4.4. METODA PROGRAMOWANEGO PRZECINKA

Dotychczas omówiliśmy dwie metody obliczeniowe: stało- i zmiennoprzecinkową. Jak już podkreśliliśmy, zasadniczą wadą pierwszej z tych metod jest mała dokładność obliczeń, wadą drugiej zaś jest mała prędkość liczenia. Obecnie omówimy jeszcze jedną metodę zwaną metodą programowanego przecinka lub metodą zmiennych skali. Idea tej metody jest następująca: Rozbijamy obliczenia na takie etapy, aby przy wykonywaniu każdego z tych etapów można było prowadzić obliczenia ze stałym przecinkiem, odpowiednie przenormowania liczb następuje przy przejściu z jednego etapu do drugiego. Oczywiście, jeśli całe obliczenia będziemy traktowali jako jeden etap, to będziemy po prostu rozwiązywali zagadnienie w stałym przecinku, jeśli zaś będziemy każde działanie arytmetyczne traktowali jako odrębny etap obliczeniowy, będziemy prowadzili obliczenia ze zmiennym przecinkiem. Podobnie jak przy obliczeniach stałoprzecinkowych, wprowadzimy obecnie mnożniki skalujące, jeśli mamy jakąś wielkość y_i, co do modułu większą lub równą jedności, będziemy w obliczeniach zastępowali ją wielkością $Y_i = \mu_i y_i$ gdzie $|Y_i| < 1$, zaś $0 < \mu_i < 1$. Podobnie dla wielkości x_i bliskich zeru wprowadzimy wielkość $X_i = \frac{x_i}{\nu_i}$, gdzie $0 < r \leqslant |X_i| < 1$, zaś $0 < \nu_i < 1$, r jest wyznaczone przez założoną dokładność. Wielkości μ_i, ν_i będziemy nazywali mnożnikami skalującymi.

W odróżnieniu od stałego przecinka, gdzie z góry ustaliliśmy wartość mnożników skalujących, przy obliczeniach z programowanym przecinkiem ustalamy tylko kryteria wyboru przez program tych mnożników. Przedstawiona dalej metoda programowanego przecinka jest stosowana w Centrum Obliczeniowym Uniwersytetu Moskiewskiego (Bibliografia [18]).

Wielkości występujące w obliczeniach przedstawimy w postaci:

$$y_i = 2^{p_i} Y_i, \tag{4-11}$$

gdzie p_i jest liczbą całkowitą, Y_i zaś spełnia nierówność:

$$|Y_i| < 1. \tag{4-12}$$

Mnożnikami skalującymi w tym przypadku są wielkości 2^{-p_i}.

W odróżnieniu od omawianego w punkcie 4.3 zmiennego przecinka wielkości y_i będą przechowywane w dwóch komórkach pamięci: Y_i w komórce długiej i p_i w komórce krótkiej. Wielkości Y_i będziemy nazywali mantysą liczby y_i, wielkość p_i zaś skalą liczby y_i. Proces rozwiązywania przez maszynę zagadnienia rozbijamy na takie etapy, aby skale przy realizacji każdego z etapów były ustalone. Przy przejściu od jednego etapu obliczeń do następnego następuje przeskalowanie wielkości. Czynność tę wykonuje specjalna część programu. Warunki określające prawidłowy wybór skali określamy następująco.

Wielkości Y_i po przeskalowaniu spełniają nierówność

$$0 < r \leqslant |Y_i| < R \leqslant 1, \tag{4-13}$$

przy czym wielkość R dobiera się tak, aby w czasie wykonywania kolejnego etapu obliczeń nie nastąpiło przekroczenie zakresu, wielkość r zaś określa się na podstawie założeń dotyczących dokładności obliczeń.

