§ 2. Погрешности округления

1. Представление вещественных чисел в ЭВМ. Одним из источников вычислительных погрешностей является приближенное представление вещественных чисел в ЭВМ, обусловленное конечностью разрядной сетки. Хотя исходные данные представляются в ЭВМ с большой точностью, накопление погрешностей округления в процессе счета может привести к значительной результирующей погрешности, а некоторые алгоритмы могут оказаться и вовсе непригодными для реального счета на ЭВМ.

Напомним о способах представления чисел в ЭВМ и связанных с ними погрешностях округления. Более подробно этот круг вопросов рассматривается в [6, 8, 15, 29].

При ручном счете пользуются десятичной системой счисления. Например, запись 103,67 определяет число

$$1\cdot 10^2+0\cdot 10^1+3\cdot 10^0+6\cdot 10^{-1}+7\cdot 10^{-2}.$$

Здесь 10 — основание системы счисления, запятая отделяет дробную часть числа от целой, 1, 0, 3, 6, 7 — числа из базисного набора {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, с помощью которого можно представить любое вещественное число.

ЭВМ работают, как правило, в двоичной системе, когда любое число записывается в виде последовательности нулей и единиц. Например, запись 0, 0101 в двоичной системе определяет число

$$0\cdot 2^0+0\cdot 2^{-1}+1\cdot 2^{-2}+0\cdot 2^{-3}+1\cdot 2^{-4}.$$

Как двоичная, так и десятичная системы относятся к позиционным системам счисления. В *позиционной системе с основанием* r запись

$$a=\pm a_n a_{n-1}\ldots a_0,\ a_{-1}a_{-2}\ldots \tag{1}$$

означает, что

$$a=\pm\left(a_n r^n+a_{n-1}r^{n-1}+\ldots+a_0 r^0+a_{-1}r^{-1}+a_{-2}r^{-2}+\ldots\right).$$

Будем считать далее, что r — целое число, большее единицы. Каждое из чисел a_i может принимать одно из значений $\{0, 1, \ldots, r-1\}$. Числа a_i называются *разрядами*, например: a_3 — третий разряд перед запятой, a_{-2} — второй разряд после запятой.

Запись вещественного числа в виде (1) называется также его представлением в форме *числа с фиксированной запятой*. В ЭВМ чаще всего используется представление чисел в форме *с плавающей запятой*, т. е. в виде

$$a=Mr^p, \tag{2}$$

где r — основание системы счисления, p — целое число (положительное, отрицательное или нуль) и

$$r^{-1}\leqslant|M|<1. \tag{3}$$

Число M представляется в форме числа с фиксированной запятой и называется *мантиссой числа* a. Число p называется *порядком числа* a. В виде (2) можно единственным образом представить

любое вещественное число кроме нуля. Единственность обеспечивается условием нормировки (3).

Например, число 103,67 в форме с плавающей запятой имеет вид $0{,}10367 \cdot 10^3$, т. е. $M=0{,}10367$, $p=3$. Двоичное число $0{,}0101 = 0{,}101 \cdot 2^{-1}$ имеет в двоичной системе мантиссу $M=0{,}101$ и порядок $p=-1$.

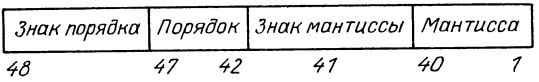

Рис. 2. Разрядная сетка

В ЭВМ для записи каждого числа отводится фиксированное число разрядов (*разрядная сетка*). Например, в ЭВМ БЭСМ-6 для записи числа, представленного в форме с плавающей запятой, отводится 48 двоичных разрядов, которые распределяются следующим образом: в разрядах с 1 по 40 помещается абсолютное значение мантиссы, в 41 разряде — знак мантиссы, в разрядах от 42 до 47 — абсолютная величина порядка, в 48 разряде — знак порядка (см. рис. 2). Отсюда легко найти диапазон чисел, представимых в ЭВМ БЭСМ-6. Поскольку максимальное значение порядка в двоичной системе равно 111 111=63 и мантисса не превосходит единицы, то с помощью указанной разрядной сетки можно представить числа, абсолютная величина которых лежит примерно в диапазоне от 2^{-63} до 2^{63}, т. е. от 10^{-19} до 10^{19}.

