Metody numeryczne

Piotr Tatjewski

Metody numeryczne

Warszawa 2013
Oficyna Wydawnicza Politechniki Warszawskiej

Opiniodawca
Jerzy Klamka

Opracowanie redakcyjne
Teresa Woźniak

Skład komputerowy wykonał Autor

Projekt okładki
Danuta Czubek-Puchalska

ISBN 978-83-7814-160-0

Księgarnia internetowa Oficyny Wydawniczej PW www.wydawnictwopw.pl
tel. 22 234-75-03; fax 22 234-70-60; e-mail: oficyna at wpw.pw.edu.pl

Oficyna Wydawnicza PW, ul. Polna 50, 00-644 Warszawa. Wydanie 1.
Druk i oprawa: Drukarnia Oficyny Wydawniczej Politechniki Warszawskiej, tel. 22 234-55-93

Spis treści

Przedmowa

Prezentowana wersja niniejszej pracy powstała na podstawie wieloletnich doświadczeń autora w prowadzeniu przedmiotów z zakresu metod numerycznych na Wydziale Elektroniki i Technik Informacyjnych Politechniki Warszawskiej, w ostatnich latach przedmiotów „Metody Numeryczne" (dla studentów kierunku Informatyka) i „Numerical Methods" (dla studentów studiów w języku angielskim Electrical and Computer Engineering). Jest to kolejna wersja, będąca podsumowaniem wielu poprzednich, sukcesywnie doskonalonych i na bieżąco dostępnych dla studentów na wydziałowej stronie przedmiotu. O ile prezentowana wersja w niezbyt dużym stopniu różni się od bezpośrednio poprzedzającej (z roku 2012), o tyle bardzo istotnie różni się od wszystkich poprzednich, zawierając wiele daleko idących modyfikacji, uzupełnień i rozszerzeń. Zawiera nowy rozdział 8 dotyczący metod numerycznego różniczkowania i całkowania, wiele nowych lub zmienionych podrozdziałów, przykładów i rysunków oraz nieco inny układ treści.

Autor chciałby w tym miejscu podziękować wszystkim studentom aktywnie uczestniczącym w prowadzonych zajęciach, za zgłaszane uwagi do sposobu prezentacji, za wykrywane usterki edycyjne. Uwagi te w istotnym stopniu przyczyniały się do powstawania kolejnych, coraz lepszych, wersji niniejszego opracowania.

Autor chciałby też podziękować recenzentowi, profesorowi Jerzemu Klamce, za szereg uwag, które przyczyniły się do dalszej poprawy jakości pracy. Podziękowanie należy się także pani Teresie Woźniak z redakcji Oficyny Wydawniczej Politechniki Warszawskiej, za wnikliwą korektę tekstu pracy.

Piotr Tatjewski
Warszawa, wrzesień 2013

Rozdział 1

Pojęcia podstawowe

1.1. Maszynowa reprezentacja liczb, błędy reprezentacji

Reprezentacja zmiennoprzecinkowa $x_{t,r}$ liczby x określona jest zależnością

$$x_{t,r} = m_t \cdot P^{c_r},$$

gdzie: m_t – mantysa, t – liczba znaków mantysy,
c_r – cecha, r – liczba znaków cechy,
P – podstawa, ograniczymy się do $P = 2$.

Aby reprezentacja zmiennoprzecinkowa była dobrze określona, trzeba zdefiniować, do jakiego zakresu liczb należy mantysa – tzn. trzeba dokonać *normalizacji*. Najbardziej naturalną reprezentację znormalizowaną liczby dostajemy, zakładając, że

$$0.5 \leq |m_t| < 1, \tag{1.1}$$

gdyż wówczas t pozycjom mantysy odpowiada t pierwszych ujemnych potęg liczby 2, tzn. pierwszych t pozycji po przecinku mantysy jako liczby w postaci binarnej, np. $m_7 = 1011011$ odpowiada liczbie 0,1011011, $m_4 = 1011$ odpowiada liczbie 0,1011. Rozważając przykładową binarną reprezentację zmiennoprzecinkową dla $t = 4$ i $r = 2$ oraz dwóch bitów znaków, możemy napisać:

$$\mathbf{0}10|\mathbf{0}1011,$$

gdzie linią pionową oddzielono pola wykładnika i mantysy, a pierwsze pozycje każdego pola zajmują bity znaków (wytłuszczone); 0 odpowiada znakowi plus, gdyż $(-1)^0 = 1$. Podana liczba w systemie dziesiętnym wynosi

$$x = 2^{(-1)^0(1\cdot 2^1 + 0\cdot 2^0)} \cdot (-1)^0(1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} + 1 \cdot 2^{-4}) = 2.75$$

i kolejne potęgi liczby 2 przechodzą płynnie od pozycji mantysy do pozycji wykładnika (rosnąco). Natomiast *w* standardzie IEEE 754 dla reprezentacji zmiennoprzecinkowej przyjęto inną *konwencję normalizacji*:

$$1 \leq |m_t| < 2, \tag{1.2}$$

w której pozycja przecinka (oddzielającego część całkowitą od ułamkowej, dziesiętnie) jest po pierwszej pozycji mantysy traktowanej jako samodzielna liczba, np. $m_7 = 1011011$ odpowiada liczbie 1,011011, $m_4 = 1011$ odpowiada liczbie 1,011 (jedynka jest zawsze na pierwszej pozycji każdej mantysy znormalizowanej). Ta sama liczba co poprzednio (dziesiętnie), w tej konwencji zapisana jest w postaci (mantysa pomnożona przez 2, wykładnik zmniejszony o 1)

$$x = 2^{(-1)^0(0\cdot 2^1+1\cdot 2^0)} \cdot (-1)^0(1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3}) = 2.75$$

Co więcej, aby uniknąć stosowania bitu wykładnika, jest on w standardzie IEEE 754 podawany bez znaku, ale przesunięty o wartość odpowiednio przyjętą dla liczby jego bitów.

Wg *IEEE Standard 754*, w formacie 32-bitowym (*single precision*) mamy:
- mantysa 24-bitowa, z normalizacją: $1 \leq |m_{24}| < 2$,
- wykładnik 8-bitowy kodowany z przesunięciem o $2^7 - 1 = 127$, tzn.

bit znaku (1 bit)	wykładnik przesunięty (8 bitów)	znormalizowana mantysa 24-pozycyjna (pierwszy bit domyślny, zawsze równy 1) (23 bity)

Reprezentacja liczby w tym standardzie jest więc następująca:

$$x_{IEEE754,32} = (-1)^s \cdot m_{24} \cdot 2^{c_8 - 127},$$

gdzie s – bit znaku, c_8 – wykładnik przesunięty o 127, tzn. zakres wykładnika jest od -127 (8 zer) do $+128$ (8 jedynek), a zerowy wykładnik (dla liczb, dziesiętnie, z zakresu [1,2)) jest reprezentowany przez "01111111". Reprezentacja liczby 2,75 przedstawianej uprzednio ma w rozważanym standardzie następującą postać binarną (dla ułatwienia odczytu oddzielono pola znaku, wykładnika i mantysy):

$$0|10000000|01100000000000000000000000000000$$

Liczba w systemie dwójkowym z mantysą o długości 24 bity odpowiada liczbie o 7 cyfrach znaczących w systemie dziesiętnym.

Trzeba pamiętać, że zbiór liczb maszynowych $M \subset R$ jest *zbiorem skończonym*, przy czym:
- im większe t, tym M jest liczniejszy dla tego samego zakresu, a więc „gęściejszy",
- im większe r, tym M pokrywa większy zakres liczb (przedział na osi liczbowej).

Zakładając że reprezentacja cechy jest dokładna (pokrywa cały interesujący zakres liczb), tj. $c_r = c$, oraz oznaczając maszynowe przybliżenie liczby x przez $\mathrm{rd}(x)$ (od angielskiego terminu *rounding* – zaokrąglenie), mamy

$$\mathrm{rd}\,(x) = m_t \cdot 2^c.$$

Przybliżenie jest najlepsze, jeśli zachodzi relacja

$$|\mathrm{rd}\,(x) - x| \leq \min_{g \in M} |g - x|\,.$$

Postulat ten jest spełniony przez typowe *zaokrąglenie*, dla mantysy znormalizowanej wg (1.1) możemy je zapisać w postaci

$$m_t = \sum_{i=1}^{t} e_{-i} \cdot 2^{-i} + e_{-(t+1)} \cdot 2^{-t}, \tag{1.3}$$

gdzie $e_{-1} = 1$, $e_{-i} = 0$ lub 1, $i = 2, 3, ..., t+1$. Stąd, błąd zaokrąglenia (*roundoff error*) może być oszacowany następująco:

$$|m - m_t| \leq 2^{-(t+1)},$$

gdzie m oznacza mantysę dokładną ($x = m \cdot 2^c$). Jeśli na pierwszej odrzucanej pozycji mantysy jest zero, to operacja zaokrąglania nie występuje (następuje obcięcie bitów) i mantysa m_t jest z niedomiarem nie większym od $2^{-(t+1)}$. Natomiast jeśli na pierwszej odrzucanej pozycji mantysy jest jedynka – co oznacza, że odrzucana część mantysy jest liczbą z zakresu $[2^{-(t+1)}, 2^{-t})$ – to zaokrąglanie (tzn. dodanie liczby 2^{-t}) występuje i mantysa m_t jest z nadmiarem, nie większym od $2^{-(t+1)}$.

W istocie interesują nas *błędy względne reprezentacji zmiennopozycyjnej liczb*. Dla takiego błędu mamy

$$\frac{\mathrm{rd}\,(x) - x}{x} = \frac{m_t 2^c - m 2^c}{m 2^c} = \frac{m_t - m}{m},$$

skąd *błąd względny* zaokrąglenia można oszacować następująco:

$$\left| \frac{\mathrm{rd}\,(x) - x}{x} \right| = \frac{|m_t - m|}{|m|} \leq \frac{2^{-(t+1)}}{2^{-1}} = 2^{-t},$$

gdyż dla mantysy znormalizowanej wg (1.1) mamy $|m| \geq 2^{-1}$.

Natomiast uzyskana zależność

$$\left| \frac{\mathrm{rd}\,(x) - x}{x} \right| = \frac{|m_t - m|}{|m|} \leq 2^{-t} \tag{1.4}$$

jest uniwersalna, nie zależy od przyjętego wzorca normalizacji, gdyż różne normalizacje różnią się tylko pomnożeniem mantysy przez odpowiednią potęgę liczby 2, por. (1.1) i (1.2), ulegającą redukcji przy rozważaniu błędu względnego. Stąd:

> *maksymalny błąd względny reprezentacji zmiennoprzecinkowej zależy jedynie od liczby bitów mantysy* i nazywany jest *dokładnością maszynową* (*precyzją maszynową – machine precision*), i oznaczany jest tradycyjnie przez eps.

Dla mantysy t-bitowej i zaokrąglania (metodą jak w (1.3), niezależnie od pozycji przecinka) dokładność maszynowa $eps = 2^{-t}$.

Zależność (1.4) można dla każdej liczby x zapisać w równoważnej postaci

$$\mathrm{rd}(x) - x = x \cdot \varepsilon, \ \text{ gdzie } \ |\varepsilon| \leq 2^{-t} = eps,$$

czyli

$$\mathrm{rd}\,(x) = x\,(1+\varepsilon)\,, \quad |\varepsilon| \leq eps, \tag{1.5}$$

gdzie ε to liczba (dodatnia lub ujemna) reprezentująca względny błąd zaokrąglenia liczby x. Równanie (1.5) to *fundamentalna zależność wykorzystywana w analizie numerycznej*.

Łatwo zauważyć, że stosując słowo o podwójnej długości mantysy, mamy

$$eps_{mdl} = (eps)^2,$$

gdzie indeks „mdl" oznacza „*mantissa double length*". Wg IEEE Standard 754 liczba podwójnej precyzji (*double precision*) kodowana jest na 64 bitach, następująco:

bit znaku (1 bit)	wykładnik przesunięty (11 bitów)	znormalizowana mantysa 53-pozycyjna (pierwszy bit domyślny, zawsze równy 1) (52 bity)

Liczba w systemie dwójkowym z mantysą o długości 53 bity odpowiada liczbie o 16 cyfrach znaczących w systemie dziesiętnym.

Aproksymacja maszynowa liczby przez tzw. *obcięcie* (*truncation*) ma, przy normalizacji (1.1), mantysę postaci (por. (1.3))

$$m_t = \sum_{i=1}^{t} e_{-i} \cdot 2^{-i}.$$

Błąd reprezentacji maszynowej, zmiennopozycyjnej nazywa się teraz *błędem obcięcia* (*truncation error*). Jego oszacowanie wyprowadza się identycznie jak poprzednio, tylko dostajemy teraz błąd dwukrotnie większy, $eps = 2^{-t+1}$ zamiast $eps = 2^{-t}$,

$$\mathrm{rd}\,(x) = x\,(1+\varepsilon)\,, \quad |\varepsilon| \leq 2^{-t+1} = eps.$$

1.2. Arytmetyka zmiennopozycyjna

Elementarne działania arytmetyczne to dodawanie i odejmowanie($+, -$), mnożenie ($\cdot$) i dzielenie ($/$). Wynikiem każdego z tych działań na liczbach maszynowych nie jest na ogół liczba maszynowa. Natomiast, oznaczając przez „*fl*" wynik działania zmiennopozycyjnego (od ang. *floating point*) wyniki elementarnych działań arytmetycznych możemy zapisać w postaci

$$fl\,(x \pm y) = (x \pm y) \cdot (1+\varepsilon)\,,$$

$$fl\,(x \cdot y) = (x \cdot y)\,(1+\varepsilon)\,,$$

$$fl\,(x/y) = (x/y)\,(1+\varepsilon)\,,$$

gdzie $|\varepsilon| \leq eps$, tzn. przyjmujemy, że błąd elementarnych operacji arytmetycznych to jedynie błąd zaokrąglenia dokładnego wyniku tych operacji. Wynika to ze standardu IEEE 754, który wymaga organizacji jednostki arytmetycznej komputera zapewniającej osiąganie takiej dokładności elementarnych operacji arytmetycznych.

Co więcej, z reguły podstawowe funkcje elementarne również realizowane są w sposób zapewniający wysoką dokładność, np. pierwiastek realizowany jest z dokładnością operacji elementarnych: $fl(\sqrt{x}) = \sqrt{x}(1+\varepsilon),\ |\varepsilon| \leq eps$.

Uwaga. Dokładność maszynową eps można również określić jako najmniejszą dodatnią liczbę maszynową g taką, że zachodzi relacja $fl\,(1+g) > 1$, tzn.

$$eps \stackrel{\text{df}}{=} \min\{g \in M : fl(1+g) > 1\ ,\ g > 0\}. \tag{1.6}$$

W programach występują zwykle całe ciągi elementarnych operacji arytmetycznych i obliczania funkcji elementarnych, ponadto wykorzystywane dane wejściowe to najczęściej liczby obarczone błędami reprezentacji maszynowej. Wszystkie te błędy ulegają *propagacji i modyfikacji* przy kolejnych operacjach, co może prowadzić do poważnej *kumulacji błędów*. Stąd całkowity błąd wyniku zadania obliczanego numerycznie jest z reguły znacznie większy od dokładności maszynowej. Błąd taki analizować można na dwa sposoby:

– *metodą probablistyczną*, zakładając, że poszczególne błędy jednostkowe to niezależne, nieskorelowane zmienne losowe i wyznaczając np. *błąd średni*,

– *metodą najgorszego przypadku*, wyznaczając *oszacowanie maksymalnego możliwego modułu błędu*.

Przykład 1.1. Wyznaczymy oszacowanie błędu maksymalnego zadania dodawania trzech liczb:

$$y = a + b + c,$$

algorytmem (ustala jednoznaczną kolejność wykonywania operacji elementarnych)

$$y = (a + b) + c.$$

Realizując algorytm w arytmetyce zmiennopozycyjnej, mamy

$$y = [(a(1+\varepsilon_a) + b(1+\varepsilon_b))(1+\varepsilon_1) + c(1+\varepsilon_c)](1+\varepsilon_2),$$

gdzie $\varepsilon_a, \varepsilon_b, \varepsilon_c$ to błędy reprezentacji liczb a, b i c, a ε_1 i ε_2 reprezentują błędy kolejnych dodawań; wszystkie te epsilony są co do modułu mniejsze od eps.

Przekształcając powyższe wyrażenie, dostajemy

$$\begin{aligned} y &= [(a + a\varepsilon_a + b + b\varepsilon_b)(1 + \varepsilon_1) + c + c\varepsilon_c)](1 + \varepsilon_2) \\ &= [a + a\varepsilon_a + b + b\varepsilon_b + a\varepsilon_1 + a\varepsilon_a\varepsilon_1 + b\varepsilon_1 + b\varepsilon_b\varepsilon_1 + c + c\varepsilon_c)](1 + \varepsilon_2) \\ &= a + a\varepsilon_a + b + b\varepsilon_b + a\varepsilon_1 + a\varepsilon_a\varepsilon_1 + b\varepsilon_1 + b\varepsilon_b\varepsilon_1 + c + c\varepsilon_c + \\ &\quad + a\varepsilon_2 + a\varepsilon_a\varepsilon_2 + b\varepsilon_2 + b\varepsilon_b\varepsilon_2 + a\varepsilon_1\varepsilon_2 + a\varepsilon_a\varepsilon_1\varepsilon_2 + \\ &\quad + b\varepsilon_1\varepsilon_2 + b\varepsilon_b\varepsilon_1\varepsilon_2 + c\varepsilon_2 + c\varepsilon_c\varepsilon_2. \end{aligned}$$

Zauważmy teraz, że ponieważ moduł każdego pojedynczego błędu ($|\varepsilon_i|$) jest mniejszy od eps, więc moduły iloczynów dwóch epsilonów są mniejsze od eps^2, a trzech od eps^3, przy czym $eps^2 \ll eps$, tym bardziej $eps^3 \ll eps$. Stąd przy analizie możemy te składniki pominąć, piszemy wówczas zamiast „=" znak „$\underset{1}{=}$". Symbol ten oznacza, w przybliżeniu, pozostawienie składników sumy z błędami danych występującymi jedynie liniowo (dokładniej, chodzi o pominięcie składników malejących silniej w stosunku $k \cdot 2^{-t}$ niż pozostałe wraz ze wzrostem t, gdzie k pewna stała, zob. [1]). Postępując w ten sposób, dostajemy

$$\begin{aligned} y &\underset{1}{=} a + a\varepsilon_a + b + b\varepsilon_b + a\varepsilon_1 + b\varepsilon_1 + c + c\varepsilon_c + a\varepsilon_2 + b\varepsilon_2 + c\varepsilon_2 \\ &= a + b + c + a(\varepsilon_a + \varepsilon_1 + \varepsilon_2) + b(\varepsilon_b + \varepsilon_1 + \varepsilon_2) + c(\varepsilon_c + \varepsilon_2) \\ &= (a + b + c)\left[1 + \frac{a(\varepsilon_a + \varepsilon_1 + \varepsilon_2)}{a + b + c} + \frac{b(\varepsilon_b + \varepsilon_1 + \varepsilon_2)}{a + b + c} + \frac{c(\varepsilon_c + \varepsilon_2)}{a + b + c}\right] \\ &= (a + b + c)\left[1 + \delta\right], \end{aligned}$$

gdzie δ reprezentuje błąd względny wyniku. Szacując maksymalny moduł tego błędu, otrzymujemy

$$\begin{aligned} |\delta| &= \left|\frac{a(\varepsilon_a + \varepsilon_1 + \varepsilon_2)}{a + b + c} + \frac{b(\varepsilon_b + \varepsilon_1 + \varepsilon_2)}{a + b + c} + \frac{c(\varepsilon_c + \varepsilon_2)}{a + b + c}\right| \\ &\leq \left|\frac{a(\varepsilon_a + \varepsilon_1 + \varepsilon_2)}{a + b + c}\right| + \left|\frac{b(\varepsilon_b + \varepsilon_1 + \varepsilon_2)}{a + b + c}\right| + \left|\frac{c(\varepsilon_c + \varepsilon_2)}{a + b + c}\right| \\ &\leq \frac{|a|3eps}{|a + b + c|} + \frac{|b|3eps}{|a + b + c|} + \frac{|c|2eps}{|a + b + c|} \\ &= \frac{(|a| + |b|)3eps + |c|2eps}{|a + b + c|} \leq \frac{(|a| + |b| + |c|)}{|a + b + c|}3eps. \end{aligned}$$

Powyższe oszacowanie wskazuje, że błąd względny dodawania trzech liczb może być bardzo duży, jeśli moduł sumy $a+b+c$ jest mały, a moduły poszczególnych jej składników są duże (liczby mogą być dodatnie i ujemne). Zjawisko to może wystąpić przy dodawaniu dowolnej liczby argumentów (poczynając od dwóch) i związane jest z *redukcją cyfr znaczących*. □

1.3. Uwarunkowanie zadania

Matematyczne zadanie obliczeniowe można przedstawić jako pewne odwzorowanie $\phi : R^n \longmapsto R^m$,

$$\mathbf{w} = \phi(\mathbf{d}),$$

gdzie $\mathbf{d} = [d_1 \ d_2 \ \cdots \ d_n]^{\mathrm{T}}$ – wektor danych, $\mathbf{w} = [w_1 \ w_2 \ \cdots \ w_m]^{\mathrm{T}}$ – wektor wyniku (rozwiązania). W praktyce dysponujemy jedynie reprezentacjami maszynowymi danych, tj. $\mathrm{rd}(d_i) = d_i(1+\varepsilon_i)$, gdzie $|\varepsilon_i| \leq eps$, $i = 1, ..., n$.

Definicja. Zadanie obliczeniowe nazywamy *źle uwarunkowanym* (*ill-conditioned*), jeśli niewielkie (względne) zaburzenia danych zadania powodują duże (względne) zmiany jego rozwiązania. *Wskaźnikiem uwarunkowania* (*condition number*) nazywamy wielkość charakteryzującą ilościowo maksymalny możliwy stosunek (iloraz) względnego błędu wyniku do względnego błędu (zaburzenia) danych.
Ponieważ rozważamy błędy względne, to definicja powyższa dotyczy w istocie *względnego wskaźnika uwarunkowania (relative condition number)* – nie będziemy jednak tego dalej podkreślać.

Przykład 1.2. Rozważmy zadanie obliczania wartości iloczynu skalarnego dwóch wektorów: $\phi(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{n} a_i b_i$, dla

a) $\mathbf{a} = [1 \ \ 2 \ \ 3]$, $\mathbf{b} = [2 \ \ 6 \ \ -5]$,

b) $\mathbf{a} = [1.02 \ \ 2.04 \ \ 2.96]$, $\mathbf{b} = [2 \ \ 6 \ \ -5]$.

W przypadku a) mamy $w = \phi(\mathbf{a}, \mathbf{b}) = -1$. W przypadku b) mamy $w = -0.52$, tzn. zaburzenia części danych na poziomie 2% spowodowały zaburzenie (błąd) wyniku na poziomie 48%.

Jak łatwo sprawdzić, zastępując wektor $\mathbf{b}$ wektorem $[2 \ \ 6 \ \ 5]$, uzyskamy:
a) $w = \phi(\mathbf{a}, \mathbf{b}) = 29$, b) $w = 29.08$, tzn. zaburzenie (błąd) wyniku na poziomie zaburzenia danych w obu przypadkach . □

Wniosek. *Uwarunkowanie zadania zależy od konkretnych wartości danych, dla których to zadanie liczymy.*

Oznaczając przez $\Delta\mathbf{w} = [\Delta w_1 \ \Delta w_2 \ \cdots \ \Delta w_m]^{\mathrm{T}}$ błąd bezwzględny wyniku, błąd względny wyniku można przedstawić w postaci

$$\frac{\|\Delta\mathbf{w}\|}{\|\mathbf{w}\|} = \frac{\|\phi(\mathbf{d}+\Delta\mathbf{d}) - \phi(\mathbf{d})\|}{\|\phi(\mathbf{d})\|},$$

gdzie $\Delta\mathbf{d} = [\Delta d_1 \ \Delta d_2 \ \cdots \ \Delta d_n]^{\mathrm{T}}$ to wektor błędów bezwzględnych danych.

Oznaczając przez $\frac{\|\Delta\mathbf{d}\|}{\|\mathbf{d}\|}$ względny błąd wektora danych, *wskaźnik uwarunkowania* zadania ϕ można ogólnie zdefiniować w następujący sposób:

$$\mathrm{cond}_\phi(\mathbf{d}) = \lim_{\delta\to 0} \sup_{\|\Delta\mathbf{d}\|\le\delta} \frac{\frac{\|\phi(\mathbf{d}+\Delta\mathbf{d})-\phi(\mathbf{d})\|}{\|\phi(\mathbf{d})\|}}{\frac{\|\Delta\mathbf{d}\|}{\|\mathbf{d}\|}}, \tag{1.7}$$

tzn. jest to maksymalny możliwy wzrost względnego błędu wyniku, w stosunku do względnego błędu danych, przy braniu pod uwagę wszystkich możliwych bardzo małych (dążących do zera) odchyłek danych.

Wzór (1.7) w ogólności nie jest wygodny, ale może być przekształcony do bardziej konstruktywnej postaci, jeśli zadanie $\phi(\mathbf{d})$ można przedstawić w postaci *ciągłego i różniczkowalnego odwzorowania o znanej postaci funkcyjnej*.

Załóżmy, że $\phi(\mathbf{d}) = f(\mathbf{d}),\ f : \mathbb{R}^n \to \mathbb{R}$, gdzie funkcja f jest różniczkowalna, oraz ograniczmy się do normy euklidesowej wektorów. Ponieważ dla dostatecznie małych odchyłek danych $\triangle\mathbf{d}$

$$|f(\mathbf{d}+\Delta\mathbf{d}) - f(\mathbf{d})| \approx \left| f'(\mathbf{d}) \cdot \triangle\mathbf{d} \right|$$

oraz dla wektorowej normy euklidesowej i każdego skalara $\delta > 0$

$$\max_{\triangle\mathbf{d}\le\delta} \left| f'(\mathbf{d}) \cdot \triangle\mathbf{d} \right| = \left\| f'(\mathbf{d}) \right\| \|\triangle\mathbf{d}\| \left(= \left\| f'(\mathbf{d}) \right\| \delta \right)$$

(maksimum jest osiągane dla przyrostu danych wzdłuż kierunku gradientu), więc bezpośrednio z (1.7) mamy

$$\mathrm{cond}_f(\mathbf{d}) = \lim_{\delta\to 0} \sup_{\|\Delta\mathbf{d}\|\le\delta} \frac{\frac{|f(\mathbf{d}+\Delta\mathbf{d})-f(\mathbf{d})|}{|f(\mathbf{d})|}}{\frac{\|\Delta\mathbf{d}\|}{\|\mathbf{d}\|}} = \frac{\left\| f'(\mathbf{d}) \right\| \|\mathbf{d}\|}{|f(\mathbf{d})|}. \tag{1.8}$$

Zależność powyższa wyraźnie pokazuje (co podkreślono we wniosku na poprzedniej stronie), że wartość wskaźnika uwarunkowania zależy od konkretnych wartości danych $\mathbf{d}$, tzn. dla każdego zadania ϕ, wskaźnik $\mathrm{cond}_\phi(\mathbf{d})$ jest w istocie odwzorowaniem $\mathbb{R}^n \supseteq D \ni \mathbf{d} \to \mathrm{cond}_\phi(\mathbf{d}) \in \mathbb{R}_+$, gdzie D jest zbiorem wartości danych, dla których dane zadanie ϕ jest dobrze określone.

Należy zaznaczyć, że dla konkretnej, *skończonej odchyłki wektora danych* wskaźnik uwarunkowania (1.8) daje jedynie *oszacowanie* względnej zmiany wyniku – tym dokładniejsze, im odchyłka wektora danych jest mniejsza i jej kierunek bliższy kierunkowi gradientu funkcji f.

Przykład 1.3. Wyznaczymy wskaźnik uwarunkowania (1.8) dla zadania obliczania funkcji iloczynu skalarnego, $\phi(\mathbf{d}) = f(\mathbf{a},\mathbf{b}) = \sum_{i=1}^n a_i b_i$, stosując normę euklidesową wektorów. Wektor danych zdefiniujemy w postaci

$$\mathbf{d} = [a_1\ a_2\ \cdots\ a_n\ b_1\ b_2\ \cdots\ b_n]^{\mathrm{T}}.$$

Wyznaczając wskaźnik uwarunkowania wg (1.8), mamy

$$\mathrm{cond}_f(\mathbf{d}) = \frac{\left\|[b_1\ b_2\ \cdots\ b_n\ a_1\ a_2\ \cdots\ a_n]^{\mathrm{T}}\right\|_2 \|\mathbf{d}\|_2}{|\sum_{i=1}^{n} a_i b_i|} = \frac{\sum_{i=1}^{n}(a_i^2 + b_i^2)}{|\sum_{i=1}^{n} a_i b_i|}. \qquad (1.9)$$

Dla danych rozważonych w przykładzie 1.2 mamy:

1) $\mathbf{a} = [1\ \ 2\ \ 3]$, $\mathbf{b} = [2\ \ 6\ \ -5]$, stąd $\mathrm{cond}_f(\mathbf{d}) = 79$,

2) $\mathbf{a} = [1\ \ 2\ \ 3]$, $\mathbf{b} = [2\ \ 6\ \ +5]$, stąd $\mathrm{cond}_f(\mathbf{d}) = \frac{79}{29} \approx 2.7$. □

Przykład 1.4. Rozważymy uwarunkowanie zadania obliczania aproksymacji pochodnej funkcji $g(x)$ ilorazem różnicowym wstecznym, tzn.

$$f(\mathbf{d}) = \frac{g(x) - g(x-h)}{h},$$

gdzie $\mathbf{d} = [g(x)\ g(x-h)]^{\mathrm{T}}$ jest wektorem danych obarczonych błędami względnymi $\varepsilon_1, \varepsilon_2$, tzn.

$$\begin{aligned} fl(g(x)) &= g(x)(1+\varepsilon_1), \quad \varepsilon_1 \le Eps, \\ fl(g(x-h)) &= g(x-h)(1+\varepsilon_2), \quad \varepsilon_2 \le Eps, \end{aligned}$$

gdzie $Eps \ge eps$, gdyż błędy $\varepsilon_1, \varepsilon_2$ reprezentują błędy względne obliczonych numerycznie wartości funkcji $g(x)$ i $g(x-h)$ (funkcja g może mieć postać złożoną). Przyjmiemy ponadto, że krok $h > 0$ jest potęgą dwójki, tzn. bez błędu reprezentacji maszynowej (bez błędu danych). Stąd mamy

$$\begin{aligned} \mathrm{cond}_f(\mathbf{d}) &= \frac{\left\|[1/h, -1/h]^{\mathrm{T}}\right\|_2 \left\|[g(x), g(x-h)]^{\mathrm{T}}\right\|_2}{|g(x) - g(x-h)|/h} \\ &= \frac{\sqrt{2}\sqrt{g(x)^2 + g(x-h)^2}}{|g(x) - g(x-h)|} \approx \frac{2|g(x)|}{|g(x) - g(x-h)|}. \qquad (1.10) \end{aligned}$$

Z uzyskanej zależności wynika, że nie można dobierać zbyt małego kroku h przy numerycznej aproksymacji pochodnej, gdyż malenie kroku skutkuje zmniejszaniem różnicy $|g(x) - g(x-h)|$, a stąd uwarunkowanie zadania pogarsza się. Z drugiej strony, im mniejszy krok tym, teoretycznie, lepsza aproksymacja pochodnej – efekt przeciwny. W rozdziale 8.1 pokażemy, że dla zadania numerycznej aproksymacji pochodnej istnieje optymalna wartość kroku minimalizująca sumę błędu metody aproksymacji i błędów numerycznych. □

Przykład 1.5. Rozważymy wskaźniki uwarunkowania funkcji

$$x_i : \mathbb{R}^3 \ni (a, b, c) \to \mathbb{C}, \quad i = 1, 2,$$

wyznaczających pierwiastki wielomianu kwadratowego $w(x) = ax^2 + bx + c$, $a \neq 0$, według popularnych wzorów:

$$x_1(a,b,c) = \frac{-b+\sqrt{b^2-4ac}}{2a}, \tag{1.11}$$

$$x_2(a,b,c) = \frac{-b-\sqrt{b^2-4ac}}{2a}. \tag{1.12}$$

Wektor danych to wektor współczynników wielomianu, $\mathbf{d} = [a\ b\ c]^{\mathrm{T}}$. Mamy

$$x_1^{'}(a,b,c) = \left[\frac{2ac-b^2+b\sqrt{b^2-4ac}}{2a^2\sqrt{b^2-4ac}} \quad \frac{b-\sqrt{b^2-4ac}}{2a\sqrt{b^2-4ac}} \quad \frac{-1}{\sqrt{b^2-4ac}}\right],$$

$$x_2^{'}(a,b,c) = \left[\frac{-2ac+b^2+b\sqrt{b^2-4ac}}{2a^2\sqrt{b^2-4ac}} \quad \frac{-b-\sqrt{b^2-4ac}}{2a\sqrt{b^2-4ac}} \quad \frac{1}{\sqrt{b^2-4ac}}\right].$$

Stosując teraz wzór (1.8), dostajemy

$$\mathrm{cond}_{x_1}(\mathbf{d}) = \frac{2a\left\|x_1^{'}(\mathbf{d})\right\|\sqrt{a^2+b^2+c^2}}{|-b+\sqrt{b^2-4ac}|},$$

$$\mathrm{cond}_{x_2}(\mathbf{d}) = \frac{2a\left\|x_2^{'}(\mathbf{d})\right\|\sqrt{a^2+b^2+c^2}}{|-b-\sqrt{b^2-4ac}|}.$$

Analizując powyższe wzory, zauważamy dwie charakterystyczne cechy.

Po pierwsze, moduły pochodnych cząstkowych wzrastają do nieskończoności wraz z dążeniem b^2-4ac do zera, stąd i wskaźniki uwarunkowania zachowują się podobnie. Przypomnijmy, że wartość $b^2-4ac = 0$ odpowiada podwójnemu pierwiastkowi rzeczywistemu, który jest granicą między parami pierwiastków rzeczywistych i zespolonych (przy zmianach współczynników równania).

Po drugie, wskaźnik uwarunkowania funkcji definiującej pierwiastek o mniejszym module bardzo wzrasta, gdy $b^2 \gg 4ac$, prowadząc do znacznie większego błędu względnego niż dla pierwiastka o większym module. Możemy tego uniknąć, wykorzystując tylko ten z wzorów (1.11), (1.12), który dla aktualnych wartości a, b i c definiuje pierwiastek o większym module, tzn. ten, dla którego moduł licznika jest większy – a drugi pierwiastek wyznaczając z jednego z wzorów Viéte'a, najlepiej z wzoru na sumę pierwiastków ($x_1 + x_2 = -b/a$).

W niektórych aplikacjach potrzebny jest jedynie pierwiastek o mniejszym module. Wówczas, zamiast postępowania opisanego powyżej, wygodniej jest wykorzystać alternatywne wobec (1.11) i (1.12) wzory na pierwiastki

$$x_1(a,b,c) = \frac{2c}{-b-\sqrt{b^2-4ac}}, \tag{1.13}$$

$$x_2(a,b,c) = \frac{2c}{-b+\sqrt{b^2-4ac}}, \tag{1.14}$$

które uzyskujemy, podstawiając zależności (1.11) i (1.12) do wzoru Viéte'a na iloczyn pierwiastków ($x_1 \cdot x_2 = c/a$). Poszukując pierwiastka o mniejszym module, korzystamy wówczas z tego z wzorów (1.13), (1.14), w którym mianownik ma większy moduł (funkcja wyznaczająca ten pierwiastek jest lepiej uwarunkowana). □

Jeśli błędy danych są jedynie błędami reprezentacji maszynowej liczb, tzn. $\mathrm{rd}(d_i) = d_i(1+\varepsilon_i)$, $\Delta d_i = d_i\varepsilon_i$, $|\varepsilon_i| \leq eps$, $i = 1, ..., n$, to dla każdej z najbardziej użytecznych norm wektorów, $\|\cdot\|_1$, $\|\cdot\|_2$ i $\|\cdot\|_\infty$ (zob. rozdział 2), mamy $\frac{\|\Delta \mathbf{d}\|}{\|\mathbf{d}\|} \leq eps$, np. dla normy drugiej

$$\frac{\|\Delta \mathbf{d}\|}{\| \mathbf{d} \|} = \frac{\sqrt{(d_1\varepsilon_1)^2 + (d_2\varepsilon_2)^2 + \cdots + (d_n\varepsilon_n)^2}}{\| \mathbf{d} \|} \leq \frac{\| \mathbf{d} \| eps}{\| \mathbf{d} \|} = eps. \quad (1.15)$$

Z definicji (1.7) wynika bezpośrednio nierówność (przybliżona)

$$\frac{\|\Delta \mathbf{w}\|}{\|\mathbf{w}\|} \lesssim \mathrm{cond}_\phi(\mathbf{d}) \cdot \frac{\|\Delta \mathbf{d}\|}{\| \mathbf{d} \|}, \quad (1.16)$$

przy czym przybliżenie jest tym lepsze, im błędy danych są mniejsze. Jeśli zadania nie potrafimy opisać przez sformułowanie jego jawnej postaci funkcyjnej, to na ogół trudno jest wyznaczyć wskaźnik uwarunkowania bezpośrednio z definicji (1.7). Wówczas można przyjmować za wskaźnik uwarunkowania *najmniejszą z możliwych do wyznaczenia wartości* $\mathrm{cond}_\phi(\mathbf{d})$, dla której dla wszystkich możliwych i małych odchyłek danych zachodzi relacja (por. (1.16))

$$\frac{\|\Delta \mathbf{w}\|}{\|\mathbf{w}\|} \leq \mathrm{cond}_\phi(\mathbf{d}) \cdot \frac{\|\Delta \mathbf{d}\|}{\| \mathbf{d} \|}. \quad (1.17)$$

Jeśli błędy danych są jedynie *błędami reprezentacji maszynowej liczb*, to biorąc pod uwagę (1.15), zależność (1.17) można zapisać w postaci

$$\frac{\|\Delta \mathbf{w}\|}{\|\mathbf{w}\|} \leq \mathrm{cond}_\phi(\mathbf{d}) \cdot eps. \quad (1.18)$$

Omówiony sposób postępowania pokażemy na przykładzie zadania iloczynu skalarnego, dla którego wyznaczyliśmy wskaźnik uwarunkowania (1.8) w przykładzie 1.3.

Przykład 1.6. Niech $\phi(\mathbf{a}, \mathbf{b}) = f(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{n} a_i b_i$. Dane zadania są z błędami, tzn. $\mathrm{rd}(a_i) = a_i(1+\alpha_i)$, $\mathrm{rd}(b_i) = b_i(1+\beta_i)$, gdzie α_i i β_i są wartościami błędów względnych reprezentacji, $|\alpha_i| \leq eps$ i $|\beta_i| \leq eps$. Wyznaczymy oszacowanie od

góry wskaźnika uwarunkowania, dokonując kolejnych szacowań i pomijając składniki z błędami względnymi reprezentacji występującymi nieliniowo, w iloczynach – stosując wówczas symbol „$\underset{1}{=}$” wprowadzony uprzednio w przykładzie 1.1:

$$\begin{aligned}
\frac{\|\Delta w\|}{\|w\|} &= \frac{|\Delta w|}{|w|} = \frac{\left|\sum_{i=1}^{n} a_i\,(1+\alpha_i)\cdot b_i\,(1+\beta_i) - \sum_{i=1}^{n} a_i b_i\right|}{\left|\sum_{i=1}^{n} a_i b_i\right|} \\
&= \frac{\left|\sum_{i=1}^{n} (a_i b_i \alpha_i + a_i b_i \beta_i + a_i b_i \alpha_i \beta_i)\right|}{\left|\sum_{i=1}^{n} a_i b_i\right|} \underset{1}{=} \frac{\left|\sum_{i=1}^{n} (a_i b_i \alpha_i + a_i b_i \beta_i)\right|}{\left|\sum_{i=1}^{n} a_i b_i\right|} \\
&\leq \frac{\sum_{i=1}^{n} |a_i b_i\,(\alpha_i+\beta_i)|}{\left|\sum_{i=1}^{n} a_i b_i\right|} = \frac{\sum_{i=1}^{n} |a_i b_i|\cdot|\alpha_i+\beta_i|}{\left|\sum_{i=1}^{n} a_i b_i\right|} \\
&\leq \frac{\sum_{i=1}^{n} |a_i b_i|\cdot 2eps}{\left|\sum_{i=1}^{n} a_i b_i\right|} = \mathrm{cond}_f^s\,(\mathbf{a},\mathbf{b})\cdot eps, \qquad (1.19)
\end{aligned}$$

gdzie

$$\mathrm{cond}_f^s\,(\mathbf{d}) = \mathrm{cond}_f^s\,(\mathbf{a},\mathbf{b}) = \frac{2\sum_{i=1}^{n} |a_i b_i|}{\left|\sum_{i=1}^{n} a_i b_i\right|}, \qquad (1.20)$$

zaś indeks górny „s” dodano dla odróżnienia od (1.9) i podkreślenia, że jest to szacowany wskaźnik uwarunkowania. Wartość eps w (1.19) może być traktowana jako ograniczenie górne względnego błędu wektora danych, błędu określonego jako

$$\max\{|\alpha_1|, ..., |\alpha_n|, |\beta_1|, ..., |\beta_n|\},$$

dla wektora danych $\mathbf{d} = [\mathbf{a}^{\mathrm{T}}\ \mathbf{b}^{\mathrm{T}}]^{\mathrm{T}}$.

□

Trzeba podkreślić, że wskaźnik uwarunkowania opisuje maksymalny możliwy, względny błąd rozwiązania powodowany *jedynie przez błędy danych*, tzn. przy wszystkich operacjach arytmetycznych prowadzących od danych do wyniku wykonywanych bez błędów – przy dokładnym rozwiązaniu zadania z zaburzonymi danymi.

1.4. Algorytm i jego numeryczne realizacje

Rozważymy trzy *podstawowe i różne* pojęcia:

- **Zadanie** obliczeniowe (matematyczne): $\mathbf{w} = \phi(\mathbf{d})$;
- **Algorytm** $A(\mathbf{d})$ obliczenia wyniku zadania $\phi(\mathbf{d})$, tj. sposób wyznaczenia wyniku zgodnie z jednoznacznie określoną kolejnością wykonywania elementarnych działań arytmetycznych;
- **Numeryczna realizacja** $fl(A(\mathbf{d}))$ **algorytmu** $A(\mathbf{d})$, polegająca na:
  a) zastąpieniu liczb występujących w sformułowaniu $A(\mathbf{d})$ ich reprezentacjami zmiennopozycyjnymi,
  b) wykonaniu operacji arytmetycznych (działań elementarnych, obliczaniu wartości funkcji standardowych) w arytmetyce zmiennopozycyjnej „fl", tj. w sposób przybliżony.

Przykład 1.7. Rozważymy zadanie: $\phi(a,b) = a^2 - b^2$, przy czym zakładamy $\mathrm{rd}(a) = a$, $\mathrm{rd}(b) = b$ – tzn. zakładamy brak błędów reprezentacji danych, ponieważ interesuje nas wpływ jedynie błędów elementarnych operacji arytmetycznych. Rozważymy dwa algorytmy:

Algorytm $A1(a,b)$: $a \cdot a - b \cdot b$,

Algorytm $A2(a,b)$: $(a+b)\cdot(a-b)$.

Realizacja numeryczna algorytmu $A1(a,b)$:

$$\begin{aligned}
fl(A1(a,b)) &= fl(a\cdot a - b\cdot b)\\
&= fl[fl(a\cdot a) - fl(b\cdot b)]\\
&= [a^2(1+\varepsilon_1) - b^2(1+\varepsilon_2)](1+\varepsilon_3)\\
&= [a^2 - b^2 + a^2\varepsilon_1 - b^2\varepsilon_2](1+\varepsilon_3)\\
&= a^2 - b^2 + a^2\varepsilon_1 - b^2\varepsilon_2 + (a^2-b^2)\varepsilon_3 + a^2\varepsilon_1\varepsilon_3 - b^2\varepsilon_2\varepsilon_3\\
&\underset{1}{=} a^2 - b^2 + a^2\varepsilon_1 - b^2\varepsilon_2 + (a^2-b^2)\varepsilon_3\\
&= (a^2-b^2)(1 + \frac{a^2\varepsilon_1 - b^2\varepsilon_2}{a^2-b^2} + \varepsilon_3)\\
&= (a^2-b^2)(1+\delta_1),
\end{aligned}$$

gdzie

$$\delta_1 = \frac{a^2\varepsilon_1 - b^2\varepsilon_2}{a^2-b^2} + \varepsilon_3$$

jest błędem względnym wyniku,

$$|\delta_1| = \left|\frac{a^2\varepsilon_1 - b^2\varepsilon_2}{a^2-b^2} + \varepsilon_3\right| \le \frac{a^2+b^2}{|a^2-b^2|}eps + eps.$$

Realizacja numeryczna algorytmu $A2(a,b)$:

$$\begin{aligned} fl(A2(a,b) &= fl[(a+b)\cdot(a-b)] \\ &= fl[fl(a+b)\cdot fl(a-b)] \\ &= [(a+b)(1+\varepsilon_1)\cdot(a-b)(1+\varepsilon_2)](1+\varepsilon_3) \\ &= (a^2-b^2)(1+\varepsilon_1)(1+\varepsilon_2)(1+\varepsilon_3) \\ &\underset{1}{=} (a^2-b^2)(1+\varepsilon_1+\varepsilon_2+\varepsilon_3) \\ &= (a^2-b^2)(1+\delta_2), \end{aligned}$$

gdzie

$$\delta_2 = \varepsilon_1+\varepsilon_2+\varepsilon_3$$

jest błędem względnym wyniku,

$$|\delta_2| = |\varepsilon_1+\varepsilon_2+\varepsilon_3| \leq 3eps.$$

Widać wyraźnie, że algorytm $A1$ posiada gorsze własności numeryczne: gdy $a^2 \approx b^2$, może dojść do silnego (o wiele rzędów wielkości) wzrostu błędu względnego δ_1, podczas gdy uzyskane oszacowanie błędu względnego δ_2 algorytmu $A2$ jest niewrażliwe na wartości danych a i b. □

1.5. Stabilność numeryczna algorytmów

Rozważmy ponownie matematyczne zadanie obliczeniowe $\phi(\cdot),\ \mathbf{w} = \phi(\mathbf{d})$, oraz względny błąd danych jako oszacowany przez eps, $\frac{\|\Delta\mathbf{d}\|}{\|\mathbf{d}\|} \leq eps$, zob. (1.15). Zgodnie z definicją wskaźnika uwarunkowania (1.16) mamy wówczas

$$\frac{\| \phi(\mathbf{d}+\Delta\mathbf{d}) - \phi(\mathbf{d}) \|}{\|\phi(\mathbf{d})\|} \leq \mathrm{cond}_\phi(\mathbf{d}) \cdot eps,$$

gdzie $\mathbf{w} = \phi(\mathbf{d})$ jest *dokładnym* wynikiem matematycznym odpowiadającym dokładnym (nie zaburzonym) danym $\mathbf{d}$, natomiast $\phi(\mathbf{d}+\Delta\mathbf{d})$ jest *dokładnym* wynikiem matematycznym odpowiadającym zaburzonym wartościom danych $\mathbf{d}+\Delta\mathbf{d}$ – w rozważanym przypadku błędy numeryczne operacji arytmetycznych związanych z numeryczną realizacją $fl(A(\mathbf{d}))$ algorytmu $A(\mathbf{d})$ obliczającego wynik zadania $\phi(\mathbf{d})$ nie są brane pod uwagę. Rozważenie wpływu tych błędów prowadzi do definicji pojęcia numerycznej stabilności algorytmów.

Definicja. Algorytm $A(\mathbf{d})$ realizujący zadanie $\phi(\mathbf{d})$ nazywamy *numerycznie stabilnym*, jeśli istnieje taka stała dodatnia K_s, że dla każdych danych $\mathbf{d} \in D$ (D jest

zbiorem danych, dla których zadanie jest dobrze postawione) i dla każdego dostatecznie małego eps (tj. dostatecznie silnej arytmetyki) zachodzi nierówność:

$$\frac{\|fl\left(A\left(\mathbf{d}\right)\right)-\phi\left(\mathbf{d}\right)\|}{\|\phi\left(\mathbf{d}\right)\|}\leq K_s\cdot\mathrm{cond}_\phi(\mathbf{d})\cdot eps, \tag{1.21}$$

gdzie stałą K_s nazywamy *wskaźnikiem stabilności*.

Stąd, algorytm numerycznie stabilny to taki algorytm, który dla dowolnych danych (z zakresu D) gwarantuje uzyskanie rozwiązania z *błędem względnym ograniczonym*, co najwyżej K_s razy większym od błędu względnego powodowanego jedynie zaburzeniami danych.

Z zależności (1.21) wynika, że całkowity błąd względny praktycznie uzyskiwanego rozwiązania algorytmem numerycznie stabilnym zależy od:

– uwarunkowania zadania dla aktualnych wartości danych ($\mathrm{cond}_\phi(\mathbf{d})$),
– stosowanej arytmetyki zmiennopozycyjnej (eps),
– wskaźnika stabilności numerycznej algorytmu (K_s), tj. jakości algorytmu.

Ponadto, z definicji algorytmu numerycznie stabilnego wynika natychmiast

$$\lim_{eps\to 0}\frac{\|\ fl\left(A\left(\mathbf{d}\right)\right)-\phi\left(\mathbf{d}\right)\ \|}{\|\phi\left(\mathbf{d}\right)\|}=0. \tag{1.22}$$

Zależność powyższa bywa bezpośrednio podawana jako *definicja numerycznej stabilności*, zob. [2].

Komentarz*. Wielkość

$$P\left(\mathbf{d},\phi\right)=\mathrm{cond}_\phi(\mathbf{d})\cdot eps\cdot\|\mathbf{w}\|+eps\cdot\|\mathbf{w}\| \tag{1.23}$$

nazywana jest *błędem nieuniknionym* (też *optymalnym poziomem błędu* [3]). Jest to bowiem wielkość *niezależna od zastosowanego algorytmu* obliczania zadania $\phi(\mathbf{d})$. Błąd nieunikniony składa się z dwóch składników: pierwszego, odpowiadającego maksymalnemu możliwemu bezwzględnemu błędowi rozwiązania powodowanemu jedynie zaburzeniami danych i drugiego, będącego oszacowaniem maksymalnego błędu reprezentacji zmiennopozycyjnej wyniku. Błąd nieunikniony zależy od *uwarunkowania zadania* ($\mathrm{cond}_\phi\left(\mathbf{d}\right)$) *i od stosowanej arytmetyki zmiennopozycyjnej (*eps). Bezpośrednio z definicji błędu nieuniknionego (1.23) wynika relacja

$$\lim_{eps\to 0}P\left(\mathbf{d},\phi\right)=0.$$

Stabilność numeryczną algorytmu $A\left(\mathbf{d}\right)$ można teraz zdefiniować zależnością

$$\|fl\left(A\left(\mathbf{d}\right)\right)-\phi\left(\mathbf{d}\right)\|\leq K_s\cdot P\left(\mathbf{d},\phi\right),$$

*Materiał uzupełniający.

równoważną definicji uprzednio wprowadzonej. Algorytm numerycznie stabilny jest to więc algorytm wyliczający numerycznie wynik z błędem na poziomie co najwyżej K_s razy większym od błędu nieuniknionego (optymalnego poziomu błędu). □

Numeryczną stabilność algorytmów dowodzi się często tzw. *metodą pozornych równoważnych zaburzeń.* Polega ona na wykazaniu (jeśli to możliwe), że wynik obliczeń w arytmetyce zmiennopozycyjnej $fl(A(\mathbf{d}))$ jest *zaburzonym dokładnym rozwiązaniem* zadania o zaburzonych danych, tj.

$$fl\left(A\left(\mathbf{d}\right)\right) = \phi\left(\mathbf{d} + \mathbf{\Delta d}\right) \cdot \left(1 + \mathbf{e}\right),$$

gdzie

$$\frac{|\Delta d_i|}{|d_i|} \leq k_i eps,$$

$$|e_j| \leq k_j eps,$$

zaś k_i i k_j są stałymi. Algorytmy spełniające powyższą równość nazywamy *numerycznie poprawnymi* [1]. *Każdy algorytm numerycznie poprawny jest numerycznie stabilny.*

Efektywność algorytmu oceniamy liczbą elementarnych operacji arytmetycznych potrzebnych do uzyskania wyniku, w szczególności *liczbą mnożeń (i dzieleń)* M oraz *dodawań (i odejmowań)* D. Podanie obu tych liczb jest informacją bardziej precyzyjną, w porównaniu do oceny efektywności algorytmu jedynie przez całkowitą liczbę działań zmiennoprzecinkowych (*floating point operations*) potrzebnych do wykonania algorytmu, co jest oceną rownież spotykaną.

Zadania

1. Narysuj zbiory punktów spełniające: $||\mathbf{x}||_1 \leq 1$, $||\mathbf{x}||_2 \leq 1$, $||\mathbf{x}||_\infty \leq 1$, dla $\mathbf{x} \in \mathbb{R}^2$.

2. Oblicz normy pierwszą, nieskończoność i Frobeniusa macierzy

$$\mathbf{A} = \begin{bmatrix} 1 & 4 & 3 \\ 2 & 2 & 1 \\ 1 & 3 & 2 \end{bmatrix}.$$

3. Zaproponuj algorytm obliczeniowy (tj. kolejność działań) dla zadania

$$\phi\left(\mathbf{x}\right) = x_1 + x_2 + x_3, \qquad \text{gdzie } x_1 > x_2 > x_3 > 0,$$

tak, aby uzyskać jak najlepsze oszacowanie wpływu błędów zaokrągleń.

4. Oblicz wskaźnik uwarunkowania funkcji $f(a,b) = a^2 - b^2$.

5. Wyznacz oszacowanie błędu wyniku, tj. maksymalny moduł błędu względnego, powstającego przy realizacji w arytmetyce zmiennopozycyjnej algorytmów:
$$A1(a,b) = (a - a \cdot b) \cdot (a + 2 \cdot b),$$
$$A2(a,b) = a \cdot (1 - b) \cdot (a + 2 \cdot b),$$
zakładając zerowe błędy reprezentacji danych: $\mathrm{rd}(a) = a$, $\mathrm{rd}(b) = b$.

6. Wyznacz oszacowanie błędu wyniku, tj. maksymalny moduł błędu względnego, powstającego przy realizacji w arytmetyce zmiennopozycyjnej algorytmu
$$A(a,b) = (a + b) \cdot (a - b),$$
zakładając niezerowe błędy reprezentacji danych: $\mathrm{rd}(a) = a(1 + \varepsilon_a)$, $\mathrm{rd}(b) = b(1 + \varepsilon_b)$. Porównaj uzyskany rezultat z uzyskanym w przykładzie 1.7, zinterpretuj.

7. Wyznacz oszacowanie błędu wyniku, tj. maksymalny moduł błędu względnego, powstającego przy obliczaniu w arytmetyce zmiennopozycyjnej aproksymacji pochodnej funkcji $f(x)$ ilorazem różnicowym opartym na różnicy skończonej
$$\frac{f(x+h) - f(x)}{h}$$
Uwaga: można przyjąć, żę krok h jest potęgą dwójki, ale nie wolno zaniedbać błędów reprezentacji liczb $f(x)$ i $f(x+h)$. Błędy te (względne) należy przyjąć w postaci:
$fl(f(x)) = f(x)(1 + \varepsilon_1),\ |\varepsilon_1| \leq Eps,$
$fl(f(x+h)) = f(x+h)(1 + \varepsilon_2), |\varepsilon_2| \leq Eps, \quad (Eps \geq eps).$

8. Porównać jakość numeryczną (tj. oszacowania maksymalnego błędu względnego) i efektywność dwóch algorytmów obliczania wartości wielomianu $w(x)$,
$$w(x) = x^3 + a_1 \cdot x^2 + a_2 \cdot x + a_3,$$
A1: $x \cdot x \cdot x + a_1 \cdot x \cdot x + a_2 \cdot x + a_3$,
A2: $x \cdot [x \cdot (x + a_1) + a_2] + a_3$ (schemat Hornera).

9.* Wyznaczyć oszacowanie dla $|E_i|$, jeśli
$$fl\left(\sum_{i=1}^{n} a_i b_i\right) = \sum_{i=1}^{n} a_i b_i \left(1 + E_i\right),$$
oraz $\mathrm{rd}(a_\mathrm{i}) = a_\mathrm{i}$, $\mathrm{rd}(b_\mathrm{i}) = b_\mathrm{i}$, $i = 1, 2, ..., n$.

*Zadanie dodatkowe.

Rozdział 2

Układ równań liniowych, rozkłady trójkątne macierzy

2.1. Normy wektorów i macierzy

Przypomnijmy aksjomaty normy $||\cdot||$ ($\mathbb{V}$ – przestrzeń liniowa, $\mathbb{K}$ - ciało liczb rzeczywistych $\mathbb{R}$ lub zespolonych $\mathbb{C}$):

1. $\|\mathbf{x}\| \geq 0, \quad \|\mathbf{x}\| = 0 \Leftrightarrow \mathbf{x} = \mathbf{0}, \quad \forall\, \mathbf{x} \in \mathbb{V},$
2. $\|\alpha\mathbf{x}\| = |\alpha|\,\|\mathbf{x}\|, \quad \forall\, \alpha \in \mathbb{K}, \quad \forall\, \mathbf{x} \in \mathbb{V},$
3. $\|\mathbf{x}+\mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|, \quad \forall\, \mathbf{x}, \mathbf{y} \in \mathbb{V}.$

Normy wektorów

Niech elementami przestrzeni liniowej będą wektory liczbowe, tj. $\mathbb{V} = \mathbb{R}^n$. Normami Höldera wektorów nazywamy normy postaci:

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}}, p = 1, 2, 3, ...$$

Szczególne znaczenie mają normy:

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i| \quad - \text{ norma pierwsza,} \tag{2.1}$$

$$\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2} \quad - \text{norma euklidesowa,} \tag{2.2}$$

$$\|\mathbf{x}\|_\infty = \max_{1\leq i\leq n} |x_i| \quad - \text{norma maksimum.} \tag{2.3}$$

Normy te są równoważne, tzn.

$$\forall\, \mathbf{x} \in \mathbb{R}^n \quad \exists\, \alpha, \beta \in \mathbb{R} \quad \alpha\|\mathbf{x}\|_a \leq \|\mathbf{x}\|_b \leq \beta\|\mathbf{x}\|_a\,.$$

Na przykład:

$$\|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_1 \leq n\|\mathbf{x}\|_\infty\,,$$

$$\|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_2 \leq \sqrt{n}\,\|\mathbf{x}\|_\infty\,,$$
$$\frac{1}{\sqrt{n}}\,\|\mathbf{x}\|_1 \leq \|\mathbf{x}\|_2 \leq \|\mathbf{x}\|_1\,.$$

Normy macierzy

Macierze można traktować jako operatory liniowe. Zbiór wszystkich macierzy $\mathbf{A}$ o wymiarach $m \times n$,

$$\mathbb{R}^n \ni \mathbf{x} \to \mathbf{A}\mathbf{x} = \mathbf{y} \in \mathbb{R}^m,$$

tworzy przestrzeń liniową. Przestrzeń tę oznaczamy przez $\mathbb{V} = \mathbb{L}\,(\mathbb{R}^n, \mathbb{R}^m)$. Aksjomaty normy zapisane dla elementów tej przestrzeni:

1. $\|\mathbf{A}\| \geq 0, \quad \|\mathbf{A}\| = 0 \Leftrightarrow \mathbf{A} = \mathbf{0}$,
2. $\|\alpha\mathbf{A}\| \leq |\alpha|\,\|\mathbf{A}\|\,, \quad \forall \alpha \in \mathbb{K}, \quad \forall \mathbf{A} \in \mathbb{L}\,(\mathbb{R}^n, \mathbb{R}^m)\,,$
3. $\|\mathbf{A} + \mathbf{B}\| \leq \|\mathbf{A}\| + \|\mathbf{B}\|\,, \quad \forall \mathbf{A}, \mathbf{B} \in \mathbb{L}\,(\mathbb{R}^n, \mathbb{R}^m)\,.$

Normę macierzy nazywamy *indukowaną* przez normę wektora, jeśli

$$\|\mathbf{A}\| = \sup_{\mathbf{x}\neq 0} \frac{\|\mathbf{A}\mathbf{x}\|}{\|\mathbf{x}\|} \quad (\text{lub:} \quad \|\mathbf{A}\| = \sup_{\{\mathbf{x}:\|\mathbf{x}\|=1\}} \|\mathbf{A}\mathbf{x}\|\,) \tag{2.4}$$

Dla norm indukowanych prawdziwa jest zależność:

$$\|\mathbf{A}\mathbf{B}\| \leq \|\mathbf{A}\|\,\|\mathbf{B}\|\,, \quad \forall \mathbf{A}, \mathbf{B} \in \mathbb{L}\,(\mathbb{R}^n, \mathbb{R}^m)\,, \tag{2.5}$$

gdyż

$$\begin{aligned}\|\mathbf{A}\mathbf{B}\| &= \sup_{\mathbf{x}\neq 0} \frac{\|\mathbf{A}\mathbf{B}\mathbf{x}\|}{\|\mathbf{x}\|} = \sup_{\mathbf{x}\neq 0} \frac{\|\mathbf{A}\mathbf{B}\mathbf{x}\|}{\|\mathbf{B}\mathbf{x}\|} \cdot \frac{\|\mathbf{B}\mathbf{x}\|}{\|\mathbf{x}\|} \\ &\leq \sup_{\mathbf{y}\neq 0} \frac{\|\mathbf{A}\mathbf{y}\|}{\|\mathbf{y}\|} \cdot \sup_{\mathbf{x}\neq 0} \frac{\|\mathbf{B}\mathbf{x}\|}{\|\mathbf{x}\|} = \|\mathbf{A}\|\,\|\mathbf{B}\|\,.\end{aligned}$$

Normy indukowane są *normami zgodnymi*, tzn.

$$\|\mathbf{A}\mathbf{x}\| \leq \|\mathbf{A}\|\,\|\mathbf{x}\|\,, \quad \forall \mathbf{x} \in \mathbb{R}^n \quad \forall \mathbf{A} \in \mathbb{L}\,(\mathbb{R}^n, \mathbb{R}^m)\,. \tag{2.6}$$

Najważniejsze normy indukowane macierzy to:

$$\|\mathbf{A}\|_1 = \max_{1\leq j\leq n} \sum_{i=1}^{n} |a_{ij}| \quad - \text{ norma pierwsza,} \tag{2.7}$$

$$\|\mathbf{A}\|_2 = \max_{\lambda \in \mathrm{sp}(\mathbf{A}^{\mathrm{T}}\mathbf{A})} \sqrt{\lambda} \quad - \text{ norma spektralna (druga),} \tag{2.8}$$

$$\|\mathbf{A}\|_\infty = \max_{1\leq i\leq n} \sum_{j=1}^{n} |a_{ij}| \quad - \text{ norma maksimum,} \tag{2.9}$$

gdzie sp($\mathbf{A}^{\mathrm{T}}\mathbf{A}$) oznacza widmo (spektrum) macierzy $\mathbf{A}^{\mathrm{T}}\mathbf{A}$, tzn. zbiór wszystkich jej wartości własnych.

Na podstawie definicji normy indukowanej można łatwo wyprowadzić wzory na macierzowe normy pierwszą i maksimum, np. dla normy maksimum:

$$\begin{aligned}
\|\mathbf{A}\mathbf{x}\|_\infty &= \max_{1\le i\le m}\left|\sum_{j=1}^{n} a_{ij}x_j\right| \le \max_{1\le i\le m}\sum_{j=1}^{n}|a_{ij}x_j| \\
&= \max_{1\le i\le m}\sum_{j=1}^{n}|a_{ij}||x_j| \le \max_{1\le i\le m}\sum_{j=1}^{n}\left(|a_{ij}|\max_{1\le j\le n}|x_j|\right) \\
&= \left(\max_{1\le i\le m}\sum_{j=1}^{n}|a_{ij}|\right)\max_{1\le j\le n}|x_j| = \|\mathbf{A}\|_\infty\|\mathbf{x}\|_\infty
\end{aligned}$$

Norma Frobeniusa (norma euklidesowa) macierzy definiowana jest w postaci

$$\|\mathbf{A}\|_F \stackrel{\mathrm{df}}{=} \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}\,. \tag{2.10}$$

Nie jest ona indukowana przez żadną normę (gdyż $\|\mathbf{I}\|_F = \sqrt{n}$, a dla wszystkich norm indukowanych $\|\mathbf{I}\| = 1$). Jest jednak zgodna z normą euklidesową (wektorową), ponieważ

$$\|\mathbf{A}\|_2 \le \|\mathbf{A}\|_F \le \sqrt{\min(m,n)}\cdot\|\mathbf{A}\|_2 \Rightarrow \|\mathbf{A}\mathbf{x}\|_2 \le \|\mathbf{A}\|_F\|\mathbf{x}\|_2\,.$$

Promień spektralny (*spectral radius*) macierzy definiuje się jako największy z modułów wartości własnych kwadratowej macierzy $\mathbf{A}$:

$$\mathrm{sr}(\mathbf{A}) \stackrel{\mathrm{df}}{=} \max_{\lambda\in\mathrm{sp}(\mathbf{A})}|\lambda|\,. \tag{2.11}$$

Dla dowolnej (zgodnej) normy macierzy $\mathbf{A}$ mamy

$$\mathrm{sr}(\mathbf{A}) \le \|\mathbf{A}\|\,, \tag{2.12}$$

albowiem

$$\|\mathbf{A}\|\,\|\mathbf{v}\| \ge \|\mathbf{A}\mathbf{v}\| = \|\lambda\mathbf{v}\| = |\lambda|\,\|\mathbf{v}\|\,,$$

czyli

$$\forall\lambda\in\mathrm{sp}(\mathbf{A}) \quad |\lambda| \le \|\mathbf{A}\|\,.$$

2.2. Uwarunkowanie macierzy, układu równań liniowych

Rozważamy układ równań liniowych postaci $\mathbf{A}\mathbf{x} = \mathbf{b}$, gdzie:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ a_{31} & a_{32} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}, \quad a_{ij}, b_i \in \mathbb{R},$$

oraz macierz $\mathbf{A}$ jest nieosobliwa.

a) Przypadek zaburzenia wektora $\mathbf{b}$, czyli $\mathbf{b} + \delta\mathbf{b} \Rightarrow \mathbf{x} + \delta\mathbf{x}$. Mamy

$$\begin{aligned} \mathbf{A}\mathbf{x} &= \mathbf{b}, \\ \mathbf{A}\left(\mathbf{x} + \delta\mathbf{x}\right) &= \left(\mathbf{b} + \delta\mathbf{b}\right), \\ \mathbf{A}\mathbf{x} + \mathbf{A}\,\delta\mathbf{x} &= \mathbf{b} + \delta\mathbf{b}, \\ \delta\mathbf{x} &= \mathbf{A}^{-1}\delta\mathbf{b}. \end{aligned}$$

Dla dowolnych norm zgodnych, mamy

$$\|\delta\mathbf{x}\| \leq \left\|\mathbf{A}^{-1}\right\| \|\delta\mathbf{b}\|,$$

z kolei

$$\|\mathbf{b}\| = \|\mathbf{A}\mathbf{x}\| \leq \|\mathbf{A}\| \|\mathbf{x}\| \quad \Rightarrow \quad \|\mathbf{x}\| \geq \frac{\|\mathbf{b}\|}{\|\mathbf{A}\|}.$$

Po podzieleniu dwóch powyższych nierówności stronami, dostajemy

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \left\|\mathbf{A}^{-1}\right\| \|\mathbf{A}\| \cdot \frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}, \tag{2.13}$$

czyli

$$\operatorname{cond}\left(\mathbf{A}\right) = \left\|\mathbf{A}^{-1}\right\| \|\mathbf{A}\|. \tag{2.14}$$

Podane powyższym wzorem wyrażenie $\operatorname{cond}(\mathbf{A})$ nazywane jest *wskaźnikiem uwarunkowania macierzy* $\mathbf{A}$ – dokładnie jest to wskaźnik uwarunkowania rozwiązania układu równań liniowych względem zaburzeń wektora prawej strony układu.

Zauważmy, że dla norm indukowanych Höldera

$$\operatorname{cond}(\mathbf{A})_p = \left\|\mathbf{A}^{-1}\right\|_p \|\mathbf{A}\|_p \geq 1,$$

gdyż

$$1 = \|\mathbf{I}\|_p = \left\|\mathbf{A}^{-1}\mathbf{A}\right\|_p \leq \left\|\mathbf{A}^{-1}\right\|_p \|\mathbf{A}\|_p.$$

Wskaźnik uwarunowania może przyjmować bardzo duże wartości, np. dla macierzy Hilberta $\mathbf{H}_n = (h_{ij})$,

$$h_{ij} = \frac{1}{i+j-1}, \quad i,j = 1,2,...,n$$

oraz normy drugiej mamy: $\mathrm{cond}(\mathbf{H}_6)_2 = 1.5 \cdot 10^7$,

$$\mathrm{cond}(\mathbf{H}_{10})_2 = 1.6 \cdot 10^{13}.$$

b) Przypadek zaburzenia elementów macierzy $\mathbf{A}$, $\mathbf{A} + \delta\mathbf{A} \Rightarrow \mathbf{x} + \delta\mathbf{x}$ (zakładamy, że $\det(\mathbf{A} + \delta\mathbf{A}) \neq 0$, macierz zaburzona jest też nieosobliwa, istnieje jednoznaczne rozwiązanie zaburzonego układu równań). Mamy

$$(\mathbf{A} + \delta\mathbf{A})(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b}.$$

Stąd

$$\mathbf{A}\cdot\mathbf{x} + \mathbf{A}\cdot\delta\mathbf{x} + \delta\mathbf{A}\cdot\mathbf{x} + \delta\mathbf{A}\cdot\delta\mathbf{x} = \mathbf{b},$$

$$\mathbf{A}\mathbf{x} = \mathbf{b} \;\Rightarrow\; \mathbf{A}\cdot\delta\mathbf{x} + \delta\mathbf{A}\cdot\mathbf{x} + \delta\mathbf{A}\cdot\delta\mathbf{x} = 0,$$

$$\delta\mathbf{x} = \mathbf{A}^{-1}(\delta\mathbf{A}\cdot\mathbf{x} + \delta\mathbf{A}\cdot\delta\mathbf{x}).$$

Wówczas, dla każdej normy macierzy zgodnej z normą wektora:

$$\|\delta\mathbf{x}\| \leq \left\|\mathbf{A}^{-1}\right\| (\|\delta\mathbf{A}\|\,\|\mathbf{x}\| + \|\delta\mathbf{A}\|\,\|\delta\mathbf{x}\|),$$

$$\left(1 - \left\|\mathbf{A}^{-1}\right\|\|\delta\mathbf{A}\|\right)\|\delta\mathbf{x}\| \leq \left\|\mathbf{A}^{-1}\right\|\|\delta\mathbf{A}\|\,\|\mathbf{x}\|.$$

Przyjmując, że $\left\|\mathbf{A}^{-1}\right\|\|\delta\mathbf{A}\| < 1$, dostajemy

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \frac{\left\|\mathbf{A}^{-1}\right\|\|\mathbf{A}\|\frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|}}{1 - \|\mathbf{A}^{-1}\|\,\|\mathbf{A}\|\frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|}},$$

czyli

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \frac{\mathrm{cond}(\mathbf{A})\cdot\frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|}}{1 - \mathrm{cond}(\mathbf{A})\cdot\frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|}}. \tag{2.15}$$

Zauważmy, że dla małych zaburzeń współczynników macierzy $\mathbf{A}$ (np. na poziomie reprezentacji maszynowej) i dla umiarkowanych wartości współczynnika uwarunkowania $\mathrm{cond}(\mathbf{A})$ mamy $\mathrm{cond}(\mathbf{A})\frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|} \ll 1$, tzn. ostatnie równanie można przybliżyć równaniem $\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \mathrm{cond}(\mathbf{A})\frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|}$.

2.3. Eliminacja Gaussa, rozkład LU

Metody rozwiązywania układów równań liniowych można podzielić na dwie podstawowe grupy:

1. *Metody skończone* – wynik otrzymujemy po skończonej, ściśle określonej liczbie przekształceń zależnej od wymiarowości zadania (np. metoda eliminacji Gaussa, rozkładu LL^{T}),
2. *Metody iteracyjne* – startując z przybliżenia początkowego rozwiązania (znanego, założonego), w kolejnych krokach przybliżenie to jest poprawiane, zbiega do rozwiązania. Nieznana jest liczba kroków, zależna od wymaganej dokładności rozwiązania.

2.3.1. Układ równań z macierzą trójkątną

Rozważmy układ równań o następującej *postaci trójkątnej* (górnej):

$$\begin{array}{ccccccccccccc}
a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1,n-1}x_{n-1} & + & a_{1n}x_n & = & b_1, \\
 & & a_{22}x_2 & + & \cdots & + & a_{2,n-1}x_{n-1} & + & a_{2n}x_n & = & b_2, \\
 & & & & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
 & & & & & & a_{n-1,n-1}x_{n-1} & + & a_{n-1,n}x_n & = & b_{n-1}, \\
 & & & & & & & & a_{nn}x_n & = & b_n.
\end{array} \tag{2.16}$$

Rozwiązanie tego układu zaczynamy od wiersza ostatniego i idąc wierszami „w górę", otrzymujemy:

$$\begin{aligned}
x_n &= \frac{b_n}{a_{nn}}, \\
x_{n-1} &= \frac{(b_{n-1} - a_{n-1,n}x_n)}{a_{n-1,n-1}}, \\
x_k &= \frac{\left(b_k - \sum\limits_{j=k+1}^{n} a_{kj}x_j\right)}{a_{kk}}, \qquad k = n-2, n-3, ..., 1.
\end{aligned} \tag{2.17}$$

Jeśli przez $\widetilde{\mathbf{x}}$ oznaczymy rozwiązanie numeryczne układu równań (uzyskane w arytmetyce zmiennopozycyjnej, tj. z błędami zaokrągleń operacji arytmetycznych), to można łatwo wyznaczyć oszacowanie:

$$(\mathbf{A} + \delta\mathbf{A})\,\widetilde{\mathbf{x}} = \mathbf{b},$$

gdzie

$$\begin{aligned}
&|\delta a_{kj}| \le (n - j + 2) \cdot eps \cdot |a_{kj}|, \quad k = 1, 2, ..., n-1, \quad j = k+1, ..., n, \\
&|\delta a_{kk}| \le 2 \cdot eps \cdot |a_{kk}|, \quad k = 1, 2, ..., n.
\end{aligned}$$

Stąd bezpośrednio wyliczyć też można następujące użyteczne oszacowanie:

$$\|\delta\mathbf{A}\|_\infty \le \left(\frac{1}{2}n^2 + \frac{n}{2} + 1\right) \cdot eps \cdot a,$$

gdzie a jest elementem macierzy $\mathbf{A}$ o największym module.

Łatwo sprawdzić, że przedstawiony algorytm wymaga $D = \frac{1}{2}n^2 - \frac{1}{2}n$ dodawań i odejmowań oraz $M = \frac{1}{2}n^2 + \frac{1}{2}n$ mnożeń i dzieleń, tzn. obu typów działań rzędu $\frac{1}{2}n^2$, co często oznaczamy skrótowo $D = O(\frac{1}{2}n^2)$, $M = O(\frac{1}{2}n^2)$.

2.3.2. Eliminacja Gaussa

Algorytm eliminacji Gaussa dzieli się na dwa etapy:

1. Eliminacja zmiennych – w wyniku przekształceń macierzy $\mathbf{A}$ i wektora $\mathbf{b}$ otrzymamy równoważny układ równań z macierzą trójkątną górną.
2. Postępowanie odwrotne (*back-substitution*) – stosujemy algorytm rozwiązania układu z macierzą trójkątną.