Podobnie jak dla działań zmiennoprzecinkowych (punkt 4.3) działania arytmetyczne na liczbach postaci $x = 2^p X$, $y = 2^q Y$ określone są wzorami

$$xy = 2^{p+q}(XY), \tag{4-14}$$

$$\frac{x}{y} = 2^{p-q}\left(\frac{X}{Y}\right), \tag{4-15}$$

$$x \pm y = 2^p(X \pm 2^{-(p-q)}\, Y), \quad \text{jeśli} \quad p \geqslant q, \tag{4-16}$$

$$x \pm y = 2^q(2^{-(p-q)}\, X \pm Y), \quad \text{jeśli} \quad p < q. \tag{4-17}$$

Przy dzieleniu należy pamiętać, że skale p i q należy dobrać tak, aby $\left|\frac{X}{Y}\right| < 1$, podobnie przy dodawaniu i odejmowaniu należy tak dobrać skale, aby przy wykonaniu działań w nawiasach nie nastąpiło przekroczenie zakresu.

Należy podkreślić, że nie wszystkie liczby występujące w obliczeniach przedstawiamy w postaci par: *mantysa—skala*; mianowicie wszystkie stałe występujące w obliczeniach, podobnie jak przy obliczeniach stałoprzecinkowych, mnożymy przez tak dobrane mnożniki, aby liczby te były co do modułu mniejsze od jedności, np. $\frac{e}{4}$, $\frac{\pi}{4}$, $\ln 2$, $\frac{2}{\pi}$ itp. Odwrotności mnożników, które użyliśmy do przenormowania stałych, uwzględniamy przekształcając wielkości zmienne, które w trakcie obliczeń są mnożone przez wyżej wymienione stałe.

Szczególnie wygodnie jest stosować metodę programowanego przecinka przy całkowaniu równań różniczkowych zwyczajnych oraz przy rozwiązywaniu równań przestępnych. Trudniej jest stosować tę metodę przy rozwiązywaniu układów równań algebraicznych liniowych. W zasadzie można jednak wszelkie problemy obliczeniowe rozwiązywać za pomocą programowanego przecinka, nie zawsze jednak jest to opłacalne, ze względu na ilość pracy, jaką trzeba włożyć dla odpowiedniego przekształcenia algorytmu.

Jako przykład rozpatrzymy całkowanie równań różniczkowych zwyczajnych metodą Rungego—Kutty—Gilla (Bibliografia [19]). Program dla tej metody z programowanym przecinkiem składa się z czterech części:

1) sterowanie obliczeniami i kontrola prawidłowości obliczeń,
2) obliczenie prawych stron równań,
3) wykonanie jednego kroku całkowania,
4) kontrola i zmiana skali po wykonaniu każdego kroku obliczeniowego.

Dany jest układ równań

$$\left.\begin{aligned} y_1' &= f_1(y_1, y_2, \dots, y_n), \\ y_2' &= f_2(y_1, y_2, \dots, y_n), \\ &\dots\dots\dots\dots\dots \\ y_n' &= f_n(y_1, y_2, \dots, y_n), \end{aligned}\right\} \tag{4-18}$$

W przedstawionym wyżej układzie zmienna niezależna nie występuje w postaci jawnej. W przypadku układu, w którym zmienna niezależna występuje w postaci jawnej, przez dołączenie jednego równania, w którym pochodna zmiennej niezależnej jest równa jedności, przechodzimy do układu podanej postaci.

Wzory na numeryczne całkowanie metodą Rungego-Kutty-Gilla dla kroku o długości h mają postać:

$$\left.\begin{aligned} k_{i0} &= 2^m h f_i(y_{00}, y_{10}, \dots, y_{n0}),\\ r_{i1} &= \tfrac{1}{2}(k_{i0}-q_{i0}),\\ y_{i1} &= y_{i0}+2^{-m}r_{i1},\\ q_{i1} &= q_{i0}+3\cdot 2^m(2^{-m}r_{i1})-\tfrac{1}{2}k_{i0}, \end{aligned}\right\}\quad \text{dla } i=1,2,\dots,n, \tag{4-19a}$$