Ту же 48-разрядную сетку можно использовать для представления чисел с фиксированной запятой. Пусть, например, разряды с 1 по 24 отводятся для записи дробной части числа и разряды с 25 по 47 — для записи целой части числа. Тогда максимальное число, которое можно представить с помощью данной разрядной сетки, будет равно

$$\underbrace{11\ldots1}_{\text{23 разряда}},\ \underbrace{11\ldots1}_{\text{24 разряда}} < 2^{23} \approx 10^7.$$

Следовательно, в данном случае диапазон допустимых чисел в 10^{12} раз меньше, чем при использовании представления с плавающей запятой. Возможностью существенного увеличения диапазона допустимых чисел при той же разрядной сетке и объясняется преимущественное использование в ЭВМ представления чисел в форме с плавающей запятой. Комплексное число представляется в ЭВМ в виде пары вещественных чисел.

2. Округление чисел в ЭВМ. Будем считать в дальнейшем, что вещественные числа представляются в ЭВМ в форме с плавающей запятой. Минимальное положительное число M_0, которое может быть представлено в ЭВМ с плавающей запятой, называется *машинным нулем*. Мы видим, что для ЭВМ БЭСМ-6 число $M_0 \approx 10^{-19}$. Число $M_\infty = M_0^{-1}$ называется *машинной бесконечностью*. Все вещественные числа, которые могут быть представлены в данной ЭВМ, расположены по абсолютной величине в диапазоне от M_0 до M_∞. Если в процессе счета какой-либо задачи появится вещест-

венное число, меньшее по модулю чем M_0, то ему присваивается нулевое значение. Так, на ЭВМ БЭСМ-6 в результате перемножения двух чисел 10^{-11} и 10^{-10} получим нуль. При появлении в процессе счета вещественного числа, большего по модулю чем M_∞, происходит так называемое переполнение разрядной сетки, после чего ЭВМ прекращает счет задачи. Отметим, что нуль и целые числа представляются в ЭВМ особым образом, так что они могут выходить за пределы диапазона $M_0 \div M_\infty$.

Из-за конечности разрядной сетки в ЭВМ можно представить точно не все числа из диапазона $M_0 \div M_\infty$, а лишь конечное множество чисел. Число a, не представимое в ЭВМ точно, подвергается *округлению*, т. е. оно заменяется близким ему числом $\tilde{a}$, представимым в ЭВМ точно. Точность представления чисел в ЭВМ с плавающей запятой характеризуется *относительной погрешностью*

$$|a-\tilde{a}|/|a|.$$

Величина относительной погрешности зависит от способа округления. Простейшим, но не самым точным способом округления является отбрасывание всех разрядов мантиссы числа a, которые выходят за пределы разрядной сетки.

Найдем границу относительной погрешности при таком способе округления. Пусть для записи мантиссы в ЭВМ отводится t двоичных разрядов. Предположим, что надо записать число, представленное в виде бесконечной двоичной дроби

$$a = \pm 2^p \left(\frac{a_1}{2} + \frac{a_2}{2^2} + \ldots + \frac{a_t}{2^t} + \frac{a_{t+1}}{2^{t+1}} + \ldots \right), \tag{4}$$

где каждое из a_j равно 0 или 1. Отбрасывая все лишние разряды, получим округленное число

$$\tilde{a} = \pm 2^p \left(\frac{a_1}{2} + \frac{a_2}{2^2} + \ldots + \frac{a_t}{2^t} \right).$$

Таким образом, для погрешности округления

$$a - \tilde{a} = \pm 2^p \left(\frac{a_{t+1}}{2^{t+1}} + \frac{a_{t+2}}{2^{t+2}} + \ldots \right)$$

справедлива оценка

$$|a-\tilde{a}| \leqslant 2^p \frac{1}{2^{t+1}} \left(1 + \frac{1}{2} + \frac{1}{2^2} + \ldots \right) = 2^{p-t}.$$