Etap eliminacji zmiennych

Wyjściowy układ równań (górny indeks „(k)" – układ równań przed k-tym krokiem metody):

$$\begin{array}{ccccccccc} a_{11}^{(1)}x_1 & + & a_{12}^{(1)}x_2 & + & \cdots & + & a_{1n}^{(1)}x_n & = & b_1^{(1)}, \\ a_{21}^{(1)}x_1 & + & a_{22}^{(1)}x_2 & + & \cdots & + & a_{2n}^{(1)}x_n & = & b_2^{(1)}, \\ \cdot & & \cdot & & \cdot & & \cdot & & \cdot \\ a_{n1}^{(1)}x_1 & + & a_{n2}^{(1)}x_2 & + & \cdots & + & a_{nn}^{(1)}x_n & = & b_n^{(1)}. \end{array}$$

Krok 1 – wyzerowanie elementów kolumny pierwszej, oprócz elementu w wierszu pierwszym. Wyznaczamy najpierw współczynniki

$$l_{i1} \stackrel{\text{df}}{=} \frac{a_{i1}^{(1)}}{a_{11}^{(1)}}, \qquad i = 2, 3, ..., n.$$

Pierwszy wiersz $\mathbf{w}_1$ mnożymy przez l_{i1} i odejmujemy od wiersza i-tego $\mathbf{w}_i$, kolejno dla $i = 2, 3, ..., n$:

$$\mathbf{w}_i = \mathbf{w}_i - l_{i1}\mathbf{w}_1 \quad \Longleftrightarrow \quad \begin{array}{ll} a_{ij}^{(2)} & = a_{ij}^{(1)} - l_{i1}a_{1j}^{(1)},\ j = 1, 2, ..., n, \\ b_i^{(2)} & = b_i^{(1)} - l_{i1}b_1^{(1)}, \qquad i = 2, 3, ..., n. \end{array}$$

Otrzymujemy

$$\begin{array}{ccccccccc} a_{11}^{(1)}x_1 & + & a_{12}^{(1)}x_2 & + & \cdots & + & a_{1n}^{(1)}x_n & = & b_1^{(1)}, \\ & & a_{22}^{(2)}x_2 & + & \cdots & + & a_{2n}^{(2)}x_n & = & b_2^{(2)}, \\ & & \cdot & & \cdot & & \cdot & & \cdot \\ & & a_{n2}^{(2)}x_2 & + & \cdots & + & a_{nn}^{(2)}x_n & = & b_n^{(2)}. \end{array}$$

$$\text{tzn.} \quad \mathbf{A}^{(2)}\mathbf{x} = \mathbf{b}^{(2)}.$$

Krok 2 – wyzerowanie elementów kolumny drugiej, z wyjątkiem elementów w wierszach 1 i 2, w sposób analogiczny do poprzedniego kroku. Obliczamy

$$l_{i2} \stackrel{\text{df}}{=} \frac{a_{i2}^{(2)}}{a_{22}^{(2)}}, \qquad i = 3, 4, ..., n.$$

Drugi wiersz $\mathbf{w}_2$ mnożymy kolejno przez l_{i2} i odejmujemy od i-tego wiersza $\mathbf{w}_i$, $i = 3, 4, ..., n$,

$$\mathbf{w}_i = \mathbf{w}_i - l_{i2}\mathbf{w}_2 \quad \Longleftrightarrow \quad \begin{array}{ll} a_{ij}^{(3)} & = a_{ij}^{(2)} - l_{i2}a_{2j}^{(2)}, \quad j = 2, 3, ..., n, \\ b_i^{(3)} & = b_i^{(2)} - l_{i2}b_2^{(2)}, \qquad\qquad i = 3, 4, ..., n, \end{array}$$

uzyskując układ równań

$$\mathbf{A}^{(3)}\mathbf{x} = \mathbf{b}^{(3)}.$$

Ogólnie, po $k-1$ **krokach** otrzymujemy układ równań:

$$\begin{array}{ccccccccccccc} a_{11}^{(1)}x_1 & + & a_{12}^{(1)}x_2 + & \cdots & + & a_{1k}^{(1)}x_k & + & \cdots & + & a_{1n}^{(1)}x_n & = & b_1^{(1)}, \\ & & a_{22}^{(2)}x_2 + & \cdots & + & a_{2k}^{(2)}x_k & + & \cdots & + & a_{2n}^{(2)}x_n & = & b_2^{(2)}, \\ & & & & & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & & & a_{kk}^{(k)}x_k & + & \cdots & + & a_{kn}^{(k)}x_n & = & b_k^{(k)}, \\ & & & & & a_{k+1,k}^{(k)}x_k & + & \cdots & + & a_{k+1,n}^{(k)}x_n & = & b_{k+1}^{(k)}, \\ & & & & & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & & & a_{nk}^{(k)}x_k & + & \cdots & + & a_{nn}^{(k)}x_n & = & b_n^{(k)}. \end{array}$$

W k-**tym kroku** eliminujemy za pomocą równania k-tego zmienną x_k z równań $k+1, k+2, ..., n$ – odejmując od każdego z nich, kolejno, równanie k-te pomnożone przez

$$l_{ik} \stackrel{\text{df}}{=} \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}, \qquad i = k+1, k+2, ..., n.$$

Tak więc przekształcenia w k-tym kroku dane sa zależnościami:

$$\mathbf{w}_i = \mathbf{w}_i - l_{ik}\mathbf{w}_k \quad \Longleftrightarrow \quad \begin{array}{ll} a_{ij}^{(k+1)} & = a_{ij}^{(k)} - l_{ik}a_{kj}^{(k)}, \quad j = k, k+1, ..., n, \\ b_i^{(k+1)} & = b_i^{(k)} - l_{ik}b_k^{(k)}, \qquad i = k+1, k+2, ..., n. \end{array}$$

W rezultacie, po $n-1$ krokach uzyskujemy układ równań

$$\mathbf{A}^{(n)}\mathbf{x} = \mathbf{b}^{(n)},$$

gdzie $\mathbf{A}^{(n)}$ to macierz trójkątna górna.

2.3.3. Rozkład LU macierzy

Opisane postępowanie metodą eliminacji Gaussa prowadzi do tzw. *rozkładu LU macierzy*, tj. do rozłożenia macierzy $\mathbf{A}$ na iloczyn dwóch macierzy: dolnej trójkątnej $\mathbf{L}$ (*lower triangular*) i górnej trójkątnej $\mathbf{A}^{(n)} = \mathbf{U}$ (*upper triangular*).

W każdym kroku eliminacji Gaussa dokonujemy jedynie kilku *przekształceń liniowych*, każde z nich odpowiada mnożeniu przez pewną macierz (operator liniowy). Łatwo sprawdzić, że krok 1 jest równoważny lewostronnemu pomnożeniu układu równań (tj. macierzy $\mathbf{A}$ i wektora $\mathbf{b}$) przez nieosobliwą macierz $\mathbf{L}^{(1)}$:

$$\mathbf{A}^{(1)}\mathbf{x} = \mathbf{b}^{(1)} \;\rightarrow\; \mathbf{A}^{(2)}\mathbf{x} = \mathbf{b}^{(2)} \;\equiv\; \mathbf{L}^{(1)}\mathbf{A}^{(1)}\mathbf{x} = \mathbf{L}^{(1)}\mathbf{b}^{(1)}, \text{ gdzie}$$

$$\mathbf{L}^{(1)} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ -l_{21} & 1 & 0 & \cdots & 0 \\ -l_{31} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -l_{n1} & 0 & 0 & \cdots & 1 \end{bmatrix}.$$

Ogólnie, krok k-ty jest równoważny pomnożeniu układu przez macierz $\mathbf{L}^{(k)}$:

$$\mathbf{L}^{(k)}\mathbf{A}^{(k)}\mathbf{x} = \mathbf{L}^{(k)}\mathbf{b}^{(k)}, \tag{2.18}$$

gdzie $\mathbf{L}^{(k)}$ różni się od macierzy diagonalnej jednostkowej $\mathbf{I}$ tylko k-tą kolumną, w której pod jedynką zamiast zer znajdują się elementy $-l_{ik}$, $i = k+1, ..., n$, tzn.

$$\mathbf{L}^{(k)} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 1 & & \vdots & \vdots & & 0 \\ & & \ddots & 0 & 0 & & \\ \vdots & & & 1 & 0 & & \vdots \\ & & & -l_{k+1,k} & 1 & & \\ & & & \vdots & & \ddots & 0 \\ 0 & 0 & \cdots & -l_{nk} & 0 & \cdots & 1 \end{bmatrix}. \tag{2.19}$$

Łatwo sprawdzić, że odwrócenie macierzy $\mathbf{L}^{(k)}$ powoduje jedynie zmianę znaku przy współczynnikach l_{ik}, tzn.

$$(\mathbf{L}^{(k)})^{-1} = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & \mathbf{0} & \\ & & 1 & & & \\ & & l_{k+1,k} & 1 & & \\ & \mathbf{0} & \vdots & & \ddots & \\ & & l_{nk} & & & 1 \end{bmatrix}. \tag{2.20}$$

W wyniku wszystkich kroków eliminacji Gaussa otrzymujemy

$$\mathbf{A}^{(n)} = \mathbf{L}^{(n-1)}\mathbf{L}^{(n-2)}\cdots\mathbf{L}^{(1)}\mathbf{A}^{(1)}, \tag{2.21}$$

$$\mathbf{b}^{(n)} = \mathbf{L}^{(n-1)}\mathbf{L}^{(n-2)}\cdots\mathbf{L}^{(1)}\mathbf{b}^{(1)}. \tag{2.22}$$

Definiując

$$\mathbf{U} \stackrel{\text{df}}{=} \mathbf{A}^{(n)} = [\mathbf{L}^{(n-1)}\mathbf{L}^{(n-2)}\cdots\mathbf{L}^{(1)}]\mathbf{A}^{(1)} = [\mathbf{L}^{(n-1)}\mathbf{L}^{(n-2)}\cdots\mathbf{L}^{(1)}]\mathbf{A},$$

mamy

$$[\mathbf{L}^{(n-1)}\mathbf{L}^{(n-2)}\cdots\mathbf{L}^{(1)}]^{-1}\mathbf{U} = \mathbf{A}.$$

Dalej, oznaczając

$$\mathbf{L} \stackrel{\text{df}}{=} [\mathbf{L}^{(n-1)}\mathbf{L}^{(n-2)}\cdots\mathbf{L}^{(1)}]^{-1} = \left(\mathbf{L}^{(1)}\right)^{-1}\cdots\left(\mathbf{L}^{(n-1)}\right)^{-1}, \tag{2.23}$$

dostajemy *rozkład LU*:

$$\mathbf{A} = \mathbf{L}\mathbf{U}.$$

Łatwo sprawdzić, że

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ l_{21} & 1 & \cdots & 0 \\ l_{31} & l_{32} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \cdots & 1 \end{bmatrix}. \tag{2.24}$$

Zauważmy, że z równości (2.22) bezpośrednio wynika

$$\mathbf{L}\mathbf{b}^{(n)} = \mathbf{b}, \tag{2.25}$$

tzn. przekształcenie w procedurze eliminacji Gaussa wektora $\mathbf{b}$ prawych stron układu równań odpowiada rozwiązaniu układu równań (2.26). Nakład obliczeń na rozkład LU: $D = O(\frac{1}{3}n^3),\ M = O(\frac{1}{3}n^3)$.

Jeśli dysponujemy jedynie rozkładem LU macierzy $\mathbf{A}$ (mamy jedynie oryginalny wektor prawych stron $\mathbf{b}$, nie mamy przekształconego wektora $\mathbf{b}^{(n)}$), to aby rozwiązać układ równań $\mathbf{L}\mathbf{U}\mathbf{x} = \mathbf{b}$, rozwiązujemy dwa równoważne mu układy równań z macierzami trójkątnymi (dolną i górną):

$$\mathbf{L}\mathbf{y} = \mathbf{b}, \text{ a następnie} \tag{2.26}$$

$$\mathbf{U}\mathbf{x} = \mathbf{y}. \tag{2.27}$$

Znając rozkład LU macierzy, trzeba więc w celu wyznaczenia wektora $\mathbf{x}$ rozwiązać dwa układy równań z macierzami trójkątnymi, tj. wykonać rzędu n^2 mnożeń i n^2

dodawań. Nakład obliczeń potrzebny do wyznaczenia macierzy $\mathbf{L}$ i $\mathbf{U}$ jest więc dominujący w algorytmie eliminacji Gaussa, gdyż jest rzędu $\frac{1}{3}n^3$ dodawań i mnożeń. Dlatego też rozkład LU jest bardzo przydatny, gdy układ równań rozwiązujemy wielokrotnie dla różnych wartości wektora $\mathbf{b}$.

Przykład 2.1. Rozwiążemy układ równań:

$$\begin{bmatrix} 3 & 1 & 6 \\ 2 & 1 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 7 \\ 4 \end{bmatrix}.$$

Pierwszy krok (element główny jest wytłuszczany):

$$\left[\begin{array}{ccc|c} \mathbf{3} & 1 & 6 & 2 \\ 2 & 1 & 3 & 7 \\ 1 & 1 & 1 & 4 \end{array}\right] \Rightarrow \begin{array}{l} l_{21} = \frac{a_{21}^{(1)}}{a_{11}^{(1)}} = \frac{2}{3}, \quad \mathbf{w}_2 = \mathbf{w}_2 - l_{21}\mathbf{w}_1 \\ l_{31} = \frac{a_{31}^{(1)}}{a_{11}^{(1)}} = \frac{1}{3}, \quad \mathbf{w}_3 = \mathbf{w}_3 - l_{31}\mathbf{w}_1 \end{array}$$

$$\Rightarrow \left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ 0 & \frac{1}{3} & -1 & \frac{17}{3} \\ 0 & \frac{2}{3} & -1 & \frac{10}{3} \end{array}\right].$$

Przed przystąpieniem do drugiego kroku wprowadzimy sposób zapisu, który przy implementacji komputerowej znacznie oszczędza pamięć, tym bardziej, im większa jest liczba równań n. Wykorzystamy fakt, że liczba pojawiających się w każdym kroku elementów macierzy $\mathbf{L}$ jest dokładnie równa liczbie zerowanych elementów macierzy $\mathbf{A}$. Dlatego, na początku kroku drugiego, możemy zapisać wyznaczone elementy macierzy $\mathbf{L}$ w miejsce zerowanych elementów macierzy $\mathbf{A}$:

$$\left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ \frac{2}{3} & \mathbf{\frac{1}{3}} & -1 & \frac{17}{3} \\ \frac{1}{3} & \frac{2}{3} & -1 & \frac{10}{3} \end{array}\right] \quad \Rightarrow \quad l_{32} = \frac{a_{32}^{(2)}}{a_{22}^{(2)}} = 2, \quad \mathbf{w}_3 = \mathbf{w}_3 - l_{32}\mathbf{w}_2 \quad \Rightarrow$$

$$\left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ \frac{2}{3} & \frac{1}{3} & -1 & \frac{17}{3} \\ \frac{1}{3} & 2 & 1 & -8 \end{array}\right], \quad \text{czyli: } L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{2}{3} & 1 & 0 \\ \frac{1}{3} & 2 & 1 \end{bmatrix}, U = \begin{bmatrix} 3 & 1 & 6 \\ 0 & \frac{1}{3} & -1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Mając wyznaczoną macierz $\mathbf{U} = \mathbf{A}^{(3)}$ i przekształcony wektor prawych stron $\mathbf{b}^{(3)} = [2 \ \frac{17}{3} \ -8]^{\mathrm{T}}$, wyliczamy rozwiązanie z układu równań

$$\begin{bmatrix} 3 & 1 & 6 \\ 0 & \frac{1}{3} & -1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ \frac{17}{3} \\ -8 \end{bmatrix},$$

uzyskując $\hat{\mathbf{x}} = [19 \ -7 \ -8]^{\mathrm{T}}$. □

2.3.4. Eliminacja Gaussa z wyborem elementu głównego

Wykonując przedstawiony algorytm eliminacji Gaussa, możemy spotkać blokującą obliczenia sytuację, gdy $a_{kk}^{(k)} = 0$. Unikamy tego, stosując algorytm eliminacji Gaussa *z wyborem elementu głównego*. Wybór ten może być częściowy lub pełny.

Eliminacja Gaussa z częściowym wyborem elementu głównego

Jeżeli na początku k-tego kroku mamy

$$\begin{array}{ccccccccccccc} a_{11}^{(1)}x_1 & + & a_{12}^{(1)}x_2 & + & \cdots & + & a_{1k}^{(1)}x_k & + & \cdots & + & a_{1n}^{(1)}x_n & = & b_1^{(1)}, \\ & & \cdot & & \cdot & & \cdot & & \cdot & & \cdot & & \cdot \\ & & & & & & a_{kk}^{(k)}x_k & + & \cdots & + & a_{kn}^{(k)}x_n & = & b_k^{(k)}, \\ & & & & & & \cdot & & \cdot & & \cdot & & \cdot \\ & & & & & & a_{ik}^{(k)}x_k & + & \cdots & + & a_{in}^{(k)}x_n & = & b_i^{(k)}, \\ & & & & & & \cdot & & \cdot & & \cdot & & \cdot \\ & & & & & & a_{nk}^{(k)}x_k & + & \cdots & + & a_{nn}^{(k)}x_n & = & b_n^{(k)}, \end{array}$$

to jako element główny wybieramy, spośród elementów $a_{jk}^{(k)}$ ($k \leq j \leq n$), element o największym module (oznaczony jako (ik)-ty), tzn.

$$\left|a_{ik}^{(k)}\right| = \max_j \left\{ \left|a_{jk}^{(k)}\right|, \quad j = k, k+1, ..., n \right\}. \tag{2.28}$$

Następnie zamieniamy wiersz i-ty z k-tym i stosujemy dalej algorytm standardowy eliminacji Gaussa dla k-tego kroku. Z nieosobliwości macierzy $\mathbf{A}$ wynika bezpośrednio, że zawsze $a_{ik}^{(k)} \neq 0$ – gdyby wszystkie elementy $a_{jk}^{(k)}$, $j = k, k+1, ..., n$, kolumny k-tej były zerowe, to macierz $\mathbf{A}$ byłaby osobliwa. Algorytm eliminacji Gaussa z wyborem elementu głównego stosujemy zawsze (*w każdym kroku*, bez względu na wartość elementu $a_{kk}^{(k)}$), gdyż *prowadzi to do mniejszych błędów numerycznych.*

Przestawienie (w kroku k-tym) wierszy k-tego z i-tym odpowiada pomnożeniu aktualnej macierzy przez macierz $\mathbf{P}^{(k)}$, różną od jednostkowej jedynie położeniem dwóch jedynek:

$$\mathbf{P}^{(k,i)} = \begin{bmatrix} 1 & 0 & & & \cdots & & & 0 & 0 \\ 0 & \ddots & & & & & & & 0 \\ & & 1 & & & & & & \\ & & & 0 & 0 \cdots & 0 & 1 & & \\ & & & 0 & 1 & & 0 & & \\ \vdots & & & \vdots & & \ddots & \vdots & & \vdots \\ & & & 0 & & & 1 & 0 & \\ & & & 1 & 0 \cdots & 0 & 0 & & \\ & & & & & & & 1 & \\ 0 & & & & & & & \ddots & 0 \\ 0 & 0 & & & \cdots & & & 0 & 1 \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \cdots \text{ wiersz } k \\ \\ \\ \\ \cdots \text{ wiersz } i \\ \\ \\ \\ \end{matrix}$$

$$\begin{matrix} & & \vdots & & \vdots & & \\ & & \text{kolumna } k & & \text{kolumna } i & & \end{matrix} \tag{2.29}$$

Pomnożenie macierzy $\mathbf{A}$ przez $\mathbf{P}^{(k,i)}$:
– *lewostronnie*, zamienia w $\mathbf{A}$ wiersz k-ty z i-tym,
– *prawostronnie*, zamienia w $\mathbf{A}$ kolumnę k-tą z i-tą.

Ponadto, macierz $\mathbf{P}^{(k,i)}$ ma następujące własności:

$$\begin{aligned} \det(\mathbf{P}^{(k,i)}) &= -1, \\ (\mathbf{P}^{(k,i)})^{-1} &= \mathbf{P}^{(k,i)}. \end{aligned}$$

Mamy więc następujący algorytm rozkładu LU z częściowym (kolumnowym) wyborem elementu głównego:

$$\mathbf{A}^{(n)} = (\mathbf{L}^{(n-1)}\mathbf{P}^{(n-1)}) \cdots (\mathbf{L}^{(2)}\mathbf{P}^{(2)})(\mathbf{L}^{(1)}\mathbf{P}^{(1)})\mathbf{A}^{(1)}. \tag{2.30}$$

Ze względu na zamiany wierszy, *finalny rozkład będzie dotyczył nie oryginalnej macierzy* $\mathbf{A}$, *ale macierzy* $\mathbf{PA}$, gdzie $\mathbf{P}$ to macierz wszystkich zamian wierszy, tzn.

$$\mathbf{LU} = \mathbf{PA}\ , \quad \text{gdzie } \mathbf{P} = \mathbf{P}^{(n-1)}\mathbf{P}^{(n-2)} \cdots \mathbf{P}^{(1)}. \tag{2.31}$$

Wykażemy zależność (2.31), wyprowadzając jednocześnie formułę opisującą macierz $\mathbf{L}$. Ponieważ $\mathbf{A}^{(n)} = \mathbf{U}$ i $\mathbf{A}^{(1)} = \mathbf{A}$, więc z (2.30) dla, przykładowo, $n = 4$ (macierz $\mathbf{A}$ wymiaru 4×4) mamy

$$\begin{aligned} \mathbf{U} &= (\mathbf{L}^{(3)}\mathbf{P}^{(3)})(\mathbf{L}^{(2)}\mathbf{P}^{(2)})(\mathbf{L}^{(1)}\mathbf{P}^{(1)})\mathbf{A} \\ &= \mathbf{L}^{(3)}\mathbf{P}^{(3)}\mathbf{L}^{(2)}[\mathbf{P}^{(3)}\mathbf{P}^{(3)}]\mathbf{P}^{(2)}\mathbf{L}^{(1)}[\mathbf{P}^{(2)}\mathbf{P}^{(3)}\mathbf{P}^{(3)}\mathbf{P}^{(2)}]\mathbf{P}^{(1)}\mathbf{A}, \end{aligned}$$

gdzie wstawione w nawiasach kwadratowych macierze to macierze jednostkowe. Przekształcając dalej, mamy

$$\begin{aligned}\mathbf{U} &= \mathbf{L}^{(3)}(\mathbf{P}^{(3)}\mathbf{L}^{(2)}\mathbf{P}^{(3)})(\mathbf{P}^{(3)}\mathbf{P}^{(2)}\mathbf{L}^{(1)}\mathbf{P}^{(2)}\mathbf{P}^{(3)})(\mathbf{P}^{(3)}\mathbf{P}^{(2)}\mathbf{P}^{(1)})\mathbf{A} \\ &= \tilde{\mathbf{L}}^{(3)}(\tilde{\mathbf{L}}^{(2)})(\tilde{\mathbf{L}}^{(1)})(\mathbf{P})\mathbf{A},\end{aligned}$$

skąd

$$\mathbf{L} = (\tilde{\mathbf{L}}^{(3)}\tilde{\mathbf{L}}^{(2)}\tilde{\mathbf{L}}^{(1)})^{-1} = (\tilde{\mathbf{L}}^{(1)})^{-1}(\tilde{\mathbf{L}}^{(2)})^{-1}(\tilde{\mathbf{L}}^{(3)})^{-1}. \tag{2.32}$$

Ogólnie

$$\tilde{\mathbf{L}}^{(k)} = \mathbf{P}^{(n-1)} \ \cdots \ \mathbf{P}^{(k+2)}\mathbf{P}^{(k+1)}\mathbf{L}^{(k)}\mathbf{P}^{(k+1)}\mathbf{P}^{(k+2)} \ \cdots \ \mathbf{P}^{(n-1)}, \tag{2.33}$$

$$(\tilde{\mathbf{L}}^{(k)})^{-1} = \mathbf{P}^{(n-1)} \ \cdots \ \mathbf{P}^{(k+2)}\mathbf{P}^{(k+1)}(\mathbf{L}^{(k)})^{-1}\mathbf{P}^{(k+1)}\mathbf{P}^{(k+2)} \ \cdots \ \mathbf{P}^{(n-1)}, \tag{2.34}$$

gdzie $\mathbf{L}^{(k)}$ wyznaczane jest w aktualnym kroku tak jak w eliminacji Gaussa, zaś $\tilde{\mathbf{L}}^{(k)}$ (czy $(\tilde{\mathbf{L}}^{(k)})^{-1}$) jest macierzą $\mathbf{L}^{(k)}$ (czy $(\mathbf{L}^{(k)})^{-1}$) z elementami *przestawionymi jedynie w kolumnie k*, zgodnie z przestawieniem wierszy w następnych krokach algorytmu (od kroku k+1-szego do ostatniego). Ilustrujemy to poniżej na prostym przykładzie, dla $n = 3$.

Załóżmy, że w kroku drugim (i ostatnim dla $n=3$) następuje zamiana wierszy 2 i 3, stąd przekształcenie wyznaczonej w kroku pierwszym macierzy $(\mathbf{L}^{(1)})^{-1}$ do macierzy $(\tilde{\mathbf{L}}^{(1)})^{-1}$ ma postać

$$\mathbf{P}^{(2)}(\mathbf{L}^{(1)})^{-1}\mathbf{P}^{(2)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ l_{31} & 0 & 1 \\ l_{21} & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ l_{31} & 1 & 0 \\ l_{21} & 0 & 1 \end{bmatrix} = (\tilde{\mathbf{L}}^{(1)})^{-1},$$

tzn. przemnożenie macierzy $(\mathbf{L}^{(1)})^{-1}$ lewostronnie przez $\mathbf{P}^{(2)}$ (zamiana wierszy 2 i 3) oraz prawostronnie przez $\mathbf{P}^{(2)}$ (zamiana kolumn 2 i 3) jest równoważne *zamianie elementów jedynie w pierwszej kolumnie* macierzy $(\mathbf{L}^{(1)})^{-1}$, odpowiadającej zamianom wierszy. Stąd reguła praktyczna:

> wykonując rozkład LU z częściowym (kolumnowym) wyborem elementu głównego i przestawiając w kroku k-tym wiersze w macierzy $\mathbf{A}^{(k)}$, należy identycznie przestawić wiersze w dotychczas wyznaczonej części macierzy $\mathbf{L}$ (składającej się z jej pierwszych $k-1$ kolumn), *tak jakby dotychczas wyznaczone części wierszy macierzy* $\mathbf{L}$ *były związane z wierszami macierzy* $\mathbf{A}^{(k)}$ (pokazane jest to w przykładzie 2.2 podanym dalej).

Realizacja powyższej reguły praktycznej jest wyjątkowo prosta, jeśli tworzone w k-tym kroku elementy l_{ik} macierzy $\mathbf{L}$ zapisujemy (przyporządkowujemy) w macierzy $\mathbf{A}^{(k)}$ w miejscach odpowiadających im zerowanych elementów $a_{ik}^{(k)}$ tej macierzy. Wówczas wystarczy, w następnych krokach, zamieniać miejscami całe wiersze tak zapisanej macierzy, łącznie z wyznaczonymi w poprzednich krokach elementami l_{ij}. Postępowanie powyższe pokazujemy w przykładzie 2.2.

Przykład 2.2. Rozwiążemy układ równań ten sam co w przykładzie 2.1, ale z częściowym wyborem elementu głównego.

$$\begin{bmatrix} 3 & 1 & 6 \\ 2 & 1 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 7 \\ 4 \end{bmatrix}.$$

Pierwszy krok (identyczny z pierwszym krokiem w przykładzie 2.1):

$$\left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ 2 & 1 & 3 & 7 \\ 1 & 1 & 1 & 4 \end{array}\right] \Rightarrow \begin{array}{l} l_{21} = \frac{a_{21}^{(1)}}{a_{11}^{(1)}} = \frac{2}{3} \\ l_{31} = \frac{a_{31}^{(1)}}{a_{11}^{(1)}} = \frac{1}{3} \end{array} \Rightarrow \left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ \frac{2}{3} & \frac{1}{3} & -1 & \frac{17}{3} \\ \frac{1}{3} & \frac{2}{3} & -1 & \frac{10}{3} \end{array}\right].$$

Drugi krok:

$$\mathbf{P}^{(2)} \Rightarrow \left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ \frac{1}{3} & \mathbf{\frac{2}{3}} & -1 & \frac{10}{3} \\ \frac{2}{3} & \frac{1}{3} & -1 & \frac{17}{3} \end{array}\right] \Rightarrow l_{32} = \frac{1}{2} \Rightarrow \left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ \frac{1}{3} & \frac{2}{3} & -1 & \frac{10}{3} \\ \frac{2}{3} & \frac{1}{2} & -\frac{1}{2} & 4 \end{array}\right].$$

Otrzymaliśmy oczywiście

$$\mathbf{LU} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{2}{3} & \frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 3 & 1 & 6 \\ 0 & \frac{2}{3} & -1 \\ 0 & 0 & -\frac{1}{2} \end{bmatrix} = \begin{bmatrix} 3 & 1 & 6 \\ 1 & 1 & 1 \\ 2 & 1 & 3 \end{bmatrix} = \mathbf{PA}.$$

Rozwiązanie układu równań z macierzą trójkątną górną (*backsubstitution*) wyznaczamy identycznie jak w przykładzie 2.1, oczywiście macierz $\mathbf{U}$ i wektor $\mathbf{b}^{(3)}$ są teraz nieco inne. □

Eliminacja Gaussa z pełnym wyborem elementu głównego

Każdy (k-ty) krok algorytmu eliminacji Gaussa zaczynamy od wyboru elementu głównego jako elementu o maksymalnym module spośród *wszystkich elementów*

podmacierzy $k \times k$ z prawego dolnego rogu macierzy $\mathbf{A}^{(k)}$, tzn. wybieramy element maksymalny (oznaczmy go jako (i,l)-ty) spośród elementów $a_{jp}^{(k)}$ $(j \geq k, p \geq k)$,

$$\left|a_{il}^{(k)}\right| = \max_{j,p} \left\{ \left|a_{jp}^{(k)}\right|, \;\; j,p = k, k+1, ..., n \right\}. \tag{2.35}$$

Mając wybrany element główny, tzn. określone jego indeksy kolumnowy l i wierszowy i, dokonujemy najpierw zamiany kolumn k-tej i l-tej – co oznacza identyczną *zamianę miejscami składowych w wektorze* $\boldsymbol{x}$, którą trzeba zapamiętać. Dalej postępujemy identycznie jak w algorytmie eliminacji Gaussa z wyborem częściowym kolumnowym: zamieniamy wiersze k-ty z i-tym i dokonujemy wyzerowania składowych w k-tej kolumnie macierzy przetwarzanej, poniżej diagonali. W efekcie takiego postępowania uzyskujemy układ równań z macierzą trójkątną górną postaci

$$\mathbf{U} = \mathbf{A}^{(n)} = (\mathbf{L}^{(n-1)}\mathbf{P}^{(n-1)}) \cdots (\mathbf{L}^{(2)}\mathbf{P}^{(2)})(\mathbf{L}^{(1)}\mathbf{P}^{(1)})\mathbf{A}^{(1)}\bar{\mathbf{P}}^{1}\bar{\mathbf{P}}^{2} \cdots \bar{\mathbf{P}}^{n-1}, \tag{2.36}$$

gdzie $\mathbf{P}$ i $\bar{\mathbf{P}}$ to macierze zamian wierszy i kolumn w kolejnych krokach:

$$\mathbf{P} = \mathbf{P}^{(n-1)}\mathbf{P}^{(n-2)} \cdots \mathbf{P}^{(1)}, \tag{2.37}$$

$$\bar{\mathbf{P}} = \bar{\mathbf{P}}^{(1)}\bar{\mathbf{P}}^{(\mathbf{2})} \cdots \bar{\mathbf{P}}^{(n-1)}. \tag{2.38}$$

Zwróćmy uwagę, że zamiana kolumn nie wpływa na przetwarzanie wektora prawych stron $\mathbf{b}^{(k)}$, jest on tak przekształcany jak w algorytmie z wyborem częściowym kolumnowym. Stąd postać przekształconego układu równań jest prawie taka sama jak przy wyborze częściowym

$$\mathbf{LU}\bar{\mathbf{x}} = \mathbf{Pb}, \tag{2.39}$$

jedynie, ze względu na zamiany również kolumn w macierzy przekształcanej, wektor $\bar{\mathbf{x}}$ różni się od oryginalnego wektora $\mathbf{x}$ kolejnością składowych oraz *finalny rozkład dotyczy nie oryginalnej macierzy* $\mathbf{A}$, *ale macierzy* $\mathbf{PA}\bar{\mathbf{P}}$,

$$\mathbf{LU} = \mathbf{PA}\bar{\mathbf{P}}. \tag{2.40}$$

Dodatkowy nakład obliczeń w algorytmie z wyborem pełnym, w porównaniu z wyborem częściowym kolumnowym, to przede wszystkim większy nakład na obliczenie i porównanie modułów elementów (w każdym kroku k^2 obliczeń modułów i k^2-1 porównań w stosunku do k obliczeń i $k-1$ porównań, $k = n, n-1, ..., 2$). Ponadto trzeba pamiętać o zamianach kolumn i związanych z tym zamianach miejscami składowych wektora rozwiązań $\mathbf{x}$ ($\bar{\mathbf{P}}\mathbf{x} = \bar{\mathbf{x}}$).

Przykład 2.3. Algorytm z pełnym wyborem elementu głównego zastosujemy do układu równań rozważonego w dwóch poprzednich przykładach 2.1 i 2.2 (gdzie

stosowano algorytmy bez wyboru i z wyborem częściowym kolumnowym). Układ równań jest postaci

$$\begin{bmatrix} 3 & 1 & 6 \\ 2 & 1 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 7 \\ 4 \end{bmatrix}.$$

Kolejne kroki algorytmu są następujące:

$$\left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ 2 & 1 & 3 & 7 \\ 1 & 1 & 1 & 4 \end{array}\right] \Rightarrow \bar{\mathbf{P}}^{(1)} \Rightarrow \left[\begin{array}{ccc|c} \mathbf{6} & 1 & 3 & 2 \\ 3 & 1 & 2 & 7 \\ 1 & 1 & 1 & 4 \end{array}\right]$$

$$\Rightarrow \begin{array}{c} l_{21} = \frac{3}{6} = \frac{1}{2} \\ l_{31} = \frac{1}{6} \end{array} \Rightarrow \left[\begin{array}{ccc|c} 6 & 1 & 3 & 2 \\ \hline \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 6 \\ \frac{1}{6} & \frac{5}{6} & \frac{1}{2} & \frac{11}{3} \end{array}\right] \Rightarrow \mathbf{P}^{(2)} \Rightarrow$$

$$\left[\begin{array}{ccc|c} 6 & 1 & 3 & 2 \\ \hline \frac{1}{6} & \mathbf{\frac{5}{6}} & \frac{1}{2} & \frac{11}{3} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 6 \end{array}\right] \Rightarrow l_{32} = \frac{3}{5} \Rightarrow \left[\begin{array}{ccc|c} 6 & 1 & 3 & 2 \\ \hline \frac{1}{6} & \frac{5}{6} & \frac{1}{2} & \frac{11}{3} \\ \frac{1}{2} & \frac{3}{5} & \frac{1}{5} & \frac{19}{5} \end{array}\right].$$

Mając macierz $\mathbf{U} = \mathbf{A}^{(3)} = \mathbf{PA}\bar{\mathbf{P}}$ i przekształcony wektor prawej strony $\mathbf{b}^{(3)} = [2\ \frac{11}{3}\ \frac{19}{5}]^{\mathrm{T}}$, wyliczamy rozwiązanie z układu równań $\mathbf{U}\bar{\mathbf{x}} = \mathbf{b}^{(3)}$, gdzie $\bar{\mathbf{x}} = \bar{\mathbf{P}}\mathbf{x} = [x_3\ x_2\ x_1]^{\mathrm{T}}$, jak poniżej:

$$\begin{bmatrix} 6 & 1 & 3 \\ 0 & \frac{5}{6} & \frac{1}{2} \\ 0 & 0 & \frac{1}{5} \end{bmatrix} \begin{bmatrix} x_3 \\ x_2 \\ x_1 \end{bmatrix} = \begin{bmatrix} 2 \\ \frac{11}{3} \\ \frac{19}{5} \end{bmatrix},$$

uzyskując $\hat{\mathbf{x}} = [19\ -7\ -8]^{\mathrm{T}}$, oczywiście tak jak w przykładzie 2.1. □

Błędy numeryczne rozkładu LU

Błędy numeryczne powstałe przy rozkładzie LU można analizować metodą pozornych równoważnych zaburzeń, tj. szacując równoważne tym błędom zaburzenie $\mathbf{E}$ macierzy faktoryzowanej $\mathbf{PA}$, tzn. np. dla rozkładu z wyborem częściowym elementu głównego (przypadek numerycznie bardziej krytyczny niż z wyborem pełnym) mamy

$$\widetilde{\mathbf{L}}\widetilde{\mathbf{U}} = \mathbf{PA} + \mathbf{E}, \tag{2.41}$$

gdzie $\widetilde{\mathbf{L}}$ i $\widetilde{\mathbf{U}}$ to macierze rozkładu uzyskane w arytmetyce zmiennopozycyjnej i, jednocześnie, przy dokładnym (bez błędów) rozkładzie macierzy $\mathbf{PA} + \mathbf{E}$. Można pokazać, zob. [1], że

$$\|\mathbf{E}\|_\infty \underset{1}{\leq} O(n^2)\, eps\, g_n, \tag{2.42}$$

gdzie

$$g_n = \max_{1\leq i,j,k\leq n} \left|a_{ij}^{(k)}\right| \leq 2^{n-1} a,$$

$$a = \max_{1\leq i,j\leq n} |a_{ij}|\,.$$

Podane szacowanie na g_n jest bardzo konserwatywne, w praktyce można przyjąć

$$g_n \leq \beta(n) \cdot a, \tag{2.43}$$

gdzie $\beta(n)$ jest stałą zależną od n, znacznie mniejszą od 2^{n-1}, np. przy całkowitym wyborze elementu głównego w praktyce g_n bardzo rzadko przekracza $8a$, zob. [1],[2]. Zależności (2.41) i (2.42) pokazują, że algorytm rozkładu LU jest numerycznie poprawny.

Rozważając numeryczne rozwiązanie $\tilde{\mathbf{x}}$ układu równań $\mathbf{Ax} = \mathbf{b}$, można pokazać (patrz niżej), że jest ono równoważne dokładnemu rozwiązaniu układu równań z zaburzoną macierzą $\mathbf{A}$,

$$(\mathbf{A} + \delta\mathbf{A})\tilde{\mathbf{x}} = \mathbf{b},$$

przy czym

$$\frac{\|\delta\mathbf{A}\|_\infty}{\|\mathbf{A}\|_\infty} \leq O\big(n^3\big)\, \beta(n) eps, \tag{2.44}$$

co dowodzi numerycznej poprawności algorytmu rozwiązania układu równań liniowych metodą eliminacji Gaussa (rozkładu LU) z wyborem częściowym (kolumnowym) elementu głównego – a więc tym bardziej i algorytmu z wyborem pełnym; przy czym oszacowanie błędu względnego rośnie co najmniej z trzecią potęgą wymiarowości problemu.

Oszacowanie błędów numerycznych* rozwiązania układu równań liniowych algorytmem eliminacji Gaussa – zależność (2.44).

Mamy

$$\mathbf{PAx} = \mathbf{Pb},$$
$$\mathbf{LUx} = \mathbf{Pb}.$$

Rozwiązujemy dwa układy równań:

$$\mathbf{Ly} = \mathbf{Pb} \;\text{ i }\; \mathbf{Ux} = \mathbf{y}.$$

*Materiał uzupełniający.

Oznaczając falkami ($\widetilde{\mathbf{L}}$, $\widetilde{\mathbf{U}}$, $\widetilde{\mathbf{y}}$, $\widetilde{\mathbf{x}}$, itd.) wielkości otrzymane w arytmetyce zmiennopozycyjnej, mamy:

$$\widetilde{\mathbf{L}}\widetilde{\mathbf{U}} = \mathbf{PA} + \mathbf{E},$$
$$(\widetilde{\mathbf{L}}+\widetilde{\delta\mathbf{L}})\widetilde{\mathbf{y}} = \mathbf{Pb},$$
$$(\widetilde{\mathbf{U}} + \widetilde{\delta\mathbf{U}})\widetilde{\mathbf{x}} = \widetilde{\mathbf{y}},$$

gdzie $\widetilde{\delta\mathbf{L}}$ i $\widetilde{\delta\mathbf{U}}$ oznaczają pozorne zaburzenia równoważne błędom numerycznym powstającym przy rozwiązywaniu układów trójkątnych. Po połączeniu dwóch ostatnich równań, otrzymujemy

$$(\widetilde{\mathbf{L}}+\widetilde{\delta\mathbf{L}})(\widetilde{\mathbf{U}}+\widetilde{\delta\mathbf{U}})\widetilde{\mathbf{x}} = \mathbf{Pb},$$

skąd drogą prostych przekształceń dostajemy poszukiwane wyrażenie na $\delta\mathbf{A}$ dla zaburzonego układu $(\mathbf{A} + \delta\mathbf{A})\,\widetilde{\mathbf{x}} = \mathbf{b}$:

$$\delta\mathbf{A} = \mathbf{P}^{-1}(\mathbf{E} + \widetilde{\mathbf{L}}\cdot\widetilde{\delta\mathbf{U}}+\widetilde{\delta\mathbf{L}}\cdot\widetilde{\mathbf{U}}+\widetilde{\delta\mathbf{L}}\cdot\widetilde{\delta\mathbf{U}}),$$

skąd

$$\|\delta\mathbf{A}\| \leq + \|\mathbf{E}\| + \|\widetilde{\mathbf{L}}\|\|\widetilde{\delta\mathbf{U}}\| + \|\widetilde{\delta\mathbf{L}}\|\|\widetilde{\mathbf{U}}\| + \|\widetilde{\delta\mathbf{L}}\|\|\widetilde{\delta\mathbf{U}}\|.$$

Ponieważ $|l_{ij}| \leq 1$ oraz $\widetilde{\mathbf{U}} = \mathbf{A}^{(n)}$, więc

$$\|\widetilde{\mathbf{L}}\|_\infty \leq n,$$
$$\|\widetilde{\mathbf{U}}\|_\infty \leq n g_n.$$

Dla układów z macierzą trójkątną mamy

$$\|\widetilde{\delta\mathbf{L}}\|_\infty \leq O(\frac{1}{2}n^2)eps,$$
$$\|\widetilde{\delta\mathbf{U}}\|_\infty \leq O(\frac{1}{2}n^2)eps\, g_n,$$

stąd, po uwzględnieniu oszacowania (2.42) dla $\mathbf{E}$, dostajemy

$$\|\delta\mathbf{A}\|_\infty \leq \left[O(n^2) + O(\frac{1}{2}n^3) + O(\frac{1}{2}n^3)\right] eps\, g_n.$$

Ponieważ $\|\mathbf{A}\|_\infty \geq a$, więc przyjmując $g_n \leq \beta(n)\, a$ ($\beta(n) \leq 2^{n-1}$) otrzymujemy ostatecznie, dla dostatecznie silnej arytmetyki ($eps^2 \ll eps$):

$$\frac{\|\delta\mathbf{A}\|_\infty}{\|\mathbf{A}\|_\infty} \leq O\big(n^3\big)\, \beta(n)\, eps.$$

□

2.3.5. Iteracyjne poprawianie rozwiązania

Układ równań liniowych po podstawieniu otrzymanego algorytmem numerycznym rozwiązania (oznaczmy je przez $\mathbf{x}^{(1)}$) na ogół nie jest dokładnie spełniony, tzn.

$$\mathbf{r}^{(1)} \stackrel{\text{df}}{=} \mathbf{A}\mathbf{x}^{(1)} - \mathbf{b} \neq 0,$$

gdzie błąd niespełnienia równań $\mathbf{r}^{(1)}$ nazywany jest *resztą* lub *residuum*. Oznaczając rozwiązanie dokładne przez $\widehat{\mathbf{x}}$, mamy

$$\begin{aligned}
\mathbf{x}^{(1)} &= \widehat{\mathbf{x}} + \delta\mathbf{x}, \\
\mathbf{A}\left(\mathbf{x}^{(1)} - \delta\mathbf{x}\right) &= \mathbf{b}, \\
\mathbf{A}\delta\mathbf{x} &= \mathbf{A}\mathbf{x}^{(1)} - \mathbf{b}, \\
\mathbf{A}\delta\mathbf{x} &= \mathbf{r}^{(1)}.
\end{aligned}$$

Stąd możemy, w celu poprawienia dokładności wyniku, rozwiązać układ równań z $\delta\mathbf{x}$ jako niewiadomą i wektorem reszty po prawej stronie. Procedura jest następująca:

1. Obliczamy resztę $\mathbf{r}^{(1)} = \mathbf{A}\mathbf{x}^{(1)} - \mathbf{b}$.
2. Rozwiązujemy $\mathbf{A}\delta\mathbf{x} = \mathbf{r}^{(1)}$, korzystając z rozkładu macierzy $\mathbf{A}$ wykorzystanego uprzednio przy znajdowaniu rozwiązania $\mathbf{x}^{(1)}$ (np. $\mathbf{LU}$). Uzyskujemy w efekcie kolejne przybliżenie rozwiązania $\mathbf{x}^{(2)}$:
$$\mathbf{x}^{(2)} = \mathbf{x}^{(1)} - \delta\mathbf{x}.$$
3. Obliczamy resztę $\mathbf{r}^{(2)} = \mathbf{A}\mathbf{x}^{(2)} - \mathbf{b}$, jeśli jest istotnie mniejsza od $\mathbf{r}^{(1)}$ i nadal zbyt duża, to postępowanie powtarzamy, itd.

Wektor reszt $\mathbf{r}^{(i)}$ należy obliczać w podwyższonej (podwójnej) precyzji.

Obliczanie poprawionych wartości rozwiązania $\mathbf{x}^{(2)}$, $\mathbf{x}^{(3)}, \ldots$ nie jest kosztowne obliczeniowo, w porównaniu do obliczenia wartości pierwszego rozwiąznia $\mathbf{x}^{(1)}$, gdyż najbardziej kosztowna faktoryzacja macierzy jest już obliczona (przy wyznaczaniu $\mathbf{x}^{(1)}$).

2.3.6. Metoda eliminacji zupełnej (Gaussa-Jordana)

Krok 1: Identyczny z pierwszym krokiem eliminacji Gaussa: dzielimy pierwszy wiersz przez element centralny $a_{11}^{(1)}$, a następnie zerujemy pierwszą kolumnę z wyjątkiem elementu w pierwszym wierszu.

Krok 2: Dzielimy drugi wiersz przez element centralny $a_{22}^{(2)}$, następnie, postępując analogicznie jak w eliminacji Gaussa, zerujemy *całą drugą kolumnę oprócz ele-*

mentu w drugim wierszu – tzn. zerujemy też elementy *nad diagonalą* (w taki sam sposób). Uzyskujemy

$$\begin{array}{ccccccccccc} x_1 & & + & a_{13}^{(3)}x_3 & + & \cdots & + & a_{1n}^{(3)}x_n & = & b_1^{(3)}, \\ & x_2 & + & a_{23}^{(3)}x_3 & + & \cdots & + & a_{2n}^{(3)}x_n & = & b_2^{(3)}, \\ & & & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & a_{n3}^{(3)}x_3 & + & \cdots & + & a_{nn}^{(3)}x_n & = & b_n^{(3)}, \end{array}$$

itd., aż po $n-1$ krokach uzyskamy

$$\begin{array}{ccccccc} x_1 & & & & & = & b_1^{(n)}, \\ & x_2 & & & & = & b_2^{(n)}, \\ & & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & & x_n & = & b_n^{(n)}. \end{array}$$

Nakład obliczeń jest rzędu $D = O(\frac{1}{2}n^3)$, $M = O(\frac{1}{2}n^3)$. Metodę można stosować przy jednokrotnym rozwiązaniu układu równań liniowych (nie dostajemy żadnego rozkładu macierzy), a szczególnie przy rozwiązywaniu tzw. obciętego układu równań.

2.4. Rozkład Cholesky'ego - Banachiewicza ($\mathbf{LL}^{\mathrm{T}}$)

2.4.1. Rozkład $\mathbf{LL}^{\mathrm{T}}$

Macierz symetryczną nazywamy *dodatnio określoną*, jeśli

$$\forall \mathbf{x} \neq 0 \quad \mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} > 0. \tag{2.45}$$

Twierdzenie 2.1. *Dla każdej symetrycznej dodatnio określonej macierzy* $\mathbf{A}$ *istnieje dokładnie jedna trójkątna dolna macierz* $\mathbf{L}$ *o dodatnich elementach diagonalnych taka, że:*

$$\mathbf{A} = \mathbf{L}\mathbf{L}^{\mathrm{T}}. \tag{2.46}$$

Rozkład (2.46) nazywamy *rozkładem Cholesky'ego-Banachiewicza*, lub krótko: *rozkładem* LL^{T}.

Wyznaczanie rozkładu: zapiszmy rozkład z dokładnością do poszczególnych elementów macierzy, tzn.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & \cdots & 0 \\ l_{21} & l_{22} & \cdots & 0 \\ \cdot & \cdot & \cdot & \cdot \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & \cdots & l_{n1} \\ 0 & l_{22} & \cdots & l_{n2} \\ \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdots & l_{nn} \end{bmatrix}.$$

Powyższe równanie macierzowe potraktujemy jako układ równań skalarnych z niewiadomymi elementami l_{ij} macierzy $\mathbf{L}$ ($n^2/2 + n/2$ niewiadomych i równań dla n parzystych, oraz $(n^2-1)/2+(n+1)/2$ dla n nieparzystych). Układ ten będziemy rozwiązywać równanie po równaniu, poruszając się w dół wzdłuż kolejnych kolumn macierzy $\mathbf{A}$, poczynając od elementów na diagonali, tzn. zaczynając od elementu a_{11} w pierwszej kolumnie:

$$
\begin{aligned}
&a_{11} = l_{11}^2 \quad \Rightarrow \quad l_{11} = \sqrt{a_{11}},\\
&a_{j1} = l_{j1} l_{11} \quad \Rightarrow \quad l_{j1} = a_{j1} \,/\, l_{11}, \qquad j = 2,3,...,n,\\
&a_{22} = l_{21}^2 + l_{22}^2 \quad \Rightarrow \quad l_{22} = \sqrt{a_{22} - l_{21}^2},\\
&a_{j2} = l_{j1} \cdot l_{21} + l_{j2} \cdot l_{22} \quad \Rightarrow \quad l_{j2} = (a_{j2} - l_{j1} \cdot l_{21}) \,/\, l_{22}, \qquad j = 3,4,...,n,
\end{aligned}
$$

itd.

Ogólnie, dla $i = 1,2,3,...,n$:

$$
\begin{aligned}
&a_{ii} = l_{i1}^2 + l_{i2}^2 + ... + l_{ii}^2,\\
&a_{ji} = l_{j1} \cdot l_{i1} + l_{j2} \cdot l_{i2} + ... + l_{ji} \cdot l_{ii},, \qquad j = i+1, i+2, ..., n,
\end{aligned}
$$

skąd otrzymujemy algorytm

$$
l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2}, \tag{2.47a}
$$

$$
l_{ji} = (a_{ji} - \sum_{k=1}^{i-1} l_{jk} \cdot l_{ik}) / l_{ii}, \;\; i = 1,2,...,n, \;\; j = i+1, i+2, ..., n. \tag{2.47b}
$$

Łatwo policzyć, że nakład obliczeń potrzebny dla wyznaczenia macierzy $\mathbf{L}$ wynosi $M, D = O(\frac{1}{6}n^3)$, ponadto trzeba wykonać n pierwiastkowań.

Przykład 2.4. Wyznaczyć rozkład LL^{T} macierzy $\mathbf{A}$,

$$
\mathbf{A} = \begin{bmatrix} 1 & 2 & 4 \\ 2 & 13 & 23 \\ 4 & 23 & 77 \end{bmatrix}.
$$

Mamy:

$$
\begin{bmatrix} 1 & 2 & 4 \\ 2 & 13 & 23 \\ 4 & 23 & 77 \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix}.
$$

Stąd kolejne równania skalarne:

$$\begin{aligned} 1 = l_{11}^2 \quad &\Rightarrow \quad l_{11} = 1, \\ 2 = l_{21} l_{11} \quad &\Rightarrow \quad l_{21} = 2 \,/\, 1 = 2, \\ 4 = l_{31} l_{11} \quad &\Rightarrow \quad l_{31} = 4 \,/\, 1 = 4, \end{aligned}$$

$$\begin{aligned} 13 = l_{21}^2 + l_{22}^2 \quad &\Rightarrow \quad l_{22} = \sqrt{13-4} = 3, \\ 23 = l_{31} l_{21} + l_{32} l_{22} \quad &\Rightarrow \quad l_{32} = (23-8)/3 = 5, \\ 77 = l_{31}^2 + l_{32}^2 + l_{33}^2 \quad &\Rightarrow \quad l_{33} = \sqrt{77-16-25} = 6 \end{aligned}$$

i uzyskana macierz $\mathbf{L}$ ma postać:

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 3 & 0 \\ 4 & 5 & 6 \end{bmatrix}.$$

□

Jeśli w układzie równań liniowych $\mathbf{Ax} = \mathbf{b}$ macierz $\mathbf{A}$ spełnia założenia o rozkładzie LL^{T} (symetryczna, dodatnio określona), to rozwiązując układ równań, należy oczywiście stosować rozkład LL^{T}, a nie LU – jako dwukrotnie efektywniejszy obliczeniowo i generujący mniejsze błędy numeryczne.
Uwaga. macierze $\mathbf{L}$ z rozkładów LU i LL^{T}, mimo identycznego oznaczenia „$\mathbf{L}$" symbolizujacego macierz dolną trójkątną, *to macierze różne*. Relację między nimi wyjaśniamy szczegółowo w następnym podrozdziale.

2.4.2. Rozkład $\mathrm{LDL}^{\mathrm{T}}$, relacje między rozkładami trójkątnymi

Z rozkładem LL^{T} blisko związany jest rozkład $\mathrm{LDL}^{\mathrm{T}}$ (traktowany również jako wersja rozkładu LL^{T}), który ma postać:

$$\mathbf{A} = \overline{\mathbf{L}}\mathbf{D}\overline{\mathbf{L}}^{\mathrm{T}} =$$

$$= \begin{bmatrix} 1 & & & \mathbf{0} \\ \overline{l}_{21} & 1 & & \\ \vdots & & \ddots & \\ \overline{l}_{n1} & \overline{l}_{n2} & \cdots & 1 \end{bmatrix} \begin{bmatrix} d_{11} & & & \mathbf{0} \\ & d_{22} & & \\ & & \ddots & \\ \mathbf{0} & & & d_{nn} \end{bmatrix} \begin{bmatrix} 1 & \overline{l}_{21} & \cdots & \overline{l}_{n1} \\ & 1 & & \overline{l}_{n2} \\ & & \ddots & \\ \mathbf{0} & & & 1 \end{bmatrix}. \tag{2.48}$$

Dla macierzy symetrycznych dodatnio określonych (zob. (2.45)) oba rozkłady istnieją i w rozkładzie $\mathrm{LDL}^{\mathrm{T}}$ mamy $d_{ii} > 0,\ i = 1, ..., n$, co pokażemy poniżej, wyznaczając relacje miedzy nimi. Oznaczmy $d_{ii} = l_{ii}^2,\ i = 1, ..., n$, wówczas

$$\overline{\mathbf{L}}\mathbf{D}\overline{\mathbf{L}}^{\mathrm{T}} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ \overline{l}_{21} & 1 & & 0 \\ \vdots & & \ddots & \\ \overline{l}_{n1} & \overline{l}_{n2} & \cdots & 1 \end{bmatrix} \begin{bmatrix} l_{11}^2 & 0 & \cdots & 0 \\ 0 & l_{22}^2 & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & l_{nn}^2 \end{bmatrix} \begin{bmatrix} 1 & \overline{l}_{21} & \cdots & \overline{l}_{n1} \\ 0 & 1 & & \overline{l}_{n2} \\ \vdots & & \ddots & \\ 0 & 0 & & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ \overline{l}_{21} & 1 & & 0 \\ \vdots & & \ddots & \\ \overline{l}_{n1} & \overline{l}_{n2} & \cdots & 1 \end{bmatrix} \begin{bmatrix} l_{11} & 0 & \cdots & 0 \\ 0 & l_{22} & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & l_{nn} \end{bmatrix} \cdot \begin{bmatrix} l_{11} & 0 & \cdots & 0 \\ 0 & l_{22} & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} 1 & \overline{l}_{21} & \cdots & \overline{l}_{n1} \\ 0 & 1 & & \overline{l}_{n2} \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

$$= \begin{bmatrix} l_{11} & 0 & \cdots & 0 \\ l_{21} & l_{22} & \cdots & 0 \\ \cdot & \cdot & \cdot & \cdot \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \cdot \begin{bmatrix} l_{11} & l_{21} & \cdots & l_{n1} \\ 0 & l_{22} & \cdots & l_{n2} \\ \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdots & l_{nn} \end{bmatrix} = \mathbf{L}\mathbf{L}^{\mathrm{T}},$$

tzn. mamy następujące przejście od rozkładu LL^{T} do rozkładu $\mathrm{LDL}^{\mathrm{T}}$:

$$\begin{aligned} \mathbf{D} &= \mathrm{diag}\{{l_{ii}}^2\}, \\ \overline{\mathbf{L}} &= \mathbf{L}\left[\mathrm{diag}\{l_{ii}\}\right]^{-1}, \end{aligned} \tag{2.49}$$

i w odwrotnym kierunku

$$\mathbf{L} = \overline{\mathbf{L}}\,\mathrm{diag}\{\sqrt{d_{ii}}\}. \tag{2.50}$$

Wzory na elementy rozkładu $\mathrm{LDL}^{\mathrm{T}}$ najprościej można wyznaczyć podobnie jak dla rozkładu LL^{T}, tzn. przedstawiając rozkład w postaci równania macierzowego. Po przemnożeniu dwóch ostatnich macierzy rozkładu $\mathrm{LDL}^{\mathrm{T}}$ dostajemy

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} 1 & & & \mathbf{0} \\ \overline{l}_{21} & 1 & & \\ \vdots & & \ddots & \\ \overline{l}_{n1} & \overline{l}_{n2} & \cdots & 1 \end{bmatrix} \begin{bmatrix} d_{11} & d_{11}\overline{l}_{21} & \cdots & d_{11}\overline{l}_{n1} \\ 0 & d_{22} & & d_{22}\overline{l}_{n2} \\ & & \ddots & \\ 0 & 0 & \cdots & d_{nn} \end{bmatrix}$$

i rozwiązujemy równania skalarne, poruszając się w dół wzdłuż kolumn macierzy $\mathbf{A}$ i rozpoczynając od elementów diagonalnych (jak dla rozkładu LL^{T}):

$$\begin{aligned} &a_{11} = d_{11}^2 \quad \Rightarrow \quad d_{11} = a_{11}, \\ &a_{j1} = d_{11}\overline{l}_{j1} \quad \Rightarrow \quad \overline{l}_{j1} = a_{j1} \,/\, d_{11}, \qquad j = 2, 3, ..., n, \\ &a_{22} = d_{11}\overline{l}_{21}^2 + d_{22} \quad \Rightarrow \quad d_{22} = a_{22} - d_{11}\overline{l}_{21}^2, \\ &a_{j2} = \overline{l}_{j1}d_{11}\overline{l}_{21} + \overline{l}_{j2}d_{22} \quad \Rightarrow \quad l_{j2} = (a_{j2} - \overline{l}_{j1}d_{11}\overline{l}_{21}) \,/\, d_{22}, \quad j = 3, 4, ..., n, \\ &\quad \text{itd.} \end{aligned}$$

Daje to następujący algorytm:

$$d_{ii} = a_{ii} - \sum_{k=1}^{i-1} \overline{l}_{ik}^2 d_{kk}, \tag{2.51a}$$

$$\overline{l}_{ji} = (a_{ji} - \sum_{k=1}^{i-1} \overline{l}_{jk} d_{kk} \overline{l}_{ik})/d_{ii}, \;\; i = 1, ..., n, \;\; j = i+1, ..., n, \tag{2.51b}$$

który jest *dobrze określony dla macierzy symetrycznych nieosobliwych*, gdyż wówczas mamy $d_{ii} \neq 0$, $i = 1, ..., n$ (a w algorytmie występuje dzielenie przez d_{ii}). Nakład obliczeń jest rzędu takiego, jak dla rozkładu LL^{T}: $M, D = O(\frac{1}{6}n^3)$.

W ogólności *rozkład* $\mathrm{LDL}^{\mathrm{T}}$ *istnieje dla dowolnych macierzy symetrycznych*. Dla macierzy nieokreślonych (nie dodatnio lub ujemnie określonych/ półokreślonych) elementy d_{ii} macierzy diagonalnej $\mathbf{D}$ mogą przyjmować dowolne wartości: dodatnie, ujemne i zerowe. Mające dobre własności numeryczne algorytmy wyznaczania rozkładu $\mathrm{LDL}^{\mathrm{T}}$ stosują odpowiednie przestawianie wierszy (i jednocześnie kolumn, dla zachowania symetrii), uzyskujemy wówczas rozkład

$$\mathbf{PAP}' = \mathbf{LDL}^{\mathrm{T}},$$

gdzie $\mathbf{P}$ jest macierzą przestawień. Alternatywą jest tzw. blokowy rozkład $\mathrm{LDL}^{\mathrm{T}}$, gdzie na diagonali macierzy $\mathbf{D}$ występują elementy macierzowe o wymiarach 1×1 lub 2×2. Procedura „`ldl`" pakietu MATLAB umożliwia wykonywanie rozkładu $\mathrm{LDL}^{\mathrm{T}}$ w obu wymienionych wersjach (dla dowolnych macierzy symetrycznych).

Rozkład $\mathrm{LDL}^{\mathrm{T}}$ stosowany jest np. w numerycznych algorytmach optymalizacji wykorzystujących macierze drugich pochodnych funkcji optymalizowanej.

Rozkład $\mathrm{LDL}^{\mathrm{T}}$ jest niejako ogniwem pośrednim między rozkładami LU i LL^{T}, będąc bezpośrednio związany z każdym z tych rozkładów, oczywiście dla macierzy, dla których wszystkie rozkłady istnieją (symetrycznych dodatnio określonych). Relacja między rozkładami $\mathrm{LDL}^{\mathrm{T}}$ i LL^{T} została już omówiona, natomiast relacja między rozkładami $\mathrm{LDL}^{\mathrm{T}}$ i LU jest następująca:

$$\mathbf{A} = \overline{\mathbf{L}}\mathbf{D}\overline{\mathbf{L}}^{\mathrm{T}} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ \overline{l}_{21} & 1 & & 0 \\ \vdots & & \ddots & \\ \overline{l}_{n1} & \overline{l}_{n2} & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} d_{11} & 0 & \cdots & 0 \\ 0 & d_{22} & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & d_{nn} \end{bmatrix} \begin{bmatrix} 1 & \overline{l}_{21} & \cdots & \overline{l}_{n1} \\ 0 & 1 & & \overline{l}_{n2} \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ \overline{l}_{21} & 1 & & 0 \\ \vdots & & \ddots & \\ \overline{l}_{n1} & \overline{l}_{n2} & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} d_{11} & d_{11}\overline{l}_{21} & \cdots & d_{11}\overline{l}_{n1} \\ 0 & d_{22} & & d_{22}\overline{l}_{n2} \\ & & \ddots & \\ 0 & 0 & \cdots & d_{nn} \end{bmatrix} = \overline{\mathbf{L}}\mathbf{U}\,.$$

2.5. Obliczanie wyznacznika i macierzy odwrotnej

Przypomnijmy, że dla dowolnych macierzy kwadratowych $\mathbf{B}$ i $\mathbf{C}$ o tym samym wymiarze

$$\det(\mathbf{BC}) = \det \mathbf{B} \cdot \det \mathbf{C},$$

oraz że wyznacznik macierzy trójkątnej (górnej lub dolnej) jest równy iloczynowi jej elementów diagonalnych.

Dla numerycznie poprawnego obliczania *wyznacznika* korzystamy z rozkładów, tzn. obliczamy wyznacznik następująco:

$$\text{dla rozkładu } \mathbf{LU}\text{:} \quad \det \mathbf{A} = \det\left(\mathbf{P}^T\mathbf{LU}\right) = \det\mathbf{P}\det\mathbf{U} = \det\mathbf{P}\prod_{i=1}^{n} u_{ii},$$

$$\text{dla rozkładu } \mathbf{LL}^{\mathrm{T}}\text{:} \quad \det \mathbf{A} = \det\left(\mathbf{LL}^{\mathrm{T}}\right) = (\det\mathbf{L})^2 = (\prod_{i=1}^{n} l_{ii})^2,$$

$$\text{dla rozkładu } \mathbf{LDL}^{\mathrm{T}}\text{:} \quad \det \mathbf{A} = \det\left(\mathbf{LDL}^{\mathrm{T}}\right) = \det\mathbf{D} = \prod_{i=1}^{n} d_{ii}.$$

Dla numerycznie poprawnego wyznaczania *macierzy odwrotnej* korzystamy z rozkładów trójkątnych macierzy, mamy tu dwa podstawowe sposoby.

Pierwszy sposób wykorzystuje *bezpośrednie odwracanie macierzy trójkątnych*:

$$(\mathbf{LU})^{-1} = \mathbf{U}^{-1}\cdot\mathbf{L}^{-1}, \tag{2.52}$$

$$\left(\mathbf{LL}^{\mathrm{T}}\right)^{-1} = \left(\mathbf{L}^{\mathrm{T}}\right)^{-1}\cdot\mathbf{L}^{-1} = \left(\mathbf{L}^{-1}\right)^{\mathrm{T}}\cdot\mathbf{L}^{-1}. \tag{2.53}$$

Jest to metoda efektywna obliczeniowo, gdyż dla macierzy trójkątnych:
- odwrotna do macierzy trójkątnej górnej (dolnej) jest też macierz trójkątna górna (dolna),
- obliczenie odwrotności $\mathbf{Y} = \{y_{ij}\}$ pojedynczej macierzy trójkątnej jest efektywne, realizujemy je bezpośrednio z macierzowego równania definicyjnego, rozwiązując kolejno równania skalarne.

Na przykład, dla macierzy $\mathbf{L}$ mamy równanie definicyjne:

$$\mathbf{I} = \mathbf{LY} \;\Leftrightarrow\; \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & \cdots & 0 \\ l_{21} & l_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} y_{11} & 0 & \cdots & 0 \\ y_{21} & y_{22} & & 0 \\ \vdots & \vdots & \ddots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nn} \end{bmatrix}.$$

Elementy macierzy odwrotnej wyznaczamy kolejno kolumnami, poczynając od pierwszej kolumny macierzy jednostkowej, a w każdej kolumnie poruszając się w dół, rozpoczynając od elementu diagonalnego (analogicznie do rozkładu LL^{T}):

$$\begin{aligned} 1 &= l_{11}y_{11} \quad \Rightarrow \quad y_{11} = 1/l_{11}, \\ 0 &= l_{21}y_{11} + l_{22}y_{21} \quad \Rightarrow \quad y_{21} = (-l_{21}y_{11}) \,/\, l_{22}, \\ 0 &= l_{31}y_{11} + l_{32}y_{21} + l_{33}y_{31} \quad \Rightarrow \quad y_{31} = -(l_{31}y_{11} + l_{32}y_{21}) \,/\, l_{33}, \\ &\vdots \end{aligned}$$

$$\vdots$$

$$0 = l_{n1}y_{11} + \cdots + l_{nn}y_{n1} \quad \Rightarrow \quad y_{n1} = -(l_{n1}y_{11} + \cdots + l_{n,n-1}y_{n-1,1}) \,/\, l_{nn},$$
$$1 = l_{22}y_{22} \quad \Rightarrow \quad y_{22} = 1/l_{22},$$
$$0 = l_{32}y_{22} + l_{33}y_{32} \quad \Rightarrow \quad y_{32} = (-l_{32}y_{22}) \,/\, l_{33},$$
itd.

Czytelnikowi pozostawiamy sformułowanie wzorów ogólnych (analogicznych do (2.47a)-(2.47b)), potrzebnych przy efektywnej implementacji komputerowej (programowej).

Nakład obliczeń na policzenie jednej macierzy odwrotnej do macierzy trójkątnej jest $M, D = O(\frac{1}{6}n^3)$, zaś nakład obliczeń na policzenie ilorazu macierzy trójkątnych wynosi $M, D = O(\frac{1}{3}n^3)$. Stąd, obliczenie macierzy odwrotnej, np.:

– stosując rozkład LU: wymaga nakładu $M, D = O(n^3)$ ($O(\frac{1}{3}n^3)$ wyznaczenie rozkładu + $2{\cdot}O(\frac{1}{6}n^3)$ obliczenie odwrotności macierzy trójkątnych + $O(\frac{1}{3}n^3)$ przemnożenie macierzy),
– stosując rozkład LL^{T}: wymaga nakładu $M, D = O(\frac{5}{6}n^3)$ oraz n pierwiastkowań.

Drugi sposób obliczania macierzy odwrotnej polega na *rozwiązywaniu n układów równań liniowych*. Zilustrujemy go na przykładzie rozkładu LU, gdzie mamy $\mathbf{LU} = \mathbf{PA}$, skąd

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{I},$$
$$\mathbf{PA}\mathbf{A}^{-1} = \mathbf{PI},$$
$$\mathbf{LU}\mathbf{A}^{-1} = \mathbf{PI},$$

gdzie macierz $\mathbf{P}$ jest macierzą przestawień w eliminacji Gaussa. Oznaczmy

$$\mathbf{A}^{-1} = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & & y_{2n} \\ \vdots & \vdots & & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nn} \end{bmatrix},$$

wtedy ostatnią równość możemy zapisać w postaci

$$\mathbf{LU}\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nn} \\ \mathbf{y}_1 & \mathbf{y}_2 & & \mathbf{y}_n \end{bmatrix} = \mathbf{P}\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & \\ \vdots & \vdots & & \vdots \\ 0 & & \cdots & 1 \\ \mathbf{e}_1 & \mathbf{e}_2 & & \mathbf{e}_n \end{bmatrix}.$$

Powyższe równanie macierzowe jest równoważne n układom równań liniowych:

$$\begin{aligned}\mathbf{LU}\mathbf{y}_1 &= \mathbf{P}\mathbf{e}_1,\\ \mathbf{LU}\mathbf{y}_2 &= \mathbf{P}\mathbf{e}_2,\\ \vdots &= \vdots\\ \mathbf{LU}\mathbf{y}_n &= \mathbf{P}\mathbf{e}_n,\end{aligned}$$

z których każdy korzysta z tego samego rozkładu LU. Nakład obliczeń jest rzędu $\frac{1}{2}n^2 \cdot 2 \cdot n = n^3$ dodawań i mnożeń, oraz dodatkowo rzędu $\frac{1}{3}n^3$ na rozkład LU.

Uwaga. Bez istotnej potrzeby nie należy wyznaczać macierzy odwrotnej. Dla prównania, nakłady obliczeń przy rozwiązaniu układu równań $\mathbf{A}\mathbf{x} = \mathbf{b}$, z zastosowaniem z rozkładu $\mathbf{LU}$, po pierwsze: przez rozwiązanie dwóch układów równań z macierzami trójkątnymi, i po drugie: przez obliczenie $\mathbf{A}^{-1} = (\mathbf{LU})^{-1}$ i przemnożenie przez wektor $\mathbf{b}$, są następujących rzędów:

$$\begin{array}{cccc} & & \mathbf{Ly} = \mathbf{b} & \mathbf{Ux} = \mathbf{y} \\ & \diagup & O\left(\frac{1}{2}n^2\right) & + \quad O\left(\frac{1}{2}n^2\right) \\ \mathbf{LU} & & & \\ & \diagdown & (\mathbf{LU})^{-1} & \mathbf{x} = (\mathbf{LU})^{-1}\mathbf{b} \\ & & O\left(\frac{2}{3}n^3\right) & + \quad O\left(n^2\right) \end{array}$$

Ponadto, ze względu na większą liczbę obliczeń, błędy numeryczne są w pierwszym (pokazanym wyżej) przypadku mniejsze.

2.6. Iteracyjne rozwiązywanie układu równań liniowych

Metody iteracyjne rozwiązywania układu równań $\mathbf{A}\mathbf{x} = \mathbf{b}$ są stosowane przede wszystkim wówczas, gdy problem jest wielkowymiarowy i macierze są rzadkie.

Rozważmy ciąg wektorów $\mathbf{x}^{(n)}$, gdzie $n = 0, 1, ...$ i $\mathbf{x}^{(0)}$ jest danym punktem startowym (np. najlepsze znane przybliżenie rozwiązania), generowany wg wzoru

$$\mathbf{x}^{(i+1)} = \mathbf{M}\mathbf{x}^{(i)} + \mathbf{w}, \tag{2.54}$$

gdzie $\mathbf{M}$ jest pewną macierzą. Następujące twierdzenie podaje warunek konieczny i dostateczny zbieżności takiego ciągu.

Twierdzenie 2.2. *Ciąg* $\left\{\mathbf{x}^{(i)}\right\}$*określony wzorem* $\mathbf{x}^{(i+1)} = \mathbf{M}\mathbf{x}^{(i)} + \mathbf{w}$ *jest, dla dowolnego punktu* $\mathbf{x}^{(0)}$*, zbieżny do punktu* $\widehat{\mathbf{x}}$ *będącego rozwiązaniem równania* $\mathbf{x} = \mathbf{M}\mathbf{x} + \mathbf{w}$ *wtedy i tylko wtedy, gdy*

$$\mathrm{sr}\,(\mathbf{M}) < 1, \tag{2.55}$$

gdzie $\mathrm{sr}\,(\mathbf{M})$ *oznacza promień spektralny macierzy* $\mathbf{M}$*.*

Uwaga.* Z twierdzenia 2.2 wynika, że można dobierać macierz $\mathbf{M}$ i wektor $\mathbf{w}$ arbitralnie, jedynie tak, aby spełnić:

1. warunek zbieżności: $\mathrm{sr}\,(\mathbf{M}) < 1$,
2. warunek zgodności: $\widehat{\mathbf{x}} = \mathbf{M}\widehat{\mathbf{x}} + \mathbf{w}$.

Teoretycznie, można by przyjąć dowolną macierz $\mathbf{M}$ o promieniu spektralnym mniejszym od 1, a następnie z warunku zgodności obliczyć

$$\mathbf{w} = (\mathbf{I} - \mathbf{M})\,\mathbf{A}^{-1}\mathbf{b},$$

co jest oczywiście *niesensowne,* gdyż obliczając $\mathbf{A}^{-1}\mathbf{b}$, można od razu wyznaczyć rozwiązanie $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$, ponadto macierz $\mathbf{A}$ może być duża.

Sensowniejsze jest postępowanie odwrotne: wybieramy macierz kwadratową $\mathbf{N}$, przyjmując $\mathbf{w} = \mathbf{N}\mathbf{b}$. Wówczas z warunku zgodności wynika

$$\mathbf{A}^{-1}\mathbf{b} = \mathbf{M}\mathbf{A}^{-1}\mathbf{b} + \mathbf{N}\mathbf{b},$$

skąd

$$\mathbf{I} = \mathbf{M} + \mathbf{N}\mathbf{A},$$
$$\mathbf{M} = \mathbf{I} - \mathbf{N}\mathbf{A}.$$

Otrzymujemy w ten sposób ogólną postać rodziny metod iteracyjnych:

$$\mathbf{x}^{(i+1)} = (\mathbf{I} - \mathbf{N}\mathbf{A})\,\mathbf{x}^{(i)} + \mathbf{N}\mathbf{b}, \tag{2.56}$$

przy czym macierz $\mathbf{N}$ trzeba dobierać w taki sposób, aby $\mathrm{sr}\,(\mathbf{I} - \mathbf{N}\mathbf{A}) < 1$. Dobór macierzy $\mathbf{N}$ nie jest oczywiście łatwy i nie jest arbitralny w każdym przypadku praktycznym – dobieramy metodę iteracyjną spośród dostępnych znanych algorytmów. Rozumowanie powyższe pokazuje natomiast arbitralność w konstrukcji metod iteracyjnych.

