_\varepsilon$  удовлетворяет неравенству  $|\lambda_1(A)| < \lambda_\varepsilon < 1$ .*

**ДОКАЗАТЕЛЬСТВО.** Из леммы 1 следует, что за  $m$  асинхронных итераций ошибка может возрасти не более чем в  $\|A\|^m$  раз. Поскольку  $|\lambda_1(A)| < 1$ , для  $\forall \varepsilon > 0$ , такого, что  $\varepsilon < 1 - |\lambda_1(A)|$ , найдется такое  $l_0$ , что для  $\forall l \geq l_0$  будет выполняться неравенство  $\|A^l\| < (|\lambda_1(A)| + \varepsilon)^l < 1$ . Обозначим  $(|\lambda_1(A)| + \varepsilon)$  за  $\lambda_\varepsilon$ . Нетрудно видеть, что  $|\lambda_1(A)| < \lambda_\varepsilon < 1$ . Норму ошибки после  $m + l$  итераций ( $m$  асинхронных и  $l$  синхронных) можно оценить следующим образом:

$$\|\Delta x(m+l)\| \leq \|A^l\| \|A\|^m \|\Delta x(0)\| < \lambda_\varepsilon^l \|A\|^m \|\Delta x(0)\|.$$

При достаточно большом  $l$  имеем  $\lambda_\varepsilon^l \|A\|^m < 1$ , что и доказывает первую часть теоремы.

Мы видим, что за  $m + l$  итераций ошибка уменьшается в  $\lambda_\varepsilon^l \|A\|^m$  раз. Такой результат мы имели бы при геометрической сходимости с параметром  $r$ , если бы  $r^{m+l} = \lambda_\varepsilon^l \|A\|^m$ . То есть

$$r = (\lambda_\varepsilon^l \|A\|^m)^{\frac{1}{m+l}} = \lambda_\varepsilon \left(\frac{\|A\|}{\lambda_\varepsilon}\right)^{\frac{m}{m+l}},$$

что доказывает вторую часть теоремы.

**2. Алгоритмы метода Монте-Карло.** Известны различные алгоритмы метода Монте-Карло для решения системы вида (1). Мы будем рассматривать простейшие алгоритмы, состоящие в оценивании сходящегося ряда Неймана

$$(h, \tilde{x}) = \left( h, \sum_{k=0}^{\infty} A^k f \right)$$

с помощью выбранной соответствующим образом оценки — оценки на траекториях моделируемой цепи Маркова. Здесь  $h$  — заданный вектор, а  $\tilde{x}$  — решение системы (1). Как известно [3], при

$$\lambda_1(|A|) < 1 \tag{7}$$

возможно вычисление  $(h, \tilde{x})$  при помощи асинхронного алгоритма. То есть на различных процессорах моделируются траектории цепи Маркова и вычисляются оценки на них, после чего оценки, полученные на всех процессорах, усредняются. Таким образом, условия несмещенности оценок метода Монте-Карло и сходимости асинхронных итераций совпадают, на что было обращено внимание в работе [4].

Как было показано в [3], нарушение условия (7) и использование асинхронного алгоритма приводит к стохастической неустойчивости — экспоненциальному росту дисперсии. Эта трудность, вообще говоря, может быть преодолена за счёт увеличения вычислительной работы, но её экспоненциальный рост делает алгоритм нереализуемым. Альтернативой является запоминание промежуточных результатов (синхронизация).

Предлагается в случае  $\lambda_1(|A|) > 1$  («дешевых» асинхронных алгоритмов и относительно «дорогих» синхронных), как и в случае обычных асинхронных итераций, использовать смешанный алгоритм с частичной синхронизацией. Как и ранее, будем рассматривать подход, в котором чередуются асинхронные и синхронные алгоритмы. Для формального исследования такого комбинированного алгоритма мы подробно рассмотрим схемы оценивания частичных сумм ряда Неймана в асинхронном и синхронном случаях.

**2.1. Оценки частичных сумм ряда Неймана.** Рассмотрим аналог схемы Неймана—Улама для нахождения частичной суммы ряда Неймана

$$S_m = \sum_{k=0}^m A^k f,$$

где  $m \in \mathbb{N}$  задано. Случай  $m = \infty$  подробно изучен [5]. Случай конечного  $m$  требует некоторой модификации стандартных рассуждений.