Далее заметим, что из условия нормировки $|M| \geqslant 0{,}5$ (см. (3)) следует, что в разложении (4) всегда $a_1 = 1$. Поэтому $|a| \geqslant 2^p \cdot 2^{-1} = 2^{p-1}$, и для относительной погрешности округления получим оценку

$$\frac{|a-\tilde{a}|}{|a|} \leqslant 2^{-t+1}.$$

При более точных способах округления можно уменьшить погрешность по крайней мере в два раза и добиться, чтобы выполнялась оценка

$$\frac{|a-\tilde{a}|}{|a|} \leqslant 2^{-t}. \tag{5}$$

Итак, *относительная точность в ЭВМ с плавающей запятой определяется числом разрядов t, отводимых для записи мантиссы. Можно считать, что точное число a и отвечающее ему округленное число $\tilde{a}$ связаны равенством*

$$\tilde{a}=a(1+\varepsilon), \tag{6}$$

где $|\varepsilon|\leqslant 2^{-t}$. Число 2^{-t} называют иногда *машинным эпсилоном.* Оно характеризует относительную точность представления чисел в ЭВМ. Для ЭВМ БЭСМ-6 имеем $t=40$, $2^{-t}\approx 10^{-12}$, т. е. относительная точность представления чисел составляет 12 десятичных знаков.

Соотношение (6) справедливо лишь в случае $|a|\geqslant M_0$, где M_0 — машинный нуль. Если же число a мало, а именно $|a|<M_0$, то полагается $\tilde{a}=0$, что соответствует $\varepsilon=-1$ в формуле (6).

3. Накопление погрешностей округления. В процессе проведения вычислений погрешности округления могут накапливаться, так как выполнение каждой из четырех арифметических операций вносит некоторую погрешность.

Будем в дальнейшем обозначать округленное в системе с плавающей запятой число, соответствующее точному числу x, через $\mathrm{fl}(x)$ (от английского floating — плавающий). Считается, что выполнение каждой арифметической операции вносит относительную погрешность не большую, чем 2^{-t}. Это предположение можно записать в виде

$$\mathrm{fl}(a*b)=a*b(1+\varepsilon), \tag{7}$$

где звездочка означает любую из операций $+$, $-$, $\times$, $:$, и $|\varepsilon|\leqslant 2^{-t}$. Если результат выполнения арифметической операции является машинным нулем, то в формуле (7) надо положить $\varepsilon=-1$.

Может показаться, что предположение (7) не обосновано, так как согласно (6) каждое из чисел a и b записывается с относительной погрешностью 2^{-t}, следовательно, погрешность результата может достигнуть 2^{-t+1}. Однако ЭВМ обладает возможностью проводить промежуточные вычисления с двойной точностью, т. е. с мантиссой, содержащей $2t$ разрядов, причем округлению до t разрядов подвергается лишь окончательный результат. Это обстоятельство позволяет добиться выполнения соотношения (7).

Для оценки влияния погрешностей округления на результат того или иного вычислительного алгоритма очень часто используется предположение о том, что *результат вычислений, искаженный погрешностями округления, совпадает с результатом точного выполнения этого же алгоритма, но с иными входными данными.*

Рассмотрим, например, процесс вычисления суммы

$$z=y_1+y_2+y_3 \tag{8}$$

трех положительных чисел. Пусть сначала находится сумма y_1+y_2 Тогда согласно (7) получим

$$z_1=\mathrm{fl}(y_1+y_2)=(y_1+y_2)(1+\varepsilon_1), \qquad |\varepsilon_1|\leqslant 2^{-t}.$$

Затем в результате сложения z_1 и y_3 получим число

$$\tilde{z}=\mathrm{fl}(z_1+y_3)=(z_1+y_3)(1+\varepsilon_2),$$

где $|\varepsilon_2| \leqslant 2^{-t}$. Таким образом, вместо точного значения суммы z получаем приближенное значение

$$\tilde{z}=(y_1+y_2)(1+\varepsilon_1)(1+\varepsilon_2)+y_3(1+\varepsilon_2).$$

Отсюда видно, что результат выполнения алгоритма (8), искаженный погрешностями округления, совпадает с результатом точного выполнения того же алгоритма (8), примененного к другим исходным данным

$$\tilde{y}_i=(1+\varepsilon_1)(1+\varepsilon_2)y_i, \quad i=1, 2, \qquad \tilde{y}_3=(1+\varepsilon_2)y_3.$$

На этом же примере видно, что результирующая погрешность зависит от порядка выполнения операций, так что вычисление суммы (8) в обратном порядке $(y_3+y_2)+y_1$ может привести к другому результату.

