заметно снизился, так как постепенно стало выясняться, что почти все они катастрофически неустойчивы, имеют сложные вычислительные схемы, требуют непомерно большого числа процессоров и очень много памяти, процессоры загружены крайне слабо и т.п. Единственными исключениями являются алгоритмы сдваивания для вычисления суммы или произведения n чисел. Эти алгоритмы применяются на практике достаточно часто. Почти все сверхбыстрые алгоритмы основаны на математически эквивалентных преобразованиях. Но все же хочется верить, что последнее слово в их исследовании еще не сказано. Ведь до сих пор очень мало что известно об алгоритмах, которые на множестве математически эквивалентных преобразований обеспечивают достижение тех или иных оптимальных вычислительных характеристик.

ЛЕКЦИЯ 6

Компьютеры и ошибки округления

Содержание: *позиционные системы счисления, ошибки округления, наилучшее округление, преимущества сокращенных систем счисления, фиксированная и плавающая запятая, машинный нуль, точность представления чисел, обоснование вероятностных свойств ошибок округления, особенность операций сложения и вычитания, двоичная система счисления не является лучшей, ошибки округления иногда помогают.*

Известно, что при решении задач на компьютере неизбежно возникают ошибки округления. Они крайне малы, но при решении больших задач их появляется очень много. Поэтому в совокупности они могут оказывать значительное влияние на точность получаемых результатов. Тем не менее, в курсах вычислительной математики ошибкам округления уделяется неоправданно мало внимания. Возможно, именно поэтому вокруг ошибок округления возникает немало необоснованных мнений и даже мифов.

Например, в одном из них утверждается, что при решении задач на вычислительных системах параллельной архитектуры влияние ошибок округления уменьшается и это уменьшение тем значительнее, чем больше параллелизм. В обоснование этого тезиса даже приводится вроде бы вполне разумный довод. Он сводится к тому, что на параллельных системах в каждый момент времени выполняются независимые операции. А поскольку независимые операции порождают не связанные между собой ошибки округления, то и совокупное влияние ошибок на весь вычислительный процесс становится меньше. Но как было показано в предыдущей лекции, при заданном способе округления и фиксированном множестве операций ошибки округления зависят только от *графа алгоритма*, а совсем не от того, на какой вычислительной системе, последовательной или параллельной, реализуется сам алгоритм.

Широко распространен миф, согласно которому ошибки округления можно считать случайными величинами. Но, естественно, возникает вопрос, почему они случайные и что это означает? Имеются немало и других необоснованных постулатов. Чтобы они не становились руководством к действию, познакомимся с ошибками округления поближе.

Общий эффект от решения задачи и даже возможность ее решения во многом определяется тем, как в действительности выполняются операции над числами. А это в свою очередь зависит от принятой системы записи чисел или, как говорят, *системы счисления.* Наиболее совершенным принципом записи является тот, на котором основана общепринятая десятичная система. Создание вычислительной техники не связано с какими-либо принципиально другими системами счисления. Запись чисел, с которыми оперирует компьютер, основана на той же идее, что и десятичная система. В математическом плане изменения невелики и заключаются в следующем.

Зафиксируем некоторое целое положительное число $p>1$ и целые числа α_0, $\alpha_1,\ldots,\alpha_{p-1}$. Пусть любое неотрицательное число x может быть представлено в виде ряда

$$b_n p^n + b_{n-1} p^{n-1} + \ldots + b_0 + b_{-1} p^{-1} + b_{-2} p^{-2} + \ldots,$$

где каждый из коэффициентов b_i может принимать одно из значений α_0, $\alpha_1,\ldots,\alpha_{p-1}$. Перечислив подряд все коэффициенты и приписав слева знак числа, получаем похожую на десятичную запись

$$x = \pm b_n b_{n-1} \ldots b_0, b_{-1} b_{-2} \ldots$$

Такие формы записи чисел называются *позиционными* системами счисления. Их название связано с тем, что роль, которую играет каждое число b_i в записи, зависит от занимаемой им позиции. Отсчет позиции определяется положением запятой или, что то же самое, положением коэффициента b_0.