Пусть  $h$  — заданный вектор,  $h = (h_1, \dots, h_m)^T$ . Алгоритм предполагает задание стохастической матрицы  $\mathcal{P} = \|p_{ij}\|_{i,j=1}^n$ , удовлетворяющей условиям согласования:

- (а)  $p_{i,j} > 0$ , если  $a_{i,j} \neq 0$ , и распределение  $q = (q_1, \dots, q_n)$  такое, что  $q_i > 0$ , если  $h_i \neq 0$ ,

или

- (б)  $p_{i,j} > 0$ , если  $a_{j,i} \neq 0$ , и распределение  $p = (p_1, \dots, p_n)$  такое, что  $p_i > 0$ , если  $f_i \neq 0$ .

Также зададим вектор  $g = (g_1, \dots, g_m)^T$ , такой что  $g_i > 0$ ,  $i = 1, \dots, m$ , и

$$\sum_{i=1}^m g_i = 1.$$

Введем случайную величину  $\tau$ , распределенную по закону

$$\begin{pmatrix} 1 & \dots & m \\ g_1 & \dots & g_m \end{pmatrix},$$

и рассмотрим случайные величины

$$\zeta = \frac{h_{i_0} a_{i_0, i_1} \dots a_{i_{\tau-1}, i_{\tau}} f_{i_{\tau}}}{q_{i_0} p_{i_0, i_1} \dots p_{i_{\tau-1}, i_{\tau}} g_{i_{\tau}}}, \quad (8)$$

$$\zeta^* = \frac{f_{j_0} a_{j_1, j_0} \cdots a_{j_\tau, j_{\tau-1}} h_{j_\tau}}{p_{j_0} p_{j_0, j_1} \cdots p_{j_{\tau-1}, j_\tau} g_\tau}, \quad (9)$$

где  $i_0 \rightarrow i_1 \rightarrow \cdots \rightarrow i_\tau$  соответствует цепи Маркова  $(q, \mathcal{P})$ , а  $j_0 \rightarrow j_1 \rightarrow \cdots \rightarrow j_\tau$  — цепи Маркова  $(p, \mathcal{P})$ .

Легко проверить, что

$$\mathbb{E}\zeta = \left( h, \sum_{k=1}^m A^k f \right) = (h, S^m),$$

$$\mathbb{E}\zeta^* = \left( \sum_{k=1}^m f^T (A^T)^k, h \right) = (S_m^T, h).$$

Столь же просто вычисляются дисперсии оценок (8), (9), если учесть, что

$$\mathbb{E}\zeta^2 = \sum_{k=1}^m \sum_{i_0, i_1, \dots, i_k=1}^n \frac{h_{i_0}^2 a_{i_0, i_1}^2 \cdots a_{i_{k-1}, i_k}^2 f_{i_k}^2}{q_{i_0} p_{i_0, i_1} \cdots p_{i_{k-1}, i_k} g_k} = \left( \frac{h^2}{q}, \sum_{k=1}^m \left( \frac{A^2}{\mathcal{P}} \right)^k \frac{f^2}{g} \right), \quad (10)$$

$$\mathbb{E}(\zeta^*)^2 = \sum_{k=1}^m \sum_{j_0, j_1, \dots, j_k=1}^n \frac{f_{j_0}^2 a_{j_1, j_0}^2 \cdots a_{j_k, j_{k-1}}^2 h_{j_k}^2}{p_{j_0} p_{j_0, j_1} \cdots p_{j_{k-1}, j_k} g_k} = \left( \sum_{k=1}^m \frac{(f^T)^2}{p} \left( \frac{(A^T)^2}{\mathcal{P}} \right)^k, \frac{h^2}{g} \right), \quad (11)$$

где операции возведение в квадрат векторов и матриц, как и деление матрицы на матрицу и вектора на вектор, выполняются поэлементно. В дальнейшем будет отдельно оговариваться использование подобных операций над матрицами и векторами. Оценкой всех компонент  $S_m$  может быть вектор  $z = (\zeta_1, \dots, \zeta_n)^T$ , где  $\zeta_i$ ,  $i = 1, \dots, n$ , получаются из (8), если полагать  $h$  равным вектору с  $i$ -й компонентой, равной единице, и остальными, равными нулю.