Приведенный пример имеет чисто иллюстративное значение, так как число слагаемых в сумме (8) невелико, а погрешности ε_i малы. Практический интерес представляют оценки результирующей погрешности в зависимости от числа выполненных арифметических действий n. Однако прежде чем перейти к получению таких оценок, необходимо познакомиться с методами решения разностных уравнений.

4. Разностные уравнения первого порядка. Предположим, что надо вычислить сумму

$$z_n=\sum_{j=1}^{n} y_j. \tag{9}$$

Тогда вычисления организуются обычно следующим образом. Задается начальное значение $z_0=0$ и затем последовательно, начиная с $j=1$, находятся числа z_j, связанные рекуррентным соотношением

$$z_j=z_{j-1}+y_j, \quad j=1, 2, \ldots, n, \quad z_0=0. \tag{10}$$

Для вычисления произведения

$$z_n=\prod_{j=1}^{n} y_j \tag{11}$$

достаточно задать начальное значение $z_0=1$ и воспользоваться рекуррентными соотношениями

$$z_j=y_j z_{j-1}, \quad j=1, 2, \ldots, n, \quad z_0=1. \tag{12}$$

Уравнения (10) и (11) являются частными случаями *линейного разностного уравнения первого порядка*

$$z_j=q_j z_{j-1}+\varphi_j, \quad j=1, 2, \ldots, n, \tag{13}$$

где q_j, φ_j — заданные числа, z_j — искомые числа. Для уравнения (13) рассматривается задача с начальными условиями или *задача Коши*, которая состоит в отыскании всех z_j, $j=1, 2, \ldots, n$, при за-

данном начальном значении z_0. Ясно, что решение задачи Коши для разностного уравнения (13) существует и единственно.

Коэффициенты q_j, правые части φ_j и искомое решение z_j уравнения (13) можно рассматривать как функции целочисленного аргумента j, т. е. $q_j=q(j)$, $\varphi_j=\varphi(j)$, $z_j=z(j)$.

Нам потребуется прежде всего записать решение уравнения (13) в явном виде. Подставляя в (13) вместо z_{j-1} выражение

$$z_{j-1}=q_{j-1}z_{j-2}+\varphi_{j-1},$$

получим

$$z_j=q_jq_{j-1}z_{j-2}+\varphi_j+q_j\varphi_{j-1}.$$

Теперь можно подставить сюда выражение для z_{j-2}, затем — для z_{j-3} и т. д. В результате получим формулу, в которой z_j выражается через z_{j-l}, φ_{j-l+1}, φ_{j-l+2}, ..., φ_j. Эта формула имеет вид

$$z_j=Q_{jl}z_{j-l}+\sum_{k=j-l+1}^{j}Q_{j,j-k}\varphi_k,\quad l=1,2,\ldots,j-1,j,\quad j=1,2,\ldots,n, \tag{14}$$

где

$$Q_{jl}=\begin{cases}1, & l=0,\\ q_jq_{j-1}\ldots q_{j-l+1}, & 1\leqslant l\leqslant j.\end{cases} \tag{15}$$

Строго доказать формулу (14) можно индукцией по числу j при каждом фиксированном l. Нам потребуется формула (14) при $l=j$, т. е.

$$z_j=Q_{jj}z_0+\sum_{k=1}^{j}Q_{j,j-k}\varphi_k,\quad j=1,2,\ldots,n, \tag{16}$$

где согласно (15)

$$Q_{l,j-k}=\begin{cases}1, & k=j,\\ q_jq_{j-1}\ldots q_{k+1}, & 0\leqslant k\leqslant j-1.\end{cases} \tag{17}$$

В частности, если (13) является уравнением с постоянными коэффициентами, т. е. $q_j=q$ для всех j, то из (16) получим

$$z_j=q^jz_0+\sum_{k=1}^{j}q^{j-k}\varphi_k,\quad j=1,2,\ldots,n. \tag{18}$$

Явную формулу (16) можно использовать для получения различных оценок решения z_j через начальные данные z_0, заданные коэффициенты q_j и правые части φ_j.