□

Miarami efektywności metod iteracyjnych są:

1. *liczba działań arytmetycznych* potrzebnych do przeprowadzenia pojedynczej iteracji $\mathbf{x}^{(i)} \to \mathbf{x}^{(i+1)}$, wielkość zajmowanej pamięci,
2. *szybkość zbieżności*, tzn. szybkość malenia błędu

$$\mathbf{e}^{(i)} = \mathbf{x}^{(i)} - \widehat{\mathbf{x}}.$$

Promień spektralny $\mathrm{sr}(\mathbf{M})$, jeśli jest znany, zawiera informację o szybkości zbieżności, tzn. im $\mathrm{sr}(\mathbf{M})$ mniejszy, tym metoda jest (*asymptotycznie*) szybciej zbieżna.

*Materiał uzupełniajacy.

2.6.1. Metoda Jacobiego

Zdekomponujemy macierz $\mathbf{A}$ następująco:

$$\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U},$$

gdzie $\mathbf{L}$ jest macierzą poddiagonalną, $\mathbf{D}$ diagonalną, a $\mathbf{U}$ naddiagonalną. Dekompozycja taka jest zawsze możliwa, jak pokazuje następujący przykład:

$$\underset{\mathbf{A}}{\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}} = \underset{\mathbf{L}}{\begin{bmatrix} 0 & 0 & 0 \\ 4 & 0 & 0 \\ 7 & 8 & 0 \end{bmatrix}} + \underset{\mathbf{D}}{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 9 \end{bmatrix}} + \underset{\mathbf{U}}{\begin{bmatrix} 0 & 2 & 3 \\ 0 & 0 & 6 \\ 0 & 0 & 0 \end{bmatrix}}.$$

Układ równań $\mathbf{Ax} = \mathbf{b}$ można teraz zapisać w postaci

$$\mathbf{Dx} = -(\mathbf{L} + \mathbf{U})\mathbf{x} + \mathbf{b}.$$

Zakładając, że wartości na diagonali macierzy $\mathbf{A}$ są różne od zera (tzn. macierz $\mathbf{D}$ jest nieosobliwa), można (*intuicyjnie, arbitralnie*) zaproponować metodę iteracyjną

$$\mathbf{D}\mathbf{x}^{(i+1)} = -\left(\mathbf{L} + \mathbf{U}\right)\mathbf{x}^{(i)} + \mathbf{b}, \qquad i = 0, 1, 2, \ldots, \tag{2.57}$$

znaną w literaturze jako *metoda Jacobiego,* często podawaną w postaci równoważnej

$$\mathbf{x}^{(i+1)} = -\mathbf{D}^{-1}\left(\mathbf{L} + \mathbf{U}\right)\mathbf{x}^{(i)} + \mathbf{D}^{-1}\mathbf{b}, \qquad i = 0, 1, 2, \ldots \tag{2.58}$$

Metoda Jacobiego jest schematem obliczeniowym *o strukturze równoległej*, gdyż równanie (2.58) można zapisać w równoważnej postaci n równań skalarnych

$$x_j^{(i+1)} = -\frac{1}{d_{jj}}\left(\sum_{k=1}^{n}(l_{jk} + u_{jk})x_k^{(i)} - b_j\right), \qquad j = 1, 2, \ldots, n, \tag{2.59}$$

które mogą być, w całości lub częściowo (blokowo), realizowane równolegle, w komputerze umożliwiającym zrównoleglenie obliczeń.

Warunkiem dostatecznym zbieżności metody Jacobiego jest silna diagonalna dominacja macierzy $\mathbf{A}$, wierszowa lub kolumnowa, tzn.

1. $|a_{ii}| > \sum\limits_{j=1, j\neq i}^{n} |a_{ij}|, \quad i = 1, 2, \ldots, n$ – dominacja wierszowa,
2. $|a_{jj}| > \sum\limits_{i=1, i\neq j}^{n} |a_{ij}|, \quad j = 1, 2, \ldots, n$ – dominacja kolumnowa.

2.6.2. Metoda Gaussa-Seidela

Niech, tak jak poprzednio, $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$. Układ równań zapiszemy teraz w postaci

$$(\mathbf{L} + \mathbf{D})\mathbf{x} = -\mathbf{U}\mathbf{x} + \mathbf{b},$$

i podobnie jak poprzednio, zaproponujemy (*arbitralnie*) metodę iteracyjną

$$(\mathbf{D} + \mathbf{L})\mathbf{x}^{(i+1)} = -\mathbf{U}\mathbf{x}^{(i)} + \mathbf{b}, \qquad i = 0, 1, 2, ...$$

Jest to metoda zwana *metodą Gaussa-Seidela*. Istotą metody jest fakt, że pojedynczą iterację można zapisać w postaci

$$\mathbf{D}\mathbf{x}^{(i+1)} = -\mathbf{L}\mathbf{x}^{(i+1)} - \mathbf{U}\mathbf{x}^{(i)} + \mathbf{b}, \qquad i = 0, 1, 2, ... \tag{2.60}$$

Występowanie $\mathbf{x}^{(i+1)}$ po obu stronach równości nie przeszkadza, jeśli uwzględnimy określony, szeregowy porządek rozwiązywania, wynikający z postaci macierzy $\mathbf{D}$ i $\mathbf{L}$:

$$\begin{bmatrix} d_{11}x_1^{(i+1)} \\ d_{22}x_2^{(i+1)} \\ d_{33}x_3^{(i+1)} \\ \vdots \\ d_{nn}x_n^{(i+1)} \end{bmatrix} = - \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ l_{21} & 0 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 0 & & 0 \\ \vdots & \vdots & & & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & 0 \end{bmatrix} \begin{bmatrix} x_1^{(i+1)} \\ x_2^{(i+1)} \\ x_3^{(i+1)} \\ \vdots \\ x_n^{(i+1)} \end{bmatrix} - \mathbf{w}^{(i)},$$

gdzie

$$\mathbf{w}^{(i)} = \mathbf{U}\mathbf{x}^{(i)} - \mathbf{b}.$$

Otrzymujemy stąd znowu układ n równań skalarnych. Składowe nowego wektora wyznaczamy kolejno, poczynając od pierwszej:

$$\begin{aligned} x_1^{(i+1)} &= -w_1^{(i)}/d_{11}, \\ x_2^{(i+1)} &= (-l_{21} \cdot x_1^{(i+1)} - w_2^{(i)})/d_{22}, \\ x_3^{(i+1)} &= (-l_{31} \cdot x_1^{(i+1)} - l_{32} \cdot x_2^{(i+1)} - w_3^{(i)})/d_{33}, \\ &\text{itd.} \end{aligned}$$

Schemat metody Gaussa-Seidela nazywany jest *schematem obliczeniowym szeregowym*, gdyż tym razem obliczeń nie można zrównoleglać, muszą być wykonywane szeregowo w określonej kolejności – każde kolejne skalarne równanie korzysta z wyników obliczenia równań poprzednich.

Metoda jest zbieżna, jeśli $\mathbf{A}$ jest silnie diagonalnie dominująca (wierszowo lub kolumnowo), ponadto dla macierzy symetrycznych warunkiem dostatecznym zbieżności jest dodatnia określność macierzy. Metoda Gaussa-Seidela jest na ogół szybciej zbieżna niż metoda Jacobiego (jeśli obie są zbieżne i porównanie jest możliwe).

2.6.3. Testy stopu

Przy stosowaniu metod iteracyjnych zachodzi konieczność testowania i zdecydowania, kiedy należy przerwać iteracje. Praktycznie można zaproponować następujące kryteria stopu:

1. Badanie różnic kolejnych rozwiązań

$$\left\|\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}\right\| \leq \delta, \tag{2.61}$$

gdzie δ to założona tolerancja. Jednakże, ponieważ przede wszystkim chodzi o spełnienie układu równań z założona dokładnością, w przypadku spełnienia powyższego testu można dodatkowo sprawdzić (jako test wyższego rzędu, obliczeniowo bardziej kosztowny):
2. Sprawdzenie wartości normy (np. euklidesowej) wektora błędu rozwiązania:

$$\left\|\mathbf{A}\mathbf{x}^{(i+1)} - \mathbf{b}\right\| \leq \delta_2, \tag{2.62}$$

gdzie δ_2 to pewna tolerancja. Jeśli test ten nie jest spełniony z założoną dokładnością, to można zmniejszyć wartość δ w (2.61) i powrócić do obliczeń. Oczywiście wartość δ_2 nie może być za mała, możliwa do osiągnięcia dokładność rozwiązania jest ograniczona błędami numerycznymi.

Zadania

1. Wyprowadź wzór określający normę pierwszą macierzy.
2. Wyznacz zależność określającą uwarunkowanie rozwiązania układu równań liniowych $\mathbf{A}\mathbf{x} = \mathbf{b}$ w przypadku jednoczesnego zaburzenia wektora $\mathbf{b}$ i macierzy $\mathbf{A}$.
3. Następujące układy równań:

$$\text{a)}\quad \begin{aligned} 3x_1 &+ x_2 - 2x_3 - x_4 &&= 3,\\ 3x_1 &+ x_2 + 2x_3 + 3x_4 &&= -1,\\ x_1 &+ 5x_2 - 4x_3 - x_4 &&= 3,\\ 2x_1 &- 2x_2 + 2x_3 + 3x_4 &&= -8, \end{aligned}$$

$$\text{b)}\quad \begin{aligned} x_1 &+ 2x_2 - x_4 &&= 9,\\ 2x_1 &+ 3x_2 - x_3 &&= 9,\\ x &+ 4x_2 + 2x_3 - 5x_4 &&= 26,\\ 5x_1 &+ 5x_2 + 2x_3 - 4x_4 &&= 32, \end{aligned}$$

rozwiąż metodą eliminacji Gaussa z częściowym wyborem elementu głównego (w kolumnie), wyznaczając przy tym rozkład LU.

4. Pierwszy układ równań z zadania poprzedniego rozwiąż metodą eliminacji zupełnej (Gaussa-Jordana).

5. Wyznacz rozkład LL^{T} macierzy:

$$\begin{bmatrix} 4 & 2 & 4 \\ 2 & 5 & 10 \\ 4 & 10 & 84 \end{bmatrix}, \quad \begin{bmatrix} 4 & 2 & 8 & 4 \\ 2 & 17 & 16 & 6 \\ 8 & 16 & 29 & 21 \\ 4 & 6 & 21 & 39 \end{bmatrix}.$$

6. Wyznacz rozkład $\mathrm{LDL}^{\mathrm{T}}$ dla macierzy z poprzedniego zadania, stosując algorytm podany w rozdz. 2.4.2.

7. Wykaż, że odwrotność macierzy trójkątnej dolnej (górnej) też jest macierzą trójkątną dolną (górną).

8. Sformułuj ogólne (dla dowolnego wymiaru n macierzy) wzory określające elementy macierzy odwrotnej do macierzy trójkątnej dolnej, zapisz je w języku środowiska MATLAB.

9. Oblicz macierz odwrotną do macierzy

$$\mathbf{A} = \begin{bmatrix} 1 & 4 & 3 \\ 2 & 2 & 1 \\ 1 & 3 & 2 \end{bmatrix},$$

różnymi algorytmami wykorzystującymi rozkład LU:
   - odwracając macierze składowe rozkładu LU,
   - rozwiązując układy równań liniowych.

10. Napisz program rozwiązywania układu równań liniowych o dowolnym (n) wymiarze metodą eliminacji Gaussa z częściowym (kolumnowym) wyborem elementu głównego, w języku środowiska MATLAB. Wykorzystując ten program rozwiąż numerycznie nastepujące układy pięciu równań:

a)
$$\begin{aligned} x_1 + \frac{1}{2}x_2 + \frac{1}{3}x_3 + \frac{1}{4}x_4 + \frac{1}{5}x_5 &= 1, \\ \frac{1}{2}x_1 + \frac{1}{3}x_2 + \frac{1}{4}x_3 + \frac{1}{5}x_4 + \frac{1}{6}x_5 &= 0, \\ \frac{1}{3}x_1 + \frac{1}{4}x_2 + \frac{1}{5}x_3 + \frac{1}{6}x_4 + \frac{1}{7}x_5 &= 0, \\ \frac{1}{4}x_1 + \frac{1}{5}x_2 + \frac{1}{6}x_3 + \frac{1}{7}x_4 + \frac{1}{8}x_5 &= 0, \\ \frac{1}{5}x_1 + \frac{1}{6}x_2 + \frac{1}{7}x_3 + \frac{1}{8}x_4 + \frac{1}{9}x_5 &= 0, \end{aligned}$$

b)
$$\begin{aligned}
x_1 + 0.5x_2 + 0.33333x_3 + 0.25x_4 + 0.2x_5 &= 1,\\
0.5x_1 + 0.33333x_2 + 0.25x_3 + 0.2x_4 + 0.16667x_5 &= 0,\\
0.33333x_1 + 0.25x_2 + 0.2x_3 + 0.16667x_4 + 0.14286x_5 &= 0,\\
0.25x_1 + 0.2x_2 + 0.16667x_3 + 0.14286x_4 + 0.125x_5 &= 0,\\
0.2x_1 + 0.16667x_2 + 0.14286x_3 + 0.125x_4 + 0.11111x_5 &= 0.
\end{aligned}$$

Porównaj wyniki, sprawdź wskaźnik uwarunkowania macierzy.

11. Napisz program rozwiązywania układu równań liniowych o dowolnym (n) wymiarze metodą eliminacji Gaussa z częściowym (kolumnowym) wyborem elementu głównego, w języku środowiska MATLAB. Wykorzystując ten program rozwiąż numerycznie układy równań $\mathbf{Hx} = \mathbf{b}$, gdzie $\mathbf{H} = \{h_{ij}\}$ to macierz Hilberta,
$$h_{ij} = \frac{1}{i+j-1},\ i,j = 1,...,n,$$
zaś $\mathbf{b} = [1\ 1\ \cdots\ 1]^{\mathrm{T}}$, kolejno dla n = 5, 10, 20, 30. Oblicz i porównaj błędy (reszty) rozwiązań.

12.* Określ macierz **M** i wektor **w** liniowego schematu iteracyjnego (2.54), odpowiadające metodom Jacobiego i Gaussa-Seidela. Wyznacz ponadto macierz **N** ogólnego schematu iteracyjnego (2.56) odpowiadającą tym metodom.

*Zadanie dodatkowe.

Rozdział 3

Rozkład QR, wartości własne i szczególne

3.1. Rozkład ortogonalno-trójkątny (QR) macierzy

Przypomnijmy: wektory są *ortonormalne*, jeśli są ortogonalne i każdy z nich jest o długości jednostkowej.

Jeśli macierz $\mathbf{Q} \in \mathbb{R}^{m\times n}$ ($m \geq n$) ma wszystkie *kolumny wzajemnie ortogonalne*, to

$$\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{D}, \tag{3.1}$$

gdzie $\mathbf{D} \in \mathbb{R}^{n\times n}$ jest macierzą diagonalną. Jeśli kolumny macierzy $\mathbf{Q}$ są, dodatkowo, wektorami ortonormalnymi, to $\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{I}$ (ale dla $m > n$, na ogół, $\mathbf{Q}\mathbf{Q}^{\mathrm{T}} \neq \mathbf{I}$).

Macierz kwadratową $\mathbf{Q}$ nazywamy *macierzą ortogonalną*, jeśli

$$\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{I}, \tag{3.2}$$

tzn. jest to macierz ortogonalna, której kolumny są wektorami ortonormalnymi. Bardziej formalnie, definicja powyższa określa macierz ortonormalną, ale w literaturze *powszechnie macierz ortogonalna jest utożsamiana z ortonormalną, przez definiowanie macierzy ortogonalnej własnością* (3.2).

Z definicji (3.2) wynika bezpośrednio, że

$$\mathbf{Q}^{\mathrm{T}} = \mathbf{Q}^{-1}, \tag{3.3}$$

skąd mamy dalej

$$\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{Q}\mathbf{Q}^{\mathrm{T}} = \mathbf{I}. \tag{3.4}$$

Równość ta pokazuje, że macierz ortogonalna to macierz kwadratowa, której kolumny są wektorami wzajemnie ortonormalnymi, oraz wiersze są wektorami wzajemnie ortonormalnymi.

Lemat 3.1. *Jeśli* $\mathbf{Q} \in \mathbb{R}^{n\times n}$ *– macierz o kolumnach będących wektorami ortonormalnymi, to*

$$\|\mathbf{Q}\mathbf{x}\|_2 = \|\mathbf{x}\|_2 . \tag{3.5}$$

Dowód. Z definicji normy drugiej i z własności $\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{I}$, mamy

$$\forall \mathbf{x} \quad \|\mathbf{Q}\mathbf{x}\|_2 = \sqrt{(\mathbf{Q}\mathbf{x})^{\mathrm{T}}\mathbf{Q}\mathbf{x}} = \sqrt{\mathbf{x}^{\mathrm{T}}\mathbf{Q}^{\mathrm{T}}\mathbf{Q}\mathbf{x}} = \sqrt{\mathbf{x}^{\mathrm{T}}\mathbf{x}} = \|\mathbf{x}\|_2 .$$

Wniosek. Dla macierzy otrogonalnej mamy:

$$\|\mathbf{Q}\mathbf{x}\|_2 = \|\mathbf{x}\|_2\,, \quad \left\|\mathbf{Q}^{\mathrm{T}}\mathbf{x}\right\|_2 = \|\mathbf{x}\|_2\,. \tag{3.6}$$

Twierdzenie 3.1. *Każdą macierz* $\mathbf{A}_{m\times n}$ *(*$\mathbf{A} \in \mathbb{R}^{m\times n}$*,* $m \geq n$*) o liniowo niezależnych kolumnach (tzn. o pełnym rzędzie równym n) można przedstawić w postaci następujących rozkładów ortogonalno-trójkątnych:*

1. $$\mathbf{A}_{m\times n} = \bar{\mathbf{Q}}_{m\times n}\bar{\mathbf{R}}_{n\times n}, \tag{3.7}$$
   gdzie $\bar{\mathbf{Q}}_{m\times n}$ *jest macierzą o kolumnach wzajemnie ortogonalnych (ogólnie nieortonormalnych), a* $\bar{\mathbf{R}}_{n\times n}$ *jest macierzą trójkątną górną z jedynkami na diagonali;*

2. $$\mathbf{A}_{m\times n} = \mathbf{Q}_{m\times n}\mathbf{R}_{n\times n}, \tag{3.8}$$
   gdzie $\mathbf{Q}_{m\times n}$ *jest macierzą o kolumnach ortonormalnych, a* $\mathbf{R}_{n\times n}$ *jest macierzą trójkątną górną z dodatnimi elementami na diagonali;*

3. $$\mathbf{A}_{m\times n} = \mathbf{Q}_{m\times m}\left[\begin{array}{c}\mathbf{R}_{n\times n}\\ \mathbf{0}\end{array}\right]_{m\times n} = \left[\mathbf{Q}^1_{m\times n}\ \mathbf{Q}^2_{m\times(m-n)}\right]\left[\begin{array}{c}\mathbf{R}_{n\times n}\\ \mathbf{0}\end{array}\right]_{m\times n}, \tag{3.9}$$
   gdzie $\mathbf{Q}_{m\times m}$ *jest macierzą ortogonalną.*

Dowód – przez konstrukcję rozkładów:
Ad 1. Niech $\mathbf{A} = [\mathbf{a}_1\ \mathbf{a}_2\ \cdots\ \mathbf{a}_n]$. Zortogonalizujemy kolumny $\mathbf{a}_i$ macierzy $\mathbf{A}$ standardowym algorytmem Grama-Schmidta, oznaczając przez $\bar{\mathbf{q}}_i$ wektory zortogonalizowane:

$$\begin{aligned}
\bar{\mathbf{q}}_1 &= \mathbf{a}_1,\ \bar{r}_{11} \stackrel{\mathrm{df}}{=} 1,\\
\bar{\mathbf{q}}_2 &= \mathbf{a}_2 - \frac{\bar{\mathbf{q}}_1^{\mathrm{T}}\mathbf{a}_2}{\bar{\mathbf{q}}_1^{\mathrm{T}}\bar{\mathbf{q}}_1}\bar{\mathbf{q}}_1 = \mathbf{a}_2 - \bar{r}_{12}\bar{\mathbf{q}}_1,\ \bar{r}_{22} \stackrel{\mathrm{df}}{=} 1,\\
\bar{\mathbf{q}}_i &= \mathbf{a}_i - \sum_{j=1}^{i-1}\frac{\bar{\mathbf{q}}_j^{\mathrm{T}}\mathbf{a}_i}{\bar{\mathbf{q}}_j^{\mathrm{T}}\bar{\mathbf{q}}_j}\bar{\mathbf{q}}_j = \mathbf{a}_i - \sum_{j=1}^{i-1}\bar{r}_{ji}\bar{\mathbf{q}}_j,\ \bar{r}_{ii} \stackrel{\mathrm{df}}{=} 1,\ \ i = 3, ..., n.
\end{aligned} \tag{3.10}$$

Równania powyższe można przepisać w postaci:

$$\begin{aligned}
\mathbf{a}_1 &= \bar{r}_{11}\bar{\mathbf{q}}_1,\ \bar{r}_{11} \stackrel{\mathrm{df}}{=} 1,\\
\mathbf{a}_2 &= \bar{r}_{12}\bar{\mathbf{q}}_1 + \bar{r}_{22}\bar{\mathbf{q}}_2 = \sum_{j=1}^{2}\bar{r}_{j2}\bar{\mathbf{q}}_j, \quad \bar{r}_{12} = \frac{\bar{\mathbf{q}}_1^{\mathrm{T}}\mathbf{a}_2}{\bar{\mathbf{q}}_1^{\mathrm{T}}\bar{\mathbf{q}}_1},\ \bar{r}_{22} \stackrel{\mathrm{df}}{=} 1,\\
\mathbf{a}_i &= \sum_{j=1}^{i-1}\bar{r}_{ji}\bar{\mathbf{q}}_j + \bar{r}_{ii}\bar{\mathbf{q}}_i = \sum_{j=1}^{i}\bar{r}_{ji}\bar{\mathbf{q}}_j,\ \bar{r}_{ji} = \frac{\bar{\mathbf{q}}_j^{\mathrm{T}}\mathbf{a}_i}{\bar{\mathbf{q}}_j^{\mathrm{T}}\bar{\mathbf{q}}_j},\ \bar{r}_{ii} \stackrel{\mathrm{df}}{=} 1,\ i = 3, ..., n.
\end{aligned} \tag{3.11}$$

Dalej wykorzystamy znaną własność iloczynu macierzy:

> *i-ta kolumna* $\mathbf{a}_i$ *macierzy iloczynowej* $\mathbf{A} = \mathbf{BC}$ *jest kombinacją liniową kolumn pierwszej macierzy iloczynu* $\mathbf{B}$*, ze współczynnikami będącymi elementami i-tej kolumny drugiej macierzy iloczynu* $\mathbf{C}$*, tzn.*

$$\mathbf{a}_i = \mathbf{Bc_i} = [\mathbf{b}_1\ \mathbf{b}_2\ \cdots\ \mathbf{b}_n] \begin{bmatrix} c_{1i} \\ c_{2i} \\ \vdots \\ c_{ni} \end{bmatrix} = \sum_{j=1}^{n} \mathbf{b}_j c_{ji}.$$

Stosując tę własność do naszego dowodu, można zapisać:

$$\mathbf{A} = [\mathbf{a}_1\ \mathbf{a}_2\ \cdots\ \mathbf{a}_n] = [\bar{\mathbf{q}}_1\ \bar{\mathbf{q}}_2\ \cdots\ \bar{\mathbf{q}}_n] \begin{bmatrix} 1 & \bar{r}_{12} & \dots & \bar{r}_{1n} \\ 0 & 1 & \dots & \bar{r}_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} = \bar{\mathbf{Q}}\bar{\mathbf{R}}, \quad (3.12)$$

gdzie macierz $\bar{\mathbf{Q}} = \bar{\mathbf{Q}}_{m\times n} = [\bar{\mathbf{q}}_1\ \bar{\mathbf{q}}_2\ \cdots\ \bar{\mathbf{q}}_n]$ ma kolumny ortogonalne (ogólnie nieortonormalne), a macierz $\bar{\mathbf{R}} = \bar{\mathbf{R}}_{n\times n}$ jest trójkątna górna z jedynkami na diagonali, co kończy dowód tezy 1.

Ad 2. Jeśli przeprowadzimy normalizację kolumn macierzy $\bar{\mathbf{Q}}$:

$$\mathbf{Q} = \left[\frac{\bar{\mathbf{q}}_1}{\|\bar{\mathbf{q}}_1\|}\ \frac{\bar{\mathbf{q}}_2}{\|\bar{\mathbf{q}}_2\|}\ \cdots\ \frac{\bar{\mathbf{q}}_n}{\|\bar{\mathbf{q}}_n\|} \right], \quad (3.13)$$

oraz dokonamy przekształcenia

$$\mathbf{R} = \mathbf{N}\bar{\mathbf{R}}, \quad (3.14)$$

gdzie

$$\mathbf{N} = \begin{bmatrix} \|\bar{\mathbf{q}}_1\| & 0 & \dots & 0 \\ 0 & \|\bar{\mathbf{q}}_2\| & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \|\bar{\mathbf{q}}_n\| \end{bmatrix}, \quad (3.15)$$

to uzyskamy rozkład (3.8).

Ad 3. Jeśli macierz $\mathbf{Q}_{m\times n}$ uzupełnimy do macierzy kwadratowej $\mathbf{Q}_{m\times m}$ przez dodanie $m-n$ jakichkolwiek kolumn ortonormalnych wzajemnie i z kolumnami macierzy $\mathbf{Q}_{m\times n}$ (tzn. macierz $\mathbf{Q}_{m\times n}$ uzupełnimy o macierz $\mathbf{Q}^2_{m\times(m-n)}$ z (3.9), o podanych własnościach), a macierz $\mathbf{R}_{n\times n}$ uzupełnimy zerami do wymiaru $m\times n$, to uzyskamy rozkład (3.9).

Prosty sposób rozszerzenia macierzy $\mathbf{Q}_{m\times n}$: uzupełnić macierz $\mathbf{A}_{m\times n}$ do macierzy kwadratowej $\mathbf{A}_{m\times m}$ przez dodanie jakichkolwiek $m-n$ kolumn liniowo niezależnych z kolumnami macierzy $\mathbf{A}_{m\times n}$ (i względem siebie), a następnie dokonać ich ortogonalizacji algorytmem Grama-Schmidta i normalizacji. □

Ogólnie, twierdzenie 3.1 jest również słuszne dla dowolnej macierzy $\mathbf{A}_{m\times n}$, *tzn. również dla macierzy o niepełnym rzędzie* $k<n$. Oznacza to, że rozkład QR istnieje dla dowolnej macierzy. Dla macierzy o niepełnym rzędzie można go uzyskać odpowiednio rozszerzając podstawową procedurę Grama-Schmidta wykorzystaną w dowodzie twierdzenia 3.1, czy algorytmami wykorzystującymi odbicia Householdera lub obroty Givensa.

Spośród trzech rozkładów ortogonalno-trójkątnych podanych w twierdzeniu 3.1, najpopularniejszym, zwanym po prostu *rozkładem QR*, jest rozkład (3.9) z macierzą $\mathbf{Q}$ ortogonalną. Rozkład (3.8) zwany jest *rozkładem QR ekonomicznym, wąskim* (*economy-size* (MATLAB), *thin*) i np. w pakiecie MATLAB występuje jako opcja rozkładu (3.9). Oczywiście, dla macierzy kwadratowej te dwa rozkłady są identyczne.

Reasumując, numerycznego obliczenia rozkładu QR macierzy można dokonywać:

1. Dla macierzy o pełnym rzędzie, pokazaną w dowodzie twierdzenia 3.1 metodą ortogonalizacji Grama-Schmidta. Z reguły stosujemy tzw. zmodyfikowany algorytm Grama-Schmidta, który ma lepsze własności numeryczne. Dla ogólnego przypadku macierzy o dowolnym rzędzie algorytm Grama-Schmidta można też zastosować, ale trzeba go odpowiednio zmodyfikować – bardziej zalecane są metody podane poniżej.
2. Metodą odbić Householdera – dostajemy od razu rozkład QR (3.9), w razie potrzeby rozkład wąski (3.8) uzyskujemy przez usunięcie ostatnich $m-n$ kolumn macierzy $\mathbf{Q}_{m\times m}$ i $m-n$ ostatnich wierszy macierzy $\mathbf{R}_{m\times n}$, zob. rozdz. 3.6.
3. Metodą obrotów Givensa – dostajemy od razu rozkład QR (3.9), podobnie jak przy metodzie odbić Householdera, zob. rozdz. 3.5. Metoda zalecana szczególnie dla macierzy rzadkich.

Przykład 3.1. Dokonamy rozkładu QR macierzy:

$$\text{a)}\ \ \mathbf{A}=[\mathbf{a}_1\ \mathbf{a}_2]=\begin{bmatrix}1 & 1\\ 2 & -1\\ -2 & 4\end{bmatrix},$$

$$\text{b)}\ \ \mathbf{B}=[\mathbf{a}_1\ \mathbf{a}_2\ \mathbf{a}_3]=\begin{bmatrix}1 & 1 & 9\\ 2 & -1 & 0\\ -2 & 4 & 0\end{bmatrix}.$$

Ad a). Mamy

$$\begin{aligned}\bar{\mathbf{q}}_1 &= \mathbf{a}_1 = [1\ 2\ -2]^{\mathrm{T}},\\ \bar{\mathbf{q}}_2 &= \mathbf{a}_2 - \frac{\bar{\mathbf{q}}_1^{\mathrm{T}}\mathbf{a}_2}{\bar{\mathbf{q}}_1^{\mathrm{T}}\bar{\mathbf{q}}_1}\bar{\mathbf{q}}_1 = \mathbf{a}_2 - \bar{r}_{12}\bar{\mathbf{q}}_1 = \begin{bmatrix} 1\\ -1\\ 4\end{bmatrix} - \frac{-9}{9}\begin{bmatrix} 1\\ 2\\ -2\end{bmatrix} = \begin{bmatrix} 2\\ 1\\ 2\end{bmatrix}.\end{aligned}$$

Stąd dostajemy (nieunormowany) wąski rozkład (3.7)

$$\bar{\mathbf{Q}}_{\mathbf{A}} = \begin{bmatrix} 1 & 2\\ 2 & 1\\ -2 & 2\end{bmatrix}, \quad \bar{\mathbf{R}}_{\mathbf{A}} = \begin{bmatrix} 1 & -1\\ 0 & 1\end{bmatrix}. \tag{3.16}$$

Po unormowaniu kolumn macierzy $\bar{\mathbf{Q}}_{\mathbf{A}}$ zgodnie z (3.13) - (3.15) dostajemy

$$\mathbf{Q}_{\mathbf{A}} = \begin{bmatrix} \frac{1}{3} & \frac{2}{3}\\ \frac{2}{3} & \frac{1}{3}\\ -\frac{2}{3} & \frac{2}{3}\end{bmatrix}, \quad \mathbf{R}_{\mathbf{A}} = \begin{bmatrix} 3 & -3\\ 0 & 3\end{bmatrix}. \tag{3.17}$$

Ad b). Macierz $\mathbf{B}$ ma dwie pierwsze kolumny takie same jak macierz $\mathbf{A}$. Stąd wystarczy zortogonalizować tylko trzecią jej kolumnę, kontynuując obliczenia z punktu a). Zgodnie z (3.11) mamy

$$\begin{aligned}\bar{\mathbf{q}}_3 &= \mathbf{a}_3 - \frac{\bar{\mathbf{q}}_1^{\mathrm{T}}\mathbf{a}_3}{\bar{\mathbf{q}}_1^{\mathrm{T}}\bar{\mathbf{q}}_1}\bar{\mathbf{q}}_1 - \frac{\bar{\mathbf{q}}_2^{\mathrm{T}}\mathbf{a}_3}{\bar{\mathbf{q}}_2^{\mathrm{T}}\bar{\mathbf{q}}_2}\bar{\mathbf{q}}_2 = \mathbf{a}_3 - \bar{r}_{13}\bar{\mathbf{q}}_1 - \bar{r}_{23}\bar{\mathbf{q}}_2\\ &= \begin{bmatrix} 9\\ 0\\ 0\end{bmatrix} - \frac{9}{9}\begin{bmatrix} 1\\ 2\\ -2\end{bmatrix} - \frac{18}{9}\begin{bmatrix} 2\\ 1\\ 2\end{bmatrix} = \begin{bmatrix} 4\\ -4\\ -2\end{bmatrix}.\end{aligned}$$

Stąd dostajemy rozkład nieunormowany (3.7)

$$\bar{\mathbf{Q}}_{\mathbf{B}} = \begin{bmatrix} 1 & 2 & 4\\ 2 & 1 & -4\\ -2 & 2 & -2\end{bmatrix}, \quad \bar{\mathbf{R}}_{\mathbf{B}} = \begin{bmatrix} 1 & -1 & 1\\ 0 & 1 & 2\\ 0 & 0 & 1\end{bmatrix}. \tag{3.18}$$

Po unormowaniu kolumn macierzy $\bar{\mathbf{Q}}$ zgodnie z (3.13) - (3.15) dostajemy

$$\mathbf{Q}_{\mathbf{B}} = \begin{bmatrix} \frac{1}{3} & \frac{2}{3} & \frac{2}{3}\\ \frac{2}{3} & \frac{1}{3} & -\frac{2}{3}\\ -\frac{2}{3} & \frac{2}{3} & -\frac{1}{3}\end{bmatrix}, \quad \mathbf{R}_{\mathbf{B}} = \begin{bmatrix} 3 & -3 & 3\\ 0 & 3 & 6\\ 0 & 0 & 6\end{bmatrix}. \tag{3.19}$$

Zauważmy, że macierze

$$\mathbf{Q}_{\mathbf{B}} = \begin{bmatrix} \frac{1}{3} & \frac{2}{3} & \frac{2}{3}\\ \frac{2}{3} & \frac{1}{3} & -\frac{2}{3}\\ -\frac{2}{3} & \frac{2}{3} & -\frac{1}{3}\end{bmatrix}, \quad \mathbf{R}_{\mathbf{A}+} = \begin{bmatrix} 3 & -3\\ 0 & 3\\ 0 & 0\end{bmatrix} \tag{3.20}$$

stanowią rozkład QR (3.9) macierzy $\mathbf{A}$. □

Standardowy algorytm Grama-Schmidta wykorzystany w dowodzie twierdzenia 3.1 ma niekorzystne własności numeryczne, znacznie lepsze własności ma tzw. *zmodyfikowany algorytm Grama-Schmidta* [13]. Algorytm ten przeprowadza proces ortogonalizacji w innej kolejności; o ile standardowy algorytm (3.11) ortogonalizuje kolumny macierzy po kolei, to algorytm zmodyfikowany po wyznaczeniu kolejnej kolumny ortogonalnej od razu ortogonalizuje wobec niej wszystkie następne kolumny. Proces tworzenia rozkładu QR tym algorytmem polega więc na tworzeniu ciągu macierzy $\{\mathbf{A} = \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, ..., \mathbf{A}^{(n)} = \bar{\mathbf{Q}}\}$, gdzie $\mathbf{A}^{(i)} = [\bar{\mathbf{q}}_1\ \bar{\mathbf{q}}_2\ \cdots\ \bar{\mathbf{q}}_{i-1}\ \mathbf{a}_i^{(i)}\ \mathbf{a}_{i+1}^{(i)}\ \cdots\ \mathbf{a}_n^{(i)}]$. Używając pętli „for" z notacji języka MATLAB, proces ten można zapisać w postaci:

$$
\begin{array}{l}
\mathbf{A}^{(1)} = \mathbf{A};\\
\texttt{for i = 1:n,}\\
\quad \bar{\mathbf{q}}_i = \mathbf{a}_i^{(i)};\ \ \bar{r}_{ii} = 1;\\
\quad d_i = \bar{\mathbf{q}}_i^{\mathrm{T}}\bar{\mathbf{q}}_i;\\
\quad \texttt{for j = i+1:n}\\
\qquad \bar{r}_{ij} = \frac{\bar{\mathbf{q}}_i^{\mathrm{T}}\mathbf{a}_j^{(i)}}{d_i};\\
\qquad \mathbf{a}_j^{(i+1)} = \mathbf{a}_j^{(i)} - \bar{r}_{ij}\bar{\mathbf{q}}_i;\\
\quad \texttt{end}\\
\texttt{end}
\end{array}
\tag{3.21}
$$

Podana poniżej m-funkcja „qrmgs" (*QR modified Gram-Schmidt*) podaje implementację w języku MATLAB rozkładu QR (wąskiego) macierzy zmodyfikowanym algorytmem Grama-Schmidta (3.21), łącznie z normalizacją (ostatnia pętla „for"), tzn. wyjściem jest rozkład QR wąski (3.8). Program jest przeznaczony dla macierzy prostokątnych o pełnym rzędzie, o elementach zarówno rzeczywistych, jak i zespolonych (zwracamy uwagę, że kolejność wektorów w iloczynie skalarnym liczb zespolonych jest istotna, stąd dla wektorów zespolonych jest istotna w określeniu projekcji wektorów danych współczynnikami $\bar{r}_{ij}$).

```
function [Q,R]=qrmgs(A)
%rozkład QR (wąski) zmodyfikowanym alg. Grama-Schmidta dla
%macierzy mxn (m>=n) o rzędzie n, rzeczywistej lub zespolonej
[m n]=size(A);
Q=zeros(m,n); R=zeros(n,n); d=zeros(1,n);
  %rozkład z kolumnami Q ortogonalnymi (nie ortonormalnymi):
for i=1:n
   Q(:,i)=A(:,i);
   R(i,i)=1;
   d(i)=Q(:,i)'*Q(:,i);
   for j=i+1:n
```

```
        R(i,j)=(Q(:,i)'*A(:,j))/d(i);
        A(:,j)=A(:,j)-R(i,j)*Q(:,i);
    end
end
  %normowanie rozkładu (kolumny Q ortonormalne):
for i=1:n
    dd=norm(Q(:,i));
    Q(:,i)=Q(:,i)/dd;
    R(i,i:n)=R(i,i:n)*dd;
end
```

3.2. Wyznaczanie wartości własnych

3.2.1. Podstawowe definicje i własności (przypomnienie)

Wartości własne macierzy odgrywają ważną role w wielu dziedzinach nauki i techniki, w tym w szczególności w informatyce, w automatyce i robotyce.

Wartości i wektory własne macierzy kwadratowej rzeczywistej $\mathbf{A} \in \mathbb{R}^{n\times n}$ są to takie liczby $\lambda \in \mathbb{C}$ (ogólnie zespolone) i odpowiadające im wektory $\mathbf{v} \in \mathbb{C}^n$, że

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}, \tag{3.22}$$

gdzie λ – wartość własna, $\mathbf{v}$ – odpowiadający jej wektor własny. Wartości i wektory własne spełniają więc równanie

$$(\mathbf{A}-\lambda\mathbf{I})\,\mathbf{v} = 0. \tag{3.23}$$

Stąd λ jest wartością własną macierzy $\mathbf{A}$ wtedy i tylko wtedy, gdy jest pierwiastkiem jej *równania charakterystycznego*

$$\det(\mathbf{A}-\lambda\mathbf{I}) = 0. \tag{3.24}$$

Macierz kwadratowa n-wymiarowa ma dokładnie n wartości własnych i odpowiadających im wektorów własnych. Wartości własne są jednoznaczne, natomiast wektory własne to w istocie *kierunki własne*, gdyż jeśli $\mathbf{v}$ jest wektorem własnym, to również $\alpha\mathbf{v}$, $\alpha \in \mathbb{R}$, jest wektorem własnym odpowiadającym tej samej wartości własnej. Dlatego też z reguły za wektory własne przyjmuje się wektory unormowane (o długości jednostkowej).

Zbiór wszystkich wartości własnych macierzy nazywany jest *widmem macierzy*. Widmo macierzy $\mathbf{A}$ oznaczać będziemy symbolem $\mathrm{sp}(\mathbf{A})$, od angielskiego słowa *spectrum*. Przydatna bywa zależność

$$\lambda \in \mathrm{sp}(\mathbf{A}) \;\Rightarrow\; \forall\alpha \in \mathbb{C}\ (\lambda+\alpha) \in \mathrm{sp}(\mathbf{A}+\alpha\mathbf{I}), \tag{3.25}$$

która wynika wprost z definicji wartości własnej:

$$(\mathbf{A}+\alpha\mathbf{I})\mathbf{v} = \mathbf{A}\mathbf{v}+\alpha\mathbf{I}\mathbf{v} = \lambda\mathbf{v}+\alpha\mathbf{v} = (\lambda+\alpha)\mathbf{v}.$$

Definicja. Macierze $\mathbf{A}$ i $\mathbf{B}$ (kwadratowe, tego samego wymiaru) są *podobne*, jeśli istnieje macierz nieosobliwa $\mathbf{S}$ taka, że

$$\mathbf{S}^{-1}\mathbf{A}\mathbf{S} = \mathbf{B}. \tag{3.26}$$

Macierze podobne mają takie same wartości własne, gdyż

$$\begin{aligned}\det\left(\mathbf{S}^{-1}\mathbf{A}\mathbf{S}-\lambda\mathbf{I}\right) &= \det\left(\mathbf{S}^{-1}\mathbf{A}\mathbf{S}-\lambda\mathbf{S}^{-1}\mathbf{S}\right)\\ &= \det\mathbf{S}^{-1}\left(\mathbf{A}-\lambda\mathbf{I}\right)\mathbf{S}\\ &= \det\mathbf{S}^{-1}\det\left(\mathbf{A}-\lambda\mathbf{I}\right)\det\mathbf{S}\\ &= \det\left(\mathbf{A}-\lambda\mathbf{I}\right).\end{aligned}$$

Twierdzenie 3.2. *Jeśli macierz $\mathbf{A}:\mathbb{R}^n\to\mathbb{R}^n$ jest symetryczna, to wszystkie jej wartości własne i wektory własne są rzeczywiste.*

Dowód. Niech $\lambda\in\mathbb{C}$ będzie dowolną wartością własną macierzy $\mathbf{A}$, a $\mathbf{v}\in\mathbb{C}^n$ odpowiadającym jej wektorem własnym. Mamy ($\bar{v}$ oznacza liczbę sprzężoną z v, $\bar{\mathbf{v}}$ to wektor o elementach będących sprzężonymi elementami wektora $\mathbf{v}$)

$$\bar{\mathbf{v}}^{\mathrm{T}}\mathbf{A}\mathbf{v} = \bar{\mathbf{v}}^{\mathrm{T}}\lambda\mathbf{v} = \lambda\bar{\mathbf{v}}^{\mathrm{T}}\mathbf{v} = \lambda\|\mathbf{v}\|^2.$$

Z drugiej strony, korzystając z symetryczności macierzy $\mathbf{A}$, mamy

$$\bar{\mathbf{v}}^{\mathrm{T}}\mathbf{A}\mathbf{v} = \bar{\mathbf{v}}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{v} = [\mathbf{A}\bar{\mathbf{v}}]^{\mathrm{T}}\mathbf{v} = [\bar{\mathbf{A}\mathbf{v}}]^{\mathrm{T}}\mathbf{v} = [\bar{\lambda}\bar{\mathbf{v}}]^{\mathrm{T}}\mathbf{v} = \bar{\lambda}\bar{\mathbf{v}}^{\mathrm{T}}\mathbf{v} = \bar{\lambda}\|\mathbf{v}\|^2.$$

Wykazaliśmy więc, że $\lambda\|\mathbf{v}\|^2 = \bar{\lambda}\|\mathbf{v}\|^2$, skąd wynika $\lambda=\bar{\lambda}$, czyli wartość własna jest rzeczywista.

Z kolei, z faktu iż macierz $(\mathbf{A}-\lambda\mathbf{I})$ układu równań liniowych $(\mathbf{A}-\lambda\mathbf{I})\mathbf{v}=\mathbf{0}$ jest rzeczywista i osobliwa oraz prawa strona tego układu jest wektorem zerowym, wynika bezpośrednio istnienie niezerowego i rzeczywistego rozwiązania tego układu równań, czyli wektor własny $\mathbf{v}$ jest rzeczywisty, co kończy dowód. □

Można łatwo pokazać, rozumując podobnie jak w dowodzie twierdzenia 3.2, że jeśli wszystkie wartości własne macierzy symetrycznej są różne (tzn. jednokrotne), to wszystkie wektory własne są ortogonalne (ortonormalne). Natomiast, dla wielokrotnych wartości własnych wektory własne (unormowane) są ogólnie niejednoznaczne, mówimy wówczas o *podprzestrzeniach własnych* (różne wektory własne odpowiadające wielokrotnej wartości własnej definiują podprzestrzeń własną tej wartości) – ale można wówczas zawsze wybrać zbiór n ortonormalnych wektorów własnych. Reasumując, *dla każdej macierzy symetrycznej istnieje zbiór n ortonormalnych wektorów własnych, tworzący bazę ortonormalną przestrzeni $\mathbb{R}^n$.*

Twierdzenie 3.3. *Jeśli wszystkie wektory własne macierzy są ortonormalne, to można tę macierz sprowadzić do postaci diagonalnej* $\mathrm{diag}\{\lambda_1, \ldots, \lambda_n\}$*, przez podobieństwo.*

Dowód. Niech $\{\mathbf{v}_1, ..., \mathbf{v}_n\}$ będzie zbiorem wzajemnie ortogonalnych wektorów własnych macierzy $\mathbf{A}$. Z definicji mamy

$$\mathbf{A}\mathbf{v}_i = \lambda_i \mathbf{v}_i, \quad i = 1, ..., n.$$

Definiując macierz

$$\mathbf{V} \stackrel{\mathrm{df}}{=} [\mathbf{v}_1\ \mathbf{v}_2\ \cdots\ \mathbf{v}_n],$$

możemy powyższy zbiór n równości zapisać równoważnie równaniem macierzowym

$$\mathbf{AV} = \mathbf{V}\,\mathrm{diag}\,(\lambda_i)$$

Po pomnożeniu tego równania stronami przez $\mathbf{V}^{\mathrm{T}}$ otrzymujemy

$$\mathbf{V}^{\mathrm{T}}\mathbf{AV} = \mathbf{V}^{\mathrm{T}}\mathbf{V}\,\mathrm{diag}\{\lambda_i\}. \tag{3.27}$$

Ponieważ wektory $\mathbf{v}_1, ..., \mathbf{v}_n$ są ortonormalne, więc

$$\mathbf{v}_i^{\mathrm{T}}\mathbf{v}_j = 0 \quad \text{dla } i \neq j,$$

$$\mathbf{v}_i^{\mathrm{T}}\mathbf{v}_i = 1; \qquad i, j = 1, ..., n.$$

Stąd

$$\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I},$$

czyli macierz $\mathbf{V}$ jest ortogonalna, $\mathbf{V}^{\mathrm{T}} = \mathbf{V}^{-1}$. Równanie (3.27) można więc zapisać w postaci

$$\mathbf{V}^{-1}\mathbf{AV} = \mathrm{diag}\{\lambda_i\},$$

co kończy dowód. □

Wniosek. Każdą macierz symetryczną można sprowadzić przez podobieństwo do macierzy diagonalnej $\mathrm{diag}\{\lambda_1, \ldots, \lambda_n\}$, a macierzą przekształcenia jest macierz, której kolumnami są ortonormalne wektory własne macierzy wyjściowej.

Przechodząc do przypadku ogólnego, czyli również *macierzy niesymetrycznych*, musimy rozważyć również wartości i wektory własne zespolone, stąd również *macierze o elementach zespolonych*.

Niech $\mathbf{A}^*$ oznacza transpozycję i sprzężenie elementów zespolonych macierzy kwadratowej zespolonej $\mathbf{A}$, tzn. $\mathbf{A}^* = \bar{\mathbf{A}}^{\mathrm{T}}$. Macierz $\mathbf{A}$ kwadratowa o elementach ogólnie zespolonych jest:

– *hermitowska*, gdy $\mathbf{A} = \mathbf{A}^*$,

– *unitarna*, gdy $\mathbf{A}^*\mathbf{A} = \mathbf{I}$ $(\text{tzn. } \mathbf{A}^* = \mathbf{A}^{-1})$.

Macierz kwadratowa o elementach zespolonych $\mathbf{A} \in \mathbb{R}^{n\times n}$ jest *normalna*, jeśli

$$\mathbf{A}^*\mathbf{A} = \mathbf{A}\mathbf{A}^*. \tag{3.28}$$

Jeśli macierz jest normalna, *to istnieje zbiór jej n ortonormalnych wektorów własnych* (istnieje zbiór wektorów własnych tworzący bazę ortonormalną przestrzeni $\mathbb{R}^n$). Dlatego też takie macierze są diagonalizowalne i słuszne jest dla nich uogólnienie twierdzenia 3.3. Oczywiście, wszystkie macierze hermitowskie, tj. symetryczne w przypadku macierzy o elementach rzeczywistych, są normalne.

Możliwość diagonalizacji macierzy niesymetrycznych nie jest więc cechą powszechną, natomiast *dla dowolnej macierzy* (też niesymetrycznej, też o elementach zespolonych) słuszne jest następujące twierdzenie Schura:

Twierdzenie 3.4. *Każda macierz kwadratowa n-wymiarowa* $\mathbf{A}$ *jest podobna do macierzy górnej trójkątnej (ogólnie zespolonej)* $\mathbf{R}$*, zaś macierz przekształcenia* $\mathbf{U}$ *jest macierzą unitarną,*

$$\mathbf{U}^*\mathbf{A}\mathbf{U} = \mathbf{U}^{-1}\mathbf{A}\mathbf{U} = \begin{bmatrix} \lambda_1 & x & \cdots & x & x \\ 0 & \lambda_2 & \cdots & x & x \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \lambda_n \end{bmatrix} = \mathbf{R},$$

gdzie x *oznacza element niezerowy.*

Wybrane metody wyznaczania wartości własnych:

- *Metody wyznacznikowe*, wykorzystujące fakt, że wartości własne są zerami wielomianu charakterystycznego – efektywne dla pojedynczej czy niewielu wartości własnych.
- *Metoda Jacobiego* – diagonalizacja macierzy symetrycznej za pomocą ciągu obrotów Givensa (efektywne do wymiaru $n \approx 10$), zob. rozdz. 3.5.2.
- *Metoda QR* – najbardziej ogólna, efektywna.

Metody wyznacznikowe wyznaczania wartości własnych

Teoretycznie wartości własne można obliczać z definicji, znajdując je jako pierwiastki (zera) wielomianu charakterystycznego, to znaczy rozwiązania równania $\det(\lambda\mathbf{I} - \mathbf{A}) = 0$. Znajdujemy kolejno pojedyncze pierwiastki wielomianu charakterystycznego odpowiednią metodą ogólną, z reguły po sprowadzeniu macierzy do prostszej postaci (Hessenberga lub trójdiagonalnej). Metody realizujące tę strategię, to np. (zob np. [1]):

1. Metoda *bisekcji wykorzystująca ciągi Sturma* – dla macierzy symetrycznych. Pojedynczą wartość własną znajdujemy metodą bisekcji (zob. rozdz. 6.1.1), decyzje o wyborze jednego z pododcinków po każdym podziale podejmuje się łatwo, korzystając z własności tzw. wielomianów Sturma. Macierz wyjściową

należy najpierw sprowadzić do *trójdiagonalnej*, tzn. mającej elementy niezerowe jedynie na diagonali, naddiagonali i poddiagonali (jest to postać Hessenberga macierzy symetrycznych), zob. np. rozdz. 3.6, rozdz. 3.5.

2. Metoda *Hymana* – pierwiastki wielomianu charakterystycznego znajdujemy metodą Newtona (zob. rozdz. 6.1.4), wykorzystując specyficzny sposób obliczania wartości wielomianu i jego pochodnej. Macierz wyjściową należy najpierw sprowadzić do *postaci Hessenberga*, tzn. prawie trókątnej górnej, poniżej diagonali zawierającej elementy niezerowe tylko na poddiagonali, zob. np. rozdz. 3.6.

3.2.2. Metoda QR znajdowania wartości własnych

Metoda opiera się na sukcesywnym wykorzystywaniu rozkładu QR macierzy.

Metoda QR dla macierzy symetrycznych

Dla zwiększenia efektywności przed przystąpieniem do obliczeń zaleca się sprowadzić daną macierz do *postaci trójdiagonalnej*. Nie jest to niezbędne, metoda działa też dla macierzy nieprzekształconych, tyle że wówczas każda iteracja wymaga większego nakładu obliczeń – większy jest nakład obliczeń na rozkład QR macierzy, na mnożenie macierzy. Natomiast rozkład QR zachowuje postać trójdiagonalną.

Idea pojedynczego kroku metody QR (przekształcenie $\mathbf{A}^{(k)}$ do $\mathbf{A}^{(k+1)}$):

$$\begin{aligned} \mathbf{A}^{(k)} &= \mathbf{Q}^{(k)}\mathbf{R}^{(k)} \quad \text{(faktoryzacja)}, \\ \mathbf{A}^{(k+1)} &= \mathbf{R}^{(k)}\mathbf{Q}^{(k)}. \end{aligned}$$

Ponieważ $\mathbf{Q}^{(k)}$ jest ortogonalna, więc

$$\mathbf{R}^{(k)} = (\mathbf{Q}^{(k)})^{-1}\mathbf{A}^{(k)} = \mathbf{Q}^{(k)\mathrm{T}}\mathbf{A}^{(k)},$$

skąd mamy

$$\mathbf{A}^{(k+1)} = \mathbf{Q}^{(k)\mathrm{T}}\mathbf{A}^{(k)}\mathbf{Q}^{(k)},$$

czyli macierz $\mathbf{A}^{(k+1)}$ jest przekształconą przez podobieństwo macierzą $\mathbf{A}^{(k)}$, a więc ma te same watości własne.

Algorytm podstawowy (metoda QR bez przesunięć):

$$\begin{aligned} \mathbf{A}^{(1)} &= \mathbf{A}, \\ \mathbf{A}^{(1)} &= \mathbf{Q}^{(1)}\mathbf{R}^{(1)} \quad \text{(faktoryzacja)}, \end{aligned}$$

$$\mathbf{A}^{(2)} = \mathbf{R}^{(1)}\mathbf{Q}^{(1)} \ \ (= \mathbf{Q}^{(1)\mathrm{T}}\mathbf{A}^{(1)}\mathbf{Q}^{(1)}),$$
$$\mathbf{A}^{(2)} = \mathbf{Q}^{(2)}\mathbf{R}^{(2)} \quad \text{(faktoryzacja)},$$
$$\mathbf{A}^{(3)} = \mathbf{R}^{(2)}\mathbf{Q}^{(2)} \ \ (= \mathbf{Q}^{(2)\mathrm{T}}\mathbf{A}^{(2)}\mathbf{Q}^{(2)}) = \mathbf{Q}^{(2)\mathrm{T}}\mathbf{Q}^{(1)\mathrm{T}}\mathbf{A}^{(1)}\mathbf{Q}^{(1)}\mathbf{Q}^{(2)}),$$
itd.
$$\mathbf{A}^{(k)} \longrightarrow \ \ \mathbf{V}^{-1}\mathbf{A}\mathbf{V} = \mathrm{diag}\,\{\lambda_\mathrm{i}\}\,.$$

Reasumując, dla macierzy $\mathbf{A}$ symetrycznej, macierz $\mathbf{A}^{(k)}$ zbiega do macierzy diagonalnej $\mathrm{diag}\,(\lambda_i)$.

Szybkość zbieżności: jeśli macierz $\mathbf{A}$ ma n wartości własnych o różnych modułach,

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n| > 0,$$

to można pokazać, że

$$a_{i,i}^{(k)} \longrightarrow_{k\to\infty} \lambda_i, \ \ i-1,...,n,$$

$$a_{i+1,i}^{(k)} \longrightarrow_{k\to\infty} 0, \ \ i = 1,...,n-1$$

i zbieżność elementu poddiagonalnego $a_{i+1,i}^{(k)}$ do zera jest liniowa z ilorazem zbieżności $\left|\frac{\lambda_{i+1}}{\lambda_i}\right|$, czyli

$$\frac{|a_{i+1,i}^{(k+1)}|}{|a_{i+1,i}^{(k)}|} \approx \left|\frac{\lambda_{i+1}}{\lambda_i}\right|.$$

Stąd, jeśli wartości własne leżą blisko siebie, to metoda jest wolno zbieżna. Aby poprawić szybkość zbieżności, stosuje się algorytm metody QR z przesunięciami.

Pojedyncza iteracja algorytmu metody QR z przesunięciami:

$$\begin{aligned}\mathbf{A}^{(k)} - p_k\mathbf{I} &= \mathbf{Q}^{(k)}\mathbf{R}^{(k)},\\ \mathbf{A}^{(k+1)} &= \mathbf{R}^{(k)}\mathbf{Q}^{(k)} + p_k\mathbf{I}\\ &= \mathbf{Q}^{(k)\mathrm{T}}(\mathbf{A}^{(k)} - p_k\mathbf{I})\mathbf{Q}^{(k)} + p_k\mathbf{I}\\ &= \mathbf{Q}^{(k)\mathrm{T}}\mathbf{A}^{(k)}\mathbf{Q}^{(k)},\end{aligned}$$

gdyż macierze $\mathbf{Q}^{(k)}$ są ortogonalne. Zbieżność jest wtedy liniowa z ilorazem

$$\frac{|\lambda_{i+1} - p_k|}{|\lambda_i - p_k|},$$

skąd wynika, że najlepszym przesunięciem p_k byłoby aktualne przybliżenie wartości własnej λ_{i+1}. Praktycznie można zastosować postępowanie następujące: jeśli

$$\mathbf{A}^{(k)} = \left[\begin{array}{ccccc} d_1^{(k)} & \cdot & \cdot & \cdot & 0 \\ \cdot & \ddots & & & \vdots \\ \cdot & & d_{n-2}^{(k)} & e_{n-2}^{(k)} & 0 \\ \cdot & & e_{n-2}^{(k)} & d_{n-1}^{(k)} & e_{n-1}^{(k)} \\ \cdot & \cdot & 0 & e_{n-1}^{(k)} & d_n^{(k)} \end{array}\right] \tag{3.29}$$

(rozważamy postać trójdiagonalną macierzy przekształcanej), to za p_k przyjmujemy wartość $d_n^{(k)}$ lub, *co jest lepszym rozwiązaniem*, bliższą $d_n^{(k)}$ wartość własną podmacierzy 2×2 z prawego dolnego rogu macierzy $\mathbf{A}^{(k)}$, tzn. macierzy

$$\left[\begin{array}{cc} d_{n-1}^{(k)} & e_{n-1}^{(k)} \\ e_{n-1}^{(k)} & d_n^{(k)} \end{array}\right],$$

itd., aż $d_n^{(k)} = \lambda_n$ – czego wskaźnikiem jest $e_{n-1}^{(k)} = 0$, zob. np. [11].

W przypadku operowania na macierzy pełnej (nietrójdiagonalnej), wskaźnikiem równości $d_n^{(k)} = \lambda_n$ jest wyzerowanie wszystkich elementów ostatniego wiersza, oczywiście poza elementem diagonalnym (ostatnim).

Następnie postępujemy analogicznie z macierzą zmniejszoną do wymiarowości $(n-1)\times(n-1)$, przez odrzucenie ostatniego wiersza i ostatniej kolumny (tzw. deflacja), itd. Podsumowujemy ten schemat poniżej, podając strukturę algorytmu z przesunięciami.

Struktura algorytmu QR z przesunięciami

1. Znajdujemy wartość własną λ_n jako bliższą $d_n^{(k)}$ wartość własną podmacierzy 2×2 z prawego dolnego rogu macierzy $\mathbf{A}^{(k)}$ (procedurą opisaną powyżej).
2. Opuszczamy ostatni wiersz i ostatnią kolumnę aktualnej macierzy $\mathbf{A}^{(k)}$ (deflacja), tzn. uwzględniamy dalej podmacierz $\mathbf{A}_{n-1}^{(k)}$ (porównaj z macierzą pełną (3.29)).
$$\mathbf{A}_{n-1}^{(k)} = \left[\begin{array}{cccc} d_1^{(k)} & \cdot & \cdot & 0 \\ \cdot & \ddots & & \\ \cdot & & d_{n-2}^{(k)} & e_{n-2}^{(k)} \\ \cdot & \cdot & e_{n-2}^{(k)} & d_{n-1}^{(k)} \end{array}\right].$$
3. Znajdujemy następną wartość własną, λ_{n-1}, w analogiczny sposób, tzn. przekształcamy macierz $\mathbf{A}_{n-1}^{(k)}$ aż do uzyskania $e_{n-2}^{(k)} = 0$. W przypadku operowania

na macierzy pełnej (nietrójdiagonalnej), iterujemy aż do wyzerowania wszystkich elementów ostatniego wiersza, poza elementem diagonalnym $d_{n-1}^{(k)}$).

4. Opuszczamy ostatni wiersz i kolumnę aktualnej macierzy $\mathbf{A}_{n-1}^{(k)}$ (deflacja), itd.

$$\vdots$$

aż do znalezienia wszystkich wartości własnych.

Zarys metody QR dla macierzy niesymetrycznych

Podobnie jak w przypadku macierzy symetrycznych macierz wyjściową zaleca się przekształcić wstępnie do *postaci Hessenberga* – nie jest to konieczne, ale zmniejsza nakład obliczeniowy, gdyż rozkład QR macierzy o strukturze Hessenberga wymaga mniej obliczeń niż macierzy pełnej, a jest on wykonywany w każdej iteracji, oraz macierz przekształcona w wyniku pojedynczej iteracji zachowuje strukturę Hessenberga.

Konstrukcja algorytmu QR dla macierzy dowolnych (tj. ogólnie niesymetrycznych) pozostaje w istocie taka sama jak dla macierzy symetrycznych, musimy tylko wziąć pod uwagę możliwość wystąpienia *wartości własnych zespolonych*. Stąd, istotne jest wybieranie przesunięcia zawsze jako wartości własnej podmacierzy 2×2 z prawego dolnego rogu aktualnej macierzy, wartości najbliższej ostatniemu elementowi diagonalnemu aktualnej macierzy, oraz operowanie arytmetyką zespoloną. Macierz przekształcana zbiega na ogół do macierzy trókątnej górnej z wartościami własnymi na diagonali (twierdzenie 3.4), jedynie w szczególnych przypadkach do macierzy diagonalnej.

Można uniknąć arytmetyki zespolonej, stosując algorytm operujący jedynie na macierzach rzeczywistych, jego podstawą jest odpowiednie wyznaczanie i zmienianie przesunięć co dwa kroki algorytmu – mamy tu odpowiednie wzory i programy [9].

Metoda QR dla macierzy niesymetrycznych, w swych podstawowych wersjach omówionych w tym rozdziale, nie zawsze jest zbieżna. Na przykład, dla macierzy

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

algorytm metody QR bez przesunięć jest niezbieżny, gdyż

$$\mathbf{A} = \mathbf{QR} = \mathbf{AI}, \text{ czyli } \mathbf{A} = \mathbf{A}^{(1)} = \mathbf{A}^{(2)} = \mathbf{A}^{(3)} = \cdots$$

Ponadto, algorytm z przesunięciami też jest tu niezbieżny (nic nie zmienia), gdyż wartości własne podmacierzy 2×2 z prawego dolnego rogu macierzy $\mathbf{A}$ są zerowe. W takich przypadkach skutecznym sposobem jest wstępne przekształcenie macierzy przez podobieństwo.

Przykład 3.2. Porównamy liczby iteracji potrzebne do obliczenia wartości własnych przykładowych macierzy, metodą QR bez przesunięć i z przesunięciami. Dla uproszczenia programów, macierze nie będą wstępnie przekształcane do postaci Hessenberga (postaci trójdiagonalnej dla macierzy symetrycznych).

Do obliczania wartości własnych macierzy symetrycznych metodą QR bez przesunięć wykorzystano następujący prosty algorytm, napisany w języku środowiska MATLAB:

```
function eigenvalues=EigvalQRNoShift(D,tol,imax)
%tol -tolerancja (górna granica wartości) elementów zerowanych
%imax - maksymalna liczba iteracji
n=size(D,1);
i=1;
while i <= imax & max(max(D-diag(diag(D)))) > tol
   [Q1,R1]=qr(D);
   D=R1*Q1;  %macierz przekształcona
   i=i+1;
end
if i > imax
      error('imax exceeded program terminated');
   end
eigenvalues=diag(D);
```

Natomiast zastosowany algorytm metody QR z przesunięciami podany jest poniżej:

```
function eigenvalues=EigvalQRshifts(A,tol,imax)
%tol - tolerancja (górna granica wartości) elementów zerowanych
%imax - max liczba iteracji dla liczenia jednej wart. własnej
n=size(A,1);
eigenvalues=diag(zeros(n));
INITIALsubmatrix=A; %macierz początkowa (oryginalna)
for k=n:-1:2,
   DK=INITIALsubmatrix; %macierz startowa dla jednej wart. wł.
   i=0;
   while i <= imax  &  max(abs(DK(k,1:k-1))) > tol
      DD=DK(k-1:k,k-1:k); %2x2 podmacierz dolnego prawego rogu
      [ev1,ev2]=quadpolynroots(1,-(DD(1,1)+DD(2,2)),...
                          ...DD(2,2)*DD(1,1)-DD(2,1)*DD(1,2));
      if abs(ev1-DD(2,2)) < abs(ev2-DD(2,2)),
          shift=ev1; %najbliższa DK(k,k) wart. własna podm. DD
      else
```

```
            shift=ev2; %najbliższa DK(k,k) wart. własna podm. DD
        end
        DK=DK-eye(k)*shift; %macierz przesunięta
        [Q1,R1]=qr(DK); %factoryzacja QR
        DK=R1*Q1+eye(k)*shift; %macierz przekształcona
        i=i+1;
    end
    if i > imax
        error('imax exceeded program terminated');
    end
    eigenvalues(k)=DK(k,k);
    if k > 2,
        INITIALsubmatrix=DK(1:k-1,1:k-1); %deflacja macierzy
    else
        eigenvalues(1)=DK(1,1); %ostatnia wartość własna
    end
end
```

W podanych programach do faktoryzacji QR macierzy zastosowano procedurę „qr" MATLAB-a, gdyż m-funkcja „qrmgs" opisana uprzednio w rozdz. 3.1 jest tylko dla macierzy o pełnym rzędzie. W drugim programie, do liczenia wartości własnych podmacierzy 2×2 z prawego dolnego rogu macierzy przekształcanej napisano m-funkcję „quadpolynroots", obliczającą pierwiastki wielomianu drugiego stopnia o współczynnikach będących jej argumentami (dla właściwej organizacji tego algorytmu, zob. przykład 1.5).

Program „EigvalQRshifts" w zastosowaniu do macierzy symetrycznych operuje na liczbach rzeczywistych, wyznaczając wszystkie wartości własne (rzeczywiste). Może on być też zastosowany do macierzy niesymetrycznych, jeśli operować będziemy na liczbach zespolonych. MATLAB w razie potrzeby przełącza się automatycznie na arytmetykę liczb zespolonych, tzn. jeśli pojawi się w czasie obliczeń zespolona wartość własna – a taka możliwość istnieje przy obliczaniu wartości własnych macierzy niesymetrycznej 2×2 funkcją „quadpolynroots".

Obliczenia wykonano dla kilku macierzy niesymetrycznych wymiarów 5×5 i 10×10 generowanych przez instrukcję „rand" MATLABa, następnie przekształcanych w macierze symetryczne przez dodanie macierzy transponowanej. Wyniki, w postaci liczb iteracji liczonych jako liczby wywołania instrukcji „qr" (rozkładu QR), zamieszczono w tabeli 3.1, dla trzech przykładowych macierzy każdego wymiaru i rodzaju. W ostatniej kolumnie tabeli podano ponadto liczby zespolonych wartości własnych. Zaprezentowane wyniki pokazują wyraźnie, że algorytm QR z przesunięciami jest znacznie efektywniejszy i pewniejszy. Nie tylko wymaga

on znacznie mniejszego nakładu obliczeń, ale w pewnych przypadkach jedynie on umożliwia uzyskanie rozwiązania.

Tabela 3.1. Liczby iteracji algorytmów QR bez przesunięć i z przesunięciami

tolerancja	tol=0.00001		tol=0.0000001		liczba zespolonych
alg. z przesunięciami	nie	tak	nie	tak	wartości własnych
macierz symetryczna	35	7	47	8	0
5×5	154	6	niezb.	23	0
	91	8	124	9	0
macierz symetryczna	93	16	129	16	0
10×10	132	24	193	76	0
	niezb.	15	niezb.	18	0
macierz niesymetr.	–	7	–	8	2
5×5	–	55	–	78	4
	–	10	–	11	2
macierz niesymetr.	–	24	–	27	6
10×10	–	44	–	54	4
	–	26	–	45	4

□

3.3. Wartości szczególne, dekompozycja SVD

Rozważmy macierz $\mathbf{A} \in \mathbb{R}^{m\times n}$, gdzie $m \geq n$. Macierz $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ ma wymiar $n\times n$ i jest symetryczna dodatnio półokreślona, tzn.

$$\forall \lambda \in \mathrm{sp}\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right) \quad \lambda_i \geq 0.$$

Stąd, jej wartości własne można przedstawić w postaci

$$\lambda_i = (\sigma_i)^2, \qquad \text{gdzie} \quad \sigma_i \geq 0, \qquad i = 1, ..., n.$$

Liczby σ_i nazywamy *wartościami szczególnymi* lub *singularnymi* (*singular values*) macierzy $\mathbf{A}$.

Wniosek. Jeśli macierz kwadratowa $\mathbf{A}$ jest symetryczna i dodatnio półokreślona, to jej wartości własne są wartościami szczególnymi.

Jeśli macierz jest tylko symetryczna (nie jest dodatnio półokreślona), to

$$\sigma_i = |\lambda_i|, \quad \lambda_i \in \mathrm{sp}\,(\mathbf{A}), \qquad i = 1, ..., n.$$

Bardzo ważny dla zastosowań jest rozkład macierzy wg wartości szczególnych, zwany *dekompozycją SVD* (*singular value decomposition*). Dekompozycja SVD jest sformułowana w twierdzeniu podanym poniżej.

Twierdzenie 3.5. *Dla dowolnej macierzy* $\mathbf{A}_{m\times n}$ *(*$\mathbf{A} \in \mathbb{R}^{m\times n}$, $m \geq n$*) istnieją macierze ortonormalne* $\mathbf{U}_{m\times m}$, $\mathbf{V}_{n\times n}$ *i macierz* $\mathbf{\Sigma}_{m\times n}$ *takie, że:*

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}},$$

gdzie

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{D}_{n\times n} \\ \mathbf{0}_{(m-n)\times n} \end{bmatrix}_{m\times n} = \left[\begin{array}{cccccc} \sigma_1 & 0 & \cdots & & 0 \\ 0 & \ddots & & & \\ \vdots & & \sigma_k & & \vdots \\ & & & \ddots & 0 \\ 0 & & \cdots & 0 & \sigma_n \\ \hline & & \mathbf{0} & & \end{array}\right],$$

- $\mathbf{D}_{n\times n}$= diag $(\sigma_i,\ i = 1, ..., n)$, *gdzie* $\sigma_i \geq 0$. *Jeśli występują* σ_i *zerowe, to* $\mathbf{D}$ *jest osobliwa (a* $\mathbf{A}$ *jest niepełnego rzędu – liczba niezerowych wartości szczególnych jest równa rzędowi* $k \leq n$ *macierzy),*
- $\mathbf{V}_{m\times m}$ *jest macierzą, której kolumnami są ortonormalne wektory własne macierzy* $\mathbf{A}^{\mathrm{T}}\mathbf{A}$,
- $\mathbf{U}_{n\times n}$ *jest macierzą, której kolumnami są ortonormalne wektory własne macierzy* $\mathbf{A}\mathbf{A}^{\mathrm{T}}$.

Uwaga. Numeryczne wyznaczanie σ_i przez obliczanie wartości własnych macierzy $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ *grozi utratą dokładności*, co pokazuje prosty przykład podany poniżej.

Przykład 3.3. Policzymy wartości szczególne macierzy $\mathbf{A}$, korzystając z definicji, dokładnie oraz numerycznie dla danej precyzji maszynowej eps, gdzie

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ \varepsilon & 0 \\ 0 & \varepsilon \end{bmatrix},\ \varepsilon < \sqrt{eps}.$$

Dokładne wartości szczególne tej macierzy to

$$\sigma_1(\mathbf{A}) = \sqrt{2+\varepsilon^2}, \quad \sigma_2(\mathbf{A}) = |\varepsilon|.$$

Mamy natomiast

$$\mathbf{A}^{\mathrm{T}}\mathbf{A} = \begin{bmatrix} 1+\varepsilon^2 & 1 \\ 1 & 1+\varepsilon^2 \end{bmatrix},$$
$$fl\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right) = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \text{ gdyż } \varepsilon^2 < eps.$$

Stąd, wartości szczególne $\mathbf{A}$ liczone jako pierwiastki wartości własnych macierzy $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ w arytmetyce zmiennopozycyjnej wynoszą $\widetilde{\sigma}_1 = \sqrt{2}$, $\widetilde{\sigma}_2 = 0$, gdzie tylda ($\widetilde{\ }$) oznacza wartości numeryczne. Ale przecież $\sigma_2(\mathbf{A}) = |\varepsilon|$, co nie zgadza się z dokładnością maszynową. Na przykład, dla $eps = 10^{-12}$, $|\varepsilon| = 10^{-7} < \sqrt{eps} = 10^{-6}$ powinniśmy dostać $\sigma_2(\mathbf{A}) = 10^{-7}$, a nie 0. □

Stosuje się algorytmy SVD *nieprowadzące do utraty dokładności* (np. algorytm Goluba-Reinscha [11]). Dekompozycja SVD ma kluczowe znaczenie w wielu zadaniach numerycznych, w tym wymagających dużej precyzji, np. stosuje się ją do określania rzędu macierzy, przy rozwiązywaniu liniowego zadania najmniejszych kwadratów z macierzą niepełnego rzędu.

3.4. Liniowe zadanie najmniejszych kwadratów

Liniowe zadanie najmniejszych kwadratów (LZNK) można sformułować następująco:
Dla danych macierzy $\mathbf{A} \in \mathbb{R}^{m\times n}$ $(m > n)$ *i wektora* $\mathbf{b} \in \mathbb{R}^m$ *znaleźć wektor* $\widehat{\mathbf{x}} \in \mathbb{R}^n$ *taki, że*

$$\forall \mathbf{x} \in \mathbb{R}^n \qquad \|\mathbf{b} - \mathbf{A}\widehat{\mathbf{x}}\|_2 \leq \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2 . \tag{3.30}$$

LZNK to zadanie polegające na rozwiązaniu układu m równań liniowych z n zmiennymi, gdzie $n < m$ (układ nadokreślony), w sensie minimalizacji normy drugiej wektora niespełnienia równań (wektora błędu, wektora reszt).

