3. Kwadratury

Janusz Miller, Jarosław Wąs

3.1. Wprowadzenie

Kwadratury służą do numerycznego całkowania. Są one stosowane do obliczania przybliżonych wartości całek oznaczonych z funkcji, zazwyczaj danych przepisem matematycznym. Oprócz zastosowań tych metod do samego obliczania całki kwadratury są podstawą konstrukcji algorytmów rozwiązywania zagadnień – można powiedzieć nadrzędnych – takich jak numeryczne rozwiązywanie równań różniczkowych.

Rozdział ten jest wprowadzeniem do zagadnień poruszanych w dalszej części książki, w szczególności w rozdziałach 5–7. Dlatego zakres przeglądu kwadratur jest tu ograniczony do tych, które najczęściej są używane w metodach numerycznego rozwiązywania zagadnień początkowych i brzegowych równań różniczkowych zwyczajnych (nie obejmuje np. kwadratur Czebyszewa, Eulera–Maclaurina, całek niewłaściwych, podwójnych itd.).

Rozróżnijmy dwie okoliczności stosowania numerycznych metod całkowania funkcji.

1) Wartości funkcji podcałkowej mogą być wyznaczone dla dowolnego argumentu, np. gdy funkcja podcałkowa jest zadana skończonym przepisem (wzorem).

2) Wartość funkcji jest znana tylko dla skończonej liczby wartości argumentu (liczby punktów), np. gdy pochodzą z pomiaru w dyskretnych punktach.

Koncentrujemy się na pierwszym przypadku, sporadycznie wskazując na różnice w wyborze metody, gdy zachodzi przypadek drugi.

Dla jednoznaczności terminologii przyjmijmy określenia używanych dalej pojęć.

Klasa wielomianów $\mathtt{P}_n$ – wszystkie wielomiany stopnia $k \leq n$.

Kwadratura – metoda numeryczna polegająca na **przybliżeniu całki** za pomocą odpowiedniej sumy ważonej wartości całkowanej funkcji obliczonych w kilku punktach.

Wartość dokładną całki będziemy zapisywać funkcjonałem

$$I[f] = \int_a^b f(x)dx,$$

a jej przybliżenie numeryczne

$$Q[f] = (b-a)\sum_{k=0}^{n} \beta_k f(x_k). \tag{3.1}$$

Błąd przybliżenia jest różnicą

$$E[f] = I[f] - Q[f]. \tag{3.2}$$

Rzędem metody całkowania jest liczba p taka, że

$$Q[W_p] = I[W_p], \quad \text{dla} \quad W_p \in \mathtt{P}_p$$

i

$$\exists W_{p+1}: \; Q[W_{p+1}] \neq I[W_{p+1}].$$

Inaczej mówiąc, rząd jest równy p, jeżeli metoda jest dokładna dla klasy wielomianów $\mathtt{P}_p$ i istnieje wielomian stopnia $p+1$, dla którego błąd metody jest różny od zera.

3.2. Kwadratury z węzłami równoodległymi

Przyjmijmy, że wartości funkcji f można wyznaczyć dla argumentów, x_k takich że $x_{k+1} - x_k = h$, $k = m, m+1, \ldots, M-1$. Sformułujmy zadanie w sposób ogólny: wyznaczyć przybliżoną wartość całki z funkcji f na przedziale $[x_0, x_q]$ na podstawie wartości funkcji w punktach $x_m, x_{m+1}, \ldots, x_M$. Zwróćmy uwagę na brak zależności między wartościami parametrów q, m, M, (poza warunkami $q > 0$, $m < M$)[1]. Ogólną kwadraturę można zapisać wzorem [1]

$$\int_{x_0}^{x_q} f(x)dx \approx h\sum_{k=m}^{M} \beta_k f(x_k).$$

Wzór ten umożliwia przejrzystą klasyfikację – określaną jako Q.q.M.m – kwadratur opartych na węzłach równooddalonych. Przykłady najprostszych kwadratur: $h = x_q - x_0$, $\xi_i \in [x_0, x_q]$, $i = 1, \ldots, 9$. zawiera tabela 3.1.

Wąską, ale często stosowaną podgrupę tych metod, stanowią Q.q.q.0 – zamknięte kwadratury Newtona–Cotesa.

[1] W rozdziale 5 powrócimy do kwadratur z $m < M = 0$.

Tabela 3.1
Przykładowe kwadratury oparte na węzłach równoodległych

Q.q.M.m	**Kwadratura**	β_k	p	**E[f]**
Q.1.0.0	prostokątów (w przód)	1	0	$+\frac{1}{2}h^2 f'(\xi_1)$
Q.1.1.1	prostokątów (wstecz)	1	0	$-\frac{1}{2}h^2 f'(\xi_2)$
Q.2.1.1	prostokątów p. środk.	1	1	$+\frac{1}{3}\frac{h}{2}^3 f''(\xi_3)$
Q.1.1.0	trapezów (NC1)	$\frac{1}{2}, \frac{1}{2}$	1	$-\frac{1}{12}h^3 f''(\xi_4)$
Q.2.2.0	Simpsona (NC2)	$\frac{1}{6}, \frac{2}{3}, \frac{1}{6}$	3	$-\frac{1}{90}\frac{h}{2}^5 f^{(4)}(\xi_5)$
Q.1.2.0	reguła 3/8	$\frac{1}{8}, \frac{3}{8}, \frac{3}{8}, \frac{1}{8}$	3	$-\frac{3}{80}\frac{h}{3}^5 f^{(4)}(\xi_6)$
Q.4.4.0	Boole'a (NC4)	$\frac{7}{90}, \frac{32}{90}, \frac{12}{90}, \frac{32}{90}, \frac{7}{90}$	5	$-\frac{8}{945}\frac{h}{4}^7 f^{(6)}(\xi_7)$
Q.6.6.0	NC 6	$\frac{41}{840}, \frac{216}{840}, \frac{27}{840}, \frac{272}{840},$	7	$\sim h^9 f^{(8)}(\xi_8)$
Q.8.8.0	NC 8	> 0	9	$\sim h^{11} f^{(10)}(\xi_9)$
...				

3.2.1. Metody Newtona–Cotesa (NC)

Kwadratura Newtona–Cotesa (NC) jest to kwadratura w postaci $Q(f) = I(L_n)$, gdzie L_n jest wielomianem interpolacyjnym w postaci Lagrange'a opartym na $n+1$ równoodległych węzłach x_0, x_1, ..., x_n. Jeżeli $x_0 = a$ i $x_n = b$, to mówimy o zamkniętych kwadraturach Newtona–Cotesa

$$x_k = a + k\frac{b-a}{n}, \quad k = 0, 1, \ldots, n. \tag{3.3}$$

$$L_n(x) = \sum_{k=0}^{n} f(x_k)\, l_k(x), \quad l_k(x) = \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i}. \tag{3.4}$$

Współczynniki β_k we wzorze (3.1) są całkami z wielomianów Lagrange'a

$$\beta_k = \int_a^b l_k(x)dx.$$

Współczynniki te, dla kwadratur NC1–NC10, można znaleźć między innymi na stronie https://mathworld.wolfram.com/Newton-CotesFormulas.html. Dla $n < 8$ oraz $n = 10$ są one dodatnie. Wybór kwadratury wysokiego rzędu, ze współczynnikami różnych znaków, stwarza niebezpieczeństwo wzmocnienia szumu numerycznego – gdy n wzrasta, to rząd wielkości kwadratury się nie zmienia, ale jej wartość jest obliczana jako ważona średnia o rosnących (co do bezwzględnej wartości) współczynnikach wagowych, a to wiąże się z odejmowaniem dużych (w stosunku do wyniku) liczb. Zjawisko to trudno zaobserwować dla niewielkich wartości $n \sim 10$.

Oszacowanie błędu

Po całkowaniu błędu interpolacji Lagrange'a otrzymujemy oszacowanie każdej kwadratury utworzonej na bazie tej interpolacji

$$f \in C^{n+1} \Rightarrow \exists \xi \in [a,b]: \quad E[f] = \int_a^b \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)(x-x_1)\dots(x-x_n)dx. \tag{3.5}$$

W przypadku węzłów równoodległych całkę z wielomianu można oszacować dokładniej. W przypadku zamkniętej kwadratury NC przydatne jest twierdzenie 3.1, a dość złożony dowód przytacza [2].

Twierdzenie 3.1. *Istnieje $\xi \in [a,b]$ takie, że błąd (3.2) wynosi:*

$$E_h[f] = \frac{h^{n+3} f^{(n+2)}(\xi)}{(n+2)!} \int_0^n t^2(t-1)\cdots(t-n)dt, \tag{3.6}$$

gdy n jest parzyste i $f \in C^{n+2}$ i

$$E_h[f] = \frac{h^{n+2} f^{(n+1)}(\xi)}{(n+1)!} \int_0^n t(t-1)\cdots(t-n)dt, \tag{3.7}$$

gdy n jest nieparzyste i $f \in C^{n+1}$.