В литературе, связанной с вычислительной математикой, слово "позиция" чаще всего заменяется словом "разряд". Нумерация разрядов устанавливается в убывающем порядке подряд слева направо, причем первый разряд слева от запятой имеет нулевой номер. Различаются разряды числа до запятой и разряды после запятой. Число p называется *основанием* системы счисления, числа α_0, $\alpha_1,\ldots,\alpha_{p-1}$ – *базисными.* Если используется система счисления с основанием p, то правую часть представления числа x называют p-ичной дробью. Дробь называется *бесконечной*, если в ее записи имеется бесконечно много ненулевых коэффициентов, и *конечной* в противном случае. Обычно в записи дроби опускаются все первые и последние нулевые коэффициенты. Опускается и запятая, если все коэффициенты после нее являются нулевыми.

Выбор базисных чисел α_0, α_1,…, α_{p-1} определяется в основном требованиями удобства работы с вещественными числами в данной системе счисления. Не видно каких-либо особых преимуществ, которое дало бы использование базисных чисел, превосходящих по модулю основание системы счисления. Поэтому будем считать, что $|\alpha_k|<p$ для всех k=0,1,…, p–1. В вычислительной технике чаще всего используются системы счисления с базисными числами $\alpha_k=k$. В дальнейшем, если не сделано каких-либо оговорок, выполнение этого условия предполагается.

Арифметические операции над числами, заданными в любой позиционной системе счисления, реализуются по таким же правилам, что и в десятичной системе. При этом используются таблицы сложения и умножения не десятичной системы, а системы с основанием p. Позиционные системы счисления широко используются для представления чисел в вычислительной технике. Наиболее часто применяется простейшая из них – *двоичная* система счисления. Использование именно позиционных систем объясняется возможностью реализации в них достаточно простых алгоритмов выполнения арифметических операций над числами.

Никакие технические средства не позволяют выполнять операции над числами, заданными бесконечными дробями. Поэтому замена любого числа конечной дробью является необходимой операцией. *Округлением* числа x до s разрядов в заданной системе счисления называется операция замены этого числа таким числом x_s, все младшие разряды которого в той же системе счисления, начиная с s–1-го, являются нулевыми. Разность $x_s–x$ называется *ошибкой округления*.

Заметим, что в данном определении ничего не говорится ни о способе выполнения операции округления, ни о том, насколько округленное число близко к округляемому. Это не случайно. В практике конструирования компьютеров операции округления реализуются самыми различными способами. Единственное, что их объединяет, – это малость в том или ином смысле ошибок округления, по крайней мере, для большинства чисел.

Один из простейших способов округления заключается в следующем. Пусть задана p-ичная дробь $x=\pm b_n\ldots b_s b_{s-1} b_{s-2}\ldots$. В качестве результата выполнения операции округления числа x до s разрядов берется число $x_s=\pm b_n\ldots b_s$. Основное достоинство данного способа округления – простота реализации. Однако сразу же видны и некоторые недостатки.

Предположим, что в качестве базисных берутся числа 0, 1,…, p–1. Тогда для ошибки округления справедливо соотношение $|x_s–x|\le p^s$. Равенство достигается только в одном случае, когда число x представлено бесконечной дробью и во всех его младших разрядах, начиная с s–1-го, стоят числа p–1. Уже сравнение с общепринятым "школьным" правилом округления в десятичной системе показывает, что в рассмотренном способе оценка ошибки вдвое больше. Но более важным является то, что независимо от своей величины ошибка округления всегда имеет вполне определенный знак, противоположный знаку округляемого числа. Это явление нежелательно, так как оно приводит к более быстрому накоплению ошибок округления.

