Анализ точности численного решения стохастических дифференциальных уравнений на суперкомпьютерах*

С.С. Артемьев, В.Д. Корнеев

Институт вычислительной математики и математической геофизики
Сибирского отделения Российской академии наук,
Новосибирск, Россия

В работе проведены исследования точности оценок функционалов от решений СДУ на суперкомпьютерах. Показано, что для точной оценки математического ожидания и дисперсии решений линейных СДУ с мультипликативным шумом первого и второго порядка, требуется моделирование ансамблей траекторий размера $10^9 \div 10^{12}$, что невозможно осуществить на ПК из-за высокой трудоемкости. Проведены исследования поведения математического ожидания и дисперсии решения нелинейного стохастического уравнения Ван-дер-Поля. Описываются способы распараллеливания статистических алгоритмов на многопроцессорном кластере. Приводятся результаты численных экспериментов, полученных на суперкомпьютере Сибирского суперкомпьютерного центра.

1. Введение

Использование методов Монте-Карло (ММК) для нахождения решений краевых задач математической физики на основе вероятностных представлений часто требует численного решения сопутствующих СДУ и вычисления интегралов вдоль моделируемых траекторий. В настоящей работе исследуется зависимость точности статистических алгоритмов решений простейших СДУ от размера ансамбля моделируемых траекторий и от размера шага интегрирования обобщённого метода Эйлера. Приводятся результаты численных экспериментов по оценке моментов решений СДУ с растущей дисперсией и оценке среднего времени первого выхода траекторий СДУ с мультипликативным шумом на границу заданной области.

Точность оценок функционалов от решений СДУ зависит не только от размеров ансамблей моделируемых траекторий, но и от размера шага интегрирования используемого численного метода решения СДУ. Особую трудность представляет статистическое моделирование стохастических осцилляторов. Математические модели в виде СДУ с осциллирующими решениями возникают в самых разных областях науки [1–3]. Особый интерес представляет анализ возможных переходов от одного типа осцилляций к другому, например, при прогнозировании аварий и катастроф, вызываемых ростом амплитуды колебаний. В связи с этим возникает задача оценки устойчивости заданного режима работы. Как показали ранее проведённые эксперименты [4], при численном решении осциллирующих СДУ зачастую возникает неустойчивость численного решения, т. е. сильный рост дисперсии осцилляций. В связи с этим возникает необходимость уменьшения размера шага интегрирования на несколько порядков. Кроме того, малые объёмы моделируемых траекторий дают совершенно неверные оценки моментов решений в случае сильной асимметрии их плотностей распределения.

Оценки математического ожидания функционалов от решений СДУ, требующие вычисления интегралов вдоль моделируемых траекторий, сверхмалые размеры шага интегрирования обобщённого метода Эйлера и огромные объёмы моделируемых траекторий ведут к необходимости использования суперкомпьютеров с большим количеством процессоров, чтобы получить удовлетворительную точность численного анализа за приемлемое время

*Работа выполнена при финансовой поддержке гранта РФФИ (проект № 11-01-00252).

счета. Первые исследования авторов по численному решению СДУ с растущей дисперсией на суперкомпьютерах были описаны в работе [5].

В настоящей работе помимо численных экспериментов приводятся описания параллельных программ и оценок зависимости времени счета от числа используемых процессоров. Численные эксперименты проводятся на кластере НКС-30Т Сибирского суперкомпьютерного центра при Институте вычислительной математики и математической геофизики СО РАН.

2. Решаемые задачи

Для численного решения задачи Коши для общих СДУ в смысле Ито (японский математик)

$$dy = f(y)\,dt + \sigma(y)\,dw(t), \qquad y(0) = y_0, \quad 0 \le t \le T \tag{1}$$

обычно используется наименее трудоёмкий обобщенный метод Эйлера:

$$y_{n+1} = y_n + h\,f(y_n) + \sigma(y_n)\sqrt{h}\;\xi_{n+1}. \tag{2}$$

Здесь y_{n+1} - численное решение на сетке $t_{n+1} = t_n + h$, $\{\xi_{n+1}\}_0^{N-1}$ - последовательность независимых между собой и с y_n стандартных гауссовских случайных величин. Для моделирования ξ_n используется стандартная формула $\xi = \sqrt{-2\ln\alpha_1}\,\sin 2\pi\alpha_2$, где α_1, α_2 - равномерно распределенные случайные величины в (0,1) [6].

1) Решение СДУ с мультипликативным шумом

ММК часто рекомендуются для нахождения решений краевых задач для эллиптических уравнений с использованием вероятностного представления [7]. Например, для одномерной задачи Дирихле

$$\frac{1}{2}\,\sigma^2(y)\frac{d^2u}{dy^2} + f(y)\frac{du}{dy} + 1 = 0 \tag{3}$$

в интервале $[a, b]$ с граничными условиями $u(a) = u(b) = 0$ решение $u(y_0)$ может быть представлено в вероятностном виде

$$u(y_0) = E\tau(y_0), \tag{4}$$

где τ - время первого выхода решения СДУ (1) из $[a, b]$.

