Tomáš Ondruch, 3eMAI/2

Rombergova metoda numerické integrace

Úvod

Rombergova metoda integrace se používá pro numerický výpočet hodnoty určitého integrálu $\int_a^b f(x)dx$. Metoda byla představena v roce 1955 německým matematikem Wernerem Rombergem. Typově se jedná o *Newton-Cotesovu formuli* a jde o vylepšení *lichoběžníkové formule* s použitím extrapolace. Podrobněji se popisu metody věnuje následující odstavec.

Princip metody

V tomto odstavci bude stručně popsán princip metody a budou zmíněny vybrané pojmy, které se k dané metodě pojí. Vychází se z platnosti *Eulerova-Maclaurinova vzorce*, který popisuje vztah mezi aproximací integrálu $Q_T^n(f)$ a skutečnou hodnotou integrálu $I(f)$, kde dolní index *T* u aproximace napovídá, že se vychází z lichoběžníkové formule (angl. *trapezoidal rule*). Dále vzorec obsahuje také hodnoty (spojitých) derivací funkce $f(x)$, velikost kroku h a *Bernoulliova čísla* B. U funkce $f(x)$ je nutno zdůraznit, že počet spojitých derivací musí být dostatečně velký. Při srovnání Eulerova-Maclaurinova vzorce s mocninným rozvojem extrapolované funkce $F(h)$ dle teorie pro *opakovanou Richardsonovu extrapolaci* lze pozorovat značná typová podobnost obou vzorců. Skutečně, vzorce jsou shodné, pokud položíme $F(h) \coloneqq Q_T^n(f)$, $h \coloneqq \frac{b-a}{n}$, $a_0 \coloneqq I(f)$, $p_i \coloneqq 2i$, kde a, b představují integrační meze intervalu, n je počet dílků dělení intervalu, a_0 je koeficient u absolutního členu v mocninném rozvoji funkce $F(h)$ a p_i jsou přirozená čísla, konkrétně hodnoty exponentů v mocninném rozvoji funkce $F(h)$. Jelikož máme nyní úlohu zcela „převedenu“ na problém opakované Richardsonovy extrapolace, lze pro další postup určování hodnoty integrálu využít algoritmus právě této metody. Ten spočívá ve vytvoření tabulky pro členy T, kde $T_{si} = F_{i+1}(h_{s-1})$, $h_s = hq^{-s}$, $s = 0,1,\dots,q > 1$. Tabulka pro T_{si} je vyplňována postupně po řádcích, kdy obecně v s-tém řádku jsou počítány členy $T_{s0}, T_{s1}, \dots, T_{si}$, $i = 0,1,\dots s$. Průběžně je prováděno srovnání se zadanou absolutní a relativní tolerancí přesnosti (ε_A, resp. ε_R), konkrétně $\left|\frac{(T_{si}-T_{si-1})}{(4^i-1)}\right| < \max(\varepsilon_R|T_{si}|, \varepsilon_A)$. Je-li nerovnost splněna, pak se hodnota integrálu položí $I(f) \approx T_{si}$ a algoritmus výpočtu je ukončen. V opačném případě výpočet pokračuje počítáním dalších členů T_{si}, dokud nerovnost není splněna.

Využití metody a její srovnání s dalšími vybranými metodami

Použití Rombergovy metody pro výpočet hodnoty integrálu $\int_a^b f(x)dx$ je velmi vhodné, klademe-li vysoké požadavky na přesnost výpočtu. Rombergova metoda má řád diskretizační chyby $O(h^{2m+2})$ – pro $m = 0$ dostáváme *složené lichoběžníkové pravidlo*, $m = 1$ odpovídá *složenému Simpsonovu pravidlu*, $m = 2$ se vztahuje ke *složenému Boolovu pravidlu*. U Rombergovy metody je ovšem obecně $m \in \mathbb{N}$, tj. může být dosaženo vyšší přesnosti. V kritériu přesnosti výpočtu Rombergově metodě konkuruje pouze metoda *Gaussových kvadraturních formulí*, u nichž se ovšem setkáme s vyšší časovou náročností výpočtu. Stejný problém se objevuje také u Simpsonova složeného pravidla. Další výhodou Rombergova pravidla je snadné nastavení velikosti ekvidistantního kroku, případně počtu dílků dělení. Jeho nevýhodou v některých případech může být nemožnost nastavení nerovnoměrného dělení intervalu.