Хотя описанный способ округления чисел и не является лучшим, тем не менее именно с ним тесно связаны все другие способы округления. В самом деле, как бы ни выполнялась операция округления, ее результатом будет число, все младшие разряды которого, начиная с s–1-го, являются нулевыми. Следовательно, операцию округления всегда можно трактовать как отбрасывание всех разрядов, начиная с s–1-го, и последующее добавление или вычитание некоторого числа, кратного p^s. Для того чтобы ошибка округления была малой, необходимо и достаточно, чтобы было малым именно это число.

Числа, имеющие нулевые младшие разряды, начиная с s–1-го, образуют на вещественной оси равномерную сетку с шагом p^s. Среди этих чисел есть число x^*_s, наиболее близкое к x. Ясно, что $| x^*_s - x| \leq p^s/2$. Наилучшее приближение x^*_s к x будет единственным, если имеет место строгое неравенство, и таких приближений будет два, если имеет место равенство. Замена числа x числом x^*_s является операцией округления до s разрядов, причем лучшей во многих отношениях. Однако по сравнению с описанной ранее операцией она имеет два существенных недостатка. Во-первых, для ее реализации необходимо всегда дополнительно осуществлять проверку, какой из двух кандидатов должен быть взят в качестве округленного числа x^*_s. После этой проверки примерно в половине случаев нужно дополнительно выполнить еще одну операцию сложения чисел. Так как округление осуществляется после каждой арифметической операции, реализация наилучшего по точности округления приводит к замедлению работы арифметических устройств компьютера. Кроме этого, есть некоторая неоднозначность в выполнении данного варианта операции округления, когда имеется два наилучших приближения x^*_s к числу x. Хотя такая неоднозначность встречается относительно редко, она совсем не безобидна, как будет показано в дальнейшем.

Естественным является желание объединить достоинства обоих способов округления. Покажем, как этого можно добиться путем использования специальных систем счисления.

До сих пор предполагалось, что в качестве базисных чисел p-ичной системы счисления используются числа 0, 1,…, p–1. При этом оказалось, что лучший по точности способ округления не является самым простым в реализации и приводит к замедлению выполнения арифметических операций. Рассмотрим теперь p-ичные позиционные системы счисления с другими наборами базисных чисел. Пусть основание p системы счисления является нечетным и в качестве базисных выбраны числа $\alpha_k=(1+2k-p)/2$, k=0, 1,…, p–1. Такая система называется *сокращенной*.

В этом случае для всех k из заданного диапазона выполняется неравенство $|\alpha_k|\le(p-1)/2$. Допустим, что перед округлением до s-го разряда округляемое число было представлено p-ичной дробью до r-го разряда, где r – целое число и $r<s$. Снова рассмотрим число x_s. Однако теперь, принимая во внимание неравенства $p>1$, $r-s+1\le0$, получаем, что

$$|x_s-x|\le|\alpha_{s-1}p^{s-1}+\ldots+\alpha_r p^r|\le((p-1)p^{s-1}/2)(1+p^{-1}+\ldots+p^{r-s+1})=$$

$$=((p-1)p^{s-1}/2)(p-p^{r-s+1})/(p-1)<p^s/2.$$

Из этих соотношений вытекает несколько интересных выводов. Главный из них состоит в том, что в любой сокращенной системе счисления простое отбрасывание всех младших разрядов, начиная с s–1-го, дает правильно округленное число. В таких системах нет необходимости задавать отдельно знак числа, так как он совпадает со знаком старшего разряда. В любой сокращенной системе не возникает никакой неоднозначности при правильном округлении, о чем говорилось ранее. Можно также показать, что $x_s>x$, если первый из ненулевых отброшенных разрядов отрицательный, и $x_s<x$, если он положительный. Число $p^s/2$ не может быть представлено конечной дробью и т.д.

Среди сокращенных позиционных систем счисления простейшей является *троичная* система. Как уже отмечалось, в современной вычислительной технике наиболее широко используется двоичная система. С точки зрения округления чисел этот выбор не является лучшим. Но если троичная система счисления так хороша, то почему же нет большого числа построенных на ней компьютеров?