Porównanie rzędów pochodnych funkcji f w obu wyrażeniach[2] prowadzi do wniosku, że rząd kwadratur wzrasta o 2, gdy n wzrasta o 1 z wartości nieparzystej do parzystej. Ten fakt, na który wskazuje zbliżona do schodkowej zmiana błędu ze zmianą rzędu kwadratur NC widoczna na wykresie rysunku 3.1, decyduje o popularności kwadratur NC opartych na nieparzystej liczbie węzłów, np. kwadratur Simpsona czy NC8.

Twierdzenie analogiczne do twierdzenia 3.1 odnosi się do otwartych kwadratur NC. Różnice wyrażeń określających błąd sprowadzają się do innych granic całki z wielomianu – przedział $[0,n]$ zostaje rozszerzony o 2, do przedziału $[-1,n+1]$. W konsekwencji stałe błędu kwadratur otwartych są większe od ich odpowiedników dla kwadratur zamkniętych. Dlatego kwadratury zamknięte są na ogół częściej stosowane niż kwadratury otwarte, ale z wyjątkami dla nas istotnymi, ponieważ kwadratury otwarte stosuje się w rozwiązywaniu równań różniczkowych, na co wskazują między innymi Burden i Faires [3] – do tego zagadnienia powrócimy w rozdziale 5.

Kwadratury złożone

Powyżej wskazano, że możliwości redukcji błędu kwadratur w wyniku zwiększania rzędu metody są ograniczone. Innym sposobem zmniejszania błędu jest skracanie odległości h

[2] Po odwołaniu się do definicji rzędu metody i faktu, że dla wielomianów $w_n(x)$ klasy $\mathtt{P}_n$ $w_n^{(m)}(x) \equiv 0$ dla $m > n$.

między węzłami przez podział przedziału całkowania $[a,b]$ na m podprzedziałów wyznaczonych punktami $a = x_0 < x_1 < x_2 < \ldots < x_m = b$. **Kwadratura złożona** jest przybliżeniem całki na przedziale $[a,b]$ w postaci sumy kwadratur obliczonych w każdym podprzedziale $[x_{i-1}, x_i]$, $i = 1, \ldots, m$.

W przypadku ogólnym, gdy kwadratury w podprzedziałach są obliczane niezależnie, wówczas koszt obliczeniowy jest m razy większy od kosztu kwadratury obliczanej dla całego przedziału.

Na przykład złożona kwadratura trapezowa wymagałaby $2m$-krotnego obliczenia wartości funkcji podcałkowej, m mnożeń przez długości podprzedziałów oraz sumy m składników. W celu zmniejszenia kosztu obliczeniowego przedział $[a,b]$ jest dzielony na m odcinków równej długości $h = (b-a)/m$. Wtedy tę kwadraturę można zapisać wzorem

$$Q[f] = \frac{h}{2}\left[f(a) + f(b) + 2\sum_{j=1}^{m-1} f(x_j)\right], \quad h = \frac{b-a}{m}, \quad x_j = a + jh. \tag{3.8}$$

Jej koszt obliczeniowy został zredukowany o $m-1$ obliczeń wartości funkcji, i $m-1$ mnożeń.

3.2.2. Metoda Romberga

W literaturze używane są określenia „kwadratura", „metoda" lub „schemat Romberga". Pojęcia „kwadratura Romberga" będziemy używać dla wyniku czwartego etapu metody (schematu) Romberga. Poniżej skrótowo przedstawimy założenia tej metody, a do zastosowania jej w algorytmach adaptacyjnych powrócimy w punkcie następnym.

Metoda Romberga służy do wyznaczania wstępnego przybliżenia całki złożonymi kwadraturami trapezowymi, a następnie, po zastosowaniu ekstrapolacji Richardsona[3], kolejnymi przybliżeniami o coraz wyższym rzędzie dokładności. Jeden z kilku możliwych schematów przewiduje przyjęcie a priori rzędu docelowego przybliżenia. Przyjmijmy ogólnie, że będzie to nieparzysta liczba $2n+1$, choć najczęściej jest ona równa 7 (do uzasadnienia tego wyboru nawiązuje Eksperyment 3.1). Przedział całkowania $[a,b]$ o długości h jest dzielony na $m = 2^n$ podprzedziałów $[x_0 = a, x_1], [x_1, x_2], \ldots, [x_{m-1}, x_m = b]$. W punktach x_i, $i = 0, 1, \ldots, m$ obliczane są wartości funkcji $f(x_i)$.

Na początku obliczane są kwadratury trapezowe, każda na przedziale $[a,b]$. Pierwsza jest obliczona w wersji podstawowej na przedziale o długości $h_0 = h$, a kolejne w wersji złożonej, z podziałem na podprzedziały o długościach $h_1 = h/2, h_2 = h/4, \ldots, h_n = h/2^n$. Kwadratury te oznaczmy symbolami odpowiednio $T_0(h), T_0(h/2), \ldots, T_0(h/2^n)$. W celu ograniczenia kosztu obliczeniowego stosuje się algorytm wykorzystujący prawidłowość, że około połowa składników sumy (3.8) tworzącej kwadraturę $T_0(h/2^i)$ została obliczona w trakcie wyznaczania wartości $T_0(h/2^{i-1})$.

Wartość dokładną całki możemy więc aproksymować przybliżeniami $T_0(\cdot)$ tego samego (pierwszego) rzędu, ale z parametrem $h/2^i$. Pierwszy warunek zastosowania kilkukrotnej

[3] Skrócony opis podano w podrozdziale 1.1.

ekstrapolacji Richardsona do podwyższenia rzędu przybliżeń jest zatem spełniony – wyrażeniem A w szeregu (1.7) jest kwadratura T_0. Warunkiem podwyższenia rzędu dokładności przybliżenia co najmniej o 1 przez każdy etap ekstrapolacji Richardsona jest istnienie szeregu postaci (1.7), czyli takiego, w którym kolejne wyrazy szeregu są funkcjami potęgowymi zmiennej h_i (czyli współczynniki $a_0, a_1, \ldots$ nie zależą od h_i).

Znalezienie rozwinięcia w szereg błędu kwadratury trapezowej nie jest trudne. Po rozwinięciu funkcji podcałkowej w otoczeniu punktu centralnego podprzedziału całkowania $x_c = (x_a + x_b)/2$ i trzykrotnym jego wykorzystaniu (raz przez całkowanie go wyraz po wyrazie dla reprezentacji całki oraz drugi i trzeci raz – dla reprezentacji kwadratury trapezowej) – wyrazimy nim wartości funkcji $f(x_a)$ i $f(x_b)$ i otrzymamy

$$\int_{x_a}^{x_b} f(x)dx = h\frac{f(x_a)+f(x_b)}{2} - \frac{h^3}{12}f^{(2)}(x_c) - \frac{h^5}{480}f^{(4)}(x_c) - \frac{h^7}{53760}f^{(6)}(x_c) - \ldots$$

Należy zwrócić uwagę, że szeregu tego nie można zastosować do ekstrapolacji Richardsona w przypadku, gdy wyrażenie A we wzorze (1.7) jest kwadraturą złożoną. Wartości współczynników a_i zależałyby od punktu centralnego każdego podprzedziału, a więc zależałyby od ich liczby i od h_i.

Ideę sposobu uzyskania szeregu odpowiedniego dla ekstrapolacji Richardsona można sprowadzić do zastąpienia ciągu wyrazów z pochodnymi w punktach centralnych podprzedziałów dwoma ciągami wyrazów pochodnych obliczanych na końcach podprzedziału, w którym obliczana jest całka. Przy sumowaniu całek liczonych na podprzedziałach wyrazy tych ciągów powinny się zredukować z wyjątkiem pochodnych obliczanych w punktach a i b. Realizacja tej idei wymaga zastosowania wzoru Eulera–Maclaurina z wielomianami Bernoulliego. Ze względu na dużo większą złożoność wyprowadzenie takiego szeregu pomijamy, polecając obszerne wprowadzenie do tego zagadnienia podane w [4]. Ostatecznie otrzymujemy szereg

$$\int_a^b f(x)dx = h_n\left(\frac{f(a)+f(b)}{2} + \sum_{i=1}^{m-2}[f(x_i)+f(x_{i+1})]\right) + \sum_{k=1}^{\infty} A_{2k}h_n^{2k}[f^{(2k-1)}(a) - f^{(2k-1)}(b)].$$

Wartości współczynników A_{2k} i ich związek z liczbami Bernoulliego nie mają znaczenia przy korzystaniu z szeregu w ekstrapolacji Richardsona. Istotne jest tylko to, że ich wartość nie zależy od liczby podprzedziałów m, a pochodne kolejnych rzędów są obliczane w tych samych punktach (końcach przedziału) niezależnie od liczby podprzedziałów. Zmieniając jedynie oznaczenia, możemy ten szereg zapisać w postaci

$$\int_a^b f(x)dx = T_0(h_n) + a_0h_n^2 + a_1h_n^4 + a_2h_n^6 + \ldots,$$

a więc takiej, jaka jest wymagana w ekstrapolacji Richardsona. Warto zwrócić uwagę na wykładniki potęg h_n w kolejnych wyrazach szeregu (3.2.2). Mają one wartości tylko parzyste,

co sprawia, że wyrugowanie w jednym etapie ekstrapolacji jednego wyrazu szeregu zwiększa rząd przybliżenia o 2.