Лемма 1. *Если для некоторого $q\geqslant 0$ выполнены неравенства*

$$|q_j|\leqslant q,\quad j=1,2,\ldots,n, \tag{19}$$

то для решения уравнения (13) *справедливы оценки*

$$|z_j|\leqslant q^j|z_0|+\sum_{k=1}^{j}q^{j-k}|\varphi_k|,\quad j=1,2,\ldots,n. \tag{20}$$

Доказательство. Из (17) и (19) получаем, что

$$|Q_{j,j-k}| \leqslant q^{j-k}, \quad k=0, 1, \ldots, j.$$

Отсюда и из (16) следуют оценки (20).

Замечание. Оценки (20) неулучшаемы в том смысле, что для уравнения (13) с постоянными коэффициентами и положительными z_0, φ_k, $k=1, 2, \ldots, J$, неравенства (20) выполняются согласно (18) со знаком равенства.

5. Оценки погрешностей округления. Приведем примеры оценок погрешностей округления, возникающих в результате выполнения вычислительных алгоритмов. Нас будет интересовать в основном зависимость результирующей погрешности от числа арифметических действий n и от величины $\varepsilon=2^{-t}$, определяемой разрядностью ЭВМ.

Пример 1. Вычисление произведения

$$z_n = \prod_{j=1}^{n} y_j$$

n вещественных чисел проводится по формуле

$$z_j=y_j z_{j-1}, \quad j=1, 2, \ldots, n, \quad z_0=1. \tag{21}$$

Предположим, что в результате округления вместо точного значения z_{j-1} получено приближенное значение $\tilde{z}_{j-1}$. Тогда согласно (7) вместо $y_j\tilde{z}_{j-1}$ получим величину

$$\mathrm{fl}(y_j\tilde{z}_{j-1})=y_j\tilde{z}_{j-1}(1+\varepsilon_j),$$

где $|\varepsilon_j| \leqslant \varepsilon=2^{-t}$. Таким образом, вместо z_j получаем

$$\tilde{z}_j=(1+\varepsilon_j)y_j\tilde{z}_{j-1},$$

т. е. приближенное значение $\tilde{z}_j$ удовлетворяет рекуррентному соотношению

$$\tilde{z}_j=\tilde{y}_j\tilde{z}_{j-1}, \quad j=1, 2, \ldots, n, \quad \tilde{z}_0=1, \tag{22}$$

где $\tilde{y}_j=y_j(1+\varepsilon_j)$. Результирующая погрешность равна

$$z_n-\tilde{z}_n = \prod_{j=1}^{n} y_j - \prod_{j=1}^{n} (1+\varepsilon_j)\, y_j,$$

поэтому относительная погрешность есть

$$\frac{z_n-\tilde{z}_n}{z_n} = 1-\prod_{j=1}^{n}(1+\varepsilon_j).$$

Для оценки относительной погрешности заметим, что

$$|1+\varepsilon_j| \leqslant 1+\varepsilon, \quad j=1, 2, \ldots, n, \quad \varepsilon=2^{-t},$$

поэтому с точностью до величин второго порядка малости относительно ε можно считать, что

$$\left|\frac{z_n-\tilde{z}_n}{z_n}\right| \leqslant n\varepsilon = n2^{-t}. \tag{23}$$

При выводе оценки (23) предполагалось, что $\varepsilon=2^{-t}$, т. е. при перемножении не возникает чисел, меньших машинного нуля или больших машинной бесконечности. Однако может оказаться, что на каком-то этапе вычислений в качестве промежуточного результата будет получен либо машинный нуль M_0, либо машинная бесконечность M_∞. Поскольку оба указанных случая приводят к неверному окончательному результату, необходимо видоизменить вычислительный алгоритм. Оказывается, что здесь существенным является порядок действий.