Рядовой пользователь не видит систему счисления, лежащую в основе работы компьютера, поскольку он отгорожен от нее языком программирования. С точки зрения пользователя троичная система хороша только тем, что позволяет без каких-либо дополнительных усилий получать более точные результаты. Безусловно, это очень важно. Однако достижение лучшей точности очень редко становится приоритетным условием для конструкторов компьютеров. Поэтому в первую очередь на выбор системы счисления влияют другие факторы. Это и трудности построения большого числа базисных элементов, имеющих 3 устойчивых состояния, и традиции конструирования элементной базы, и конкуренция в среде производителей компьютеров, и многое другое.

Тем не менее, совсем не очевидно, на каких принципах и с какой целевой функцией будут строиться компьютеры в будущем. И вполне возможно, что троичная система счисления еще окажется востребованной. Тем более что успешный опыт создания компьютера на такой системе имеется. Это машина "Сетунь". Она была разработана в вычислительном центре Московского университета в конце 50-х годов прошлого столетия. Ее главным конструктором является к.т.н. Н.П.Брусенцов, программное обеспечение выполнялось под руководством проф. Е.А.Жоголева.

Запоминание цифровой информации во всех компьютерах основано на использовании достаточно простых однотипных элементов. Каждый из таких элементов представляет некоторое физическое устройство, имеющее p устойчивых состояний, где $p>1$. При этом само устройство допускает возможность перевода любого своего состояния в любое другое. Эти элементы называются *базисными* и служат для моделирования одного числового разряда p-ичной системы счисления.

Компьютер не может содержать бесконечно много базисных элементов. Поэтому он всегда имеет возможность оперировать лишь с конечным числом конечных p-ичных дробей. Это важный вывод, из которого вытекают все основные особенности компьютерной арифметики.

Требование унифицированного выполнения арифметических операций над числами приводит к необходимости унифицированного изображения в компьютере всех конечных дробей. Будем для простоты рассматривать дроби без знака, считая, что их знак либо учитывается в самой системе счисления, либо изображается каким-то иным способом.

Пусть на изображение каждой дроби отводится одно и то же число τ базисных элементов. Ясно, что на τ элементах можно изображать не более τ разрядов любого числа. Чтобы это изображение можно было прочитать, необходимо установить взаимно однозначное соответствие между базисными элементами, отведенными для изображения каждого числа, и положением разрядов в числе относительно запятой. В зависимости от того, является ли это соответствие одним и тем же для всех изображаемых чисел или зависит от самого числа, различают два способа представления чисел в компьютере. Называются они представлением с фиксированной и плавающей запятой.

Предположим, что каждые τ базисных элементов служат для изображения τ последовательных разрядов чисел. Причем положение этих разрядов относительно запятой фиксировано и является одним и тем же для всех дробей. Будем считать, что на изображение разрядов, стоящих слева от запятой, отводится r элементов, где $r\geq 0$. Такой способ представления чисел называется представлением с *фиксированной запятой*. С помощью этого способа можно точно запоминать любую из конечных p-ичных дробей, имеющих не более r ненулевых разрядов слева от запятой и не более $\tau - r$ ненулевых разрядов справа от запятой. Все такие дроби x лежат в диапазоне $-p^r < x < p^r$.

Один из недостатков представления чисел с фиксированной запятой виден сразу. Если p-ичная дробь много меньше p^r по модулю, то бóльшая часть из отведенных базисных элементов изображает старшие нулевые разряды и фактически не используется. Поэтому приближение числа такой дробью связано с большой относительной ошибкой. Однако для чисел, близких к p^r по модулю, для представления старших ненулевых разрядов используются все τ базисных элементов. В этом случае относительная ошибка приближения числа дробью является минимальной. *Абсолютная ошибка* представления чисел с фиксированной запятой всегда лежит в одних и тех же пределах независимо от величины самих чисел.