Na pierwszym etapie ekstrapolacji z przybliżeń $T_0(h)$ i $T_0(h/2)$ otrzymujemy nowe przybliżenie, oznaczmy je

$$T_1(h) = \frac{4T_0(h/2) - T_0(h)}{3},$$

dla którego szereg reprezentujący błąd nie zawiera wyrazu z potęgą h^2. Analogicznie z $T_0(h/2)$ i $T_0(h/4)$ otrzymujemy przybliżenia $T_1(h/2)$ itd.

Drugi etap ekstrapolacji przebiega podobnie – otrzymujemy przybliżenia $T_2(h)$, $T_2(h/2), \ldots$, których błąd jest szeregiem bez wyrazu z potęgą o wykładniku 4. Na trzecim etapie ekstrapolacji otrzymujemy przybliżenie rzędu 7 określane często jako kwadratura Romberga. Ogólny wzór na kwadraturę obliczaną na etapie i ma postać

$$T_i(h_j) = \frac{4^{j-1}T_i(h_{j-1}) - T_{i-1}(h_{j-1})}{4^{j-1} - 1}.$$

Choć liczba etapów jest teoretycznie nieograniczona, to w praktyce najczęściej ogranicza się ją do 3. To zagadnienie zilustrujemy w Eksperymencie 3.1.

Zainteresowanych pominiętymi tu szczegółami algorytmu metody Romberga, a opisanymi w wielu pozycjach literatury z dziedziny metod numerycznych, odsyłamy do np. [3, 4].

3.2.3. Algorytmy adaptacyjne

Dotychczas rozważaliśmy algorytmy rozwiązywania problemu numerycznego, w których wybór metody i ustalenie jej parametrów (np. rzędu lub kroku) poprzedzały numeryczne rozwiązywanie problemu. Dokładność rozwiązania jest zdeterminowana początkowym wyborem metody i parametrów.

W przeciwieństwie do tego sposobu, przy zastosowaniu algorytmów adaptacyjnych, a priori wybieramy metodę i zadajemy granice dopuszczalnego błędu. Natomiast parametry metody są dobierane automatycznie w trakcie rozwiązywania problemu tak, aby błąd rozwiązania mieścił się w założonym zakresie, a koszt obliczeniowy był możliwie najmniejszy. Bardziej zaawansowane algorytmy adaptacyjne obejmują także autonomiczny wybór metody, a nie tylko jej parametrów.

W algorytmach adaptacyjnych pojawia się więc konieczność oceny błędu rozwiązania. Jeżeli błąd jest większy niż dopuszczalny, to podwyższany jest rząd metody p lub skracany krok (długość podprzedziału) h. Pojawia się pytanie, czy działaniami odwrotnymi (obniżenie rzędu, wydłużenie kroku) można zmniejszyć koszt obliczeniowy. W algorytmach adaptacyjnych powszechnie stosowanych do kwadratur taki problem nie pojawia się, bo obliczenia aproksymacji prowadzone są w kierunku od aproksymacji zgrubnych do coraz bardziej dokładnych i są kończone w chwili osiągnięcia błędu tolerowanego.

Ocena błędu

Zacznijmy od sposobów sprawdzania, czy dokładność kwadratury, osiągnięta przy aktualnej wartości kroku i rządu, odpowiada założonej tolerancji.

Błąd kwadratur NC może być zapisany wzorami (3.6) lub (3.7), ale oszacowanie wartości tego błędu wymaga oszacowania wartości $n+2$ lub $n+1$ pochodnej funkcji f. Znajomość pochodnych wysokiego rzędu, a dodatkowo oszacowanie ich wartości jest w praktyce obliczeniowej utrudnieniem wykluczającym użyteczność tego sposobu.

Ominięcie tej przeszkody jest możliwe, gdy tę samą całkę przybliżamy dwoma kwadraturami tego samego rzędu. Ekstrapolacja Richardsona (wspomniana w podrozdziale 1.1) polega na usuwaniu głównej części błędu przybliżenia. Algorytm ten można uzupełnić o obliczenie wartości usuwanej części błędu. W przypadku dostatecznie małego h można przyjąć, że będzie to wystarczająco dokładne oszacowanie całego błędu.

Algorytm jest elementarny. Z układu równań (1.8) i (1.9) rugujemy wartość dokładną A^* i otrzymujemy przepis na główną część błędu przybliżenia $A(h/2)$

$$\frac{a_0 h^{k_0}}{2^{k_0}} \approx \frac{A\left(\frac{h}{2}\right) - A(h)}{2^{k_0}-1}. \tag{3.9}$$

Jest to przybliżona wartość błędu bezwzględnego. Używając wyrażenia (1.1) jako granicy dopuszczalnego błędu, warunek spełnienia wymaganej dokładności można zapisać w postaci

$$\frac{\left|Q\left[f;\frac{h}{2}\right] - Q[f;h]\right|}{2^{k_0}-1} < \max\left(\texttt{RelTol}\cdot\left|Q\left[f;\frac{h}{2}\right]\right|, \texttt{AbsTol}\right). \tag{3.10}$$

Kryterium nie posługuje się wartościami błędów ani górnym ograniczeniem błędu (do tego konieczne byłoby oszacowanie z góry pochodnej odpowiedniego rzędu funkcji podcałkowej), lecz tylko przybliżonymi wartościami głównej części błędów. Powstaje więc niebezpieczeństwo, że kryterium to będzie spełnione, mimo że faktyczny błąd będzie przekraczał zadaną wielkość i nie ma gwarancji osiągnięcia zadanej dokładności. Dlatego stosuje się dodatkowe „zabezpieczenia", do których powrócimy przy prezentacji listingu przykładowej procedury.

Algorytm modyfikujący liczbę podprzedziałów

Rozważmy przypadek, gdy wybraną już metodę chcemy wykorzystać w algorytmie adaptacyjnym. Algorytm najpierw oblicza kwadraturę na długim podprzedziale. Jeżeli nie ma ograniczenia od góry, to rozpoczyna od całego przedziału całkowania. Na dalszych etapach algorytm skraca długość h podprzedziału (przez jego połowienie) do momentu osiągnięcia wymaganej dokładności. Długość h nigdy nie jest, z punktu widzenia dokładności i kosztu, za krótka, nie ma więc potrzeby wydłużania jej.

Wybór sposobu skracania długości h przez połowienie podprzedziału[4] jest podyktowany minimalizacją kosztu obliczeniowego. Jeżeli wybrana kwadratura jest oparta na 2^n+1

[4] Buduje się też algorytmy stosujące podział na trzy równej długości podprzedziały.

węzłach – np. Simpsona (NC2), Boole'a (NC4), NC8, ... NC2^n – to obliczenie kwadratur dla każdej z dwóch połówek podprzedziału będzie wymagało obliczenia wartości funkcji podcałkowej tylko w 2^{n-1} „nowych" węzłach (mniej niż połowie wszystkich węzłów), przy wykorzystaniu wartości funkcji w pozostałych węzłach obliczonych we wcześniejszych iteracjach. Przykładowo w kwadraturze Simpsona ($n = 1$) jest to jeden „nowy" węzeł, a w kwadraturze NC8 ($n = 3$) cztery węzły.

W środowiskach obliczeniowych dopuszczających rekurencyjne wywoływanie funkcji algorytm ma przejrzystą strukturę. Omówimy ją w skrócie, a szczegóły można znaleźć w listingu przedstawionym poniżej. Na każdym poziomie, począwszy od pierwszego, rekurencyjnie wywoływana procedura dzieli zadany (wejściowy) przedział całkowania na – nazwijmy – lewą i prawą połówkę. Po wykonaniu obliczeń kwadratur dla obydwóch połówek ich suma jest porównywana z kwadraturą obliczoną na poprzednim poziomie dla całego wejściowego przedziału. Jeżeli porównanie to wskazuje, że błąd mieści się w tolerowanym zakresie, to następuje powrót z rekurencji. W przeciwnym razie procedura jest wywoływana dwukrotnie – raz dla lewej połówki i drugi raz dla prawej. Jeżeli rekurencja osiąga maksymalny dopuszczalny poziom bez dostatecznego obniżenia błędu, to wyjście z procedury nie zawiera obliczonego wyniku.