Пусть, например, $M_0=2^{-p}$ и $M_\infty=2^p$ при некотором $p>0$. Предположим, что надо перемножить пять чисел $y_1=2^{p/2}$, $y_2=2^{p/4}$, $y_3=2^{3p/4}$, $y_4=2^{-p/2}$, $y_5=2^{-3p/4}$. Каждое из этих чисел и их произведение $2^{p/4}$ принадлежат допустимому диапазону чисел (M_0, M_∞). Однако произведение $y_1y_2y_3=2^{3p/2}>M_\infty$, поэтому при указанном порядке действий дальнейшее выполнение алгоритма становится невозможным. Если проводить вычисление в порядке $y_5y_4y_3y_2y_1$, то получим $y_5y_4=2^{-5p/4}<M_0$, следовательно, $\mathrm{fl}(y_5y_4)=0$ и все произведение окажется равным нулю, т. е. получим неверный результат. В данном примере к верному результату приводит вычисление произведения в порядке

$$y_5y_3y_1y_4y_2.$$

В случае произвольного числа n сомножителей можно предложить следующий алгоритм вычисления произведения (см. [6]). Предположим, что

$$|y_1|\leqslant|y_2|\leqslant\ldots\leqslant|y_n|,$$

причем $|y_1|\leqslant 1$, $|y_n|\geqslant 1$. Будем сначала проводить умножение в порядке $y_1y_ny_{n-1}\ldots$ до тех пор, пока впервые не получим число, большее единицы. Затем полученное частичное произведение будем последовательно умножать на y_2, y_3 и т. д. до тех пор, пока новое частичное произведение не станет меньше единицы. Процесс повторяется до тех пор, пока все оставшиеся сомножители будут либо только большими единицы по модулю, либо только меньшими. Далее умножение проводится в произвольном порядке.

Пример 2. Рассмотрим процесс вычисления суммы

$$z_n=y_1+y_2+\ldots+y_n. \tag{24}$$

Для простоты изложения предположим, что все y_i положительны и больше машинного нуля. Тогда в процессе вычислений не может появиться нулевого результата. Алгоритм вычисления суммы (24) состоит в решении разностного уравнения (10) при начальном значении $z_0=0$.

Получим уравнение, которому удовлетворяет приближенное решение $\tilde{z}_j$. Предположим, что вместо точного значения z_{j-1} в результате накопления погрешностей округления получено приближенное значение $\tilde{z}_{j-1}$. Тогда согласно (7) вместо z_j получим число

$$\tilde{z}_j=\mathrm{fl}(\tilde{z}_{j-1}+y_j)=(1+\varepsilon_j)(\tilde{z}_{j-1}+y_j),$$

где $|\varepsilon_j|\leqslant 2^{-t}$.

Таким образом, приближенное значение $\tilde{z}_j$ удовлетворяет разностному уравнению

$$\tilde{z}_j=q_j\tilde{z}_{j-1}+\tilde{y}_j,\quad j=1, 2, \ldots, n,\quad \tilde{z}_0=0, \tag{25}$$

где $q_j=1+\varepsilon_j$, $\tilde{y}_j=(1+\varepsilon_j)y_j$. Можно считать, что уравнение (25) получено из исходного уравнения (10) путем внесения возмущений в коэффициенты и в правые части, причем для каждого j возмущение пропорционально ε_j и не превосходит 2^{-t}.

Оценим теперь результирующую погрешность $\tilde{z}_n-z_n$. Для этого выпишем в явном виде решения уравнений (10) и (25), предполагая, что $z_0=\tilde{z}_0=0$. Согласно (16), (18) имеем

$$z_n=\sum_{k=1}^{n} y_k, \quad \tilde{z}_n=\sum_{k=1}^{n} Q_{n,n-k}\tilde{y}_k,$$

где $\tilde{y}_k=q_k y_k$. Поэтому для погрешности получим следующее выражение:

$$\tilde{z}_n-z_n=\sum_{k=1}^{n} E_{nk}y_k, \tag{26}$$

где

$$E_{nk}=q_kQ_{n,n-k}-1=\begin{cases} q_n-1, & k=n, \\ q_nq_{n-1}\ldots q_{k+1}q_k-1, & k=1, 2, \ldots, n-1. \end{cases} \tag{27}$$

Коэффициент E_{nk} в формуле (26) указывает, какую долю погрешности вносит k-е слагаемое суммы (24) в общую погрешность. Покажем, что чем меньше номер k, тем большая погрешность вносится за счет y_k. Для этого оценим приближенно величины E_{nk}. Так как $q_j=1+\varepsilon_j$ и $|\varepsilon_j|<\varepsilon=2^{-t}$, то $|q_n|\leqslant 1+\varepsilon$, $|q_nq_{n-1}\ldots q_{k+1}q_k|\leqslant (1+\varepsilon)^{n-k+1}$. Отбрасывая величины второго порядка малости относительно ε, можно считать, что

$$|q_nq_{n-1}\ldots q_k|\leqslant 1+(n-k+1)\varepsilon,$$

и тогда

$$|E_{nk}|\leqslant (n-k+1)\varepsilon, \quad k=1, 2, \ldots, n. \tag{28}$$

Из формулы (26) легко получить оценку относительной погрешности $|\tilde{z}_n-z_n|/|z_n|$. Заметим сначала, что для положительных $y_1, \ldots, y_n$ последовательность z_j, определенная согласно (10), неотрицательная и монотонно возрастающая, т. е.

$$0\leqslant z_{k-1}\leqslant z_k, \quad k=1, 2, \ldots, n.$$

Поэтому для $y_k=z_k-z_{k-1}$ справедливо неравенство

$$0\leqslant y_k\leqslant |z_k|+|z_{k-1}|\leqslant 2|z_n|, \quad k=1, 2, \ldots, n.$$

Отсюда и из (26) получим оценку

$$|\tilde{z}_n-z_n|\leqslant 2|z_n|\sum_{k=1}^{n}|E_{nk}|.$$

Учитывая приближенное неравенство (28), приходим к следующей

оценке относительной погрешности:

$$\left|\frac{\tilde{z}_n - z_n}{z_n}\right| \leqslant \varepsilon n(n+1), \qquad \varepsilon = 2^{-t}.$$

Следовательно, относительная погрешность, возникающая при суммировании n положительных чисел, оценивается примерно как $n^2 2^{-t}$, где t — число разрядов, отводимое для записи мантиссы. Например, при $2^{-t} = 10^{-12}$, $n = 10^3$ получаем, что результирующая относительная погрешность не превзойдет 10^{-6}.

§ 3. Разностные уравнения второго порядка

1. Задача Коши и краевые задачи для разностных уравнений. В п. 4 § 2 рассматривалась задача Коши для разностного уравнения первого порядка. Обратимся теперь к линейным разностным уравнениям второго порядка

$$a_j y_{j-1} - c_j y_j + b_j y_{j+1} = -f_j, \tag{1}$$

где a_j, b_j, c_j, f_j — заданные коэффициенты и правая часть и y_j — искомое решение. Индекс j в уравнении (1) пробегает некоторое допустимое множество J целых чисел. Например,

$$J = \{0, 1, 2, \ldots\}, \quad J = \{1, 2, \ldots, N-1\}, \quad J = \{0, \pm 1, \pm 2, \ldots\},$$

где $N > 1$ — заданное целое число. Всюду в дальнейшем будем предполагать, что $b_j \neq 0$, $a_j \neq 0$ для всех допустимых j.

Коэффициенты, правую часть и решение уравнения (1) следует рассматривать как функции целочисленного аргумента $j \in J$, т. е. $y_j = y(j)$, $f_j = f(j)$ и т. д.

Уравнение (1) имеет бесконечное множество решений. Каждое отдельное решение называется *частным решением* уравнения (1). *Общим решением* уравнения (1) называется такое двухпараметрическое семейство решений, которое содержит любое частное решение. В пп. 3, 4 будет показано, каким образом строится общее решение уравнения (1).

Для того чтобы из совокупности всех решений уравнения (1) выделить единственное, необходимо задать те или иные дополнительные условия.

Задача Коши состоит в отыскании решения y_j, $j = 0, 1, 2, \ldots$, уравнения (1), удовлетворяющего при $j = 0, 1$ заданным начальным условиям

$$y_0 = \mu_1, \qquad y_1 = \mu_2. \tag{2}$$

Если $b_j \neq 0$ для всех допустимых j, то уравнение (1) можно разрешить относительно y_{j+1}, т. е. записать в виде

$$y_{j+1} = -\frac{a_j}{b_j} y_{j-1} + \frac{c_j}{b_j} y_j - \frac{f_j}{b_j}. \tag{3}$$

Отсюда следует, что задача Коши имеет единственное решение.