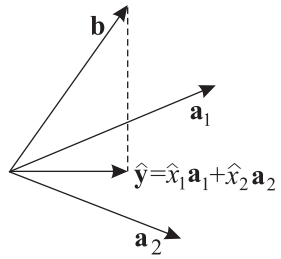
Interpretacja geometryczna: $\mathbf{A}\widehat{\mathbf{x}}$ jest rzutem ortogonalnym wektora $\mathbf{b}$ na podprzestrzeń rozpinaną przez kolumny macierzy $\mathbf{A} = [\mathbf{a}_1\ \mathbf{a}_2\ \cdots\ \mathbf{a}_n]$, tzn. na podprzestrzeń

$$\mathbb{Y} \subset \mathbb{R}^m = \{\mathbf{y} \in \mathbb{R}^m :\ \mathbf{y} = \mathbf{A}\mathbf{x} = x_1\mathbf{a}_1 + \cdots + x_n\mathbf{a}_n, \mathbf{x} = [x_1\ \cdots\ x_n]^{\mathrm{T}} \in \mathbb{R}^n\},$$

co jest dla przypadku $n = 2$ zilustrowane na rysunku 3.1. Zwróćmy uwagę, że norma błędu (niespełnienia układu równań) może być zerowa jedynie wówczas, kiedy wektor prawej strony $\mathbf{b}$ jest liniowo zależny od kolumn macierzy $\mathbf{A}$, tzn. należy do podprzestrzeni $\mathbb{Y}$, co na ogół dla $m > n$ zdarza się rzadko.

Zadanie LZNK może mieć niejednoznaczne rozwiązanie, dla ujednoznacznienia wymaga się zwykle albo wektora rozwiązania $\widehat{\mathbf{x}}$ o najmniejszej normie, tzn.

$$(\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2 = \|\mathbf{b} - \mathbf{A}\widehat{\mathbf{x}}\|_2) \Rightarrow \|\widehat{\mathbf{x}}\|_2 \leq \|\mathbf{x}\|_2 , \tag{3.31}$$



Rys. 3.1. Interpretacja geometryczna rozwiązania $\widehat{\mathbf{x}} = [\widehat{x}_1\ \widehat{x}_2]^{\mathrm{T}}$ zadania LZNK dla $n = 2$

albo wektora o największej liczbie zerowych składników. Zauważmy ponadto, że LNZK jest zadaniem równoważnym minimalizacji następującej funkcji kwadratowej:

$$J(x) = (\mathbf{b} - \mathbf{A}\mathbf{x})^{\mathrm{T}}(\mathbf{b} - \mathbf{A}\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x} - 2\mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{b} + \mathbf{b}^{\mathrm{T}}\mathbf{b}. \tag{3.32}$$

Metody rozwiązywania LZNK

Oznaczmy przez k rząd macierzy $\mathbf{A} \in \mathbb{R}^{m\times n}$, $k \leq n$.

Rozważmy najpierw przypadek, gdy $k = n$, tzn. *macierz* $\mathbf{A}$ *jest pełnego rzędu*. Wówczas symetryczna i kwadratowa macierz $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ jest dodatnio określona (ma wszystkie wartości własne dodatnie), stąd funkcja $J(x)$ jest ściśle wypukła i ma jednoznaczne minimum w punkcie spełniającym warunek konieczny minimum – zerowanie gradientu funkcji:

$$J'(x)^{\mathrm{T}} = 2\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x} - 2\mathbf{A}^{\mathrm{T}}\mathbf{b} = \mathbf{0}.$$

Wynikający stąd układ równań liniowych

$$\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x} = \mathbf{A}^{\mathrm{T}}\mathbf{b} \tag{3.33}$$

nazywany jest *układem równań normalnych* i ma on w rozważanym przypadku jednoznaczne rozwiązanie. Jednakże dla słabo uwarunkowanej macierzy $\mathbf{A}$ wskaźnik uwarunkowania macierzy $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ staje się jeszcze bardziej niekorzystny, gdyż

$$\mathrm{cond}_2(\mathbf{A}^{\mathrm{T}}\mathbf{A}) = (\mathrm{cond}_2\mathbf{A})^2 = (\frac{\sigma_1}{\sigma_n})^2,$$

gdzie σ_1 i σ_n to największa i najmniejsza wartość szczególna macierzy $\mathbf{A}$. Wówczas zalecane jest skorzystanie z rozkładu $\mathbf{QR}$ macierzy $\mathbf{A}$, przy czym wygodnie jest skorzystać z rozkładu wąskiego (3.8), $\mathbf{A}_{m\times n} = \mathbf{Q}_{m\times n}\mathbf{R}_{n\times n}$. Układ równań normalnych (3.33) możemy wówczas zapisać w postaci

$$\mathbf{R}^{\mathrm{T}}\mathbf{Q}^{\mathrm{T}}\mathbf{Q}\mathbf{R}\mathbf{x} = \mathbf{R}^{\mathrm{T}}\mathbf{Q}^{\mathrm{T}}\mathbf{b}. \tag{3.34}$$

Ze względu na ortonormalność kolumn macierzy $\mathbf{Q}$ mamy $\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{I}$. Uwzględniając ponadto nieosobliwość macierzy $\mathbf{R}$ (gdyż $k = n$), dostajemy następujący dobrze określony układ równań liniowych

$$\mathbf{R}\mathbf{x} = \mathbf{Q}^{\mathrm{T}}\mathbf{b}. \tag{3.35}$$

Zauważmy, że jeśli rozkład QR wyznaczamy jedynie dla rozwiązania LZNK, to można wykorzystać w tym celu rozkład wąski nieznormalizowany $\bar{\mathbf{Q}}\bar{\mathbf{R}}$ (3.7), wówczas zamiast równania (3.35) należy rozwiązać równanie

$$\bar{\mathbf{R}}\mathbf{x} = (\bar{\mathbf{Q}}^{\mathrm{T}}\bar{\mathbf{Q}})^{-1}\bar{\mathbf{Q}}^{\mathrm{T}}\mathbf{b}. \tag{3.36}$$

W równaniu tym wyrażenie $(\bar{\mathbf{Q}}^{\mathrm{T}}\bar{\mathbf{Q}})^{-1}$ jest bardzo łatwe do policzenia, gdyż

$$\bar{\mathbf{Q}}^{\mathrm{T}}\bar{\mathbf{Q}} = \mathrm{diag}\{d_1, ..., d_n\},$$

gdzie wartości $d_i = \bar{\mathbf{q}}_i^{\mathrm{T}}\bar{\mathbf{q}}_i$ są wyznaczane podczas rozkładu QR metodą ortogonalizacji Grama-Schmidta, zob. (3.21).

Rozważymy teraz przypadek, gdy $k < n$, tzn. *macierz* $\mathbf{A}$ *jest niepełnego rzędu*. Wówczas do rozwiązania LZNK zalecany jest algorytm oparty na rozkładzie SVD. Stosując do macierzy $\mathbf{A}$ rozkład SVD i następnie lemat 3.1, mamy

$$\begin{aligned}\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2 &= \left\|\mathbf{b} - \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}\mathbf{x}\right\|_2 \\ &= \left\|\mathbf{U}^{\mathrm{T}}\mathbf{b} - \mathbf{\Sigma}\left(\mathbf{V}^{\mathrm{T}}\mathbf{x}\right)\right\|_2 \\ &= \left\|\widetilde{\mathbf{b}} - \mathbf{\Sigma}\widetilde{\mathbf{x}}\right\|_2,\end{aligned}$$

gdzie $\widetilde{\mathbf{b}} = \mathbf{U}^{\mathrm{T}}\mathbf{b}$, $\widetilde{\mathbf{x}} = \mathbf{V}^{\mathrm{T}}\mathbf{x}$.

Oznaczmy $\widetilde{\mathbf{b}} = [\widetilde{\mathbf{b}}_1^{\mathrm{T}}\ \widetilde{\mathbf{b}}_2^{\mathrm{T}}]^{\mathrm{T}}$, $\widetilde{\mathbf{b}}_1 \in \mathbb{R}^n, \widetilde{\mathbf{b}}_2 \in \mathbb{R}^{m-n}$, wówczas

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2 = \left\|\widetilde{\mathbf{b}} - \mathbf{\Sigma}\widetilde{\mathbf{x}}\right\|_2 = \left\| \begin{array}{c} \widetilde{\mathbf{b}}_1 - \mathbf{D}\widetilde{\mathbf{x}} \\ \widetilde{\mathbf{b}}_2 \end{array} \right\|_2 = \left\| \begin{array}{c} \widetilde{b}_1 - \sigma_1\widetilde{x}_1 \\ \vdots \\ \widetilde{b}_k - \sigma_k\widetilde{x}_k \\ \widetilde{b}_{k+1} \\ \vdots \\ \widetilde{b}_m \end{array} \right\|_2,$$

gdzie $k \leq n$ jest liczbą niezerowych wartości szczególnych (rzędem macierzy $\mathbf{A}$). Stąd norma $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2$ osiąga minimum równe $\sqrt{\sum_{i=k+1}^{m}(\widetilde{b}_i)^2}$, dla każdego $\widetilde{\mathbf{x}}$ postaci:

$$\widetilde{\mathbf{x}} = \left[\begin{array}{c} \widetilde{x}_1 = \widetilde{b}_1/\sigma_1 \\ \vdots \\ \widetilde{x}_k = \widetilde{b}_k/\sigma_k \\ \widetilde{x}_{k+1}\ \text{dowolne} \\ \vdots \\ \widetilde{x}_n\ \text{dowolne} \end{array} \right]. \tag{3.37}$$

Rozwiązanie jednoznaczne $\widehat{\widetilde{\mathbf{x}}}$ uzyskujemy przyjmując

$$\widehat{\widetilde{\mathbf{x}}} = \begin{bmatrix} \widetilde{b}_1/\sigma_1 \\ \vdots \\ \widetilde{b}_k/\sigma_k \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{3.38}$$

co można zapisać w postaci

$$\widehat{\widetilde{\mathbf{x}}} = \mathbf{\Sigma}^{+}\widetilde{\mathbf{b}}, \text{ gdzie } \mathbf{\Sigma}^{+} = \left[\mathbf{D}^{+}_{n\times n} \ \mathbf{0}_{n\times(m-n)}\right], \ \mathbf{D}^{+} = \operatorname{diag}\left\{\frac{1}{\sigma_1}, ..., \frac{1}{\sigma_{\mathrm{k}}}, 0, ..., 0\right\}.$$

Stąd mamy

$$\mathbf{V}^{\mathrm{T}}\widehat{\mathbf{x}} = \mathbf{\Sigma}^{+}\mathbf{U}^{\mathrm{T}}\mathbf{b},$$

czyli

$$\widehat{\mathbf{x}} = \mathbf{V}\mathbf{\Sigma}^{+}\mathbf{U}^{\mathrm{T}}\mathbf{b}, \tag{3.39}$$

gdzie macierz

$$\mathbf{A}^{+} = \mathbf{V}\mathbf{\Sigma}^{+}\mathbf{U}^{\mathrm{T}} \tag{3.40}$$

jest nazywana *macierzą pseudoodwrotną* macierzy $\mathbf{A}$.

Przykład 3.4. Należy rozwiązać układ równań $\mathbf{A}\mathbf{x} = \mathbf{b}$, w sensie minimalizacji sumy kwadratów niespełnienia równań, gdzie

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 2 & -1 \\ -2 & 4 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 3 \\ 1 \\ 1 \end{bmatrix}.$$

Macierz $\mathbf{A}$ jest pełnego rzędu (kolumny są liniowo niezależne), stąd możemy znaleźć rozwiązanie LZNK przez rozwiązanie układu równań normalnych (3.33), lub z wykorzystaniem rozkładu QR.

Układ równań normalnych ma postać

$$\begin{bmatrix} 9 & -9 \\ -9 & 18 \end{bmatrix}\mathbf{x} = \begin{bmatrix} 3 \\ 6 \end{bmatrix},$$

skąd dostajemy rozwiązanie $\mathbf{x} = [\frac{4}{3}\ 1]^{\mathrm{T}}$.

Do rozwiązania metodą rozkładu QR wystarczy rozkład wąski, rozkład taki dla podanej macierzy $\mathbf{A}$ wyliczyliśmy w przykładzie 3.1, stąd dostajemy do rozwiązania układ równań

$$\mathbf{R}\mathbf{x} = \mathbf{Q}^{\mathrm{T}}\mathbf{b} \quad \Leftrightarrow \quad \begin{bmatrix} 3 & -3 \\ 0 & 3 \end{bmatrix}\mathbf{x} = \begin{bmatrix} 1 \\ 3 \end{bmatrix},$$

który oczywiście prowadzi do tego samego rozwiązania.

Zwróćmy uwagę, że układ równań $\mathbf{Rx} = \mathbf{Q}^{\mathrm{T}}\mathbf{b}$ jest łatwiejszy do rozwiązania niż układ równań normalnych, gdyż jest to układ równań z macierzą trójkątną. Co więcej, macierz $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ jest gorzej uwarunkowana od macierzy $\mathbf{R}$. Natomiast trzeba najpierw wyznaczyć rozkład QR, a w przypadku stosowania układu równań normalnych wyliczyć iloczyn macierzy $\mathbf{A}^{\mathrm{T}}\mathbf{A}$. □

3.5. Przekształcenie Givensa, z zastosowaniami*

3.5.1. Przekształcenie (obrót) Givensa

Macierz Givensa obrotu o kąt θ w płaszczyźnie wersorów $\mathbf{e}_p$ i $\mathbf{e}_q$ jest postaci:

$$\mathbf{G}_{pq} = \begin{bmatrix} 1 & 0 & & & & \cdots & & & & 0 \\ 0 & \ddots & & & & & & & & \\ & & 1 & & & & & & & \\ & & & c & 0 & \cdots & 0 & s & & \\ & & & 0 & 1 & & 0 & 0 & & \\ \vdots & & & \vdots & & \ddots & & \vdots & & \vdots \\ & & & 0 & 0 & & 1 & 0 & & \\ & & & -s & 0 & \cdots & 0 & c & & \\ & & & & & & & & 1 & \\ & & & & & & & & \ddots & 0 \\ 0 & & & & & \cdots & & & 0 & 1 \end{bmatrix} \begin{matrix} \\ \\ \\ \text{wiersz } p \\ \\ \\ \\ \text{wiersz } q \\ \\ \\ \end{matrix} \tag{3.41}$$

$$\begin{matrix} & & \vdots & & & \vdots & & \\ & & \text{kolumna } p & & & \text{kolumna } q & & \end{matrix}$$

gdzie:

$$c = \cos\theta,\ s = \sin\theta,$$

tzn. macierz ta różni się od macierzy jednostkowej jedynie czterema elementami $(c, s, -s, c)$ ulokowanymi w wierszach i kolumnach o indeksach p i q, jak to przedstawiono w (3.41). Ponieważ $c^2 + s^2 = 1$, więc $\mathbf{G}_{pq}\left(\mathbf{G}_{pq}\right)^{\mathrm{T}} = \mathbf{I}$, tj.macierz $\mathbf{G}_{pq}$ jest ortogonalna.

Jeśli przekształcamy wektor $\mathbf{a}$,

$$\mathbf{b} = \mathbf{G}_{pq}\mathbf{a},$$

*Materiał uzupełniający.

to odpowiednie wzory dla współrzędnych są następujące:

$$\begin{aligned} b_j &= a_j \qquad \text{dla } j \neq p, q \ , \\ b_p &= ca_p + sa_q, \\ b_q &= -sa_p + ca_q. \end{aligned}$$

Wyzerowanie współrzędnej b_q wektora $\mathbf{b}$ obrotem Givensa:

$$\begin{aligned} b_q &= -sa_p + ca_q = 0, \\ 1 &= s^2 + c^2, \end{aligned}$$

skąd uzyskujemy:

$$c = \frac{a_p}{\sqrt{a_p^2 + a_q^2}}, \quad s = \frac{a_q}{\sqrt{a_p^2 + a_q^2}} \ ,$$

przy czym $c = s = 0$, gdy $a_p = a_q = 0$.

Zauważmy, że w wyniku działań:

$\mathbf{G}_{pq}\mathbf{A}$, $(\mathbf{G}_{pq})^{\mathrm{T}}\,\mathbf{A}$ – zmienione zostają tylko wiersze p-ty i q-ty w $\mathbf{A}$,

$\mathbf{A}\mathbf{G}_{pq}$, $\mathbf{A}\,(\mathbf{G}_{pq})^{\mathrm{T}}$ – zmienione zostają tylko kolumny p-ta i q-ta w $\mathbf{A}$.

Przekształcenie obrotami Givensa zachowujące podobieństwo macierzy

Ze względu na ortogonalność macierzy obrotu $\mathbf{G}_{pq}$, przekształcenie (dwustronne) $\mathbf{A}' = (\mathbf{G}_{pq})^{\mathrm{T}}\,\mathbf{A}\mathbf{G}_{pq}$ zachowuje podobieństwo macierzy. Zauważmy, że macierz $\mathbf{A}'$ różni się od $\mathbf{A}$ *jedynie elementami w wierszach i kolumnach o indeksach p i q*. Łatwo wyprowadzić odpowiednie relacje:

$$\begin{aligned} a'_{ij} &= a_{ij} && \text{dla } i, j \neq p, q, \\ a'_{pj} &= ca_{pj} - sa_{qj} && \text{dla } j \neq p, q, \\ a'_{qj} &= sa_{pj} + ca_{qj} && \text{dla } j \neq p, q, \\ a'_{jp} &= ca_{jp} - sa_{jq} && \text{dla } j \neq p, q, \\ a'_{jq} &= sa_{jp} + ca_{jq} && \text{dla } j \neq p, q, \end{aligned}$$

$$\begin{aligned} a'_{pp} &= (ca_{pp} - sa_{qp})c - s\,(ca_{pq} - sa_{qq}) \\ &= c^2 a_{pp} + s^2 a_{qq} - sc\,(a_{pq} + a_{qp})\,, \\ a'_{qq} &= (sa_{pp} + ca_{qp})s + c\,(sa_{pq} + ca_{qq}) \\ &= s^2 a_{pp} + c^2 a_{qq} + sc\,(a_{pq} + a_{qp})\,, \\ a'_{pq} &= c^2 a_{pq} - s^2 a_{qp} + sc\,(a_{pp} - a_{qq})\,, \\ a'_{qp} &= c^2 a_{qp} - s^2 a_{pq} + sc\,(a_{pp} - a_{qq})\,. \end{aligned}$$

W przypadku macierzy $\mathbf{A}$ *symetrycznej* relacje te redukują się do:

$$\begin{aligned}
a'_{ij} &= a_{ij} \qquad \text{dla } i,j \neq p,q,\\
a'_{pj} &= a'_{jp} = ca_{jp} - sa_{jq} \qquad \text{dla } j \neq p,q,\\
a'_{qj} &= a'_{jq} = sa_{jp} + ca_{jq} \qquad \text{dla } j \neq p,q,\\
a'_{pp} &= c^2 a_{pp} + s^2 a_{qq} - 2sca_{pq},\\
a'_{qq} &= s^2 a_{pp} + c^2 a_{qq} + 2sca_{pq},\\
a'_{pq} &= a'_{qp} = \left(c^2 - s^2\right) a_{pq} + sc\left(a_{pp} - a_{qq}\right).
\end{aligned}$$

Na przekształceniach Givensa zachowujących podobieństwo macierzy (dwustronnych) oparta jest metoda Jacobiego znajdowania wartości i wektorów własnych macierzy symetrycznej – przez kolejne zerowanie elementów położonych poza diagonalą (patrz następny rozdział), czy algorytm przekształcenia macierzy symetrycznej do macierzy podobnej trójdiagonalnej – przez kolejne zerowanie elementów położonych poza diagonalą, naddiagonalą i poddiagonalą. Przekształcenie Givensa można też wykorzystać do wyznaczania rozkładu QR macierzy (zob. rozdz. 3.5.3).

3.5.2. Metoda Jacobiego wyznaczania wartości własnych macierzy symetrycznej

Jest to metoda wyznaczania wartości własnych (i wektorów własnych, jeśli potrzeba) macierzy symetrycznej $\mathbf{A} \in \mathbb{R}^{n\times n}$, polegająca na przekształcaniu macierzy do postaci diagonalnej ciągiem obrotów Givensa, zachowujących podobieństwo:

$$\mathbf{A}' = \mathbf{G}_{pq}^{\mathrm{T}} \mathbf{A} \mathbf{G}_{pq}.$$

W pojedynczej iteracji, polegającej na zerowaniu elementu a_{pq}, zmianie podlegają jedynie elementy zawierające indeksy p i q (zob. rozdz. 3.5.1):

$$\begin{aligned}
a'_{jp} &= ca_{jp} - sa_{jq}, \qquad j = 1,...,n, \quad j \neq p,q,\\
a'_{jq} &= ca_{jq} + sa_{jp}, \qquad j = 1,...,n, \quad j \neq p,q,\\
a'_{pp} &= c^2 a_{pp} + s^2 a_{qq} - 2sca_{pq},\\
a'_{qq} &= s^2 a_{pp} + c^2 a_{qq} + 2sca_{pq},\\
a'_{pq} &= \left(c^2 - s^2\right) a_{pq} + sc\left(a_{pp} - a_{qq}\right).
\end{aligned}$$

W metodzie Jacobiego żądamy: $a'_{pq} = 0$ (z symetrii wynika $a'_{qp} = 0$), czyli

$$\left(c^2 - s^2\right) a_{pq} + sc\left(a_{pp} - a_{qq}\right) = 0.$$

Stąd

$$\mathrm{ctg}2\theta = \frac{c^2 - s^2}{2sc} = \frac{a_{qq} - a_{pp}}{2a_{pq}}, \quad |\theta| \leq \frac{\pi}{4}, \quad \theta = \pm\frac{\pi}{4} \text{ dla } a_{pp} = a_{qq}.$$

Oznaczając (zob. [9])

$$t = \mathrm{tg}\,\theta = \frac{s}{c},$$
$$\alpha = \mathrm{ctg}\,2\theta,$$

otrzymujemy równanie dla t:

$$t^2 + 2\alpha t - 1 = 0.$$

Rozwiązaniem uwzględnianym jest mniejszy co do modułu pierwiastek:

$$t = \frac{\mathrm{sgn}\,(\alpha)}{|\alpha| + \sqrt{\alpha^2 + 1}},$$

lub

$$t = \frac{1}{2\alpha}, \text{ gdy } \alpha \gg 1 \quad - \text{ dla uniknięcia nadmiaru.}$$

Z definicji $c = \frac{1}{\sqrt{1+t^2}}$, $s = tc$. Określając ponadto $r = \frac{s}{1+c}$, dostajemy stabilniejsze numerycznie wzory:

$$\begin{aligned}
a'_{jp} &= a_{jp} - s\,(a_{jq} + r a_{jp})\,, \qquad j = 1, ..., n, \;\; j \neq p, q,\\
a'_{jq} &= a_{jq} + s\,(a_{jp} - r a_{jq})\,, \qquad j = 1, ..., n, \;\; j \neq p, q,\\
a'_{pp} &= a_{pp} - t a_{pq},\\
a'_{qq} &= a_{qq} + t a_{pq}.
\end{aligned}$$

Definiując

$$S = \sum_{i,j=1, i\neq j}^{n} |a_{ij}|^2, \;\; S' = \sum_{i,j=1, i\neq j}^{n} |a'_{ij}|^2,$$

można pokazać, że

$$S' = S - 2|a_{pq}|^2,$$

tzn. suma modułów elementów pozadiagonalnych zmniejsza się w każdym kroku, metoda jest więc zbieżna. Macierz przekształcenia będąca iloczynem macierzy obrotów zbiega do macierzy, której kolumnami są wektory własne.

W oryginalnej metodzie Jacobiego indeksy p, q wybiera się w każdym kroku tak, aby wskazywały element pozadiagonalny o największym module. Praktycznie stosuje się tzw. *cykliczną metodę Jacobiego*. Polega ona na tym, że zeruje się elementy pozadiagonalne po kolei, w określonym porządku – np. wierszami. Typowe macierze wymagają od 6 do 10 cykli, całkowity nakład obliczeń jest rzędu od $12n^3$ do $20n^3$. Metoda Jacobiego jest w praktyce efektywna dla macierzy o wymiarze do ok. 10×10.

3.5.3. Rozkład QR macierzy obrotami Givensa

Niech będzie dana macierz $\mathbf{A} \in \mathbb{R}^{n\times n}$. Zastosujemy następującą *notację*:
x – elementy nieprzekształcone,
$*$ – elementy przekształcone (zmienione).

Pierwszy krok: $n-1$ obrotów Givensa $\mathbf{G}_{1j}$, $j = 2, ..., n$, dających w efekcie macierz przekształcenia pierwszego kroku $\mathbf{G}^{(1)}$ (indeks górny określa numer kroku):

$$\mathbf{G}^{(1)}\mathbf{A} = \mathbf{G}_{1n} \cdots \mathbf{G}_{13}\mathbf{G}_{12}\mathbf{A}.$$

Obroty te zerują elementy pierwszej kolumny poza pierwszym – kolejno element drugi, trzeci, itd., tzn.:

$$\mathbf{G}^{(1)}\mathbf{A} = \mathbf{G}_{1n} \cdots \mathbf{G}_{13} \left(\mathbf{G}_{12} \begin{bmatrix} \mathrm{x} & \mathrm{x} & \mathrm{x} & \\ \mathrm{x} & \mathrm{x} & \mathrm{x} & \\ \mathrm{x} & \mathrm{x} & \mathrm{x} & \\ \mathrm{x} & \mathrm{x} & \mathrm{x} & \cdots \\ \vdots & \vdots & \vdots & \\ \mathrm{x} & \mathrm{x} & \mathrm{x} & \end{bmatrix} \right)$$

$$= \mathbf{G}_{1n} \cdots \mathbf{G}_{13} \begin{bmatrix} * & * & * & \\ 0 & * & * & \\ \mathrm{x} & \mathrm{x} & \mathrm{x} & \\ \mathrm{x} & \mathrm{x} & \mathrm{x} & \cdots \\ \vdots & \vdots & \vdots & \\ \mathrm{x} & \mathrm{x} & \mathrm{x} & \end{bmatrix} = \cdots = \begin{bmatrix} * & * & * & \\ 0 & * & * & \\ 0 & * & * & \\ 0 & * & * & \cdots \\ \vdots & \vdots & \vdots & \\ 0 & * & * & \end{bmatrix}.$$

Drugi krok: transformacje zerujące kolejno elementy drugiej kolumny, poczynając od trzeciego:

$$\mathbf{G}^{(2)} \left(\mathbf{G}^{(1)}\mathbf{A} \right) = \mathbf{G}_{2n} \cdots \mathbf{G}_{24}\mathbf{G}_{23} \left(\mathbf{G}^{(1)}\mathbf{A} \right),$$

$$\mathbf{G}^{(2)} \left(\mathbf{G}^{(1)}\mathbf{A} \right) = \mathbf{G}_{2n} \cdots \mathbf{G}_{24}\mathbf{G}_{23} \left(\begin{bmatrix} \mathrm{x} & \mathrm{x} & \mathrm{x} & \\ 0 & \mathrm{x} & \mathrm{x} & \\ 0 & \mathrm{x} & \mathrm{x} & \\ 0 & \mathrm{x} & \mathrm{x} & \cdots \\ \vdots & \vdots & \vdots & \\ 0 & \mathrm{x} & \mathrm{x} & \end{bmatrix} \right) = \begin{bmatrix} \mathrm{x} & \mathrm{x} & \mathrm{x} & \\ 0 & * & * & \\ 0 & 0 & * & \\ 0 & 0 & * & \cdots \\ \vdots & \vdots & \vdots & \\ 0 & 0 & * & \end{bmatrix},$$

i analogicznie w kolejnych krokach, aż do

$$\mathbf{G}^{(n-1)} \cdots \mathbf{G}^{(2)}\mathbf{G}^{(1)}\mathbf{A} = \bar{\mathbf{Q}}\mathbf{A} = \mathbf{R},$$

gdzie $\mathbf{R}$ to macierz trójkątna górna, a $\bar{\mathbf{Q}}$ ortogonalna, czyli $\mathbf{A} = \mathbf{QR}$, gdzie macierz $\mathbf{Q} = \bar{\mathbf{Q}}^{\mathrm{T}}$ jest ortogonalna.

Nakład obliczeń dla wyznaczenia macierzy $\mathbf{R}$ jest rzędu $\frac{4}{3}n^3$ mnożeń i $\frac{2}{3}n^3$ dodawań, oraz $\frac{1}{2}n^2$ pierwiastkowań.

3.6. Przekształcenie Householdera, z zastosowaniami*

3.6.1. Przekształcenie (odbicie) Householdera

Przekształceniem (odbiciem) Householdera nazywamy przekształcenie zdefiniowane macierzą $\mathbf{P}$ postaci

$$\mathbf{P} = \mathbf{I} - 2\mathbf{w}\mathbf{w}^{\mathrm{T}}, \qquad \text{gdzie } \mathbf{w} \in \mathbb{R}^n,\ \|\mathbf{w}\| = 1.$$

Własności:

$$\mathbf{P} = \mathbf{P}^{\mathrm{T}} \quad (\text{symetria}),$$
$$\mathbf{P}\mathbf{P}^{\mathrm{T}} = \mathbf{I} \quad (\text{tzn. } \mathbf{P} = \mathbf{P}^{-1}, \text{ortogonalność}),$$

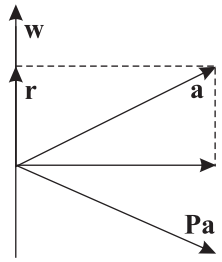
gdyż

$$\left(\mathbf{I} - 2\mathbf{w}\mathbf{w}^{\mathrm{T}}\right)\left(\mathbf{I} - 2\mathbf{w}\mathbf{w}^{\mathrm{T}}\right) = \mathbf{I} - 4\mathbf{w}\mathbf{w}^{\mathrm{T}} + 4\mathbf{w}\left(\mathbf{w}^{\mathrm{T}}\mathbf{w}\right)\mathbf{w}^{\mathrm{T}} = \mathbf{I}\,.$$

Interpretacja geometryczna:

$$\mathbf{y} = \mathbf{P}\mathbf{a} = \left(\mathbf{I} - 2\mathbf{w}\mathbf{w}^{\mathrm{T}}\right)\mathbf{a} = \mathbf{a} - 2\left(\mathbf{w}^{\mathrm{T}}\mathbf{a}\right)\mathbf{w} = \mathbf{a} - 2\mathbf{r},$$

gdzie $\mathbf{r}$ to rzut prostopadły $\mathbf{a}$ na $\mathbf{w}$. To znaczy, że $\mathbf{Pa}$ jest odbiciem zwierciadlanym wektora $\mathbf{a}$ względem płaszczyzny prostopadłej do wektora $\mathbf{w}$, zob. rys. 3.2. Odpowiednio dobierając $\mathbf{w}$, można uzyskać dowolne położenie wektora $\mathbf{y} = \mathbf{Pa}$.



Rys. 3.2. Interpretacja geometryczna odbicia Householdera

*Materiał uzupełniający.

Najczęstszym zastosowaniem przekształcenia Householdera jest dokonanie obrotu danego wektora $\mathbf{a} \in \mathbb{R}^n$ tak, aby uzyskany wektor $\mathbf{Pa} \in \mathbb{R}^n$ był równoległy do pierwszego wersora układu współrzednych $\mathbf{e}_1 = [1\,0\,0\,\cdots\,0]^{\mathrm{T}} \in \mathbb{R}^n$.

W tym celu przedstawiamy $\mathbf{P}$ jako

$$\mathbf{P} = \mathbf{I} - \frac{1}{K}\mathbf{u}\mathbf{u}^{\mathrm{T}}, \tag{3.42}$$

gdzie

$$\mathbf{u} = \mathbf{a} \pm \|\mathbf{a}\|\,\mathbf{e}_1, \quad K = \frac{1}{2}\|\mathbf{u}\|^2,$$

zaś znak dobieramy tak, aby uzyskać większą normę $\mathbf{u}$, tzn.

$$\mathbf{u} = \mathbf{a} + \operatorname{sgn}(a_1)\,\|\mathbf{a}\|\,\mathbf{e}_1,$$

gdzie

$$a_1 = (\mathbf{e}_1)^{\mathrm{T}}\,\mathbf{a}.$$

Wówczas:

$$\begin{aligned}
\mathbf{Pa} &= \mathbf{a} - \frac{\mathbf{u}}{K}\left(\mathbf{a} \pm \|\mathbf{a}\|\,\mathbf{e}_1\right)^{\mathrm{T}}\mathbf{a} \\
&= \mathbf{a} - \frac{2\mathbf{u}\left(\|\mathbf{a}\|^2 \pm \|\mathbf{a}\|\,a_1\right)}{\|\mathbf{a} \pm \|\mathbf{a}\|\,\mathbf{e}_1\|^2} \\
&= \mathbf{a} - \frac{2\mathbf{u}\left(\|\mathbf{a}\|^2 \pm \|\mathbf{a}\|\,a_1\right)}{\|\mathbf{a}\|^2 + \|\mathbf{a}\|^2 \pm 2\,\|\mathbf{a}\|\,a_1} \\
&= \mathbf{a} - \mathbf{u} = \mp\,\|\mathbf{a}\|\,\mathbf{e}_1.
\end{aligned}$$

W praktyce, nie należy obliczać elementów macierzy $\mathbf{P}$, lecz wynik działania tej macierzy na wektor:

$$\mathbf{Pa} = \left(\mathbf{I} - 2\mathbf{w}\mathbf{w}^{\mathrm{T}}\right)\mathbf{a} = \mathbf{a} - (2\mathbf{w}^{\mathrm{T}}\mathbf{a})\mathbf{w}. \tag{3.43}$$

Korzystając z przedstawienia (3.42), mamy tę zależność w postaci

$$\mathbf{Pa} = \left(\mathbf{I} - \frac{1}{K}\mathbf{u}\mathbf{u}^{\mathrm{T}}\right)\mathbf{a} = \mathbf{a} - \left(\frac{\mathbf{u}^{\mathrm{T}}\mathbf{a}}{K}\right)\mathbf{u}. \tag{3.44}$$

W zastosowaniach dokonujemy często przekształcenia Householdera całych macierzy (zob. np. rozdz. 3.6.2), wówczas obliczamy wyniki działania macierzy przekształcenia na kolejne kolumny macierzy.

W praktyce wykorzystywane jest również obustronne przekształcenie macierzy z wykorzystaniem odbić Householdera, aby otrzymać macierz podobną (zob. np. rozdz. 3.6.3), tzn.

$$\mathbf{A}' = \mathbf{PAP}.$$

Również wówczas nie należy obliczać elementów macierzy przekształcenia i dokonywać mnożenia macierzy, ale obliczać prościej wynik takiego działania. Definiując bowiem

$$\mathbf{r} = \frac{\mathbf{A}\mathbf{u}}{K}, \quad \mathbf{s}^{\mathrm{T}} = \frac{\mathbf{u}^{\mathrm{T}}\mathbf{A}}{K}, \quad h = \frac{\mathbf{u}^{\mathrm{T}}\mathbf{r}}{2K},$$

mamy

$$\mathbf{AP} = \mathbf{A}\left(\mathbf{I} - \frac{\mathbf{u}\mathbf{u}^{\mathrm{T}}}{K}\right) = \mathbf{A} - \mathbf{r}\mathbf{u}^{\mathrm{T}}.$$

Stąd

$$\begin{aligned}\mathbf{A}' = \mathbf{PAP} &= \left(\mathbf{I} - \frac{\mathbf{u}\mathbf{u}^{\mathrm{T}}}{K}\right)\left(\mathbf{A} - \mathbf{r}\mathbf{u}^{\mathrm{T}}\right) \\ &= \mathbf{A} - \mathbf{r}\mathbf{u}^{\mathrm{T}} - \mathbf{u}\mathbf{s}^{\mathrm{T}} + 2h\mathbf{u}\mathbf{u}^{\mathrm{T}} \\ &= \mathbf{A} - (\mathbf{r} - h\mathbf{u})\,\mathbf{u}^{\mathrm{T}} - \mathbf{u}\,(\mathbf{s} - h\mathbf{u})^{\mathrm{T}}\,. \end{aligned} \tag{3.45}$$

Dla macierzy *symetrycznych* mamy dodatkowo

$$\mathbf{s} = \frac{\mathbf{A}^{\mathrm{T}}\mathbf{u}}{K} = \frac{\mathbf{A}\mathbf{u}}{K} = \mathbf{r}$$

i końcowy wzór upraszcza się do postaci

$$\mathbf{A}' = \mathbf{A} - (\mathbf{r} - h\mathbf{u})\,\mathbf{u}^{\mathrm{T}} - \mathbf{u}\,(\mathbf{r} - h\mathbf{u})^{\mathrm{T}}\,. \tag{3.46}$$

Przekształcenie Householdera jest algorytmem numerycznie poprawnym.

3.6.2. Rozkład QR macierzy odbiciami Householdera

Niech będzie dana macierz $\mathbf{A} \in \mathbb{R}^{n\times n}$. Przy opisie algorytmu będziemy stosować *notację*:

- x – elementy nieprzekształcone wektora $\mathbf{a}$ sprowadzanego na zadany kierunek w danym kroku algorytmu,
- x – pozostałe elementy nieprzekształcone,
- $*$ – elementy przekształcone (zmienione).

Pierwszy krok: sprowadzenie pierwszej kolumny macierzy $\mathbf{A}$ na kierunek $\mathbf{e}_1$:

$$\mathbf{P} = \mathbf{P}^{(1)}, \quad \mathbf{P}^{(1)}\mathbf{A} = \mathbf{P}^{(1)} \begin{bmatrix} x & \mathrm{x} & \mathrm{x} & \\ x & \mathrm{x} & \mathrm{x} & \\ x & \mathrm{x} & \mathrm{x} & \cdots \\ \vdots & \vdots & \vdots & \\ x & \mathrm{x} & \mathrm{x} & \end{bmatrix} = \begin{bmatrix} * & * & * & \\ 0 & * & * & \\ 0 & * & * & \cdots \\ \vdots & \vdots & \vdots & \\ 0 & * & * & \end{bmatrix}.$$

Drugi krok: postępujemy identycznie jak w kroku pierwszym, ale dla pierwszej kolumny podmacierzy o wymiarze $n-1$ (powstałej przez pominięcie pierwszego wiersza i pierwszej kolumny). Stąd, macierzą przekształcenia jest $\mathbf{P}^{(2)}$:

$$\mathbf{P}^{(2)}\left(\mathbf{P}^{(1)}\mathbf{A}\right)=\begin{bmatrix}1 & 0\,0 & \cdots\\ 0 & & \\ 0 & \widetilde{\mathbf{P}}^{(2)}_{(n-1)} & \\ \vdots & & \\ 0 & & \end{bmatrix}\begin{bmatrix}\mathrm{x} & \mathrm{x} & \mathrm{x} & & \\ 0 & x & \mathrm{x} & & \\ 0 & x & \mathrm{x} & \cdots & \\ \vdots & \vdots & \vdots & & \\ 0 & x & \mathrm{x} & & \end{bmatrix}=\begin{bmatrix}\mathrm{x} & \mathrm{x} & \mathrm{x} & & \\ 0 & * & * & & \\ 0 & 0 & * & \cdots & \\ \vdots & \vdots & \vdots & & \\ 0 & 0 & * & & \end{bmatrix},$$

gdzie $\widetilde{\mathbf{P}}^{(2)}_{(n-1)}$ jest macierzą przekształcenia Householdera wektora o wymiarze $n-1$, sprowadzającego pierwszą kolumnę $(n-1)$-wymiarowej podmacierzy macierzy $\mathbf{P}^{(1)}\mathbf{A}$ na kierunek pierwszego wersora przestrzeni $\mathbb{R}^{n-1}$.

Kolejne kroki wykonujemy analogicznie, aż do uzyskania

$$\mathbf{P}^{(n-1)}\mathbf{P}^{(n-2)}\,\ldots\,\mathbf{P}^{(2)}\mathbf{P}^{(1)}\mathbf{A}=\mathbf{Q}^{-1}\mathbf{A}=\mathbf{R},$$

gdzie $\mathbf{R}$ – macierz trójkątna górna, tzn.

$$\mathbf{A}=\mathbf{Q}\mathbf{R},$$
$$\mathbf{Q}=\mathbf{P}^{(1)}\mathbf{P}^{(2)}\,\ldots\,\mathbf{P}^{(n-2)}\mathbf{P}^{(n-1)},$$

przy czym macierz $\mathbf{Q}$ jest ortogonalna, co bezpośrednio wynika z

$$\mathbf{P}^{(i)}\left(\mathbf{P}^{(i)}\right)^{\mathrm{T}}=\mathbf{P}^{(i)}\mathbf{P}^{(i)}=\mathbf{I},\quad i=1,...,n-1.$$

Nakład obliczeń do wyznaczenia macierzy $\mathbf{R}$ jest rzędu $\frac{2}{3}n^3$ mnożeń i tyleż dodawań, oraz $n-1$ pierwiastkowań. Do jawnego obliczenia macierzy $\mathbf{Q}$ potrzeba dodatkowo rzędu n^3 działań.

Przedstawiony algorytm można oczywiście zastosować do wyznaczania rozkładu QR macierzy także prostokątnej, $\mathbf{A}\in\mathbb{R}^{m\times n},\ m>n$.

3.6.3. Redukcja macierzy do postaci Hessenberga odbiciami Householdera, z zachowaniem podobieństwa

Macierzą o postaci Hessenberga nazywamy macierz $\mathbf{H}$, która ma strukturę prawie trójkątną górną (gwiazdkami oznaczamy elementy mogące przyjmować wartości niezerowe):

$$\mathbf{H}=\begin{bmatrix}* & * & * & * & \cdots & *\\ * & * & * & * & \cdots & *\\ 0 & * & * & * & \cdots & *\\ 0 & 0 & * & * & \cdots & *\\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots\\ 0 & 0 & \ldots & 0 & * & *\end{bmatrix}.$$

Przekształcać będziemy odbiciami Householdera, zachowując podobieństwo macierzy, tzn. dwustronnie. Sposób postępowania będzie podobny do stosowanego w rozdziale 3.6.2, ale przekształcane wektory będą teraz miały wymiar o jeden mniejszy.

Pierwszy krok: zerowanie elementów w pierwszej kolumnie macierzy $\mathbf{A} = \mathbf{A}^{(1)}$, poczynając od trzeciego.

W tym celu konstruujemy macierz Householdera $\widetilde{\mathbf{P}}^{(1)} \in \mathbb{R}^{(n-1)\times(n-1)}$ odbijającą wektor $[a_{21}\ a_{31}\ \cdots\ a_{n1}]^{\mathrm{T}}$ na kierunek pierwszego wersora przestrzeni $\mathbb{R}^{n-1}$, tj. do postaci $[*\ 0\ \cdots\ 0]^{\mathrm{T}} \in \mathbb{R}^{n-1}$. Następnie, konstruujemy macierz przekształcenia w pierwszym kroku, $\mathbf{P}^{(1)} \in \mathbb{R}^{n\times n}$, postaci

$$\mathbf{P}^{(1)} = \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \widetilde{\mathbf{P}}^{(1)} \end{bmatrix},$$

oraz dokonujemy przekształcenia obustronnego

$$\mathbf{A}^{(2)} = \mathbf{P}^{(1)}\mathbf{A}^{(1)}\mathbf{P}^{(1)}.$$

Łatwo sprawdzić, że lewostronne pomnożenie przez $\mathbf{P}^{(1)}$ przekształca odpowiednio pierwszą kolumnę $\mathbf{A}^{(1)}$ (nie zmieniając pierwszego wiersza), a prawostronne pomnożenie tej kolumny nie zmienia, ze względu na strukturę macierzy $\mathbf{P}^{(1)}$, stąd macierz $\mathbf{A}^{(2)}$ ma pożądaną postać

$$\mathbf{A}^{(2)} = \begin{bmatrix} * & * & * & \cdots & * \\ * & * & * & \cdots & * \\ 0 & * & * & \cdots & * \\ 0 & * & * & \cdots & * \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & * & \dots & * & * \end{bmatrix}.$$

Drugi krok: zerowanie elementów w drugiej kolumnie macierzy $\mathbf{A}^{(2)}$, poczynając od elementu czwartego.

W tym celu konstruujemy macierz Householdera $\widetilde{\mathbf{P}}^{(2)} \in \mathbb{R}^{(n-2)\times(n-2)}$ odbijającą podwektor $[a_{32}^{(2)}\ a_{42}^{(2)}\ \cdots\ a_{n2}^{(2)}]^{\mathrm{T}}$ drugiej kolumny macierzy $\mathbf{A}^{(2)}$ na kierunek pierwszego wersora przestrzeni $\mathbb{R}^{n-2}$, tj. do postaci $[*\ 0\ \cdots\ 0]^{\mathrm{T}} \in \mathbb{R}^{n-2}$. Następnie, konstruujemy macierz przekształcenia $\mathbf{P}^{(2)} \in \mathbb{R}^{n\times n}$ postaci

$$\mathbf{P}^{(2)} = \begin{bmatrix} \mathbf{I}_2 & \mathbf{0} \\ \mathbf{0} & \widetilde{\mathbf{P}}^{(2)} \end{bmatrix},$$

gdzie $\mathbf{I}_2$ jest dwuwymiarową macierzą jednostkową, oraz dokonujemy przekształcenia obustronnego

$$\mathbf{A}^{(3)} = \mathbf{P}^{(2)}\mathbf{A}^{(1)}\mathbf{P}^{(2)}.$$

Łatwo sprawdzić, że lewostronne pomnożenie przez $\mathbf{P}^{(2)}$ przekształca odpowiednio drugą kolumnę macierzy $\mathbf{A}^{(1)}$ nie zmieniając pierwszej (i pierwszych dwóch wierszy), a następnie prawostronne pomnożenie nie zmienia dwóch pierwszych kolumn, ze względu na strukturę macierzy $\mathbf{P}^{(2)}$, stąd uzyskana macierz jest postaci

$$\mathbf{A}^{(3)} = \begin{bmatrix} * & * & * & * & \cdots & * \\ * & * & * & * & \cdots & * \\ 0 & * & * & * & \cdots & * \\ 0 & 0 & * & * & \cdots & * \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & * & \dots & * & * \end{bmatrix}.$$

W kroku trzecim postępujemy analogicznie, tj. konstruujemy przekształcenie Householdera $\widetilde{\mathbf{P}}^{(3)} \in \mathbb{R}^{(n-3)\times(n-3)}$ zerujące elementy trzeciej kolumny macierzy $\mathbf{A}^{(3)}$, poczynając od elementu piątego, oraz macierz przekształcenia $\mathbf{P}^{(3)}$ o analogicznej strukturze (z podmacierzą $\mathbf{I}_3$ w lewym górnym rogu). Postępujemy podobnie w kolejnych krokach, aż do kroku ostatniego, $n-2$, uzyskując

$$\mathbf{A}^{(n-1)} = \mathbf{P}^{(n-2)}\mathbf{P}^{(n-3)} \cdots \mathbf{P}^{(2)}\mathbf{P}^{(1)}\mathbf{A}\mathbf{P}^{(1)}\mathbf{P}^{(2)} \cdots \mathbf{P}^{(n-3)}\mathbf{P}^{(n-2)} = \mathbf{H}.$$

Uwaga. Nie należy obliczać wszystkich elementów macierzy $\widetilde{\mathbf{P}}^{(i)}$, lecz wyniki działania tych macierzy na kolejne wektory (kolumny podmacierzy), wykorzystując odpowiednio zależność (3.45).

Potrzebna liczba mnożeń i dodawań jest rzędu $\frac{5}{3}n^3$, dla macierzy symetrycznych może być zredukowana do $\frac{2}{3}n^3$.

Zadania

1. Stosując standardowy algorytm ortogonalizacji Grama-Schmidta, oblicz, kolejno, rozkłady QR (3.7), (3.8) i (3.9) macierzy

$$\begin{bmatrix} 1 & 3 & 2 \\ 1 & 1 & 4 \\ 1 & 3 & 4 \\ 1 & 1 & 2 \end{bmatrix}.$$

2. Oblicz wartości własne i wektory własne macierzy

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \quad \begin{bmatrix} 1 & 2 & 0 \\ 0 & 2 & 0 \\ -2 & -2 & -1 \end{bmatrix}.$$

Sprawdź, na przykładzie macierzy jak powyżej, następujący fakt: macierz $\mathbf{V}$, której kolumnami są wzajemnie ortogonalne wektory własne macierzy $\mathbf{A}$, diagonalizuje $\mathbf{A}$ przez przekształcenie podobieństwa.

3. Uzasadnij, że wyznacznik macierzy symetrycznej jest równy iloczynowi jej wartości własnych.

4. Twierdzenie Cayley'a-Hamiltona: Każda macierz kwadratowa $\mathbf{A}$ jest pierwiastkiem swojego wielomianu charakterystycznego $w(\lambda)$, tzn. $w(\mathbf{A}) = \mathbf{0}$. Wykorzystaj to twierdzenie do sformułowanie sposobu znajdowania macierzy odwrotnej (wskazówka: pomnóż $w(\mathbf{A})$ przez $\mathbf{A}^{-1}$).

5. Wyznacz rozkład SVD i macierz pseudoodwrotną macierzy

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

6. Pokaż, że wskaźnik uwarunkowania macierzy kwadratowej nieosobliwej w normie drugiej jest równy ilorazowi jej największej i najmniejszej wartości szczególnej (wskazówka: korzystając z rozkładu SVD, pokaż, że wartości szczególne macierzy odwrotnej są odwrotnościami wartości szczególnych macierzy odwracanej).

7. Znajdź rozwiązanie $\hat{\mathbf{x}}$ układu równań $\mathbf{A}\mathbf{x} = \mathbf{b}$, gdzie

$$\mathbf{A} = \begin{bmatrix} 1 & 3.5 & 3 \\ 1 & -0.5 & 1 \\ 1 & 3.5 & 7 \\ 1 & -0.5 & -3 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 17.5 \\ 1.5 \\ 25.5 \\ -6.5 \end{bmatrix},$$

w sensie najmniejszej normy drugiej wektora błędu rozwiązania $\mathbf{A}\hat{\mathbf{x}} - \mathbf{b}$:
   a) wykorzystując rozkład QR macierzy $\mathbf{A}$ (wyliczając i wykorzystując najdogodniejszą z form rozkładu podanych w twierdzeniu 3.1),
   b) wykorzystując układ równań normalnych.

Rozdział 4

Aproksymacja

Zadaniem aproksymacji jest przybliżenie funkcji $f(x)$, określonej dokładnie w zadanym przedziale lub jedynie w przybliżeniu, w skończonej liczbie punktów tego przedziału, inną prostszą funkcją $F(x)$, należącą do wybranej klasy funkcji aproksymujących. Aproksymowanie funkcji ciągłej w danym przedziale prostszą funkcją ciągłą stosujemy, gdy skomplikowana postać analityczna tej pierwszej nie pozwala na jej wykorzystanie np. w stosowanych metodach analizy czy projektowania. Z kolei przybliżanie funkcją ciągłą zależności znanej jedynie w skończonej liczbie punktów (aproksymacja dyskretna, punktowa) szeroko wykorzystuje się w technice modelowania, np. [4].

Oznaczmy:

$f\,(x)$ – funkcja oryginalna, aproksymowana (znana dokładnie lub punktowo),

$F\,(x)$ – funkcja aproksymująca.

Niech:

X – przestrzeń funkcyjna liniowa, $f \in X$,

X_n – (n+1)-wymiarowa podprzestrzeń przestrzeni funkcyjnej X, z bazą złożoną z funkcji $\phi_0\,(x)\,, ..., \phi_n\,(x)$, tzn.

$$F\,(x) \in X_n \quad \Leftrightarrow \quad F\,(x) = a_0\phi_0\,(x) + a_1\phi_1\,(x) + ... + a_n\phi_n\,(x)\,,$$

gdzie $a_i \in \mathbb{R},\ i = 0, 1, ..., n,$ to współczynniki rozwinięcia w bazie.

Zadanie aproksymacji można zdefiniować jako zadanie znalezienia funkcji F^*, $F^* \in X_n$, najbliższej funkcji f, np. w sensie odległości $\delta(f–F)$ definiowanej przez pewną normę $\|\cdot\|$,

$$\forall F \in X_n \quad \delta(f-F^*) \stackrel{\text{df}}{=} \|f - F^*\| \leq \|f - F\|\,.$$

Tak więc aproksymacja polegać będzie na dobraniu współczynników $a_0, ..., a_n$ funkcji F w bazie przestrzeni X_n tak, aby zminimalizować normę $\|f - F\|$.

Przykładami rodzajów aproksymacji funkcji, wynikających z przyjęcia określonej postaci normy, są:

– *Aproksymacja jednostajna ciągła* funkcji $f(x)$, ciągłej na przedziale domkniętym $[a, b\,]$:

$$\|F - f\| = \max_{x\in[a,b]} |F\,(x) - f\,(x)| \qquad \text{(z normą Czebyszewa).} \qquad (4.1)$$

– *Aproksymacja średniokwadratowa ciągła* funkcji $f(x)$ (z wagą $p(x)$), całkowalnej z kwadratem na przedziale $[a, b]$, tzn. $f(x) \in L_p^2[a, b]$:

$$\|F - f\| = \sqrt{\int_a^b p\,(x)\,[F\,(x) - f\,(x)]^2\,dx}, \tag{4.2}$$

gdzie $p\,(x)$ – funkcja wagowa.

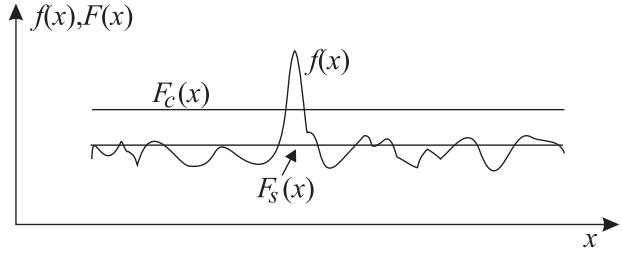
– *Aproksymacja jednostajna dyskretna (punktowa)* funkcji $f\,(x)$, znanej jedynie na skończonym zbiorze $N+1$ punktów:

$$\|F - f\| = \max\{|F(x_0) - f(x_0)|, |F(x_1) - f(x_1)|, \cdots, |F(x_N) - f(x_N)|\}. \tag{4.3}$$

– *Aproksymacja średniokwadratowa dyskretna (punktowa)* (metoda najmniejszych kwadratów) funkcji $f\,(x)$, znanej jedynie na skończonym zbiorze $N+1$ punktów:

$$\|F - f\| = \sqrt{\sum_{j=0}^{N} p\,(x_j)\,[F\,(x_j) - f\,(x_j)]^2}. \tag{4.4}$$

Na rysunku 4.1 zilustrowano istotę różnicy miedzy aproksymacją jednostajną a aproksymacją średniokwadratową (w przypadku ciągłym). Pokazano aproksymację linii krzywej odcinkiem wielomianu zerowego rzędu (tzn. odcinkiem prostej): prosta $F_c(x)$ jest rezultatem aproksymacji ciągłej jednostajnej (maksymalne odchyłki krzywej od tej prostej w górę i w dół są identyczne – stąd najmniejsza odległość definiowana normą Czebyszewa (4.1)), zaś prosta $F_s(x)$ jest rezultatem aproksymacji średniokwadratowej (całka z kwadratu różnicy $f(x) - F_s(x)$ na przedziale aproksymacji jest najmniejsza). W pierwszym przypadku istotna jest tylko *największa odchyłka*, zaś w drugim na miarę odległości wpływają *wszystkie odchyłki (różnice)*, bez względu na znak.



Rys. 4.1. Interpretacja geometryczna różnicy między aproksymacjami ciągłymi krzywej $f(x)$ wielomianem zerowego rzędu: jednostajną $F_c(x)$ i średniokwadratową $F_s(x)$

4.1. Aproksymacja średniokwadratowa dyskretna

Niech $f(x)$ przyjmuje na pewnym zbiorze punktów $x_0, x_1, ..., x_N$ ($x_i \neq x_j$) znane wartości $y_j = f(x_j)$, $j = 0, 1, 2, ..., N$.

Niech $\phi_i(x)$, $i = 0, 1, ..., n$, będzie układem funkcji bazowych przestrzeni funkcji aproksymujących $X_n \subseteq X$, tzn.

$$\forall\, F \in X_n \quad F(x) = \sum_{i=0}^{n} a_i \phi_i(x). \tag{4.5}$$

Zadanie aproksymacji:
wyznaczyć wartości współczynników $a_0, a_1, ..., a_n$ określających funkcję aproksymującą (4.5) tak, aby zminimalizować błąd średniokwadratowy zdefiniowany zależnością

$$H(a_0, ..., a_n) \stackrel{\text{df}}{=} \sum_{j=0}^{N} \left[f(x_j) - \sum_{i=0}^{n} a_i \phi_i(x_j) \right]^2. \tag{4.6}$$

W sformułowaniu powyższym nie uwzględniliśmy, dla uproszczenia, zmiennej funkcji wagowej $p(x_j)$.

Wyznaczymy poszukiwane współczynniki $a_0, a_1, ..., a_n$ z warunków koniecznych minimum (tu również dostatecznych i jednoznacznych, gdyż funkcja jest ściśle wypukła):

$$\frac{\partial H}{\partial a_k} = -2 \sum_{j=0}^{N} \left[f(x_j) - \sum_{i=0}^{n} a_i \phi_i(x_j) \right] \cdot \phi_k(x_j) = 0, \qquad k = 0, ..., n,$$

tzn.

$$a_0 \sum_{j=0}^{N} \phi_0(x_j) \cdot \phi_0(x_j) + a_1 \sum_{j=0}^{N} \phi_1(x_j) \cdot \phi_0(x_j) + \cdots + a_n \sum_{j=0}^{N} \phi_n(x_j) \cdot \phi_0(x_j) = \sum_{j=0}^{N} f(x_j) \cdot \phi_0(x_j),$$

$$a_0 \sum_{j=0}^{N} \phi_0(x_j) \cdot \phi_1(x_j) + a_1 \sum_{j=0}^{N} \phi_1(x_j) \cdot \phi_1(x_j) + \cdots + a_n \sum_{j=0}^{N} \phi_n(x_j) \cdot \phi_1(x_j) = \sum_{j=0}^{N} f(x_j) \cdot \phi_1(x_j),$$

$$\vdots$$

$$\vdots$$

$$a_0 \sum_{j=0}^{N} \phi_0(x_j)\cdot\phi_n(x_j) + a_1 \sum_{j=0}^{N} \phi_1(x_j)\cdot\phi_n(x_j) + \cdots + a_n \sum_{j=0}^{N} \phi_n(x_j)\cdot\phi_n(x_j) = \sum_{j=0}^{N} f(x_j)\cdot\phi_n(x_j).$$

Powyższy układ równań liniowych względem współczynników $a_0, ..., a_n$ to tzw. *układ równań normalnych*, a macierz tego układu to tzw. *macierz Grama*. Układ ten można zapisać w zwięzłej postaci, wykorzystując pojęcie iloczynu skalarnego

$$\langle \phi_i, \phi_k \rangle \stackrel{\text{df}}{=} \sum_{j=0}^{N} \phi_i(x_j)\,\phi_k(x_j). \tag{4.7}$$

Układ równań normalnych przyjmuje wówczas prostą w zapisie postać

$$\begin{bmatrix} \langle\phi_0,\phi_0\rangle & \langle\phi_1,\phi_0\rangle & \cdots & \langle\phi_n,\phi_0\rangle \\ \langle\phi_0,\phi_1\rangle & \langle\phi_1,\phi_1\rangle & \cdots & \langle\phi_n,\phi_1\rangle \\ \vdots & \vdots & \vdots & \vdots \\ \langle\phi_0,\phi_n\rangle & \langle\phi_1,\phi_n\rangle & \cdots & \langle\phi_n,\phi_n\rangle \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} \langle\phi_0, f\rangle \\ \langle\phi_1, f\rangle \\ \vdots \\ \langle\phi_n, f\rangle \end{bmatrix}. \tag{4.8}$$

Zdefiniujmy macierz $\mathbf{A}$ postaci

$$\mathbf{A} = \begin{bmatrix} \phi_0(x_0) & \phi_1(x_0) & \cdots & \phi_n(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \cdots & \phi_n(x_1) \\ \phi_0(x_2) & \phi_1(x_2) & \cdots & \phi_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(x_N) & \phi_1(x_N) & \cdots & \phi_n(x_N) \end{bmatrix} \tag{4.9}$$

oraz wektory zmiennych decyzyjnych i wartości funkcji oryginalnej jako

$$\begin{aligned} \mathbf{a} &= [a_0\ a_1 \cdots a_n]^{\mathrm{T}}, \\ \mathbf{y} &= [y_0\ y_1 \cdots y_N]^{\mathrm{T}}, \quad y_j = f(x_j), \quad j = 0, 1, ..., N. \end{aligned}$$

Wówczas funkcję celu (4.6) zadania aproksymacji możemy zapisać w postaci

$$H(\mathbf{a}) = (\parallel \mathbf{y} - \mathbf{A}\mathbf{a} \parallel_2)^2. \tag{4.10}$$

Zadanie aproksymacji średniokwadratowej dyskretnej jest więc liniowym zadaniem najmniejszych kwadratów (LZNK). Zwróćmy uwagę, że kolumny macierzy

$\mathbf{A}$ są liniowo niezależne (co wynika z liniowej niezależności funkcji bazowych), a więc macierz $\mathbf{A}$ jest pełnego rzędu (równego $n+1$).

Wykorzystując definicję (4.9) macierzy $\mathbf{A}$ układ równań normalnych (4.8) możemy zapisać w postaci

$$\mathbf{A}^{\mathrm{T}}\mathbf{A}\,\mathbf{a} = \mathbf{A}^{\mathrm{T}}\mathbf{y}. \tag{4.11}$$

Ponieważ macierz $\mathbf{A}$ jest pełnego rzędu, więc macierz Grama $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ jest nieosobliwa. Stąd również wynika jednoznaczność rozwiązania układu równań normalnych. Natomiast macierz $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ może być źle uwarunkowana – jej wskaźnik uwarunkowania jest bowiem kwadratem wskaźnika uwarunkowania macierzy $\mathbf{A}$. Zalecanym sposobem rozwiązania zadania aproksymacji przy źle uwarunkowanym układzie równań normalnych jest wykorzystanie rozkładu **QR** macierzy $\mathbf{A}$, zob. metody rozwiązywania LZNK w rozdziale 3.4.

Przykład 4.1. Rozważymy zadanie aproksymacji w dwuwymiarowej ($n=1$) bazie funkcyjnej złożonej z funkcji bazowych postaci

$$\phi_0(x) = x, \quad \phi_1(x) = e^x,$$

dla zestawu danych

x_j	-2	-1	0	1	2
y_j	-3	-1.2	0.2	3	7.5

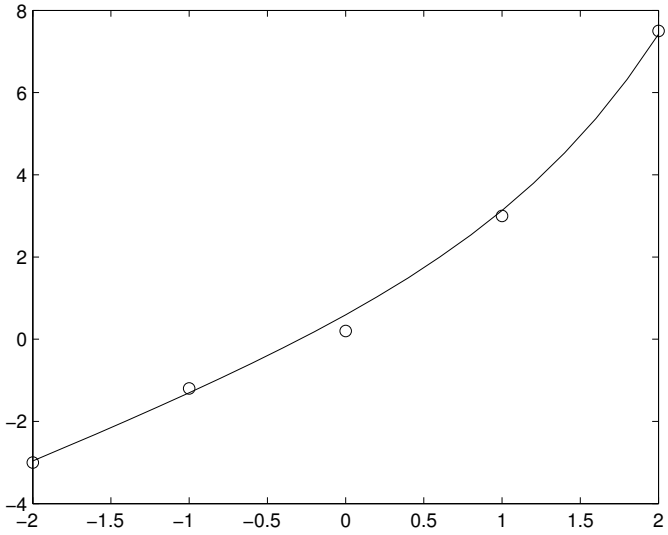
.

Mamy 5 punktów, tzn. $N=4$. Na początku wyliczymy elementy macierzy Grama i wektora prawej strony układu równań normalnych:

$$\begin{aligned}
\langle\phi_0,\phi_0\rangle &= \sum_{j=0}^{4} x_j x_j = 4+1+0+1+4 = 10,\\
\langle\phi_0,\phi_1\rangle &= \sum_{j=0}^{4} x_j e^{x_j} = -2e^{-2} - e^{-1} + 0 + e^{1} + 2e^{2} = 16.8578,\\
\langle\phi_1,\phi_0\rangle &= \langle\phi_0,\phi_1\rangle = 16.8578,\\
\langle\phi_1,\phi_1\rangle &= \sum_{j=0}^{4} e^{x_j} e^{x_j} = (e^{-2})^2 + (e^{-1})^2 + (e^{0})^2 + (e^{1})^2 + (e^{2})^2 = 63.1409,\\
\langle\phi_0,f\rangle &= \sum_{j=0}^{4} x_j y_j = 6 + 1.2 + 0 + 3 + 15 = 25.2,\\
\langle\phi_1,f\rangle &= \sum_{j=0}^{4} e^{x_j} y_j = -3e^{-2} - 1.2e^{-1} + 0.2e^{0} + 3e^{1} + 7.5e^{2} = 62.9253.
\end{aligned}$$

Dostajemy stąd układ równań normalnych postaci

$$\begin{bmatrix} 10 & 16.8578 \\ 16.8578 & 63.1409 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} 25.2 \\ 62.9253 \end{bmatrix}.$$



Rys. 4.2. Położenie danych i funkcji aproksymującej z przykładu 4.1

Rozwiązaniem tego układu równań jest wektor $[a_0 \ a_1]^{\mathrm{T}} = [1.5213 \ 0.5925]^{\mathrm{T}}$. Stąd dostajemy funkcję aproksymującą

$$F(x) = 1.5213x + 0.5925e^x.$$

Na rysunku 4.2 przedstawiono dane (kółkami) i przebieg funkcji aproksymującej. □

4.1.1. Aproksymacja wielomianami w bazie naturalnej

Często stosowanymi funkcjami aproksymującymi są wielomiany algebraiczne $W_n(x)$ (n – stopień wielomianu). Podstawą takiego postępowania jest klasyczne twierdzenie Weierstrassa o jednostajnej aproksymacji wielomianami algebraicznymi funkcji ciągłej $f(x)$ na przedziale domkniętym $[a, b]$, które można podać w postaci

$$\forall \varepsilon > 0 \quad \exists n \quad \forall x \in [a, b] \quad |f(x) - W_n(x)| \leq \varepsilon. \tag{4.12}$$

Przy aproksymacji wielomianem jego stopień jest najczęściej znacznie mniejszy od liczby punktów, na podstawie których dokonujemy aproksymacji, tzn.

$$N \gg n.$$

Przyjmijmy *bazę wielomianów tzw. naturalną (potęgową)*:

$$\phi_0(x) = 1, \ \ \phi_1(x) = x, \ \phi_2(x) = x^2, \ \ \cdots, \ \ \phi_n(x) = x^n, \tag{4.13}$$

tzn.

$$F(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n. \tag{4.14}$$

Wprowadzając oznaczenia pomocnicze:

$$g_{ik} \overset{\text{df}}{=} \sum_{j=0}^{N} (x_j)^i (x_j)^k = \sum_{j=0}^{N} (x_j)^{i+k},$$

$$\varrho_k \overset{\text{df}}{=} \sum_{j=0}^{N} f(x_j) (x_j)^k,$$

uzyskujemy układ równań normalnych w postaci

$$\begin{array}{l} a_0 g_{00} + a_1 g_{10} + ... + a_n g_{n0} = \varrho_0 \\ a_0 g_{01} + a_1 g_{11} + ... + a_n g_{n1} = \varrho_1 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_0 g_{0n} + a_1 g_{1n} + ... + a_n g_{nn} = \varrho_n \end{array} \quad \equiv \quad \mathbf{G} \cdot \mathbf{a} = \varrho. \tag{4.15}$$

Przykład 4.2. Dla zestawu danych

x_j	0	0.2	0.4	0.6	0.8	1.0
y_j	1.15	0.7	0.5	0.4	0.25	0.2

znaleźć wielomian algebraiczny aproksymujący stopnia: a) pierwszego; b) drugiego. Wyznaczyć błąd aproksymacji, w sensie normy maksimum (tj. maksymalny punktowy błąd aproksymacji).

a) Po wyliczeniu współczynników g_{ij} i ϱ_i dostajemy układ równań normalnych w postaci

$$\begin{bmatrix} 6 & 3 \\ 3 & 2.2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} 3.2 \\ 0.98 \end{bmatrix},$$

skąd po rozwiązaniu wyznaczamy współczynniki wielomianu aproksymującego liniowego

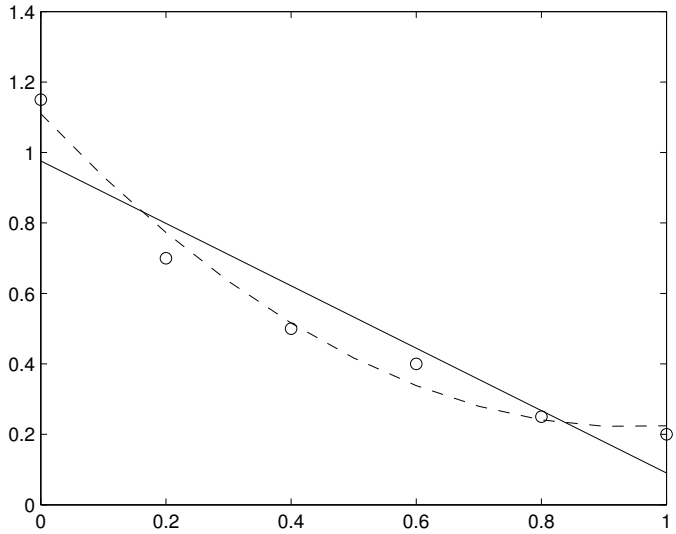
$$w_1(x) = 0.976 - 0.886x.$$

b) Po wyliczeniu współczynników g_{ij} i ϱ_i dostajemy układ równań normalnych postaci

$$\begin{bmatrix} 6 & 3 & 2.2 \\ 3 & 2.2 & 1.8 \\ 2.2 & 1.8 & 1.566 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 3.2 \\ 0.98 \\ 0.612 \end{bmatrix},$$

skąd po rozwiązaniu wyznaczamy współczynniki wielomianu aproksymującego kwadratowego

$$w_2(x) = 1.11 - 1.886x + x^2.$$



Rys. 4.3. Położenie danych i funkcji aproksymujących z przykładu 4.2

Błędy aproksymacji są następujące:

x_j	0	0.2	0.4	0.6	0.8	1.0	błąd
$w_1(x_j)-y_j$	-0.174	0.103	0.130	0.0564	0.033	-0.09	0.174
$w_2(x_j)-y_j$	-0.04	0.073	0.0156	-0.062	-0.009	0.024	0.073

Na rysunku 4.3 przedstawiono dane (kółkami) i przebiegi funkcji aproksymujących: linią ciągłą wielomianu pierwszego stopnia i linią przerywaną wielomianu drugiego stopnia. □

Uwaga. Macierz Grama $\mathbf{G}$ układu równań normalnych dla aproksymacji wielomianami w bazie naturalnej ($\phi_0(x) = 1,\ \ \phi_1(x) = x,\ \ \cdots,\ \ \phi_n(x) = x^n$) wraz ze wzrostem rzędu n wielomianu aproksymującego szybko traci dobre uwarunkowanie (dla $n > 5$ jest już często źle uwarunkowana).

Fakt ten można wyjaśnić następująco: niech $x\epsilon[0,1]$, $x = 0 + jh$, $h = 1/N$, $j = 0, 1, 2, ..., N$. Dla dostatecznie dużych wartości N można zastosować przybliżenie

$$g_{ik} = (N+1)\frac{1}{N+1}\sum_{j=0}^{N}(x_j)^{i+k} \approx (N+1)\int_0^{1+\frac{1}{N}} x^{i+k}dx$$

$$= (N+1)\frac{(1+\frac{1}{N})^{i+k+1}}{i+k+1} \approx (N+1)\frac{1^{i+k+1}}{i+k+1} = (N+1)\frac{1}{i+k+1}.$$

Stosując to przybliżenie, dostajemy

$$\mathbf{G} = (N+1)\begin{bmatrix} 1 & \frac{1}{2} & \cdots & \frac{1}{n+1} \\ \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n+2} \\ \vdots & \vdots & \ddots & \\ \frac{1}{n+1} & \frac{1}{n+2} & & \frac{1}{2n+1} \end{bmatrix} = (N+1)\cdot\mathbf{G}_N.$$

Macierz $\mathbf{G}_N$ jest typu macierzy Hilberta, która znana jest ze złego uwarunkowania, szybko pogarszającego się ze wzrostem wartości n.

Stąd, aproksymację wielomianami z bazą naturalną $\{1,\ x,\ x^2,\ x^3,\ ...\}$ stosuje się w praktyce dla małych wartości n, nieprzekraczających $n = 4$. Złego uwarunkowania można uniknąć, stosując bazę złożoną z wielomianów ortogonalnych.

4.1.2. Aproksymacja funkcjami z przestrzeni o bazie ortogonalnej

Funkcje $h(x)$ i $g(x)$ są ortogonalne na zbiorze punktów $x_0, x_1, ..., x_N$, jeśli spełniają warunek zerowania iloczynu skalarnego (4.7), tzn.

$$\langle h(x), g(x)\rangle = 0, \qquad \text{czyli} \qquad \sum_{j=0}^{N} h(x_j)\, g(x_j) = 0.$$

Ogólnie, funkcje bazowe $\psi_0, ..., \psi_n$ są wzajemnie *ortogonalne*, jeśli

$$\sum_{j=0}^{N} \psi_i(x_j)\, \psi_k(x_j) = 0 \quad \text{dla każdego } i \neq k, \quad i,k = 0,1,...,n.$$

Funkcje te nazywamy ponadto *ortonormalnymi*, jeśli

$$\sum_{j=0}^{N} \psi_i(x_j)\, \psi_i(x_j) = 1, \quad i = 0,1,...n.$$

Metoda ortogonalizacji Grama-Schmidta

Niech:
$\phi_0, ..., \phi_n$ – znana baza złożona z funkcji nieortogonalnych,
$\psi_0, ..., \psi_n$ – baza złożona z funkcji ortogonalnych.