2) Стохастические осцилляторы

а) Мультипликативные шумы зачастую связаны с "шумящими" коэффициентами в ОДУ. Рассматривается случай "шумящих" коэффициентов в линейном колебательном контуре, который задается СДУ второго порядка вида

$$\frac{d^2y}{dt^2} + (\lambda + \sigma_1\,\frac{dw_1}{dt})\,\frac{dy}{dt} + (\beta^2 + \sigma_2\,\frac{dw_2}{dt})\,y = 0 \tag{5}$$

с постоянными λ, β, σ_1, σ_2. В частном случае СДУ (5) вида

$$\frac{d^2y}{dt^2} + (\beta^2 + \frac{dw}{dt})\,y = 0, \qquad y(0) \in N(1,1), \qquad \frac{dy}{dt}(0) = 0, \tag{6}$$

когда "шумит" только частота колебаний решения и отсутствует декремент затухания, математическое ожидание точного решения задаётся простой формулой $m(t) = cos(\beta t)$.

б) Стохастическое нелинейное уравнение Ван-дер-Поля с одним "шумящим" коэффициентом, записанное в виде системы СДУ в смысле Ито вида

$$\begin{aligned} dy_1 &= y_2\,dt, \\ dy_2 &= (\lambda\,y_2\,(1 - d\,y_1^2) - \beta^2 y_1)\,dt + \sigma y_1\,d\omega(t), \end{aligned} \tag{7}$$

описывает колебания нелинейного контура (см., например, [1]). В (7) постоянные λ, d, β определяют скорость переходных участков в решении. Для математического ожидания точного решения СДУ (7) не существует ни явного формульного представления, ни замкнутой системы ОДУ для его численного расчета. Единственным конструктивным способом анализа нелинейных СДУ с большим шумом является ММК.

3. Описание параллельных программ

Описываются два способа распараллеливания алгоритмов на многопроцессорном кластере. Распараллеливание реализуется в системе параллельного программирования MPI [8]. Первый способ распараллеливания связан с оценкой математического ожидания решения СДУ на всем интервале интегрирования и отличается одинаковой продолжительностью моделирования всех траекторий из ансамбля. Поскольку в ММК моделируются независимые реализации решения СДУ, это допускает эффективную организацию их параллельного выполнения на многопроцессорном кластере. Схема распараллеливания в данном случае проста: различные процессоры вычислительной системы осуществляют полностью независимое решение, т.е. вычисляют последовательности, полученные на основе разных (в каждом процессоре) псевдослучайных чисел.

Пусть K - количество процессоров в вычислительной системе, реализующей алгоритм и M - размер ансамбля моделируемых траекторий. Положим $M_k = M/K$ - размер ансамбля на одном процессоре. Тогда формула для оценки математического ожидания решения в узле сетки t_n в параллельном исполнении имеет вид

$$m_n = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{M_k}\sum_{m=1}^{M_k} y_n^{(m,k)}. \tag{8}$$

Здесь $y_n^{(m,k)}$ - значение m -ой реализации решения СДУ в n -ом узле сетки, полученное на k -ом процессоре.

При распараллеливании данного алгоритма временные затраты сведены к минимуму: здесь время, затраченное на финальное осреднение независимых результатов, практически играет небольшую роль [9, 10].

Второй способ распараллеливания связан с моделированием траекторий до их первого выхода на границу заданной области и отличается различной продолжительностью моделирования каждой траектории из ансамбля. При этом фиксируется первый выход моделируемой последовательности $\{y_n\}$ за границу заданной области и фиксируется количество итерационных шагов, осуществленных до этого выхода. Под итерационным шагом здесь понимается вычисление следующего значения y_{n+1}. После чего процесс моделирования начинается заново из точки y_0.

Как и в предыдущем алгоритме, здесь моделируются независимые реализации решения СДУ и это позволяет эффективно организовать их параллельное выполнение. Пусть M^{min} - задаваемое минимальное количество выходов на границу области, определяющее размер ансамбля. Положим $M_k^{min} = M^{min}/K$. Тогда в параллельном исполнении формулу оценки среднего времени до первого выхода траекторий, стартующих из y_0, можно записать в виде

$$\hat{\tau}(y_0) = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{M_k^{min}}\sum_{m=1}^{M_k^{min}} n_m^{(k)}h. \tag{9}$$

Здесь $n_m^{(k)}$ - количество итерационных шагов, осуществленных до выхода на границу m -ой реализации на k -ом процессоре. Отметим, что по окончании моделирования число выходов реализаций на границу области может сильно разниться от процессора к процессору. Главное, что бы суммарное число выходов было не меньше M^{min}. Заметим, что в отличие

от первого алгоритма, здесь шаг h должен быть достаточно малым, что бы при фиксации момента выхода траектории за границу области не было большой потери точности оценки.

Схема распараллеливания здесь иная, чем в первом алгоритме, так как необходимо постоянно, через некоторое заданное количество итерационных шагов, сканировать по всем процессорам величины M_k^{min}, вычислять их сумму и проверять на достижение заданного общего минимума M^{min}. Количество итерационных шагов в каждом процессоре до выхода реализации за границы области сильно зависит от параметров в (3), причем приблизительное среднее число таких шагов неизвестно. Частота обменов данными между компьютерами оказывает сильное влияние на время выполнения параллельного алгоритма. Поэтому необходимо оптимизировать взаимодействия процессоров и тем самым минимизировать время решения задач при проведении большого количества численных экспериментов.