Представление чисел с *плавающей запятой* заключается в следующем. Всякое ненулевое число x можно записать в виде $x=a \cdot p^b$, где b – целое число и $1/p \leq |a| < 1$. Число a называется *мантиссой* числа x, число b – его *порядком*. Пусть на изображение порядка без знака отводится r базисных элементов, на изображение мантиссы без знака $\tau-r$ элементов. Если теперь порядок и мантисса представлены как дроби с фиксированной запятой, то это и будет представлением числа x с плавающей запятой.

Заметим, что порядок всегда представляется точно, так как он является целым числом. Мантисса же будет представлена точно лишь для тех p-ичных дробей, которые имеют не более $\tau-r$ ненулевых старших разрядов. Точно или с округлением могут быть представлены с плавающей запятой только те числа x, порядок которых удовлетворяет неравенству $|b| < p^r$. Независимо от величины этих чисел *относительная ошибка* их представления лежит в одних и тех же пределах. Числа, для которых $b > p^r$, не могут быть представлены с плавающей запятой в компьютере с заданной величиной r. Все числа, для которых $b < -p^r$, также как и число 0, заменяются в компьютере числом с нулевой мантиссой. В разных компьютерах порядок таких чисел может быть разным. Число ω, для которого $a=1/p$ и $b=-p^r+1$ называется *машинным нулем*. Оно совпадает с минимальным положительным числом, которое можно представить с плавающей запятой в компьютере при заданных числах r и p.

В современных компьютерах применяются оба способа представления чисел. Выбор того или иного способа зависит от типа решаемых задач. На компьютерах и больших вычислительных системах широкого назначения нередко допускается использование обеих форм представления чисел.

Операции над числами с фиксированной запятой выполняются быстрее, чем над числами с плавающей запятой. Это связано с тем, что при реализации операций в режиме с плавающей запятой по существу приходится выполнять все действия с парами чисел с фиксированной запятой. Поэтому при решении тех задач, где положение запятой в числовых данных более или менее определено, использование представления чисел с фиксированной запятой позволяет получить ощутимый выигрыш во времени. К таким задачам относятся, например, финансовые расчеты, задачи количественного учета, многие задачи управления и т.п. При решении научно-технических задач более удобно представление чисел с плавающей запятой, поскольку в них, как правило, приходится иметь дело с числовыми данными из очень широкого диапазона.

Умелое использование фиксированной запятой при решении конкретной задачи иногда позволяет добиться большей скорости и большей точности, чем использование плавающей запятой. Еще большего эффекта можно достичь путем разумного сочетания вычислений с фиксированной и плавающей запятой. Однако это тема совсем другой лекции.

Несмотря на то, что каждая из отдельных ошибок округления очень мала, в совокупности они могут оказывать значительное влияние на точность результата, получаемого в процессе реализации алгоритма. Особенно в тех случаях, когда число выполняемых операций достаточно велико. Более того, из-за ошибок округления некоторые алгоритмы просто нельзя реализовать. Поэтому при решении серьезных задач оцениванию влияния ошибок округления уделяется особое внимание.

Имеется много различных подходов к проведению такого оценивания. Гарантированные выводы о точности можно делать только на основе получения мажорантных оценок. Но мажорантные оценки редко достигаются. К тому же их получение требует виртуозного, граничащего с искусством владения соответствующей техникой. В этом можно убедиться на примере оценивания влияния ошибок округления в алгоритмах линейной алгебры [2]. Поэтому в целях создания более полной картины распределения ошибок округления весьма заманчиво считать отдельные ошибки *случайными* независимыми величинами. Заманчиво потому, что подобная гипотеза приводит к вероятностным оценкам, существенно лучшим по сравнению с мажорантными. Однако не менее заманчиво считать отдельные ошибки зависимыми случайными величинами, так как можно предположить, что знание характера зависимости также приведет к лучшим оценкам. Но тогда какими же их считать и каковы они в действительности?

В общем случае ответ на этот вопрос связан со сложными теоретико-числовыми исследованиями, знакомство с которыми не входит в нашу задачу. Ограничимся здесь лишь изложением нескольких фактов. Тем не менее, даже эти факты позволят показать интересные свойства ошибок округления и дадут веские основания для выбора правдоподобной гипотезы совместного распределения всей совокупности ошибок округления в вычислительном процессе.