Как видно из (10) и (11), поведение дисперсии с ростом  $m$  определяется первыми собственными числами матриц  $A^2/\mathcal{P}$  и  $(A^T)^2/\mathcal{P}$  соответственно. Так, если выполнено  $\lambda_1(A^2/\mathcal{P}) < 1$  и  $\lambda_1((A^T)^2/\mathcal{P}) < 1$ , то соответствующие дисперсии будут ограничены при стремлении  $m$  к бесконечности. В противном случае будет наблюдаться экспоненциальный рост дисперсии.

Теперь для заданного вектора  $y = (y_1, \dots, y_n)^T$  введем оценки  $\eta = \eta(y)$  и  $\eta^* = \eta^*(y)$  такие, что

$$\mathbb{E}\eta = (h, A^{m+1}y),$$

$$\mathbb{E}\eta^* = (y^T (A^T)^{m+1}, h).$$

Оценки  $\eta$  и  $\eta^*$  строятся на траекториях цепей Маркова  $(p, \mathcal{P})$  и  $(q, \mathcal{P})$  длины  $m+1$  и выражения для них имеют вид

$$\eta = \frac{h_{i_0} a_{i_0, i_1} \cdots a_{i_m, i_{m+1}} y_{i_{m+1}}}{q_{i_0} p_{i_0, i_1} \cdots p_{i_m, i_{m+1}}}, \quad (12)$$

$$\eta^* = \frac{y_{j_0} a_{j_1, j_0} \cdots a_{j_{m+1}, j_m} h_{j_{m+1}}}{p_{j_0} p_{j_0, j_1} \cdots p_{j_m, j_{m+1}}}. \quad (13)$$

Как и ранее, дисперсии оценок (12), (13) вычисляются просто, с учетом того, что выражения для вторых моментов имеют вид

$$\mathbb{E}\eta^2 = \left( \frac{h^2}{q}, \left( \frac{A^2}{\mathcal{P}} \right)^{m+1} y^2 \right),$$

$$\mathbb{E}(\eta^*)^2 = \left( \frac{(y^T)^2}{p} \left( \frac{(A^T)^2}{\mathcal{P}} \right)^{m+1}, h^2 \right),$$

где операции возведение в квадрат векторов и матриц, как и деление матрицы на матрицу и вектора на вектор, выполняются поэлементно. Поведение дисперсии, также как и для оценок (8), (9), с ростом  $m$  определяется первыми собственными числами матриц  $A^2/\mathcal{P}$  и  $(A^T)^2/\mathcal{P}$ .

Таким образом, при помощи пар оценок  $\eta$ ,  $\zeta$  и  $\eta^*$ ,  $\zeta^*$  для заданного вектора  $y$  и натурального  $m$  может быть получена оценка вектора  $A^{m+1}y + \sum_{k=0}^m A^k f$ . Поручая моделирование цепей Маркова и вычисление оценок по ним разным процессорам, мы вновь получим асинхронный алгоритм.

**2.2. Оценки суммы ряда Неймана.** Оценки (8), (12) и (9), (13) могут служить основой алгоритмов с частичной синхронизацией для получения решения системы (1) без ограничений вида (7). Алгоритмы состоят в последовательном вычислении оценок векторов  $y(j)$ , где  $y(j)$  связаны соотношением

$$y(0) = f, \quad y(j) = A^m y(j-1) + \sum_{k=0}^{m-1} A^k f, \quad (14)$$

$$y(j+1) = Ay(j) + f, \dots, y(j+l) = Ay(j+l-1) + f, \quad j = 1, l+2, 2l+2, \dots \quad (15)$$

или соотношением

$$y(0) = f, \quad y(j) = A^{m+1}y(j-1) + \sum_{k=0}^m A^k f, \quad j = 1, 2, \dots \quad (16)$$

для некоторого фиксированного натурального  $m$ . Первый случай соответствует частичной синхронизации, когда аналогично описанному выше подходу для асинхронных итераций после асинхронного алгоритма следуют синхронные. Если  $j$  устремить к бесконечности, то  $y(j)$  в (14)–(16) будет стремиться к искомой сумме  $\sum_{k=0}^{\infty} A^k f$ .