В алгоритмах 2 и 3 параллельные вычисления оптимизируются по двум параметрам: вначале а) во всех процессорах вычисляется размер временного интервала (рис.1), через который потом осуществляются межпроцессорные обмены данными и, затем, б) выбираются типы и схемы межпроцессорных обменов.

Интервал *Is* измеряется в количествах итерационных шагов и устанавливается одинаковым на всех процессорах. При вычислении интервала *Is* его размер динамически подстраивается к параметрам решаемой задачи. Поскольку алгоритм решения задачи один и тот же на всех процессорах, то и время выполнения одного итерационного шага будет одинаковым на всех процессорах. На рис. 1 PPi - логические MPI-номера процессоров, фигурными скобками обозначен диапазон для процессов, кривыми линиями обозначены моделируемые траектории, $n_m^{(k)}$ - количество итераций (шагов) до соответствующего выхода на границу, *Is* - интервал, через который процессоры взаимодействуют друг с другом. При вычислении *Is* используется синхронная, редуцированная MPI-функция ($MPI\_Allreduce(\ldots)$) [8].

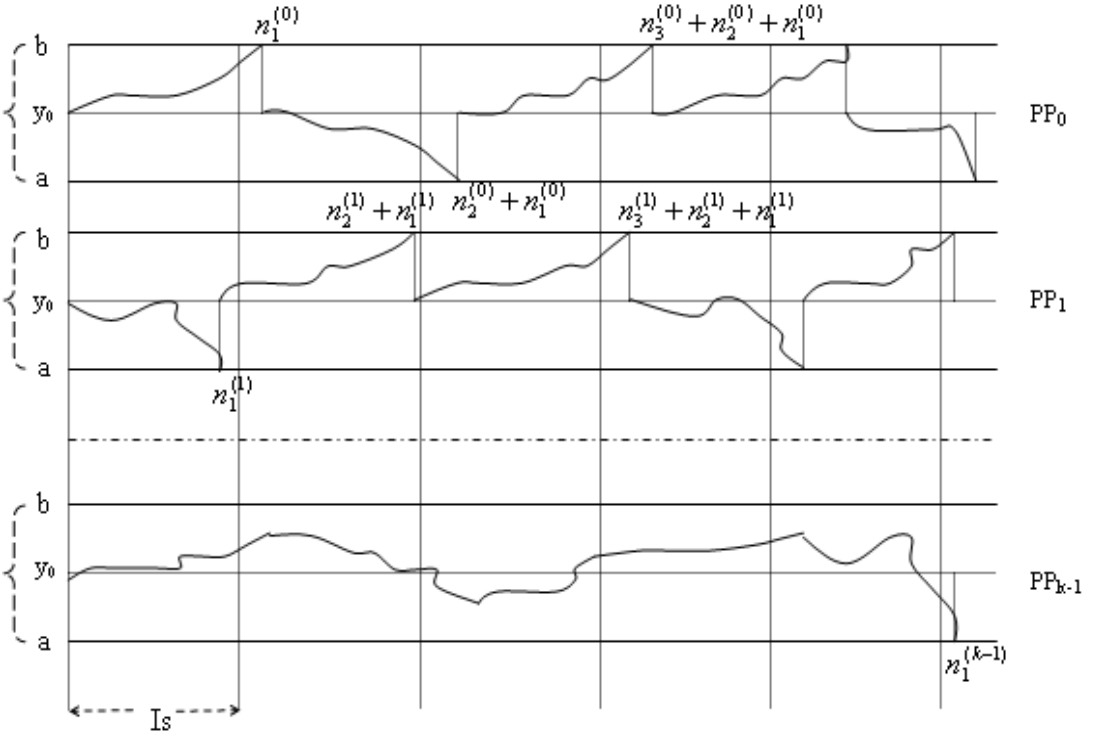

Рис. 1. Схема организации процесса вычислений для второго алгоритма

В данной работе реализуется следующая стратегия динамического выбора размера интервала *Is*. Пусть GSM_k^{min} - текущее суммарное (глобальное) по всем процессорам количество выходов реализаций, а через GSn_k - текущее суммарное (глобальное) по всем процессорам количество итерационных шагов, связанных с достигнутыми выходами.

1. В начальный момент счета устанавливается некоторый размер интервала $Is = N_{Is}$.

2. Если $GSM_k^{min} = 0$, то все процессоры увеличивают размер интервала Is в два раза. Таким образом, все процессоры в следующий раз выйдут на взаимодействие уже через интервал в два раза больший предыдущего.

3. Если $GSM_k^{min} \neq 0$ и $GSn_k \neq 0$, то каждый процессор присваивает своей переменной Is значение $C \cdot (GSn_k)/GSM_k^{min}$. После этого Is больше не меняется.

Здесь специально отметим, что суммируются только те итерационные шаги, которые связаны с фактом выхода за границы области. Отметим так же, что в результате редуцированной (MPI-) операции суммирования, значения счетчиков GSM_k^{min} и GSn_k будут находиться в каждом процессоре.

После вычисления интервала Is следует выбор типа и схемы взаимодействий процессоров. Вычисления продолжаются в двух вариантах: 1) с синхронными межпроцессорными взаимодействиями (2-й) алгоритм или 2) с асинхронными взаимодействиями (3-й алгоритм).