Poniższy listing w języku C jest zapisem algorytmu w wersji rekurencyjnej. Dla uproszczenia przyjęliśmy, że wybrana została metoda Simpsona, a adaptacji podlegają długości przedziałów.

```
#define LEVEL_MAX 10;
double fun(double x) {
  return 1/((x-0.3)*(x-0.3)+0.001)+1/((x-0.9)*(x-0.9)+0.004)-6;
}
double oblicz_rek(double a,double fa,double c,double fc,double b,
                  double fb,double tol,int poziom){
  double c1,c2,fc1,fc2,Q0,Q1,Q2,Q12,R;
  Q0 = (b-a)/6 * (fa +4*fc + fb);
  c1 = (a+c)/2;  fc1 = fun(c1); Q1 = (c-a)/6 * (fa +4*fc1 + fc);
  c2 = (b+c)/2;  fc2 = fun(c2); Q2 = (b-c)/6 * (fc +4*fc2 + fb);
  Q12 = Q1 + Q2;
  if(fabs(Q12-Q) < tol)  return Q12;
//  R  = (16*Q12 - Q0)/15;
//  if(fabs(R-Q12) < tol)  return R;
  if(poziom > LEVEL_MAX)  return NaN;
  return oblicz_rek(a,fa,c1,fc1,c,fc,tol/2,poziom+1)
       + oblicz_rek(c,fc,c2,fc2,b, fb,tol/2,poziom+1);
}
int main(void) {
  double a=0., b=1., c=(a+b)/2, tol=1e-5;
  int poziom = 0;
  calka = oblicz_rek(a,fun(a),c,fun(c),b,fun(b),tol,poziom);
  return 0;
}
```

Listing ten nie zawiera kliku ważnych w praktyce obliczeniowej elementów, np. sygnalizacji sytuacji wyjątkowych. Przekroczenie maksymalnego poziomu rekurencji jest tu sygnalizowane tylko zwróceniem symbolu nieoznaczonego `NaN` jako wyniku. Nie ma też kontroli minimalnej długości podprzedziału.

Rozbudowy wymaga test zakończenia rekurencji. Przedstawiona jest wersja najbardziej elementarna. W komentarzu zapisano alternatywny, dokładniejszy sposób oceny błędu wynikający z wzoru (3.9). Testy te powinny uwzględniać kontrolę błędu względnego i bezwzględnego, np. jak we wzorze (3.10).

Jednak oszacowania te nie są pewne, tzn. nie gwarantują, że faktyczny błąd nie przekracza wartości szacowanej. Dlatego stosuje się dodatkowe „zabezpieczenia". Jeżeli funkcja podcałkowa zmienia znak, to możliwy jest przypadek, gdy wartości całek na podprzedziałach są duże, ale na całym przedziale redukują się do wartości bliskich zeru. Pojawi się opisywany w podrozdziale 1.1 błąd wynikający z utraty najbardziej znaczących cyfr wyniku odejmowania. Bezpiecznym rozwiązaniem, ale powiększającym koszt, jest znalezienie miejsc zerowych funkcji podcałkowej i obliczanie kwadratur osobno w każdym przedziale tego samego znaku funkcji.

Jeżeli wymagamy zmniejszenia ryzyka „przypadkowego" spełnienia testu błędu na „za wczesnym" poziomie rekurencji, to możemy rozbudować algorytm o żądanie spełnienia warunku na dopuszczalny błąd na dwóch lub więcej sąsiednich poziomach rekurencji – oczywiście powiększając koszt obliczeniowy.

Powszechnie stosowanym zabezpieczeniem przed mylącym oszacowaniem błędu kwadratury dla funkcji okresowych – np. $\int_0^1 [\cos(2^n \pi) - 1] dx$ – jest podział całego przedziału całkowania na kilka podprzedziałów o niewspółmiernych długościach.

W przypadku mniej złożonych funkcji podcałkowych należy rozważyć, czy iteracyjna wersja algorytmu (z własnym stosem) nie będzie mniej kosztowna obliczeniowo od wersji rekurencyjnej.

Algorytm adaptacji rzędu metody

W poprzednim punkcie była mowa o wersji z eliminacją głównej części błędu realizowaną na jednym z etapów ekstrapolacji Richardsona. W rezultacie otrzymane przybliżenie jest wynikiem zastosowania metody wyższego rzędu niż użyta tam przykładowo metoda Simpsona. Rozszerzenie tej idei prowadzi do modyfikacji przedstawionego wcześniej schematu Romberga.

Można byłoby w tym miejscu zadać pytanie, dlaczego w algorytmie adaptacji liczby podprzedziałów wcześniej obliczone wartości funkcji są rozdzielane między dwie połówki podprzedziału i po dodaniu nowo obliczonych wartości funkcji wykorzystane do obliczenia kwadratur tego samego rzędu, skoro jest obliczana kwadratura wyższego rzędu, $2n$? Taką ideę realizuje adaptacyjna metoda Romberga. Jest to przykład metody zmiennokrokowej i zmiennorzędowej. Poniżej zostanie przedstawiony zarys tej koncepcji, natomiast jej szczegóły można znaleźć np. w [3, 4].

W algorytmie adaptacyjnym z użyciem metody Romberga nie zakładamy z góry rzędu dokładności wyniku. Wstępne przybliżanie wartości całki kwadraturami trapezowymi ograniczamy do dwóch kwadratur: $T_0(h)$ i $T_0(h/2)$. To umożliwia wyznaczenie kwadratury $T_1(h)$ rzędu 3. i wystarcza do oceny głównej części błędu. Jeżeli błąd ten jest dostatecznie mały, to ekstrapolację kończymy na tym etapie. W przypadku przeciwnym obliczamy kwadraturę $T_0(h/4)$, która pozwala wyznaczyć $T_2(h)$ rzędu 5. Test błędu, jak poprzednio, decyduje, czy procedurę kontynuować w kierunku podwyższania rzędu i obniżania błędu. Istotnym ograniczeniem jest szybki wzrost kosztu obliczeniowego po osiągnięciu kwadratury Romberga $T_4(h)$.

Szersze niż w przypadku kwadratur możliwości adaptacji dokładności drogą zmiany rzędu metody są wykorzystywane w algorytmach adaptacyjnych dla zagadnień początkowych w równaniach różniczkowych omawianych w podrozdziale 5.5.2.

Przypadek szczególny

Punkt ten jest dygresją odbiegającą od głównego nurtu rozważań – opisuje przypadek, gdy wszystkie dostępne wartości funkcji zostały wyznaczone przed obliczaniem całki, np. gdy dane pochodzą z wykonanych już pomiarów, zrealizowanych w równych odległościach zmiennej niezależnej, np. czasu.

Szkic algorytmu korzystającego ze schematu Romberga mógłby wyglądać następująco:

- start w celu oceny błędu zacząć od kwadratur opartych na pierwszych dziewięciu pomiarach – trapezowych i Simpsona,
- ocena błędu kwadratury,
- porównanie z błędem danych, jeżeli błąd kwadratur jest większy, to kontynuować schemat Romberga, a w przeciwnym przypadku przejść do następnych ośmiu punktów,
- Jeżeli są przesłanki o zmienności wartości i błędu danych, to w dalszych punktach sprawdzać, czy trzeba dostosować rząd.

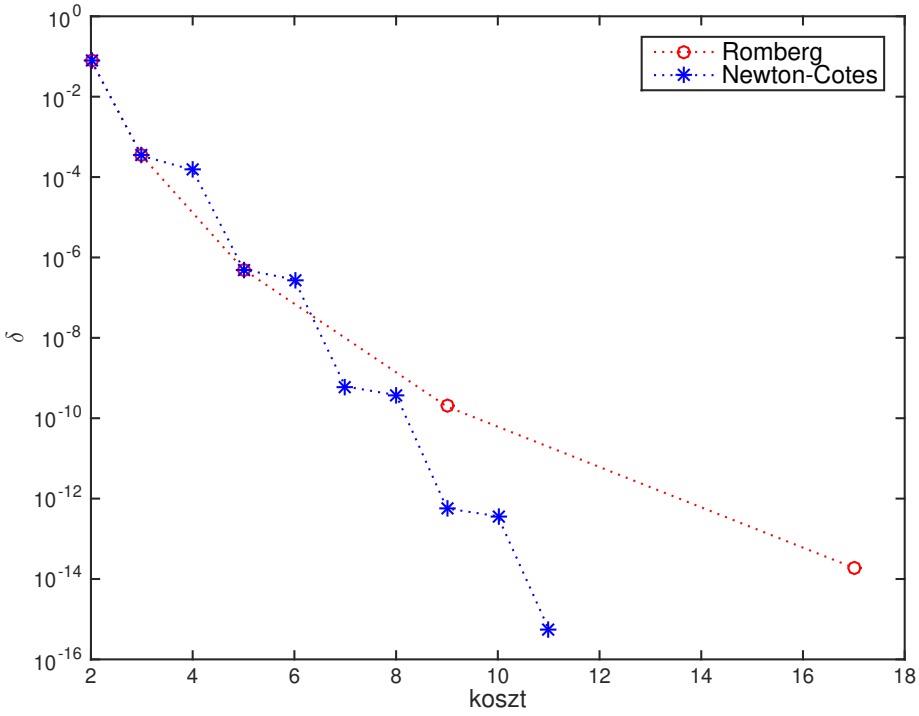
3.2.4. Eksperymenty obliczeniowe

Eksperyment 3.1. Porównanie błędu zamkniętych kwadratur NC i Romberga

Obliczamy numeryczne przybliżenia całki z funkcji wykładniczej $f(x) = \exp(x)$ w przedziale $[1,2]$. Na przedziale tym są równomiernie rozłożone węzły x_k, $k = 1,\ldots,n$. W każdej próbie jest obliczona jedna kwadratura obejmująca cały przedział całkowania, tzn. nie stosujemy kwadratur złożonych.