При использовании метода Монте-Карло оценками векторов  $y(j)$  будут случайные векторы  $\Xi(j) = (\xi_1(j), \dots, \xi_n(j))^T$ . Скалярное произведение  $(h, y(j))$  может оцениваться с помощью оценок вида (8), (12) — в этом случае оценкой для  $(h, y(j))$  будет

$$\eta(\Xi(j-1)) + \zeta + (h, f),$$

или оценок вида (9), (13) — в этом случае  $(h, y(j))$  будет оцениваться как

$$\eta^*(\Xi(j-1)) + \zeta^* + (h, f).$$

В каждом случае берется среднее из некоторого количества  $N$  реализаций оценок. Оценка всех компонент  $y(j)$  может быть получена, если полагать  $h$  равным векторам с одной из компонент, равной 1, и остальными, равными 0.

При этом во многих случаях интерес представляет оценка  $y(j)$  для  $T \leq j < +\infty$  (разностные схемы).

Заметим, что один и тот же вектор ( $S_m$  в (16) и  $S_{m-1}$  в (14)–(15)) оценивается на каждой итерации, поэтому можно организовать алгоритм таким образом, чтобы оценка этого вектора уточнялась с каждой новой итерацией, используя оценки с предыдущих шагов итераций.

Особый интерес представляет задача оценки ковариации случайных векторов, возникающих при использовании метода Монте-Карло для оценки последовательности векторов итерационного процесса. Ковариация вектора  $\Xi(j)$  с ростом  $j$  может оставаться ограниченной, а может экспоненциально расти. В последнем случае алгоритм будет стохастически неустойчивым, что делает невозможным его применение на практике.

Рассмотрим теперь итерационный процесс  $y(j)$  с начальным вектором  $y(0)$  и

$$y(j) = By(j-1) + d, \quad j = 1, 2, \dots, \quad (17)$$

где  $B$  – матрица  $n \times n$ , а  $d$  – вектор длины  $n$ . Пусть при каждом  $j$  вычисление слагаемых в правой части (17) происходит с помощью рандомизированной процедуры. Таким образом, вместо последовательности  $y(j)$  возникает последовательность случайных векторов  $\Xi(j) = (\xi_1(j), \dots, \xi_n(j))^T$ ,  $j = 0, 1, 2, \dots$ , с начальным вектором  $\Xi(0)$ , связанных соотношением

$$\Xi(j) = \mathbb{B}_j \Xi(j-1) + \mathbb{D}(j), \quad (18)$$

где  $\mathbb{B}_j = \|\beta_{i,k}\|_{i,k=1}^n$  – случайные матричные операторы,  $\mathbb{D}(j)$  – случайные векторы. При этом для любого натурального  $j$  выполнено  $\mathbb{E}\Xi(j) = y(j) = (y_1(j), \dots, y_n(j))^T$ ,  $\mathbb{E}\mathbb{B}_j = B$ ,  $\mathbb{E}\mathbb{D}(j) = d$ .

Следующая лемма определяет характер поведения ковариации векторов ошибок  $\mathcal{E}(j) = \Xi(j) - y(j)$ .

**Лемма 2.** Пусть случайные операторы  $\mathbb{B}_j$ , векторы  $\mathbb{D}(j)$  и  $\Xi(k)$  независимы в совокупности при любых  $j = 0, 1, 2, \dots$  и  $k < j$  в том смысле, что случайные величины  $\alpha_1, \alpha_2, \alpha_3$ , где  $\alpha_1$  – произвольный элемент оператора  $\mathbb{B}_j$ ,  $\alpha_2$  – произвольный элемент  $\mathbb{D}(j)$ ,  $\alpha_3$  – произвольный элемент  $\Xi(k)$ , независимы в совокупности. Тогда для матрицы ковариации вектора  $\mathcal{E}(j)$  справедливо соотношение

$$\begin{aligned} \text{vec cov} \mathcal{E}(j) = & B \otimes B^T \text{vec cov} \mathcal{E}(j-1) + \mathbb{E}(\Delta_j \otimes \Delta_j^T) \text{vec cov} \mathcal{E}(j-1) + \\ & + \mathbb{E}(\Delta_j \otimes \Delta_j^T) \text{vec}(y(j-1)y(j-1)^T) + \text{vec cov} \delta(j), \end{aligned} \quad (19)$$

где  $\Delta_j = \mathbb{B}_j - B$ ,  $\delta(j) = \mathbb{D}(j) - d$ ,  $\text{vec}$  – операция векторизации матрицы, а  $\otimes$  – операция кронекеровского произведения матриц.