Algorytm ortogonalizacji Grama-Schmidta (standardowy):

$$\begin{aligned} \psi_0 &= \phi_0, \\ \psi_1 &= \phi_1 - \frac{\langle \psi_0, \phi_1\rangle}{\langle \psi_0, \psi_0\rangle}\psi_0, \end{aligned}$$

$$\psi_i = \phi_i - \sum_{j=0}^{i-1} \frac{\langle \psi_j, \phi_i \rangle}{\langle \psi_j, \psi_j \rangle} \psi_j, \qquad i = 2, ..., n. \tag{4.16}$$

Dla funkcji bazowych ortogonalnych otrzymujemy następujący, bardzo dobrze uwarunkowany, układ równań normalnych:

$$\begin{aligned} a_0 \langle \psi_0, \psi_0 \rangle &= \langle f, \psi_0 \rangle , \\ a_1 \langle \psi_1, \psi_1 \rangle &= \langle f, \psi_1 \rangle , \\ &\vdots \\ a_n \langle \psi_n, \psi_n \rangle &= \langle f, \psi_n \rangle . \end{aligned} \tag{4.17}$$

Przykład 4.3. Zadanie sformułowane w poprzednim przykładzie rozwiążemy stosując bazę wielomianów ortogonalnych, uzyskując ją przez ortogonalizację bazy naturalnej metodą Grama-Schmidta.

a) Ortogonalizując bazę $\{1,\ x\}$, dostajemy

$$\begin{aligned} \psi_0(x) &= 1, \\ \psi_1(x) &= x - \frac{\sum_{j=0}^{5} x_j}{\sum_{j=0}^{5} 1} = x - \frac{1}{2}. \end{aligned}$$

Czytelnikowi pozostawiamy sprawdzenie, że układ równań normalnych ma teraz postać:

$$\begin{aligned} a_0 \cdot 6 &= 3.2, \\ a_1 \cdot 0.7 &= -0.62, \end{aligned}$$

skąd, po wyznaczeniu współczynników, dostajemy wielomian aproksymujący

$$w_1(x) = 0.533 - 0.886(x - 0.5) = 0.976 - 0.886x.$$

b) Ponieważ baza $\{1,\ x,\ x^2\}$ jest rozszerzeniem bazy z punktu a) o element x^2, więc wystarczy zortogonalizować tę funkcję wobec funkcji ortogonalnych uzyskanych w punkcie a), tzn. wykonać kolejny krok algorytmu Grama-Schmidta. Dostajemy stąd

$$\begin{aligned} \psi_2(x) &= x^2 - \frac{\sum_{j=0}^{5} (x_j)^2}{\sum_{j=0}^{5} 1} 1 - \frac{\sum_{j=0}^{5} (x_j)^2 (x_j - \frac{1}{2})}{\sum_{j=0}^{5} (x_j - \frac{1}{2})^2} (x - \frac{1}{2}) \\ &= x^2 - x + \frac{2}{15}. \end{aligned}$$

Współczynniki a_0 i a_1 pozostają bez zmian, należy jedynie wyznaczyć a_2 z równania

$$a_2 \sum_{j=0}^{5} ({x_j}^2 - x_j + \frac{2}{15})^2 = \sum_{j=0}^{5} y_j ({x_j}^2 - x_j + \frac{2}{15}),$$

co daje w rezultacie $a_2 \approx 1$ i wielomian aproksymujący w postaci

$$w_2(x) = 0.976 - 0.886x + (x^2 - x + 0.133) = 1.11 - 1.886x + x^2.$$

□

Zastosowany powyżej standardowy algorytm Grama-Schmidta ma niekorzystne własności numeryczne – dokładność ortogonalizacji może szybko maleć wraz z obliczaniem kolejnych zortogonalizowanych funkcji bazowych. Dlatego też praktycznie stosujemy tzw. *zmodyfikowany algorytm Grama-Schmidta*, który ma lepsze własności numeryczne (zob. rozdz. 3.1). Proces ortogonalizacji bazy tym algorytmem, używając pętli „`for`" w notacji języka MATLAB, można zapisać w postaci:

```
for i = 0:n
    ψ_i = φ_i;
    for j = i+1:n
        φ_j = φ_j − ⟨ψ_i,φ_j⟩/⟨ψ_i,ψ_i⟩ ψ_i;
    end
end
```
(4.18)

Alternatywnym rozwiązaniem jest zastosowanie algorytmu Grama-Schmidta (4.16) z *reortogonalizacją*, tzn. ponowną ortogonalizacją, przeprowadzaną po wykonaniu całego algorytmu, lub na bieżąco, po wyznaczeniu każdej kolejnej funkcji bazowej ψ_i [13].

4.2. Aproksymacja Padé

Aproksymującą funkcję ciągłą nie zawsze wyznacza się jako funkcję minimalizującą pewną miarę odległości funkcji oryginalnej i funkcji aproksymującej. Przykładem innego podejścia jest aproksymacja Padé, ważna z punktu widzenia zastosowań praktycznych. Funkcja aproksymująca jest tu *funkcją wymierną*

$$R_{n,k}(x) = \frac{a_0 + a_1 x + ... + a_n x^n}{1 + b_1 x + ... + b_k x^k}, \tag{4.19}$$

gdzie dla jednoznaczności sformułowania przyjęto $b_0 = 1$.

Załóżmy, że funkcja aproksymowana $f(x)$ ma pochodne dowolnego rzedu i rozwińmy ją w szereg MacLaurina (tzn. szereg Taylora w punkcie $x = 0$ – będzie to *punkt aproksymacji*):

$$f(x) = \sum_{i=0}^{\infty} c_i x^i, \tag{4.20}$$

$$c_i = \frac{1}{i!} f^{(i)}(0), \quad i = 0, 1, 2, ... \tag{4.21}$$

Warunkiem aproksymacji Padé jest, aby *jak najwięcej pierwszych współczynników rozwinięcia funkcji $R_{n,k}$ w szereg MacLaurina było równych współczynnikom c_i rozwinięcia funkcji $f(x)$.*

Ponieważ mamy $n + k + 1$ stopni swobody (jest to liczba współczynników funkcji $R_{n,k}$ do wyznaczenia), to możemy żądać, aby pierwszych $n+k+1$ współczynników obu rozwinięć było równe, który to warunek można zapisać w postaci

$$\sum_{i=0}^{\infty} c_i x^i = \frac{a_0 + a_1 x + ... + a_n x^n}{1 + b_1 x + ... + b_k x^k} + O(x^{n+k+1}), \tag{4.22}$$

gdzie $O(x^{n+k+1})$ można traktować jako wielomian uporządkowany wg rosnących potęg x, zaczynający się od wyrazu z x^{n+k+1} (najniższa potęga x w tym wielomianie). Z (4.22) wynika bezpośrednio, że pierwszych $n+k+1$ współczynników rozwinięcia funkcji $R_{n,k}$ w szereg MacLaurina musi być równe współczynnikom c_i ($i = 0, 1, ..., n + k$).

Uwaga. Z powyższego rozumowania wynika, że warunek aproksymacji Padé może być również sformułowany jako warunek równości wartości funkcji $f(x)$ i $R_{n,k}(x)$ i ich $n + k$ pochodnych w punkcie $x = 0$:

$$f(0) = R_{n,k}(0),$$

$$f^{(j)}(0) = R^{(j)}_{n,k}(0), \qquad j = 1, ..., n + k.$$

Po pomnożeniu stronami (4.22) przez wielomian mianownika funkcji $R_{n,k}$, równanie to możemy zapisać w równoważnej postaci

$$(\sum_{i=0}^{\infty} c_i x^i)(1 + \sum_{i=1}^{k} b_i x^i) = \sum_{i=0}^{n} a_i x^i + O(x^{n+k+1}). \tag{4.23}$$

Precyzyjniej, eliminując z pierwszej sumy po lewej stronie tego równania te elementy, które prowadzą wyłącznie do składników x^i wielomianu całej lewej strony z potęgami $i \geq n + k + 1$, dostajemy

$$(\sum_{i=0}^{n+k} c_i x^i)(1 + \sum_{i=1}^{k} b_i x^i) = \sum_{i=0}^{n} a_i x^i + O(x^{n+k+1}). \tag{4.24}$$

Definiując wielomian o współczynnikach d_i w postaci

$$\sum_{i=0}^{n+2k} d_i x^i = (\sum_{i=0}^{n+k} c_i x^i)(1 + \sum_{i=1}^{k} b_i x^i) - \sum_{i=0}^{n} a_i x^i, \tag{4.25}$$

można równanie (4.24) aproksymacji Padé, tzn. warunki na wyznaczenie współczynników a_i i b_i, zapisać w postaci

$$d_i(a_0,...,a_n,b_1,...b_k,c_0,...,c_{n+k})=0,\ \ i=0,1,2,...,n+k, \tag{4.26}$$

gdzie współczynniki c_i sa znanymi parametrami.

Przykład 4.4. Wyznaczymy aproksymację Padé $R_{2,2}$,

$$R_{2,2}\left(x\right)=\frac{a_0+a_1x+a_2x^2}{1+b_1x+b_2x^2},$$

funkcji $f(x)=e^x$ w zerze.

Mamy tutaj: $n=2,\ k=2$. Rozwinięcie funkcji e^x w szereg McLaurina daje 5 pierwszych ($i=0,...,n+k=4$) współczynników:

$$c_0=1,\ \ c_1=1,\ \ c_2=\frac{1}{2},\ \ c_3=\frac{1}{6},\ \ c_4=\frac{1}{24}.$$

Stąd dostajemy równanie (4.25) w postaci

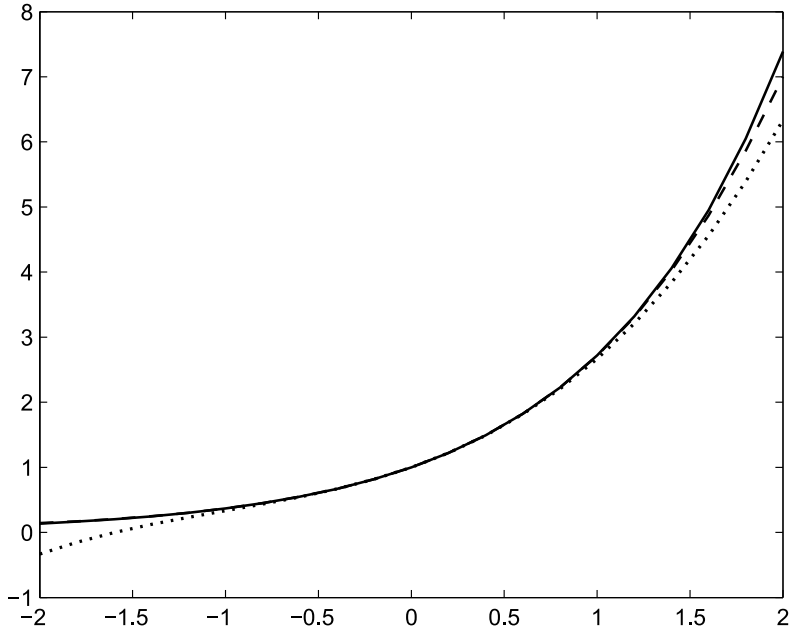
$$\sum_{i=0}^{6}d_ix^i=\left(1+x+\frac{1}{2}x^2+\frac{1}{6}x^3+\frac{1}{24}x^4\right)\left(1+b_1x+b_2x^2\right)-\left(a_0+a_1x+a_2x^2\right).$$

Po wymnożeniu dwóch pierwszych nawiasów po prawej stronie, otrzymujemy

$$\begin{aligned}\sum_{i=0}^{6}d_ix^i=1+\left(b_1+1\right)x+\left(b_2+b_1+\frac{1}{2}\right)x^2+\left(b_2+\frac{1}{2}b_1+\frac{1}{6}\right)x^3+\\+\left(\frac{1}{2}b_2+\frac{1}{6}b_1+\frac{1}{24}\right)x^4+\cdots-\left(a_0+a_1x+a_2x^2\right).\end{aligned}$$

Stąd, warunki aproksymacji (4.26) prowadzą do układu równań:

$$\begin{aligned}1&=a_0,\\ b_1+1&=a_1,\\ b_2+b_1+\frac{1}{2}&=a_2,\\ b_2+\frac{1}{2}b_1+\frac{1}{6}&=0,\\ \frac{1}{2}b_2+\frac{1}{6}b_1+\frac{1}{24}&=0.\end{aligned}$$



Rys. 4.4. Funkcje aproksymowana i aproksymujące z przykładu 4.4

Z dwóch ostatnich równań wyznaczamy współczynniki $b_1 = -\frac{1}{2}$ i $b_2 = \frac{1}{12}$, a następnie z trzech pierwszych równań dostajemy współczynniki licznika, przez proste podstawienie. Poszukiwana funkcja aproksymująca Padé $R_{2,2}(x)$ ma stąd postać

$$R_{2,2}(x) = \frac{1 + \frac{1}{2}x + \frac{1}{12}x^2}{1 - \frac{1}{2}x + \frac{1}{12}x^2} = \frac{12 + 6x + 2x^2}{12 - 6x + x^2}.$$

Na rysunku 4.4 przedstawiono przebiegi: funkcji aproksymowanej e^x (linia ciągła), aproksymującej $R_{2,2}(x)$ (linia przerywana) i funkcji $F_4(x) = \sum_{i=0}^{4} c_i x^i$ (linia kropkowana) złożonej z pierwszych 5 wyrazów szeregu MacLaurina (na których oparta jest aproksymacja Padé). Widać, że aproksymacja Padé, wykorzystująca tę sama wiedzę o funkcji aproksymowanej $f(x)$ co aproksymacja uciętym szeregiem MacLaurina $F_4(x)$, jest dokładniejsza, i to w dość dużym otoczeniu punktu aproksymacji. □

Prosty, uporządkowany *schemat wyznaczania współczynników aproksymacji Padé* zastosowany w przykładzie 4.4 jest ogólny, przedstawimy teraz jego ogólne sfor-

mułowanie. W tym celu, zapiszemy równanie (4.25) w postaci rozwiniętej

$$\sum_{i=0}^{n+2k} d_i x^i = (1+b_1 x+\cdots b_k x^k)(c_0+c_1 x+\cdots+c_{n+k}x^{n+k})-(a_0+a_1 x+\cdots+a_n x^n). \tag{4.27}$$

Można teraz łatwo sformułować wzory na wyznaczanie współczynników a_i i b_i, zgodnie z (4.26) przyrównując do zera współczynniki d_i przy kolejnych potęgach x:

1. Najpierw rozważamy równania zerujące współczynniki d_i, dla $i = n+1, ...$ $..., n+k$:

$$\begin{aligned} b_k c_{n-k+1} + b_{k-1} c_{n-k+2} + \cdots + b_1 c_n + c_{n+1} &= 0, \\ b_k c_{n-k+2} + b_{k-1} c_{n-k+3} + \cdots + b_1 c_{n+1} + c_{n+2} &= 0, \\ &\vdots \\ b_k c_n + b_{k-1} c_{n+1} + \cdots + b_1 c_{n+k-1} + c_{n+k} &= 0, \end{aligned}$$

   przyjmując $c_j = 0$ dla $j < 0$, w razie potrzeby. Otrzymujemy układ k równań liniowych, które można zapisać w postaci

$$\begin{bmatrix} c_{n-k+1} & c_{n-k+2} & \cdots & c_n \\ c_{n-k+2} & c_{n-k+3} & \cdots & c_{n+1} \\ \vdots & \vdots & \vdots & \vdots \\ c_n & c_{n+1} & \cdots & c_{n+k-1} \end{bmatrix} \begin{bmatrix} b_k \\ b_{k-1} \\ \vdots \\ b_1 \end{bmatrix} = \begin{bmatrix} -c_{n+1} \\ -c_{n+2} \\ \vdots \\ -c_{n+k} \end{bmatrix},$$

   skąd obliczamy wartości b_i, $i = 1, ..., k$.

2. Następnie wyznaczamy z równania (4.27) wartości a_i, zerując współczynniki d_i dla $i = 0, ..., n$:

$$\begin{aligned} a_0 &= c_0, \\ a_1 &= c_1 + b_1 c_0, \\ a_2 &= c_2 + b_1 c_1 + b_2 c_0, \\ &\vdots \\ a_n &= c_n + \sum_{i=1}^{\min\{n,k\}} b_i c_{n-i}. \end{aligned}$$

Tak więc w istocie trzeba rozwiązać jedynie prosty układ k równań liniowych zależnych jedynie od k współczynników b_i wielomianu mianownika, współczynniki $a_i$wielomianu licznika uzyskujemy przez proste podstawienia.

Wartości n i k rzędów wielomianów licznika i mianownika funkcji aproksymującej Padé dobierane są arbitralnie, najlepsze rezultaty uzyskuje się, przyjmując $n = k$.

Zadania

1. Dany jest następujący zestaw wartości y_j funkcji w punktach x_j:

$$\begin{array}{c||cccccc} x_j & 0 & 0.2 & 0.4 & 0.6 & 0.8 & 1.0 \\ \hline y_j & 0.5 & 0.5 & 0.7 & 1.5 & 2.2 & 2.3 \end{array}.$$

   Wyznacz wielomiany aproksymujące stopnia pierwszego, drugiego i trzeciego metodą najmniejszych kwadratów, korzystając kolejno z:
   a) wielomianów algebraicznych z bazą naturalną,
   b) wielomianów z bazą ortogonalną uzyskaną z bazy naturalnej drogą standardowej ortogonalizacji Grama-Schmidta.

   Wyznacz błąd maksymalny aproksymacji w każdym z przypadków.

2. Mając bazę funkcyjną

$$\varphi_0(x) = 1, \;\; \varphi_1(x) = x^2,$$

   a) dokonaj aproksymacji średniokwadratowej (tzn. funkcją postaci $f(x) = a_0\varphi_0(x) + a_1\varphi_1(x)$) danych

$$\begin{array}{c||ccccc} x_j & 0 & 0.5 & 1 & 1.5 & 2 \\ \hline y_j & 0.5 & 0.8 & 3 & 5 & 8.2 \end{array},$$

   b) zortogonalizuj podaną bazę i wylicz aproksymację, korzystając z bazy zortogonalizowanej.

3. Wyznacz przybliżenie Padé $R_{1,2}(x)$ funkcji $f(x) = e^x$ w zerze.

4. Wyznacz przybliżenie Padé $R_{2,2}(x)$ funkcji $f(x) = e^{-2x}$ w zerze.

5. Wyznacz przybliżenie Padé $R_{1,2}(x)$ funkcji $f(x) = e^{-x}$ w punkcie $x = 2$ (wskazówka: sprowadź zadanie do wyznaczenia aproksymacji w zerze).

Rozdział 5

Interpolacja

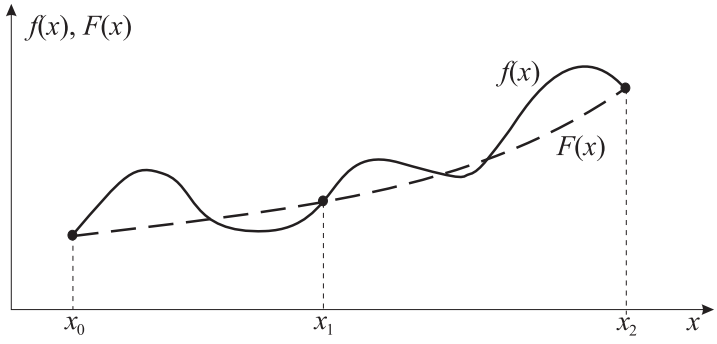
Zadanie interpolacji można postawić następująco:
Dane jest $n+1$ punktów $x_0, ..., x_n$ (tzw. węzły interpolacji), w których znane są wartości $f(x_j)$ funkcji $f(x)$. Należy wyznaczyć funkcję interpolującą $F_n(x)$, należącą do określonej (innej, prostszej) klasy funkcji, spełniającą warunki:

$$f(x_j) = F_n(x_j),\ j = 0, ..., n. \tag{5.1}$$

Interpolację stosujemy, gdy postać analityczna funkcji $f(x)$ nie jest znana, a dane są tylko jej wartości dla skończonego zbioru argumentów, lub gdy analityczna postać $f(x)$ jest znana, ale obliczanie jej wartości jest zbyt pracochłonne. Po wyznaczeniu funkcji interpolującej $F(x)$, można ją wykorzystać do przybliżonego obliczania wartości funkcji oryginalnej $f(x)$ w punktach między węzłami interpolacji.

Na rysunku 5.1 przedstawiono graficznie zadanie interpolacji funkcji ciągłej $f(x)$ w trzech punktach (węzłach interpolacji) $x_0,\ x_1$ i x_2, wielomianem algebraicznym (parabolą) $F(x)$.

Zwróćmy uwagę na *istotną różnicę między zadaniami interpolacji i aproksymacji*: w zadaniu interpolacji wyznaczamy funkcję interpolującą z pewnej klasy (np. wielomian), która ma przechodzić przez *wszystkie* węzły interpolacji i której



Rys. 5.1. Ilustracja graficzna zadania interpolacji.

dokładniejsza postać (np. w przypadku wielomianu zarówno jego rząd, jak i wartości jego współczynników) zależy od liczby i wzajemnego położenia tych punktów; natomiast w zadaniu aproksymacji wyznaczamy parametry funkcji interpolującej zdefiniowanej z dokładnością do wartości parametrów (np. współczynniki wielomianu określonego rzędu) tak, aby minimalizować pewną miarę odległości między funkcją aproksymowaną a aproksymującą – funkcja aproksymująca na ogół nie przechodzi przez żaden z punktów aproksymacji (zob. rozdz. 4).

Ważnymi klasami funkcji interpolujących są wielomiany (w szczególności algebraiczne) i funkcje sklejane, do tych klas ograniczymy dalsze rozważania.

5.1. Interpolacja wielomianami algebraicznymi

Twierdzenie 5.1. *Istnieje dokładnie jeden algebraiczny wielomian interpolacyjny stopnia co najwyżej n, który w punktach $x_0, ..., x_n$ przyjmuje wartości $y_0, ..., y_n$.*

Dowód: Szukany wielomian ma postać

$$W_n(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0. \tag{5.2}$$

Warunki interpolacji (5.1) definiują układ $n+1$ równań liniowych z $n+1$ niewiadomymi współczynnikami $a_0, \ \ldots, \ a_n$:

$$a_0 + a_1 x_j + a_2 x_j^2 + \ \cdots \ + a_n x_j^n = f(x_j), \quad j = 0, ..., n, \tag{5.3}$$

którego macierz $\mathbf{X}$ ma postać:

$$\mathbf{X} = \begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{bmatrix}. \tag{5.4}$$

Jest to tzw. macierz Vandermonde'a, której wyznacznik wyraża się wzorem:

$$\det \mathbf{X} = \prod_{1 \le i < j \le n} (x_j - x_i). \tag{5.5}$$

Wyznacznik ten jest niezerowy, jeśli wszystkie punkty są różne, $x_i \neq x_j$ dla $i \neq j$, $i, j = 0, ..., n$. Stąd układ równań (5.3) ma jednoznaczne rozwiązanie, czyli istnieje dokładnie jeden wielomian interpolacyjny stopnia nie wyższego od n, co kończy dowód. □

Wniosek. W przypadku wielomianów algebraicznych wszystkie wzory interpolacyjne są równoważne (tzn. dla tych samych danych definiują ten sam wielomian, jedynie zapisany w inny sposób).

Teoretycznie, współczynniki wielomianu interpolacyjnego można obliczyć, rozwiązując układ równań liniowych (5.3). Wymaga to jednakże nakładu obliczeń rzędu n^3, nie licząc nakładu na obliczenie współczynników macierzy (5.4). Podane w dalszej części rozdziału wzory interpolacyjne Langrange'a i Newtona pozwalają obliczyć wielomiany interpolacyjne znacznie efektywniej, wymagają nakładu obliczeń rzędu n^2.

5.1.1. Wzór interpolacyjny Lagrange'a

Oznaczmy:

$$L_j(x) \stackrel{\text{df}}{=} \prod_{i=0, i\neq j}^{n} \frac{x - x_i}{x_j - x_i}$$
$$= \frac{(x-x_0)(x-x_1)\cdots(x-x_{j-1})(x-x_{j+1})\cdots(x-x_n)}{(x_j-x_0)(x_j-x_1)\cdots(x_j-x_{j-1})(x_j-x_{j+1})\cdots(x_j-x_n)} \quad (5.6)$$

i zauważmy, że

$$L_j(x_k) = \delta_{jk} = \begin{cases} 1, & k = j \\ 0, & k \neq j. \end{cases}, \quad j,k = 0,1,\ldots,n.$$

Stąd wielomian

$$W_n(x) = \sum_{j=0}^{n} y_j L_j(x) \quad (5.7)$$

jest wielomianem interpolacyjnym, zwanym *wielomianem interpolacyjnym Lagrange'a*. Zauważmy, że definiując ten wielomian, założyliśmy jedynie, że wszystkie węzły interpolacji są różne, natomiast nie poczyniliśmy żadnych założeń dotyczących ich położenia względem siebie – może ono być dowolne (niekoniecznie zgodnie z wzrostem czy maleniem ich wartości).

Można pokazać, zob. np. [3], że jeśli funkcja f ma ciągłą $(n+1)$-szą pochodną na odcinku $[a,\ b]$ zawierającym węzły interpolacji i punkt x, to błąd (reszta) interpolacji wyraża się wzorem

$$r(x) = f(x) - W_n(x) = \frac{f^{(n+1)}(\alpha(x))}{(n+1)!}\omega_n(x), \quad (5.8)$$

gdzie $\alpha(x) \in (a,\ b)$ oraz

$$\omega_n(x) \stackrel{\text{df}}{=} (x-x_0)(x-x_1)\cdots(x-x_n). \quad (5.9)$$

Stąd, oszacowanie błędu interpolacji można wyrazić w postaci

$$|f(x) - W_n(x)| \leq \frac{M_{n+1}}{(n+1)!}|\omega_n(x)|, \quad (5.10)$$

gdzie

$$M_{n+1} = \sup_{x\in[a,b]} \left| f^{(n+1)}(x) \right|. \tag{5.11}$$

W wielu zastosowaniach wygodniejszy od wzoru Lagrange'a jest wzór Newtona.

5.1.2. Wzór interpolacyjny Newtona

Ilorazy różnicowe

Przypomnijmy definicje *ilorazów różnicowych*:

– rzędu pierwszego:

$$f[x_i, x_{i+1}] \overset{\text{df}}{=} \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i},$$

– rzędu drugiego:

$$f[x_i, x_{i+1}, x_{i+2}] \overset{\text{df}}{=} \frac{f[x_{i+1}, x_{i+2}] - f[x_i, x_{i+1}]}{x_{i+2} - x_i},$$

– rzędu n:

$$f[x_i, ..., x_{i+n}] \overset{\text{df}}{=} \frac{f[x_{i+1}, x_{i+2}, ..., x_{i+n}] - f[x_i, x_{i+1}, ..., x_{i+n-1}]}{x_{i+n} - x_i}.$$

Ilorazy różnicowe wygodnie jest obliczać tzw. *schematem tabeli trójkątnej*:

$x_i, f(x_i)$	ilorazy różnicowe		
	rzędu 1	rzędu 2	rzędu 3
$x_0, f(x_0)$	$f[x_0, x_1] = \frac{f(x_1)-f(x_0)}{x_1-x_0}$		
$x_1, f(x_1)$		$f[x_0, x_1, x_2] = \frac{f[x_1,x_2]-f[x_0,x_1]}{x_2-x_0}$	
	$f[x_1, x_2] = \frac{f(x_2)-f(x_1)}{x_2-x_1}$		$f[x_0, x_1, x_2, x_3]$
$x_2, f(x_2)$		$f[x_1, x_2, x_3] = \frac{f[x_2,x_3]-f[x_1,x_2]}{x_3-x_1}$	
	$f[x_2, x_3] = \frac{f(x_3)-f(x_2)}{x_3-x_2}$		$\{f[x_1, x_2, x_3, x_4]\}$
$x_3, f(x_3)$		$\left\{ f[x_2, x_3, x_4] = \frac{f[x_3,x_4]-f[x_2,x_3]}{x_4-x_2} \right\}$	
	$\left\{ f[x_3, x_4] = \frac{f(x_4)-f(x_3)}{x_4-x_3} \right\}$		
$\{x_4, f(x_4)\}$			

Zauważmy, że w tabeli licznik każdego kolejnego ilorazu różnicowego jest różnicą dwu sąsiednich, poprzedzających ilorazów rzędu niższego, a mianownik różnicą dwu skrajnych punktów z mianowników tych ilorazów. Zauważmy też, że dodanie kolejnego punktu (x_4) wymaga jedynie policzenia po jednym ilorazie każdego rzędu, tj. policzenia ilorazów leżących na dolnej krawędzi „trójkąta" – w tabeli

w nawiasach klamrowych pokazano elementy dolnej krawędzi powstałej po dodaniu punktu x_4.

Można łatwo wykazać przez indukcję, że

$$f[x_i, x_{i+1}, ..., x_{i+k}] = \\ = \sum_{j=i}^{j=i+k} \frac{f(x_j)}{(x_j - x_i)(x_j - x_{i+1}) \cdots (x_j - x_{j-1})(x_j - x_{j+1}) \cdots (x_j - x_{i+k})} \\ = \sum_{j=i}^{j=i+k} \frac{f(x_j)}{\prod\limits_{p=i, p\neq j}^{p=i+k} (x_j - x_p)}, \tag{5.12}$$

np.

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{f(x_0)}{x_0 - x_1} + \frac{f(x_1)}{x_1 - x_0},$$

$$f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} \\ = \left(\frac{f(x_2)}{x_2 - x_1} + \frac{f(x_1)}{x_1 - x_2} - \frac{f(x_1)}{x_1 - x_0} - \frac{f(x_0)}{x_0 - x_1}\right)\frac{1}{x_2 - x_0} \\ = \frac{f(x_0)}{(x_0 - x_1)(x_0 - x_2)} + \frac{f(x_1)}{(x_1 - x_0)(x_1 - x_2)} + \frac{f(x_2)}{(x_2 - x_0)(x_2 - x_1)}.$$

Wzór interpolacyjny Newtona

Przedstawmy wielomian interpolacyjny w postaci

$$W_n(x) = W_0(x) + \\ + [W_1(x) - W_0(x)] + [W_2(x) - W_1(x)] + \cdots + [W_n(x) - W_{n-1}(x)],$$

gdzie $W_0 = f(x_0)$. Zauważmy, że

$$W_k(x) - W_{k-1}(x) = A_k(x - x_0)(x - x_1) \cdots (x - x_{k-1}) \\ = A_k \omega_{k-1}(x), \quad k = 1, ..., n, \tag{5.13}$$

gdyż $W_k(x)$ i $W_{k-1}(x)$ mają identyczne wartości w punktach $x_0, x_1, \ldots, x_{k-1}$. Stąd

$$W_n(x) = f(x_0) + A_1\omega_0(x) + A_2\omega_1(x) + ... + A_n\omega_{n-1}(x).$$

Ponieważ przy najwyższej potędze (k-tej) wielomianu $\omega_{k-1}(x)$ stoi jedynka, gdyż

$$\omega_{k-1}(x) \stackrel{\text{df}}{=} (x - x_0)(x - x_1) \cdots (x - x_{k-1}) = x^k + \cdots,$$

a wielomian $W_{k-1}(x)$ jest stopnia $k-1$, więc z (5.13) wynika, że przy najwyższej potędze (k-tej) wielomianu $W_k(x)$ stoi A_k, tzn.

$$W_k(x) = A_k x^k + \cdots .$$

Korzystając z wzoru interpolacyjnego Lagrange'a, mamy natomiast

$$W_k(x) = \sum_{j=0}^{k} f(x_j) \prod_{i=0, i\neq j}^{k} \frac{x - x_i}{x_j - x_i} = \sum_{j=0}^{k} f(x_j) \frac{\prod\limits_{i=0, i\neq j}^{k} (x - x_i)}{\prod\limits_{i=0, i\neq j}^{k} (x_j - x_i)}$$

$$= \sum_{j=0}^{k} f(x_j) \frac{x^k + \cdots}{\prod\limits_{i=0, i\neq j}^{k} (x_j - x_i)} = \sum_{j=0}^{k} \frac{f(x_j)}{\prod\limits_{i=0, i\neq j}^{k} (x_j - x_i)} (x^k + \cdots).$$

Porównując dwa ostatnie wzory i korzystając z (5.12), dostajemy

$$A_k = \sum_{j=0}^{k} \frac{f(x_j)}{\prod\limits_{i=0, i\neq j}^{k} (x_j - x_i)} = f[x_0, x_1, ..., x_k], \qquad k = 1, ..., n. \tag{5.14}$$

Stąd ostatecznie dostajemy *wzór interpolacyjny Newtona* w postaci

$$\begin{aligned} W_n(x) &= f(x_0) + \\ &+ f[x_0, x_1]\,\omega_0(x) + f[x_0, x_1, x_2]\,\omega_1(x) + \cdots + f[x_0, \ldots, x_n]\,\omega_{n-1}(x) \\ &= f(x_0) + \sum_{k=1}^{n} f[x_0, ..., x_k]\ \omega_{k-1}(x). \end{aligned} \tag{5.15}$$

Zwróćmy uwagę, że podobnie jak w przypadku wielomianu interpolacyjnego Lagrange'a, wyprowadzając wielomian Newtona, nie poczyniliśmy żadnych założeń dotyczących wzajemnego położenia węzłów interpolacji – *węzły mogą być ułożone w dowolnej kolejności*. Zaletą wzoru Newtona jest łatwe dodawanie nowych węzłów interpolacji – dodawany, ostatni w tabeli trójkątnej węzeł może, zgodnie z uprzednią uwagą, mieć dowolną wartość.

Przykład 5.1. Dane są cztery punkty: $a = x_0 = 0$, $x_1 = 1$, $x_2 = 2$, $x_3 = b = 3$, wraz z wartościami funkcji w tych punktach: $y_0 = 1, y_1 = 2, y_2 = 4, y_3 = 3$. Należy wyznaczyć wielomian interpolacyjny, stosując, kolejno, wzory Lagrange'a i Newtona.

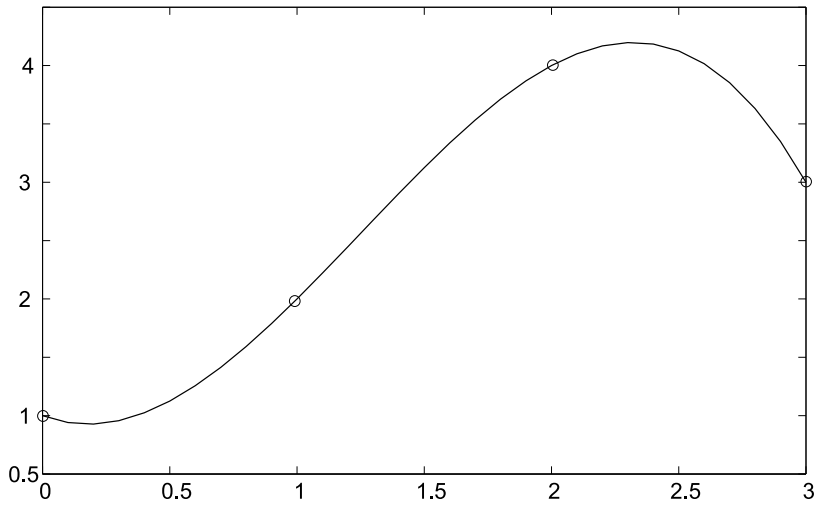
Stosując wzór Lagrange'a:

$$\begin{aligned} W_3(x) &= 1 \cdot \frac{(x-1)(x-2)(x-3)}{-6} + 2 \cdot \frac{x(x-2)(x-3)}{2} + \\ &\quad + 4 \cdot \frac{x(x-1)(x-3)}{-2} + 3 \cdot \frac{x(x-1)(x-2)}{6} \\ &= -\frac{1}{6}(x^3 - 6x^2 + 11x - 6) + (x^3 - 5x^2 + 6x) + \\ &\quad - 2(x^3 - 4x^2 + 3x) + \frac{1}{2}(x^3 - 3x^2 + 2x) \\ &= 1 - \frac{5}{6}x + 2.5x^2 - \frac{2}{3}x^3. \end{aligned}$$

Stosując wzór Newtona:

$$\begin{aligned} W_3(x) &= 1 + 1 \cdot (x-0) + \frac{1}{2}(x-0)(x-1) + \frac{-2}{3}(x-0)(x-1)(x-2) \\ &= 1 + x + \frac{1}{2}(x^2 - x) - \frac{2}{3}(x^3 - 3x^2 + 2x) \\ &= 1 - \frac{5}{6}x + 2.5x^2 - \frac{2}{3}x^3. \end{aligned} \tag{5.16}$$

Otrzymujemy, oczywiście, ten sam wielomian interpolacyjny, tylko inne są algorytmy jego wyznaczenia. Obliczony wielomian pokazany jest na rysunku 5.2. □



Rys. 5.2. Wielomian interpolacyjny (5.16)

Wzór interpolacyjny Newtona może być wykorzystany do efektywnego obliczania wartości wielomianu interpolującego między węzłami, co prowadzi do tzw. *schematu Hornera*, postaci jak np. dla $n = 4$:

$$\begin{aligned} W_n(x) &= f(x_0) + \\ &\quad + A_1(x - x_0) + \\ &\quad + A_2(x - x_0)(x - x_1) + \\ &\quad + A_3(x - x_0)(x - x_1)(x - x_2) + \\ &\quad + A_4(x - x_0)(x - x_1)(x - x_2)(x - x_3) \\ &= f(x_0) + \\ &\quad + (x - x_0)[A_1 + (x - x_1)[A_2 + (x - x_2)[A_3 + (x - x_3)A_4]]]. \end{aligned}$$

Ponieważ kolejność węzłów interpolacji jest dowolna, więc wzór Newtona można również formułować, ustawiając kolejne punkty w odwrotnej kolejności, dostajemy wówczas tzw. *wzór Newtona wstecz*

$$\begin{aligned} W_n(x) = f(x_n) + f[x_n, x_{n-1}](x - x_n) + \\ + f[x_n, x_{n-1}, x_{n-2}](x - x_n)(x - x_{n-1}) + \\ \cdots + f[x_n, x_{n-1}, \ldots, x_0](x - x_n) \cdots (x - x_1). \end{aligned} \quad (5.17)$$

Jest to przydatne, gdy w konkretnej aplikacji punkty są kolejno dodawane (ostatni jest x_n) i wygodna jest interpolacja „wstecz" od ostatniego punktu.

Przypadek węzłów równo odległych*

Przyjmijmy, że kolejne punkty są równo odległe, tzn.

$$x_i = x_0 + ih, \qquad i = 0, ..., n,$$

i oznaczmy $y_{i\pm k} = f(x_i \pm kh)$.

Przypomnijmy definicję *różnic zwykłych (progresywnych) funkcji:*
– rzędu pierwszego:

$$\Delta f(x_i) = f(x_i + h) - f(x_i), \quad \text{tzn. } \Delta y_i = y_{i+1} - y_i,$$

– rzędu drugiego:

$$\begin{aligned} \Delta^2 f(x_i) &= \Delta(\Delta f(x_i)) \\ &= \Delta[f(x_i + h) - f(x_i)] \\ &= \Delta f(x_i + h) - \Delta f(x_i) \\ &= f(x_i + 2h) - f(x_i + h) - (f(x_i + h) - f(x_i)) \\ &= f(x_i + 2h) - 2f(x_i + h) + f(x_i), \end{aligned}$$

*Materiał uzupełniający.

tzn.
$$\Delta^2 y_i = \Delta\left(\Delta y_i\right) = y_{i+2} - 2y_{i+1} + y_i,$$

– rzędu n:
$$\Delta^n y_i = \Delta\left(\Delta^{n-1} y_i\right) = \sum_{j=0}^{n} (-1)^j \binom{n}{j} y_{i+n-j}.$$

Operator Δ jest operatorem liniowym, ma szereg własności analogicznych do operatora różniczkowania, w szczególności

$$\Delta^m\left(\Delta^n f\right) = \Delta^{m+n} f.$$

Przypomnijmy również definicję *różnic wstecznych funkcji:*
– rzędu pierwszego:

$$\bigtriangledown f\left(x_i\right) = f\left(x_i\right) - f\left(x_i - h\right), \quad \text{tzn.} \bigtriangledown y_i = y_i - y_{i-1},$$

– rzędu drugiego:

$$\begin{aligned}\bigtriangledown^2 y_i &= \bigtriangledown\left(\bigtriangledown y_i\right) \\ &= \bigtriangledown\left(y_i - y_{i-1}\right) = \bigtriangledown y_i - \bigtriangledown y_{i-1} \\ &= \left(y_i - y_{i-1}\right) - \left(y_{i-1} - y_{i-2}\right) = y_i - 2y_{i-1} + y_{i-2},\end{aligned}$$

– rzędu n:
$$\bigtriangledown^n y_i = \bigtriangledown\left(\bigtriangledown^{n-1} y_i\right).$$

Można pokazać, że zachodzi następująca relacja między różnicami zwykłymi a wstecznymi:
$$\bigtriangledown^k y_i = \Delta^k y_{i-k}, \qquad k = 0, 1, ..., n.$$

Różnice zwykłe i wsteczne wygodnie jest liczyć w tabelach trójkątnych, podobnie jak ilorazy różnicowe.

Ponieważ
$$\begin{aligned} f\left(x_0\right) &= y_0 \\ f\left[x_0, x_1\right] &= \frac{y_1 - y_0}{h} = \frac{\Delta y_0}{h}, \\ f\left[x_0, x_1, x_2\right] &= \frac{\frac{\Delta y_1}{h} - \frac{\Delta y_0}{h}}{2h} = \frac{\Delta^2 y_0}{2h^2}, \\ &\vdots \\ f\left[x_0, x_1, ..., x_k\right] &= \frac{\Delta^k y_0}{k! h^k}, \quad k = 1, 2, ..., n, \end{aligned}$$

więc *wzór interpolacyjny Newtona dla węzłów równo odległych* można zapisać w postaci

$$W_n(x) = y_0 + \frac{\Delta y_0}{h}(x - x_0) + \frac{\Delta^2 y_0}{2h^2}(x - x_0)(x - x_1) + \cdots \\ \cdots + \frac{\Delta^n y_0}{n!h^n}(x - x_0)\cdots(x - x_{n-1}). \quad (5.18)$$

Podobnie, ponieważ

$$f[x_n, x_{n-1}, ..., x_{n-k}] = \frac{\nabla^k y_n}{k!h^k},$$

więc *wzór interpolacyjny Newtona wstecz dla węzłów równo odległych* można przedstawić w postaci

$$W_n(x) = y_n + \frac{\nabla y_n}{h}(x - x_n) + \frac{\nabla^2 y_n}{2h^2}(x - x_n)(x - x_{n-1}) + \cdots \\ \cdots + \frac{\nabla^n y_n}{n!h^n}(x - x_n)\cdots(x - x_1). \quad (5.19)$$

Zbieżność procesu interpolacyjnego

W przypadku ogólnym, wraz ze wzrostem rzędu wielomianu, tj. liczby węzłów interpolacji na danym odcinku, wielomian interpolacyjny nie zbiega do funkcji interpolowanej w sposób jednostajny. W szczególności, dla funkcji niegładkich może nie być w ogóle zbieżności. Stąd dla takich funkcji nie jest wskazane stosowanie interpolacji wielomianami, szczególnie wyższych rzędów. Dla tej klasy funkcji lepsze rezultaty daje z reguły interpolacja funkcjami sklejanymi.

Przykład 5.2. Na rysunku 5.3 przedstawiono przebieg wielomianów algebraicznych interpolujących funkcję $f(x) = |x|$ na odcinku $[-1, 1]$, odpowiednio w 5 i 9 równomiernie rozłożonych punktach (wielomiany 4 i 8 stopnia). □

5.2. Interpolacja funkcjami sklejanymi

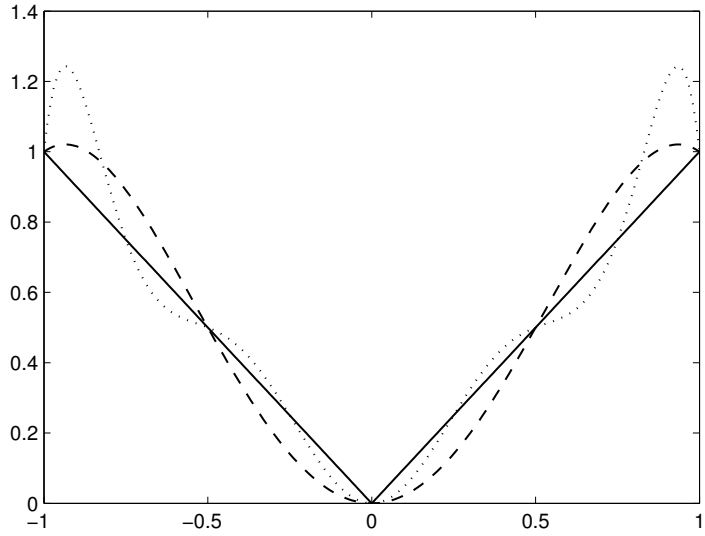
Niech będzie dany przedział $[a, b]$ i jego podział punktami x_i, $i = 0, ..., n$, taki, że:

$$a = x_0 < x_1 < x_2 < ... < x_n = b.$$

Podział ten oznaczmy przez $\triangle_n$.

Definicja. Funkcję $s(x) = s(x; \triangle_n)$ określoną na przedziale $[a, b]$ nazywamy *funkcją sklejaną* (*spline function*) stopnia $m \geq 1$, jeśli:

1. na każdym z podprzedziałów $[x_i, x_{i+1}]$ funkcja $s(x)$ jest wielomianem stopnia co najwyżej m, $i = 0, 1, ...n - 1$;



Rys. 5.3. Wykresy funkcji $f(x) = |x|$ oraz wielomianu interpolacyjnego 4 stopnia (linia przerywana) i 8 stopnia (linia kropkowana), węzły interpolacji równomiernie rozłożone na odcinku $[-1, 1]$

2. funkcja $s\,(x)$ jest klasy $C^{(m-1)}_{[a,b]}$ (ma ciągłą pochodną rzędu $m-1$ na $[a, b\,]$).

Będziemy rozważać funkcje sklejane *interpolujące*, tj. spełniające

$$s\,(x_i) = y_i, \qquad i = 0, ..., n.$$

Jednoznaczność określenia funkcji sklejanej:

Z warunku 1. dla każdego $x \in [x_i, x_{i+1}]$ mamy

$$s\,(x) = a_{im}x^m + a_{i(m-1)}x^{m-1} + \cdots + a_{i1}x + a_{i0},$$

stąd dla każdego i mamy $(m+1)$ współczynników a_{im}, co daje w sumie $n\,(m+1)$ *współczynników do wyznaczenia.*

Z kolei mamy *następujące warunki (ograniczenia):*

– $s\,(x_i) = y_i$, $i = 0, ..., n$, co daje $n+1$ warunków,
– ciągłość funkcji $s(x)$ i jej pochodnych rzędu $1, ..., m-1$ w $n-1$ punktach $x_1, ..., x_{n-1}$ (punkty sklejenia), co daje $m\,(n-1) = mn-m$ warunków.

Tak więc łącznie mamy $(m+1)\,n-m+1$ warunków na $n(m+1)$ nieznanych współczynników funkcji sklejanej. Stąd wynika następująca *liczba stopni swobody*:

$$n(m+1) - [(m+1)\,n - m + 1] = m - 1.$$

Mamy więc $m-1$ stopni swobody, tzn. dla funkcji sklejanych liniowych brak stopni swobody, dla funkcji sklejanych drugiego rzędu, 1 stopień swobody, itd. Chcąc więc jednoznacznie wyznaczać funkcję sklejaną stopnia $m > 1$ trzeba sformułować dodatkowe warunki na pochodne funkcji (wartości funkcji we wszystkich punktach są określone).

Dla jednoznaczngo wyznaczenia funkcji sklejanej stopnia drugiego trzeba założyć jeden dodatkowy warunek, np. jeden z poniższych warunków na wartość brzegową (jednostroną) pochodnej funkcji:

$$\frac{ds}{dx}\left(b^-\right) = \beta_1, \tag{5.20a}$$

$$\frac{ds}{dx}\left(a^+\right) = \alpha_1, \tag{5.20b}$$

$$\frac{ds}{dx}\left(a^+\right) = \frac{ds}{dx}\left(b^-\right). \tag{5.20c}$$

Natomiast chcąc jednoznacznie wyznaczyć funkcję sklejaną stopnia trzeciego trzeba założyć dwa dodatkowe warunki na pochodne, np. jedną z niżej podanych par warunków na warunki brzegowe:

$$\frac{ds}{dx}\left(a^+\right) = \alpha_1, \quad \frac{ds}{dx}\left(b^-\right) = \beta_1, \tag{5.21a}$$

$$\frac{d^2s}{dx^2}\left(a^+\right) = \alpha_2, \quad \frac{d^2s}{dx^2}\left(b^-\right) = \beta_2, \tag{5.21b}$$

$$\frac{ds}{dx}\left(a^+\right) = \frac{ds}{dx}\left(b^-\right), \quad \frac{d^2s}{dx^2}\left(a^+\right) = \frac{d^2s}{dx^2}\left(b^-\right). \tag{5.21c}$$

Dodatkowe warunki na wartości pochodnych nie muszą być nakładane w punktach brzegowych (krańcowych) przedziału interpolacji, można też przyjmować warunki na pochodne w punktach pośrednich (sklejenia). Przykładowo, dodatkowymi warunkami domyślnymi funkcji „`spline`" w środowisku MATLAB (realizującej funkcję sklejaną 3 stopnia) są wymagania ciągłości trzeciej pochodnej funkcji sklejanej w punktach x_1 (pierwszy punkt sklejenia) i x_{n-1} (ostatni punkt sklejenia):

$$\frac{d^3s}{dx^3}(x_1^+) = \frac{d^3s}{dx^3}(x_1^-), \quad \frac{d^3s}{dx^3}(x_{n-1}^+) = \frac{d^3s}{dx^3}(x_{n-1}^-). \tag{5.21d}$$

Punkty x_1 i x_{n-1} przestają być tym samym typowymi punktami sklejenia, stąd nazwa tych warunków „*not-a-knob conditions*". Warunki te są równoważne wymaganiu takich samych współczynników przy najwyższej (trzeciej) potędze wielomianów funkcji sklejanej określanych na dwóch pierwszych podprzedziałach i na dwóch ostatnich podprzedziałach. Stąd funkcja sklejana ma w punktach x_1 i x_{n-1} wszystkie (i ciągłe) pochodne, gdyż z definicji funkcji sklejanej ma ciągłe pochodne pierwszego i drugiego rzędu, a z warunków „*not-a-knob conditions*" wynika równość pochodnych trzeciego rzędu.

Przykład 5.3. Dane są cztery punkty $a = x_0, x_1, x_2, x_3 = b$, wraz z wartościami funkcji w tych punktach, odpowiednio y_0, y_1, y_2, y_3. Należy wyznaczyć układ równań jednoznacznie określający funkcję sklejaną 2 stopnia $(m = 2)$, przy warunku

$$\frac{ds}{dx}\left(a^{+}\right) = s_0. \tag{5.22}$$

Na każdym z trzech podprzedziałów funkcję $s(x)$ można zapisać w postaci:

$$s(x) = s_i(x) = y_i + b_i(x - x_i) + c_i(x - x_i)^2, \quad x \in [x_i, x_{i+1}], \quad i = 0, 1, 2, \tag{5.23}$$

co redukuje liczbę poszukiwanych współczynników b_i i c_i funkcji sklejanej z 9 do 6 (spełnione pierwsze $n = 3$ warunki). Pozostałe warunki to:

– wymaganie na wartość funkcji w punkcie końcowym:

$$s_2(x_3) = y_2 + b_2(x_3 - x_2) + c_2(x_3 - x_2)^2 = y_3, \tag{5.24}$$

– wymagania na ciągłość funkcji w dwóch punktach sklejenia:

$$y_0 + b_0(x_1 - x_0) + c_0(x_1 - x_0)^2 = y_1, \tag{5.25}$$

$$y_1 + b_1(x_2 - x_1) + c_1(x_2 - x_1)^2 = y_2, \tag{5.26}$$

– warunki ciągłości pierwszych pochodnych w punktach sklejenia:

$$b_0 + 2c_0(x_1 - x_0) = b_1, \tag{5.27}$$

$$b_1 + 2c_1(x_2 - x_1) = b_2, \tag{5.28}$$

– warunek brzegowy (5.22):

$$b_0 + 2c_0(x_0 - x_0) = b_0 = s_{0.} \tag{5.29}$$

Układ równań (5.24)–(5.29) jest poszukiwanym układem 6 równań do wyznaczenia sześciu współczynników b_i i c_i funkcji sklejanej. Zauważmy, że jest to *układ równań liniowych*.

Przyjmując wartości liczbowe

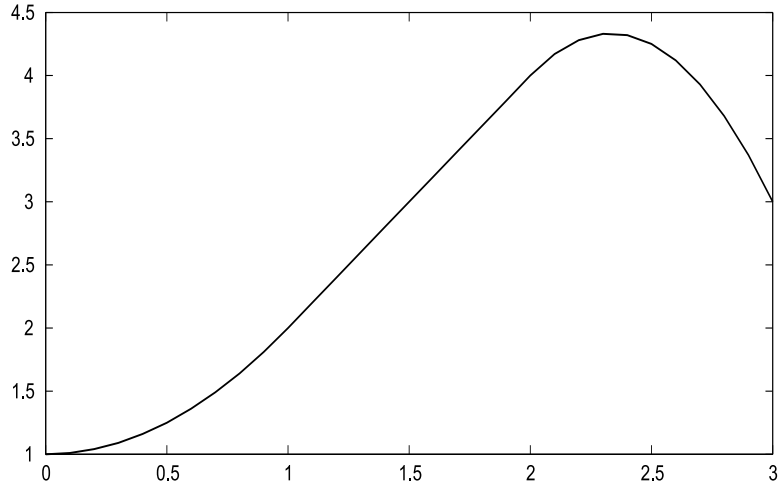
$$\begin{array}{c|cccc} x_i & 0 & 1 & 2 & 3 \\ \hline y_i & 1 & 2 & 4 & 3 \end{array}, \qquad s_0 = 0,$$

dostajemy układ równań

$$\begin{aligned} b_2 + c_2 &= -1, \\ b_0 + c_0 &= 1, \\ b_1 + c_1 &= 2, \\ b_0 + 2c_0 &= b_1, \\ b_1 + 2c_1 &= b_2, \\ b_0 &= 0. \end{aligned}$$

Po rozwiązaniu otrzymujemy następującą postać funkcji sklejanej:

$$s(x) = 1 + x^2, \quad x \in [0, 1],$$
$$s(x) = 2 + 2(x - 1), \quad x \in [1, 2],$$
$$s(x) = 4 + 2(x - 2) - 3(x - 2)^2, \quad x \in [2, 3].$$



Rys. 5.4. Wykres funkcji sklejanej z przykładu 5.3

Wykres tej funkcji zamieszczono na rysunku 5.4. Jako ćwiczenie pozostawiamy wyznaczenie funkcji sklejanej dla innych wartości s_0 warunku brzegowego, oraz sporządzenie wykresów otrzymanych funkcji. □

Przykład 5.4. Dane są trzy punkty $a = x_0 = -1,\ x_1 = 0,\ x_2 = 2$ wraz z wartościami funkcji w tych punktach, odpowiednio $y_0 = 0,\ y_1 = 4,\ y_2 = 2$, tzn. mamy tablicę danych

x_i	-1	0	2
y_i	0	4	2

.

Dla powyższych danych, należy wyznaczyć interpolującą funkcję sklejaną $s(x)$ trzeciego stopnia $(m = 3)$, przy warunkach brzegowych

$$\frac{ds}{dx}(-1^+) = 2, \tag{5.30a}$$
$$\frac{ds}{dx}(2^-) = -25. \tag{5.30b}$$

Mamy dwa podprzedziały, $[-1,0]$ i $[0,2]$, na każdym z nich funkcję $s(x)$ można zapisać w postaci:

$$\begin{aligned} s(x) = s_0(x) &= 0 + b_0(x+1) + c_0(x+1)^2 + d_0(x+1)^3, \quad x \in [-1,0], \\ s(x) = s_1(x) &= 4 + b_1(x-0) + c_1(x-0)^2 + d_1(x-0)^3, \quad x \in [0,2], \end{aligned}$$

co redukuje liczbę poszukiwanych współczynników b_i, c_i i d_i funkcji sklejanej z 8 do 6 (spełnione pierwsze $n = 2$ warunki). Pozostałe warunki to:

– wymaganie na wartość funkcji w punkcie końcowym:

$$\begin{aligned} s_1(x_2) = y_1 + b_1(x_2 - x_1) + c_1(x_2 - x_1)^2 + d_1(x_2 - x)^3 = y_2, \\ \text{tzn.} \qquad 4 + 2b_1 + 4c_1 + 8d_1 = 2, \end{aligned} \tag{5.31}$$

– wymaganie na ciągłość funkcji w punkcie sklejenia:

$$\begin{aligned} y_0 + b_0(x_1 - x_0) + c_0(x_1 - x_0)^2 + d_0(x_1 - x_0)^2 = y_1, \\ \text{tzn.} \qquad b_0 + c_0 + d_0 = 4, \end{aligned} \tag{5.32}$$

– warunek ciągłości pierwszej pochodnej w punkcie sklejenia:

$$\begin{aligned} b_0 + 2c_0(x_1 - x_0) + 3d_0(x_1 - x_0)^2 = b_1, \\ \text{tzn.} \qquad b_0 + 2c_0 + 3d_0 = b_1, \end{aligned} \tag{5.33}$$

– warunek ciągłości drugiej pochodnej w punkcie sklejenia:

$$\begin{aligned} 2c_0 + 6d_0(x_1 - x_0) = 2c_1, \\ \text{tzn.} \qquad 2c_0 + 6d_0 = 2c_1, \end{aligned} \tag{5.34}$$

– warunki brzegowe (5.30a) i (5.30b), odpowiednio:

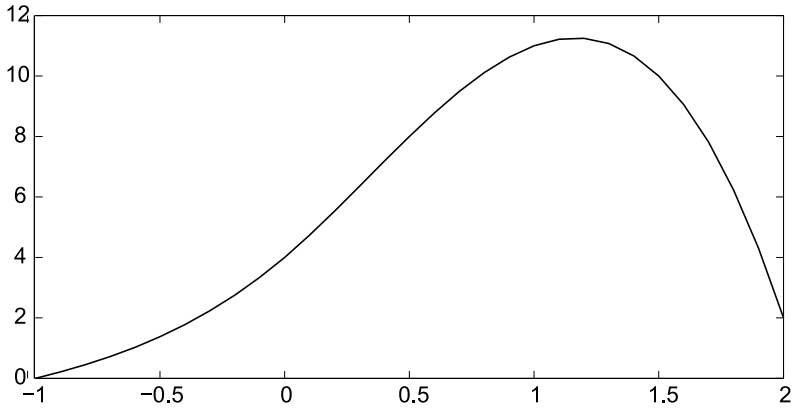
$$b_0 = 2, \tag{5.35}$$

$$\begin{aligned} b_1 + 2c_1(x_2 - x_1) + 3d_1(x_2 - x_1)^2 = -25, \\ \text{tzn.} \qquad b_1 + 4c_1 + 12d_1 = -25, \end{aligned} \tag{5.36}$$

Równania (5.31)–(5.36) są układem sześciu równań liniowych do wyznaczenia sześciu współczynników $b_i, c_i, d_i,\ i = 0, 1$. Rozwiązanie tego układu pozostawiamy czytelnikowi. Jako rezultat otrzymujemy następującą funkcję sklejaną:

$$\begin{aligned} s(x) = s_0(x) &= 2(x+1) + (x+1)^2 + (x+1)^3, \quad x \in [-1,0], \\ s(x) = s_1(x) &= 4 + 7x + 4x^2 - 4x^3, \quad x \in [0,2]. \end{aligned}$$

Wykres tej funkcji zamieszczono na rysunku 5.5. □



Rys. 5.5. Wykres funkcji sklejanej z przykładu 5.4

Przykład 5.5. Kontynuując porównanie przedstawione w przykładzie 5.2, na rysunku 5.6 przedstawiono porównanie wielomianu interpolacyjnego 8 stopnia oraz funkcji sklejanej 3 stopnia (z dodatkowymi warunkami zerowania wartości pierwszej pochodnej na krańcach odcinka interpolacji) dla funkcji $f(x) = |x|$ na odcinku $[-1, 1]$, z 9 równomiernie rozłożonymi punktami (węzłami interpolacji).

Natomiast na rysunku 5.7 przedstawiono porównanie wykresów tych samych funkcji, ale zakładając inne warunki dodatkowe na pochodne funkcji sklejanej 3 stopnia, tym razem warunki typu *not-a-knob conditions*.

Rezultaty przedstawione na powyższych rysunkach pokazują znacznie lepszą jakość interpolacji funkcją sklejaną, w porównaniu do interpolacji pojedynczym wielomianem algebraicznym wyższego stopnia. □

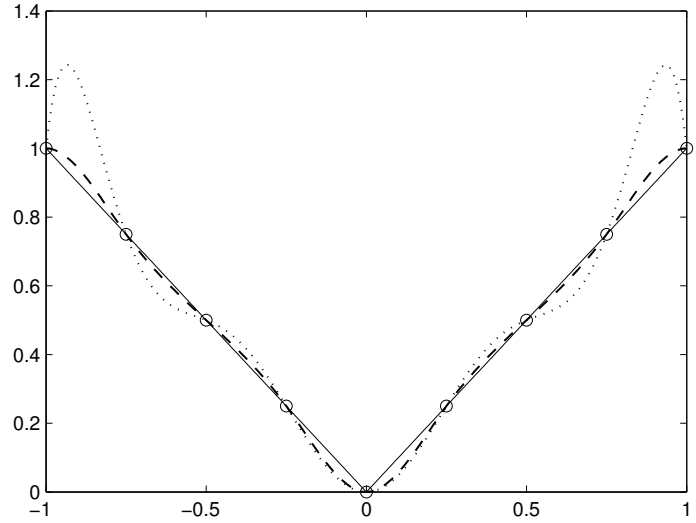
Przykład 5.6.* Wyznaczymy układ równań na współczynniki funkcji sklejanej stopnia $m = 3$, *dla zadanej wartością* n *dowolnej liczby punktów* $x_0, ..., x_n$ i wartości funkcji interpolowanej w tych punktach $y_0, ..., y_n$, dla przypadku warunków brzegowych (5.21b) – w formie wygodnej do efektywnej implementacji komputerowej.

Dla każdego podprzedziału $[x_i, x_{i+1}]$, $i = 0, ..., n-1$, definiujemy funkcję sklejaną postaci

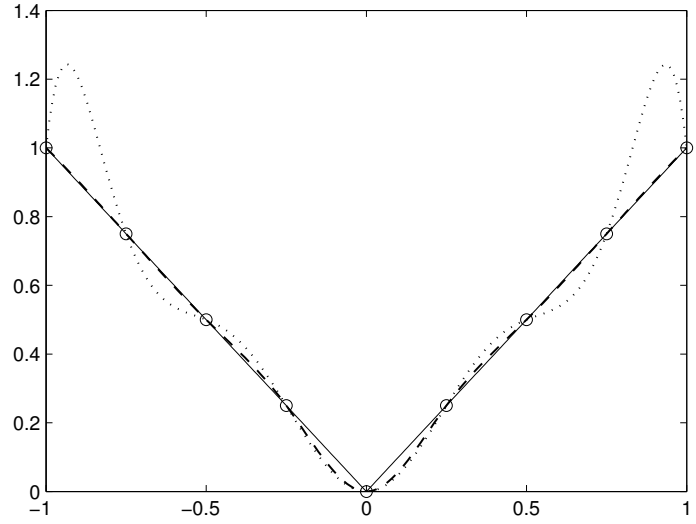
$$s(x) = s_i(x) = y_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3.$$

Oznaczmy przez h_i długości podprzedziałów: $h_i \stackrel{\text{df}}{=} x_{i+1} - x_i$, wówczas możemy wygodniej formułować warunki na współczynniki funkcji sklejanej:

*Materiał uzupełniający.



Rys. 5.6. Porównanie wielomianu interpolacyjnego 8 stopnia (linia kropkowana) i funkcji sklejanej 3 stopnia z zerowymi wartościami pochodnej pierwszego rzędu na krańcach (linia przerywana), dla funkcji $f(x) = |x|$, przy 9 węzłach interpolacji



Rys. 5.7. Porównanie wielomianu interpolacyjnego 8 stopnia (linia kropkowana) i funkcji sklejanej 3 stopnia z dodatkowymi warunkami typu *not-a-knob* (linia przerywana), dla funkcji $f(x) = |x|$, przy 9 węzłach interpolacji

– Z ciągłości $s\,(x)$ w węzłach pośrednich oraz warunku $y_n = s_{n-1}(x_n)$, mamy

$$y_{i+1} = y_i + b_i h_i + c_i h_i^2 + d_i h_i^3, \qquad i = 0, ..., n-1. \tag{5.37}$$

– Z kolei, z ciągłości $\frac{d^2 s}{dx^2}\,(x)$ w węzłach pośrednich wynika

$$2c_{i+1} = 2c_i + 6d_i h_i, \quad i = 0, ..., n-1, \tag{5.38}$$

gdzie $2c_n$ definiujemy jako dane przez drugi z warunków brzegowych (5.21b), tzn.

$$2c_n \overset{\text{df}}{=} \frac{d^2 s}{dx^2}\left(b^-\right) = \beta_2. \tag{5.39}$$

Stąd możemy zapisać (5.38) w postaci

$$d_i = \frac{c_{i+1} - c_i}{3h_i}, \quad i = 0, ..., n-1. \tag{5.40}$$

Z (5.37) i (5.40) otrzymujemy

$$y_{i+1} - y_i = b_i h_i + c_i h_i^2 + \frac{1}{3}\left(c_{i+1} - c_i\right) h_i^2,$$

czyli

$$b_i = \frac{y_{i+1} - y_i}{h_i} - \frac{1}{3}\left(c_{i+1} + 2c_i\right) h_i, \qquad i = 0, ..., n-1. \tag{5.41}$$

– Dalej, z ciągłości $\frac{ds}{dx}\,(x)$ w węzłach pośrednich wynika

$$b_{i+1} = b_i + 2c_i h_i + 3d_i h_i^2, \qquad i = 0, ..., n-2. \tag{5.42}$$

Po wstawieniu wzorów (5.41) i (5.40) na b_i i d_i do (5.42), otrzymujemy:

$$\frac{y_{i+2} - y_{i+1}}{h_{i+1}} - \frac{h_{i+1}}{3}(c_{i+2} + 2c_{i+1}) = \frac{y_{i+1} - y_i}{h_i} - \frac{h_i}{3}\left(c_{i+1} + 2c_i\right) + 2c_i h_i + (c_{i+1} - c_i)h_i,$$

i dalej, po przekształceniach:

$$\frac{y_{i+2} - y_{i+1}}{h_{i+1}} - \frac{y_{i+1} - y_i}{h_i} = (h_i - \frac{2h_i}{3})c_i + (\frac{2h_{i+1}}{3} + \frac{2h_i}{3})c_{i+1} + \frac{h_{i+1}}{3}c_{i+2}.$$

Po pomnożeniu stronami przez 3, dostajemy ostatecznie:

$$3(\frac{y_{i+2} - y_{i+1}}{h_{i+1}} - \frac{y_{i+1} - y_i}{h_i}) = h_i c_i + 2(h_{i+1} + h_i)c_{i+1} + h_{i+1}c_{i+2},$$

$i = 0, 1, ..., n-2$. Mamy więc $n-1$ równań na $n+1$ współczynników c_i (c_n też tu występuje) oraz dwa dodatkowe równania z warunków brzegowych (5.21b):

$$\frac{d^2 s}{dx^2}\left(a^+\right) = 2c_0 = \alpha_2,$$

$$\frac{d^2 s}{dx^2}\left(b^-\right) = 2c_n = \beta_2.$$

Stąd, definiując

$$v_i \stackrel{\text{df}}{=} 3\left(\frac{y_{i+1}-y_i}{h_i} - \frac{y_i - y_{i-1}}{h_{i-1}}\right), \quad i = 1, ..., n-1,$$

$$u_i \stackrel{\text{df}}{=} 2(h_i + h_{i-1}), \quad i = 1, ..., n-1,$$

dostajemy ostatecznie układ równań postaci

$$\begin{bmatrix} u_1 & h_1 & 0 & \cdots & & 0 \\ h_1 & u_2 & h_2 & & & 0 \\ 0 & h_2 & u_3 & h_3 & & 0 \\ & & & \ddots & & \\ \vdots & & & & u_{n-2} & h_{n-2} \\ 0 & 0 & 0 & \cdots & h_{n-2} & u_{n-1} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \\ \vdots \\ c_{n-2} \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} v_1 - 0.5h_0\alpha_2 \\ v_2 \\ \\ \vdots \\ v_{n-2} \\ v_{n-1} - 0.5h_{n-1}\beta_2 \end{bmatrix} \tag{5.43}$$

Po obliczeniu współczynników c_i wyliczamy współczynniki b_i i d_i, odpowiednio z wzorów (5.41), (5.40). □

Przestrzenie funkcji sklejanych (splajnów)*

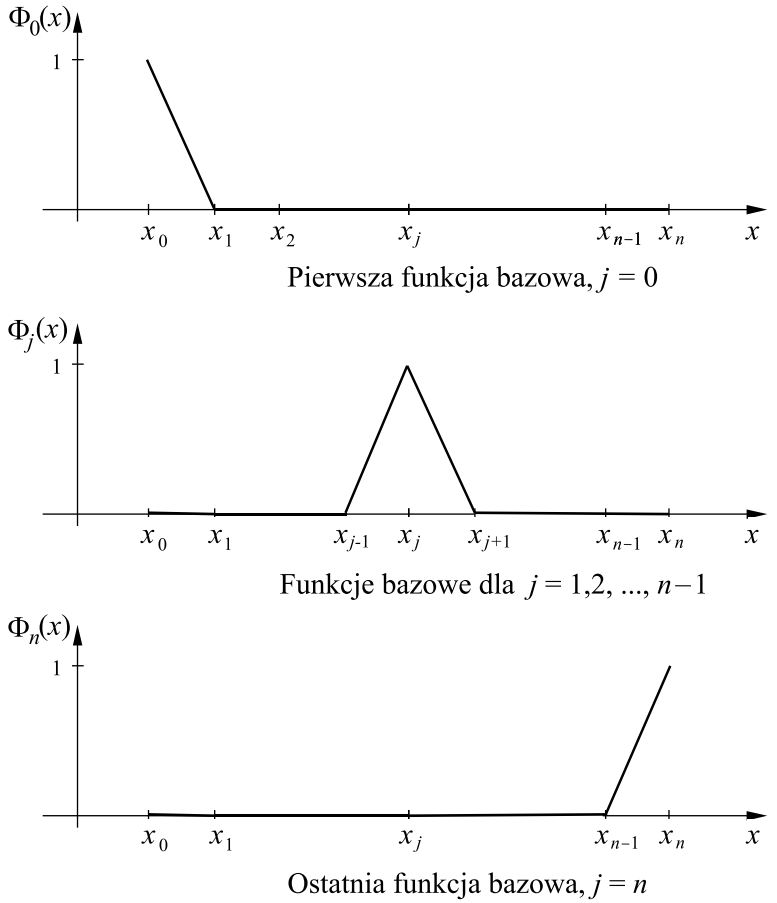
Oznaczmy przez $S_m(\triangle_n)$ zbiór wszystkich funkcji sklejanych stopnia m, określonych na odcinku $[a, b]$ z danym (ustalonym, np. równomiernym) podziałem $\triangle_n$. Zbiór ten można traktować jako *skończenie wymiarową przestrzeń liniową*.

Przykładem takiej przestrzeni, dla $m = 1$, jest *zbiór wszystkich funkcji sklejanych kawałkami liniowych, określonych na ustalonym podziale odcinka* $[a, b]$. Ograniczmy się do równomiernego podziału odcinka, tzn. $x_i = x_0 + ih, i = 0, ..., n$ ($x_0 = a, x_n = b$, $h = \frac{b-a}{n}$). Dowolną funkcję sklejaną kawałkami liniową można przedstawić w bazie przestrzeni $S_1(\triangle_n)$ złożonej z funkcji $\Phi_j(x)$, $j = 0, ..., n$,

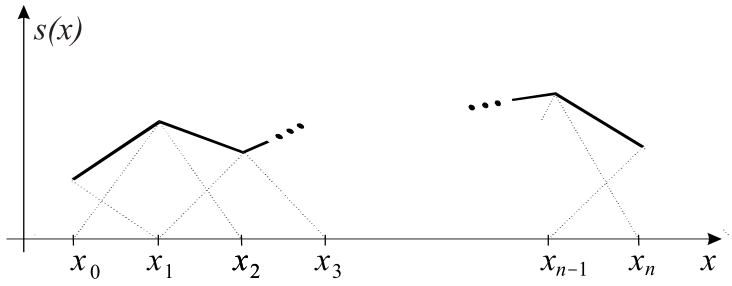
$$s(x) = \sum_{j=0}^{n} y_j \Phi_j(x),$$

gdzie postacie funkcji bazowych przedstawiono na rysunku 5.8. Przykładową funkcję sklejaną pierwszego stopnia pokazano na rysunku 5.9.

*Materiał uzupełniający.



Rys. 5.8. Postać funkcji bazowych przestrzeni funkcji sklejanych pierwszego stopnia na odcinku $[x_0,\ x_n]$



Rys. 5.9. Przykładowa funkcja sklejana pierwszego stopnia

Przestrzeń $S_1(\triangle_n)$ jest wymiaru $n+1$, funkcja $s\,(x)$ jest w pełni określona przez $n+1$ liczb $y_0, y_1, ..., y_n$. Stąd, jeśli $f\,(x)$ jest funkcją poszukiwaną na przedziale $[a, b\,]$ i ograniczymy się do jej przybliżenia interpolacyjnego funkcją sklejaną pierwszego stopnia, to nieskończenie wymiarowy problem poszukiwania takiej funkcji (np. w optymalizacji dynamicznej, itp.) sprowadzamy do problemu skończenie wymiarowego ($(n+1)$-wymiarowego). Podobnie jest dla funkcji sklejanych wyższych stopni, tylko funkcje bazowe mają tam inną postać. Podejście to ma duże znaczenie praktyczne. Największe znaczenie mają funkcje sklejane stopnia $m = 3$.

Zadania

1. Znajdź wielomian interpolacyjny, który w punktach $x_i = -2, 1, 2, 4$ przyjmuje wartości $3, 1, -3, 8$, korzystając z:
   a) wzoru Lagrange'a,
   b) wzoru Newtona.

2. Zmodyfikuj wielomian interpolacyjny z zadania 1 tak, aby dodatkowo:
   a) w punkcie $x = 6$ przyjmował wartość 6,
   b) w punkcie $x = 3$ przyjmował wartość -2.

3. Oblicz $\sqrt{117}$, korzystając z interpolacji funkcji $\sqrt{x}$ w węzłach:
   a) $81, 100, 121$,
   b) $100, 121, 144$,
   c) $100, 121, 144, 169$.

4. Wyznacz funkcję sklejaną $s(x)$ drugiego stopnia, przyjmującą w punktach $x_i = 1, 2, 4, 5$ (punkty krańcowe i podziału) wartości, odpowiednio, $2, 3, 1, 1$ i spełniającą warunek $\frac{ds}{dx}\,(1^+) = 1$.

5. Wyznacz funkcję sklejaną $s(x)$ trzeciego stopnia, przyjmującą w punktach $x_i = 0, 2, 4, 6$ (punkty krańcowe i podziału) wartości, odpowiednio, $2, 10, 4, 4$ i spełniającą warunek $\frac{ds}{dx}\,(0^+) = 2$, $\frac{ds}{dx}\,(6^-) = 17$.

6.* Sformułuj ogólny układ równań na współczynniki funkcji sklejanej trzeciego stopnia, odpowiadający układowi (5.43), ale dla warunków dodatkowych (5.21d) (*not-a-knob conditions*).

7.* Sformułuj ogólny układ równań na współczynniki funkcji sklejanej trzeciego stopnia, podobnie jak wyprowadzono układ (5.43), ale dla warunków brzegowych (5.21a) (warunki brzegowe na pochodną pierwszego rzędu).

*Zadanie dodatkowe.

Rozdział 6

Równania nieliniowe i zera wielomianów

6.1. Wyznaczanie zer funkcji nieliniowej

W rozdziale niniejszym omawiane będą *metody iteracyjne* wyznaczania rozwiązania pojedynczego równania nieliniowego, $f(x) = 0$. Aby rozwiązać takie równanie, należy najpierw oszacować przedział, w którym znajduje się rozwiązanie (pierwiastek, zero) tego równania, tzw. *przedział izolacji pierwiastka*. Jeśli możemy uczestniczyć w tym procesie obliczeniowym (praca interaktywna z komputerem), to najprościej użyć jakiegokolwiek programu graficznego wykreślającego (w przybliżeniu) przebieg funkcji f, i stąd oszacować przedziały izolacji jej pierwiastków. Jeśli chcemy proces zalgorytmizować, to podstawą metody wyznaczania przedziału izolacji jest badanie iloczynu wartości funkcji na końcach przedziału – jeżeli funkcja f jest ciągła i ma na końcach przedziału $[a, b]$ przeciwne znaki, tzn. $f(a) \cdot f(b) < 0$, to w przedziale $[a, b]$ leży co najmniej jeden jej pierwiastek.