Obliczane są kwadratury NC1–NC10. Stanowią one odniesienie dla wyników metody Romberga. Węzły dla pierwszej kwadratury trapezowej są ustawione na granicach całkowania. W kolejnych etapach schematu Romberga dodawane są kolejne węzły i obliczane według schematu Romberga kwadratury – kolejno Simpsona, Boole'a i Romberga. Metoda jest przedłużona (w stosunku do powszechnie prezentowanych czterech etapów) do etapu piątego, dla 17 węzłów.

Błąd względny δ, który został wykreślony na rysunku 3.1, jest odniesiony do wartości wyrażenia analitycznego ewaluowanego z dokładnością maszynową. Koszt obliczeniowy jest mierzony liczbą koniecznych ewaluacji funkcji podcałkowej, czyli jest wprost liczbą użytych węzłów.



Rys. 3.1. Zależność błędu względnego kwadratur Romberga i NC od kosztu obliczeniowego

Widoczny na rysunku 3.1 wykres błędu potwierdza następujące teoretyczne rozważania.

– Kwadratury Romberga oraz kwadratury NC, które są oparte na tych samych węzłach, tzn. drugim, trzecim oraz piątym, to są te same kwadratury (trapezów, Simpsona, Boole'a) i zgodnie z oczekiwaniami ich błąd jest taki sam. Jest to błąd metody, a szum numeryczny jest niezauważalny.

– Na podstawie porównania osiąganego poziomu błędu przy dziewięciu węzłach (równego 10^{-12} w przypadku NC8 i 10^{-10} w przypadku kwadratury Romberga) można potwierdzić znaną prawidłowość: dla liczby węzłów powyżej 6 efektywność kwadratur NC jest lepsza od efektywności uzyskiwanej metodą Romberga. Dla podkreślenia tego zjawiska, na wykresie umieszczono także wynik 5. etapu schematu Romberga, dla 17 węzłów.

Eksperyment 3.2. Porównanie błędu złożonych kwadratur NC

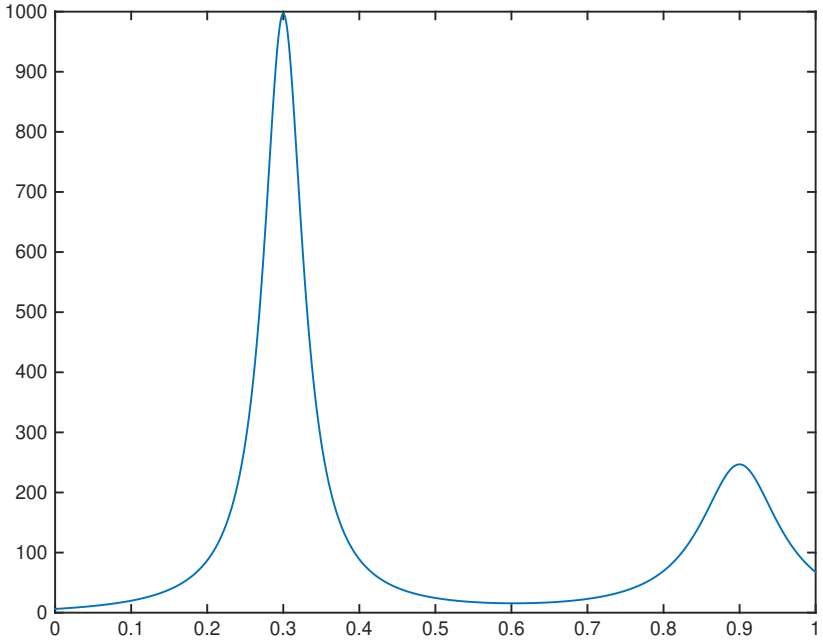
Jako przykład funkcji podcałkowej posłuży funkcja (rys. 3.2) zaproponowana przez Forsythe'a i współautorów [5] z parametrami $a = 0,001$, $b = 0,004$.

$$\int_0^1 \left(\frac{1}{(x-0,3)^2+a} + \frac{1}{(x-0,9)^2+b} - 6 \right) dx. \tag{3.11}$$

Wiedząc, że

$$\int_0^1 \frac{dx}{(x-x_0)^2+a} = \frac{1}{\sqrt{a}} \left(\operatorname{arctg} \frac{1-x_0}{\sqrt{a}} + \operatorname{arctg} \frac{x_0}{\sqrt{a}} \right),$$

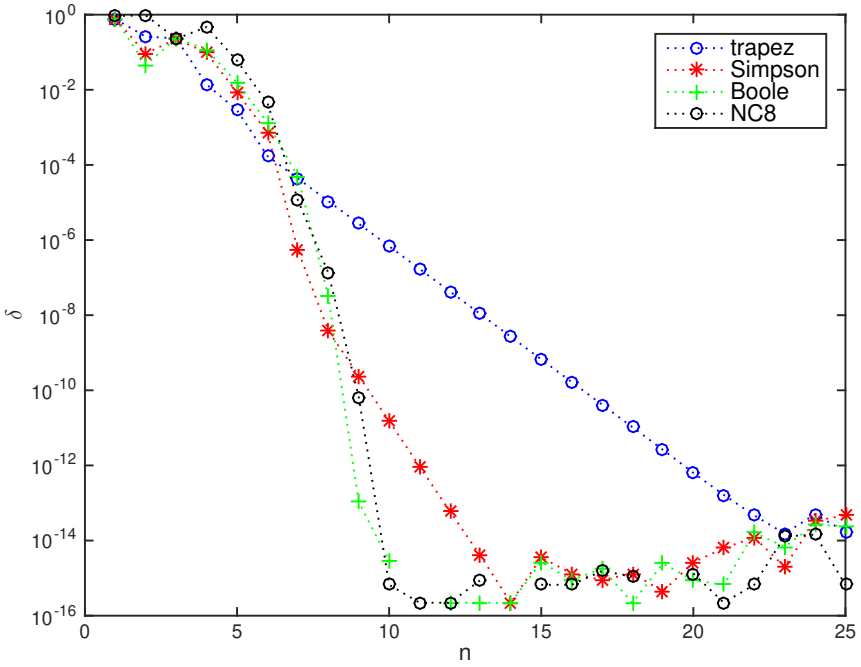
wartość dokładną całki można obliczyć analitycznie.



Rys. 3.2. Wykres funkcji podcałkowej

Na całym przedziale całkowania rozmieszczono $2^n + 1$ równoodległych węzłów. W kolejnych próbach n wzrasta o 1, a więc między każde dwa sąsiednie węzły dodawany jest nowy węzeł, zatem zagęszczenie węzłów zwiększa się dwukrotnie.

Eksperyment polega na numerycznym obliczeniu kwadratur NC i – przez wykorzystanie znanej wartości dokładnej – obliczeniu błędu względnego. Na rysunku 3.3 pokazano wykresy bezwzględnych wartości (ze względu na skalę logarytmiczną) tego błędu.



Rys. 3.3. Zależność błędu względnego złożonych kwadratur NC od kosztu.
Liczba węzłów w przedziale całkowania równa $2^n + 1$

Wykresy wskazują na trzy zjawiska.

1) Jeżeli odległość h między sąsiednimi węzłami jest stosunkowo duża, to wszystkie badane kwadratury są obciążone podobnym błędem. Oznacza to, że eliminowanie początkowych wyrazów szeregu Taylora nie pomniejsza znacząco błędu, bo nie jest on skupiony w elemencie z najniższą potęgą h.
2) Od liczby kroków rzędu $2^{10} \approx 1000$, czyli dla długości kroku $h < 0,001$, błąd kwadratur wyższych rzędów (Boole'a i NC8) jest zdominowany szumem numerycznym, a skracanie kroku jest bezcelowe. Kwadratura trapezowa osiąga tę granicę w $h \approx 10^{-7}$.
3) Kwadratury wyższych rzędów zmniejszają błąd w stosunkowo wąskim przedziale długości kroku h – i zmniejszanie to jest tu „radykalne" (dziesięciokrotne skrócenie kroku powoduje zmniejszenie błędu o dziesięć rzędów wielkości).

Porównanie powyższych obserwacji z wynikami eksperymentów z ilorazami różnicowymi wskazuje na podobieństwa zakresów pojawiania się szumu numerycznego. Wyniki te są – jakościowo – powszechnie znane, ale mniej powszechna jest wiedza o stronie ilościowej tego zjawiska.

Uwaga ogólna: Wykonanie pojedynczego eksperymentu nie uprawnia do formułowania ogólnych wniosków. Przedstawione wyniki są tylko ilustracją pewnych zjawisk, istotnych w praktyce obliczeniowej.

3.3. Kwadratury z węzłami nierównoodległymi

Zacznijmy od przypomnienia kilku informacji o wielomianach ortogonalnych.