В решаемой задаче на каждом интервале Is суммируются по всем процессорам три величины, накапливаемые в течении текущего интервала: LM_k^{min} - счетчик количества выходов реализаций на границы области, LS_{nk} - счетчик суммарного количества шагов, связанных с этими выходами и LS_{vk} - счетчик вторых моментов.

Во втором алгоритме тип обменов между процессорами - синхронный, схема обменов - "взаимодействующие равные". Суммирование указанных счетчиков осуществляется синхронной редуцированной MPI-функцией ($MPI\_Allreduce(\ldots)$).

В третьем алгоритме тип обменов - асинхронный, схема обменов "управляющий-рабочие". Нулевой процессор - управляющий, все остальные - рабочие. Управляющий принимает от рабочих счетчики, суммирует их и передает рабочим флаг завершения вычислений. Рабочие процессоры взаимодействуют с управляющим процессором асинхронно.

Вычисления векторизованы и программы решаемых задач за счет векторизации вычислений ускоряются на $50\% - 60\%$ (на каждом процессоре) по сравнению с этими же программами, но без векторизации вычислений.

В решаемых задачах необходимый объем выборки базовых случайных чисел очень велик ($\approx 2 \cdot 10^{14}$), поэтому целесообразно использовать "длиннопериодные" псевдослучайные последовательности. В описываемых ниже расчетах используется генератор псевдослучайных чисел - MT2203. Это фактически набор из 1024-х генераторов псевдослучайных чисел, которые предназначены для использования в параллельных моделированиях ММК. Каждый из них генерирует последовательность с периодичностью, равной 2 в степени 2203. Параметры генераторов обеспечивают взаимную независимость соответствующих последовательностей псевдослучайных чисел. Одним из таких параметров является идентификационный логический номер процессора, назначаемый системой параллельного программирования MPI [8].

4. Численные эксперименты

Вычисления производились на кластере НКС-30Т ССКЦ ИВМиМГ СО РАН. Далее под словом процессор имеется ввиду процессорное ядро.

Тест 1.

Для решения СДУ в смысле Ито вида

$$\begin{aligned} dy &= y\, dw(t), \quad 0 \leq t \leq 10, \\ y(0) &= 1 \end{aligned} \tag{10}$$

моделируемого по формуле

$$y_n = y_{n-1} \exp\left(-\frac{h}{2} + \sqrt{h}\ \xi_n\right), \quad y_0 = 1,$$

оцениваются первый и второй момент случайной величины τ - времени первого выхода реализаций на границу интервала $[0,\ 2]$. В табл. 1 проведены результаты расчетов по формуле (9) при $K = 64, h = 10^{-4}$ для размеров ансамбля моделируемых траекторий $M = 10^2, 10^4, 10^7$. В данном тесте точные значения $E\tau$ и $E\tau^2$ неизвестны, тем не менее

Таблица 1.

M	$\hat{\tau}$	$\hat{\tau}^2$	Время счета (сек)
10^2	171.4	223305.1	1.37
10^4	714.9	1011618.6	104.41
10^7	712.1	1008687.7	110559.60

из расчетов видно большое различие в оценках при $M = 10^2$ и $M = 10^4,\ 10^7$, что говорит о необходимости моделирования ансамблей максимального размера в задачах о достижении границ. Критерием точности может служить число совпадающих значащих цифр оценок для разных M. Малое значение оценки $\hat{\tau}^2$ для $M = 10^2$ связано с отсутствием в выборке редких реализаций, долго не выходящих на границу интервала.

В таблице 2 приведена зависимость времени счета от числа используемых процессоров для $M = 10^4$.

Таблица 2.

K	1	8	16	32	64
Время счета (сек)	3088	528	267	167	104

Тест 2.

Если в (3) положить $f(y) = 0,\ \sigma(y) = \sigma$, то сопутствующее СДУ является СДУ с аддитивным шумом:

$$dy = \sigma\, dw(t),\ y(0) = y_0, \tag{11}$$

решением которого является винеровский процесс

$$y(t) = y_0 + \sigma\, w(t), \tag{12}$$

имеющий математическое ожидание $Ey(t) \equiv y_0$ и второй момент

$$Ey^2 = E(y_0 + \sigma\, w(t))^2 = y_0^2 + \sigma^2 t.$$

Для процесса (12) известно точное среднее время первого выхода траекторий из $[a, b]$ (см., например, [12]):

$$E\tau(y_0) = \frac{(y_0 - a)(b - y_0)}{\sigma^2}. \tag{13}$$

Из (13) видим, что при уменьшении σ математическое ожидание τ растет как σ^{-2}. Для второго момента имеем

$$E\tau^2(y_0) = \frac{1}{3\sigma^4}\,[\,(b-a)^3(y_0-a) - 2\,(b-a)(y_0-a)^3 + (y_0-a)^4\,] \tag{14}$$

и рост уже как σ^{-4}. Такая зависимость от σ связана с продолжительным временем достижения процессом (12) границ интервала $[a, b]$.