Jeśli wystartujemy z małego przedziału, w którym nie ma pierwiastka (warunkiem koniecznym niezawierania pierwiastka jest $f(a) \cdot f(b) > 0$), to rozsądnym sposobem postępowania jest rozszerzanie przedziału, np. podanym poniżej algorytmem (gdzie $\beta > 1$ oraz jako krańce przedziału początkowego przyjęliśmy $x_1 = a$, $x_2 = b$; składnia programu wg MATLAB-a):

```
for j=1:n
   if f(x1)*f(x2) < 0
     a=x1;
     b=x2;
   elseif abs(f(x1)) < abs(f(x2))
     x1=x1+β(x1-x2);
   else
     x2=x2+β(x2-x1);
   end
end
```

Mając oszacowany przedział izolacji pierwiastka, szukamy jego wartości. Służy do tego wiele *metod iteracyjnych*, tj. startujących z danego przybliżenia począt-

kowego x_0 pierwiastka α (którym może być np. jeden z krańców przedziału izolacji) i sukcesywnie poprawiających to przybliżenie, $x_n \underset{n\longrightarrow\infty}{\longrightarrow} \alpha$.

Ogólnie, *rząd metody iteracyjnej (*zwany też *wykładnikiem zbieżności)* jest to największa liczba $p \geq 1$ taka, że

$$\lim_{n\to\infty} \frac{|x_{n+1} - \alpha|}{|x_n - \alpha|^p} = k < \infty, \tag{6.1}$$

gdzie k nazywane jest współczynnikem lub ilorazem zbieżności. Jeśli $p = 1$, to mówimy, że metoda jest zbieżna *liniowo* (w tym przypadku dla zbieżności wymagane jest $k < 1$). Jeśli $p = 1$ i k zbiega do zera przy wzroście n, to mówimy o zbieżności *superliniowej*. Jeśli $p = 2$, to mówimy, że metoda jest zbieżna *kwadratowo*. Im większy jest rząd metody, tym metoda jest szybsza. Dla $k > 0$ wzór (6.1) można zapisać w następującej przybliżonej postaci:

$$|x_{n+1} - \alpha| \approx k \cdot |x_n - \alpha|^p , \tag{6.2}$$

przy czym przybliżenie jest tym lepsze, im punkt x_n jest bliższy α.

Metody iteracyjne dla problemów nieliniowych są, na ogół, zbieżne tylko *lokalnie*. *Kulą zbieżności* metody iteracyjnej nazywamy otoczenie rozwiązania (pierwiastka) α o takim promieniu δ, że dla każdego punktu początkowego x_0 spełniającego $\|x_0 - \alpha\| \leq \delta$ algorytm metody jest zbieżny do α. Kula zbieżności jest, w ogólności, podzbiorem *obszaru atrakcji* rozwiązania α – zbioru wszystkich punktów początkowych, z których metoda jest zbieżna do α.

6.1.1. Metoda bisekcji (połowienia)

Startujemy z początkowego przedziału $[a, b\,] = [a_0, b_0]$ izolacji pierwiastka. W *metodzie bisekcji*, w każdej iteracji (n-tej iteracji):

1. Bieżący przedział zawierający zero funkcji, $[a_n, b_n]$, jest dzielony na dwie połowy, punktem środkowym c_n,
$$c_n = \frac{a_n + b_n}{2},$$
i obliczana jest wartość funkcji w nowym punkcie $f(c_n)$.
2. Obliczane są iloczyny $f(a_n)\cdot f(c_n)$ i $f(c_n)\cdot f(b_n)$, nowy przedział zawierający pierwiastek wybierany jest jako ten z dwóch podprzedziałów, któremu odpowiada iloczyn ujemny. Końce tego przedziału oznaczane są przez a_{n+1}, b_{n+1}.

Procedura jest powtarzana tak długo, aż np. $f(c_n) \leq \delta$, gdzie δ to założona dokładność rozwiązania. Test ten może być nieprecyzyjny dla przypadku bardzo „płaskiej" funkcji (tzn. jeśli pochodna jest bardzo mała w otoczeniu zera funkcji), dlatego też polecane jest również sprawdzanie długości przedziału, $b_n - a_n$, żądając, aby też była ona dostatecznie mała.

Dokładność rozwiązania uzyskanego metodą bisekcji zależy jedynie od ilości wykonanych iteracji, a nie zależy od dokładności obliczania wartości funkcji $f(x)$ na krańcach kolejnych przedziałów izolacji pierwiastka. Niech $\varepsilon_n = b_n - a_n$ oznacza długość przedziału w n-tym kroku (iteracji), $\varepsilon_0 = [a, b]$, $n = 0, 1, 2, ...$ Wówczas

$$\varepsilon_{n+1} = \frac{1}{2}\varepsilon_n, \tag{6.3}$$

stąd metoda bisekcji jest zbieżna liniowo ($p = 1$), z ilorazem zbieżności $k = 0.5$.

6.1.2. Metoda regula falsi

Metoda *regula falsi* (fałszywa reguła), zwana też metodą *false position*, jest bardzo podobna do metody bisekcji – różnica polega na tym, że aktualny przedział $[a_n, b_n]$ izolacji pierwiastka α dzielony jest nie na dwa równe, ale na dwa najczęściej nierówne podprzedziały, prostą (sieczną) łączącą na płaszczyźnie (f, x) punkty $(f(a_n), a_n)$ i $(f(b_n), b_n)$, przecinającą oś rzędnych w punkcie oznaczonym jako c_n. Jest to zilustrowane na rysunku 6.1.

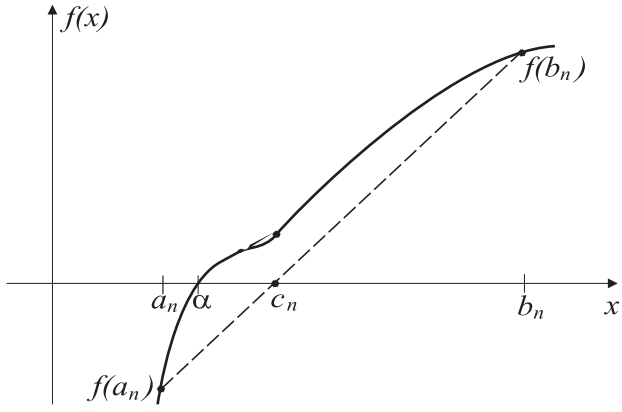
Z konstrukcji przedstawionej na rysunku mamy bezpośrednio

$$\frac{f(b_n) - f(a_n)}{b_n - a_n} = \frac{f(b_n) - 0}{b_n - c_n},$$

skąd

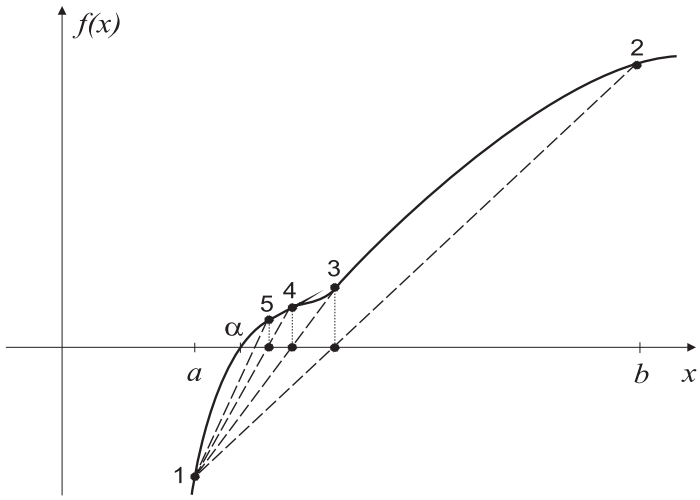
$$c_n = b_n - \frac{f(b_n)(b_n - a_n)}{f(b_n) - f(a_n)} = \frac{a_n f(b_n) - b_n f(a_n)}{f(b_n) - f(a_n)}. \tag{6.4}$$

Wybór następnego przedziału izolacji pierwiastka w metodzie regula falsi jest dokonywany tak samo jak w metodzie bisekcji, przez wyznaczanie iloczynów wartości funkcji na krańcach podprzedziałów i wybór tego podprzedziału, któremu



Rys. 6.1. Wyznaczanie nowego punktu podziału przez poprowadzenie siecznej

odpowiada iloczyn ujemny. Przykładowe działanie metody jest zilustrowane na rysunku 6.2, gdzie początkowy przedział $[a_0, b_0] = [a, b]$, a kolejne punkty oznaczono odpowiednimi cyframi (1,2,3, ...).



Rys. 6.2. Przykładowa ilustracja iteracji metody regula falsi

Ponieważ metoda regula falsi, podobnie jak metoda bisekcji, zawsze zachowuje przedział izolacji pierwiastka, więc jest zawsze zbieżna (liniowo, tzn. z $p = 1$, przy założeniu ciągłości funkcji), może być jednakże również wolno zbieżna. Na rysunku 6.2 pokazano przypadek, gdy jeden z końców izolacji pierwiastka pozostaje stały (punkt a) i metoda nie prowadzi do zmniejszania do zera przedziału izolacji pierwiastka – granicą jest przedział $[a, \alpha]$. Prowadzi to na ogół do wolnej zbieżności (czy nawet bardzo wolnej, w przypadku tzw. funkcji barierowych stosowanych w optymalizacji z ograniczeniami).

Prosta modyfikacja standardowego wzoru (6.4) metody regula falsi prowadząca do uniknięcia pokazanego zjawiska i przyspieszająca jej zbieżność polega na dwukrotnym zmniejszeniu wartości funkcji odpowiadającej nie zmieniającemu się końcowi przedziału izolacji, tzn.

$$c_n = \frac{a_n \frac{f(b_n)}{2} - b_n f(a_n)}{\frac{f(b_n)}{2} - f(a_n)} \tag{6.5a}$$

dla niezmieniającego się (np. co najmniej przez dwie iteracje) prawego końca oraz

$$c_n = \frac{a_n f(b_n) - b_n \frac{f(a_n)}{2}}{f(b_n) - \frac{f(a_n)}{2}} \tag{6.5b}$$

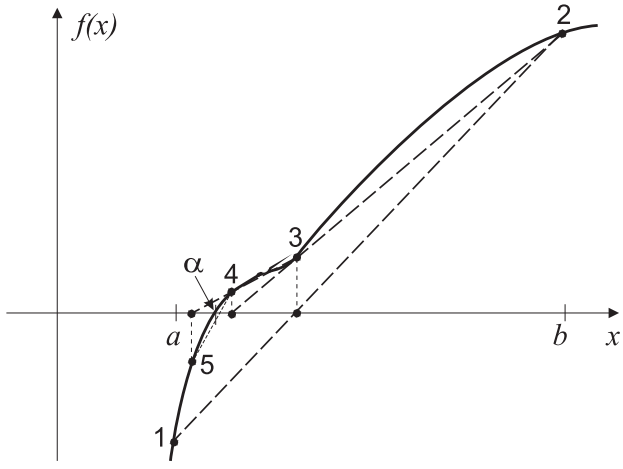
dla niezmieniającego się lewego końca (co odpowiada sytuacji przedstawionej na rysunku 6.2). Metoda łącząca standardowe (6.4) i, w razie potrzeby, zmodyfikowane (6.5a)- (6.5b) iteracje nazywana jest *metodą zmodyfikowaną regula falsi* (także *algorytmem Illinois*) i charakteryzuje się lepszą zbieżnością – tzw. zbieżnością superliniową (zob. wzór (6.1)). Zbieżność jest nadal globalna – kolejne przedziały dążą do zera.

6.1.3. Metoda siecznych

Metoda siecznych tym różni się od metody regula falsi, że sieczną prowadzimy zawsze *między dwoma ostatnio wyznaczonymi punktami* (nie dbając o zachowanie przedziału izolacji pierwiastka), jak to przedstawiono na rysunku 6.3. Jeśli te dwa ostatnie punkty oznaczymy przez x_{n-1} i x_n, to nowy punkt w metodzie siecznych również jest oczywiście zdefiniowany wzorem identycznym z (6.6), tzn.

$$x_{n+1} = x_n - \frac{f(x_n)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})} = \frac{x_{n-1}f(x_n) - x_n f(x_{n-1})}{f(x_n) - f(x_{n-1})}. \tag{6.6}$$

Rząd zbieżności metody siecznych $p = (1+\sqrt{5})/2 \approx 1.618$, tak więc metoda ta jest szybsza od metod bisekcji i regula falsi. Jednakże, jest ona zbieżna jedynie *lokalnie*, stąd w praktyce może być niezbieżna – jeśli początkowy przedział izolacji pierwiastka nie jest dostatecznie mały.



Rys. 6.3. Ilustracja metody siecznych

6.1.4. Metoda Newtona (stycznych)

Metoda Newtona, zwana też *metodą stycznych*, zakłada aproksymacje funkcji jej liniowym przybliżeniem wynikającym z uciętego rozwinięcia w szereg Taylora w aktualnym punkcie x_n (aktualnym przybliżeniu pierwiastka)

$$f(x) \approx f(x_n) + f'(x_n)(x - x_n). \tag{6.7}$$

Następny punkt, x_{n+1}, wynika z przyrównania do zera sformułowanej lokalnej liniowej aproksymacji funkcji $f(x)$, tzn. z równania

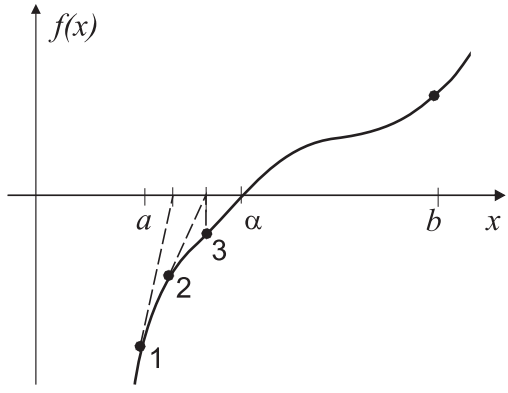
$$f(x_n) + f'(x_n)(x_{n+1} - x_n) = 0,$$

co prowadzi do zależności iteracyjnej

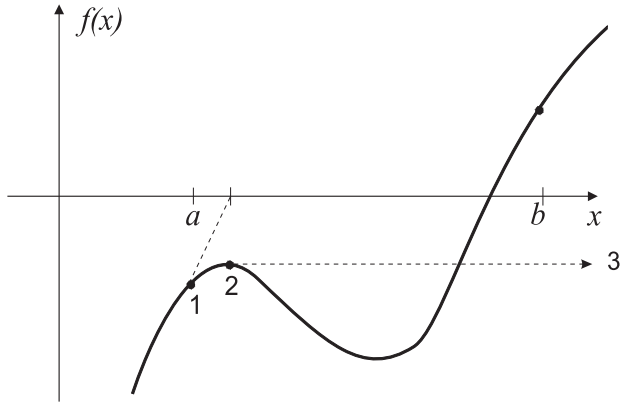
$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}. \tag{6.8}$$

Działanie metody Newtona zilustrowano na rys. 6.4. Metoda Newtona jest *zbieżna lokalnie* – jeśli zaczynamy ją stosować w punkcie zbytnio oddalonym od rozwiązania (poza obszarem atrakcji pierwiastka), to może być ona rozbieżna, jak to pokazano na rysunku 6.5. Natomiast metoda Newtona jest (lokalnie, asymptotycznie) bardzo szybka, jej zbieżność jest kwadratowa (tzn. z rzędem $p = 2$) – dość prosty dowód tej własności zamieszczamy dalej.

Metoda Newtona jest szczególnie efektywna w przypadku, gdy krzywa $f(x)$ jest bardzo stroma w otoczeniu danego pierwiastka, natomiast nie zaleca się jej stosowania w przypadku, gdy krzywa $f(x)$ jest w pobliżu pierwiastka (tzn. punktu przecięcia z osią $0x$) prawie pozioma, tzn. $f'(x)$ ma bardzo małą wartość.



Rys. 6.4. Ilustracja metody Newtona



Rys. 6.5. Metoda Newtona – przypadek iteracji rozbieżnej

Dowód* zbieżności kwadratowej metody Newtona.
Odejmując α od obu stron równości (6.8) oraz rozwijając $f\,(x)$ i $f'(x)$ w (skończone) szeregi Taylora w punkcie α, mamy (zakładając $f'(\alpha) \neq 0$)

$$\begin{aligned}
x_{n+1} - \alpha &= (x_n - \alpha) - \frac{f\,(x_n)}{f'(x_n)} \\
&= \frac{f'(x_n)\,(x_n - \alpha) - f\,(x_n)}{f'(x_n)} \\
&= \frac{[f'(\alpha) + f''(\xi_n)\,(x_n - \alpha)](x_n - \alpha)}{f'(x_n)} + \\
&\qquad - \frac{f'(\alpha)\,(x_n - \alpha) + \frac{1}{2} f''(\zeta_n)\,(x_n - \alpha)^2}{f'(x_n)} \\
&= \frac{(f''(\xi_n) - \frac{1}{2} f''(\zeta_n))(x_n - \alpha)^2}{f'(x_n)} \\
&\approx \frac{f''(\zeta_n)}{2f'(x_n)} (x_n - \alpha)^2,
\end{aligned}$$

gdzie $\xi_n \in [x_n,\ \alpha]$, $\zeta_n \in [x_n,\ \alpha]$. Stąd

$$\lim_{n\to\infty} \frac{|x_{n+1} - \alpha|}{|x_n - \alpha|^2} = \frac{|f''(\alpha)\,|}{2|f'(\alpha)\,|}.$$

*Materiał uzupełniający.

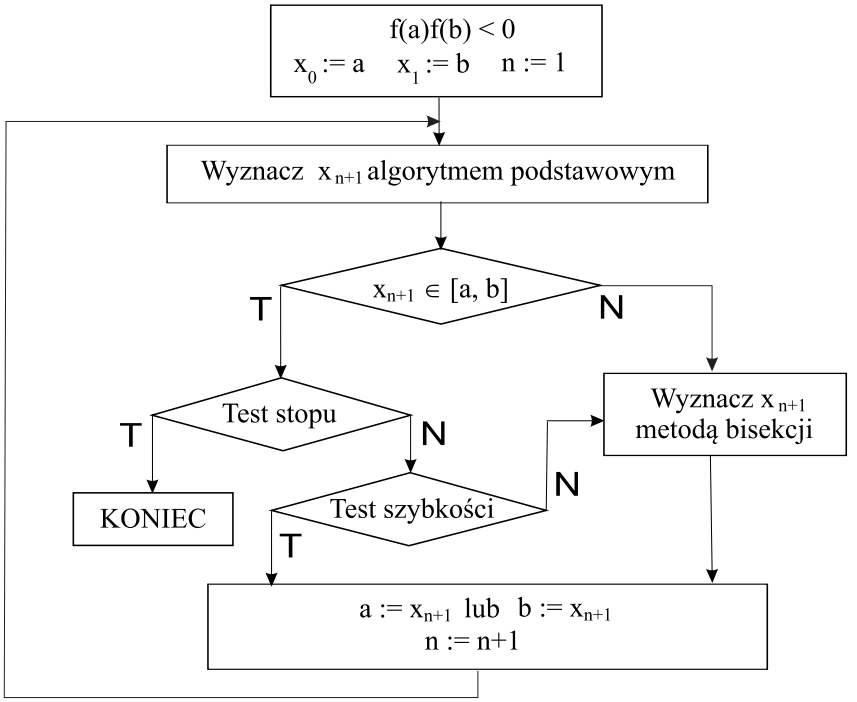
Pokazaliśmy zatem, że metoda Newtona jest zbieżna kwadratowo,

$$|x_{n+1} - \alpha| \approx \frac{|f''(\alpha)|}{2|f'(\alpha)|} \cdot |x_n - \alpha|^2 . \tag{6.9}$$

□

6.1.5. Przykładowa realizacja efektywnego algorytmu

Wymienione powyżej metody szybkie, metoda siecznych i, przede wszystkim, metoda Newtona, są metodami zbieżnymi lokalnie. Natomiast wolna metoda bisekcji jest zbieżna globalnie – tj. zawsze wtedy, gdy startujemy z przedziału zawierającego pierwiastek (i funkcja f jest ciągła), ponadto zawsze zbiega stabilnie, z tą samą szybkością. Stąd niezawodna i efektywna procedura znajdowania pierwiastka powinna być zabezpieczona przed możliwością wystąpienia niezbieżności, zalecane jest również wbudowanie mechanizmu przyspieszającego zbieżność, jeśli jest to możliwe. Przykładową praktyczną realizację takiej procedury znajdowania pojedynczego zera funkcji nieliniowej (na danym odcinku $[a, b]$) przedstawiono na rysunku 6.6.



Rys. 6.6. Schemat blokowy przykładowego efektywnego i zbieżnego algorytmu wyznaczania pojedynczego pierwiastka

Przedstawiona na rysunku 6.6 kombinacja metody podstawowej lokalnie szybkiej (Newtona, siecznych) z metodą bisekcji zapewnia globalną zbieżność algorytmu. Kluczowe dla efektywności procedury jest odpowiednie zaprojektowanie testu *szybkiej zbieżności*. Bardzo prostą propozycją jest następująca reguła

$$|f(x_{n+1})| < \beta |f(x_n)|, \quad \text{gdzie np. } \beta = \frac{1}{2}. \tag{6.10}$$

Przykładowym rozwiązaniem dla metody Newtona może być następujący test (uzasadnienie pozostawiamy czytelnikowi):

$$|f(x_{n+1})| < \frac{1}{2} \left| (b-a) \cdot f'(x_{n+1}) \right|. \tag{6.11}$$

Jeśli nie zależy nam na najszybszej metodzie, ale za to prostszej i pewnej, to alternatywą pozostaje zmodyfikowana metoda regula falsi – na ogół znacznie szybsza od metody bisekcji, a przy tym również globalnie zbieżna.

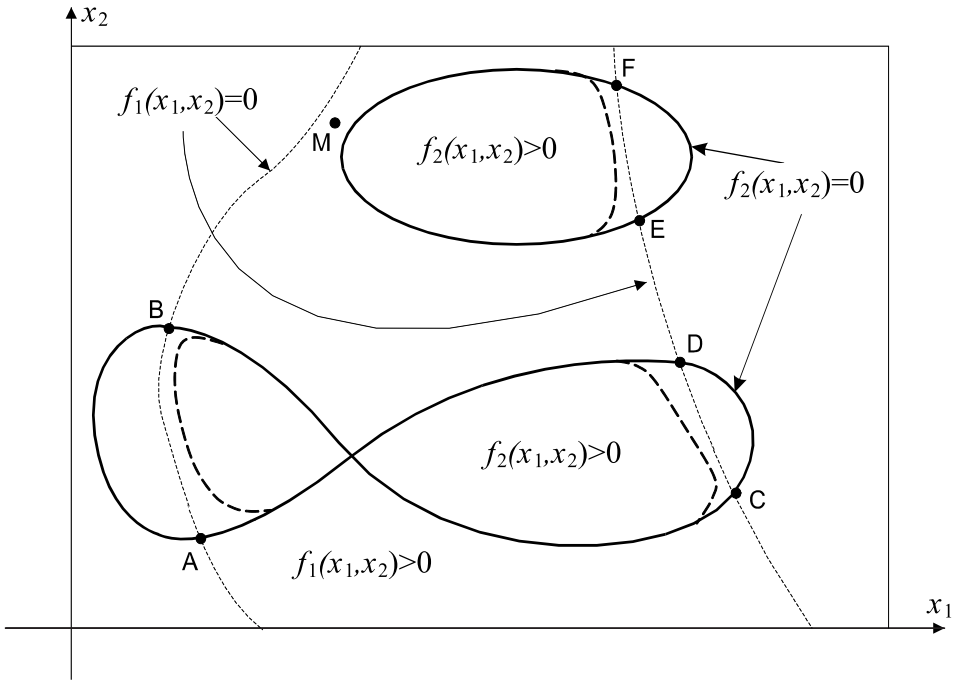
6.2. Rozwiązywanie układów równań nieliniowych

Rozważymy na wstępie następujący układ dwóch równań nieliniowych, gdzie $\mathbf{x} = [x_1\ x_2]^{\mathrm{T}} \in \mathbb{R}^2$,

$$\mathbf{f}(\mathbf{x}) \stackrel{\mathrm{df}}{=} \begin{bmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{bmatrix} = \mathbf{0}.$$

Przykładową ilustrację możliwych rozwiązań takiego układu równań zamieszczono na rysunku 6.7, przedstawiającym krzywe $f_1(\mathbf{x}) = 0$ (linia cienka przerywana) i $f_2(\mathbf{x}) = 0$ (linia ciągła gruba). Przypadek przedstawiony na rysunku ma 6 rozwiązań leżących w punktach wspólnych tych krzywych, oznaczonych literami od A do F. Zauważmy, że gdyby linie przedstawiające równanie $f_2(\mathbf{x}) = 0$ były zmodyfikowane tak jak na rysunku linią grubszą przerywaną (niewiele zmienione zachowanie się funkcji f_2!), to układ nie miałby w ogóle rozwiązań. Zwróćmy też uwagę na punkt M – w punkcie tym wartości obu funkcji są bliskie zeru, ale w dużym otoczeniu tego punktu nie ma jakiegokolwiek rozwiązania układu równań.

W przypadku układów równań nieliniowych, w celu znalezienia wszystkich rozwiązań zaleca się najpierw przeanalizować (np. graficznie, o ile wymiar problemu na to pozwala) cały zbiór X zmiennych $\mathbf{x}$, w którym poszukujemy rozwiązań i wyznaczyć podzbiory, w których znajdują się pojedyncze rozwiązania. Następnie stosujemy szybkie, ale jedynie lokalnie zbieżne metody, kolejno znajdujące pojedyncze rozwiązania, startujące z punktów położonych w podzbiorach (metody takie omówimy w dalszej części rozdziału). Prostszym podejściem do problemu jest wstępne, arbitralne podzielenie zbioru X na mniejsze podzbiory (np. siatką o określonej wielkości oczka) i wielokrotne zastosowanie metody lokalnej znajdującej



Rys. 6.7. Przykładowa ilustracja rozwiązań układu dwóch równań nieliniowych (przecięcia krzywych $f_1(\mathbf{x}) = 0$ i $f_2(\mathbf{x}) = 0$ na płaszczyźnie (x_1, x_2))

pojedyncze rozwiązanie, startującej kolejno z każdego podzbioru. Oczywiście, im gęściejszy podział, tym większa szansa znalezienia wszystkich rozwiązań – ale i większy nakład obliczeń. Stosuje się też podejście stochastyczne, polegające na startowaniu metody lokalnej z określonej liczby punktów początkowych, w sposób przypadkowy losowanych z obszaru X. Również i tym razem, im większa liczba punktów, tym większa szansa znalezienia wszystkich rozwiązań.

Zadanie rozwiązania układu równań nieliniowych jest zbliżone do zadania minimalizacji bez ograniczeń funkcji wielu zmiennych, i często trudniejsze. Sformułowanie zaś tego zadania jako zadania minimalizacji, np. minimalizacji funkcji celu

$$\min_{x_1 \cdots x_n} \sum_{i=1}^{n} f_i \left(x_1, ..., x_n\right)^2, \tag{6.12}$$

tylko pozornie upraszcza problem, gdyż funkcja (6.12) ma na ogół minima lokalne, w których jej wartość jest różna od zera (np. minimum odpowiadające punktowi M z rysunku 6.7).

6.2.1. Metoda Newtona

Odwzorowanie $\mathbf{f} : \mathbb{R}^n \longrightarrow \mathbb{R}^n$, $f(\mathbf{x}) = [f_1(\mathbf{x}) \cdots f_n(\mathbf{x})]^{\mathrm{T}}$ rozwijamy w szereg Taylora, uwzględniając dalej tylko przybliżenie liniowe:

$$\mathbf{f}(\mathbf{x}_k + \Delta\mathbf{x}_k) \approx \mathbf{f}(\mathbf{x}_k) + \mathbf{f}'(\mathbf{x}_k)\,\Delta\mathbf{x}_k,$$
$$\mathbf{f}'(\mathbf{x}_k) = \left[\frac{\partial f_i}{\partial x_j}(\mathbf{x}_k)\right]_{n\times n},$$

gdzie $\mathbf{f}'(\mathbf{x}_k)$ to tzw. macierz jakobianowa, tzn. macierz pochodnych cząstkowych funkcji $f_i(\mathbf{x})$, $i = 1, ..., n$, w punkcie $\mathbf{x}_k$. Postępując tak samo jak w przypadku skalarnym (zob. rozdz. 6.1.4), wyznaczamy $\mathbf{x}_{k+1}$ tak, by było rozwiązaniem przybliżenia liniowego

$$\mathbf{f}(\mathbf{x}_k) + \mathbf{f}'(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k) = \mathbf{0}, \tag{6.13}$$

skąd

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left[\mathbf{f}'(\mathbf{x}_k)\right]^{-1}\mathbf{f}(\mathbf{x}_k). \tag{6.14}$$

Praktycznie w obliczeniach nie odwracamy macierzy, lecz rozwiązujemy układ równań liniowych względem $\Delta\mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$:

$$\mathbf{f}'(\mathbf{x}_k)\,\Delta\mathbf{x}_k = -\mathbf{f}(\mathbf{x}_k),$$
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x_k}.$$

Metoda Newtona jest zbieżna lokalnie i jest rzędu drugiego, prawdziwa jest asymptotyczna zależność

$$\|\mathbf{x}_{k+1} - \hat{\mathbf{x}}\| \approx k\,\|\mathbf{x}_k - \hat{\mathbf{x}}\|^2, \tag{6.15}$$

gdzie $\hat{\mathbf{x}}$ jest poszukiwanym rozwiązaniem, a k jest pewna stałą (por. wzór (6.9)). Metoda Newtona nie korzysta z poprzednich punktów (jest „bez pamięci"), ale wymaga znajomości postaci analitycznej macierzy pochodnych cząstkowych. Istnieje wiele uproszczonych wariantów metody Newtona, jak również rozszerzeń dla zapewnienia globalnej zbieżności.

Dyskretna metoda Newtona

Każdy z elementów $\frac{\partial f_i}{\partial x_j}(\mathbf{x}_k)$ macierzy $\mathbf{f}'(\mathbf{x}_k)$ aproksymujemy przez iloraz różnicowy (zob. rozdz. 8.1)

$$\Delta_j f_i(\mathbf{x}_k) = \frac{f_i(\mathbf{x}_k + h_j\mathbf{e}_j) - f_i(\mathbf{x}_k)}{h_j}, \tag{6.16}$$

gdzie $\mathbf{e}_j$ to j-ty wersor układu współrzędnych, a h_j to długość kroku próbnego. Wymaga to policzenia dodatkowo n razy wartości funkcji lewych stron układu równań $\mathbf{f}(\mathbf{x}) = \mathbf{0}$, zamiast policzenia wartości pochodnych.

6.2.2. Metoda Broydena*

Ideą metody jest zastąpienie macierzy pochodnych $\mathbf{f}'(\mathbf{x}_k)$ pewną macierzą $\mathbf{B}_k$, inaczej i prościej obliczaną niż składająca się z elementów (6.16) aproksymacja macierzy jakobianowej w dyskretnej metodzie Newtona.

Dla dowolnej macierzy kwadratowej $\mathbf{A} \in \mathbb{R}^{n\times n}$, wektora $\mathbf{b} \in \mathbb{R}^n$ i dowolnych punktów $\mathbf{x}$, $\mathbf{x}' \in \mathbb{R}^n$ zdefiniujmy

$$\begin{aligned}\mathbf{p} &= \mathbf{x}' - \mathbf{x},\\ \mathbf{q} &= \mathbf{A}\mathbf{x}' - \mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{p}.\end{aligned}$$

Twierdzenie 6.1. *Dla dowolnej macierzy kwadratowej $\mathbf{B} \in \mathbb{R}^{n\times n}$, macierz $\mathbf{B}'$ zdefinowana przez*

$$\mathbf{B}' = \mathbf{B} + \frac{(\mathbf{q} - \mathbf{B}\mathbf{p})\,\mathbf{p}^{\mathrm{T}}}{\mathbf{p}^{\mathrm{T}}\mathbf{p}} \tag{6.17}$$

spełnia zależności

$$\mathbf{B}'\mathbf{p} = \mathbf{q}, \tag{6.18}$$

$$\left\|\mathbf{B}' - \mathbf{A}\right\|_2 \leq \left\|\mathbf{B} - \mathbf{A}\right\|_2. \tag{6.19}$$

Traktując $\mathbf{A}$ jak liniowe przybliżenie jakobianu $\mathbf{f}\,'(\mathbf{x})$ układu równań nieliniowych, twierdzenie podaje sposób tworzenia, iteracyjnie, aproksymacji $\mathbf{B}_k$ macierzy $\mathbf{A}$. Stąd

algorytm Broydena:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{d}_k,\ \mathbf{d}_k = \mathbf{B}_k^{-1}\mathbf{f}\left(\mathbf{x}_k\right), \tag{6.20}$$

$$\begin{aligned}\mathbf{p}_k &= \mathbf{x}_{k+1} - \mathbf{x}_k,\\ \mathbf{q}_k &= \mathbf{f}\left(\mathbf{x}_{k+1}\right) - \mathbf{f}\left(\mathbf{x}_k\right),\\ \mathbf{B}_{k+1} &= \mathbf{B}_k + \frac{(\mathbf{q}_k - \mathbf{B}_k\mathbf{p}_k)\,\mathbf{p}_k^{\mathrm{T}}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k}.\end{aligned} \tag{6.21}$$

Rezultaty lepsze od kroku o długości jednostkowej (tzw. kroku newtonowskiego: $\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{d}_k$) daje na ogół algorytm z przybliżoną optymalizacją długości kroku,

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \lambda_k\mathbf{d}_k, \tag{6.22}$$

gdzie, np.

$$\lambda_k = 2^{-j},\ \ j = \min\left\{i \geq 0 : \left\|\mathbf{f}\left(\mathbf{x}_k - 2^{-i}\mathbf{d}_k\right)\right\| < \left\|\mathbf{f}\left(\mathbf{x}_k\right)\right\|\right\}. \tag{6.23}$$

*Materiał uzupełniający.

6.2.3. Metoda iteracji prostej

W wielu zastosowaniach spotykamy się z równaniem lub układem równań postaci

$$\mathbf{x} - \mathbf{f}(\mathbf{x}) = \mathbf{0}, \tag{6.24}$$

lub też łatwo problem możemy do tej postaci sprowadzić. Możemy wtedy próbować rozwiązać układ równań *metodą iteracji prostej:*

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k). \tag{6.25}$$

Ciąg przybliżeń jest lokalnie zbieżny do rozwiązania $\hat{\mathbf{x}}$, takiego że $\hat{\mathbf{x}} = \mathbf{f}(\hat{\mathbf{x}})$ (tzn. $\hat{\mathbf{x}}$ jest punktem stałym odwzorowania $\mathbf{f}$), wtedy i tylko wtedy, gdy

$$\mathrm{sr}\left[\mathbf{f}'(\hat{\mathbf{x}})\right] < 1, \tag{6.26}$$

gdzie „sr" oznacza promień spektralny macierzy $\mathbf{f}'(\hat{\mathbf{x}})$ (por. twierdzenie 2.2). Szybkość zbieżności można poprawić stosując metodę z relaksacją:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + s\left(\mathbf{f}(\mathbf{x}_k) - \mathbf{x}_k\right), \qquad s \in (0, 1]. \tag{6.27}$$

gdzie s jest współczynnikiem relaksacji.

Algorytmy typu iteracji prostej są wolniej zbieżne niż metoda Newtona lub Broydena, lecz są o wiele prostsze. Algorytmy Jacobiego i Gaussa-Seidela iteracyjnego rozwiązywania układu równań liniowych $\mathbf{A}\mathbf{x} = \mathbf{b}$ są w swej istocie metodami iteracji prostej (liniowymi), por. rozdz. 2.6. Inne przykładowe zastosowanie to iteracje korektora w algorytmach wielokrokowych typu predyktor-korektor rozwiązywania układów równań różniczkowych, zob. rozdz. 7.2.

Rozważmy ogólny proces iteracyjny opisany operatorem $\mathbf{G}$,

$$\mathbf{x}_{k+1} = \mathbf{G}(\mathbf{x}_k). \tag{6.28}$$

Proces ten jest lokalnie asymptotycznie zbieżny do $\hat{\mathbf{x}} = \mathbf{G}(\hat{\mathbf{x}})$ wtedy i tylko wtedy, gdy

$$\mathrm{sr}\left[\mathbf{G}'(\hat{\mathbf{x}})\right] < 1. \tag{6.29}$$

Stąd, algorytm z relaksacją jest zbieżny, jeśli

$$\mathrm{sr}\left[\mathbf{I} + s\,\mathbf{f}'(\hat{\mathbf{x}}) - s\mathbf{I}\right] = \mathrm{sr}\left[(1-s)\,\mathbf{I} + s\,\mathbf{f}'(\hat{\mathbf{x}})\right] < 1. \tag{6.30}$$

6.3. Wyznaczanie zer wielomianów

Rozważamy wielomian postaci:

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0. \tag{6.31}$$

Wielomian ten posiada dokładnie n pierwiastków (zer wielomianu). Przy tym:

– pierwiastki mogą być rzeczywiste lub zespolone (pary liczb wzajemnie sprzężonych),
– pierwiastki mogą być pojedyncze lub wielokrotne.

Do szukania pierwiastków rzeczywistych wielomianów *możemy korzystać ze wszystkich przedstawionych dotychczas algorytmów poszukiwania zer funkcji nieliniowej*. Istnieją też metody nieco bardziej złożone, opracowane specjalnie dla wielomianów, gdyż wykorzystują one specyficzne właściwości tych funkcji (jak wielokrotna różniczkowalność) i znajdują również pierwiastki zespolone. Istnieje wiele takich metod, zob. np. [1], [11]. Do polecanych należą metoda Müllera i metoda Laguerre'a.

6.3.1. Metoda Müllera

Metoda polega na aproksymacji wielomianu w otoczeniu rozwiązania funkcją kwadratową. Może być traktowana jako uogólnienie metody siecznych – zamiast interpolacji w dwóch punktach funkcją liniową (tzn. sieczną) wykonujemy interpolację w trzech punktach funkcją kwadratową, zob. np. [9]. Istnieje również efektywna realizacja oparta na wykorzystaniu informacji o wielomianie jedynie w jednym punkcie, tzn. wykorzystująca do wyznaczenia funkcji kwadratowej wartości wielomianu i jego pierwszej i drugiej pochodnej w aktualnym punkcie.

Poniżej przedstawimy dwie wspomniane wyżej realizacje metody, które oznaczymy jako **MM1, MM2.**

MM1. Rozważmy trzy punkty x_0, x_1, x_2 wraz z wartościami wielomianu w tych punktach $f(x_0),\ f(x_1)$ i $f(x_2)$. Skonstruujemy funkcję kwadratową przechodzącą przez te punkty, a następnie wyznaczymy pierwiastki tej funkcji i potraktujemy jeden z nich jako kolejne, poprawione przybliżenie rozwiązania (pierwiastka wielomianu).

Przyjmijmy, bez utraty ogólności, że x_2 jest aktualną aproksymacją rozwiązania (pierwiastka wielomianu). Wprowadzimy zmienną przyrostową

$$z = x - x_2$$

i wykorzystamy różnice

$$z_0 = x_0 - x_2,$$
$$z_1 = x_1 - x_2,$$

oznaczając poszukiwaną parabolę przez

$$y(z) = az^2 + bz + c. \tag{6.32}$$

Biorąc pod uwagę trzy dane punkty, mamy

$$\begin{aligned} az_0^2 + bz_0 + c &= y(z_0) = f(x_0), \\ az_1^2 + bz_1 + c &= y(z_1) = f(x_1), \\ c &= y(0) = f(x_2). \end{aligned}$$

Stąd, do wyznaczenia a i b należy rozwiązać układ dwóch równań liniowych

$$\begin{aligned} az_0^2 + bz_0 &= f(x_0) - f(x_2), \\ az_1^2 + bz_1 &= f(x_1) - f(x_2). \end{aligned} \tag{6.33}$$

Ponieważ interesuje nas pierwiastek paraboli (6.32) o najmniejszym module (tzn. położony jak najbliżej x_2), więc do numerycznego wyznaczenia tego pierwiastka najlepiej wykorzystać wzory (1.13), (1.14):

$$z_+ = \frac{-2c}{b + \sqrt{b^2 - 4ac}}, \tag{6.34}$$

$$z_- = \frac{-2c}{b - \sqrt{b^2 - 4ac}}. \tag{6.35}$$

Do kolejnego przybliżenia rozwiązania bierzemy pierwiastek położony jak najbliżej x_2, tj. o mniejszym module,

$$x_3 = x_2 + z_{\min},$$

gdzie

$$\begin{aligned} z_{\min} &= z_+, \quad \text{gdy } |b + \sqrt{b^2 - 4ac}| \geq |b - \sqrt{b^2 - 4ac}|, \\ z_{\min} &= z_-, \quad \text{w przeciwnym przypadku.} \end{aligned}$$

Przed przejściem do następnej iteracji odrzucamy spośród x_0, x_1, x_2 punkt położony najdalej od ostatnio wyznaczonego przybliżenia rozwiązania, tj. punktu x_3.

Zwróćmy uwagę, że algorytm powinien działać prawidłowo również w przypadku, gdy $b^2 - 4ac < 0$, gdyż jest to sytuacja typowa, prowadząca do wyznaczenia zera zespolonego. Z tego powodu algorytm metody Müllera należy implementować w arytmetyce liczb zespolonych. Sformułowane powyżej wzory bezpośrednio to umożliwiają.

MM2. Wersja metody wykorzystująca informację nie o wartości wielomianu w kolejnych trzech punktach, ale *o wartości wielomianu i jego pochodnych, pierwszego i drugiego rzędu* w aktualnym punkcie (przybliżeniu zera) x_k. Wersja nieco efektywniejsza obliczeniowo z tego powodu, że obliczenie wartości wielomianu w $k+1$ punktach jest nieco kosztowniejsze niż obliczenie wartości wielomianu i jego k kolejnych pochodnych w jednym punkcie.

Z definicji paraboli $y(z)$ (6.32), dla $z \stackrel{\text{df}}{=} x - x_k$, w punkcie $z = 0$ bezpośrednio wynika:

$$\begin{aligned} y(0) &= c = f(x_k), \\ y'(0) &= b = f'(x_k)\,, \\ y''(0) &= 2a = f''(x_k)\,, \end{aligned}$$

co prowadzi do wzoru na pierwiastki

$$z_{+,-} = \frac{-2f(x_k)}{f'(x_k) \pm \sqrt{(f'(x_k))^2 - 2f(x_k)f''(x_k)}}. \tag{6.36}$$

Do przybliżenia zera α bierzemy pierwiastek paraboli o mniejszym module:

$$x_{k+1} = x_k + z_{\min}, \tag{6.37}$$

gdzie $z_{\min}$ jest wybierany spośród $\{z_+, z_-\}$ w taki sam sposób jak w wersji MM1.

Podobnie jak MM1, wersję MM2 metody należy implementować w arytmetyce liczb zespolonych.

Metoda Müllera jest zbieżna lokalnie, z rzędem zbieżności 1.84. Jest więc (lokalnie) bardziej efektywna niż metoda siecznych, jest niewiele wolniejsza od metody Newtona. Z konstrukcji metody wynika, że może ona być stosowana do poszukiwania zer rzeczywistych i zespolonych nie tylko wielomianów, ale i innych funkcji nieliniowych (analitycznych).

6.3.2. Metoda Laguerre'a

Metodę tę definiuje wzór

$$x_{k+1} = x_k - \frac{nf(x_k)}{f'(x_k) \pm \sqrt{(n-1)[(n-1)(f'(x_k))^2 - nf(x_k)f''(x_k)]}}, \tag{6.38}$$

gdzie n jest stopniem wielomianu, a znak w mianowniku wybieramy tak, żeby miał on większy moduł (identycznie jak w metodzie Müllera), zob. [1], zob. też [9] dla nieco innego sformułowania algorytmu, wraz z jego intuicyjnym wyprowadzeniem.

Zwróćmy uwagę na podobieństwo wzorów (6.36)-(6.37) i (6.38). Wzór Laguerre'a (6.38) jest nieco bardziej złożony, uwzględnia też stopień wielomianu (można go traktować jako uogólnienie wzorów (6.36) i (6.37)), stąd metoda Laguerre'a jest w ogólności nieco szybsza. W przypadku wielomianu o zerach rzeczywistych metoda Laguerre'a jest zbieżna z każdego rzeczywistego punktu startowego, jest więc zbieżna globalnie. Mimo braku analizy zbieżności dla przypadku zer zespolonych, praktyka numeryczna wskazuje na dobre własności metody również w tym przypadku, choć mogą wystąpić przypadki niezbieżności. Metoda uważana jest za jedną z najlepszych dla znajdowania zer wielomianów.

6.3.3. Deflacja czynnikiem liniowym

Po znalezieniu pojedynczego pierwiastka α należy, przed wyznaczaniem następnego, uprościć wielomian dzieląc go przez czynnik $(x-\alpha)$ zawierający znaleziony pierwiastek. Jest to tzw. *deflacja czynnikiem liniowym*. Uzyskany wielomian niższego rzędu jest już prostszy i nie wyznaczymy ponownie tego samego pierwiastka (chyba że jest wielokrotny). Z numerycznego punktu widzenia istotne jest, z jaką dokładnością dokonujemy deflacji i wyznaczamy kolejne zero wielomianu.

Oznaczmy przez $Q(x)$ wielomian uzyskany po podzieleniu $f(x)$ przez $(x-\alpha)$. Mamy

$$f(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_1 x + a_0 = (x-\alpha)\cdot Q(x),$$
$$Q(x) = q_n x^{n-1} + ... + q_2 x + q_1.$$

Algorytmy dzielenia wielomianu $f(x)$ przez jednomian $(x-\alpha)$:

Schemat Hornera (prosty):

$$\begin{aligned} q_{n+1} &\stackrel{\text{df}}{=} 0, \\ q_i &= a_i + q_{i+1}\alpha, \qquad i = n, n-1, ..., 0. \end{aligned} \tag{6.39}$$

Odwrotny schemat Hornera (zakładamy $\alpha \neq 0$):

$$\begin{aligned} q_0 &\stackrel{\text{df}}{=} 0, \\ q_{i+1} &= \frac{q_i - a_i}{\alpha}, \qquad i = 0, 1, 2, ..., n-1. \end{aligned} \tag{6.40}$$

Sklejany schemat Hornera. Schemat prosty Hornera wyznacza współczynniki, poczynając od q_n, schemat odwrotny w odwrotnej kolejności, poczynając od q_1 – dzięki kumulacji błędów numerycznych, każdy kolejny współczynnik jest wyznaczany mniej dokładnie, w obu schematach. Stąd, szczególnie dla wielomianów wyższego rzędu, dokładniejszy jest tzw. *sklejany schemat Hornera*:

- $q_n, q_{n-1}, ..., q_{k+1}$ wyznaczamy zgodnie z algorytmem Hornera,
- $q_1, q_2, ..., q_k$ wyznaczamy zgodnie z odwrotnym algorytmem Hornera,

zaś indeks k oznaczający miejsce sklejenia można przyjąć jako odpowiadający środkowemu wyrazowi wielomianu. Dokładniejszy jest następujący przepis [1]: jeśli

b_i – współczynniki q_i wyznaczone algorytmem Hornera,

c_i – współczynniki q_i wyznaczone odwrotnym algorytmem Hornera,

to indeks sklejenia k definiowany jest przez zależność

$$\frac{|c_k - b_k|^r}{|a_k|^r + |c_k|^r} = \min_{|a_j|^r + |c_j|^r > 0} \frac{|c_j - b_j|^r}{|a_j|^r + |c_j|^r}, \tag{6.41}$$

gdzie $r = 1$ dla zer rzeczywistych, $r = 2$ dla zer zespolonych.

Trzeba podkreślić, że pierwiastki wielomianów powinny być wyznaczane od tych z najmniejszymi modułami do tych z największymi, co również wpływa na zmniejszenie błędów numerycznych.

W przypadku wielomianów o współczynnikach rzeczywistych i wyznaczania zer pojedynczych oraz stosowania deflacji czynnikiem liniowym, do wyznaczenia pierwiastków zespolonych trzeba stosować arytmetykę zespoloną. Można tego uniknąć, wyznaczając pierwiastki parami i stosując deflację czynnikiem kwadratowym – realizuje to algorytm Bairstowa, zob. rozdz. 6.3.5.

6.3.4. Poprawianie zer (*root polishing*)

Postępowanie polega na poprawieniu (tzn. zwiększeniu dokładności wyznaczenia) każdego z pierwiastków wyznaczonych na podstawie wielomianów niższego rzędu (po deflacjach). Startując z takiego pierwiastka, stosujemy dowolną standardową metodę (np. Newtona, Laguerre'a), ale pracujemy z *oryginalnym (pełnego rzędu n) wielomianem*. Dla par zespolonych, jeśli chcemy uniknąć arytmetyki zespolonej, możemy stosować metodę Bairstowa (zob. rozdz. 6.3.5), oczywiście też na oryginalnym wielomianie. Najlepiej poprawiać kolejne pierwiastki w miarę ich wyznaczania.

6.3.5. Metoda Bairstowa*

Istotą tej metody jest szukanie jednocześnie dwóch pierwiastków wielomianu i deflacja trójmianem kwadratowym.

Oznaczmy

$$m(x) = m(x; p, r) = x^2 - px - r, \tag{6.42}$$

i niech $Q(x; p, r)$ – iloraz dzielenia wielomianu $f(x)$ przez $m(x)$. Mamy

$$f(x) = \left(x^2 - px - r\right) Q(x; p, r) + q_1(p, r)\, x + q_0(p, r). \tag{6.43}$$

Istotą algorytmu jest wyznaczenie współczynników p i r zerujących resztę z dzielenia, tzn. spełnienie układu równań

$$q_0(p, r) = 0, \tag{6.44}$$

$$q_1(p, r) = 0. \tag{6.45}$$

Powyższy układ równań rozwiązujemy metodą Newtona:

$$\begin{bmatrix} p_{k+1} \\ r_{k+1} \end{bmatrix} = \begin{bmatrix} p_k \\ r_k \end{bmatrix} - \begin{bmatrix} \frac{\partial q_0}{\partial p}(p_k, r_k) & \frac{\partial q_0}{\partial r}(p_k, r_k) \\ \frac{\partial q_1}{\partial p}(p_k, r_k) & \frac{\partial q_1}{\partial r}(p_k, r_k) \end{bmatrix}^{-1} \begin{bmatrix} q_0(p_k, r_k) \\ q_1(p_k, r_k) \end{bmatrix}. \tag{6.46}$$

*Materiał uzupełniający.

Ponieważ wielomian $f(x)$ nie zależy od r, to po zróżniczkowaniu (6.43) mamy

$$\frac{\partial}{\partial r}f(x)\equiv 0=\left(x^2-px-r\right)\frac{\partial Q}{\partial r}(x;p,r)-Q(x;p,r)+\frac{\partial q_1}{\partial r}(p,r)\,x+\frac{\partial q_0}{\partial r}(p,r), \tag{6.47}$$

skąd

$$Q(x;p,r)=\left(x^2-px-r\right)\frac{\partial Q}{\partial r}(x;p,r)+\frac{\partial q_1}{\partial r}(p,r)\,x+\frac{\partial q_0}{\partial r}(p,r). \tag{6.48}$$

Podobnie, ponieważ wielomian $f(x)$ nie zależy od p, więc

$$\frac{\partial}{\partial p}f(x)\equiv 0=\left(x^2-px-r\right)\frac{\partial Q}{\partial p}(x;p,r)-xQ(x;p,r)+\frac{\partial q_1}{\partial p}(p,r)\,x+\frac{\partial q_0}{\partial p}(p,r), \tag{6.49}$$

skąd

$$xQ(x;p,r)=\left(x^2-px-r\right)\frac{\partial Q}{\partial p}(x;p,r)+\frac{\partial q_1}{\partial p}(p,r)\,x+\frac{\partial q_0}{\partial p}(p,r). \tag{6.50}$$

Struktura iteracji (k-tej) algorytmu Bairstowa

Mając dane p_k, r_k:

1. Wyznaczamy $Q(x)=Q(x;p_k,r_k)$, $q_1(p_k,r_k)$ i $q_0(p_k,r_k)$ dzieląc $f(x)$ przez wielomian $m_k(x)$, $m_k(x)=x^2-p_kx-r_k$, np. zgodnie z formułami:

$$\begin{aligned} q_n &= a_n,\\ q_{n-1} &= pq_n+a_{n-1},\\ q_i &= pq_{i+1}+rq_{i+2}+a_i, \qquad i=n-2,n-3,...,0, \end{aligned} \tag{6.51}$$

   gdzie

$$\begin{aligned} f(x) &= a_nx^n+a_{n-1}x^{n-1}+...+a_1x+a_0=(x^2-p_kx-r_k)\cdot Q(x),\\ Q(x) &= q_nx^{n-2}+...+q_3x+q_2. \end{aligned}$$

2. Wyznaczamy $\frac{\partial q_1}{\partial r}(p_k,r_k)$ i $\frac{\partial q_0}{\partial r}(p_k,r_k)$, dzieląc $Q(x;p_k,r_k)$ przez $m_k(x)$, zgodnie z (6.48).
3. Wyznaczamy $\frac{\partial q_1}{\partial p}(p_k,r_k)$ i $\frac{\partial q_0}{\partial p}(p_k,r_k)$, dzieląc $xQ(x;p_k,r_k)$ przez $m_k(x)$, zgodnie z (6.50) (lub prościej, stosując (6.52) i (6.53), patrz uwaga poniżej).
4. Wyznaczamy p_{k+1}, r_{k+1} zgodnie z (6.46), rozwiązując układ dwóch równań liniowych

$$\begin{bmatrix} \frac{\partial q_0}{\partial p}(p_k,r_k) & \frac{\partial q_0}{\partial r}(p_k,r_k)\\ \frac{\partial q_1}{\partial p}(p_k,r_k) & \frac{\partial q_1}{\partial r}(p_k,r_k) \end{bmatrix}\begin{bmatrix} p_{k+1}-p_k\\ r_{k+1}-r_k \end{bmatrix}=-\begin{bmatrix} q_0(p_k,r_k)\\ q_1(p_k,r_k) \end{bmatrix}.$$

Iteracje powtarzamy aż do uzyskania $q_0(p_k,r_k)=0$ i $q_1(p_k,k_k)=0$.

Uwaga. Krok 3. może być wykonany prościej, unikając dzielenia wielomianów. Mianowicie, oznaczmy przez x_1 i x_2 zera wielomianu $m_k(x) = x^2 - p_k x - r_k$. Z (6.48) mamy

$$Q(x_i; p_k, r_k) = \frac{\partial q_1}{\partial r}(p_k, r_k)\, x_i + \frac{\partial q_0}{\partial r}(p_k, r_k), \quad i = 1, 2.$$

Z powyższej równości i z (6.49) wynika, że dla $i = 1, 2$

$$-x_i\left(\frac{\partial q_1}{\partial r}(p_k, r_k)\, x_i + \frac{\partial q_0}{\partial r}(p_k, r_k)\right) + \frac{\partial q_1}{\partial p}(p_k, r_k)\, x_i + \frac{\partial q_0}{\partial p}(p_k, r_k) = 0,$$

tzn.

$$-x_i^2\frac{\partial q_1}{\partial r}(p_k, r_k) + x_i\left(-\frac{\partial q_0}{\partial r}(p_k, r_k) + \frac{\partial q_1}{\partial p}(p_k, r_k)\right) + \frac{\partial q_0}{\partial p}(p_k, r_k) = 0.$$

Ponieważ z definicji $x_i^2 = p_k x_i + r_k, \;\; i = 1, 2$, więc również

$$x_i\left(-\frac{\partial q_0}{\partial r}(p_k, r_k) + \frac{\partial q_1}{\partial p}(p_k, r_k) - p_k\frac{\partial q_1}{\partial r}(p_k, r_k)\right) + \\ -r_k\frac{\partial q_1}{\partial r}(p_k, r_k) + \frac{\partial q_0}{\partial p}(p_k, r_k) = 0.$$

To oznacza, że (dla przynajmniej jednego $x_i \neq 0$)

$$-\frac{\partial q_0}{\partial r}(p_k, r_k) + \frac{\partial q_1}{\partial p}(p_k, r_k) - p_k\frac{\partial q_1}{\partial r}(p_k, r_k) = 0,$$
$$-r_k\frac{\partial q_1}{\partial r}(p_k, r_k) + \frac{\partial q_0}{\partial p}(p_k, r_k) = 0.$$

Stąd, w końcu dostajemy proste wzory:

$$\frac{\partial q_1}{\partial p}(p_k, r_k) = p_k\frac{\partial q_1}{\partial r}(p_k, r_k) + \frac{\partial q_0}{\partial r}(p_k, r_k), \tag{6.52}$$

$$\frac{\partial q_0}{\partial p}(p_k, r_k) = r_k\frac{\partial q_1}{\partial r}(p_k, r_k). \tag{6.53}$$

□

Zadania

1. Napisz programy (np. w środowisku MATLAB) metod bisekcji, zmodyfikowanej regula falsi, siecznych i Newtona znajdowania zera funkcji nieliniowej. Zastosuj te programy do dokładnego wyliczenia zer następujących funkcji (po uprzednim wstępnym określeniu przedziałów izolacji zer interaktywną metodą graficzną, kreśląc przybliżony wykres funkcji):

a) $f(x) = 2x^3 - 2.5x - 5$, na odcinku $[1, 2]$,
b) $f(x) = e^{-x} - \sin(\pi x/2)$, na odcinku $[0, 2.5]$.
Porównaj i skomentuj wyniki.

2. Napisz program (np. w środowisku MATLAB) metody Newtona rozwiązywania układu nieliniowych równań algebraicznych. Zastosuj ten program do znalezienia rozwiązania układu równań:

$$\begin{aligned} -0.2x_1 + 0.2x_2 - x_3 + 6 &= 0, \\ x_1^2 + x_2^2 + x_3^2 - 36 &= 0, \\ x_1^2 + x_2^2 - 0.65^2(x_3 - 2)^2 &= 0. \end{aligned}$$

3.* Napisz program (np. w środowisku MATLAB) metody Broydena rozwiązywania układu nieliniowych równań algebraicznych. Zastosuj ten program do znalezienia rozwiązania układu równań z poprzedniego zadania. Porównaj efektywność metod.

4. Napisz program (np. w środowisku MATLAB) metod Müllera (w obu wariantach) i Laguerre'a znajdowania pierwiastka wielomianu. Zastosuj te programy do znalezienia wszystkich pierwiastków następujących wielomianów (stosując deflację liniową):
a) $f(x) = x^4 - 5x^3 + 10x^2 - 10x + 4$,
b) $f(x) = x^4 + x^3 - 5x^2 + 2x + 2$.
Porównaj i skomentuj wyniki.

5.* Stosując metodę Bairstowa, znajdź parę pierwiastków zespolonych sprzężonych wielomianu $f(x) = x^5 - 10x^4 + 35x^3 - 50x^2 + 24x$.

*Zadanie dodatkowe.

Rozdział 7

Równania różniczkowe zwyczajne

Równania różniczkowe są powszechnie stosowane do modelowania matematycznego układów dynamicznych. Układy równań różniczkowych opisujące rzeczywiste układy dynamiczne są na ogół nieliniowe i z reguły nie są znane metody wyznaczenia ich rozwiązań analitycznych – jedynym sposobem jest wyznaczenie rozwiązań metodami numerycznymi.

W rozdziale niniejszym rozważane będzie zagadnienie numerycznego wyznaczania rozwiązania układu równań różniczkowych zwyczajnych pierwszego rzędu, przy podanej wartości rozwiązania w punkcie początkowym (tzw. zagadnienie początkowe, Cauchy'ego). Metody numeryczne rozwiązywania tego zagadnienia stanowią podstawę algorytmiczną *pakietów symulacji ciągłych układów dynamicznych.* Ponadto, są one elementem algorytmów wyznaczania rozwiązań również bardziej złożonych układów równań różniczkowych, jak zagadnienia dwugraniczne, równania różniczkowe cząstkowe, zob. np. [9].

Zmienną niezależną oznaczać będziemy przez $x \in \mathbb{R}^1$ – *najczęściej reprezentuje ona czas*, ale może być również zmienną przestrzenną, jeśli np. rozważany układ równań zwyczajnych powstał z układu równań cząstkowych przy ustalonej wartości zmiennej reprezentującej czas. Zmienne zależne (rozwiązania układu równań) oznaczać będziemy przez y_i, $i = 1, ..., m$. Szukać będziemy rozwiązania układu równań postaci

$$\frac{dy_i(x)}{dx} = f_i\left(x, y_1(x), ..., y_m(x)\right), \qquad i = 1, ..., m, \tag{7.1}$$

na odcinku $x \in [a, b]$, przy warunkach początkowych $y_i(a) = y_{ia},\ i = 1, ..., m.$

Stosować będziemy zapis skrócony, z pominięciem argumentu funkcji rozwiązania,

$$\frac{dy_i}{dx} = f_i\left(x, y_1, ..., y_m\right), \quad y_i(a) = y_{ia}, \quad i = 1, ..., m, \quad x \in [a, b]. \tag{7.2}$$

Oznaczając $\mathbf{y} = [y_1\ y_2 \cdots y_m]^{\mathrm{T}} \in \mathbb{R}^m$, $\mathbf{f} = [\ f_1\ f_2 \cdots f_m]^{\mathrm{T}}$, $f_i : \mathbb{R}^{1+m} \to \mathbb{R}$, $i = 1, ..., m$, układ równań (7.2) można przedstawić w zapisie wektorowym:

$$\mathbf{y}'(x) = \mathbf{f}\left(x, \mathbf{y}\right), \quad \mathbf{y}(a) = \mathbf{y}_a, \quad x \in [a, b]. \tag{7.3}$$

Twierdzenie 7.1. *Jeśli spełnione są warunki:*
1. *funkcja* $\mathbf{f}$ *jest ciągła na zbiorze*

$$\mathbf{D} = \{(x, \mathbf{y}) : \quad a \leq x \leq b, \quad \mathbf{y} \in \mathbb{R}^m\}, \tag{7.4}$$

2. *funkcja* $\mathbf{f}$ *spełnia warunek Lipschitza względem* $\mathbf{y}$*, tzn.*

$$\exists\, L > 0 \quad \forall\, x \in [a, b] \;\; \forall\, \mathbf{y}, \overline{\mathbf{y}} \in \mathbb{R}^m \quad \| \mathbf{f}(x, \mathbf{y}) - \mathbf{f}(x, \overline{\mathbf{y}}) \| \leq L \| \mathbf{y} - \overline{\mathbf{y}} \|, \tag{7.5}$$

to dla każdych warunków początkowych $\mathbf{y}_a$ *istnieje dokładnie jedna funkcja* $\mathbf{y}(x)$*, w sposób ciągły różniczkowalna i spełniająca*

$$\mathbf{y}'(x) = \mathbf{f}(x, \mathbf{y}), \quad \mathbf{y}(a) = \mathbf{y}_a, \quad x \in [a, b]. \tag{7.6}$$

Można ponadto pokazać, że zagadnienie spełniające założenia tego twierdzenia jest *dobrze postawione*, tzn. jego rozwiązanie zależy w sposób ciągły od zmian (odchyleń) warunków początkowych i zaburzeń parametrów funkcji $\mathbf{f}$.

Często stosowanymi grupami metod numerycznego rozwiązywania równań różniczkowych zwyczajnych są tzw. *metody jednokrokowe* i *metody wielokrokowe*. W niniejszym opracowaniu zajmiemy się algorytmami należącymi do tych grup metod.

Metody numeryczne znajdowania rozwiązania układu równań różniczkowych to *metody różnicowe*, tj. przybliżona wartość rozwiązania obliczana jest w kolejnych, dyskretnych punktach x_n,

$$a = x_0 \leq x_1 \leq \cdots \leq x_n < \cdots \; b,$$

gdzie kolejne kroki metody $h_n = x_{n+1} - x_n$ mogą być ustalone lub zmienne.

Definicja (zbieżności metody różnicowej). Mówimy, że metoda jest *zbieżna*, jeśli dla każdego układu równań różniczkowych mającego jednoznaczne rozwiązanie $\mathbf{y}(x)$, przy krokach h_n dążących do zera zachodzi

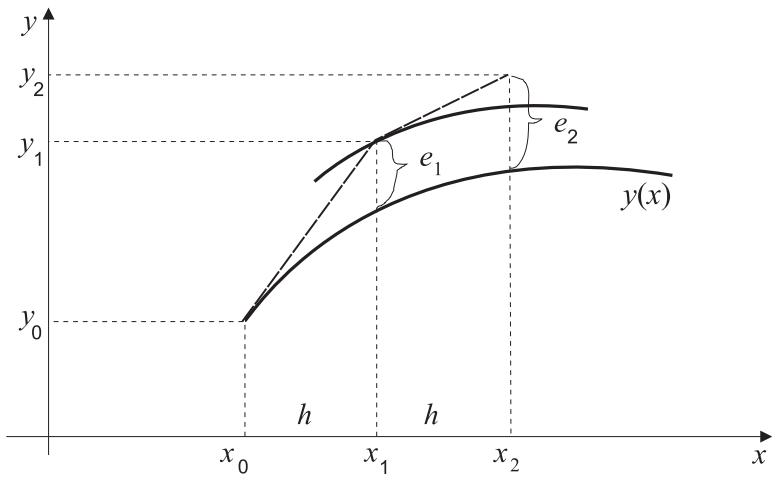
$$\lim_{h_n \to 0} \mathbf{y}(x_n; h_n) \to \mathbf{y}(x), \tag{7.7}$$

gdzie $\mathbf{y}(x_n; h_n)$ oznacza rozwiązanie przybliżone uzyskane tą metodą.

Najprostszą jest *metoda Eulera*, zdefiniowana następująco:

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h_n \mathbf{f}(x_n, \mathbf{y}_n), \qquad n = 0, 1, ..., \quad \mathbf{y}(x_0) = \mathbf{y}_a. \tag{7.8}$$

Graficzna interpretacja metody Eulera, w przypadku jednowymiarowym i dla stałego kroku $h_n = h$, jest przedstawiona na rysunku 7.1.



Rys. 7.1. Graficzna interpretacja metody Eulera

Przykład 7.1.* Wykażemy zbieżność metody Eulera danej wzorem (7.8), w przypadku jednowymiarowym (jedno równanie, $m = 1$, dla uproszczenia zapisu). Założymy, że $y(x) \in C^2$ na odcinku $[a, b]$ oraz krok jest stały, $h_n = h,\ n = 0, 1, ...$

Rozwijając $y(x)$ w punkcie x_n w (skończony) szereg Taylora, mamy

$$y\left(x_n + h\right) = y\left(x_n\right) + y'(x_n)h + y''(\xi_n)\frac{h^2}{2},\ \ \xi_n \in \left[x_n, x_n + h\right]. \tag{7.9}$$

Całkowity błąd metody po n krokach definiujemy następująco:

$$e_n \stackrel{\text{df}}{=} y\left(x_n\right) - y_n,$$

przy czym, dla przykładu, błąd e_2 jest zilustrowany na rysunku 7.1.

Odejmując stronami (7.8) od (7.9) oraz korzystając z założenia, że funkcja f spełnia warunek Lipschitza, otrzymujemy

$$\begin{aligned} e_{n+1} &= e_n + h\left[f\left(x_n, y\left(x_n\right)\right) - f\left(x_n, y_n\right)\right] + T_n, \\ |e_{n+1}| &\leq |e_n| + hL\,|e_n| + |T_n|, \end{aligned}$$

gdzie L jest stałą Lipschitza, zaś

$$T_n = y''(\xi_n)\frac{h^2}{2}.$$

*Materiał uzupełniający.

Niech

$$M_z = \frac{1}{2} \max_{x\in[a,b]} \left|y''(x)\right|,$$

wówczas, szacując błąd, uzyskujemy

$$\begin{aligned}
|e_{n+1}| &\le (1+Lh)\,|e_n| + h^2 M_z \\
&\le (1+hL)^2\,|e_{n-1}| + h^2 M_z\,((1+Lh)+1) \\
&\le \ \cdots \ \le h^2 M_z \sum_{j=0}^{n} (1+Lh)^j \\
&= h^2 M_z \frac{1-(1+Lh)^{n+1}}{1-(1+Lh)} \\
&= hM_z \frac{(1+Lh)^{n+1}-1}{L}.
\end{aligned}$$

Stąd, przy założeniu, że $L \neq 0$, po podstawieniu $n{+}1 = \frac{x_{n+1}-a}{h}$ i po wykorzystaniu tożsamości $1 + Lh < e^{Lh}$, otrzymujemy

$$\begin{aligned}
|e_{n+1}| &\le hM_z \frac{(1+Lh)^{\frac{x_{n+1}-a}{h}}-1}{L} \\
&\le hM_z \frac{e^{L(b-a)}-1}{L} = \xi_0 h,
\end{aligned}$$

co dowodzi zbieżności metody Eulera, gdyż $|e_{n+1}| \to 0$ dla $h \to 0$. □

Metoda Eulera jest najprostszą, ale i najmniej dokładną. Natomiast tzw. zmodyfikowana metoda Eulera (metoda punktu środkowego) czy metoda Heuna, podane poniżej, są jeszcze stosunkowo bardzo proste, ale generują znacznie mniejsze błędy.

Zmodyfikowana metoda Eulera (metoda punktu środkowego):

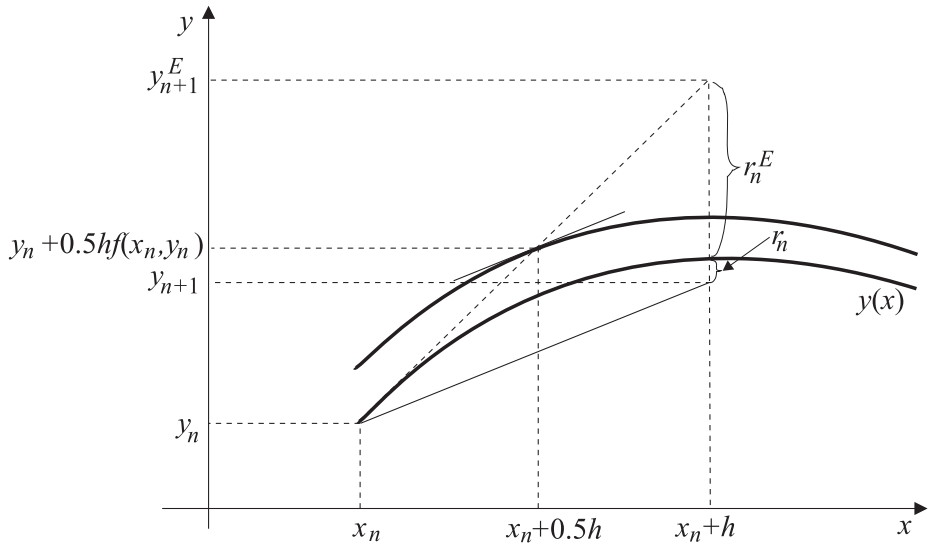
$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}(x_n + \frac{1}{2}h,\ \mathbf{y}_n + \frac{1}{2}h\mathbf{f}\,(x_n, \mathbf{y}_n)). \tag{7.10}$$

Interpretacja geometryczna jednego kroku zmodyfikowanej metody Eulera przedstawiona jest na rysunku 7.2, gdzie pokazano również, dla porównania, punkt y^E_{n+1} otrzymany w wyniku zastosowania jednego kroku zwykłej metody Eulera. Pokazane są również błędy obu metod powstałe w jednym kroku (tzw. błędy aproksymacji (7.15), zob. rozdz. 7.1), oznaczone przez r_n i r^E_n, odpowiednio.

Metoda Heuna:

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \frac{1}{2}h\left[\mathbf{f}\,(x_n, \mathbf{y}_n) + \mathbf{f}\,(x_n + h, \mathbf{y}_n + h\mathbf{f}\,(x_n, \mathbf{y}_n))\right]. \tag{7.11}$$

Czytelnikowi pozostawiamy zastanowienie się nad interpretacją graficzną metody Heuna i porównanie jej z metodami Eulera, zwykłą i zmodyfikowaną.



Rys. 7.2. Graficzna interpretacja jednego kroku zmodyfikowanej metody Eulera (indeksem „E" oznaczono rezultaty kroku zwykłej metody Eulera, pokazane dla porównania)

Przykład 7.2. Dla następującego układu dwóch równań różniczkowych (model epidemii):

$$\begin{aligned} \frac{dy_1}{dx} &= -a\, y_1 y_2, \\ \frac{dy_2}{dx} &= a\, y_1 y_2 - b\, y_2, \end{aligned} \tag{7.12}$$

sformułujemy równania jednego (n-tego) kroku metody zmodyfikowanej Eulera, kroku o długości h_n, startującego z punktu x_n (aktualna chwila), któremu odpowiada rozwiązanie (początkowe dla kroku n) $\mathbf{y}_n = [(y_1)_n\ (y_2)_n]^T$.

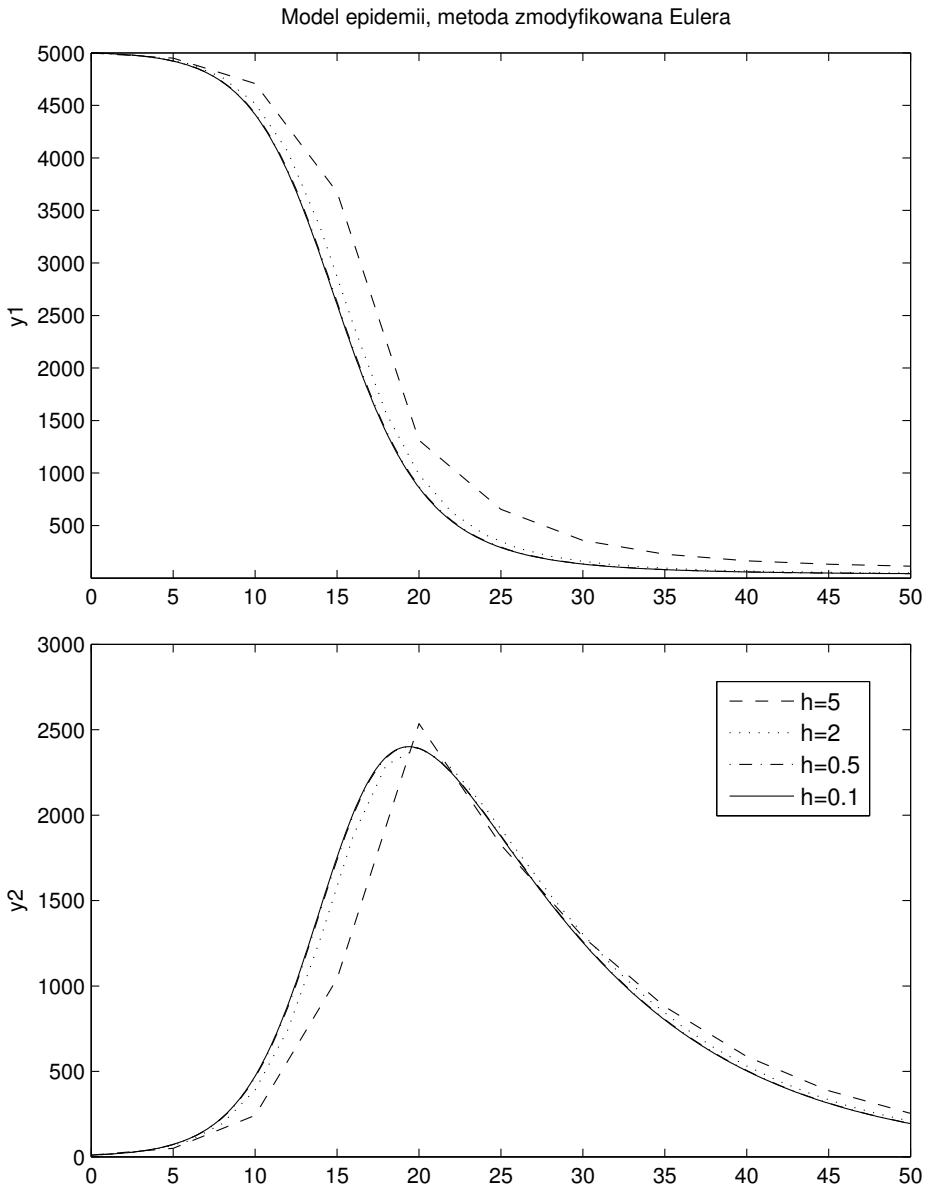
Efektywnie jest najpierw obliczyć rozwiązanie metodą Eulera z krokiem $\frac{1}{2}h_n$:

$$\begin{aligned} (y_1)_{n+\frac{1}{2}} &= (y_1)_n + \frac{1}{2}h_n[-a\,(y_1)_n(y_2)_n], \\ (y_2)_{n+\frac{1}{2}} &= (y_2)_n + \frac{1}{2}h_n[a\,(y_1)_n(y_2)_n - b\,(y_2)_n], \end{aligned}$$

a dopiero potem punkt następny (docelowy) obliczyć z równania metody zmodyfikowanej Eulera (punktu środkowego):

$$\begin{aligned} (y_1)_{n+1} &= (y_1)_n + h_n\left(-a\,(y_1)_{n+\frac{1}{2}}(y_2)_{n+\frac{1}{2}}\right), \\ (y_2)_{n+1} &= (y_2)_n + h_n\left(a\,(y_1)_{n+\frac{1}{2}}(y_2)_{n+\frac{1}{2}} - b\,(y_2)_{n+\frac{1}{2}}\right). \end{aligned}$$

Na rysunku 7.3 przedstawiono przebiegi symulacji modelu epidemii z parametrami $a = 0.0001,\ b = 0.1$ metodą zmodyfikowaną Eulera, z czterema różnymi stałymi krokami, startując z warunków początkowych $[(y_1)_0\ (y_2)_0]^{\mathrm{T}} = [5000\ 10]^{\mathrm{T}}$.