ДОКАЗАТЕЛЬСТВО. Подставим в (18) выражения для  $\mathbb{B}_j$ ,  $\mathbb{D}(j)$  и  $\Xi(j)$ :

$$y(j) + \mathcal{E}(j) = (B + \Delta_j)(y(j-1) + \mathcal{E}(j-1)) + d + \delta(j).$$

Поскольку  $y(j)$  подчинен (17), получим выражения для  $\mathcal{E}(j)$  и  $\mathcal{E}(j)^T$ :

$$\mathcal{E}(j) = B\mathcal{E}(j-1) + \Delta_j y(j-1) + \Delta_j \mathcal{E}(j-1) + \delta(j) \quad (20)$$

и

$$\mathcal{E}(j)^T = \mathcal{E}(j-1)^T B^T + y(j-1)^T \Delta_j^T + \mathcal{E}(j-1)^T \Delta_j^T + \delta(j)^T. \quad (21)$$

Далее необходимо перемножить правые и левые части равенств (20), (21) и вычислить математическое ожидание от всех членов получившегося равенства. При этом стоит отметить, что математические ожидания многих слагаемых правой части будут равны нулю. Так,  $\mathbb{E}\Delta_j \mathcal{E}(j-1)y(j-1)^T \Delta_j^T = 0$  в силу того, что  $\mathcal{E}(j-1)$  не зависит от  $\Delta_j$  и  $\mathbb{E}\mathcal{E}(j) = 0$ . Учитывая эти соображения, имеем

$$\text{cov} \mathcal{E}(j) = B \text{cov} \mathcal{E}(j-1) B^T + \mathbb{E}(\Delta_j \text{cov} \mathcal{E}(j-1) \Delta_j^T) + \mathbb{E}(\Delta_j y(j-1)y(j-1)^T \Delta_j^T) + \text{cov} \delta(j).$$



Теперь, применяя операцию векторизации к матрицам  $\text{cov}\mathcal{E}(j)$ ,  $\text{cov}\mathcal{E}(j-1)$ ,  $y(j-1)y(j-1)^T$  и  $\text{cov}\delta(j)$ , получим (19), что и доказывает лемму.

Из леммы 2 можно вывести

**Следствие 1.** *Для стохастической устойчивости алгоритма (18) необходимо и достаточно, чтобы модуль первого собственного числа матрицы  $B \otimes B^T + \mathbb{E}(\Delta_j \otimes \Delta_j^T)$  был меньше единицы.*

Известно [6], что собственными числами  $B \otimes B^T$  являются  $\lambda_i(B)\lambda_k(B)$ ,  $i, k = 1, \dots, n$ . Заметим также, что  $\|B \otimes B^T\| = \|B\|\|B^T\|$ .

Для удобства введем следующие обозначения:  $\mathcal{M}_j = \mathbb{E}(\Delta_j \otimes \Delta_j^T)$ ,  $\mathcal{B}_j = B \otimes B^T + \mathcal{M}_j$  и  $\mathcal{D}_j = \mathbb{E}(\Delta_j \otimes \Delta_j^T)\text{vec}(y(j-1)y(j-1)^T) + \text{vec}\text{cov}\delta(j)$ . Тогда выражение для ковариации из леммы 2 примет вид

$$\text{vec}\text{cov}\mathcal{E}(j) = (B \otimes B^T + \mathcal{M}_j)\text{vec}\text{cov}\mathcal{E}(j-1) + \mathcal{D}_j.$$

Элементы матрицы  $\mathcal{M}_j$  имеют порядок  $1/N_j$ , где  $N_j$  — количество моделируемых траекторий на  $j$ -й итерации, то есть  $\mathcal{M}_j = \frac{1}{N_j}\mathcal{M}'$ , где  $\mathcal{M}'$  — матрица с элементами, не зависящими от  $N_j$ .

Лемма 2 и следствие из нее позволяют сделать определенные выводы относительно стохастической устойчивости рандомизированных процедур для итерационных процессов (14)–(15) и (16).