Для решения СДУ (11), моделируемого по формуле

$$y_n = y_{n-1} + \sigma\xi_n\sqrt{h},$$

оценивается среднее время и второй момент величины τ - времени первого выхода реализаций на границу интервала $[-1, 1]$. Из (13) и (14) имеем

$$E\tau(y_0) = \frac{1-y_0^2}{\sigma^2},$$

$$E\tau^2(y_0) = \frac{1}{3\sigma^4}\left[8\,(y_0+1) - 4\,(y_0+1)^3 + (y_0+1)^4\right].$$

В табл. 3 проведены результаты расчетов при $K = 64,\ h = 10^{-4}$ для размеров ансамбля $M = 10^2, 10^3, 10^6, 10^8$. В первом случае, при $y_0 = 0$, траектории СДУ стартуют из центра интервала $[-1, 1]$. Во втором случае, $y_0 = 0.9$, траектории стартуют из точки вблизи правой границы интервала и сильное влияние на точность оценок оказывают редкие реализации, выходящие на левую границу интервала. Расчеты показывают сильное различие в точно-

Таблица 3.

y_0	σ	$E\tau(y_0)$	$E\tau_2(y_0)$	M	$\hat{\tau}$	$\hat{\tau}^2$	Время счета (сек)
0	10^{-1}	10^2	$1.667\cdot10^4$	10^2	65.4	5652.9	0.17
				10^3	95.5	15132.7	0.87
				10^6	100.1	16726.8	749.88
				10^8	100.1	16709.3	79026.5
	10^{-2}	10^4	$1.667\cdot10^8$	10^2	5644.1	41852534.9	6.05
				10^3	9114.6	135506216.4	52.66
				10^6	10004.2	166840038.2	60655.44
				10^8	–	–	–
0.9	10^{-1}	19	$2.654\cdot10^3$	10^2	1.4	3.6	0.06
				10^3	10.4	714.6	0.27
				10^6	19.0	2663.6	177.50
				10^8	19.1	2674.1	18126.60
	10^{-2}	1900	$2.654\cdot10^7$	10^2	117.9	23078.5	0.34
				10^3	1034.8	6664365.1	11.07
				10^6	1904.5	26630171.4	13887.96
				10^8	–	–	–

сти оценок для разных M, как для случая y_0 в центре интервала $[-1, 1]$, так и вблизи границы для различных σ. Во всех тестах очень низкая точность оценок вторых моментов для всех $M = 10^2, 10^3$. Большие размеры моделируемого ансамбля для $K = 64$ требуют слишком большого времени счета, более двух суток. С этим связано отсутствие некоторых результатов расчетов при $M = 10^8$.

В таблице 4 приведены временные характеристики счета 2-го (синхронного) и 3-го (асинхронного) алгоритмов для этого же теста при K =16, 32 и 64, размеров ансамбля $M = 10^6$ и для двух точек старта реализаций: $y_0 = 0$ и $y_0 = 0.9$. Из таблицы видно, что при K=16 и 32 времена вычислений обоих алгоритмов близки для малого (2048 шагов) и большого (131072 шагов) размеров интервала *Is*. Но при K =64 для *Is* =2048 асинхронный

алгоритм (254 сек.) уступает синхронному (177 сек.), а для $Is = 131072$ наоборот, асинхронный (540 сек.) уже существенно превосходит синхронный (749 сек.). Отсюда следует, что после нахождения интервала Is, можно минимизировать дальнейшие вычисления. Например, для $Is \leq 32768$ - реализуется синхронный алгоритм, а для $Is > 32768$ - асинхронный алгоритм.

Таблица 4.

y_0	Is	Тип алгоритма	16 проц. (сек)	32 проц. (сек)	64 проц. (сек)
0.0	131072	Синхронный	2072	1050	749
		Асинхронный	2083	1065	540
0.9	2048	Синхронный	474	263	177
		Асинхронный	449	294	254

Тест 3.

Оценка математического ожидания и второго момента решения $y(t)$ линейного СДУ (6). Математическое ожидание при начальных условиях $m(0) = 1$, $\frac{dm}{dt}(0) = 0$ задается формулой $m(t) = cos(\beta t)$. При $\beta = 2\pi$ функция $m(t)$ на интервале $[0, 100]$ имеет 100 периодов колебаний с одинаковой амплитудой 1. СДУ (5) можно переписать в виде линейной системы