Rys. 7.3. Symulacja modelu epidemii zmodyfikowaną metodą Eulera, z różnymi ustalonymi krokami całkowania

Przebiegi są niedokładne dla większych kroków, potem stabilizują się – uzyskujemy przebiegi praktycznie nierozróżnialne dla kroków $h = 0.5$ i $h = 0.1$. Czytelnikowi pozostawiamy sprawdzenie, przy jakich krokach można uzyskać porównywalne przebiegi dla (zwykłej) metody Eulera. □

W dalszej części niniejszego rozdziału siódmego, dla uproszczenia zapisu, *nie będziemy używać liter pogrubionych („bold") dla wyróżniania zmiennych wektorowych* – ale wszystkie prezentowane zależności są słuszne zarówno dla pojedynczego równania ($m=1$), jak i dla układu równań różniczkowych ($m>1$).

7.1. Metody jednokrokowe

Ogólny wzór określający pojedynczy krok tzw. *metod jednokrokowych* (przy stałej długości kroku h) można zdefiniować następująco:

$$y_{n+1} = y_n + h\,\Phi_f(x_n, y_n; h), \tag{7.13}$$

gdzie

$$y(x_0) = y_0 = y_a, \;\; x_n = x_0 + nh, \;\; n = 0, 1, \ldots,$$

zaś $\Phi_f\,(x_n, y_n; h)$ to funkcja definiująca metodę. Ponieważ $h \neq 0$, więc wzór (7.13) można zapisać w postaci

$$\Phi_f\,(x_n, y_n; h) = \frac{y_{n+1} - y_n}{h}.$$

Zdefiniujmy

$$\Delta_f\,(x_n, y_n; h) = \frac{y\,(x_n + h) - y\,(x_n)}{h}.$$

Mówimy, że metoda jest *zbieżna*, gdy

$$h \to 0 \quad \Rightarrow \quad y\,(x_n; h) \to y\,(x)\,,$$

tzn. gdy

$$h \to 0 \quad \Rightarrow \quad y_n \to y(x_n),\; y_{n+1} \to y(x_n + h),$$

co implikuje

$$\Phi_f\,(x_n, y_n; h) \underset{h\to 0}{\longrightarrow} \Delta_f\,(x_n, y_n; h)\,.$$

Z kolei mamy

$$\Delta_f\,(x_n, y_n; h) \underset{h\to 0}{\longrightarrow} y'(x_n) = f\,(x_n, y_n)\,.$$

Stąd *warunkiem aproksymacji* (też: *warunkiem zgodności* metody z równaniem) nazywamy warunek

$$\Phi_f\,(x, y; 0) = f\,(x, y)\,. \tag{7.14}$$

Jeśli spełnione są założenia twierdzenia 7.1 oraz $\Phi_f\,(x, \cdot; h)$ spełnia warunek Lipschitza, to warunek powyższy jest *koniecznym i dostatecznym warunkiem zbieżności metody jednokrokowej*.

Błąd aproksymacji (lokalny *błąd metody*) $r_n(h)$ – błąd powstały w jednym kroku (tj. przy założeniu, że startujemy w tym kroku z dokładnej wartości rozwiązania $y_n = y(x_n)$) definiujemy następująco:

$$r_n(h) \stackrel{\text{df}}{=} y(x_n + h) - [y(x_n) + h\Phi_f(x_n, y_n; h)], \tag{7.15}$$

gdzie $y(x_n + h)$ jest rozwiązaniem, dla $x = x_n + h$, układu równań

$$\begin{aligned} y'(x) &= f(x, y(x)), \\ y(x_n) &= y_n, \qquad x \in [x_n, b], \end{aligned} \tag{7.16}$$

tzn. rozwiązaniem przechodzącym przez punkt $y(x_n) = y_n$.

Ilustrację graficzną błędu aproksymacji pokazują rysunki 7.1 i 7.2. Błąd e_1 na rysunku 7.1 to błąd aproksymacji $r_0(h)$ w pierwszym kroku, błąd po dwóch krokach to suma błędów aproksymacji w tych krokach, $e_2 = r_0(h) + r_1(h)$, itd. Na rysunku 7.2 pokazano natomiast bezpośrednio błędy aproksymacji metod zmodyfikowanej i zwykłej Eulera (przyjmując $y_n = y(x_n)$).

Zakładając odpowiednią gładkość funkcji $r_n(h)$ i rozwijając ją w szereg Taylora w punkcie $x = x_n$ (tzn. dla $h = 0$), otrzymujemy

$$r_n(h) = r_n(0) + r_n'(0)h + \frac{1}{2}r_n''(0)h^2 + \cdots$$

Mówimy, że *metoda jest rzędu p*, jeśli zachodzą równości

$$r_n(0) = 0, \;\; r_n'(0) = 0, \;\cdots\; r_n^{(p)}(0) = 0, \;\; r_n^{(p+1)}(0) \neq 0. \tag{7.17}$$

Wtedy

$$r_n(h) = \frac{r_n^{(p+1)}(0)}{(p+1)!}h^{p+1} + O\left(h^{p+2}\right), \tag{7.18}$$

gdzie pierwszy składnik to tzw. *część główna błędu aproksymacji*, zaś drugi jest funkcją rzędu nie niższego niż funkcja h^{p+2} (tzn. iloraz $\frac{O(h^{p+2})}{h^{p+2}}$ jest ograniczony w otoczeniu zera, lub inaczej: $O(h^{p+2})$ dąży do zera nie wolniej niż h^{p+2}) .

Przykład 7.3. Dla metody Eulera

$$r_n(h) = y(x_n + h) - y_n - hf(x_n, y_n).$$

Po rozwinięciu $y(x_n + h)$ w szereg Taylora w punkcie x_n, otrzymujemy

$$y(x_n + h) = y(x_n) + y'(x_n)h + \frac{1}{2}y''(x_n)h^2 + \ldots$$

Z założenia $y_n = y(x_n)$, $y'(x_n) = f(x_n, y_n)$, skąd

$$y(x_n + h) = \underbrace{y_n + hf(x_n, y_n)}_{y_{n+1}} + y''(x_n)h^2 + \ldots$$

Stąd bezpośrednio

$$r_n(h) = y''(x_n)h^2 + O\left(h^3\right),$$

zatem metoda Eulera jest rzędu pierwszego. □

7.1.1. Metody Rungego-Kutty (RK)

Metody te można zdefiniować następującym wzorem:

$$y_{n+1} = y_n + h \cdot \sum_{i=1}^{m} w_i k_i, \tag{7.19a}$$

gdzie

$$k_1 = f(x_n, y_n), \tag{7.19b}$$

$$k_i = f(x_n + c_i h, y_n + h \cdot \sum_{j=1}^{i-1} a_{ij} k_j), \quad i = 2, 3, ..., m, \tag{7.19c}$$

przy czym $$\sum_{j=1}^{i-1} a_{ij} = c_i, \quad i = 2, 3, ..., m.$$

Do wykonania jednego kroku metody należy obliczyć wartości prawych stron równań różniczkowych *dokładnie m razy*, dlatego będziemy mówić o m-etapowym kroku (iteracji) metody, czy krócej o metodzie m-etapowej. Parametry $w_i,\ a_{ij},\ c_i$ dobiera się tak, aby przy ustalonym m rząd metody był możliwie wysoki. Jeśli przez $p(m)$ oznaczymy maksymalny możliwy do uzyskania rząd metody, to udowodniono, że

$$\begin{aligned} p(m) &= m && \text{dla } m = 1, 2, 3, 4, \\ p(m) &= m - 1 && \text{dla } m = 5, 6, 7, \\ p(m) &\leq m - 2 && \text{dla } m. \geq 8 \end{aligned}$$

Największe znaczenie praktyczne mają metody z $m = 4$ i rzędu 4 – udany kompromis między dokładnością (rząd metody) a nakładem obliczeń na jedną iterację i związanym z tym wpływem błędów zaokrągleń.

Metoda RK 4. rzędu (RK4, „klasyczna"):

$$y_{n+1} = y_n + \frac{1}{6} h (k_1 + 2k_2 + 2k_3 + k_4), \tag{7.20a}$$

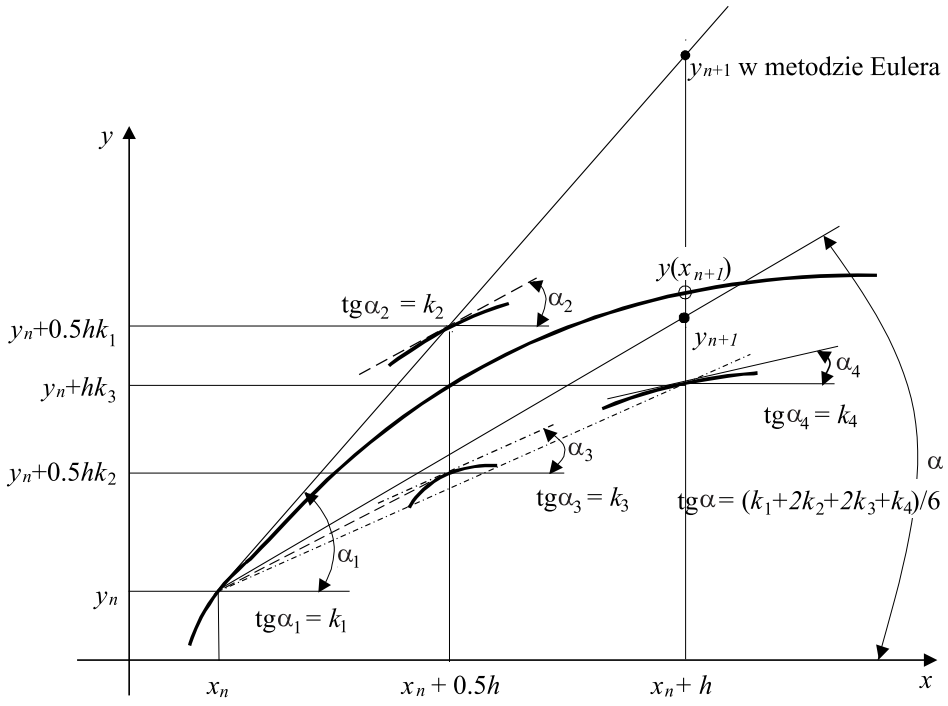
$$k_1 = f(x_n, y_n), \tag{7.20b}$$

$$k_2 = f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1), \tag{7.20c}$$

$$k_3 = f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_2), \tag{7.20d}$$

$$k_4 = f(x_n + h, y_n + hk_3). \tag{7.20e}$$

Interpretacja graficzna wyznaczania kolejnego punktu wg powyższych wzorów przedstawiona jest na rysunku 7.4 (dla jednego równania). Współczynnik k_1 jest pochodną rozwiązania w punkcie (x_n, y_n) (ilustrowaną styczną do krzywej rozwiązania przechodzącego przez ten punkt). Wartość k_2 wyznaczamy jak w zmodyfikowanej metodzie Eulera – jako pochodną rozwiązania wyznaczanego zwykłą



Rys. 7.4. Graficzna interpretacja pojedynczego kroku metody RK4.

metodą Eulera w punkcie $(x_n + 0.5h, y_n + 0.5hk_1)$ (środkowym przedziału), pochodnej tej odpowiada styczna oznaczona linią przerywaną. Następnie, wartość k_3 wyznaczamy podobnie jak k_2, ale tym razem w punkcie $(x_n + 0.5h, y_n + 0.5hk_2)$, tzn. startując z początku przedziału z nachyleniem k_2 (linia przerywana). Pochodnej k_3 odpowiada na rysunku styczna oznaczona linią kreska-kropka. W końcu, z nachyleniem tej stycznej startujemy z punktu początkowego do punktu $x_n + h$, tzn. wyznaczamy pochodną rozwiązania k_4 w punkcie $(x_n + h, y_n + hk_3)$. Mamy w ten sposób wyznaczone 4 wartości pochodnej rozwiązania: po jednej na końcach przedziału i dwie w jego środku. Aproksymacja pochodnej dla pełnego kroku metody wyznaczana jest jako ważona średnia arytmetyczna tych wartości, z wagami 1 na krańcach i wagą 2 w punkcie środkowym – stąd wzór (7.20a).

Przykład 7.4. Dla układu dwóch równań różniczkowych (7.12):

$$\begin{aligned} \frac{dy_1}{dx} &= -a\, y_1 y_2, \\ \frac{dy_2}{dx} &= a\, y_1 y_2 - b\, y_2, \end{aligned}$$

sformułujemy równania jednego kroku metody RK4, z długością kroku h_n, z punktu x_n (aktualna chwila), któremu odpowiada rozwiązanie $y_n = [(y_1)_n\ (y_2)_n]^{\mathrm{T}}$:

$$\begin{aligned}
k_{1,1} &= -a \cdot (y_1)_n \cdot (y_2)_n, \\
k_{1,2} &= a \cdot (y_1)_n \cdot (y_2)_n - b \cdot (y_2)_n, \\
k_{2,1} &= -a[(y_1)_n + \frac{1}{2}h_n k_{1,1}] \cdot [(y_2)_n + \frac{1}{2}h_n k_{1,2}], \\
k_{2,2} &= a[(y_1)_n + \frac{1}{2}h_n k_{1,1}] \cdot [(y_2)_n + \frac{1}{2}h_n k_{1,2}] - b[(y_2)_n + \frac{1}{2}h_n k_{1,2}], \\
k_{3,1} &= -a[(y_1)_n + \frac{1}{2}h_n k_{2,1}] \cdot [(y_2)_n + \frac{1}{2}h_n k_{2,2}], \\
k_{3,2} &= a[(y_1)_n + \frac{1}{2}h_n k_{2,1}] \cdot [(y_2)_n + \frac{1}{2}h_n k_{2,2}] - b[(y_2)_n + \frac{1}{2}h_n k_{2,2}], \\
k_{4,1} &= -a[(y_1)_n + h_n k_{3,1}] \cdot [(y_2)_n + h_n k_{3,2}], \\
k_{4,2} &= a[(y_1)_n + h_n k_{3,1}] \cdot [(y_2)_n + h_n k_{3,2}] - b[(y_2)_n + h_n k_{3,2}], \\
(y_1)_{n+1} &= (y_1)_n + \frac{1}{6}h_n(k_{1,1} + 2k_{2,1} + 2k_{3,1} + k_{4,1}), \\
(y_2)_{n+1} &= (y_2)_n + \frac{1}{6}h_n(k_{1,2} + 2k_{2,2} + 2k_{3,2} + k_{4,2}).
\end{aligned}$$

□

Wybór długości kroku

Podstawowym zagadnieniem przy praktycznej implementacji metod rozwiązywania równań różniczkowych zwyczajnych jest kwestia doboru długości kroku całkowania h_n. Przy wyznaczaniu długości kroku występują dwie przeciwstawne tendencje:

– jeśli krok h_n maleje, to maleje błąd metody (błąd aproksymacji), dla metody zbieżnej błąd maleje do zera przy h dążącym do zera,
– jeśli krok h_n maleje, to zwiększa się liczba iteracji (liczba kroków) potrzebnych do wyznaczenia rozwiązania na zadanym odcinku $[a, b]$, a stąd liczba obliczeń i związanych z nimi błędów numerycznych.

Stąd, nadmierne zmniejszanie długości kroku nie jest wskazane – krok powinien być wystarczający do uzyskania założonej dokładności, ale nie powinien być znacznie mniejszy od wartości, przy której wymagana dokładność jest osiągana.

W myśl powyższych uwag, stały krok całkowania będzie odpowiedni jedynie dla problemów mających rozwiązania o podobnej szybkości zmienności w całym przedziale $[a, b]$, a i w tym przypadku zgadnięcie optymalnej długości kroku nie jest z góry możliwe. Długość taka może być dobrana w trybie interaktywnym przez użytkownika przez wielokrotne zastosowanie metody przy różnych długościach kroku, ale najlepiej, aby była dobierana automatycznie przez metodę. Ponieważ rozwiązania bardzo wielu układów równań różniczkowych są o różnej szybkości

zmienności w granicach przedziału całkowania $[a, b]$, szybkości nierzadko różniącej się o rzędy wielkości, to zastosowanie stałej długości kroku nie jest korzystne. Dla zachowania dokładności w każdym kroku, stała długość kroku musi być bowiem ustalona na małych wartościach odpowiadających przedziałom najszybszej zmienności rozwiązania, co prowadzi do niepotrzebnego zwiększenie nakładu obliczeń na odcinkach o wolniejszej zmienności, a stąd w całym przedziale $[a, b]$. Dlatego też wskazane jest zmienianie długości kroku w trakcie działania metody – automatyczny dobór długości kroku powinien być wbudowany w algorytm każdej efektywnej metody znajdującej rozwiązanie z założoną dokładnością.

Najistotniejszą informacją konieczną dla automatycznego wyznaczania odpowiedniej długości kroku jest *oszacowanie wartości błędu metody (błędu aproksymacji) w danym jej kroku*.

Szacowanie wartości błędu według zasady podwójnego kroku

W celu szacowania błędu, oprócz kroku o długości h wykonujemy dodatkowo, równolegle (tzn. startując też z tego samego punktu x_n) i dokładnie tą samą metodą, dwa dodatkowe kroki o długości $0.5h$ każdy. Wprowadzając oznaczenia:

$y_n^{(1)}$ – nowy punkt uzyskany w kroku o długości h,

$y_n^{(2)}$ – nowy punkt wyznaczony przez dwa kroki o długościach $0.5h$,

i oznaczając przez $r^{(1)}$ błąd aproksymacji w kroku pojedynczym, zaś przez $r^{(2)}$ sumaryczny błąd aproksymacji po dwóch mniejszych krokach (o długości $0.5h$ każdy), zasadę podwójnego kroku można zilustrować, jak to przedstawiono na rysunku 7.5.

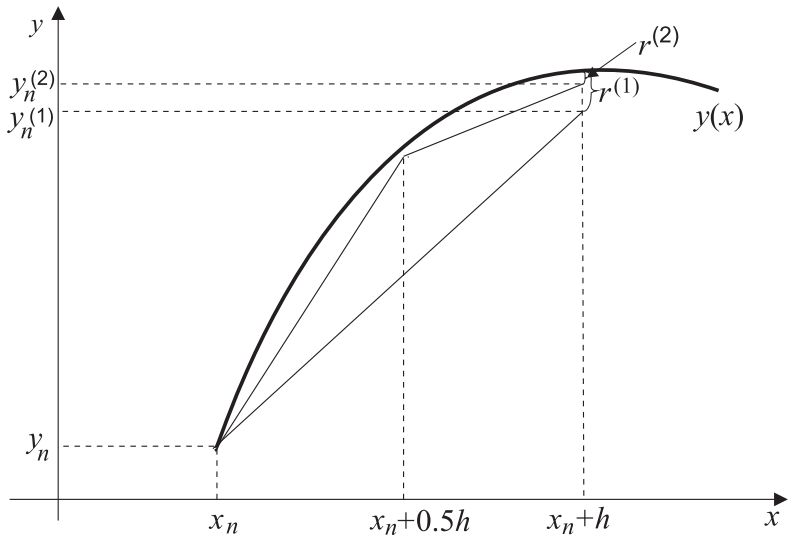
Zakładając identyczny błąd aproksymacji w każdym z dwóch kroków pomocniczych o długości $0.5h$ (świadome przybliżenie, tym lepsze im mniejsza wartość h) oraz oznaczając przez p rząd rozważanej metody, mamy

$$y(x_n + h) = y_n^{(1)} + \underbrace{\frac{r_n^{(p+1)}(0)}{(p+1)!} \cdot h^{p+1}}_{\text{część główna błędu}} + O(h^{p+2}) \; - \; \text{po kroku pojedynczym},$$

$$y(x_n + h) \approx y_n^{(2)} + \underbrace{2 \cdot \frac{r_n^{(p+1)}(0)}{(p+1)!} \left(\frac{h}{2}\right)^{p+1}}_{\text{część główna błędu}} + O\left(h^{p+2}\right) \; - \; \text{po kroku podwójnym}.$$

Po wyznaczeniu nieznanego współczynnika $\gamma = \frac{r_n^{(p+1)}(0)}{(p+1)!}$ z pierwszego równania i wstawieniu go do drugiego, otrzymamy

$$y\left(x_n + h\right) = y_n^{(2)} + \frac{h^{p+1}}{2^p} \frac{y\left(x_n + h\right) - y_n^{(1)}}{h^{p+1}} + O\left(h^{p+2}\right).$$



Rys. 7.5. Graficzna interpretacja zasady podwójnego kroku

Po dalszych przekształceniach

$$y\left(x_n+h\right)\left(1-\frac{1}{2^p}\right)=y_n^{(2)}-\frac{y_n^{(1)}}{2^p}+O\left(h^{p+2}\right)$$
$$=y_n^{(2)}\left(1-\frac{1}{2^p}\right)+\frac{y_n^{(2)}}{2^p}-\frac{y_n^{(1)}}{2^p}+O\left(h^{p+2}\right).$$

Stąd, po pomnożeniu stronami przez $\frac{2^p}{2^p-1}$, dostajemy bezpośrednio

$$y\left(x_n+h\right)=y_n^{(2)}+\frac{y_n^{(2)}-y_n^{(1)}}{2^p-1}+O\left(h^{p+2}\right). \tag{7.21}$$

Przekształcając podobnie, uzyskujemy także

$$y\left(x_n+h\right)=y_n^{(1)}+2^p\frac{y_n^{(2)}-y_n^{(1)}}{2^p-1}+O\left(h^{p+2}\right). \tag{7.22}$$

Z (7.22) wynika *oszacowanie błędu pojedynczego kroku o długości* h (za oszacowanie błędu przyjmuje się oszacowanie jego części głównej), w postaci

$$\delta_n(h)=\frac{2^p}{2^p-1}(y_n^{(2)}-y_n^{(1)}). \tag{7.23}$$

Natomiast, występujący w (7.21) składnik części głównej błędu jest *oszacowaniem błędu dwóch kolejnych kroków o długości* $0.5h$:

$$\delta_n\left(2\times\frac{h}{2}\right)=\frac{y_n^{(2)}-y_n^{(1)}}{2^p-1}. \tag{7.24}$$

Zwróćmy uwagę,że oszacowanie (7.24) jest 2^p razy mniejsze niż oszacowanie (7.23). Stąd, przyjmowanie $y_{n+1} = y_n^{(1)}$ z oszacowaniem błędu (7.23), jakkolwiek poprawne, prowadzi do niewykorzystania dokładniejszego wyniku uzyskanego w dwóch mniejszych krokach. Praktyczniejsze jest przyjmowanie $y_{n+1} = y_n^{(2)}$, z szacowaniem błędu metody co dwa (mniejsze) kroki, równym (7.24) – i, być może, z nieco ostrożniejszym współczynnikiem bezpieczeństwa, zob. podrozdział 7.1.3.

7.1.2. Metody Rungego-Kutty-Fehlberga (RKF)

Rozważmy dwie metody RK, m-etapową rzędu p i $(m{+}1)$-etapową rzędu $p{+}1$.

– Metoda RK rzędu p:

$$\begin{aligned} &y_{n+1} = y_n + h \cdot \sum_{i=1}^{m} w_i^* k_i, \\ &k_1 = f\left(x_n, y_n\right), \\ &k_i = f(x_n + c_i h,\ y_n + h \cdot \sum_{j=1}^{i-1} a_{ij} k_j), \qquad i = 2, 3, ..., m. \end{aligned} \tag{7.25}$$

– Metoda RK rzędu $p{+}1$:

$$\begin{aligned} &y_{n+1} = y_n + h \cdot \sum_{i=1}^{m+1} w_i k_i, \\ &k_1 = f\left(x_n, y_n\right), \\ &k_i = f(x_n + c_i h,\ y_n + h \cdot \sum_{j=1}^{i-1} a_{ij} k_j), \qquad i = 2, 3, ..., m+1. \end{aligned} \tag{7.26}$$

W obu metodach współczynniki w_i^* i w_i są różne, ale *równe są współczynniki* c_i i a_{ij}, dla $j = 1, ..., i-1$, $i = 2, ..., m$, czyli współczynniki k_i są równe dla $i = 1, ..., m$. Jeśli wykonamy krok obydwiema metodami, z tego samego punktu, to dostaniemy:

– dla metody rzędu p:

$$y\left(x_n + h\right) = \underbrace{y_n + h \cdot \sum_{i=1}^{m} w_i^* k_i\left(h\right)}_{y_{n+1}^{(0)}} + \underbrace{\frac{r_n^{(p+1)}(0)}{(p+1)!} h^{p+1}}_{\text{część główna błędu}} + O\left(h^{p+2}\right), \tag{7.27}$$

– dla metody rzędu $p{+}1$:

$$y\left(x_n + h\right) = \underbrace{y_n + h \cdot \sum_{i=1}^{m+1} w_i k_i\left(h\right)}_{y_{n+1}^{(1)}} + \underbrace{\frac{r_n^{(p+2)}(0)}{(p+2)!} h^{p+2} + O\left(h^{p+3}\right)}_{O(h^{p+2})}. \tag{7.28}$$

Po odjęciu stronami powyższych równań i pominięciu „ogona" składającego się z funkcji rzędu $O\left(h^{p+2}\right)$, otrzymujemy oszacowanie błędu metody rzędu p (niższego)

$$\delta_n(h) = \frac{r_n^{(p+1)}(0)}{(p+1)!}h^{p+1} = h \cdot \sum_{i=1}^{m} (w_i - w_i^*) \cdot k_i(h) + h w_{m+1} k_{m+1}(h). \quad (7.29)$$

Omówiona para metod RK to para *metod włożonych*, zwanych metodami RKF (Rungego-Kutty-Fehlberga). Zwięzły zapis współczynników metody m-etapowej wygodnie jest podać w sposób pokazany w tabeli 7.1. Dla pary metod włożonych, m- i $(m+1)$-etapowej, możemy wówczas stosować zapis wspólny, jak to pokazano w tabeli 7.2.

Tabela 7.1. Sposób zapisu parametrów metody m-etapowej RKF

c_i	a_{ij}				w_i^*
0					w_1^*
c_2	a_{21}				w_2^*
c_3	a_{31}	a_{32}			w_3^*
$\vdots$	$\vdots$	$\vdots$			$\vdots$
c_m	a_{m1}	a_{m2}	$\cdots$	$a_{m,m-1}$	w_m^*

Tabela 7.2. Sposób zapisu parametrów pary metod włożonych, m- i $(m+1)$-etapowej

c_i	a_{ij}				w_i^*	w_i
0					w_1^*	w_1
c_2	a_{21}				w_2^*	w_2
$\vdots$	$\vdots$				$\vdots$	$\vdots$
c_m	a_{m1}		$a_{m,m-1}$		w_m^*	w_m
c_{m+1}	$a_{m+1,1}$	$\cdots$	$a_{m+1,m-1}$	$a_{m+1,m}$		w_{m+1}

Parametry par metod włożonych 1. i 2. rzędu (RKF12), 2. i 3. rzędu (RKF23) oraz 4. i 5. rzędu (RKF45), podano w tabelach 7.3, 7.4 i 7.5.

Tabela 7.3. Parametry pary metod włożonych 1. i 2. rzędu (RKF12)

c_i	a_{ij}		w_i^*	w_i
0			$\frac{1}{256}$	$\frac{1}{512}$
$\frac{1}{2}$	$\frac{1}{2}$		$\frac{255}{256}$	$\frac{255}{256}$
1	$\frac{1}{256}$	$\frac{255}{256}$		$\frac{1}{512}$

Tabela 7.4. Parametry pary metod włożonych 2. i 3. rzędu (RKF23)

c_i	a_{ij}			w_i^*	w_i
0				$\frac{214}{891}$	$\frac{533}{2106}$
$\frac{1}{4}$	$\frac{1}{4}$			$\frac{1}{33}$	0
$\frac{27}{40}$	$-\frac{189}{800}$	$\frac{729}{800}$		$\frac{650}{891}$	$\frac{800}{1053}$
1	$\frac{214}{891}$	$\frac{1}{33}$	$\frac{650}{891}$		$-\frac{1}{78}$

Tabela 7.5. Parametry pary metod włożonych 4. i 5. rzędu (RKF45)

c_i	a_{ij}					w_i^*	w_i
0						$\frac{25}{216}$	$\frac{16}{135}$
$\frac{1}{4}$	$\frac{1}{4}$					0	0
$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$				$\frac{1408}{2565}$	$\frac{6656}{12825}$
$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$			$\frac{2197}{4104}$	$\frac{28561}{56430}$
1	$\frac{439}{216}$	-8	$\frac{3680}{513}$	$-\frac{845}{4104}$		$-\frac{1}{5}$	$-\frac{9}{50}$
$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$		$\frac{2}{55}$

Wykonanie jednego kroku obliczeń za pomocą metody RKF45 wymaga, łącznie z szacowaniem błędu, sześciu obliczeń funkcji prawej strony układu równań różniczkowych. Dla porównania: wykonanie jednego kroku obliczeń za pomocą metody RK4 wraz z szacowaniem błędu według zasady podwójnego kroku, realizowanej zgodnie z wzorem (7.21), wymaga jedenastu (4+3+4) obliczeń wartości funkcji prawej strony – ale jest to metoda licząca z krokiem $0.5h$, jeśli jako punkty następne y_{n+1} przyjmujemy punkty wyznaczane w wyniku wykonania dwóch kroków połówkowych (co jest zalecanym sposobem implementacji metod RK z szacowaniem błędu według zasady podwójnego kroku). Należy więc porównywać łączny nakład na dwa kroki każdej metody – stąd w przypadku kroków udanych, spełniających test dokładności ($s\alpha \geq 1$), nakład obliczeń jest porównywalny, a nawet nieco mniejszy dla RK4: 11 obliczeń wobec 12. Natomiast w przypadku kroków nieudanych metody RKF są efektywniejsze niż RK, ponieważ w RKF powtarzamy tylko jeden krok nieudany, a w RK z szacowaniem błędu metodą podwójnego kroku powtarzamy dwa kroki (o długości $0.5h$).

7.1.3. Wyznaczanie zmienionej długości kroku

Ogólny wzór na część główną błędu metody rzędu p dla kroku h jest w postaci

$$\delta_n(h) = \gamma \cdot h^{p+1}, \quad \text{gdzie} \quad \gamma = \frac{r_n^{(p+1)}(0)}{(p+1)!} \tag{7.30}$$

jest pierwszym niezerowym współczynnikiem w rozwinięciu błędu aproksymacji w szereg Taylora. Gdy zmieniamy krok z h na αh, to

$$\delta_n(\alpha h) = \gamma \cdot (\alpha h)^{p+1},$$

skąd

$$\delta_n(\alpha h) = \alpha^{p+1} \cdot \delta_n(h). \tag{7.31}$$

Założenie dokładności obliczeń na wartości ε oznacza

$$|\delta_n(\alpha h)| = \varepsilon. \tag{7.32}$$

Z (7.31) i (7.32) uzyskujemy

$$\alpha^{p+1}|\delta_n(h)| = \varepsilon,$$

skąd dostajemy współczynnik modyfikacji kroku α,

$$\alpha = \left(\frac{\varepsilon}{|\delta_n(h)|}\right)^{\frac{1}{p+1}}. \tag{7.33}$$

Wzór ten jest oczywiście słuszny dla dowolnej metody z oszacowaniem błędu $\delta_n(h)$. Jest on też słuszny dla metod RK z szacowaniem błędu wg zasady podwójnego kroku zależnością (7.24), tzn. gdy $\delta_n(h) = \delta_n\left(2 \times \frac{h}{2}\right)$. Albowiem

$$\delta_n\left(2 \times \frac{h}{2}\right) \approx 2\gamma \left(\frac{h}{2}\right)^{p+1},$$

$$\delta_n\left(2 \times \frac{\alpha h}{2}\right) \approx 2\gamma \left(\frac{\alpha h}{2}\right)^{p+1} = \alpha^{p+1} \cdot 2\gamma \left(\frac{h}{2}\right)^{p+1} = \alpha^{p+1} \cdot \delta_n\left(2 \times \frac{h}{2}\right),$$

tj. dostaliśmy zależność analogiczną do (7.31).

W praktyce, dla uwzględnienia niedokładności oszacowania błędu, stosuje się jeszcze współczynnik bezpieczeństwa s, stąd

$$h_{n+1} = s \cdot \alpha \cdot h_n, \quad \text{gdzie } s < 1 \quad \text{(np. dla RK4, RKF45: } s \approx 0.9\text{)}. \tag{7.34}$$

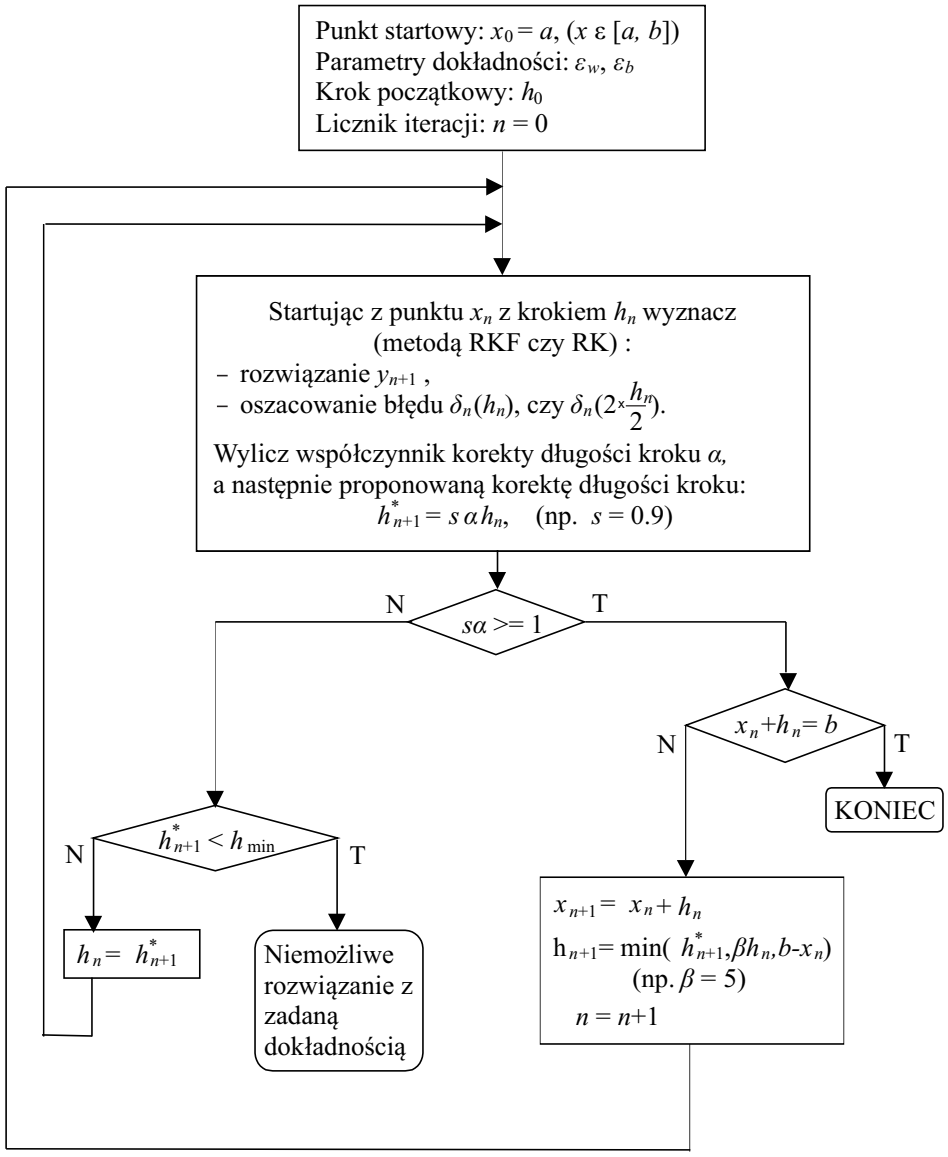
Ponadto, parametr dokładności obliczeń ε (będący parametrem użytkownika) określa się na ogół następująco:

$$\varepsilon = |y_n| \cdot \varepsilon_w + \varepsilon_b, \tag{7.35}$$

gdzie

ε_w – dokładność względna,

ε_b – dokładność bezwzględna.



Rys. 7.6. Schemat blokowy podstawowej realizacji metod RK i RKF

.

Dla układu k równań przyjmuje się współczynnik wyliczony dla najbardziej krytycznego równania (na najgorszy przypadek), np. dla metod RK z szacowaniem błędu wg zasady podwójnego kroku, mamy

$$\delta_n(h)_i = \frac{(y_i)_n^{(2)} - (y_i)_n^{(1)}}{2^p - 1}, \qquad i = 1, 2, ..., k, \tag{7.36}$$

$$\varepsilon_i = \left|(y_i)_n^{(2)}\right| \cdot \varepsilon_w + \varepsilon_b, \tag{7.37}$$

$$\alpha = \min_{1 \le i \le k} \left(\frac{\varepsilon_i}{|\delta_n(h)_i|}\right)^{\frac{1}{p+1}}. \tag{7.38}$$

Na rysunku 7.6 przedstawiony jest schemat blokowy podstawowej realizacji metod Rungego-Kutty (RKF czy RK). Warto zwrócić uwagę na zastosowane w tym schemacie, heurystyczne ograniczenie maksymalnego wzrostu długości kroku h_n w jednej iteracji, do wartości co najwyżej βh_n, gdzie β jest heurystycznym współczynnikiem.

7.2. Metody wielokrokowe

Ogólna postać wzoru definiującego krok (iterację) *metody k-krokowej* liniowej, ze stałą długością kroku h, jest następująca:

$$y_n = \sum_{j=1}^{k} \alpha_j \cdot y_{n-j} + h \sum_{j=0}^{k} \beta_j \cdot f(x_{n-j}, y_{n-j}), \tag{7.39}$$

gdzie $y_0 = y(x_0) = y_a,\ x_n = x_0 + nh,\ x_0 = a,\ x \in [a, b]$.

Metoda wielokrokowa jest *jawna* (*explicit method*), jeśli $\beta_0 = 0$. Dla metody jawnej wartość y_n zależy jawnie (*explicite*) od wartości y i $f(x, y)$ jedynie w poprzednich (już obliczonych) punktach, tj od wartości $y_{n-1},\ y_{n-2},\ \dots,\ y_{n-k}$, $f_{n-1} = f(x_{n-1}, y_{n-1})$, $f_{n-2} = f(x_{n-2}, y_{n-2})$, ... ,$f_{n-k} = f(x_{n-k}, y_{n-k})$.

Metoda wielokrokowa jest *niejawna*(*implicit method)*, jeśli $\beta_0 \neq 0$. Dla metody niejawnej nowa wartość y_n obliczana jest na podstawie k poprzednich wartości y_{n-1}, ..., y_{n-k} i $f_{n-1}, \dots, f_{n-k}$, gdzie $f_j = f(x_j, y_j)$, oraz *dodatkowo również wartości w punkcie bieżącym* $f_n = f(x_n, y_n)$, tj. do wyznaczenia y_n trzeba w istocie rozwiązać równanie algebraiczne $\varphi(y_n) = 0$ (nieliniowe, jeśli funkcja prawej strony f jest nieliniowa), gdzie

$$\varphi(y_n) \stackrel{\text{df}}{=} -y_n + \sum_{j=1}^{k} \alpha_j y_{n-j} + h \cdot \sum_{j=1}^{k} \beta_j f(x_{n-j}, y_{n-j}) + h\beta_0 f(x_n, y_n). \tag{7.40}$$

Dla $\beta_0 \neq 0$ równanie to może mieć niejednoznaczne rozwiązanie, natomiast jeśli

$$h < (L\beta_0)^{-1},$$

to (przy spełnieniu założeń twierdzenia 7.1) jest ono jednoznaczne, gdzie L jest stałą Lipschitza funkcji f (względem y, jak w twierdzeniu 7.1).

Równanie algorytmu metody wielokrokowej (7.39) jest równaniem różnicowym. Do pierwszego zastosowania tego równania na przedziale $[a, b]$ z krokiem h potrzeba k pierwszych wartości rozwiązania, $y_0, ..., y_{k-1}$, których nie można wyznaczyć procedurą k-krokową. Zatem trzeba stworzyć specjalną *procedurę startową* obliczającą te wartości. Dla pierwszych k kroków można zastosować np. odpowiedni (o podobnej dokładności) algorytm RK.

7.2.1. Metody Adamsa

Równanie różniczkowe

$$\begin{aligned} y'(x) &= f(x, y(x)), \\ y(a) &= y_a, \quad x \in [a, b], \end{aligned}$$

równoważne jest równaniu całkowemu

$$y(x) = y(a) + \int_a^x f(t, y(t))\, dt.$$

Metody Adamsa dostajemy, rozważając to równanie na przedziale $[x_{n-1}, x_n]$:

$$y(x_n) = y(x_{n-1}) + \int_{x_{n-1}}^{x_n} f(t, y(t))\, dt. \tag{7.41}$$

Metody jawne (Adamsa-Bashfortha)

Funkcję podcałkową f w (7.41) przybliżamy wielomianem interpolacyjnym $W(x)$ stopnia co najwyżej $k-1$ opartym na węzłach $x_{n-1}, ..., x_{n-k}$. Przyjmując przybliżenie $y(x_{n-j}) \approx y_{n-j}$ i stosując wzór interpolacyjny Lagrange'a (zob. rozdz. 5.1.1), mamy

$$f(x, y(x)) \approx W(x) = \sum_{j=1}^{k} f(x_{n-j}, y_{n-j})\, L_j(x), \tag{7.42}$$

$$y_n = y_{n-1} + \sum_{j=1}^{k} f(x_{n-j}, y_{n-j}) \int_{x_{n-1}}^{x_n} L_j(t)\, dt,$$

gdzie $L_j(x)$ to wielomiany Lagrange'a,

$$L_j(x) = \prod_{m=1, m\neq j}^{k} \frac{x - x_{n-m}}{x_{n-j} - x_{n-m}}.$$

Stąd, po scałkowaniu, przy założeniu $x_{n-j} = x_n - jh$, $j = 1, 2, ..., k$, otrzymujemy

$$y_n = y_{n-1} + h \sum_{j=1}^{k} \beta_j f(x_{n-j}, y_{n-j}), \tag{7.43}$$

gdzie wartości współczynników β dla kilku wartości k podano w tabeli 7.6.

Tabela 7.6. Parametry metod jawnych Adamsa (metod Adamsa-Bashfortha)

k	β_1	β_2	β_3	β_4	β_5	β_6	β_7
1	1						
2	$\frac{3}{2}$	$-\frac{1}{2}$					
3	$\frac{23}{12}$	$-\frac{16}{12}$	$\frac{5}{12}$				
4	$\frac{55}{24}$	$-\frac{59}{24}$	$\frac{37}{24}$	$-\frac{9}{24}$			
5	$\frac{1901}{720}$	$-\frac{2774}{720}$	$\frac{2616}{720}$	$-\frac{1274}{720}$	$\frac{251}{720}$		
6	$\frac{4277}{1440}$	$-\frac{7923}{1440}$	$\frac{9982}{1440}$	$-\frac{7298}{1440}$	$\frac{2877}{1440}$	$-\frac{475}{1440}$	
7	$\frac{198721}{60480}$	$-\frac{447288}{60480}$	$\frac{705549}{60480}$	$-\frac{688256}{60480}$	$\frac{407139}{60480}$	$-\frac{134472}{60480}$	$\frac{19087}{60480}$

Metody niejawne (Adamsa-Moultona)

Funkcję podcałkową w (7.41) przybliżamy wielomianem interpolacyjnym stopnia co najwyżej k opartym na węzłach $x_n, x_{n-1}, ..., x_{n-k}$, z wartościami rozwiązania $y(x_{n-j}) \approx y_{n-j}$. Następnie, postępując tak jak w przypadku poprzednim metod Adamsa-Bashfortha, otrzymamy

$$\begin{aligned} y_n &= y_{n-1} + h \sum_{j=0}^{k} \beta_j^* \cdot f(x_{n-j}, y_{n-j}) \\ &= y_{n-1} + h \cdot \beta_0^* \cdot f(x_n, y_n) + h \sum_{j=1}^{k} \beta_j^* \cdot f(x_{n-j}, y_{n-j}). \end{aligned} \tag{7.44}$$

Wartości parametrów β_j^* dla kilku wartości k podano w tabeli 7.7, gdzie $k = 1^+$ oznacza wsteczną metodę Eulera (zob. przykład 7.7 w rozdziale 7.3).

Tabela 7.7. Parametry metod niejawnych Adamsa (metod Adamsa-Moultona)

k	β_0^*	β_1^*	β_2^*	β_3^*	β_4^*	β_5^*	β_6^*	β_7^*
1^+	1							
1	$\frac{1}{2}$	$\frac{1}{2}$						
2	$\frac{5}{12}$	$\frac{8}{12}$	$-\frac{1}{12}$					
3	$\frac{9}{24}$	$\frac{19}{24}$	$-\frac{5}{24}$	$\frac{1}{24}$				
4	$\frac{251}{720}$	$\frac{646}{720}$	$-\frac{264}{720}$	$\frac{106}{720}$	$-\frac{19}{720}$			
5	$\frac{475}{1440}$	$\frac{1427}{1440}$	$-\frac{798}{1440}$	$\frac{482}{1440}$	$-\frac{173}{1440}$	$\frac{27}{1440}$		
6	$\frac{19087}{60480}$	$\frac{65112}{60480}$	$-\frac{46461}{60480}$	$\frac{37504}{60480}$	$-\frac{20211}{60480}$	$\frac{6312}{60480}$	$-\frac{863}{60480}$	
7	$\frac{36799}{120960}$	$\frac{139849}{120960}$	$-\frac{121797}{120960}$	$\frac{123133}{120960}$	$-\frac{88547}{120960}$	$\frac{41499}{120960}$	$-\frac{11351}{120960}$	$\frac{1375}{120960}$

7.2.2. Błąd aproksymacji

Błędem aproksymacji odpowiadającym punktowi x_n nazywamy różnicę

$$r_n(h) \stackrel{\text{df}}{=} [\sum_{j=1}^{k} \alpha_j y(x_{n-j}) + h \sum_{j=0}^{k} \beta_j f(x_{n-j}, y(x_{n-j}))] - y(x_n), \qquad (7.45)$$

tzn. błąd aproksymacji $r_n(h)$ to błąd metody, który informuje o tym, jaki błąd wnosi metoda w jednym kroku (n-tym, od x_{n-1} do x_n), a więc przy przyjęciu w równaniu metody (wyrażenie w nawiasie kwadratowym) punktów dokładnych, $y_{n-j} = y(x_{n-j}),\ j = 0, \ldots, k$. Jest to definicja taka sama jak w przypadku metod jednokrokowych, zob. wzór (7.15) w rozdziale 7.1.

Uwaga*. Łącząc równanie metody i równanie błędu dostajemy:

$$y_n - y(x_n) = h\beta_0[f(x_n, y_n) - f(x_n, y(x_n))] + r_n(h).$$

Stosując twierdzenie o wartości średniej, dostajemy dalej

$$r_n(h) = (1 - h\beta_0 \frac{\partial f}{\partial y}(x_n, \zeta_n))(y_n - y(x_n)), \qquad (7.46)$$

gdzie $\zeta_n \in (y_n, y(x_n))$. Tak więc *dla metody jawnej* błąd aproksymacji *jest równy różnicy* $y_n - y(x_n)$ (identycznie jak dla metod jednokrokowych), a dla metody *niejawnej jest proporcjonalny do tej różnicy,* przy czym współczynnik proporcjonalności jest tym bliższy wartości 1, im mniejsza jest długość kroku h. □

*Materiał uzupełniający.

Przyjmując $x_{n-j} = x_n - jh$ (stały krok), $j = 0, 1, ..., k,$ oraz uwzględniając, że wartości prawych stron układu równań różniczkowych są wartościami pochodnych jego rozwiązania, mamy

$$r_n(h) = \sum_{j=0}^{k} \alpha_j y(x_n - jh) + h \sum_{j=0}^{k} \beta_j y'(x_n - jh), \quad \text{gdzie } \alpha_0 = -1. \tag{7.47}$$

Rozwijając $y(x_n - jh)$ i $y'(x_n - jh)$ w szereg Taylora w otoczeniu x_n (przyjmując, że funkcja $y(x)$ jest odpowiednio różniczkowalna), a następnie grupując uzyskane człony w kolejności rosnących rzędów pochodnych, dostajemy

$$r_n(h) = c_0 y(x_n) + \sum_{m=1}^{p+1} h^m c_m y^{(m)}(x_n) + O\left(h^{p+2}\right), \tag{7.48}$$

gdzie

$$\begin{aligned} c_0 &= \sum_{j=0}^{k} \alpha_j = -1 + \sum_{j=1}^{k} \alpha_j, \\ c_1 &= -\sum_{j=1}^{k} j\alpha_j + \sum_{j=0}^{k} \beta_j, \\ c_m &= \frac{1}{m!} \sum_{j=1}^{k} (-j)^m \alpha_j + \frac{1}{(m-1)!} \sum_{j=1}^{k} (-j)^{(m-1)} \beta_j, \ \ m \geq 2. \end{aligned} \tag{7.49}$$

Definicja. *Metoda wielokrokowa jest rzędu p*, jeśli w rozwinięciu (7.49) dostajemy

$$c_0 = 0, c_1 = 0, \ldots, c_p = 0, \ c_{p+1} \neq 0. \tag{7.50}$$

Stąd, dla metody rzędu p błąd aproksymacji wyraża się wzorem

$$r_n(h) = c_{p+1} h^{p+1} y^{(p+1)}(x_n) + O(h^{p+2}), \tag{7.51}$$

gdzie wartość c_{p+1} to *stała błędu*, zaś $c_{p+1} h^{p+1} y^{(p+1)}(x_n)$ to *część główna błędu* aproksymacji.

Wartości rzędów i stałych błędu metod jawnych Adamsa podano w tabeli 7.8, a metod niejawnych w tabeli 7.9, gdzie $k = 1^+$ oznacza wsteczną metodę Eulera (zob. przykład 7.7 w rozdziale 7.3). Zwróćmy uwagę na istotne różnice miedzy metodami jawnymi i niejawnymi: dla tej samej wartości k rząd metody niejawnej jest o jeden większy, zaś przy tym samym rzędzie stała błędu metody niejawnej jest znacznie mniejsza.

Tabela 7.8. Rzędy i stałe błędu metod jawnych Adamsa

k	1	2	3	4	5	6	7
p	1	2	3	4	5	6	7
c_{p+1}	$-\frac{1}{2}$	$-\frac{5}{12}$	$-\frac{3}{8}$	$-\frac{251}{720}$	$-\frac{95}{288}$	$-\frac{19087}{60480}$	$-\frac{36799}{120960}$

Tabela 7.9. Rzędy i stałe błędu metod niejawnych Adamsa

k	1^+	1	2	3	4	5	6	7
p		2	3	4	5	6	7	8
c^*_{p+1}	$\frac{1}{2}$	$\frac{1}{12}$	$\frac{1}{24}$	$\frac{19}{720}$	$\frac{3}{360}$	$\frac{863}{60480}$	$\frac{275}{24192}$	$\frac{339533}{3628800}$

7.2.3. Stabilność i zbieżność

Metoda wielokrokowa jest *stabilna*, jeśli jej równanie różnicowe dla $h = 0$, określone zależnością

$$-y_n + \sum_{j=1}^{k} \alpha_j y_{n-j} = -y_n + \alpha_1 y_{n-1} + \cdots + \alpha_{k-1} y_{n-k-1} + \alpha_k y_{n-k} = 0, \quad (7.52)$$

jest stabilne. *Warunek stabilności* równania (7.52): zera jego *wielomianu charakterystycznego*

$$\rho(z) = -z^k + \alpha_1 z^{k-1} + ... + \alpha_{k-1} z + \alpha_k \quad (7.53)$$

mają moduły nie większe od 1 (a w przypadku równych 1 są to zera pojedyncze).

Twierdzenie 7.2. *Metoda k-krokowa jest zbieżna wtedy i tylko wtedy, gdy jest stabilna i jest co najmniej rzędu pierwszego (tzn. $c_0 = c_1 = 0$ – tzw. warunek aproksymacji, zgodności).*

Zdefiniujemy też wielomian związany ze współczynnikami β_j (tzn. ze składnikami równania metody (7.39) zależnymi od pochodnych rozwiązania – wartościami $f(x_{n-j}, y_{n-j}),\ j = 0, ..., k$), w postaci

$$\sigma(z) = \beta_0 z^k + ... + \beta_{k-1} z + \beta_k. \quad (7.54)$$

Odpowiednio dobierając wartości współczynników α_j i β_j, można konstruować metody wielokrokowe spełniające warunki zbieżności (co jest wymaganiem koniecznym dla każdej potencjalnie użytecznej metody), mające możliwie duży rząd i/lub możliwie małe wartości stałej błędu.

Przykład 7.5. Rozważmy ogólnie metodę jawną dwukrokową. Mamy

$$y_n = \alpha_1 y_{n-1} + \alpha_2 y_{n-2} + h\beta_1 f\left(x_{n-1}, y_{n-1}\right) + h\beta_2 f\left(x_{n-2}, y_{n-2}\right).$$

Stąd

$$\rho(z) = -z^2 + \alpha_1 z + \alpha_2,$$
$$\sigma(z) = \beta_1 z + \beta_2.$$

Mamy cztery współczynniki, stąd możemy zaprojektować metodę co najmniej rzędu drugiego. Warunki rzędu drugiego (zawierające warunki aproksymacji) są następujące:

$$\begin{aligned} c_0 = 0 &\Rightarrow -1 + \alpha_1 + \alpha_2 = 0, \\ c_1 = 0 &\Rightarrow -\alpha_1 - 2\alpha_2 + \beta_1 + \beta_2 = 0, \\ c_2 = 0 &\Rightarrow \frac{1}{2}(\alpha_1 + 4\alpha_2) - \beta_1 - 2\beta_2 = 0. \end{aligned}$$

Są to trzy równania z czterema niewiadomymi. Traktując β_2 jako parametr, mamy trzy równania z trzema niewiadomymi. Po rozwiązaniu tego układu równań otrzymujemy

$$\alpha_1 = -2\beta_2, \quad \alpha_2 = 1 + 2\beta_2, \quad \beta_1 = \beta_2 + 2,$$

skąd

$$\rho(z) = -z^2 + -2\beta_2 z + 1 + 2\beta_2.$$

Zauważmy, że wartość $z_1 = 1$ jest pierwiastkiem wielomianu charakterystycznego $\rho(z)$ (dla metody zbieżnej jedynka jest zawsze pierwiastkiem tego wielomianu, ponieważ $\rho(1) = -1 + \sum_{i=1}^{n} \alpha_i = c_0 = 0$). Uwzględniając ten fakt, wielomian $\rho(z)$ możemy zapisać w postaci

$$\rho(z) = -(z-1)(z-z_2),$$

skąd wyliczamy $z_2 = -(1 - 2\beta_2)$. Warunki stabilności wymagają, aby

$$-1 \leq z_2 < 1, \quad \text{co implikuje} \quad -1 < \beta_2 \leq 0.$$

Dowolna wartość β_2 z powyższego zakresu da nam metodę rzędu co najmniej 2. Można wykorzystać β_2 do doboru zarówno odpowiedniej wartości stałej błędu, jak i wartości z_2, mającej wpływ na cechy stabilnościowe metody. Łatwo wyliczyć, że

$$c_3 = \frac{1}{6}\beta_2 - \frac{1}{3}.$$

Dla $\beta_2 = 0$ dostajemy metodę o minimalnym module c_3, wówczas $|c_3| = \frac{1}{3}$ oraz $z_2 = -1$ (metoda dwukrokowa Nystroema). Dla $\beta_2 = -\frac{1}{2}$ dostajemy $c_3 = -\frac{5}{12}$ i $z_2 = 0$ (metoda dwukrokowa Adamsa-Bashfortha). □

Decydująca dla praktyki jest stabilność równania różnicowego metody dla *skończonych wartości kroku całkowania* h. Przy $h \neq 0$ zachowanie się ciągów $\{y_n\}$ generowanych przez metodę zależy jednakże również od cech rozwiązywanego problemu, tzn. funkcji $f(x, y)$ prawej strony układu równań różniczkowych (określających pochodne rozwiązania). Stabilność dla kroków niezerowych definiuje się dla zadania liniowego (tzn. z liniowymi funkcjami $f(x, y)$), z asymptotycznie stabilnym rozwiązaniem. Dla zadania nieliniowego odpowiada to rozważaniu jego przybliżenia liniowego.

Na wstępie wystarczy rozważyć jednowymiarowy problem liniowy

$$\begin{aligned} y'(x) &= \lambda \cdot y(x), \\ y(0) &= 1, \quad x \in [0, b\,],\ b \gg 0, \end{aligned} \tag{7.55}$$

gdzie $\lambda \in \mathbb{C}$ (ze względu na późniejsze uogólnienie dla układu równań), zaś z założenia asymptotycznej stabilności wynika $\operatorname{Re}\lambda < 0$, czyli rozwiązanie problemu zbiega do zera wraz ze zmianami x od zera 0 do b. Stosując do tego problemu ogólną metodę wielokrokową (7.39), uzyskujemy następujące rownanie różnicowe

$$y_n = \sum_{j=1}^{k} \alpha_j y_{n-j} + h \cdot \sum_{j=0}^{k} \beta_j \lambda y_{n-j}, \tag{7.56}$$

które zapiszemy w postaci

$$\sum_{j=0}^{k} (\alpha_j + h\lambda\beta_j)\, y_{n-j} = 0, \qquad \text{gdzie } \alpha_0 = -1. \tag{7.57}$$

Przypomnijmy, że równanie (7.57) jest asymptotycznie stabilne wtedy i tylko wtedy, gdy wszystkie zera jego wielomianu charakterystycznego leżą wewnątrz koła jednostkowego. Wielomian charakterystyczny równania (7.57) można zaś zapisać w postaci

$$\widetilde{\rho}\,(z; h\lambda) = \rho\,(z) + h\lambda\,\sigma(z)\,, \tag{7.58}$$

gdzie wielomiany $\rho\,(z)$ i $\sigma(z)$ dane są zależnościami (7.53) i (7.54).

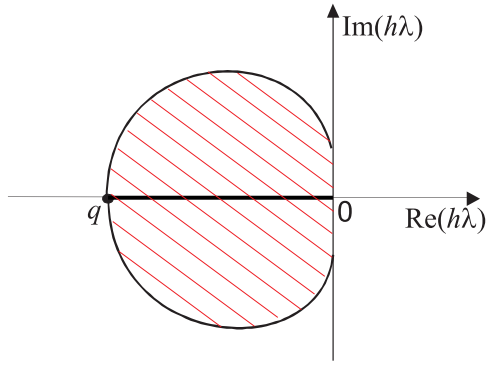
Mówimy, że metoda wielokrokowa z krokiem $h > 0$ jest *absolutnie stabilna* dla problemu (7.55), jeśli równanie różnicowe (7.57) jest asymptotycznie stabilne, tzn. opisuje asymptotycznie stabilny dyskretny system dynamiczny (ciąg uzyskiwanych punktów rozwiązania numerycznego $y_0 = 1, y_1, y_2, \ldots, y_n, y_{n+1}, \ldots$ zbiega do zera przy $n \to \infty$). Abstrahując nawet od pochodzenia iloczynu $h\lambda$, można sformułować definicję:

Definicja. Zbiór wszystkich wartości $h\lambda$, gdzie $\lambda \in \mathbb{C},\ \operatorname{Re}\lambda < 0,\ h \in \mathbb{R},\ h > 0$ (czyli $h\lambda \in \mathbb{C},\ \operatorname{Re}(h\lambda) < 0$), dla którego zera wielomianu (7.58) są położone

wewnątrz koła jednostkowego, nazywamy *obszarem absolutnej stabilności* metody wielokrokowej (zdefiniowanej wielomianami $\rho(z)$ i $\sigma(z)$), metodę nazywamy *absolutnie stabilną* w tym obszarze.

Obszary absolutnej stabilności są podzbiorami płaszczyzny zmiennej zespolonej położonymi w lewej półpłaszczyźnie (ponieważ $\mathrm{Re}(h\lambda) < 0$). Można pokazać, że jeśli wszystkie pierwiastki wielomianu $\rho(z)$, poza jednym pojedynczym równym jedności (który zawsze występuje dla metody zbieżnej, jak to wyjaśniono w przykładzie 7.5), mają moduły mniejsze od jedności, to dla dostatecznie małych wartości $h\lambda$ metoda jest absolutnie stabilna (jej obszar absolutnej stabilności ma niepuste wnętrze).

Kształt obszarów absolutnej stabilności dla metod Adamsa pokazano na rysunku 7.7. Odcinek $[q, 0]$ przedstawiony na tym rysunku nazywany jest *odcinkiem absolutnej stabilności*, wartości lewego krańca q tego odcinka ($|q|$ jest jego długością) podane są dalej w tabeli 7.10.



Rys. 7.7. Kształt obszarów absolutnej stabilności metod Adamsa

Przedstawiona definicja absolutnej stabilności przenosi się natychmiast na przypadek wielowymiarowy

$$\begin{aligned} &\mathbf{y}'(x) = \mathbf{A}\mathbf{y}(x), \qquad \mathbf{y} \in \mathbb{R}^m, \\ &\mathbf{y}(0) = 1, \quad x \in [0, b], \; b \gg 0, \end{aligned} \tag{7.59}$$

gdzie wszystkie wartości własne λ_i macierzy $\mathbf{A}$ mają ujemne części rzeczywiste, $\mathrm{Re}\lambda_i < 0$, ponieważ układ (7.59) ma mieć asymptotycznie stabilne rozwiązania. Dla absolutnej stabilności metody wielokrokowej z krokiem h, zastosowanej do problemu (7.59), *warunkiem koniecznym i dostatecznym jest, aby dla każdej wartości własnej λ_i macierzy $\mathbf{A}$ iloczyn $h\lambda_i$ należał do obszaru absolutnej stabilności metody*.

7.2.4. Metody predyktor-korektor

Najpraktyczniejszą metodą wielokrokową byłaby metoda o:

1. wysokim rzędzie i małej stałej błędu,
2. możliwie dużym obszarze absolutnej stabilności,
3. możliwie małej liczbie obliczeń na iterację.

Metody jawne gorzej spełniają dwa pierwsze warunki, natomiast metody niejawne spełniają je znacznie lepiej, ale nie wypełniają warunku trzeciego, gdyż w każdej iteracji trzeba rozwiązywać względem y_n równanie nieliniowe

$$-y_n + \sum_{j=1}^{k} \left(\alpha_j^* y_{n-j} + h\beta_j^* f_{n-j}\right) + h\beta_0^* f\left(x_n, y_n\right) = 0. \tag{7.60}$$

Praktyczne realizacje metod wielokrokowych to *algorytmy typu predyktor - korektor* (algorytmy PK, *predictor-corrector*), będące połączeniem metod jawnych i niejawnych w jeden algorytm. Dla metody k-krokowej realizacja w postaci *struktury predyktor- korektor* $\mathrm{P}_k\mathrm{EK}_k\mathrm{E}$ ma postać:

$$\begin{aligned}
&\text{P:} \quad y_n^{[0]} = \sum_{j=1}^{k} \alpha_i y_{n-j} + h \sum_{j=1}^{k} \beta_j f_{n-j}, && \text{(P – predykcja)}\\
&\text{E:} \quad f_n^{[0]} = f\left(x_n, y_n^{[0]}\right), && \text{(E – ewaluacja)}\\
&\text{K:} \quad y_n = \sum_{j=1}^{k} \alpha_j^* y_{n-j} + h \sum_{j=1}^{k} \beta_j^* f_{n-j} + h\beta_0^* f_n^{[0]}, && \text{(K – korekcja)}\\
&\text{E:} \quad f_n = f\left(x_n, y_n\right). && \text{(E – ewaluacja)}
\end{aligned}$$

Interpretacja: Iteracja predyktora to obliczenie dobrego punktu początkowego (tym lepszego, im mniejszy krok h i wyższy rząd predyktora) dla iteracji algorytmu korektora rozwiązującego nieliniowe równanie algebraiczne metody niejawnej korektora (7.60) metodą iteracji prostej (zob. rozdz. 6.2.3). W algorytmie $\mathrm{P}_k\mathrm{EC}_k\mathrm{E}$ przedstawionym powyżej wykonujemy tylko jedną iterację algorytmu korektora. Dokładniejsze wyniki otrzymamy wykonując więcej iteracji tego algorytmu w jednym kroku metody, dla m iteracji korektora, tj. w metodzie $\mathrm{P}_k(\mathrm{EK}_k)^m\mathrm{E}$, mamy:

$$\begin{aligned}
&\text{P:} \quad y_n^{[0]} = \sum_{j=1}^{k} \alpha_i y_{n-j} + h \sum_{j=1}^{k} \beta_j f_{n-j}, \quad s = 0,\\
&\text{E}_s\text{:} \quad f_n^{[s]} = f(x_n, y_n^{[s]}),\\
&\text{K:} \quad y_n^{[s+1]} = \sum_{j=1}^{k} \alpha_j^* y_{n-j} + h \sum_{j=1}^{k} \beta_j^* f_{n-j} + h\beta_0^* f_n^{[s]},\\
&\quad s = s + 1, \ \text{jeśli } s < m \ \text{ idź do E}_s,\\
&\text{E:} \quad f_n = f(x_n, y_n^{[m]}).
\end{aligned}$$

Podkreślmy: *metoda PK to w istocie przybliżony sposób realizacji metody niejawnej (korektora)*, algorytm predyktora gra tu rolę pomocniczą polegająca na efektywnym wyliczeniu dobrego punktu startowego.

Dla *metod Adamsa* algorytm $\mathrm{P}_k\mathrm{EK}_k\mathrm{E}$ ma postać:

$$\begin{aligned}
&\text{P:} \quad y_n^{[0]} = y_{n-1} + h\sum_{j=1}^{k}\beta_j f_{n-j},\\
&\text{E:} \quad f_n^{[0]} = f(x_n, y_n^{[0]}),\\
&\text{K:} \quad y_n = y_{n-1} + h\sum_{j=1}^{k}\beta_j^* f_{n-j} + h\beta_0^* f_n^{[0]},\\
&\text{E:} \quad f_n = f\left(x_n, y_n\right).
\end{aligned}$$

Wartości lewego krańca q odcinka absolutnej stabilności dla metod Adamsa jawnej, niejawnej i $\mathrm{P}_k\mathrm{EK}_k\mathrm{E}$ przedstawiono w tabeli 7.10.

Tabela 7.10. Wartości lewego krańca q odcinka absolutnej stabilności metod Adamsa

	metoda Adamsa		
k	jawna	niejawna	$\mathrm{P}_k\mathrm{EK}_k\mathrm{E}$
1	-2	$-\infty$	-2
2	-1	-6	-2.4
3	-0.55	-3	-2
4	-0.3	-1.83	-1.4
5	-0.18	-1.18	-1.05
6	-0.12	-0.78	-0.76

Następujące twierdzenie, dotyczące rzędu metod PK, ma podstawowe znaczenie dla praktycznej wartości tych metod (zob. [7]).

Twierdzenie 7.3. *Jeśli rząd predyktora p_p jest nie mniejszy od rzędu korektora p, $p_p \geq p$, to dla dowolnej liczby iteracji korektora $m = 1, 2, ...$*

$$y_n^{[m]} - y\left(x_n\right) = c_{p+1}^* h^{p+1} y^{(p+1)}\left(x_n\right) + O\left(h^{p+2}\right), \tag{7.61}$$

gdzie c_{p+1}^* jest stałą błędu korektora.
Jeśli natomiast rząd korektora p jest większy od rzędu predyktora p_p, $p = p_p + r$, $r > 0$, to wzór powyższy zachodzi dla $m > r$.

Twierdzenie powyższe mówi, że jeśli predyktor jest dostatecznie dokładny (tj. jego rząd jest nie mniejszy od rzędu korektora), to dla dostatecznie małych wartości kroku h, tzn. gdy część główna błędu jest dominująca, uzyskanie maksymalnego rzędu (tj. rzędu metody korektora) następuje w algorytmie PK już po jednej iteracji korektora (jednym kroku metody iteracji prostej rozwiązującej nieliniowe równanie korektora (7.60)). Natomiast jeśli predyktor jest mniej dokładny, to do uzyskania maksymalnego rzędu potrzeba więcej iteracji korektora.

7.2.5. Metody predyktor-korektor ze zmiennym krokiem*

Oszacowania błędu aproksymacji metod PK

Zgodnie z twierdzeniem o rzędzie metody PK, jeśli tylko rząd predyktora p_p jest nie niższy od rzędu korektora p, to rząd metody PK jest rzędem korektora p. Przyjmując za podstawę oszacowania błędu aproksymacji $\delta_n(h)$ metody PK część główną błędu korektora,

$$\delta_n(h) = c^*_{p+1} h^{p+1} y^{(p+1)}(x_n), \tag{7.62}$$

trzeba jedynie oszacować pochodną rozwiązania $y^{(p+1)}(x_n)$, np. przybliżając ją odpowiednią różnicą wsteczną (zob. rozdz. 5.1.2):

$$y^{(p+1)}(x_n) \approx \frac{\bigtriangledown^{(p+1)} y(x_n)}{h^{p+1}},$$

skąd

$$h^{p+1} y^{(p+1)}(x_n) \approx \bigtriangledown^{(p+1)} y(x_n) \approx \bigtriangledown^{(p+1)} y_n, \tag{7.63}$$

gdzie różnice wsteczne liczone są na podstawie wartości $y_n, y_{n-1}, \ldots, y_{n-k}$, tzn.

$$\begin{aligned}
\bigtriangledown y_n &= y_n - y_{n-1},\\
\bigtriangledown^2 y_n &= \bigtriangledown y_n - \bigtriangledown y_{n-1},\\
\bigtriangledown^3 y_n &= \bigtriangledown^2 y_n - \bigtriangledown^2 y_{n-1},\\
&\text{itd.}
\end{aligned}$$

Dostajemy więc oszacowanie błędu

$$|\delta_n(h)| \approx |c^*_{p+1} \bigtriangledown^{p+1} y_n|, \tag{7.64}$$

gdzie c^*_{p+1} to stała błędu korektora. Oszacowanie (7.64) można obliczać na podstawie wartość różnicy wstecznej $\bigtriangledown^{p+1} y_n$ wyznaczanej w oparciu o wartości y_n,

*Materiał uzupełniający.

$y_{n-1}, \ldots, y_{n-p-1}$. Istnieje jednak efektywniejszy sposób szacowania błędu aproksymacji w przypadku równych rzędów predyktora i korektora.

Rozważmy metodę PK *z równym rzędem p predyktora i korektora.* Zgodnie z twierdzeniem o rzędzie metody PK rząd ten jest też równy p. Oznaczając przez $y_n^{[0]}$ wynik działania predyktora, a przez y_n wynik po korekcji algorytmem korektora (y_n oznacza ogólnie rezultat m iteracji korektora, $y_n = y_n^{[m]}$, $m = 1$ lub $m > 1$), możemy napisać

$$y_n^{[0]} - y(x_n) = c_{p+1} y^{(p+1)}(x_n) h^{p+1} + O(h^{p+2}),$$
$$y_n - y(x_n) = c_{p+1}^* y^{(p+1)}(x_n) h^{p+1} + O(h^{p+2}).$$

Zaniedbując oba człony $O(h^{p+2})$, eliminujemy z dwóch powyższych równań pochodną $y^{(p+1)}(x_n)$. Stąd

$$y_n - y(x_n) = \frac{c_{p+1}^*}{c_{p+1}} (y_n^{[0]} - y(x_n)).$$

Po dodaniu do obu stron $-\frac{c_{p+1}^*}{c_{p+1}}(y_n - y(x_n))$, uzyskujemy równość

$$\frac{c_{p+1} - c_{p+1}^*}{c_{p+1}} (y_n - y(x_n)) = \frac{c_{p+1}^*}{c_{p+1}} (y_n^{[0]} - y_n),$$

którą możemy przepisać w równoważnej postaci

$$y_n - y(x_n) = \frac{c_{p+1}^*}{c_{p+1} - c_{p+1}^*} (y_n^{[0]} - y_n). \tag{7.65}$$

Równość ta określa nam już estymatę błędu aproksymacji (określoną z dokładnością do pominiętych członów $O(h^{p+2})$):

$$\delta_n (h_{n-1}) = \frac{c_{p+1}^*}{c_{p+1} - c_{p+1}^*} (y_n^{[0]} - y_n), \tag{7.66}$$

którą bardzo łatwo policzyć, korzystając z wartości $y_n^{[0]}$ i y_n. Na przykład, dla metody PK Adamsa $\mathrm{P_4EK_3E}$ z 4-etapowym predyktorem Adamsa-Bashfortha i 3-etapowym korektorem Adamsa-Moultona, uzyskujemy (stałe błędu podane są w tablicach 7.8 i 7.9, obie metody są rzędu 4)

$$y_n - y(x_n) = -\frac{19}{270} (y_n^{[0]} - y_n).$$

Dla metod $\mathrm{P}_k\mathrm{EK}_k\mathrm{E}$ Adamsa, gdzie rząd predyktora jest niższy o jeden od rzędu korektora, mamy dwie możliwości postępowania:

1. Zamiast metody $\mathrm{P}_k\mathrm{EK}_k\mathrm{E}$ stosujemy metodę $\mathrm{P}_k(\mathrm{EK}_k)^2\mathrm{E}$, tzn. wykonujemy dwie iteracje korektora w każdym kroku ($m = 2$, zob. twierdzenie 7.3 o rzędzie metody PK). Pozwala nam to na zachowanie rzędu metody PK równego rzędowi k-krokowego korektora – ale nie możemy stosować efektywnego sposobu szacowania błędu opartego na zależności (7.66). Pozostaje mniej efektywny sposób oparty na wykorzystaniu ogólnej zależności (7.64).
2. Wykonujemy wstępny krok i szacujemy błąd tak jak w metodzie $\mathrm{P}_k\mathrm{EK}_{k-1}\mathrm{E}$, tj. dla $(k-1)$-krokowego korektora – rzędy predyktora i korektora są wówczas równe. Natomiast już po oszacowaniu i akceptacji błędu krok finalny korektora możemy wykonać metodą k-krokową jako dokładniejszą od $(k-1)$-krokowej, co powinno nam tylko poprawić jakość oszacowania błędu (krok metodą niejawną Adamsa rzędu $k+1$ powinien być dokładniejszy niż metodą rzędu k). Ten sposób postępowania będzie szczególnie efektywny, jeśli przejście od wartości obliczonych $(k-1)$-krokowym korektorem do wartości wyznaczanych korektorem k-krokowym będzie proste, wymagające niewielu obliczeń. Jest tak, jeśli implementujemy metody Adamsa w tzw. *wersji różnicowej.*

Metody Adamsa w wersji różnicowej

Stosując zamiast wzoru interpolacyjnego Lagrange'a wzór interpolacyjny Newtona (zob. rozdz. 5.1), dostajemy *równoważną postać wzoru metod niejawnych Adamsa* (metod Adamsa-Bashfortha), wykorzystującą nie wartości funkcji prawych stron równań różniczkowych, ale ich różnice wsteczne:

$$y_n = y_{n-1} + h \cdot \sum_{j=0}^{k-1} \gamma_j \cdot \nabla^j f_{n-1}, \tag{7.67}$$

gdzie $f_{n-j} \stackrel{\mathrm{df}}{=} f\left(x_{n-j}, y_{n-j}\right)$, a różnice wsteczne liczone są na podstawie wartości $f_{n-1}, f_{n-2}, \ldots, f_{n-k}$,

$$\begin{aligned} \nabla f_{n-1} &= f_{n-1} - f_{n-2}, \\ \nabla^2 f_{n-1} &= \nabla f_{n-1} - \nabla f_{n-2}, \\ \nabla^3 f_{n-1} &= \nabla^2 f_{n-1} - \nabla^2 f_{n-2}, \\ &\mathrm{itd}. \end{aligned}$$

Podobnie uzyskuje się *równoważną postać wzoru dla metod Adamsa-Moultona* wykorzystującą różnice wsteczne:

$$y_n = y_{n-1} + h \cdot \sum_{j=0}^{k} \gamma_j^* \cdot \nabla^j f_n, \tag{7.68}$$

gdzie różnice wsteczne liczone są na podstawie wartości $f_n \stackrel{\mathrm{df}}{=} f_n(x_n, y_n), f_{n-1}, f_{n-2}, \ldots, f_{n-k}$.