$$\left.\begin{aligned} k_{i1} &= 2^m h f_i(y_{01}, y_{11}, \dots, y_{n1}),\\ r_{i2} &= (1-\sqrt{\tfrac{1}{2}})(k_{i1}-q_{i1}),\\ y_{i2} &= y_{i1}+2^{-m}r_{i2},\\ q_{i2} &= q_{i1}+3\cdot 2^m(2^{-m}r_{i2})-(1-\sqrt{\tfrac{1}{2}})k_{i1}, \end{aligned}\right\}\quad \text{dla } i=1,2,\dots,n, \tag{4-19b}$$

$$\left.\begin{aligned} k_{i2} &= 2^m h f_i(y_{02}, y_{i2}, \dots, y_{n2}),\\ r_{i3} &= (1+\sqrt{\tfrac{1}{2}})(k_{i2}-q_{i2}),\\ y_{i3} &= y_{i2}+2^{-m}r_{i3},\\ q_{i3} &= q_{i2}+3\cdot 2^m(2^{-m}r_{i3})-(1+\sqrt{\tfrac{1}{2}})k_{i2}, \end{aligned}\right\}\quad \text{dla } i=1,2,\dots,n, \tag{4-19c}$$

$$\left.\begin{aligned} k_{i3} &= 2^m h f_i(y_{03}, y_{13}, \dots, y_{n3}),\\ r_{i4} &= \tfrac{1}{6}(k_{i3}-2q_{i3}),\\ y_{i4} &= y_{i3}+2^{-m}r_{i4},\\ q_{i4} &= q_{i3}+3\cdot 2^m(2^{-m}r_{i4})-\tfrac{1}{2}k_{i3}, \end{aligned}\right\}\quad \text{dla } i=1,2,\dots,n, \tag{4-19d}$$

Na początku obliczeń y_{i0} przyjmujemy równe warunkom początkowym, q_{i0} zaś kładziemy równe zeru. Przy przejściu od s-tego kroku całkowania do $s+1$-ego kładziemy y_{i0} i q_{i0} kroku $s+1$ równe y_{i4} i q_{i4} kroku s-tego. Wielkość m dobieramy tak, aby przy obliczeniu $2^m hf_i(y_1, \dots, y_n)$ dla $i = 1,2,\dots,n$ i wszelkich dopuszczalnych wartości $y_1, y_2, \dots, y_n$ nie nastąpiło przekroczenie zakresu. Mnożnik 2^m wprowadzimy dla zmniejszenia błędów zaokrągleń.

Obecnie nadamy wzorom (4-19) postać dogodną do obliczeń z programowanym przecinkiem.

Dokonamy podstawień

$$\left.\begin{aligned} y_{ij} &= 2^{p_i}Y_{ij},\\ r_{ij} &= 2^{p_i}R_{ij},\\ k_{ij-1} &= 2^{p_i}K_{ij-1},\\ q_{ij} &= 2^{p_i}Q_{ij}, \end{aligned}\right\} \tag{4-20}$$