**Теорема 3.** *Если  $|\lambda_1(A)| < 1$  и на каждой итерации (14)–(15) вычисления осуществляются при помощи метода Монте-Карло, то для любого натурального  $t$  существует такое  $N'$ , что для  $\forall N_j > N'$ ,  $j = 1, 2, \dots$ , где  $N_j$  — количество моделируемых траекторий цепи Маркова для оценки  $y(j)$  в (14)–(15), рандомизированный итерационный процесс (14)–(15) будет стохастически устойчивым.*

**ДОКАЗАТЕЛЬСТВО.** Рассмотрим (14) и заметим, что матрица  $B$  из леммы 2 равна  $A^m$ , при этом в явном виде матрица  $A^m$  не известна и оценивается при помощи метода Монте-Карло. Учитывая, что  $\|\mathcal{B}_j\| \leq \|B\|\|B^T\| + \frac{1}{N_j}\|\mathcal{M}'\|$ , норму ковариации ошибки после вычисления (14) методом Монте-Карло можно оценить как

$$\|\text{vec}\text{cov}\mathcal{E}(j)\| \leq \left( \|A^m\| \|(A^m)^T\| + \frac{1}{N_j}\|\mathcal{M}'\| \right) \|\text{vec}\text{cov}\mathcal{E}(j-1)\| + \|\mathcal{D}_j\|.$$

Согласно (15) последующие  $l$  итераций ковариация ошибки будет изменяться в соответствии с

$$\text{vec}\text{cov}\mathcal{E}(j+1) = \left( A \otimes A^T + \frac{1}{N_{j+1}}\mathcal{M}' \right) \text{vec}\text{cov}\mathcal{E}(j) + \mathcal{D}_{j+1}.$$

Так как  $|\lambda_1(A)| < 1$ , верно  $|\lambda_1(A \otimes A^T)| = \lambda_1^2(A) < 1$  и существует такое  $N'$ , что для  $\forall N_{j+1} > N'$  первое собственное число матрицы  $A \otimes A^T + \frac{1}{N_{j+1}}\mathcal{M}'$  по модулю будет меньше единицы. Тогда, согласно следствию 1, рандомизированный итерационный процесс будет стохастически устойчивым.

Оптимальный выбор параметров  $m$ ,  $l$ ,  $N'$  из теоремы 3 зависит от матрицы  $A$ , вида оценок и свойств многопроцессорной системы. Как видно, при достаточно большом  $N'$  предложенная процедура соответствует случаю частичной синхронизации в детерминированном случае, за исключением того, что на каждом шаге итерации возникает случайная ошибка. Помимо очевидного увеличения числа моделируемых

траекторий  $N'$  дисперсия случайной ошибки может быть уменьшена путем применения различных техник. Важно заметить, что если количество процессоров больше порядка системы, то часть процессоров при использовании асинхронных итераций будет простаивать, так как для расчета одной компоненты очередной итерации используется не более одного процессора. Использование метода Монте-Карло в тех же условиях позволяет эффективно использовать все имеющиеся процессоры, равномерно распределяя по ним моделируемые траектории цепей Маркова.

Похожую теорему можно сформулировать для (16).

**Теорема 4.** *Если  $|\lambda_1(A)| < 1$  и на каждой итерации (16) вычисления осуществляются при помощи метода Монте-Карло, то для любого натурального  $m$  существует такое  $N'$ , что для  $\forall N_{j+1} > N'$ , где  $N_j$  — количество моделируемых траекторий цепи Маркова для оценки  $y(j)$  в (16), рандомизированный итерационный процесс (16) будет стохастически устойчивым.*

**ДОКАЗАТЕЛЬСТВО.** При применении метода Монте-Карло для вычисления очередного  $y(j)$  в выражении (16) ковариация ошибки, согласно лемме 2, будет изменяться следующим образом:

$$\text{vec cov}\mathcal{E}(j) = \left( A^m \otimes (A^m)^T + \frac{1}{N_j} \mathcal{M}' \right) \text{vec cov}\mathcal{E}(j-1) + \mathcal{D}_j.$$

Сделанное предположение о матрице  $A$ , а именно  $|\lambda_1(A)| < 1$ , означает, что выполнено неравенство  $|\lambda_1(A^m)| = |\lambda_1((A^m)^T)| = |\lambda_1(A)|^m < 1$ . Модуль первого собственного числа матрицы  $A^m \otimes (A^m)^T$  равен  $|\lambda_1(A)|^{2m}$ . Этот факт позволяет заключить, что для любого натурального  $m$  существует такое  $N'$ , что для  $\forall N_{j+1} > N'$  модуль первого собственного числа матрицы  $A^m \otimes (A^m)^T + \frac{1}{N_{j+1}} \mathcal{M}'$  будет меньше единицы. А это в свою очередь согласно следствию 1 означает, что процедура вычисления (16) методом Монте-Карло будет стохастически устойчивой.