Ciąg wielomianów $\mathbf{P}_n^*$: $P_0(x), P_1(x), \ldots, P_n(x)$ jest **ortogonalny** z ciągłą funkcją wagową $w(x) \geq 0$ na przedziale $[a,b]$, jeżeli

$$\int_a^b w(x)P_k(x)P_j(x)dx \begin{cases} \neq 0 & \text{dla } k = j, \\ = 0 & \text{dla } k \neq j. \end{cases}$$

Twierdzenie 3.2. *Wielomiany ortogonalne mają tylko pierwiastki rzeczywiste, jednokrotne, leżące w przedziale* (a,b).

Kwadratury z funkcją wagową. Rozważamy przybliżenie całki funkcji f z funkcją wagową w kwadraturą o współczynnikach β_k i węzłach w x_k:

$$\int_a^b w(x)f(x)dx \approx \sum_{k=0}^{n} \beta_k f(x_k). \tag{3.12}$$

Współczynniki β kwadratury są obliczane przez scałkowanie wielomianu interpolacyjnego Lagrange'a

$$\beta_k = \int_a^b w(x) \prod_{j=0, j\neq k}^{n} \frac{x - x_j}{x_k - x_j} dx$$

i zależą tylko od funkcji wagowej i od rozkładu węzłów, a nie zależą od funkcji $f(x)$.

Kwadratury Gaussa–Laquerre'a i Gaussa–Hermite'a (i inne, niewymienione w tabeli 3.2) pozwalają obliczać przybliżone wartości całek niewłaściwych ze względu na granice całkowania, ale nie znajdują szerokiego zastosowania w algorytmach rozwiązywania równań różniczkowych. Dlatego w dalszej części rozdziału będziemy omawiać kwadratury z wagą $w(x) \equiv 1$, tj. kwadratury typu Gaussa–Legendre'a.

3.3.1. Kwadratury Gaussa

Ideę poszukiwania maksymalnego rzędu kwadratury można interpretować następująco. Szukamy takich współczynników wagowych β_k i takiego rozkładu węzłów x_k, dla których równość (przybliżona) (3.12) jest spełniona dokładnie, gdy funkcją f jest dowolny wielomian jak najwyższego stopnia. Wartość całki (wyrażenia lewej strony (3.12)) jest punktem w przestrzeni współczynników wielomianów. Wartość kwadratury (prawej strony (3.12)) jest punktem w przestrzeni $(2n+2)$-wymiarowej ($n+1$ współczynników β_k i $n+1$ węzłów). Szukamy jednoznacznego odwzorowania przestrzeni całki z wielomianów w przestrzeń kwadratur. Zatem wymiar przestrzeni całki nie może być większy od $2n+2$, czyli stopień wielomianu nie może być większy od $2n+1$. Precyzyjnie wyraża to twierdzenie w [6, str. 74].

Twierdzenie 3.3. *Dla współczynników*

$$\beta_k = -\frac{a_{n+1}}{a_n}\frac{1}{p^*_{n+1}(x_k)p^{*\prime}_n(x_k)},$$

gdzie x_k są zerami wielomianu ortogonalnego p^*_n, a a_i jest współczynnikiem przy x^i wielomianu p^*_i, równość

$$\int_a^b w(x)f(x)dx = \sum_{k=0}^{n}\beta_k p(x_k) \tag{3.13}$$

jest spełniona dla każdego wielomianu $p(x)$ klasy $\mathtt{P}_{2n+1}$, a wielomiany p^*_{n+1} i p^*_n są wielomianami ortogonalnymi[5] *odpowiadającymi funkcji wagowej $w(x) \geq 0$.*

Jeżeli funkcja wagowa $w(x) \equiv 1$, to dla funkcji f innych niż wielomiany klasy $\mathtt{P}_{2n+1}$ błąd kwadratury (3.12) wynosi

$$E_n(f,h) = \int_a^b w(x)f(x)dx - \sum_{k=0}^{n}\beta_k p(x_k) = \frac{(n!)^4 h^{2n+1}}{(2n+1)[(2n)!]^3} f^{(2n)}(\xi), \tag{3.14}$$

gdzie $h = b-a$, $a < \xi < b$.

Twierdzenie 3.4. *Kwadraturą o maksymalnym rzędzie $(2n+1)$ jest kwadratura interpolacyjna, której węzłami są pierwiastki wielomianu ortogonalnego na przedziale $[a,b]$.*

Przykładowe zestawienia przedziału całkowania, funkcji wagowej i rodzaju wielomianu ortogonalnego zawiera tabela 3.2.

Tabela 3.2
Zestawienie popularnych kwadratur Gaussa

$[a,b]$	$w(x)$	Symbol	Nazwa kwadratury
$[-1,1]$	1	$P_n(x)$	Gaussa–Legendre'a
$(-1,1)$	$\frac{1}{\sqrt{1-x^2}}$	$T_n(x)$	Gaussa–Czebyszewa I
$[-1,1]$	$\sqrt{1-x^2}$	$U_n(x)$	Gaussa–Czebyszewa II
$[0,\infty]$	e^{-x}	$L_n(x)$	Gaussa–Laguerre'a
$[-\infty,\infty]$	e^{-x^2}	$H_n(x)$	Gaussa–Hermite'a

[5] Symbol $'$ w mianowniku twierdzenia 3.3 oznacza pochodną.

Biorąc pod uwagę obliczenia z ograniczoną dokładnością maszynową, należy zwrócić uwagę na dwie prawidłowości:

1) Współczynniki kwadratur Gaussa[6] są dodatnie: $\beta_k > 0,\ k = 0,1,\ldots,n$. Ryzyko znacznej utraty cyfr znaczących przy odejmowaniu nie jest duże (odejmowania nie można całkowicie wykluczyć, bo funkcja podcałkowa może w przedziale całkowania zmienić znak).
2) Wartości pierwiastków wielomianów ortogonalnych x_k oraz współczynniki kwadratury β_k są liczbami niewymiernymi, więc w obliczeniach musi pojawić się błąd zaokrąglenia.

3.3.2. Kwadratury Lobatta i Radaua

Kwadratury Gaussa z częścią węzłów poza pierwiastkami wielomianu ortogonalnego są rozważane np. w [6] w podrozdziale 2.7.1. Jeżeli zażądamy ustawienia m węzłów u_k w punktach innych niż to przewiduje (3.13), to przybliżenie całki Gaussa opartej na $m+n$ węzłach zapisuje się za pomocą dwóch sum

$$\int_{-1}^{1} w(x)f(x)dx \approx \sum_{k=1}^{m} a_k f(u_k) + \sum_{k=0}^{n} w_k f(x_k), \tag{3.15}$$

gdzie stałe a_k, w_k i x_k są dobrane tak, aby przybliżona równość (3.15) była spełniona dokładnie dla wielomianów możliwie najwyższej $(m+2n+1)$ klasy.

Trzy popularne przypadki, do których powrócimy w rozdziale 5, to:

1) $m = 1$ – wymuszenie położenia jednego węzła w $u_1 = -1$ (kwadratura Radaua-I),
2) $m = 1$ – wymuszenie położenia jednego węzła w $u_1 = 1$ (kwadratura Radaua-II),
3) $m = 2$ – dwa węzły umieszczone w $u_1 = -1$ i $u_2 = 1$ (kwadratura Lobatta-III).

Błąd przybliżenia wyraża się wzorem analogicznym do (3.14) ze zmniejszonym wykładnikiem potęgi h i rzędem pochodnej – o 1 w przypadku kwadratur Radaua, a o 2 w kwadraturach Lobatta.

Eksperyment 3.3. Porównanie błędu i kosztu

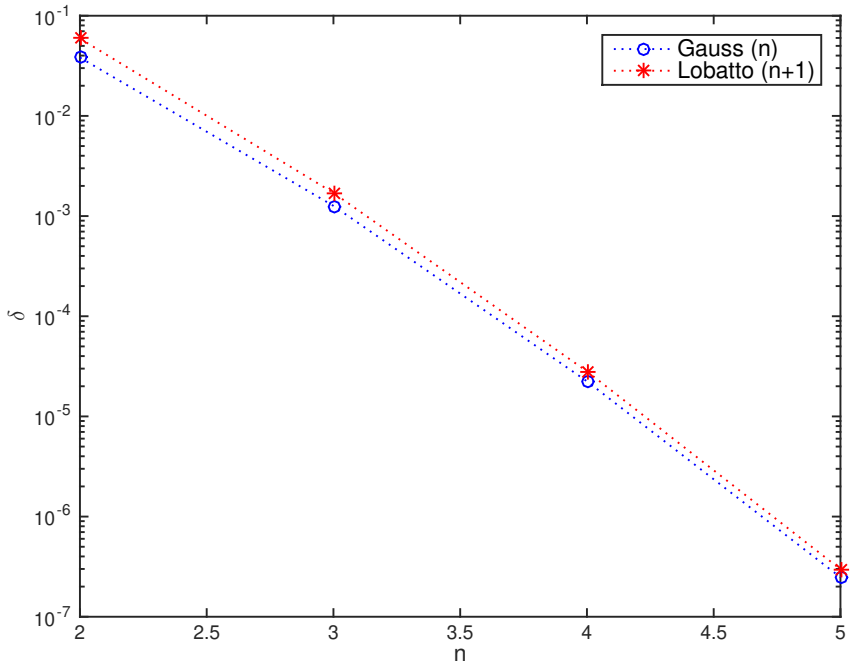
Celem eksperymentu jest ilustracja zależności relacji między kosztem a błędem kwadratur Gaussa i Lobatta-III od długości kroku. Eksperyment 3.3 przebiegał analogicznie jak Eksperyment 3.2. Zastosowano kwadratury w wersji złożonej. Cały przedział całkowania był dzielony n razy, za każdym razem liczba podprzedziałów wzrastała dwukrotnie. Całka na każdym podprzedziale była przybliżana kwadraturą Gaussa opartą na dwóch węzłach lub kwadraturą Lobatta-III opartą na trzech węzłach. Po każdym podziale były obliczane wartości funkcji w węzłach, w przypadku kwadratury Gaussa położonych w innych miejscach niż wcześniej wyznaczane. Stosując wybraną („3-węzłową”) kwadraturę Lobatta, można byłoby

[6] Dla kwadratur NC tylko dla $n \leq 7$ oraz $n = 9$.

wyliczać wartość funkcji tylko w jednym nowym węźle, bo wartości w dwóch pozostałych były obliczone wcześniej – podobnie jak w przypadku kwadratury Simpsona (bo w istocie kwadratury Simpsona i Lobatta-III są identyczne). Jest to przypadek szczególny, więc tej oszczędności nie uwzględniono.