V minulosti již také byly vytvořeny další variace Rombergovy metody, které na rozdíl od základního algoritmu nevycházejí z lichoběžníkového pravidla. Byly tak vytvořeny algoritmy pro *Romberg-Simpsonovu metodu*, *rozšířenou Romberg-Simpsonovu metodu* či *Romberg-Gaussovu metodu*. Pomocí těchto metod bylo spočítáno několik určitých integrálů a následně byly porovnány přesnosti vypočítaných hodnot. Romberg-Simpsonova metoda dosáhla nejlepších výsledků. Důkladnější popis tohoto výzkumu však přesahuje rámec semestrální práce, proto nyní přejdu k dalšímu odstavci, který se bude zabývat vypracovaným skriptem pro MatLab.

Vypracovaný program

V prostředí MatLab byl vytvořen program, jehož algoritmus realizuje výpočet hodnoty určitého integrálu na daném intervalu právě pomocí algoritmu *Rombergovy integrace.* Při tvorbě algoritmu byly použity poznámky a stručné úryvky kódu, které jsou k nalezení v pdf-dokumentu *Vybrané statě z numerických metod*, který vytvořil doc. RNDr. Libor Čermák, CSc. Funkčnost programu byla mimo jiné ověřena na vzorovém příkladu, kterému se věnuje následující část textu.

Vzorový příklad – nastavení a popis fáze testování

Veškeré vstupy, tj. integrovaná funkce f, dolní mez a, horní mez b, počáteční počet dílků dělení n_0 a požadovaná přesnost ε, jsou pro jednoduchost a bezproblémové spouštění skriptu obsaženy přímo ve zdrojovém kódu programu. Potřebné výstupy, kterými jsou tabulka hodnot T_{si}, získaná aproximace numericky vypočítané hodnoty integrálu I a absolutní chyba výpočtu $deltaI$, jsou pro uživatele připraveny k uložení.

Vzorové volání programu v prostředí MatLab:

```
[TsiTab, I, deltaI]=rombergOndruch
```

, kde `deltaI` slouží ke srovnání vypočítaného řešení s přesnou hodnotou integrálu. Vypočítá se jako

```
deltaI = abs(I-exactVal);
```

, kde `exactVal` představuje známou přesnou hodnotu integrálu.

Pro důkladné ověření funkčnosti programu byl skript ve fázi ladění testován na různé varianty zadání. Nejčastější problém při vytváření komplexnějších zkušebních příkladů bylo stanovení přesné hodnoty integrálu, které bylo prováděno analyticky „na papíře“. Při tvorbě vzorového příkladu byl však nakonec zvolen jiný postup.

Vzorový příklad - zadání

Motivací k vytvoření příkladu bylo zadání, které bylo nalezeno na internetu. Jedná se o výpočet integrálu v chybové funkci

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt$$

pro $x = 1$.

Vstupními daty tedy jsou:

```
f = @(x) (2/sqrt(pi))*exp(-x*x);
a = 0;
b = 1;
n0 = 20; %libovolně zvoleno
epsil = 1e-8; %libovolně zvoleno
exactVal=erf(1);
```

, přičemž bylo ověřeno, že výsledek `erf(1)` v MatLabu je roven hodnotě nalezené v tabulce hodnot pro chybovou funkci.

Vzorový příklad – výsledky

Po spuštění programu v okně MatLabu se v okně vypisuje index s aktuálního řádku vyplňované tabulky T_{si}. Po ukončení výpočtu vzorového příkladu se vypíší výsledky:

```
TsiTab =

   0.842527817080724                   0                   0
   0.842657551684586   0.842700796552540                   0
   0.842689982802334   0.842700793174917   0.842700792949742

I =

   0.842700792949742

deltaI =

     2.708944180085382e-14
```

Při analýze získaných výsledků si lze povšimnout, že hodnota `deltaI` je velmi malá a nižší než `1e-8`, tedy výpočet proběhnul při splnění požadované tolerance. Bylo také ověřeno, že při volbě vyššího počátečního počtu dílků dělení n_0 se velikost tabulky hodnot T_{si} zmenšuje a pro n_0 v řádu tisíců se také významně vylepšuje přesnost výpočtu, kdy dosažená absolutní chyba `deltaI` je značně malá oproti zadané toleranci ε. V takovém případě ovšem z důvodu nutnosti počítání funkčních hodnot ve všech bodech dělení znatelně roste časová náročnost výpočtu.

Jelikož při testování programu nebylo objeveno zadání, pro které by program při výpočtu selhal, lze považovat vypracovaný skript za funkční a připravený k praktickému použití. V úvahu samozřejmě připadá také případná optimalizace kódu k dosažení nižší časové a paměťové náročnosti.