При использовании (16) и метода Монте-Карло может быть построен стохастически устойчивый алгоритм. Он будет обладать большей асинхронностью, чем алгоритм, построенный на основе (14)–(15), но у такого алгоритма при равном числе моделируемых траекторий ковариация оценок будет больше.

**3. Численные эксперименты.** Для асинхронных итераций численные эксперименты проводились для случайным образом смоделированной матрицы  $A$  размерности  $4 \times 4$ , такой что  $\lambda_1(A) = 0.76$ ,  $\lambda_1(|A|) = 1.64$ ,  $\|A\| = 2.04$ .

На рисунке 1 изображена норма вектора ошибок для асинхронных итераций с множеством  $\mathcal{J} = \{J_j\}_{j=1}^{\infty}$  таким, что  $k \in J_j$  для  $k = 1, 2, 3, 4$ , если  $j \bmod k = 0$ , а множество  $\mathcal{S}$  такое, что  $s_i(j) = j - 1$ ,  $i = 1, \dots, n$ ,  $j = 1, 2, \dots$ . В данном случае наблюдается экспоненциальный рост ошибки.

На рисунке 2 изображена норма вектора ошибок для асинхронных итераций с частичной синхронизацией. Теорема 2 гарантирует сходимость нормы ошибки к нулю при  $m = 2$ ,  $l = 6$ , однако сходимость будет наблюдаться уже при значениях  $m = 2$ ,  $l = 3$ .

Для метода Монте-Карло численные эксперименты проводились для случайным образом смоделированной матрицы  $A$  размерности  $10 \times 10$ , такой что  $\lambda_1(A) = 0.79$ ,  $\lambda_1(|A|) = 3.21$ ,  $\|A\| = 8.45$ .

На рисунке 3 сплошной линией изображена норма вектора ошибок при решении (16) методом Монте-Карло с использованием оценок (8) и (12) для параметров

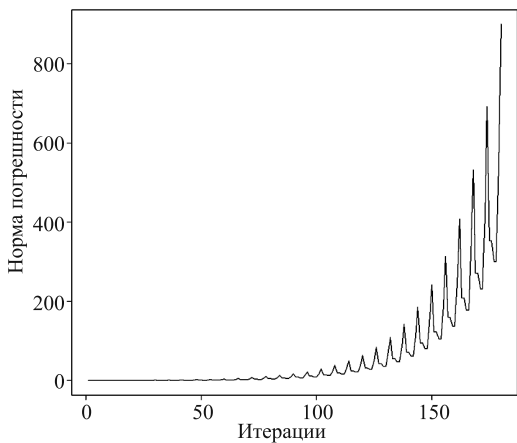


Рис. 1.

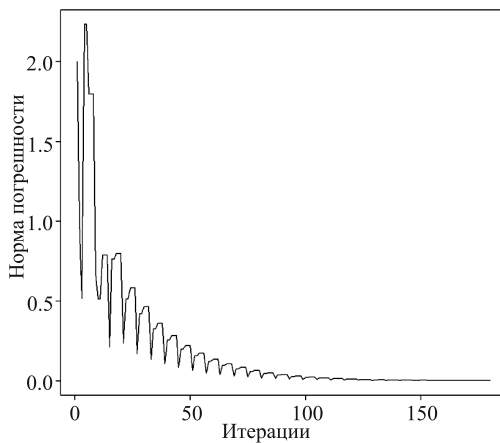


Рис. 2.

$m = 2, N_j = 50000, j = 1, \dots, 30$ . Пунктирной линией изображены доверительные интервалы.

На рисунке 4 сплошной линией изображена норма вектора ошибок при решении (14)–(15) методом Монте-Карло использованием оценок (8) и (12) для параметров  $m = 2, l = 4$ . При этом  $N_j = 50000$  при расчете (14) и  $N_j = 5000$  при расчете каждой итерации в (15). Пунктирной линией изображены доверительные интервалы.

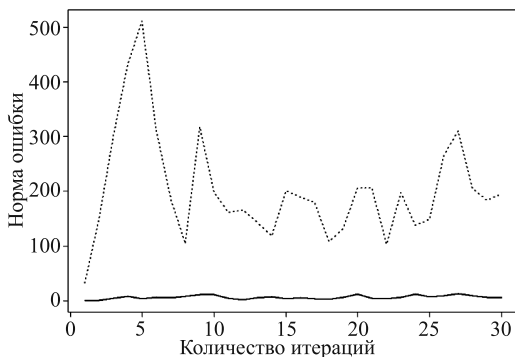


Рис. 3.