Druga redukcja kosztu występuje zawsze w kwadraturach złożonych (a w konsekwencji także w zastosowaniach do rozwiązywania równań różniczkowych, omawianych w rozdziale 5). Polega na nakładaniu się węzłów na granicach przylegających przedziałów, co prowadzi do możliwości zmniejszenia liczby obliczeń wartości funkcji o 1 w każdym podprzedziale. Dobór do porównania tej pary kwadratur podyktowany był właśnie zrównaniem kosztu – w obu przypadkach wymagają dwóch nowych węzłów w każdym podprzedziale. Rząd obu kwadratur też jest taki sam, równy dla kwadratury Gaussa $(n = 1)$ $2 \cdot n + 1$, a dla kwadratury Lobatta $(m = 2, n = 0)$ $2 + 2 \cdot 0 + 1$, równy 2.

Wykres na rysunku 3.4 wskazuje na mniejszy błąd kwadratury Gaussa niż Lobatta. Różnica ta wynika z mniejszej wartości stałej błędu. Błąd szumu numerycznego nie jest zauważalny. Lepsza efektywność kwadratur Lobatto opartych na nieparzystej liczbie węzłów pojawia się, gdy jest uwzględniona możliwość korzystania z węzła centralnego obliczonego na wcześniejszym etapie – przesunięcie na wykresie punktów Lobatta o jednostkę w lewo wskazywałoby na zmniejszenie błędu o około dwa rzędy wielkości, przy tym samym koszcie obliczeniowym.



Rys. 3.4. Porównanie błędu kwadratur Gaussa i Lobatta przy wyrównanym koszcie obliczeniowym

3.3.3. Algorytmy adaptacyjne

W kwadraturze Gaussa punkty, w których są obliczane wartości funkcji podcałkowej, są wyznaczone zerami wielomianu ortogonalnego p_n^* (3.15). Przy założeniu, że funkcja wagowa $w(x) \equiv 1$, w kwadraturach Lobatta punkty te są zerami pochodnej tego wielomianu[7].
W kwadraturach Radaua-I[8] punkty te są zerami funkcji

$$\frac{p_{n-1}^*(x) + p_n^*(x)}{x-1}.$$

Wartości β_k i x_k są dostępne w wielu publikacjach, np. [6].

Z powyższych wzorów oraz z własności wielomianów ortogonalnych wynika, że położenie punktów x_k jest inne w każdej kwadraturze (poza punktem centralnym i ewentualnymi punktami na granicach przedziału).

Podobnie jak w przypadku węzłów równoodległych, konieczna jest ocena błędu i ewentualna jego korekta przez modyfikację kroku lub zmianę rzędu kwadratury. Wszystkie te operacje wymagają dodatkowego obliczenia innej kwadratury. W przypadku węzłów równoodległych można dobrać jako drugą kwadraturę taką, która w części (ponad połowie) jest oparta na węzłach pierwszej. W przypadku węzłów nierównoodległych takie „powtórne wykorzystanie" jest ograniczone do węzłów centralnych (jeżeli są w obydwóch) i ewentualnymi węzłami na końcach przedziału (w przypadku kwadratur Radaua lub Lobatta). Obliczanie wartości funkcji w pozostałych węzłach drugiej kwadratury podnosi koszt obliczeniowy.

Schemat Gaussa–Kronroda

Kwadratura Gaussa jest kwadraturą, która przy zadanej liczbie węzłów osiąga maksymalny rząd. Jeżeli konieczne jest obliczenie przybliżenia tej samej całki kwadraturą Gaussa innego rzędu, to musi to być kwadratura oparta na innych węzłach – położenia węzłów każdej kwadratury Gaussa nie mogą się nakładać, z wyjątkiem węzłów centralnych, jeżeli takie występują.

Jeżeli z góry wiadomo, że dla jednej całki należy obliczyć dwie kwadratury, np. do wyznaczenia przybliżenia całki i oceny błędu[9], to (przy n i $n+1$ węzłach) wybór kwadratur Gaussa prowadzi do obliczeń funkcji w $2n+1$ punktach i wyniku rzędu $2n+1$. Taka relacja rzędu do liczby obliczeń funkcji obowiązuje dla kwadratur NC z liczbą węzłów 2^k+1 oraz w pierwszych trzech krokach (etapach) schematu Romberga. Przewagą kwadratur Gaussa pozostaje już tylko większa elastyczność doboru liczby węzłów.

Schemat Kronroda jest propozycją poprawy relacji rzędu do kosztu w kwadraturach Gaussa.

[7] Nie licząc punktów na granicach przedziału.
[8] Przy dodatkowym założeniu, że przedziałem całkowania jest $[-1,1]$.
[9] Przy założeniu, że obliczany błąd jest błędem kwadratury niższego rzędu.

Szkic schematu Kronroda [6, str. 82] wygląda następująco.

1. Startujemy z węzłami $u_0, u_1, \ldots, u_n$ dobranymi według metody Gaussa.
2. Otrzymujemy przybliżenie rzędu $2n+1$.
3. Dodajemy węzły $x_0, x_1, \ldots, x_{n+1}$.
4. Otrzymujemy[10] metodę rzędu $3n+4$.

Przykładami algorytmów stosujących kwadratury Gaussa są – obecnie uznawany za najbardziej efektywny – algorytm opublikowany przez Lauriego [7] oraz kwadratura Gaussa–Lobatta implementowana w procedurze MATLAB-a `integral` (zob. https:/ / www.ams.org / journals / mcom / 1997-66-219 / S0025-5718-97-00861-2 / S0025-5718-97-00861-2.pdf).

Eksperyment 3.4. Porównanie algorytmów adaptacyjnych

Celem eksperymentu jest ilustracja ewolucji algorytmów całkowania numerycznego na przykładzie środowiska obliczeniowego MATLAB. Porównano cztery relacje między kosztem a błędem czterech adaptacyjnych, rekurencyjnych procedur (w nawiasie podano zastosowaną kwadraturę):

1) `quad` (Simpsona),
2) `quad8` (NC8),
3) `quadl` (Lobatta),
4) `quadgk` (Gaussa–Kronroda).

Należy zaznaczyć, że pierwsze dwie procedury są już wycofane z biblioteki, a w tym eksperymencie służą jako punkty odniesienia dla współczesnych metod.

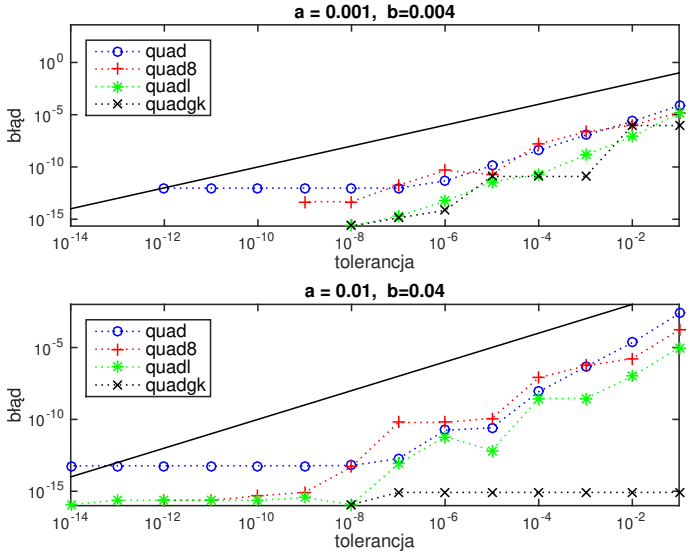
Obliczana jest ta sama całka (3.11), ale z dwoma zestawami parametrów. Algorytmom adaptacyjnym nie narzucamy długości kroku (jak w poprzednim eksperymencie), lecz wymaganą dokładność. W pierwszej części doświadczenia zobaczymy, jaki jest w praktyce „margines dokładności" otrzymywanych wyników. Na rysunku 3.5 przedstawiono wykres zależności błędu od zadanej tolerancji (względnej i bezwzględnej na tym samym poziomie).