Wartości parametrów γ_j i γ_j^* wyznaczone dla metod Adamsa w postaci wzorów iteracyjnych z różnicami wstecznymi (7.67) i (7.68), podane są w tabeli 7.11. Stałe błędu związane są jednoznacznie nie tylko ze współczynnikami α_j i β_j, ale oczywiście też ze współczynnikami γ_j. Można pokazać, zob. np. [7], że $c_{k+1} = -\gamma_k$, $c_{k+1}^* = -\gamma_k^*$ (por. z (7.49)).

Tabela 7.11. Wartości parametrów metod Adamsa w wersji różnicowej

j	0	1	2	3	4	5	6	7
γ_j	1	$\frac{1}{2}$	$\frac{5}{12}$	$\frac{3}{8}$	$\frac{251}{720}$	$\frac{95}{288}$	$\frac{19087}{60480}$	$\frac{36799}{120960}$
γ_j^*	1	$-\frac{1}{2}$	$-\frac{1}{12}$	$-\frac{1}{24}$	$-\frac{19}{720}$	$-\frac{3}{160}$	$-\frac{863}{60480}$	$-\frac{1375}{120960}$

Mając metody jawne i niejawne Adamsa w postaci różnicowej, można algorytm predyktor-korektor Adamsa $\mathrm{P}_k\mathrm{EK}_{k-1}\mathrm{E}$ zapisać w *równoważnej postaci różnicowej*:

$$\mathrm{P}: \qquad y_n^{[0]} = y_{n-1} + h\sum_{j=0}^{k-1} \gamma_j \bigtriangledown^j f_{n-1}, \tag{7.69a}$$

$$\mathrm{E}: \qquad f_n^{[0]} = f(x_n, y_n^{[0]}), \tag{7.69b}$$

$$\mathrm{K}: \qquad y_n = y_{n-1} + h\sum_{j=0}^{k-1} \gamma_j^* \nabla^j f_n^{[0]}, \tag{7.69c}$$

$$\mathrm{E}: \qquad f_n = f\left(x_n, y_n\right), \tag{7.69d}$$

gdzie $\bigtriangledown^j f_{n-1}$, $j = 0, 1, ..., k-1$, są różnicami wstecznymi określonymi na podstawie wartości $f_{n-1}, ..., f_{n-k}$, natomiast $\nabla^j f_n^{[0]}$, $j = 0, 1, ..., k-1$, są różnicami wstecznymi określonymi na podstawie wartości $f_n^{[0]}, f_{n-1}, ..., f_{n-k+1}$. Ponieważ w przypadku tym predyktor i korektor są tego samego rzędu, to do szacowania błędu aproksymacji można zastosować wzór (7.66), który po zastosowaniu parametrów wersji różnicowych przyjmuje postać

$$\delta_n\left(h_{n-1}\right) = \frac{\gamma_k^*}{\gamma_k - \gamma_k^*}(y_n^{[0]} - y_n). \tag{7.70}$$

Uwzględniając zależność (zob. np. tabela 7.11)

$$\gamma_j^* = \gamma_j - \gamma_{j-1},$$

można podaną wyżej podstawową postać różnicową metody PK Adamsa $\mathrm{P}_k\mathrm{EK}_{k-1}\mathrm{E}$ przekształcić do prostszej postaci równoważnej, z algorytmem korektora:

$$\mathrm{K}: \qquad y_n = y_n^{[0]} + h\gamma_{k-1}\nabla^k f_n^{[0]}, \tag{7.71}$$

oraz wzorem (7.70) na szacowanie błędu aproksymacji w postaci

$$\delta_n\left(h_{n-1}\right) = -h\gamma_k^*\nabla^k f_n^{[0]}, \tag{7.72}$$

gdzie $\nabla^k f_n^{[0]}$ jest różnicą wsteczną określoną na podstawie wartości $f_n^{[0]},\ f_{n-1}, \ldots,$ f_{n-k}, zob. [7]. Zwróćmy uwagę, że różnicę tę można obliczyć bezpośrednio po obliczeniu wartości predyktora – nie dokonując iteracji korektora, jeśli uzyskana dokładność jest zbyt mała. Jednak oszczędność ta jest w istocie niewielka, gdyż mając obliczone $y_n^{[0]}$ oraz $\nabla^k f_n^{[0]}$, wykonanie iteracji korektora wg wzoru (7.71) jest bardzo proste.

Zależność (7.72) można również bardzo prosto przekształcić do postaci

$$\delta_n\left(h_{n-1}\right) = y_n - y_n^{(+1)}, \tag{7.73}$$

gdzie y_n jest wynikiem kroku algorytmem $\mathrm{P}_k\mathrm{EK}_{k-1}\mathrm{E}$ Adamsa (7.69a)-(7.69d), a $y_n^{(+1)}$ jest wynikiem kroku algorytmem $\mathrm{P}_k\mathrm{EK}_k\mathrm{E}$ Adamsa:

$$\mathrm{P}: \quad y_n^{[0]} = y_{n-1} + h\sum_{j=0}^{k-1}\gamma_j \bigtriangledown^j f_{n-1}, \tag{7.74a}$$

$$\mathrm{E}: \quad f_n^{[0]} = f(x_n, y_n^{[0]}), \tag{7.74b}$$

$$\mathrm{K}: \quad y_n^{(+1)} = y_{n-1} + h\sum_{j=0}^{k}\gamma_j^*\nabla^j f_n^{[0]}, \tag{7.74c}$$

$$\mathrm{E}: \quad f_n = f\left(x_n, y_n\right). \tag{7.74d}$$

Zależność (7.73) uzyskujemy, odejmując od siebie wzory korektora (7.69c) i (7.74c) oraz porównując wynik z (7.72).

Stosowanie oszacowania błędu algorytmu $\mathrm{P}_k\mathrm{EK}_{k-1}\mathrm{E}$ do algorytmu $\mathrm{P}_k\mathrm{EK}_k\mathrm{E}$, tj. dokładniejszego, jedynie może zwiększyć wiarygodność oszacowania błędu. Stąd, stosując algorytm $\mathrm{P}_k\mathrm{EK}_k\mathrm{E}$ z oszacowaniem błędu dla $\mathrm{P}_k\mathrm{EK}_{k-1}\mathrm{E}$, możemy bardzo efektywnie (bez dodatkowego nakładu obliczeń) wyznaczać to oszacowanie z wzoru (7.73). Przyczyną tego jest fakt, że przy stosowaniu wersji różnicowych, obliczając wartości rozwiązania korektorem k-krokowym, niejako „po drodze" liczymy wartości korektorem $(k-1)$-krokowym.

Zmienny krok i rząd całkowania

Konstruując algorytm wielokrokowy *ze zmienną długością kroku*, proponowaną długość nowego kroku wyznaczamy identycznie jak w metodach jednokrokowych, zob. rozdz. 7.1.3,

$$h_n = s \cdot \alpha \cdot h_{n-1}. \tag{7.75}$$

We wzorze tym współczynnik α dany jest zależnością

$$\alpha = \left[\frac{\varepsilon}{|\delta_n (h_{n-1})|} \right]^{\frac{1}{p+1}}, \tag{7.76}$$

gdzie $|\delta_n (h_{n-1})|$ jest oszacowaniem części głównej błędu aproksymacji (w iteracji z punktu x_{n-1} do x_n, z krokiem h_{n-1}), s jest współczynnikiem bezpieczeństwa, zaś ε jest parametrem wymaganej dokładnośći,

$$\varepsilon = |y_{n-1}| \cdot \varepsilon_w + \varepsilon_b. \tag{7.77}$$

W omawianych w niniejszym rozdziale metodach wielokrokowych opartych na równoodległych punktach trudniej jest dokonywać zmiany długości kroku niż w metodach jednokrokowych.

Postępowanie przy zwiększaniu długości kroku, tzn. gdy $s\alpha > 1$ i nowy punkt aktualny x_n jest zaakceptowany, polega na wykonaniu kroku z nowo wyznaczoną długością $h_n = s\alpha h_{n-1}$, dla którego musimy mieć wyznaczone wartości rozwiązania i prawych stron układu równań (tzn. wartości y i $f(x, y)$) w punktach $x = x_n - j(s\alpha h_{n-1})$, $j = 1, ..., k+1$. Jest to $k-1$ *nowych, (zmienionych) punktów* $x_{n-1}, ..., x_{n-k+1}$, *równo odległych o nowy krok* h_n, *a nie o poprzedni krok* h_{n-1}. Jedynie w punkcie x_n wartość i pochodną mamy, bo to ostatni nowo wyznaczony punkt, z niego wystartujemy do następnego kroku algorytmu. Natomiast dotychczas wyznaczone punkty rozwiązania są po zmianie długości kroku w innych niż wielokrotność h_n odległościach od x_n.

Ogólną regułą postępowania w takiej sytuacji jest (przy zwiększaniu kroku): *Korzystając z dotychczas obliczonych wartości rozwiązania, wyznaczamy funkcję interpolującą poszukiwane rozwiązanie na przedziale* $[x_{n-l}, x_n]$ *obejmującym potrzebne do wyznaczenia nowe punkty* $x_{n-j} = x_{n-1} - jh_n$, $j = 1, ..., k+1$, *a następnie obliczamy wartości funkcji* $f(x_{n-j}, y_{n-j})$ *prawych stron układu równań odpowiadających tym nowym punktom i wartościom* y_{n-j} *funkcji interpolującej w tych punktach.*

Ponadto, ponieważ taka korekta długości kroku nieco kosztuje, to można decydować się na nią z pewną rezerwą, wprowadzając odpowiednie mechanizmy ograniczające zbytnią częstotliwość zmian długości kroku (np. „nie zwiększaj długości kroku, jeśli proponowana zmiana jest zbyt mała, a w poprzedniej iteracji krok był zmniejszany").

Postępowanie przy zmniejszaniu długości kroku (a więc po kroku nieudanym, niespełniającym wymagań dokładności) jest analogiczne, tylko punktem ostatnim jest nie nowo wyznaczony punkt x_n, a poprzedni punkt x_{n-1} („cofamy się" do niego), natomiast interpolację możemy zawsze wykonać na ostatnich (niezmienionych) k punktach.

Dla metod Adamsa potrzebne są poprzednie wartości jedynie funkcji prawych stron układu równań $f_{n-j} = f(x_{n-j}, y_{n-j})$, stąd efektywniej jest interpolo-

wać od razu te wartości. Można tu zastosować wielomian interpolacyjny Newtona „wstecz" (5.17), zob. rozdz. 5.1.2.

W metodach wielokrokowych obok zmiany długości kroku stosuje się *zmiany rzędu metody* wielokrokowej PK do sterowania dokładnością obliczeń. Zmiany rzędu są szczególnie wygodne przy stosowaniu metody PK Adamsa w postaci różnicowej. Zwiększanie rzędu powinno prowadzić do zmniejszania błędu, a więc zwiększania dokładności obliczeń. Należy jednak zaznaczyć, że przynajmniej teoretycznie, zwiększenie rzędu nie zawsze musi spowodować zmniejszenie oszacowania błędu. W ogólnym wzorze na oszacowanie błędu występuje iloczyn stałej błędu i różnicy wstecznej, $c^*_{p+1} \cdot \bigtriangledown^{p+1} y_n$, gdzie różnica wsteczna jest oszacowaniem iloczynu $h^{p+1} y^{(p+1)}(x_n)$. Teoretycznie istnieją funkcje, np. $\exp(\beta x),\ \beta > 1$, dla których wartość pochodnej wzrasta wraz z jej rzędem. Jeśli ten wzrost jest szybszy niż malenie, wraz ze wzrostem rzędu p, czynnika $c^*_{p+1} h^{p+1}$, to możliwa jest sytuacja wzrostu błędu wraz ze wzrostem p (a więc i malenia błędu z maleniem p, symetrycznie).

Metody ze stałym krokiem mogą być wygodne do stosowania np. w programach symulacji dynamiki układów. Natomiast *efektywne, profesjonalne implementacje metod Adamsa wykorzystują znacznie bardziej złożone algorytmicznie wersje z nierówno odległymi punktami (zob. np. [7]), z bieżącymi zmianami zarówno długości kroku, jak i rzędu metody. Metody te są samostartujące, zmiana długości kroku i rzędu jest w nich z kroku na krok łatwa. Są to jedne z najlepszych aktualnie znanych algorytmów w swojej klasie.*

7.3. Równania źle uwarunkowane

Rozważmy przykład:

$$\mathbf{y}'(x) = \mathbf{A}\mathbf{y}(\mathbf{x}),$$

$$\mathbf{A} = \begin{bmatrix} -667 & 333 \\ 666 & -334 \end{bmatrix}, \quad \mathbf{y}_0 = \begin{bmatrix} 0 \\ 3 \end{bmatrix}, \quad x \in [0, 10]\,.$$

Jak łatwo wyliczyć

$$\mathrm{sp}\,(\mathbf{A}) = \{\lambda_1, \lambda_2\} = \{-1, -1000\}\,,$$

zaś rozwiązanie jest w postaci

$$y_1(x) = e^{-x} - e^{-1000x},$$
$$y_2(x) = 2e^{-x} + e^{-1000x}.$$

Rozwiązanie jest szybkozmienne w krótkiej początkowej fazie (gdzie dominuje e^{-1000x}), a następnie bardzo wolnozmienne. Efektywna procedura powinna cał-

kować pierwszy odcinek z krokiem 1000 razy mniejszym (stosunek wartości własnych). Na przykład, dla metody P_3EK_3E z warunku absolutnej stabilności mamy (odcinek absolutnej stabilności wynosi $[-2, 0]$):

$$|\lambda_1 h| < 2 \Rightarrow h < \frac{2}{|\lambda_1|} = 2,$$

$$|\lambda_2 h| < 2 \Rightarrow h < \frac{2}{|\lambda_2|} = 2 \cdot 10^{-3}.$$

Procedura całkuje najpierw z krokiem mniejszym od $2 \cdot 10^{-3}$, którego potem nie może skutecznie zwiększyć, gdyż wychodzi poza obszar absolutnej stabilności – mimo że należałoby całkować dalej z krokiem 1000 razy większym. Algorytm usiłuje zwiększać krok (tak mu podpowiada szacowanie błędów), ale stając się niestabilnym, generuje związaną z tym lawinę błędów – i musi skracać krok aż wejdzie w obszar absolutnej stabilności.

Mówimy, że układ liniowych równań różniczkowych $\mathbf{y}' = \mathbf{A}\mathbf{y}$ jest *źle uwarunkowany („sztywny”, stiff*), jeśli

$$\frac{|\lambda_{\max}|}{|\lambda_{\min}|} \ggg 1.$$

gdzie $\lambda_{\min}$ i $\lambda_{\max}$ to najmniejsza i największa (co do modułu) wartość własna macierzy $\mathbf{A}$. W przypadku nieliniowego układu równań bierzemy pod uwagę macierz jego przybliżenia liniowego (macierz jakobianową, tj. macierz pierwszych pochodnych funkcji prawych stron układu równań).

Dla układów źle uwarunkowanych należy stosować metody numeryczne o bardzo dużych, nieskończonych obszarach stabilności absolutnej. Nie są takimi omawiane wcześniej metody predyktor-korektor Adamsa czy metody Runge-Kutty (obszary absolutnej stabilności omawianych metod RK są tego samego rzędu co metod Adamsa, zob. np. [7]).

Przykład 7.6. Dla *metody Eulera* (metoda jawna) mamy:

$$y_n = y_{n-1} + h \cdot f\left(x_{n-1}, y_{n-1}\right).$$

Dla równania

$$y' = \lambda y, \ (\lambda < 0) \tag{7.78}$$

daje to algorytm

$$y_n = y_{n-1} + h\lambda y_{n-1} = (1 + h\lambda)\, y_{n-1},$$

tzn.

$$y_n = (1 + h\lambda)^n\, y_0.$$

Metoda jest absolutnie stabilna, tzn. $(1 + h\lambda)^n \to 0$, gdy $|1 + h\lambda| < 1$, skąd

$$h\,|\lambda| < 2, \text{ tj. } h < \frac{2}{|\lambda|}.$$

□

Przykład 7.7. Dla tzw. *wstecznej metody Eulera* (metoda niejawna)

$$y_n = y_{n-1} + h \cdot y\left(x_n, y_n\right)$$

i dla równania różniczkowego (7.78) mamy:

$$y_n = y_{n-1} + h\lambda y_n,$$

skąd

$$y_n = \frac{1}{1-\lambda h} y_{n-1}.$$

Ponieważ

$$\forall \lambda < 0 \quad \frac{1}{1-\lambda h} < 1,$$

więc metoda wsteczna Eulera jest absolutnie stabilna w całej lewej półpłaszczyźnie (dla każdego $h\lambda < 0$). □

Można skonstruować metody o wyższej krokowości (większych liczbach k poprzednich punktów uwzględnianych w algorytmie) i wyższym rzędzie, i równocześnie o dużych obszarach absolutnej stabilności. Przykładem są metody typu BDF podane poniżej.

Metody BDF (wstecznego różniczkowania)

Metody BDF (*Backward Differentiation Formulae*) *jawne* zdefiniowane są wzorem

$$y_n = \sum_{j=1}^{k} \alpha_j y_{n-j} + h \cdot \beta_1 f\left(x_{n-1}, y_{n-1}\right). \tag{7.79}$$

Natomiast metody BDF *niejawne* dane sa zależnością

$$y_n = \sum_{j=1}^{k} \alpha_j^* y_{n-j} + h \cdot \beta_0^* f\left(x_n, y_n\right). \tag{7.80}$$

Parametry metod BDF podano w tabelach 7.12 i 7.13. Ponieważ nieskończony obszar absolutnej stabilności mają jedynie metody niejawne, więc stosuje się jedynie strukturę predyktor-korektor (PK). Wykonujemy przy tym nie jedną iterację korektora, ale tyle iteracji, ile potrzeba aż do uzyskania zbieżności wg określonego kryterium stopu. Iteracje korektora, tzn. rozwiązywanie równania nieliniowego korektora, wykonywane są metodą iteracji prostej, możliwe i polecane jest też zastosowanie efektywniejszego algorytmu Newtona.

Kształt obszarów absolutnej stabilności metod BDF niejawnych pokazano na rysunku 7.8. Metody BDF stosowane w strukturze PK mają oczywiście takie same

Tabela 7.12. Parametry metod jawnych BDF

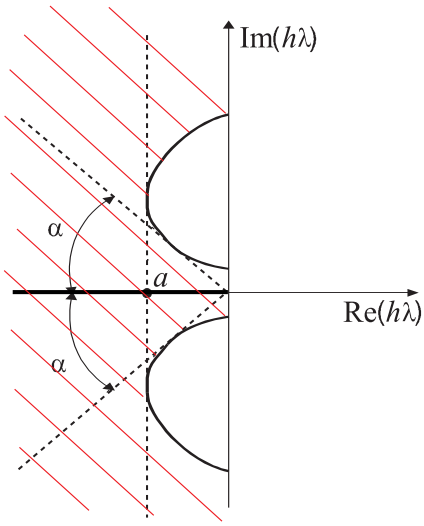
c_{p+1}	p	k	β_1	α_1	α_2	α_3	α_4	α_5	α_6
$-\frac{1}{2}$	1	1	1	1					
$-\frac{1}{3}$	2	2	2	0	1				
$-\frac{1}{4}$	3	3	3	$-\frac{3}{2}$	3	$-\frac{1}{2}$			
$-\frac{1}{5}$	4	4	4	$-\frac{10}{3}$	6	-2	$\frac{1}{3}$		
$-\frac{1}{6}$	5	5	5	$-\frac{65}{12}$	10	-5	$\frac{5}{3}$	$-\frac{1}{4}$	
$-\frac{1}{7}$	6	6	6	$-\frac{77}{10}$	15	-10	5	$-\frac{3}{2}$	$\frac{1}{5}$

Tabela 7.13. Parametry metod niejawnych BDF

c_{p+1}	p	k	β_0^*	α_1^*	α_2^*	α_3^*	α_4^*	α_5^*	α_6^*
$\frac{1}{2}$	1	1	1	1					
$\frac{2}{9}$	2	2	$\frac{2}{3}$	$\frac{4}{3}$	$-\frac{1}{3}$				
$\frac{3}{22}$	3	3	$\frac{6}{11}$	$\frac{18}{11}$	$-\frac{9}{11}$	$\frac{2}{11}$			
$\frac{12}{125}$	4	4	$\frac{12}{25}$	$\frac{48}{25}$	$-\frac{36}{25}$	$\frac{16}{25}$	$-\frac{3}{25}$		
$\frac{10}{137}$	5	5	$\frac{60}{137}$	$\frac{300}{137}$	$-\frac{300}{137}$	$\frac{200}{137}$	$-\frac{75}{137}$	$\frac{12}{137}$	
$\frac{20}{343}$	6	6	$\frac{60}{147}$	$\frac{360}{147}$	$-\frac{450}{147}$	$\frac{400}{147}$	$-\frac{225}{147}$	$\frac{72}{147}$	$-\frac{10}{147}$

obszary absolutnej stabilności, ze względu na iterowanie korektora do zbieżności – iteracja predyktora wyznacza tylko punkt startowy.

Definicja absolutnej stabilności odnosi się do wszystkich różnicowych metod numerycznych całkowania układów równań różniczkowych, w tym do metod jednokrokowych przedstawionych w rozdziale 7.1. Przedstawione tam metody jednokrokowe RK i RKF są metodami jawnymi, o ograniczonych obszarach absolutnej stabilności, stąd nie są efektywne dla źle uwarunkowanych układów równań różniczkowych. Dla takich układów opracowano także *niejawne metody jednokrokowe (typu Rungego-Kutty)*, o dużych obszarach absolutnej stabilności. Najprostszym przykładem takiej metody jest niejawna metoda Eulera rozważona uprzednio w tym rozdziale.



k	a	α
1	0.0	90°
2	0.0	90°
3	0.1	88°
4	0.7	73°
5	2.4	51°
6	6.1	18°

Rys. 7.8. Kształt obszarów absolutnej stabilności metod BDF niejawnych

Zadania

1. Zbadaj zbieżność oraz określ rząd i stałą błędu metody

$$y_n = \frac{9y_{n-1} - y_{n-3}}{8} + 3h\frac{f_n + 2f_{n-1} - f_{n-2}}{8},$$

gdzie $f_i = f\,(x_i, y_i)$.

2. Dany jest wielomian $\rho(z) = -z^2+1$. Dobierz tak wielomian $\sigma(z)$, aby uzyskać metodę jawną dwukrokową ($k = 2$) stabilną, maksymalnego rzędu.

3. Co można powiedzieć o absolutnej stabilności metod jawnych dwukrokowych maksymalnego rzędu? (por. przykład w rozdz. 7.2.3)

4. Zbadaj stabilność metod jawnych BDF jedno-, dwu- i trzykrokowej.

5. Zbadaj, dla jakiego zakresu wartości kroku h stabilne będzie zastosowanie metody dwukrokowej jawnej Adamsa do zadania

$$y\,'(x) = -5\,y(x).$$

6. Układ równań opisujący proces typu „drapieżcy-ofiary” ma postać:

$$y_1'(x) = a\,y_1(x) - b\,y_1(x)\,y_2(x),$$
$$y_2'(x) = c\,y_1(x)\,y_2(x) - d\,y_2(x),$$

gdzie zmienna $y_1(x)$ opisuje liczność populacji ofiar, a $y_2(x)$ liczność populacji drapieżców.

Przyjmując wartości parametrów: $a = 10$, $b = 0.2$, $c = 0.02$, $d = 5$ oraz warunki początkowe: $y_1(0) = 200$, $y_2(0) = 10$, należy wyznaczyć przebiegi zmiennych $y_1(x)$ i $y_2(x)$ dla $x \in [0,\ 4]$, stosując:

a) metodę RK4 ze stałym krokiem, dobierając wartość kroku metodą prób, tzn. powtarzając obliczenia z coraz mniejszym krokiem, aż do znalezienia takiej jego wartości, której dalsze zmniejszanie nie wpływa istotnie na rozwiązanie (np. wykresy zmiennych optycznie nierozróżnialne);

b) metodą RK4 ze zmiennym krokiem, dobieranym automatycznie przy zastosowaniu szacowania błędu metodą podwójnego kroku.

Zastosowane algorytmy zaimplementować w środowisku MATLAB, wykreślając przebiegi zmiennych. Sprawdzić poprawność rezultatów, wykorzystując jedną z dostępnych w tym środowisku procedur rozwiązywania układów równań różniczkowych.

7. Wykonaj zadanie poprzednie, ale zastosuj metodę RKF45 zamiast RK4, w punkcie b) szacując błąd metodą właściwą dla metod RKF.

8. Wykonaj zadanie 6a, ale zamiast metody RK4 stosując czterokrokową metodę PK Adamsa P_4EK_4E.

9.* Wykonaj zadanie 6b, ale zamiast metody RK4 zastosuj czterokrokową metodę PK Adamsa P_4EK_4E. Szacowanie błędu wykonaj tak jak dla metody P_4EK_3E, na podstawie różnic wartości wyznaczanych wzorami predyktora i korektora.

*Zadanie dodatkowe.

Rozdział 8

Różniczkowanie i całkowanie numeryczne

8.1. Różniczkowanie numeryczne

Metody numerycznego różniczkowania funkcji, czyli formuły aproksymacji numerycznej pochodnych, są istotne dla wielu algorytmów obliczeniowych, m.in. służących do: wyznaczania prędkości i przyspieszenia dla znanych funkcji położenia (np. w robotyce), numerycznego wyznaczania pochodnych w algorytmach optymalizacji funkcji i w algorytmach rozwiązywania układów równań nieliniowych, formułowania dyskretnych odpowiedników algorytmów sterowania z czasem ciągłym, analizy wrażliwościowej (modeli, praw sterowania, rozwiązań optymalnych, itp.), rozwiązywania układów równań różniczkowych zwyczajnych czy cząstkowych, i wielu innych problemów.

Najprostsze aproksymacje pochodnej (pierwszego rzędu) funkcji uzyskujemy na podstawie ilorazu różnicowego, przy czym możliwe są różne usytuowania punktów definiujących ten iloraz, prowadzące do aproksymacji przez:

1. iloraz różnicowy wsteczny (oparty na występującej w liczniku różnicy wstecznej – *backward difference*)

$$f'(x) \approx \frac{f(x) - f(x-h)}{h} = D_{f-}(x,h), \tag{8.1}$$

2. iloraz różnicowy zwykły (zwany też progresywnym, oparty na różnicy zwykłej, progresywnej – *forward difference*)

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} = D_{f+}(x,h), \tag{8.2}$$

3. iloraz różnicowy centralny (oparty na różnicy centralnej – *central difference*)

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} = D_{fc}(x,h), \tag{8.3}$$

gdzie h jest krokiem aproksymacji, tzn. odległością punktu testowego (punktów testowych) od punktu, w którym aproksymujemy pochodną. Dla uproszczenia, często mówimy o aproksymacji pochodnej ilorazem różnicowym centralnym, zwykłym

(progresywnym) czy wstecznym jako o aproksymacji różnicą centralną, zwykłą (progresywną) czy wsteczną.

Jak można się spodziewać, różnica centralna prowadzi do najmniejszych błędów (ale nie zawsze może być stosowana). Błąd aproksymacji można wyznaczyć na podstawie rozwinięć w szereg Taylora. Dla różnicy wstecznej mamy

$$f(x-h) = f(x) - f^{'}(x)h + \frac{f^{(2)}(\alpha)}{2!}h^2, \tag{8.4}$$

gdzie $\alpha \in [x-h, x]$. Stąd

$$f^{'}(x) = \frac{f(x) - f(x-h)}{h} + \frac{f^{(2)}(\alpha)}{2!}h = D_{f-}(x,h) - O(h), \tag{8.5}$$

gdzie $O(h) = \frac{f^{(2)}(\alpha)}{2!}h$ jest błędem metody aproksymacji (błędem obcięcia). Podobny rezultat otrzymamy dla różnicy zwykłej. Natomiast dla różnicy centralnej mamy (zakładając $f \in C^3$):

$$\begin{aligned} f(x+h) &= f(x) + f^{'}(x)h + \frac{f^{(2)}(x)}{2!}h^2 + \frac{f^{(3)}(\alpha_+)}{3!}h^3, \\ f(x-h) &= f(x) - f^{'}(x)h + \frac{f^{(2)}(x)}{2!}h^2 - \frac{f^{(3)}(\alpha_-)}{3!}h^3, \end{aligned}$$

gdzie $\alpha_+ \in (x, x+h)$, $\alpha_- \in (x-h, x)$. Po odjęciu ostatniego równania od poprzedniego, otrzymujemy

$$f(x+h) - f(x-h) = 2f^{'}(x)h + \frac{f^{(3)}(\alpha_+) + f^{(3)}(\alpha_-)}{3!}h^3, \tag{8.6}$$

skąd dostajemy

$$D_{fc}(x,h) = f^{'}(x) + \frac{f^{(3)}(\alpha_+) + f^{(3)}(\alpha_-)}{12}h^2 = f^{'}(x) + O(h^2). \tag{8.7}$$

Aproksymacja różnicą centralną jest więc o rząd wielkości dokładniejsza od aproksymacji różnicami wsteczną czy zwykłą (przyczyną tego jest jej równoważność z aproksymacją trzypunktową, opartą na punktach $x-h, x, x+h$, zob. zadanie 3). Stąd, jeśli mamy wyznaczać numerycznie pochodną znanej funkcji, to zalecane jest zastosowanie aproksymacji opartej na różnicy centralnej. W wielu praktycznych aplikacjach, np. w zadaniach sterowania, mamy jednakże zadanie bieżącej aproksymacji pochodnej aktualnie mierzonego (czy wyznaczanego na podstawie aktualnych pomiarów) sygnału, a więc nie mamy wartości przyszłych. Wówczas stosujemy aproksymację ilorazem różnicowym wstecznym, lub bardziej złożone i dokładniejsze formuły aproksymacji pochodnej wykorzystujące większą liczbę poprzednich punktów.

Przykład 8.1. Policzymy numerycznie, w środowisku MATLAB (a więc w podwójnej precyzji), pochodną funkcji $\cos(x)$ w punkcie $x = \pi/4$, z krokiem $h = 0.01$, stosując iloraz różnicowy wsteczny i centralny.

Dla ilorazu różnicowego wstecznego mamy ($\tilde{D}$ oznacza numerycznie wyliczoną wartość aproksymacji pochodnej D)

$$\begin{aligned}\tilde{D}_{f-}\Big(\frac{\pi}{4}, 0.01\Big) &= \frac{0.707106781186548 - 0.714142376103440}{0.01} \\ &= -0.703559491689199.\end{aligned}$$

Dla ilorazu różnicowego centralnego mamy

$$\begin{aligned}\tilde{D}_{fc}\Big(\frac{\pi}{4}, 0.01\Big) &= \frac{0.700000476180791 - 0.714142376103440}{0.02} \\ &= -0.707094996132451.\end{aligned}$$

Stąd błędy bezwzględne ($\cos'(\pi/4) = -\sin(\pi/4) = -0.707106781186548$):

$$\tilde{D}_{f-}\Big(\frac{\pi}{4}, 0.01\Big) - \cos'\Big(\frac{\pi}{4}\Big) = 0.003547289497349,$$

$$\tilde{D}_{fc}\Big(\frac{\pi}{4}, 0.01\Big) - \cos'\Big(\frac{\pi}{4}\Big) = 0.00001178505409615838.$$

Obliczenia potwierdzają znacznie lepszą dokładność metody aproksymacji opartej na ilorazie różnicowym centralnym. □

W dotychczasowych rozważaniach teoretycznych braliśmy pod uwagę *jedynie błędy metody*, a przecież występują jeszcze błędy numeryczne reprezentacji liczb i zmiennopozycyjnych operacji arytmetycznych. Zakładając, że krok h jest potęgą dwójki (tzn. jego reprezentacja maszynowa h i dzielenie przez h są dokładne) oraz że moduły błędów reprezentacji liczb $f(x)$ i $f(x-h)$ są nie większe od Eps, tzn. $fl(f(x)) = f(x)(1+\varepsilon_1)$, $fl(f(x-h)) = f(x-h)(1+\varepsilon_2)$, $|\varepsilon_i| \leq Eps,\ i = 1, 2$ ($Eps \geq eps$), mamy dla różnicy wstecznej

$$\begin{aligned}\tilde{D}_{f-}(x,h) &= \frac{[f(x)(1+\varepsilon_1) - f(x-h)(1+\varepsilon_2)](1+\varepsilon_3)}{h} \\ &= \frac{f(x) - f(x-h) + f(x)\varepsilon_1 - f(x-h)\varepsilon_2}{h}(1+\varepsilon_3) \\ &= \frac{f(x) - f(x-h)}{h}\left[1 + \frac{f(x)\varepsilon_1 - f(x-h)\varepsilon_2}{f(x) - f(x-h)}\right](1+\varepsilon_3) \\ &\underset{1}{=} \frac{f(x) - f(x-h)}{h}\left[1 + \frac{f(x)\varepsilon_1 - f(x-h)\varepsilon_2}{f(x) - f(x-h)} + \varepsilon_3\right] \\ &= \frac{f(x) - f(x-h)}{h}\left[1 + \delta(f,x,h)\right],\end{aligned}$$

gdzie $|\varepsilon_3| \leq eps$, a $\tilde{D}_{f+}(x,h)$ oznacza aproksymację pochodnej ilorazem różnicowym wstecznym uzyskaną numerycznie (z błędami reprezentacji liczb i operacji arytmetycznych). Stąd oszacowanie błędu względnego $\delta(f,x,h)$ wynosi

$$\begin{aligned} |\delta(f,x,h)| &= \left| \frac{f(x)\varepsilon_1 - f(x-h)\varepsilon_2}{f(x)-f(x-h)} + \varepsilon_3 \right| \\ &\leq \frac{|f(x)|+|f(x-h)|}{|f(x)-f(x-h)|} Eps + eps \\ &\approx \frac{2|f(x)|}{|f(x)-f(x-h)|} Eps + eps. \end{aligned} \tag{8.8}$$

Ponieważ stosujemy do aproksymacji małe kroki (im mniejszy krok, tym mniejszy błąd metody), więc pierwszy ze składników powyższego błędu względnego numerycznego jest z reguły znacznie większy od drugiego (eps) – i może być bardzo duży, przede wszystkim z powodu bardzo małej liczby w mianowniku ułamka.

Z powyższej analizy wynika, że występuje przeciwstawny wpływ długości kroku h na błąd metody i błąd operacji numerycznych – zmniejszanie h powoduje malenie pierwszego błędu i wzrost drugiego (przez zmniejszanie modułu różnicy $f(x+h)-f(x)$). Stąd można pokusić się o wyznaczanie optymalnego kroku, minimalizującego łączny błąd. W tym celu przedstawmy błąd numeryczny w postaci

$$|\delta(f,x,h)| \approx \frac{2|f(x)|}{|f'(x)|h} Eps + eps. \tag{8.9}$$

Oszacowanie łącznego błędu (bezwzględnego) wyraża się stąd wzorem

$$|\tilde{D}_{f-}(x,h) - f'(x)| \leq \frac{|f^{(2)}(\alpha)|}{2} h + \frac{2}{h}|f(x)|Eps + |f'(x)|eps. \tag{8.10}$$

Funkcja ta ma minimum, warunek konieczny tego minimum (pochodna równa zeru) prowadzi do równania

$$\frac{|f^{(2)}(\alpha)|}{2} - \frac{2}{h^2}|f(x)|Eps = 0,$$

skąd dostajemy optymalną długość kroku, wyrażającą najlepszy kompromis między błędami metody i numerycznym, w postaci

$$\hat{h}_- = 2\sqrt{\frac{|f(x)|}{|f^{(2)}(\alpha)|} Eps}. \tag{8.11}$$

Chcąc zastosować ten wzór do doboru kroku, powinno się oszacować numeryczne przybliżenie drugiej pochodnej (patrz rozważania niżej). Jeśli nawet nie obliczamy

współczynnika występującego przy Eps pod znakiem pierwiastka, to można przyjąć następującą „inżynierską" regułę: *Optymalny krok h powinien być rzędu $\sqrt{Eps}$*, tzn. rzędu pierwiastka z oszacowania błędów względnych reprezentacji wartości funkcji.

Dla aproksymacji pochodnej opartej na różnicach zwykłej i centralnej analizę błędów prowadzi się w analogiczny sposób. Optymalna długość kroku dla różnicy zwykłej jest taka sama jak dla wstecznej, dla różnicy centralnej zależy od pochodnych trzeciego rzędu. Obliczenia te pozostawiamy czytelnikowi (zadania na końcu rozdziału).

Pochodne drugiego rzędu można obliczać, korzystając z rekurencyjnej definicji pochodnej rzędu pierwszego, np. dla aproksymacji opartych na różnicach wstecznej i centralnej mamy:

1. dla różnic wstecznych

$$\begin{aligned} f^{(2)}(x) &\approx \frac{D_{f-}(x,h) - D_{f-}(x-h,h)}{h} \\ &= \frac{\frac{f(x)-f(x-h)}{h} - \frac{f(x-h)-f(x-2h)}{h}}{h} \\ &= \frac{f(x) - 2f(x-h) + f(x-2h)}{h^2} = D^2_{f-}(x,h), \end{aligned}$$

2. dla różnic centralnych

$$\begin{aligned} f^{(2)}(x) &\approx \frac{D_{fc}(x+\frac{h}{2},\frac{h}{2}) - D_{fc}(x-\frac{h}{2},\frac{h}{2})}{h} \\ &= \frac{\frac{f(x+h)-f(x)}{h} - \frac{f(x)-f(x-h)}{h}}{h} \\ &= \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} = D^2_{fc}(x,h). \end{aligned}$$

Analogicznie można wyznaczać aproksymacje pochodnych wyższych rzędów. Należy pamiętać, że już zadanie numerycznego wyznaczania aproksymacji pochodnej pierwszego rzędu jest źle uwarunkowane numerycznie, stąd będzie tak tym bardziej dla pochodnych wyższych rzędów.

Przedstawione wyżej wzory na numeryczne aproksymacje pochodnych pierwszego i drugiego rzędu oparte są na minimalnej liczbie obliczeń wartości funkcji (dla pochodnych pierwszego rzędu – dwa obliczenia; dla pochodnych drugiego rzędu – trzy obliczenia). Można też sformułować wzory wykorzystujące większą liczbę punktów. Można je wyznaczyć jako odpowiednie pochodne wielomianu interpolacyjnego opartego na założonej liczbie punktów – w istocie wszystkie wzory

podane wyżej też można wyprowadzić jako pochodne wielomianów interpolacyjnych opartych na odpowiednio dobranych punktach. Podany poniżej przykład wyjaśnia ten sposób postępowania.

Przykład 8.2. Sformułujemy wielomian interpolacyjny Lagrange'a oparty na trzech równo odległych punktach $(x,\ x-h,\ x-2h)$ i wyznaczymy numeryczną trzypunktową aproksymację pochodnej pierwszego rzędu funkcji interpolowanej w punkcie x_0 jako pochodną tego wielomianu.

Wielomian interpolacyjny Lagrange'a:

$$\begin{aligned} W_2(x) &= f(x_0)\frac{(x-(x_0-h))(x-(x_0-2h))}{(x_0-(x_0-h))(x_0-(x_0-2h))} + \\ &\quad + f(x_0-h)\frac{(x-x_0)(x-(x_0-2h))}{(x_0-h-(x_0))(x_0-h-(x_0-2h))} + \\ &\quad + f(x_0-2h)\frac{(x-x_0)(x-(x_0-h))}{(x_0-2h-(x_0))(x_0-2h-(x_0-h))} \\ &= f(x_0)\frac{(x-x_0+h)(x-x_0+2h)}{2h^2} + \\ &\quad + f(x_0-h)\frac{(x-x_0)(x-x_0+2h)}{-h^2} + \\ &\quad + f(x_0-2h)\frac{(x-x_0)(x-x_0+h)}{2h^2}. \end{aligned}$$

Pochodna tego wielomianu jest postaci

$$\begin{aligned} W_2'(x) &= f(x_0)\frac{(x-x_0+h)+(x-x_0+2h)}{2h^2} + \\ &\quad + f(x_0-h)\frac{(x-x_0)+(x-x_0+2h)}{-h^2} + \\ &\quad + f(x_0-2h)\frac{(x-x_0)+(x-x_0+h)}{2h^2}. \end{aligned}$$

Natomiast wartość tej pochodnej w punkcie $x = x_0$ wynosi

$$\begin{aligned} f'(x_0) &\approx W_2'(x_0) = f(x_0)\frac{(h+2h)}{2h^2} + f(x_0-h)\frac{2h}{-h^2} + f(x_0-2h)\frac{h}{2h^2} \\ &= \frac{3f(x)-4f(x-h)+f(x-2h)}{2h} = D_{f3p-}(x,h). \end{aligned} \tag{8.12}$$

□

Dokładność aproksymacji danej wzorem (8.12) jest tego samego rzędu co podany uprzednio wzór oparty na ilorazie różnicowym centralnym. Łatwo to pokazać, wykorzystując dwa rozwinięcia funkcji $f(x)$ w szereg Taylora (zakładając $f \in C^3$, z resztą szeregu w postaci Lagrange'a):

$$\begin{aligned} f(x-h) &= f(x) - f'(x)h + \frac{f^{(2)}(x)}{2!}h^2 - \frac{f^{(3)}(\alpha_1)}{3!}h^3, \\ f(x-2h) &= f(x) - f'(x)2h + \frac{f^{(2)}(x)}{2!}(2h)^2 - \frac{f^{(3)}(\alpha_2)}{3!}(2h)^3, \end{aligned}$$

gdzie $\alpha_1 \in (x-h, x)$, $\alpha_2 \in (x-2h, x)$. Po pomnożeniu pierwszego z tych równań przez 4 i odęciu od drugiego, dostajemy

$$f'(x)2h = 3f(x) - 4f(x-h) + f(x-2h) + \frac{f^{(3)}(\alpha_2)}{3!}(2h)^3 - 4\frac{f^{(3)}(\alpha_1)}{3!}h^3,$$

skąd bezpośrednio

$$f'(x) = \frac{3f(x) - 4f(x-h) + f(x-2h)}{2h} + O(h^2).$$

Zauważmy, że przedstawione powyżej wnioskowanie jest jednocześnie alternatywnym, wobec metody różniczkowania wielomianu interpolacyjnego, sposobem wyprowadzenia zależności (8.12).

Przykład 8.1. (cd.) Policzymy numerycznie, w środowisku MATLAB, pochodną funkcji $\cos(x)$ w punkcie $x = \pi/4$, z krokiem $h = 0.01$, stosując wzór (8.12). Mamy

$$\tilde{D}_{f3p-}\left(\frac{\pi}{4}, 0.01\right) = -0.707130173813869.$$

Stąd błąd bezwzględny ($\cos'(\pi/4) = -\sin(\pi/4) = -0.707106781186548$) wynosi

$$\tilde{D}_{f3p-}\left(\frac{\pi}{4}, 0.01\right) - \cos'\left(\frac{\pi}{4}\right) = -0.00002339262732165$$

i jest porównywalny z błędem metody dla aproksymacji dwupunktowej ilorazem różnicowym centralnym, a znacznie mniejszy od błędu dla aproksymacji dwupunktowej ilorazem różnicowym wstecznym. □

Trzeba być wyjątkowo ostrożnym przy bieżącym numerycznym wyznaczaniu *pochodnych funkcji bezpośrednio mierzonych i obarczonych błędami pomiaru* czy również innego rodzaju zakłóceniami. Błędy te są bowiem często wielokrotnie większe od błędów numerycznych, a zadanie numerycznej aproksymacji pochodnej jest, jak to pokazaliśmy wyżej, źle uwarunkowane. Jeśli zakłócenia pomiaru zawierają zmienną losową o szybszej zmienności niż sygnał mierzony, to należy ją wstępnie odfiltrować, różniczkując następnie sygnał już odpowiednio wygładzony. Natomiast krok należy dobierać, mając na uwadze zależność (8.11) i regułę podaną po tym wzorze.

8.2. Całkowanie numeryczne

W wielu zastosowaniach występuje zadanie numerycznego obliczenia całki funkcji nieliniowej określonej na pewnym przedziale czy zbiorze (całki wielokrotne). Podstawowe i najprostsze takie zadanie, to policzenie całki funkcji ciągłej $f(x)$ na przedziale domkniętym $[a, b]$:

$$I(f; a, b) = \int_a^b f(x)\, dx. \tag{8.13}$$

Załóżmy, że mamy dany podział odcinka $[a, b]$ $n+1$ punktami $a = x_0 < x_1 < \cdots < x_n = b$ oraz wartości $f(x_i)$ funkcji f w tych punktach. Dysponując taką informacją, chcemy policzyć przybliżoną wartość całki (8.13). Przybliżenie to będzie miało postać

$$Q(f; a, b) = \sum_{i=0}^{n} A_i f(x_i). \tag{8.14}$$

Wzór ten nazywany jest *kwadraturą* (liniową), a punkty x_i *węzłami kwadratury*. Różnicę

$$R(f; a, b) = I(f; a, b) - Q(f; a, b) \tag{8.15}$$

nazywamy *resztą* kwadratury.

Mówimy, że *kwadratura jest rzędu p*, jeśli jest dokładna (tzn. jej reszta jest równa zeru) dla wszystkich wielomianów stopnia mniejszego od p i istnieje co najmniej jeden wielomian stopnia p, dla którego reszta kwadratury jest niezerowa.

W zależności od sposobu wyznaczania współczynników A_i uzyskujemy różne kwadratury. Jedną z podstawowych, w naturalny sposób narzucających się metod wyznaczania kwadratur jest utworzenie funkcji interpolującej wartości $f(x_i)$ w węzłach kwadratury i obliczenie kwadratury jako całki takiej funkcji.

Kwadratury Newtona-Cotesa

Jeśli wielomiany użyte do konstrukcji kwadratur są wielomianami interpolacyjnymi Lagrange'a opartymi na równo odległych węzłach, to kwadratury te nazywane są *kwadraturami Newtona-Cotesa*. Przyjmijmy $x_i = a + ih,\ i = 0, ..., n$, $a + nh = b$ (tzn. $h = \frac{b-a}{n}$). Stosując zamianę zmiennych, $x = a + th,\ t \in [0, n]$, wielomian interpolacyjny Lagrange'a możemy zapisać w postaci

$$L_n(x) = L_n(a + th) = \sum_{i=0}^{n} f(x_i) \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} = \sum_{i=0}^{n} f(x_i) \prod_{j=0, j\neq i}^{n} \frac{t - j}{i - j}\,. \tag{8.16}$$

Po scałkowaniu prawej strony tej równości uzyskujemy współczynniki kwadratur

$$A_i = \int_a^b \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}\, dx = h \int_0^n \prod_{j=0, j\neq i}^{n} \frac{t - j}{i - j}\, dt, \tag{8.17}$$

gdyż przy zamianie zmiennych mamy $dx = hdt$. Kwadratury Newtona-Cotesa zapisuje się najczęściej w nieco innej, następującej postaci

$$Q(f; a, b) = \sum_{i=0}^{n} A_i f(x_i) = (b - a) \sum_{i=0}^{n} B_i^{(n)} f(x_i), \tag{8.18}$$

gdzie

$$B_i^{(n)} = \frac{A_i}{b - a} = \frac{1}{n} \int_0^n \prod_{j=0, j\neq i}^{n} \frac{t - j}{i - j}\, dt, \tag{8.19}$$

a indeks górny „(n)" podkreśla, że tak zdefiniowane współczynniki kwadratur nie zależą od długości $b - a$ odcinka całkowania, a jedynie od ilości przedziałów n. Z powyższej definicji wynika też bezpośrednio: $\sum_{i=0}^{n} B_i^{(n)} = 1$.

W dalszych rozważaniach potrzebne będzie tzw. drugie twierdzenie o wartości średniej, które przytaczamy poniżej.

Twierdzenie 8.1. *Jeśli w przedziale $[a, b]$ funkcja f jest ciągła, a funkcja g całkowalna i nieujemna lub niedodatnia, to istnieje taki punkt $\alpha \in (a,\ b)$, że*

$$\int_a^b f(x)g(x)\, dx = f(\alpha) \int_a^b g(x)\, dx. \tag{8.20}$$

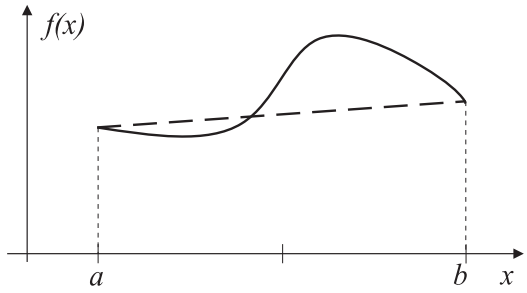
Kwadratura Newtona-Cotesa dla *n*=1. Dla $n = 1$ węzłami kwadratury są punkty $x_0 = a$ i $x_1 = b$, stąd zgodnie z (8.19) mamy

$$B_0^{(1)} = \int_0^1 \frac{t - 1}{0 - 1}\, dt = \left[t - \frac{1}{2}t^2\right]_0^1 = \frac{1}{2},$$

$$B_1^{(1)} = \int_0^1 \frac{t - 0}{1 - 0}\, dt = \left[\frac{1}{2}t^2\right]_0^1 = \frac{1}{2},$$

czyli uzyskujemy aproksymację *wzorem trapezów*

$$Q(f; a, b) = (b - a)\frac{f(a) + f(b)}{2}. \tag{8.21}$$



Rys. 8.1. Interpretacja graficzna kwadratury trapezów

Interpretację graficzną kwadratury trapezów przedstawiono na rysunku 8.1: oryginalna funkcja podcałkowa (linia ciągła) jest dla operacji całkowania zastąpiona odcinkiem prostej łączącej wartości funkcji na końcach przedziału $[a,\ b]$.

Wyznaczymy resztę kwadratury trapezów. Korzystając ze wzoru (5.8) na błąd interpolacji wielomianowej (zob. rozdz. 5.1) i drugiego twierdzenia o wartości średniej (twierdzenie 8.1), mamy

$$\begin{aligned} R(f;a,b) &= \int_a^b \frac{f^{(2)}(\alpha(x))}{2!}(x-a)(x-b)\,dx = \frac{f^{(2)}(\alpha)}{2!}\int_a^b (x-a)(x-b)\,dx \\ &= -\frac{(b-a)^3}{12}f^{(2)}(\alpha) = -\frac{1}{12}f^{(2)}(\alpha)h^3. \end{aligned} \tag{8.22}$$

Wykazaliśmy więc, że kwadratura trapezów jest rzędu drugiego (reszta jest niezerowa dla wielomianu drugiego stopnia, który ma niezerową pochodną drugiego rzędu (stałą)).

Kwadratura Newtona-Cotesa dla *n*=2. Dla $n = 2$ węzłami kwadratury są punkty $x_0 = a$, $x_1 = \frac{a+b}{2}$ i $x_2 = b$, stąd zgodnie z (8.19) mamy

$$B_0^{(2)} = \frac{1}{2}\int_0^2 \frac{(t-1)(t-2}{(0-1)(0-2)}\,dt = \frac{1}{6},$$

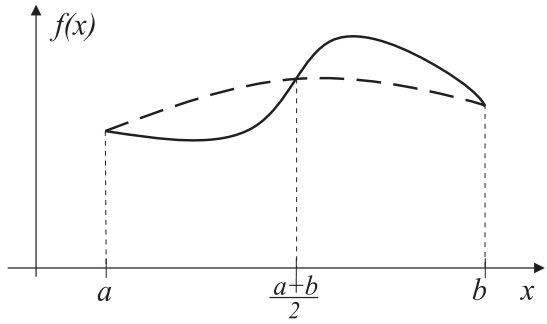
$$B_1^{(2)} = \frac{1}{2}\int_0^2 \frac{(t-0)(t-2)}{(1-0)(1-2)}\,dt = \frac{4}{6},$$

$$B_2^{(2)} = B_0^{(2)} = \frac{1}{6},$$

gdzie wykorzystaliśmy ogólną cechę symetrii kwadratur, tzn. $B_i^{(n)} = B_{n-i}^{(n)}$. Otrzymane współczynniki kwadratury to współczynniki aproksymacji wielomianem drugiego stopnia, tzw. aproksymacji *wzorem parabol*

$$Q(f;a,b) = (b-a)\frac{f(a) + 4f(\frac{a+b}{2}) + f(b)}{6}, \tag{8.23}$$

gdzie $h = \frac{b-a}{2}$. Wzór powyższy znany jest też jako *reguła Simpsona*.



Rys. 8.2. Interpretacja graficzna kwadratury Simpsona

Interpretację graficzną reguły Simpsona przedstawiono na rysunku 8.2: oryginalna funkcja podcałkowa (linia ciągła) jest dla operacji całkowania zastąpiona fragmentem paraboli łączącej wartości funkcji na końcach przedziału $[a, b]$ i ponadto przyjmującej wartość funkcji w punkcie środkowym przedziału.

Można udowodnić, że kwadratury Newtona-Cotesa oparte na n węzłach są rzędu $p = n+1$ dla wartości n nieparzystych, a rzędu $p = n+2$ dla n parzystych. Ponadto, można pokazać, że reszty tych kwadratur wyrażają się zależnością

$$R(f;a,b) = R^{(n)}(f) = -c_n f^{(p)}(\alpha) h^{p+1}, \tag{8.24}$$

gdzie $\alpha \in (a,\ b)$, a c_n to pewne stałe – np. pokazaliśmy to w przykładzie powyżej dla $n=1$. Pamiętajmy, że $h = \frac{a-b}{n}$, tzn. dla $n=1$: $h = a-b$, dla $n=2$: $h = \frac{a-b}{2}$, itd. W tabeli 8.1 podano wartości współczynników $B_i^{(n)}$, rzędów p i stałych c_n kwadratur Newtona-Cotesa dla $n = 1, 2, ..., 6$. Dla wartości n większych od 6 kwadratury Newtona-Cotesa stają się nieprzydatne, gdyż pojawiają się ujemne współczynniki (za wyjątkiem $n=9$).

Tabela 8.1. Współczynniki i parametry kwadratur Newtona-Cotesa

n	$B_i^{(n)}$	p	c_n	nazwa kwadratury
1	$\frac{1}{2}, \frac{1}{2}$	2	$\frac{1}{12}$	trapezów
2	$\frac{1}{6}, \frac{4}{6}, \frac{1}{6}$	4	$\frac{1}{90}$	Simpsona
3	$\frac{1}{8}, \frac{3}{8}, \frac{3}{8}, \frac{1}{8}$	4	$\frac{3}{80}$	Simpsona $\frac{3}{8}$
4	$\frac{7}{90}, \frac{32}{90}, \frac{12}{90}, \frac{32}{90}, \frac{7}{90}$	6	$\frac{8}{945}$	Boole'a
5	$\frac{19}{288}, \frac{75}{288}, \frac{50}{288}, \frac{50}{288}, \frac{75}{288}, \frac{19}{288}$	6	$\frac{275}{12096}$	Bodego
6	$\frac{41}{440}, \frac{216}{840}, \frac{27}{840}, \frac{272}{840}, \frac{27}{840}, \frac{216}{840}, \frac{41}{440}$	8	$\frac{9}{1400}$	Weddle'a

Złożone kwadratury Newtona-Cotesa

Dla numerycznego obliczania całki z małym błędem, na domkniętym (dowolnym) przedziale całkowania, nie wystarczy zastosowanie pojedynczej kwadratury. Na ogół dzielimy przedział na wiele podprzedziałów i na każdym z nich stosujemy kwadraturę niskiego rzędu, najczęściej regułę trapezów lub parabol (Simpsona). Powstają w ten sposób *kwadratury złożone*.

Złożona kwadratura trapezów. Dla $n=2$ przyjmijmy, że odcinek $[a,b]$ podzielony został punktami $x_i = a+ih$ na N równych części $[x_i, x_{i+1}]$ o długości $h = \frac{b-a}{N}$, $i = 0,1,...,N-1$. Stosując dla każdego $i = 0,1,...,N-1$ prostą kwadraturę trapezów (8.21) oraz wzór (8.22) na reszty aproksymacji, dostajemy

$$\begin{aligned}\int_a^b f(x)\,dx &= \sum_{i=0}^{N-1}\int_{x_i}^{x_{i+1}} f(x)\,dx = \sum_{i=0}^{N-1}\left[\frac{h}{2}\left(f(x_i)+f(x_{i+1})\right) - \frac{1}{12}f^{(2)}(\alpha_i)h^3\right] \\ &= \frac{h}{2}\left(f(a)+2\sum_{i=1}^{N-1} f(a+ih)+f(b)\right) - \frac{h^3}{12}\sum_{i=0}^{N-1} f^{(2)}(\alpha_i),\end{aligned}$$

gdzie $\alpha_i \in (x_i, x_{i+1})$. Uzyskaliśmy w ten sposób dla $n=2$ złożoną kwadraturę $Q_N(f;a,b) = T_N(f;a,b)$, zwaną *złożonym wzorem trapezów*

$$T_N(f;a,b) = \frac{h}{2}\left(f(a)+2\sum_{i=1}^{N-1} f(a+ih)+f(b)\right). \tag{8.25}$$

Zakładając ciągłość drugiej pochodnej funkcji f na odcinku $[a,b]$, resztę tej kwadratury można wyrazić w następująco:

$$R_{TN}(f;a,b) = -\frac{h^3 N}{12}\cdot\frac{1}{N}\sum_{i=0}^{N-1} f^{(2)}(\alpha_i) = -N\cdot\frac{h^3}{12}f^{(2)}(\alpha_s),$$

a następnie, po uwzględnieniu relacji między długością przedziału $[a, b]$ a wartością N, dostajemy

$$R_{TN}(f; a, b) = -(a-b)\frac{h^2}{12} f^{(2)}(\alpha_s), \tag{8.26}$$

gdzie liczba $f^{(2)}(\alpha_s)$ może być traktowana jako średnia wartość drugiej pochodnej funkcji f na przedziale $[a, b]$ (średnia tym dokładniejsza, im większe N).

Złożona kwadratura Simpsona (parabol). Dla $n = 3$, podzielmy przedział $[a, b]$ na N równych podprzedziałów punktami $x_i, i = 0, 1, 2, ..., N$, zakładając przy tym, że N jest parzyste i definiując $h = \frac{a-b}{N}$. Na każdym z podprzedziałów o długości $2h$, tzn. $[a, a + 2h]$, $[a + 2h, a + 4h]$, itd., zastosujmy wzór Simpsona (8.23). Stosując ponadto wzór (8.24) dla rzędu $p = n+1 = 4$ i stałej $c_n = c_2 = \frac{1}{90}$, dla każdego $k = 1, 2, ..., \frac{N}{2}$ (k indeksuje podprzedziały $[x_{2(k-1)}, x_{2k}]$ o długości $2h$), dostajemy

$$\begin{aligned}
\int_a^b f(x)\,dx &= \sum_{k=1}^{N/2} \left[\frac{2h}{6} \left(f(x_{2k-2}) + 4f(x_{2k-1}) + f(x_{2k}) \right) - \frac{1}{90} f^{(4)}(\alpha_k) h^5 \right] \\
&= \frac{h}{3}\Big(f(a) + 4f(x_1) + 2f(x_2) + 4f(x_3) + 2f(x_4) + \cdots \\
&\qquad \cdots + 4f(x_{N-1}) + f(b) \Big) - \sum_{k=1}^{N/2} \frac{1}{90} f^{(4)}(\alpha_k) h^5 \\
&= \frac{h}{3} \left(f(a) + 4 \sum_{k=1}^{N/2} f(x_{2k-1}) + 2 \sum_{k=1}^{N/2-1} f(x_{2k}) + f(b) \right) - \frac{h^5}{90} \sum_{k=1}^{N/2} f^{(4)}(\alpha_k) \\
&= S_N(f; a, b) + R_{SN}(f; a, b),
\end{aligned} \tag{8.27}$$

gdzie $\alpha_k \in [x_{2k-1}, x_{2k}]$, $k = 1, ..., N/2$ oraz składowa (reszta) $R_{SN}(f; a, b)$ definiowana jest ostatnią sumą. Podobnie jak poprzednio, zakładając ciągłość pochodnej czwartego rzędu funkcji f na $[a, b]$, resztę $R_{SN}(f; a, b)$ możemy wyrazić następująco

$$\begin{aligned}
R_{SN}(f; a, b) &= -\frac{h^5 N}{180} \cdot \frac{1}{N/2} \sum_{k=1}^{N/2} f^{(4)}(\alpha_k) \\
&= -N \cdot \frac{h^5}{180} f^{(4)}(\alpha_s) = -(a-b)\frac{h^4}{180} f^{(4)}(\alpha_s).
\end{aligned} \tag{8.28}$$

Po porównaniu wzorów (8.26) i (8.28) widać wyraźnie znacznie większą dokładność kwadratury Simpsona: przy tej samej liczbie punktów (a więc tym samym

kroku h) błąd kwadratury złożonej Simpsona jest $15/h^2$ razy mniejszy. Na przykład, dla $h = 0.1$ mamy $15/h^2 = 1500$. Natomiast, zakładając taki sam błąd, złożona kwadratura trapezów wymaga kroku h znacznie mniejszego.

Dokładność numerycznego obliczania całki jest na ogół określana jako wymagana dokładność względna, tzn. dla pewnej wartości ε wymagamy

$$\frac{|Q(f;a,b) - I(f;a,b)|}{|I(f;a,b)|} < \varepsilon. \tag{8.29}$$

Do określenia dokładności można też wykorzystać oszacowanie reszty kwadratury, umożliwia to oszacowanie liczby punktów N, w których będzie obliczana wartość funkcji podcałkowej. Wymaga to jednakże oszacowania średniej wartości drugiej (dla kwadratury trapezów) lub czwartej (dla kwadratury parabol) pochodnej funkcji na przedziale $[a,b]$ – dla ułatwienia czy większej pewności można szacować wartość maksymalną odpowiedniej pochodnej. Może to jednakże być trudne lub prowadzić do oszacowań obarczonych dużym nadmiarem. Stąd praktycznym sposobem postępowania jest *obliczanie ciągu kwadratur dla rosnącej liczby punktów*, tzn. $Q_{N_1}(f;a,b)$, $Q_{N_2}(f;a,b)$, $Q_{N_3}(f;a,b)$, itd., dla $N_1 < N_2 < N_3 < \cdots$, gdzie $Q_{N_i}(f;a,b)$ oznacza kwadraturę obliczoną z wykorzystaniem N_i punktów. Obliczenia przerywamy, gdy

$$\frac{|Q_{N_{j-1}}(f;a,b) - Q_{N_j}(f;a,b)|}{|Q_{N_j}(f;a,b)|} < \varepsilon, \tag{8.30}$$

przy czym można tu przyjmować za ε wartość nieco inną niż byłaby stosowana w zal. (8.29). Relacje między kolejnymi wartościami N_j powinny być takie, żeby wykorzystywać już obliczone wartości funkcji podcałkowej. Pokażemy taki sposób postępowania dla złożonych kwadratur trapezów i Simpsona.

Jeśli będziemy przyjmować, że $N_{j+1} = 2N_j$, to $h_{N_{j+1}} = h_{N_j}/2$ i uzyskamy następujący *wzór rekurencyjny dla złożonej kwadratury trapezów*:

$$T_{N_{j+1}}(f;a,b) = \frac{1}{2}T_{N_j}(f;a,b) + h_{N_{j+1}} \sum_{i=1}^{N_{j+1}/2} f(a + (2i-1)h_{N_{j+1}}), \tag{8.31}$$

gdzie $h_{N_j} = \frac{b-a}{N_j}$.

Aby wyznaczyć podobny *wzór rekurencyjny dla złożonej kwadratury Simpsona* (8.27), przedstawmy najpierw ten wzór w równoważnej postaci

$$\begin{aligned} S_N(f;a,b) &= \frac{h}{3}\left(f(a) + 2\sum_{k=1}^{N/2} f(x_{2k-1}) + 2\sum_{k=1}^{N/2-1} f(x_{2k}) + f(b) \right) + \\ &\qquad + 2\frac{h}{3}\sum_{k=1}^{N/2} f(x_{2k-1}) \\ &= S_N^1(f;a,b) + S_N^2(f;a,b). \end{aligned} \tag{8.32}$$

gdzie składowa $S_N^2(f;a,b)$ definiowana jest ostatnią sumą (w drugiej linii) wzoru (8.32). Jeśli teraz przyjmiemy $N_{j+1} = 2N_j$, tzn. $h_{N_{j+1}} = h_{N_j}/2$, to nowe punkty wyznaczone po dwukrotnym zmniejszeniu kroku stają się wszystkimi punktami o indeksach nieparzystych nowej kwadratury, a węzły poprzedniej kwadratury stają się wszystkimi punktami o indeksach parzystych nowej kwadratury, skąd bezpośrednio wynika

$$\begin{aligned} S_{N_{j+1}}(f;a,b) &= \left[\frac{1}{2}S^1_{N_j}(f;a,b) + 2\frac{h_{N_{j+1}}}{3}\sum_{k=1}^{N_{j+1}/2} f(x_{2k-1})\right] + \\ &\qquad + 2\frac{h_{N_{j+1}}}{3}\sum_{k=1}^{N_{j+1}/2} f(x_{2k-1}) \\ &= S^1_{N_{j+1}}(f;a,b) + S^2_{N_{j+1}}(f;a,b). \end{aligned} \tag{8.33}$$

Przykład 8.3. Policzymy, w środowisku MATLAB, ciągi wartości złożonych kwadratur trapezów i Simpsona dla zadania obliczania całki funkcji $f(x) = \sin(x)$ na przedziale $[0, \pi/2\,]$. Jak łatwo policzyć, wartość dokładna tej całki wynosi 1. Dla kwadratur trapezów, wartości kolejnych kwadratur $T_N(\sin; 0, \pi/2)$ wraz z resztami $R_{TN}(\sin; 0, \pi/2)$ i wartościami estymaty błędu (8.30) oraz odpowiadającymi im wartościami $N = 2, 4, 8, ...$, podano poniżej:

N	$T_N(\sin;0,\pi/2)$	$R_{TN}(\sin;0,\pi/2)$	$\lvert(T_{N/2}-T_N)/T_N\rvert$
2	0.948059448968520	0.051940551031480	
4	0.987115800972775	0.012884199027225	0.039566129896580
8	0.996785171886170	0.003214828113830	0.009700556535264
16	0.999196680485072	0.000803319514928	0.002413447368272
32	0.999799194320019	0.000200805679981	0.000602634847447
64	0.999949800092101	0.000050199907899	0.000150613332858
128	0.999987450117527	0.000012549882473	0.000037650497935
256	0.999996862535288	0.000003137464712	0.000009412447293
512	0.999999215634191	0.000000784365809	0.000002353100748
1024	0.999999803908570	0.000000196091430	0.000000588274494
$\vdots$	$\vdots$	$\vdots$	$\vdots$
2^{20}	0.999999999999747	0.000000000000253	0.000000000000535

Poniżej podajemy ponadto tekst prostego programu (w języku MATLAB) obliczającego wartości ciągu złożonych kwadratur trapezów (dla funkcji z przykładu), wykorzystującego wzór rekurencyjny (8.31):

```
%script CiagKwadraturTrapezowSinus
liczba=10;    %liczba kolejnych kwadratur złożonych
k=1;N=2;h=pi/4;
T=zeros(liczba,1);
T(1)=0.5*h*(sin(0)+2*sin(pi/4)+sin(pi/2));
for k=2:liczba
    h=h*0.5;
    N=2*N;
    T(k)=0.5*T(k-1);
    for i=1:N/2
        T(k)=T(k)+h*sin((2*i-1)*h);
    end
end
```

W następnej tabelce podano, analogicznie, wartości kolejnych kwadratur Simpsona $S_N(\sin; 0, \pi/2)$ wraz z resztami $R_{SN}(\sin; 0, \pi/2)$ i wartościami estymaty błędu (8.30) oraz odpowiadającymi im wartościami $N = 2, 4, 8, ..., 1024$:

N	$S_N(\sin; 0, \pi/2)$	$R_{SN}(\sin; 0, \pi/2)$	$\lvert(S_{N/2} - S_N)/S_N\rvert$
2	1.002279877492210	−0.002279877492210	0
4	1.000134584974194	−0.000134584974194	0.002145003832731
8	1.000008295523968	−0.000008295523968	0.000126288402598
16	1.000000516684707	−0.000000516684707	0.000007778835242
32	1.000000032265001	−0.000000032265001	0.000000484419690
64	1.000000002016129	−0.000000002016129	0.000000030248872
128	1.000000000126001	−0.000000000126001	0.000000001890127
256	1.000000000007875	−0.000000000007875	0.000000000118126
512	1.000000000000492	−0.000000000000492	0.000000000007383
1024	1.000000000000031	−0.000000000000031	0.000000000000462

Zwróćmy uwagę, że rezultaty osiągane z wykorzystaniem złożonej kwadratury Simpsona są znacznie lepsze, a praktycznie nakład obliczeń jest taki sam jak dla złożonej kwadratury trapezów, gdyż wykorzystując wzory rekurencyjne (8.31) i (8.33), dla obu kwadratur liczymy wartości funkcji całkowanej w każdym punkcie tylko raz, jedynie obróbka tych danych jest nieco inna.

Ponadto zauważmy, że wartości oszacowań błędu wg zależności (8.30) są w powyższym przykładzie dla kwadratury trapezów praktycznie tego samego rzędu co wartości reszt kwadratur, natomiast dla kwadratury Simpsona dają oszacowania bardziej konserwatywne, mniej więcej o rząd wielkości.

□

Złożone kwadratury Newtona-Cotesa, tak jak je przedstawialiśmy dotychczas, wykorzystują podział odcinka całkowania równomiernie rozłożonymi punktami (węzłami kwadratury). Podejście takie może nie być efektywne dla funkcji o różnej szybkości zmienności w różnych częściach przedziału całkowania. Efektywniejsze są *kwadratury adaptacyjne*, które wykorzystują nierównomiernie rozłożone węzły kwadratury. Popularne jest zastosowanie *adaptacyjnej kwadratury Simpsona*, która polega na sukcesywnym, adaptacyjnym podziale przedziału całkowania na podprzedziały i niezależnym testowaniu dokładności całkowania na tych podprzedziałach, po czym dalej dzielone są tylko te z nich, na których test dokładności nie jest spełniony. Metodę tę zaimplementowano w procedurze numerycznego obliczania całek oznaczonych „quad" w środowisku MATLAB. Po szczegóły algorytmu numerycznego obliczania całki adaptacyjną regułą Simpsona odsyłamy czytelnika do [8].

Istnieją kwadratury inne niż kwadratury Newtona-Cotesa, dokładniejsze dzięki wykorzystywaniu nierównomiernego podziału przedziału całkowania, np. kwadratury Gaussa, kwadratury Czebyszewa. Po informacje dotyczące tych kwadratur odsyłamy czytelnika do literatury, np. [8, 4, 11].

Zadania

1. Wyznacz oszacowanie błędu względnego powstającego przy numerycznym wyznaczaniu pochodnej pierwszego rzędu funkcji ilorazem różnicowym centralnym, wyznacz optymalną długość kroku dla tej aproksymacji, minimalizując oszacowanie całkowitego błędu bezwzględnego (tzn. sumy błędów metody i numerycznego). Przyjmij założenia dotyczące błędów reprezentacji wartości funkcji jak przy wyznaczaniu błędu (8.8) dla różnicy wstecznej w rozdz. 8.1.

2. Wyprowadź zależność (8.12) jako pochodną wielomianu interpolacyjnego Newtona.

3. Wyprowadź zależność na pochodną pierwszego rzędu analogiczną do (8.12), ale formułując wielomian interpolacyjny na podstawie punktów $x-h,\ x,\ x+h$.

4. Napisz (np. w środowisku MATLAB) programy do rekurencyjnego obliczania ciągu kwadratur trapezów i ciągu kwadratur Simpsona. Wykorzystując te programy, policz wartości ciągu dziesięciu (poczynając od $N=2$) złożonych kwadratur trapezów i Simpsona dla funkcji $1-e^{-x}\cos(4x)$ na przedziale $[0, \pi/4]$, każdy kolejny krok przyjmując jako połowę poprzedniego.

Bibliografia

[1] Dryja M., Jankowscy J.M.: Przegląd metod i algorytmów numerycznych, cz. II. WNT, Warszawa 1988.

[2] Fortuna Z., Macukow B.,. Wąsowski J: Metody numeryczne. WNT, Warszawa 1993.

[3] Jankowscy J.M.: Przegląd metod i algorytmów numerycznych, cz. I. WNT, Warszawa 1988.

[4] Klamka J., Ogonowski Z., Jamicki M., Stasiak M.: Metody Numeryczne. Wydawnictwa Politechniki Śląskiej, Gliwice 2013.

[5] Klamka J., Pawełczyk M., Wyrwał J.: Numerical methods. Wydawnictwa Politechniki Śląskiej, Gliwice 2001.

[6] Krupka J., Miękina A., Morawski R.Z., Opalski L.J.: Wstęp do metod numerycznych. Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 2009.

[7] Krupowicz A.: Metody numeryczne zagadnień początkowych równań różniczkowych zwyczajnych. PWN, Warszawa 1986.

[8] Mathews J.H., Fink K.D.: Numerical Methods Using Matlab (4th ed.). Prentice Hall, 2004.

[9] Press W.H., Teukolsky S.A., Vetterling W.T., Flannery B.P.: Numerical Recipes, Third Edition. Cambridge Univ. Press, Cambridge 2007 (dostępna również wersja elektroniczna).

[10] Rosłoniec S.: Wybrane metody numeryczne. Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 2002.

[11] Stoer J., Bulirsch R.: Wstęp do analizy numerycznej. PWN, Warszawa 1987.

[12] Stoer J., Bulirsch R.: Introduction to Numerical Analysis (3rd ed.). Springer, 2002.

[13] Watkins D.S.: Fundamentals of Matrix Computations. J. Wiley & Sons, Hoboken, New Jersey, 2010.

[14] Zalewski A. , Cegieła R.: MATLAB – obliczenia numeryczne i ich zastosowania. Wyd. Nakom, Poznań, 1996.