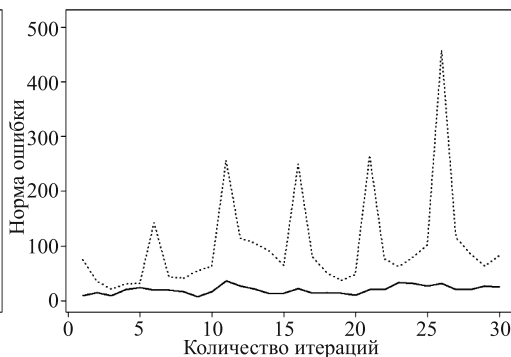


Рис. 4.

**Заключение.** Одновременное выполнение условий  $|\lambda_1(A)| < 1$  и  $\lambda_1(|A|) > 1$  может создать определенные трудности при использовании как детерминированных, так и стохастических асинхронных методов для решения системы (1).

В детерминированном случае возможным выходом является использование частичной синхронизации, когда асинхронные итерации чередуются с синхронными. Представление о соотношении количества синхронных и асинхронных итерации даёт теорема 2. Стоит отметить, что такой подход, вообще говоря, вписывается в концепцию асинхронных итераций.

Комбинируя алгоритмы асинхронного типа и алгоритмы с синхронизацией метода Монте-Карло, мы, как и в детерминированном случае, можем расширить сферу асинхронности при использовании стохастических методов. Случайный характер

ошибки создаёт свои особенности, которые, как мы видели, легко учесть. Богатый арсенал средств понижения дисперсии оценок создаёт много возможностей для совершенствования алгоритмов и, по мнению авторов, для больших систем уравнений метод Монте-Карло может быть предпочтительнее детерминированных асинхронных итераций.

## Литература

1. Chazan D., Miranker W. Chaotic relaxation // Linear Algebra and its Applications. 1969. Vol. 2, N 2. P. 199–222.
2. Baudet G. M. Asynchronous Iterative Methods for Multiprocessors // J. ACM. 1978. Vol. 25, N 2. P. 226–244.
3. Ермаков С. М. Метод Монте-Карло в вычислительной математике (Вводный курс). Невский диалект, Бином. Лаборатория знаний, 2009. С. 192.
4. Ермаков С. М. Метод Монте-Карло и асинхронные вычисления // Тезисы 1-й международной конференции общества Бернулли. Том 1. 1987. С. 462.
5. Ермаков С. М. Метод Монте-Карло и смежные вопросы. М.: Наука, 1975. С. 472.
6. Ланкастер П. Теория матриц. Наука, 1973. С. 282.

Статья поступила в редакцию 26 июня 2014 г.

## Сведения об авторах

Дмитриев Алексей Валерьевич — аспирант; alx.dmitriev@gmail.com

Ермаков Сергей Михайлович — доктор физико-математических наук, профессор; sergej.ermakov@gmail.com

## MONTE CARLO METHOD AND ASYNCHRONOUS ITERATIONS

Alexey V. Dmytriev, Sergey M. Ermakov

St.Petersburg State University, Universitetskaya nab., 7-9, St.Petersburg, 199034, Russian Federation; nkk@math.spbu.ru, nevolga@gmail.com

Asynchronous methods as well as Monte Carlo methods play an important role in using multiprocessor (multicore) computations and investigation of the connections between them and, especially, expansion the field of their applicability are important problems. System of the form  $x = Ax + f$  is considered, where  $x$  is unknown column-vector of length  $n$ ,  $F$  is a column-vector of the right-hand sides of length  $n$ ,  $A$  is  $n \times n$ -matrix of the system. Case when the first eigenvalue of  $|A|$  less than 1 is well studied and in that case there are close connections between asynchronous iterations and Monte Carlo method. If the first eigenvalue of  $|A|$  is greater than 1, whereas the first eigenvalue of  $A$  less than 1, then similar methods appear to be inapplicable. New modifications of Monte Carlo method and asynchronous method with partial synchronization are proposed. Estimates of the errors and convergence conditions for proposed methods are obtained depending on the ratio are obtained. Numerical experiments are performed. Refs 6. Figs 4.

*Keywords:* Monte Carlo methods, asynchronous methods, asynchronous iterations, parallel algorithms, statistical modeling.