Górny wykres przedstawia wyniki całkowania funkcji o wartościach maksimów lokalnych dziesięciokrotnie większych niż funkcji całkowanej na wykresie dolnym.

W obu przypadkach odległości między dopuszczalnym błędem a błędem faktycznym sięgają kilku (około trzech) rzędów wielkości. Jedynym wyjątkiem jest ustabilizowanie się błędu procedury `quad` na jednym poziomie. Jest to spowodowane (sygnalizowanym przez procedurę) osiągnięciem granicy dopuszczalnej głębokości rekursji (10. poziomu).

Na dolnym wykresie widać pewną odmienność przebiegu błędu algorytmu Gaussa–Kronroda. Nawet przy dużym dopuszczalnym błędzie wynik obliczeń redukuje błąd do poziomu błędu maszynowego. Wymaga to wyższego kosztu obliczeniowego, co pokazuje eksperyment następny.

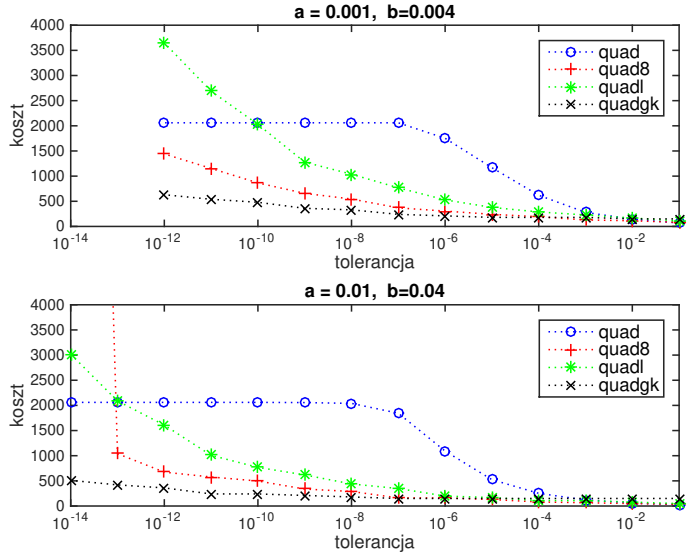
[10] Suma z węzłami u_i musi być obliczona powtórnie, bo wagi są inne niż w kwadraturze Gaussa.



Rys. 3.5. Zależność błędu względnego od zadanej tolerancji. Linią ciągłą zaznaczono maksymalny błąd określony zadaną tolerancją

Eksperyment 3.5. Zależność kosztu obliczeniowego od zadanej tolerancji

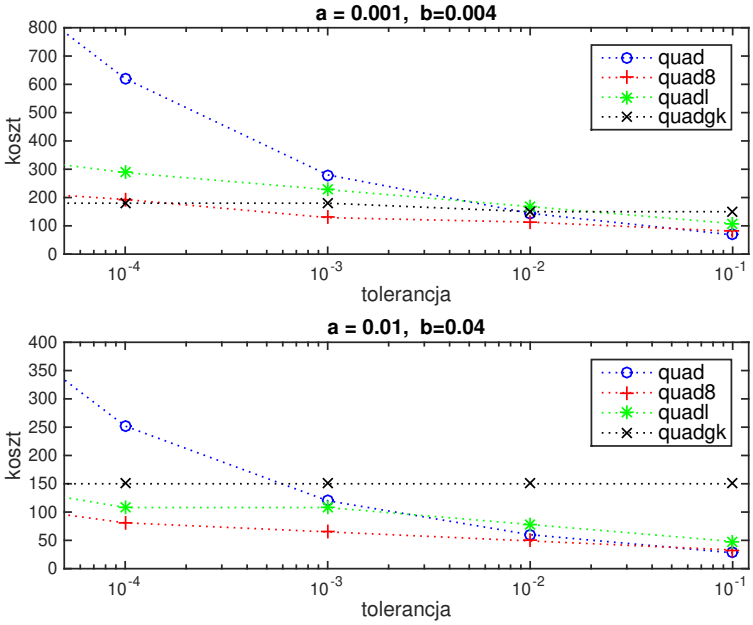
Wyniki poprzedniego doświadczenia posłużyły do przedstawienia na rysunku 3.6 zależności kosztu obliczeniowego od zadawanej dokładności.



Rys. 3.6. Zależność kosztu obliczeniowego od zadanej tolerancji

Widoczny jest wyższy koszt algorytmu z kwadraturą Lobatta niż z kwadraturą NC8, ale – w porównaniu z błędem widocznym na rysunku 3.5 – błąd jest mniejszy.

Ważniejsza obserwacja dotyczy najmłodszego algorytmu (`quadqk`) z kwadraturą Gaussa–Kronroda: w porównaniu z innymi użytymi tu procedurami daje wynik o małym błędzie przy niskim koszcie. Ta prawidłowość nie obejmuje przypadków tolerowania stosunkowo dużych błędów. Rysunek 3.7 pokazuje w powiększeniu zakres zadawanej tolerancji, w którym koszty tej procedury są wyższe od kosztów w pozostałych algorytmach. Można więc mieć wątpliwość, czy celowe jest stosowanie tej metody, gdy nie jest wymagana duża dokładność.



Rys. 3.7. Zależność kosztu obliczeniowego od zadanej tolerancji – w zakresie dużych wartości tolerancji

3.4. Podsumowanie

Kwadratury korzystające z wartości funkcji podcałkowej obliczanych w punktach równomiernie rozłożonych w przedziale całkowania są efektywne (relacja kosztu do dokładności jest korzystna) w przypadkach niskich wymagań dokładności. W przypadkach gdy wartości funkcji podcałkowej nie mogą być wyznaczone dla dowolnego argumentu, np. nie są obliczane w trakcie wyznaczania kwadratury, lecz są zadane (wcześniej obliczone lub zmierzone), to najczęściej są jedynymi metodami wyznaczania przybliżonej wartości całki.

Efektywność kwadratur bazujących na nierównomiernym rozkładzie punktów, w których znane są wartości funkcji, jest porównywalna z efektywnością kwadratur z rozkładem

równomiernym w zakresie niskich rzędów obu metod. Zalety kwadratur z rozkładem nierównomiernym (takie jak niższy koszt i mniejszy błąd) pojawiają się, gdy konieczne jest użycie kwadratur wyższych rzędów – gdy złożoność funkcji jest większa i wyższe są wymagania dokładności.

Algorytmy adaptacyjne wykonują dodatkowe obliczenia konieczne do oceny błędu i modyfikacji parametrów metody. Ten wzrost kosztu może być zrekompensowany (często z nadwyżką) zmniejszeniem liczby ewaluacji wartości funkcji podcałkowej, ale w przypadkach funkcji „spokojnych" (bez radykalnych różnic zmienności w przedziale całkowania, gdy już niski rząd metody sprowadza błąd do tolerowanego zakresu) bilans zysku i kosztu adaptacji może okazać się negatywny.

Algorytmy adaptacyjne są łatwiejsze w stosowaniu. Nie wymagają od użytkownika decyzji o doborze wartości parametrów metody, a jedynie żądanej dokładności. Trzeba jednak podkreślić konieczność upewnienia się, czy wybrany algorytm adaptacyjny gwarantuje spełnienie warunków zadanej tolerancji błędu – w większości przypadków gwarancji takiej nie ma. Jeżeli przekroczenie granicy tolerowanego błędu jest niedopuszczalne, to lepszym podejściem niż stosowanie algorytmu adaptacyjnego jest zbadanie konkretnej funkcji na tyle dokładnie, aby użytkownik sam mógł ustalić bezpieczne parametry metody. Należy tu zauważyć, że w rozdziale tym nie były rozważane inne metody gwarantujące zadaną dokładność, np. obliczenia w arytmetyce przedziałowej.

Bibliografia

[1] Young D. M., Gregory R. T., *A survey of numerical mathematics*, Addison Wesley, Reading, MA 1972.

[2] Isaacson E., Keller H., *Analysis of numerical methods*, Wiley, New York 1966.

[3] Burden R., Faires J., *Numerical Analysis*, 3rd ed., Prindle, Weber & Schmidt–KENT, Boston 1985.

[4] Kincaid D., Cheney W., *Analiza numeryczna*, przeł. S. Paszkowski, Wydawnictwa Naukowo-Techniczne, Warszawa 2006.

[5] Forsythe G., Malcolm M., Moler C., *Computer methods for mathematical computations*, Prentice-Hall, Inc., Englewood Cliffs, NJ 1977.

[6] Davis P., Rabinowitz P., *Methods of Numerical Integration*, Computer Science and Applied Mathematics, Academic Press, New York – San Francisco – London 1975.

[7] Laurie D. P., *Calculation of Gauss–Kronrod quadrature rules*, „Mathematics of Computation", 1997, 66 (219), 1133–1145, doi: https://doi.org/10.1090/S0025-5718-97-00861-2.