Изучение вероятностных свойств ошибок округления невозможно без внесения в них некоторого элемента случайности. Эту случайность нередко связывают с многократным решением одной и той же задачи на различных компьютерах, с решением задачи при случайном числе верных знаков в промежуточных вычислениях и даже при случайном округлении результатов выполнения арифметических операций. Однако в условиях реальных вычислений внесение случайности можно осуществить, вообще говоря, единственным способом.

На всех современных компьютерах ошибка округления при выполнении любого арифметического действия однозначно определяется значениями его аргументов. Поэтому при фиксированном алгоритме и фиксированных входных данных вся совокупность ошибок округления определяется *однозначно* и никакой случайности в их поведении возникнуть просто неоткуда. Если алгоритм не связан с какими-нибудь случайными процессами, то единственным источником случайности в ошибках округления может быть лишь *случайность входных данных* алгоритма.

С точки зрения вычислителей-практиков отношение к входным данным как к случайным величинам вполне оправдано. Они редко бывают известны точно, так как на них влияет много факторов. Чаще всего входные данные алгоритма получаются в результате проведения либо каких-то вычислений, либо каких-то измерений. И то, и другое сопровождается внесением во входные данные дополнительных ошибок, точный учет которых невозможен. Следовательно, при фиксированном алгоритме и способе округления все ошибки округления можно рассматривать как вполне определенные функции случайных входных данных, распределенных совместно по некоторому закону.

Перед обсуждением вероятностных свойств ошибок округления сделаем несколько уточнений. Будем считать, что вычисления проводятся с плавающей запятой, мантиссы всех чисел имеют s разрядов и округление результата выполнения любой операции всегда осуществляется наилучшим образом. Если при выборе наилучшего округления возникает неоднозначность, то будем предполагать, что она разрешается таким образом, чтобы из двух возможностей мантисса округленного числа становилась наибольшей. Это соответствует практике выбора наилучшего округления. Вместо ошибок округления самих чисел будем рассматривать нормированные в масштабе s-го разряда ошибки округления мантисс. При сделанных уточнениях все нормированные ошибки округления будут всегда принадлежать полусегменту $(-1/2, +1/2]$.

До сих пор не создана развитая теория, позволяющая в произвольном вычислительном процессе точно изучать и оценивать ошибки округления как функции случайных входных данных. Однако доказано немало отдельных фактов, которые могут служить веским поводом для выдвижения некоторой правдоподобной вероятностной гипотезы, касающейся их совместного распределения. Общим, как в доказательствах, так и в гипотезе, является рассмотрение распределений нормированных ошибок округления в *асимптотике*, т.е. в ситуации, когда число разрядов s, отводимых для преставления мантисс чисел, может быть сколь угодно большим.

Любой вычислительный процесс начинается с округления входных данных при их вводе в компьютер. Предположим, что входные данные являются случайными величинами, имеющими непрерывную плотность совместного распределения. Доказано, что независимо от того, какова в действительности эта плотность, все нормированные ошибки округлении при вводе входных данных асимптотически являются случайными попарно независимыми величинами, распределенными равномерно и непрерывно на полусегменте (–1/2, +1/2].

Более интересные результаты дает рассмотрение простейших арифметических операций. Для умножения, деления, сложения и вычитания также доказано, что независимо от того, какова плотность совместного распределения входных данных, нормированные ошибки округления асимптотически являются случайными величинами, распределенными равномерно на полусегменте (–1/2, +1/2]. Но для умножения и деления это распределение снова оказывается *непрерывным*, а вот для сложения и вычитания оно будет *дискретным*. При этом во всех сокращенных системах счисления нормированные ошибки округления для сложения и вычитания асимптотически *не имеют никакого смещения*, а во всех системах с четным основанием ошибки асимптотически всегда имеют *ненулевое* смещение. Другими словами, математическое ожидание последних ошибок не равно нулю. Не вдаваясь в детали обсуждения, лишь заметим, что появляется этот эффект исключительно за счет того, что в системах счисления с четным основанием обязательно возникает ситуация, когда при правильном округлении существуют два наилучших округленных числа.