oraz

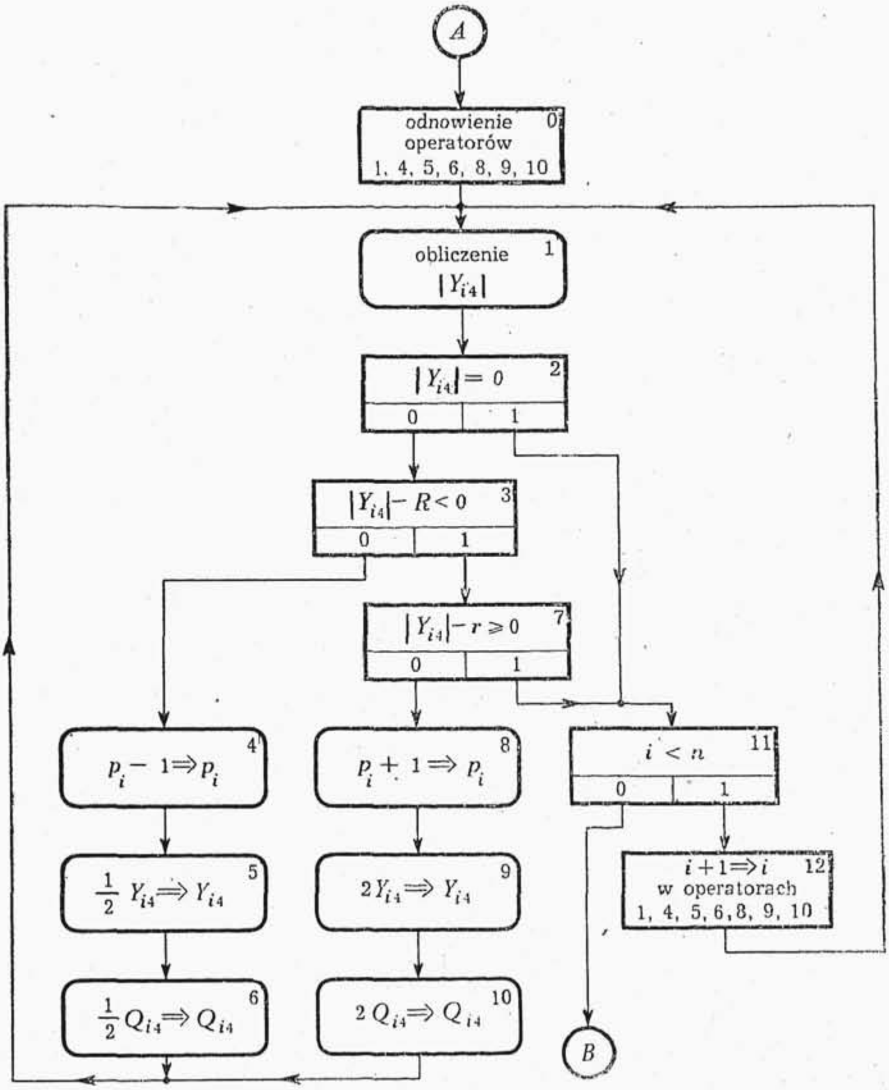
$$h = 2^{p_h}H, \quad |H|<1, \tag{4-21}$$

oraz

$$f_i(y_{1j}, y_{2j}, \dots, y_{nj}) = 2^{l_{ij}}F_{ij}, \quad |F_{ij}|<1, \tag{4-22}$$

gdzie $j = 1,2,3,4$.

Podstawiając wyrażenie (4-20), (4-21) i (4-22) do wzorów (4-19) po przekształceniu otrzymamy



Rys. 4-15. Schemat blokowy przenormowania wyników po jednym kroku metody Rungego-Kutty-Gilla z programowanym przecinkiem

$$\left.\begin{aligned} K_{i0} &= 2^{m+p_h+l_{i0}-p_i} H F_{i0}, \\ R_{i1} &= \tfrac{1}{2}(K_{i0} - Q_{i0}), \\ Y_{i1} &= Y_{i0} + 2^{-m} R_{i1}, \\ Q_{i1} &= Q_{i0} + 3R_{i1} - \tfrac{1}{2} K_{i0}, \end{aligned} \qquad \text{dla } i = 1,2, \ldots, n, \right\} \quad \text{(4-23a)}$$

$$\left.\begin{aligned} K_{i1} &= 2^{m+p_h+l_{i1}-p_i} H F_{i1}, \\ R_{i2} &= \left(1 - \sqrt{\tfrac{1}{2}}\right)(K_{i1} - Q_{i1}), \\ Y_{i2} &= Y_{i1} + 2^{-m} R_{i2}, \\ Q_{i2} &= Q_{i1} + 3R_{i2} - \left(1 - \sqrt{\tfrac{1}{2}}\right) K_{i1}, \end{aligned} \qquad \text{dla } i = 1,2, \ldots, n, \right\} \quad \text{(4-23b)}$$