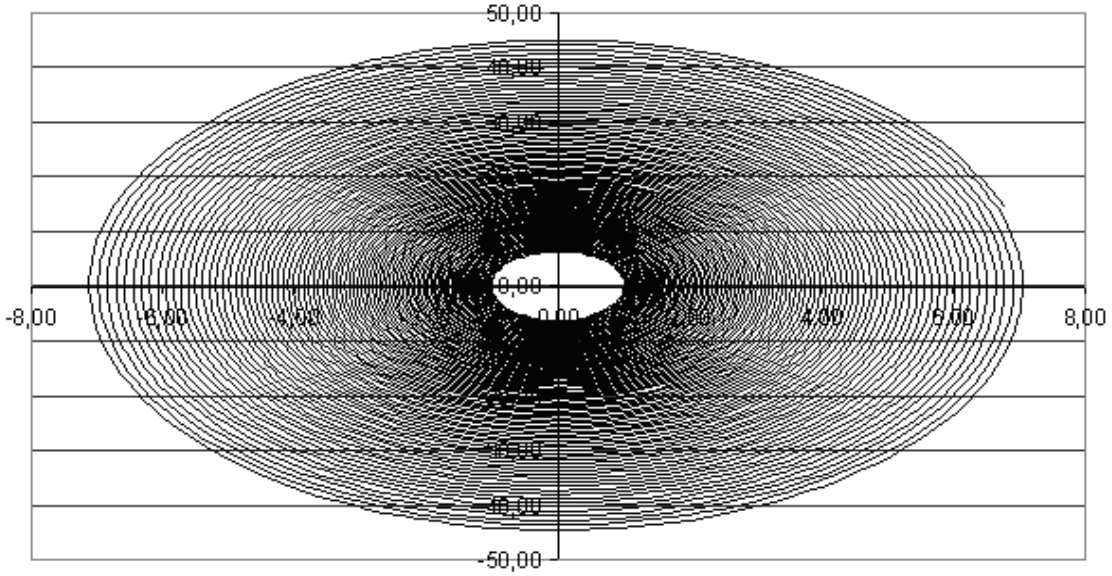

Рис. 2. Фазовая траектория оценки математического ожидания решения СДУ (6)

$$\begin{aligned} dy_1 &= y_2\, dt, \\ dy_2 &= -(\beta^2 y_1 + \lambda\, y_2)\, dt - \sigma_1\, y_2\, d\omega_1(t) - \sigma_2\, y_1\, d\omega_2(t). \end{aligned} \tag{15}$$

Применяя метод Эйлера (2) к СДУ (6), записанному в виде системы первого порядка (15),

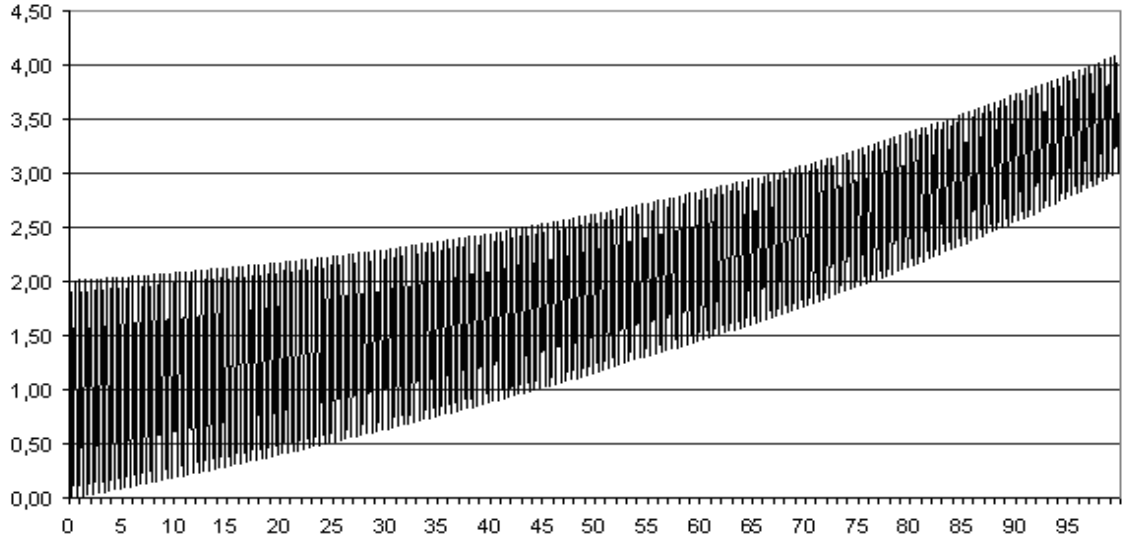

Рис. 3. График оценки второго момента решения СДУ (6)

получим следующую разностную схему:

$$\begin{aligned} y_{n+1}^{(1)} &= y_n^{(1)} + h y_n^{(2)}, \\ y_{n+1}^{(2)} &= y_n^{(2)} - h\beta^2 y_n^{(1)} + \sqrt{h}\, y_n^{(1)}\, \xi_{n+1}\,. \end{aligned} \tag{16}$$

На рис. 2 приведён график фазовой траектории $(Ey_n^{(1)},\, Ey_n^{(2)})$, расчитанной с помощью разностной схемы (16) с шагом $h = 10^{-3}$ и размером ансамбля моделируемых траекторий решения $M = 8^7$.

Расчеты показывают, что при таком размере шага интегрирования наблюдается рост амплитуды колебаний оценки математического ожидания. Устойчивую фазовую траекторию в виде эллипса $(cos\, 2\pi t,\, -2\pi\, sin\, 2\pi t)$ удаётся получить только при шаге интегрирования $h = 10^{-6}$ и меньше.

В оценке второго момента, полученной с шагом $h = 10^{-3}$, имеем максимальное значение равное 205, что говорит о полной потере точности оценки с таким шагом. Удовлетворительная точность получается только при h не более 10^{-6}, когда имеем оценку максимального значения, близкую к точному значению равному 4 (рис.3). Число моделируемых траекторий выбиралось настолько большим, чтобы в данном расчете размер ансамбля моделируемых траекторий не оказывал влияния на точность оценок.

Рост дисперсии решения СДУ (6) со временем делает проблемным точную оценку математического ожидания $Ey(t) = cos\, 2\pi t$ на большом числе периодов колебаний и требует увеличения размера ансамбля моделируемых траекторий при увеличении T, что естественно увеличивает время счёта задачи.

Время счёта на 64 процессорах с шагом $h = 10^{-6}$, $M = 8^6$ составило около 16 часов. Расчёты с более мелким шагом интегрирования требуют многосуточных вычислений.

Тест 4.

Оценка математического ожидания и второго момента решения нелинейного уравнения Ван-дер-Поля (7) с параметрами $\lambda = 20$, $b = 1$, $\beta = 2\pi$, $\sigma = 1$.

На рис. 4 приведён график одной смоделированной траектории решения $y_1(t)$ системы СДУ (7) с шагом $h = 10^{-8}$ При так выбранном параметре $\lambda = 20$ стохастическое уравне-

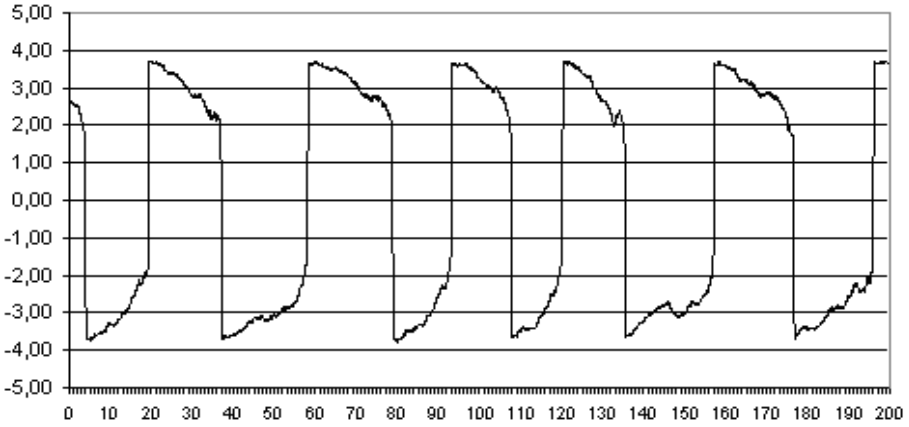

Рис. 4. График одной смоделированной траектории уравнения Ван-дер-Поля (7)

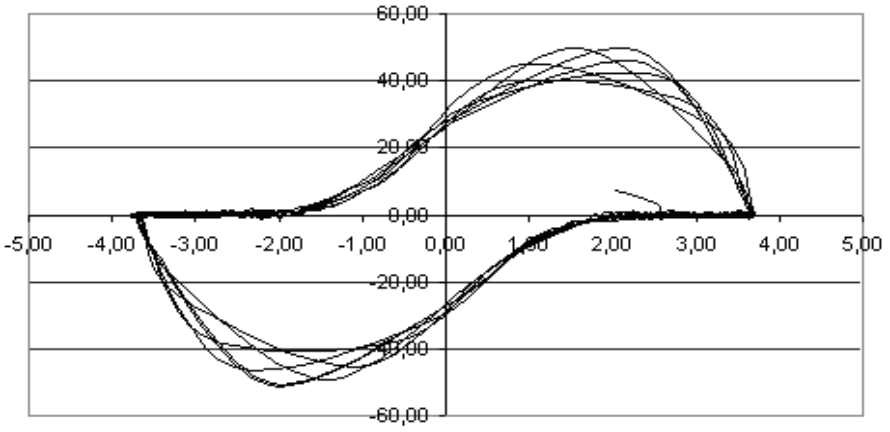

Рис. 5. Фазовая траектория решения уравнения Ван-дер-Поля (7)

ние Ван-дер-Поля можно считать "жестким" [4], имеющим быстрые переходные участки и "полочки" вблизи значений ± 3.

На рис. 5 приведён график смделированной фазовой траектории $(y_1(t), y_2(t))$. Как видно из рис.4 и рис.5, амплитуда колебаний траектории решений не уменьшается. Однако, численные расчёты показывают, что этого нельзя сказать о поведении математического ожидания $Ey_1(t)$.

На рис. 6 приведён график оценки математического ожидания $Ey_1(t)$, полученной с шагом $h = 10^{-8}$ и с размером ансамбля $M = 8^5$. Расчеты показывают, что амплитуда колебаний $Ey_1(t)$ уменьшается от периода к периоду, что находится в резком контрасте с поведением математического ожидания в линейном колебательном контуре (Тест 3). Можно сказать, что математическое ожидание нелинейного СДУ со временем "теряет информацию" о поведении каждой отдельной траектории решения (7). Дополнительные расчёты показали, что и в случае малой "жёсткости" (с $\lambda = 1$) математическое ожидание решения (7) также имеет убывающую со временем амплитуду колебаний. Такое резкое различие в поведении математического ожидания решений линейных и нелинейных осциляторов предупреждает об опасностях решения нелинейных СДУ с помощью их линеаризации.

На рис. 7 приведён график оценки второго момента решения $Ey_1^2(t)$. Как видим, достаточно быстро происходит стабилизация дисперсии решения, что также отлично от пове-

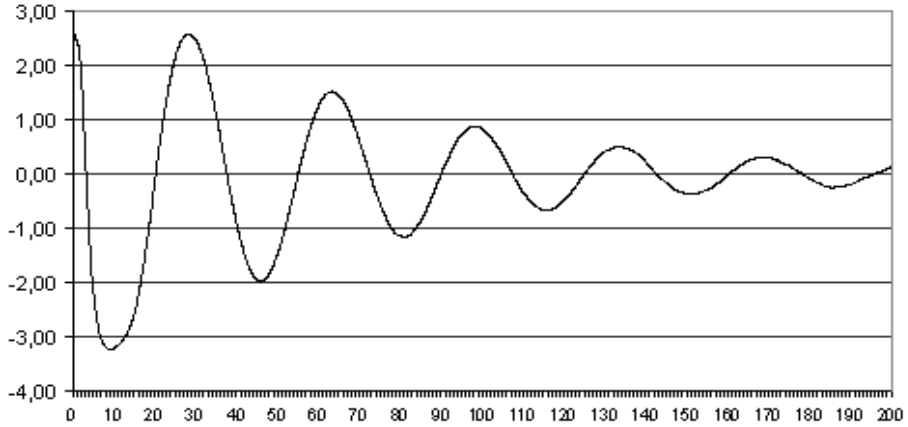

Рис. 6. График оценки математического ожидания решения уравнения Ван-дер-Поля (7)

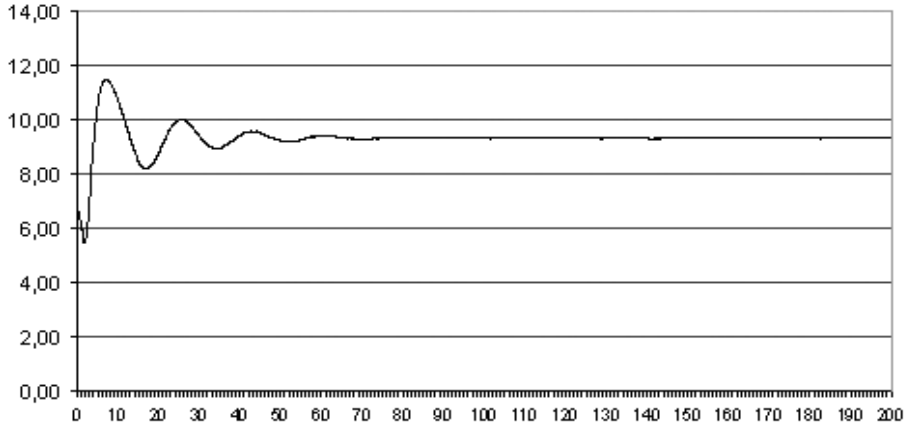

Рис. 7. График оценки второго момента решения уравнения Ван-дер-Поля (7)

дения дисперсии в линейном случае. Время счета этого теста на 128 процессорах с шагом $h = 10^{-8}$ и $M = 8^5$ составило около 73 часов. Отметим, что использование большего размера шага интегрирования ведёт к неустойчивости численного решения, что в конечном итоге приводит к переполнению разрядной сетки арифметического устройства процессоров.

5. Заключение

Расчеты наглядно показывают необходимость использования суперкомпьютеров для численного решения СДУ. Кроме того, проведенные исследования показали, что при численном анализе стохастическимх осцилляторов методом Монте-Карло использование больших размеров шагов интегрирования решений СДУ, что практически всегда делается при проведении численных экспериментов на ПК, могут приводить к совершенно ошибочным выводам.

Так же отметим, что сделанные выводы по поведению моментов решения стохастического уравнения Ван-дер-Поля с большим шумом невозможно получить никаким другим методом, кроме статичтического моделирования. Это касается как методов гаусовской аппроксимации, так и спектрального метода [13].

Литература

1. Диментберг М. Ф. Нелинейные стохастические задачи механических колебаний. —М.: Наука, 1980.
2. Бабицкий В. И. Теория виброударных систем. —М.: Наука, 1978.
3. Артемьев С.С., Якунин М. А. Математическое и статистическое моделирование в финансах. —Новосибирск: Изд. ИВМиМГ СО РАН, 2008.
4. Артемьев С.С. Численные методы решения задачи Коши для систем обыкновенных и стохастических дифференциальных уравнений. —Новосибирск: Изд. ВЦ СО РАН, 1993.
5. Артемьев С.С., Корнеев В.Д. Численное решение стохастических дифференциальных уравнений на суперкомпьютерах //Сиб. ЖВМ / РАН. Сиб. отд-ние. —Новосибирск, 2011. —Т. 14, № 1. —С. 5–17.
6. Ермаков С.М., Михайлов Г.А. Статистическое моделирование. —М.: Наука, 1982.
7. Ермаков С.М., Некруткин В.В., Сипин А.С. Случайные процессы для решения классических уравнений математической физики. —М.: Наука, 1984.
8. Snir M., Otto S. W., Huss–Lederman S., Walker D., and Dongarra J.. MPI: The Complete Reference. MIT Press. Boston, 1996.
9. Корнеев В.Д. Параллельное программирование в MPI. —Москва–Ижевск: Институт компьютерных исследований, 2003, —304 с.
10. Корнеев В.Д. Параллельное программирование кластеров: учеб. пособие/—Новосибирск: Изд-во НГТУ, 2008. —312 с.
11. http://software.intel.com/ru-ru/intel-mkl/
12. Аверина Т.А., Артемьев С.С. Анализ точности методов Монте-Карло при решении краевых задач посредством вероятностного представления//Сиб.ЖВМ/РАН. Сиб. отделение. —Новосибирск, 2008. —Т. 11, № 3. —С. 239–250.
13. Пантелеев А.В., Рыбаков К.А., Сотскова И.А. Спектральный метод анализа нелинейных стохастических систем управления. —М.: Вузовская книга, 2006.