Напомним, что все современные компьютеры построены на системах счисления с четным основанием. Следовательно, при использовании на них вероятностной модели ошибок округления нельзя предполагать, что математическое ожидание самих ошибок в общем случае будет равно нулю. Но ведь известно, что ненулевое математическое ожидание ошибок приводит к более быстрому их накоплению! Опять констатируем, что с точки зрения точности системы счисления с четным основанием и, в частности, двоичная система не являются лучшими.

Для некоторых классов алгоритмов, например, реализующих решение задач линейной алгебры, вычисление интегралов и др., поведение ошибок округления исследовано во всей их совокупности. Доказано, что независимо от того, какова плотность совместного распределения входных данных, все нормированные ошибки округления асимптотически являются попарно независимыми случайными величинами. Отдельные ошибки ведут себя так, как описано выше.

Подчеркнем, что все результаты, касающиеся свойств ошибок округления, получены без каких-либо дополнительных предположений об их поведении. Поэтому выполненные исследования и приведенные выше аргументы говорят о том, что при вероятностной оценке суммарного влияния ошибок округления в массовых вычислениях может быть использована следующая

Гипотеза. Все нормированные ошибки округления вычислительного процесса в режиме с плавающей запятой являются случайными попарно независимыми величинами, распределение которых не зависит от входных данных и результатов промежуточных вычислений. Они распределены равномерно на полусегменте (–1/2, +1/2], дискретно для операций сложения и вычитания и непрерывно для большинства других операций. Для операций сложения и вычитания в системах счисления с четным основанием математическое ожидание ошибок не равно нулю. Для непрерывно распределенных ошибок их математическое ожидание равно 0, а дисперсия не превосходит 1/12.

При практическом применении этой гипотезы следует проявлять определенную осторожность в отношении предполагаемых значений математического ожидания и дисперсии ошибок округления, возникающих при выполнении операций сложения и вычитания. Они зависят от разности порядков участвующих в операциях чисел.

Безусловно, в общем случае ошибки округления значительно затрудняют проведение вычислений. Однако иногда умелое использование ошибок округления позволяет успешно решать очень сложные в вычислительном отношении задачи, что подтверждает следующий пример.

Предположим, что ищется вектор u, принадлежащий корневому подпространству, соответствующему собственному значению λ матрицы A. Пусть $\tilde{\lambda}$ близко к λ. Возьмем произвольный вектор u_0 и построим итерационный процесс $(A-\tilde{\lambda}E)u_{k=}\alpha_k u_{k-1}$, k=1,2… Доказано, что с увеличением k векторы u_k сходятся к вектору u, причем сходимость тем быстрее, чем ближе $\tilde{\lambda}$ к λ. Данный метод получил название метода *обратных итераций*. Но чем ближе $\tilde{\lambda}$ к λ, тем более плохо обусловленной становится матрица $A-\tilde{\lambda}E$. Поэтому из-за ошибок округления при решении систем уравнений $(A-\tilde{\lambda}E)u_{k=}\alpha_k u_{k-1}$очередной реально найденный вектор u_k будет содержать очень большие погрешности. Кажется, что по этой причине метод обратных итераций должен быть несостоятельным на практике. Однако в действительности он устроен таким образом, что чем больше погрешность в векторе u_k, тем ближе сам вектор погрешности к искомому вектору u. Другими словами, чем хуже решается система уравнений из-за влияния ошибок округления, тем лучше сходится итерационный процесс.

На этом заканчивается наше краткое знакомство с ошибками округления в вычислительных процессах. Мы надеемся, что приведенное описание основ их распределения поможет лучше ориентироваться в алгоритмах вычислительной математики, причем независимо от того, реализуются ли они на последовательных или параллельных компьютерах.