$$\left.\begin{aligned} K_{i2} &= 2^{m+p_h+l_{i2}-p_i} HF_{i2}, \\ R_{i3} &= \sqrt{\tfrac{1}{2}}(K_{i2}-Q_{i2})+(K_{i2}-Q_{i2}), \\ Y_{i3} &= Y_{i2}+2^{-m}R_{i3}, \\ Q_{i3} &= Q_{i2}+3R_{i3}-\sqrt{\tfrac{1}{2}}K_{i2}-K_{i2}, \end{aligned}\right\} \text{dla } i=1,2,\ldots,n, \tag{4-23c}$$

$$\left.\begin{aligned} K_{i3} &= 2^{m+p_h+l_{i3}-p_i} HF_{i3}, \\ R_{i4} &= \tfrac{1}{3}(\tfrac{1}{2}K_{i3}-Q_{i3}), \\ Y_{i4} &= Y_{i3}+2^{-m}R_{i4}, \\ Q_{i4} &= Q_{i3}+3R_{i4}-\tfrac{1}{2}K_{i3}, \end{aligned}\right\} \text{dla } i=1,2,\ldots,n. \tag{4-23d}$$

Przy przejściu od s-tego kroku całkowania do $(s+1)$-ego sprawdzimy, czy y_{i4} obliczone w s-tym kroku spełniają nierówności postaci (4-13). W przypadku niespełnienia nierówności (4-13) dobieramy tak skale (czyli wyznaczamy nowe wartości p_i), aby przeskalowane wielkości spełniły już nierówności (4-13).

Wartości dla p_h, H, m oraz p_i^0 (początkowe wartości p_i) dobieramy tak, aby przy wykonywaniu jednego kroku całkowania nie nastąpiło przekroczenie zakresu, oraz po to, aby uniknąć normalizacji liczb przed rozpoczęciem obliczeń.

Uwaga: Jeśli $p_i = p_i^0$, to zamiast nierówności (4-13) wystarczy sprawdzić nierówność

$$|Y_{ij}| < R. \tag{4-24}$$

Powyższa uwaga nie daje zasadniczych korzyści praktycznych, ponieważ wymaga dodatkowej rozbudowy programu bez znaczniejszego wzrostu prędkości liczenia.

Przy przejściu od kroku s-tego do $(s+1)$-ego pamiętamy następujące $3n$ liczb, $Y_{14}, Y_{24}, \ldots, Y_{n4}, Q_{14}, Q_{24}, \ldots, Q_{n4}, p_1, p_2, \ldots, p_n$. Dla przechowywania wielkości Y_{i4}, Q_{i4} będziemy korzystali z komórek długich, dla przechowywania wielkości p_1 będziemy korzystali z komórek krótkich.

Na rysunku 4-15 przedstawiony jest schemat blokowy czwartej części programu z programowanym przecinkiem dla metody Rungego-Kutta-Gilla.

4.5. ORGANIZACJA BIBLIOTEKI PODPROGRAMÓW

Przez bibliotekę podprogramów rozumiemy zbiór programów, z których każdy przeznaczony jest do wykonywania pewnych obliczeń bądź pewnych czynności organizacyjnych. Programy te muszą być zbudowane tak, aby dawały się łączyć w sposób możliwie prosty z programem głównym. Ponadto programy te muszą być zapisane w adresach względnych.

Na to aby w łatwy sposób korzystać z biblioteki podprogramów, wprowadzimy specjalną klasyfikację działową. Ponadto do każdego programu dołączymy kartę z charakterystycznymi danymi podprogramu.

Wyżej omówione karty umożliwiają posługiwanie się podprogramem bez szczegółowej jego znajomości.

Tablica 4-12

Karta klasyfikacyjna podprogramu

1. Symbol klasyfikacji działowej: . . . .
2. Numer inwentarzowy: . . . .
3. Zastosowanie: . . . .
. . . .
4. Postać liczb: arytmetyka . . . .
liczby . . . .
przecinek . . . .
5. Opis formuły numerycznej lub wykonywanych czynności: . . . .
. . . .
. . . .
6. Podprogram liniowy — cykliczny — iteracyjny(1)
modyfikowany (nazwa modyfikatora) — ustalony(1)
odnawiający się — nie odnawiający się(1)
7. Wywołanie podprogramu za pomocą rozkazu . . . .
. . . .
. . . .
8. Wykorzystane podprogramy: . . . .
. . . .
9. Długość programu . . . . komórek krótkich
10. Liczba wykonywanych rozkazów: . . . .
11. Średni czas wykonywania: . . . .
. . . .
12. Rozmieszczenie danych wejściowych . . . .
. . . .
. . . .
13. Rozmieszczenie wyników . . . .
. . . .
. . . .
14. Komórki robocze: . . . .
. . . .
. . . .
15. Użyte parametry z pamięci stałej: . . . .
. . . .
16. Własne parametry logiczne: . . . .
. . . .
. . . .
17. Podprogram wykonał(a) . . . . dn. . . . .
18. Podprogram sprawdził(a) na maszynie . . . . dn. . . . .
19. Uwagi: . . . .
. . . .
. . . .

(1) Niepotrzebne skreślić.

Podamy jeszcze projekt klasyfikacji działowej podprogramów:

A. Wprowadzenie. Wyprowadzenie. Zmiana systemu pozycyjnego.

B. Kontrola pracy maszyny liczącej.

C. Demonstracja pracy maszyny liczącej.
D. Interpretacja kodów zewnętrznych. Kodowanie automatyczne.
F. Logika matematyczna.
G. Porządkowanie. Wybieranie. Zliczanie.
I. Specjalne działania arytmetyczne (zmiennoprzecinkowe, ze zwiększoną dokładnością).
J. Obliczenie wartości funkcji. Układanie tablic. Sumowanie szeregów.
K. Interpolacja. Ekstrapolacja. Aproksymacja.
M. Teoria liczb.
N. Algebra liniowa.
O. Równania algebraiczne nieliniowe, równania przestępne i układy.
Q. Przybliżone różniczkowanie. Przybliżone całkowanie.
R. Równania różnicowe.
S. Równania różniczkowe zwyczajne.
T. Równania różniczkowe cząstkowe.
U. Równania całkowe.
V. Inne zagadnienia analizy matematycznej.
X. Statystyka matematyczna.

W tablicy 4-12 przedstawiona jest karta klasyfikacyjna podprogramu.

5. ORGANIZACJA PRACY NA MASZYNIE TYPOWEJ

[5.1. Programy wprowadzające, 5.2. Programy wyprowadzające, 5.3. Uruchomienie maszyny, 5.4. Szukanie błędów w programach]

5.1. PROGRAMY WPROWADZAJĄCE

5.1.0. Programy i materiał liczbowy wprowadzamy do maszyny za pomocą specjalnych programów wprowadzających, które z symboli wydziurkowanych na taśmie perforowanej i odczytanych przez maszynę (tabl. 2-5) formują słowa (17 albo 34 bitowe) i umieszczają je pod odpowiednimi adresami w pamięci.

Mała maszyna cyfrowa nakłada dość trudne w praktyce do realizacji warunki na programy wprowadzające. Szczególnie ważne jest, aby program przeznaczony do wprowadzania rozkazów i pseudorozkazów (punkt 3.0) zapisanych w kodzie wewnętrznym w adresach względnych (punkt 3.1), dalej zwanym podstawowym programem wprowadzającym, działał stosunkowo szybko i był możliwie krótki.

Obecnie omówimy trzy programy wprowadzające, a mianowicie: