4. Obyčejné diferenciální rovnice - počáteční úlohy

V této kapitole se budeme zabývat problematikou numerického řešení úloh s počátečními podmínkami pro obyčejné diferenciální rovnice.

4.1. Formulace, základní pojmy

Počáteční problém pro ODR. *Obyčejná diferenciální rovnice* (ODR) se nazývá rovnice *n–tého řádu* (ODRn), jestliže nejvyšší derivace neznámé funkce jedné proměnné je n–tého řádu. Obecný tvar rovnice prvního řádu je $g(x, y(x), y'(x)) = 0$. Budeme předpokládat, že z rovnice je možné explicitně vyjádřit $y'(x)$, tj. že rovnici lze psát ve tvaru

$$y'(x) = f(x, y(x)). \tag{4.1}$$

Připomeňme, že obecné řešení rovnice (4.1) obsahuje integrační konstantu, která může nabýt libovolné hodnoty. Proto k jednoznačnému určení $y(x)$ nestačí sama rovnice. Potřebujeme znát hodnotu řešení pro jisté $x = a$, tj.

$$y(a) = \eta. \tag{4.2}$$

Podmínka (4.2) se nazývá *počáteční* a znamená, že integrální křivka $y = y(x)$ prochází bodem $[a, \eta]$.

Lipschitzova podmínka. Je-li v nějakém okolí D bodu $[a, \eta]$ funkce $f(x, y)$ spojitá a splňuje-li v tomto okolí *Lipschitzovu podmínku* s konstantou L, tj. platí-li

$$|f(x, u) - f(x, v)| \le L|u - v| \quad \forall\, [x, u], [x, v] \in D, \tag{4.3}$$

pak bodem $[a, \eta]$ prochází jediné řešení $y(x)$ rovnice (4.1). Jestliže má funkce $f(x, y)$ v okolí D ohraničenou parciální derivaci $|\partial_y f(x, y)| \le L$, Lipschitzova podmínka (4.3) platí. To je přímý důsledek věty o střední hodnotě:

$$f(x, u) - f(x, v) = \partial_y f(x, \xi)(u - v),$$

kde $[x, \xi]$ je nějaký vnitřní bod úsečky s krajními body $[x, u]$ a $[x, v]$.

Jiný standardní výsledek říká, že když je funkce $f(x, y)$ spojitá na $\langle a, b\rangle \times \mathbb{R}$ a splňuje tam Lipschitzovu podmínku, pak počáteční problém (4.1),(4.2) má jediné řešení definované v celém intervalu $\langle a, b\rangle$.

Rovnice (4.1) spolu s počáteční podmínkou (4.2) se nazývá *počáteční úloha* nebo také *Cauchyova úloha.*

Soustava obyčejných diferenciálních rovnic 1. řádu je soustava n rovnic pro n neznámých funkcí $y_1(x), \ldots, y_n(x)$ tvaru

$$y_j' = f_j(x, y_1, \ldots, y_n), \quad j = 1, 2, \ldots, n.$$

Počáteční podmínky jsou tvaru

$$y_j(a) = \eta_j, \quad j = 1, 2, \ldots, n.$$

Počáteční úlohu zapíšeme stručněji vektorově:

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}), \quad \mathbf{y}(a) = \boldsymbol{\eta}, \tag{4.4}$$

kde

$$\mathbf{y} = (y_1, y_2, \ldots, y_n)^T,\ \mathbf{y}' = (y_1', y_2', \ldots, y_n')^T,\ \mathbf{f}(x, \mathbf{y}) = (f_1(x, \mathbf{y}), f_2(x, \mathbf{y}), \ldots, f_n(x, \mathbf{y}))^T.$$

Existence a jednoznačnost řešení. Je-li v okolí D bodu $[a, \boldsymbol{\eta}]$ funkce $\mathbf{f}(x, \mathbf{y})$ spojitá a splňuje tam Lipschitzovu podmínku s konstantou L, tj. platí-li

$$\|\mathbf{f}(x, \mathbf{u}) - \mathbf{f}(x, \mathbf{v})\| \le L\|\mathbf{u} - \mathbf{v}\| \tag{4.5}$$

pro všechny body $[x, \mathbf{u}]$ a $[x, \mathbf{v}]$ z okolí D, prochází bodem $[a, \boldsymbol{\eta}]$ jediné řešení počáteční úlohy (4.4). Je-li $D = \langle a, b\rangle \times \mathbb{R}^n$, jediné řešení existuje v celém intervalu $\langle a, b\rangle$.

Z věty o střední hodnotě plyne

$$\mathbf{f}(x, \mathbf{u}) - \mathbf{f}(x, \mathbf{v}) = \mathbf{J}(x, *)(\mathbf{u} - \mathbf{v}),$$

kde $\mathbf{J}(x, *) = \{\partial f_i(x, \boldsymbol{\xi}_i)/\partial y_j\}_{i,j=1}^n$ je *Jacobiova matice* funkce $\mathbf{f}(x, \mathbf{y})$, jejíž složky v i-tém řádku jsou vyhodnoceny v nějakém vnitřním bodě $[x, \boldsymbol{\xi}_i]$ úsečky s koncovými body $[x, \mathbf{u}]$ a $[x, \mathbf{v}]$. Jsou-li všechny parciální derivace $\partial \mathbf{f}/\partial \mathbf{y}$ v D ohraničené, tj. platí-li

$$\left|\frac{\partial f_i(x, \mathbf{y})}{\partial y_j}\right| \le L \quad \forall\, [x, \mathbf{y}] \in D \text{ a pro } i, j = 1, 2, \ldots, n,$$

je Lipschitzova podmínka (4.5) zřejmě splněna.

Rovnice n-tého řádu. Každou rovnici n-tého řádu tvaru

$$y^{(n)} = F(x, y, y', \ldots, y^{(n-1)}) \tag{4.6}$$

s počátečními podmínkami

$$y(a) = \eta_1, \quad y'(a) = \eta_2, \ldots, y^{(n-1)}(a) = \eta_n$$

lze snadno převést na soustavu (4.4). Stačí položit $y_1 = y$, $y_2 = y'$, $\ldots$, $y_n = y^{(n-1)}$, dále $f_j = y_{j+1}$, $j = 1, 2, \ldots, n-1$ a $f_n = F(x, y_1, y_2, \ldots, y_n)$.

V dalším budeme vždy předpokládat, že uvažovaná počáteční úloha má v intervalu $\langle a, b\rangle$ jediné řešení. Budeme také předpokládat, že funkce $\mathbf{f}(x, \mathbf{y})$ splňuje Lipschitzovu podmínku a má tolik spojitých derivací, kolik jich v dané situaci bude zapotřebí.

Jedna rovnice prvního řádu s jednou neznámou funkcí je v aplikacích méně významná než soustavy rovnic. Metody přibližného řešení se však snadněji odvodí pro jednu rovnici a lze je aplikovat bezprostředně i na soustavy. Také analýza numerických metod je pro jednu rovnici podstatně snadnější. Proto se v následujícím výkladu někdy omezíme jen na jednu rovnici. Z velkého množství metod uvedeme jen některé a to takové, které jsou pro své dobré vlastnosti široce používány.

4.2. Úvod do numerických metod řešení

Numerickým řešením počáteční úlohy rozumíme výpočet přibližných hodnot hledaného řešení $y(x)$ v bodech x_i dosti hustě vykrývajících interval $\langle a, b\rangle$. Nechť tedy $a = x_0 < x_1 < \cdots < x_N = b$ je dostatečně jemné *dělení* intervalu $\langle a, b\rangle$. Body x_i nazýváme *uzly*, vzdálenost $h_i = x_{i+1} - x_i$ dvou sousedních uzlů nazýváme *délkou kroku* (stručně *krokem*), a množinu $\{x_0, x_1, \ldots, x_N\}$ všech uzlů nazýváme *sítí*. Jsou-li všechny kroky stejně dlouhé, tj. $h_i = h := (b-a)/N$, hovoříme o *rovnoměrném* (nebo také *ekvidistantním*) dělení intervalu $\langle a, b\rangle$. V tom případě je $x_i = a + ih$, $i = 0, 1, \ldots, N$. Hodnotu přesného řešení v uzlu x_i budeme značit $y(x_i)$ a hodnotu přibližného řešení y_i. Jestliže se nám podaří najít přibližné řešení y_i, $i = 0, 1, \ldots, N$, můžeme přibližně vypočítat hodnotu řešení $y(x)$ v libovolném bodě $x \in \langle a, b\rangle$ interpolací.

Numerická metoda pro řešení počáteční úlohy (4.1),(4.2) je předpis pro postupný výpočet aproximací $y_1, y_2, \ldots, y_N$, $y_0 = \eta$ vždy. Metoda se nazývá *k-kroková*, závisí-li předpis pro výpočet aproximace y_{i+1} na předchozích aproximacích $y_i, y_{i-1}, \ldots, y_{i+1-k}$. Speciálně *jednokroková metoda* počítá přibližné řešení y_{i+1} v uzlu x_{i+1} jen pomocí znalosti přibližného řešení y_i v uzlu x_i, přibližná řešení y_{i-1}, $y_{i-2}, \ldots$ spočtená v předchozích uzlech x_{i-1}, $x_{i-2}, \ldots$ nepoužívá.

Explicitní Eulerova metoda. Nejjednodušší numerickou metodou pro řešení úlohy (4.1),(4.2) je *explicitní Eulerova metoda*, stručně EE metoda. EE metodu snadno odvodíme z Taylorovy formule

$$y(x_{i+1}) = y(x_i+h) = y(x_i)+hy'(x_i)+\tfrac{1}{2}h^2y''(\xi_i), \quad \xi_i \in (x_i, x_{i+1}). \tag{4.7}$$

Uvážíme-li, že $y'(x_i) = f(x_i, y(x_i))$, a zanedbáme-li člen $\frac{1}{2}h^2y''(\xi_i)$, dostaneme formuli

$$y(x_{i+1}) \approx y(x_i)+hf(x_i, y(x_i)). \tag{4.8}$$

Tento výsledek je však nepoužitelný, protože neznáme hodnotu $y(x_i)$ a navíc se v tomto vztahu vyskytuje znaménko přibližné rovnosti. Proto zaměníme ve formuli (4.8) výrazy $y(x_i)$ a $y(x_{i+1})$ jejich přibližnými hodnotami y_i, y_{i+1} a znaménko přibližné rovnosti $\approx$ nahradíme znaménkem rovnosti. EE metoda je tedy předpis

$$y_{i+1} = y_i+hf(x_i, y_i) \tag{4.9}$$

pro $i = 0, 1, \ldots, N-1$. $y_0 = y(x_0) = \eta$ je přesné.

Diskretizační chyby. Do bodu $[x_1, y_1]$, kde $y_1 = y_0 + hf(x_0, y_0)$, se dostaneme posunem po tečně sestrojené k přesnému řešení $y(x)$ ve startovacím bodě $[x_0, y_0]$. Posouváme se doprava tak dlouho, až $x_1 - x_0 = h$, kde h je zvolený krok. Nový bod $[x_1, y_1]$ už na integrální křivce $y(x)$ neleží, podle (4.7) je $y(x_1) - y_1 = \frac{1}{2}h^2y''(\xi_0)$. Výraz $\frac{1}{2}h^2y''(\xi_0)$ označíme jako le_0 a nazveme *lokální chybou* (*le* jako *local error*) v uzlu x_1.

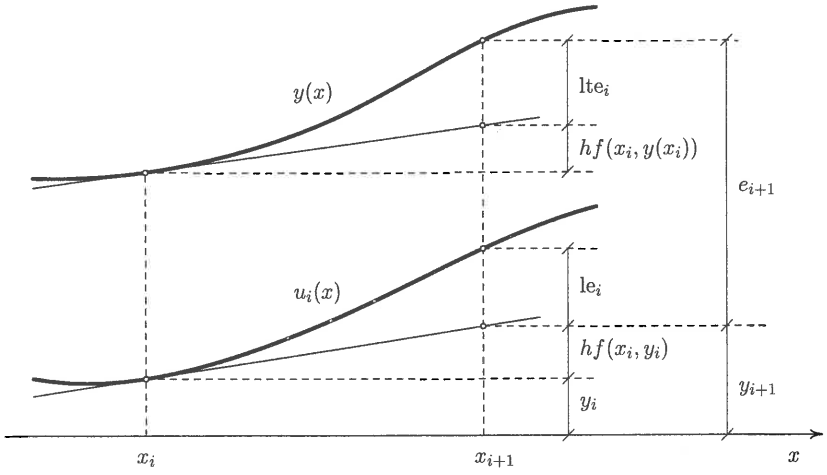
Dále spočteme $y_2 = y_1+hf(x_1, y_1)$. Vidíme, že do bodu $[x_2, y_2]$ se dostaneme posunem po přímce $y - y_1 = f(x_1, y_1)(x - x_1)$. Tato přímka je tečnou k integrální křivce procházející bodem $[x_1, y_1]$, tj. k funkci $u_1(x)$, kde

$$u_1' = f(x, u_1), \quad u_1(x_1) = y_1.$$

Funkci $u_1(x)$ nazveme *lokálním řešením* úlohy (4.1),(4.2) příslušným uzlu x_1. Do bodu $[x_2, y_2]$ se tedy dostaneme po tečně sestrojené k lokálnímu řešení $u_1(x)$ v bodě $[x_1, y_1]$. Nový bod $[x_2, y_2]$ na integrální křivce $u_1(x)$ zase neleží,

$$u_1(x_2) = y_1+hf(x_1, y_1)+\frac{1}{2}h^2u_1''(\eta_1), \quad \text{kde } \eta_1 \in (x_1, x_2),$$

takže $u_1(x_2) = y_2 + le_1$, kde $le_1 = \frac{1}{2}h^2u_1''(\eta_1)$ je lokální chyba v uzlu x_2. Celková chyba $y(x_2) - y_2$ po dvou krocích se označuje e_2 a nazývá se *globální diskretizační chyba* (*global truncation error*). Globální diskretizační chyba je důsledkem hromadění účinků vzniklých chyb lokálních. Říkáme, že globální diskretizační chyba vzniká kumulací lokálních chyb. Zřejmě $e_1 = le_0$ avšak $le_0 + le_1 \approx e_2$ je pouze aproximace e_2.



Obr. 4.1. Diskretizační chyby

Stejně postupujeme dál. Předpokládejme, že jsme už spočetli přibližné řešení y_i v uzlu x_i. Sestrojíme lokální řešení $u_i(x)$ definované předpisem

$$u_i' = f(x, u_i), \quad u_i(x_i) = y_i,$$

a bod $[x_{i+1}, y_{i+1}]$ dostaneme posunem po tečně sestrojené k lokálnímu řešení $u_i(x)$ v bodě $[x_i, y_i]$. Přitom vznikne lokální chyba $le_i = u_i(x_{i+1}) - y_{i+1}$, pro kterou platí

$$u_i(x_{i+1}) = u_i(x_i)+hf(x_i, u_i(x_i))+le_i, \text{ kde } le_i = \tfrac{1}{2}h^2u_i''(\eta_i) \text{ a } \eta_i \in (x_i, x_{i+1}). \tag{4.10}$$

Celková chyba $e_{i+1} = y(x_{i+1}) - y_{i+1}$ vznikne kumulací lokálních chyb le_j pro $j = 0, 1, \ldots, i$.

Možná jste si všimli zdánlivé nesrovnalosti: u lokální chyby le_i jsme nepoužili slůvko diskretizační. To byl úmysl, spojení *lokální diskretizační chyba* (*local truncation error*) je rezervováno pro výraz $lte_i = \frac{1}{2}h^2y''(\xi_i)$ definovaný rovností (4.7), tj.

$$y(x_{i+1}) = y(x_i)+hf(x_i, y(x_i))+lte_i, \text{ kde } lte_i = \tfrac{1}{2}h^2y''(\xi_i) \text{ a } \xi_i \in (x_i, x_{i+1}). \tag{4.11}$$

Obě chyby le_i a lte_i jsou téhož řádu řádu $O(h^2)$, stejné však nejsou. Lokální chyba le_i je chyba, které se dopustíme v jednom kroku metody. Lokální diskretizační chybu lte_i lze charakterizovat podobně: je to chyba, které se dopustíme v jednom kroku metody, ovšem za tzv. *lokalizačního předpokladu*, že $y_i = y(x_i)$ je přesné. Lokální chyba je tedy chyba, která při výpočtu danou metodou v každém kroku skutečně vzniká. Lokální diskretizační chyba vyjadřuje chybu, které se dopustíme, když do formule dosadíme přesné řešení. V dalším si však ukážeme, že pro malá h je rozdíl mezi oběma chybami prakticky zanedbatelný. Obě lokální chyby le_i a lte_i, spolu s globální chybou e_{i+1}, jsou zakresleny v obrázku 4.1.

Řád, konvergence. Řekneme, že *metoda je řádu p*, jestliže pro její lokální diskretizační chybu lte_i platí

$$lte_i = O(h^{p+1}). \tag{4.12}$$

EE metoda je tedy řádu 1. Pro globální diskretizační chybu e_i EE metody lze dokázat

$$e_i = y(x_i) - y_i = O(h). \tag{4.13}$$

Rovnici (4.13) je třeba chápat tak, že existují kladné konstanty h_0 a K takové, že

$$\max_{0\le i\le N} |e_i| \le Kh \quad \text{pro všechna } 0 < h \le h_0 .$$

Zejména tedy

$$\max_{0\le i\le N} |y(x_i) - y_i| \to 0 \quad \text{pro } h \to 0 .$$

Metody s touto vlastností se nazývají *konvergentní metody*. Protože globální diskretizační chyba je řádu $O(h)$ říkáme, že *konvergence EE metody je řádu 1.*

Důkaz konvergence EE metody lze najít například v [34], [35]. Tvrzení (4.13) však lze snadno ověřit v případě, že $f(x, y) = f(x)$ nezávisí na y. Pak totiž

$$y(x_{j+1}) = y(x_j) + hf(x_j) + \tfrac{1}{2}h^2 f'(\xi_j) ,$$
$$y_{j+1} = y_j + hf(x_j) .$$

Odečtením obou rovnic dostaneme $e_{j+1} = e_j + \frac{1}{2}h^2 f'(\xi_j)$, a protože $e_0 = 0$, je

$$e_i = \tfrac{1}{2}h^2[f'(\xi_0) + f'(\xi_1) + \cdots + f'(\xi_{i-1})] .$$

Označíme-li $M = \max_{a\le\xi\le b} |f'(\xi)|$, pak $|e_i| \le \frac{1}{2}h^2 iM \le \frac{1}{2}[hN]Mh = \frac{1}{2}(b-a)Mh$, neboť $Nh = b - a$.

Příklad 4.1. Obecné řešení rovnice $y' = 4x\sqrt{y}$ je $y(x) = (x^2 + C)^2$, počáteční podmínka $y(1) = 4$ určí partikulární řešení $y(x) = (x^2 + 1)^2$. Řešme tuto počáteční úlohu EE metodou na intervalu $\langle 1, 3\rangle$ s různým krokem h. Nejprve zvolme $N = 10$, tj. $h = 0,2$, $x_0 = 1$, $x_1 = 1,2$, $x_2 = 1,4, \ldots, x_{10} = 3$. Z počáteční podmínky máme $y_0 = 4$, dále podle (4.9)

počítáme

$$y_1 = y_0 + hf(x_0, y_0) = y_0 + h4x_0\sqrt{y_0} = 4 + 0,2 \cdot 4 \cdot 1 \cdot \sqrt{4} = 5,6$$
$$y_2 = y_1 + hf(x_1, y_1) = y_1 + h4x_1\sqrt{y_1} = 5,6 + 0,2 \cdot 4 \cdot 1,2 \cdot \sqrt{5,6} = 7,8718$$
$$\ldots$$
$$y_N = y_{N-1} + hf(x_{N-1}, y_{N-1}) = 81,826 ,$$

zatímco přesná hodnota $y(3) = 100$, tedy absolutní chyba pro $x = 3$ je 18,2 a relativní chyba je 18,2%.
Pro $N = 20$ je $h = 0,1$. Z počáteční podmínky máme opět $y_0 = 4$, dále

$$y_1 = y_0 + hf(x_0, y_0) = y_0 + h4x_0\sqrt{y_0} = 4 + 0,1 \cdot 4 \cdot 1\sqrt{4} = 4,8$$
$$y_2 = y_1 + hf(x_1, y_1) = y_1 + h4x_1\sqrt{y_1} = 4,8 + 0,1 \cdot 4 \cdot 1,1 \cdot \sqrt{4,8} = 5,763992$$

atd. až konečně $y_N = 90,40$. Absolutní chyba i relativní chyba se zmenšily přibližně na polovinu, což odpovídá tomu, že EE metoda je řádu 1.

Vztah mezi lokální chybou a lokální diskretizační chybou. Jak už jsme uvedli, chyby le_i a lte_i jsou pro malá h prakticky k nerozlišení. Dokažme si to. Z Taylorova rozvoje dostáváme

$$y(x_{i+1}) = y(x_i) + hy'(x_i) + \tfrac{1}{2}h^2 y''(x_i) + O(h^3) ,$$

takže lokální diskretizační chybu lte_i lze vyjádřit ve tvaru

$$lte_i = \tfrac{1}{2}h^2 y''(x_i) + O(h^3) . \tag{4.14}$$

Člen $\frac{1}{2}h^2 y''(x_i)$ se nazývá *hlavní člen lokální diskretizační chyby* lte_i. Podobně lze lokální chybu le_i vyjádřit ve tvaru

$$le_i = \tfrac{1}{2}h^2 u_i''(x_i) + O(h^3) .$$

Odečtením obou vyjádření dostaneme

$$lte_i - le_i = \tfrac{1}{2}h^2[y''(x_i) - u_i''(x_i)] + O(h^3) .$$

Derivováním rovnice (4.1) obdržíme

$$y''(x) = f_x(x, y) + f_y(x, y)y' = f_x(x, y) + f_y(x, y)f(x, y) ,$$

takže

$$y''(x_i) - u_i''(x_i) = f_x(x_i, y(x_i)) + f_y(x_i, y(x_i))f(x_i, y(x_i)) - \\ -[f_x(x_i, u_i(x_i)) + f_y(x_i, u_i(x_i))f(x_i, u_i(x_i))] ,$$

a odtud užitím věty o střední hodnotě po snadné úpravě dostaneme

$$|y''(x_i) - u_i''(x_i)| \le M|y(x_i) - u_i(x_i)| = M|y(x_i) - y_i| ,$$

kde konstanta M závisí na prvních a druhých parciálních derivacích funkce $f(x,y)$. Avšak $y(x_i) - y_i = O(h)$ podle (4.13), takže $y''(x_i) - u_i''(x_i) = O(h)$ a

$$lte_i = le_i + O(h^3)\,. \tag{4.15}$$

Protože jak lte_i tak le_i jsou řádu $O(h^2)$, lze pro malá h rozdíl mezi oběma chybami prakticky zanedbat.

Zaokrouhlovací chyby. Doposud jsme vůbec neuvažovali *zaokrouhlovací chyby*. Ukážeme si, že jejich vliv nelze zanedbat. Pro jednoduchost předpokládejme, že

$$\widetilde{y}_0 = \eta\,, \quad \widetilde{y}_{i+1} = \widetilde{y}_i + hf(x_i,\widetilde{y}_i) + \varepsilon_i\,, \quad i = 0,1,\ldots,N-1\,,$$

a nechť $|\varepsilon_i| \le \varepsilon$. Pro $r_i := y_i - \widetilde{y}_i$ proto platí

$$r_{i+1} = r_i + h[f(x_i,y_i) - f(x_i,\widetilde{y}_i)] - \varepsilon_i\,,$$

takže užitím Lipschitzovy podmínky (4.3) dostáváme

$$|r_{i+1}| \le |r_i| + hL|y_i - \widetilde{y}_i| + \varepsilon_i \le (1+hL)|r_i| + \varepsilon\,.$$

Protože $r_0 = 0$, je $|r_1| \le \varepsilon$, $|r_2| \le (1+hL)\varepsilon + \varepsilon$, ...,

$$|r_i| \le \varepsilon[1 + (1+hL) + (1+hL)^2 + \cdots + (1+hL)^{i-1}] =$$
$$\varepsilon\frac{(1+hL)^i - 1}{hL} \le \varepsilon\frac{e^{ihL}-1}{hL} \le K\,\varepsilon h^{-1}\,, \quad \text{kde } K = [e^{(b-a)L} - 1]/L.$$

Pro celkovou chybu EE metody tedy platí

$$\max_{0\le i\le N} |y(x_i) - \widetilde{y}_i| \le K_D h + \varepsilon K_R h^{-1}\,, \tag{4.16}$$

kde h je dostatečně malý krok a K_D,K_R jsou konstanty na h nezávislé. Protože konstanta ε je malá, vliv zaokrouhlování se projeví až pro značně velký počet kroků N (tj. pro velmi malé h), v tom případě může dojít dokonce k převážení účinků zaokrouhlovacích chyb a vypočtené hodnoty $\widetilde{y}_i$ tak mohou být i přes ohromné „výpočetní náklady" zcela bezcenné.

Stabilita. Abychom minimalizovali náklady, snažíme se při reálném výpočtu měnit délku kroku tak, aby přibližné řešení bylo dostatečně přesné a současně aby celkový objem výpočtů byl co nejmenší. Řízení délky kroku je podrobně popsáno v dalších odstavcích. V zásadě však jde o to, jak volit kroky co nejdelší. Vzniká proto přirozená otázka: jak velký krok je ještě možné volit, aby přibližné řešení bylo přijatelné? Z řady dobrých důvodů je účelné pro odpověď na tuto otázku použít testovací úlohu

$$y' = \lambda y\,, \quad y(0) = 1\,, \tag{4.17}$$

kde $\lambda < 0$ je daná konstanta. Co je na této úloze tak zajímavého? Přesné řešení $y(x) = e^{\lambda x}$ je klesající funkce, $e^{\lambda x} \to 0$ pro $x \to +\infty$. Počáteční hodnotu $y(0) = 1$ lze považovat za jednotkovou poruchu a nezávisle proměnnou x za čas. Úlohu (4.17) lze proto interpretovat

jako proces stabilizace: s rostoucím časem dochází k útlumu počáteční poruchy neustálým přibližováním se k nulovému stacionárnímu stavu. Takové chování vykazuje celá řada dějů reálného světa, úloha (4.17) je proto rozumným modelem. Co bychom tedy měli očekávat od numerické metody aplikované na řešení této modelové úlohy? Přijatelné numerické řešení se musí blížit ke stacionárnímu stavu, tedy $y_i \to 0$ pro $i \to \infty$. Nemusí být nutně $y_i > 0$, jde nám totiž přednostně o zmenšování velikosti poruchy, požadujeme však

$$|y_{i+1}| < |y_i|\,, \tag{4.18}$$

což zaručuje, aby utlumování bylo nepřetržité.

EE metoda dá pro testovací úlohu (4.18)

$$y_{i+1} = y_i + h\lambda y_i = (1+h\lambda)y_i = \cdots = (1+h\lambda)^{i+1}y_0\,.$$

Aby platilo (4.18), musí být

$$|1+h\lambda| < 1\,.$$

Předpokládejme nyní, že v úloze (4.17) je λ komplexní číslo a označme $\widehat{h} := h\lambda$. Množina všech $\widehat{h}$, pro která je při řešení testovací úlohy (4.17) zvolenou numerickou metodou splněna podmínka (4.18), se nazývá *oblast absolutní stability* numerické metody. Odvodili jsme, že pro EE metodu je oblast absolutní stability jednotkový kruh $|z+1| < 1$ komplexní roviny se středem v bodě $[-1,0]$. Průnik oblasti absolutní stability s reálnou osou se nazývá *interval absolutní stability*. Pro EE metodu je interval absolutní stability interval $(-2,0)$. Pro reálné $\lambda < 0$ je tedy třeba vybrat $h < 2/|\lambda|$. Řekneme, že metoda je *absolutně stabilní* pro zvolené komplexní číslo λ a krok h, jestliže $\widehat{h} = h\lambda$ leží v oblasti absolutní stability této metody. EE metoda je tedy absolutně stabilní pro λ a h taková, pro která $|h\lambda + 1| < 1$.

Tvar a velikost oblasti absolutní stability metody je spolu s řádem metody základní charakteristikou kvality numerické metody. EE metoda z tohoto pohledu příliš kvalitní není: je pouze řádu 1 a oblast její absolutní stability je malá. EE metoda se používá jen výjimečně. Zabývali jsme se jí především proto, že je velmi jednoduchá a proto na ní lze snadno ukázat, s jakými problémy se při numerickém řešení počátečních úloh pro ODR1 setkáváme a jaké pojmy se při popisu numerických metod běžně používají.

Implicitní Eulerova metoda. Vysvětlení jednoho pojmu jsme ale zatím přeskočili: proč mluvíme o explicitní metodě, co tím máme na mysli? To nejlépe pochopíme, když zavedeme *implicitní Eulerovu metodu* (stručně IE metodu) jako předpis

$$y_{i+1} = y_i + hf(x_{i+1}, y_{i+1})\,. \tag{4.19}$$

Metoda je implicitní proto, že neznámá y_{i+1} je argumentem funkce $f(x,y)$, takže z rovnice (4.19) nelze odvodit explicitní vzoreček y_{i+1} = „něco známého". y_{i+1} je proto třeba určit jako řešení $\widehat{z}$ rovnice

$$g(z) := z - y_i - hf(x_{i+1}, z) = 0\,. \tag{4.20}$$

Je-li $f(x,y)$ lineární funkcí proměnné y, tj. $f(x,y) = a(x)y + b(x)$, je řešení takové rovnice snadné,

$$y_{i+1} = \frac{y_i + hb(x_{i+1})}{1 - ha(x_{i+1})}.$$

Obecně je však třeba řešit rovnici (4.20) přibližně pomocí vhodné iterační metody. To je ve srovnání s EE metodou problém navíc. Aby mělo použití IE metody nějaký smysl, musí mít IE metoda oproti EE metodě také nějakou přednost. Pokusme se ji najít.

Nejdříve prozkoumáme přesnost IE metody. Lokální diskretizační chyba lte_i IE metody je definována předpisem

$$y(x_{i+1}) = y(x_i) + hf(x_{i+1}, y(x_{i+1})) + lte_i,$$

kde $y(x)$ je přesné řešení rovnice (4.1). Z Taylorova rozvoje $y(x_i) = y(x_{i+1}) - hy'(x_{i+1}) + \frac{1}{2}h^2y''(x_{i+1}) + O(h^3)$, a protože $y''(x_{i+1}) = y''(x_i) + O(h)$, dostaneme

$$lte_i = -\tfrac{1}{2}h^2y''(x_i) + O(h^3). \tag{4.21}$$

IE metoda je tedy řádu 1. Srovnáním (4.21) a (4.14) vidíme, že hlavní člen lokální diskretizační chyby IE metody je co do velikosti stejný jako hlavní člen lokální diskretizační chyby EE metody, má však opačné znaménko. Obě metody jsou tedy stejně přesné.

Podívejme se dále na stabilitu IE metody. Pro testovací úlohu (4.17) je

$$y_{i+1} = y_i + h\lambda y_{i+1}, \quad \text{odtud} \quad y_{i+1} = \frac{1}{1-h\lambda}y_i = \cdots = \left(\frac{1}{1-h\lambda}\right)^i y_0$$

a tedy podmínka (4.18) platí pro $|1 - h\lambda| > 1$. Oblast absolutní stability IE metody je tedy obrovská, je to celý vnějšek $|z - 1| > 1$ jednotkové kružnice komplexní roviny se středem v bodě $[1, 0]$. Pro $\mathrm{Re}(\lambda) < 0$ lze proto volit krok $h > 0$ libovolně velký a podmínka stability (4.18) bude pořád splněna. Metoda, jejíž oblast absolutní stability obsahuje celou zápornou polorovinu $\mathrm{Re}(z) < 0$, se nazývá *A-stabilní metoda*. IE metoda je tedy A-stabilní.

Jak řešit nelineární rovnici? Je to právě mimořádná stabilita, která je onou hledanou předností IE metody ve srovnání s EE metodou. Tento klad je však třeba vykoupit nutností řešit obecně nelineární rovnici (4.20). To neumíme jinak než užitím nějaké iterační metody. Počáteční aproximace se určí snadno, například pomocí EE metody položíme

$$z_0 = y_i + hf(x_i, y_i).$$

Jak ale počítat další aproximace? Zkusme metodu prosté iterace

$$z_{s+1} = \varphi(z_s), \quad \text{kde } \varphi(z) = y_i + hf(x_{i+1}, z).$$

Konvergence $z_s \to y_{i+1}$ nastane, když

$$|\varphi'(z)| = h|f_y(x_{i+1}, z)| \le \alpha < 1, \quad \text{kde } \alpha \text{ je nějaká konstanta}.$$

Tato podmínka představuje omezení délky kroku h. Konkrétně pro testovací rovnici (4.17) dostaneme $|h\lambda| < 1$, což je podmínka ještě přísnější, než jakou vyžaduje podmínka absolutní stability EE metody. Metoda prosté iterace je tedy naprosto nevhodná.

Protože z_0 je dosti dobrá aproximace y_{i+1}, dá se očekávat rychlá konvergence Newtonovy metody. Praktická zkušenost potvrzuje, že tomu tak skutečně je. Počítáme proto

$$z_{s+1} = z_s - \frac{g(z_s)}{g'(z_s)} = z_s - \frac{z_s - y_i - hf(x_{i+1}, z_s)}{1 - h\,f_y(x_{i+1}, z_s)}, \quad s = 0, 1, \ldots, S_i,$$

a nakonec položíme $y_{i+1} = z_{S_i+1}$. Počet iterací S_i se volí, zhruba řešeno, tak, aby odhad chyby $|\widehat{z} - z_{S_i+1}|$ Newtonovy iterace byl menší než $q\varepsilon$, kde ε je požadovaná velikost lokální chyby le_i IE metody a $q < 1$, např. $q = 0,5$. Přibližné splnění podmínky $|\mathrm{le}_i| \le \varepsilon$ zajistíme výběrem vhodné délky h kroku.

Jestliže derivaci f_y nepočítáme v každé iteraci, dostáváme modifikaci Newtonovy metody, kterou budeme nazývat *zjednodušenou Newtonovou metodou*. Úspory, které zjednodušená Newtonova metoda přináší, vyniknou zejména při řešení soustav ODR. Více si o tom dozvíme v odstavci 4.5.2.

4.3. Jednokrokové metody

se vyznačují tím, že přibližné řešení $\mathbf{y}_{i+1}$ v uzlu $x_{i+1} = x_i + h$ se počítá ze vztahu, v němž kromě neznámé $\mathbf{y}_{i+1}$ vystupuje dále už jen předchozí uzel x_i, přibližné řešení $\mathbf{y}_i$ v uzlu x_i, krok h a samozřejmě také pravá strana $\mathbf{f}(x, \mathbf{y})$ diferenciální rovnice. Jednokrokovou metodu lze tedy formálně zapsat ve tvaru

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h\boldsymbol{\Phi}(x_i, \mathbf{y}_i, \mathbf{y}_{i+1}, h; \mathbf{f}), \tag{4.22}$$

kde $\boldsymbol{\Phi}$ je funkce čtyř proměnných x_i, $\mathbf{y}_i$, $\mathbf{y}_{i+1}$ a h, závislá na funkci $\mathbf{f}(x, \mathbf{y})$. Pokud funkce $\boldsymbol{\Phi}$ na $\mathbf{y}_{i+1}$ nezávisí, jde o metodu explicitní, pokud $\boldsymbol{\Phi}$ na $\mathbf{y}_{i+1}$ závisí, jde o metodu implicitní. Pro explicitní Eulerovu metodu (4.9) je $\boldsymbol{\Phi} = \mathbf{f}(x_i, \mathbf{y}_i)$ a pro implicitní Eulerovu metodu (4.19) je $\boldsymbol{\Phi} = \mathbf{f}(x_i + h, \mathbf{y}_{i+1})$.

Lokální diskretizační chyba obecné jednokrokové metody (4.22) je definována rovnicí

$$\mathbf{y}(x_{i+1}) = \mathbf{y}(x_i) + h\boldsymbol{\Phi}(x_i, \mathbf{y}(x_i), \mathbf{y}(x_{i+1}), h; \mathbf{f}) + \mathbf{lte}_i, \tag{4.23}$$

kde $\mathbf{y}(x)$ je řešení úlohy (4.4). Ze srovnání (4.22) a (4.23) plyne, že lokální diskretizační chyba explicitní jednokrokové metody je chyba, které se dopustíme v jednom kroku za předpokladu, že $\mathbf{y}_i = \mathbf{y}(x_i)$ je přesné. Vskutku, označíme-li

$$\tilde{\mathbf{y}}_{i+1} = \mathbf{y}(x_i) + h\boldsymbol{\Phi}(x_i, \mathbf{y}(x_i), \mathbf{y}(x_{i+1}), h; \mathbf{f}), \quad \text{pak} \quad \mathbf{lte}_i = \mathbf{y}(x_{i+1}) - \tilde{\mathbf{y}}_{i+1}.$$

Předpoklad $\mathbf{y}_i = \mathbf{y}(x_i)$ bývá označován jako *lokalizační předpoklad*. Největší celé číslo p, pro něž platí

$$\mathbf{lte}_i = O(h^{p+1}), \tag{4.24}$$

se nazývá řád metody. Rovností (4.24) přitom míníme, že každá složka vektoru $\mathbf{lte}_i$ je řádu $O(h^{p+1})$. Víme již, že EE metoda a IE metoda je řádu 1. Dokázali jsme to sice zatím

jen pro případ jedné rovnice, tvrzení však platí také pro soustavy. Tak například pro IE metodu dostaneme

$$\begin{aligned}\mathbf{lte} :=&\mathbf{y}(x+h)-\mathbf{y}(x)-h\mathbf{f}(x+h,\mathbf{y}(x+h))=\\ &h\mathbf{y}'(x+h)+O(h^2)-h\mathbf{y}'(x+h)=O(h^2)\,.\end{aligned}$$

4.3.1. Metody Taylorova rozvoje

Explicitní Eulerovu metodu jsme odvodili z Taylorova rozvoje

$$\mathbf{y}(x+h)=\mathbf{y}(x)+h\mathbf{y}'(x)+\tfrac{1}{2}h^2\mathbf{y}''(x)+\tfrac{1}{6}h^3\mathbf{y}'''(x)+\cdots+\tfrac{1}{p!}h^p\mathbf{y}^{(p)}(x)+O(h^{p+1}) \tag{4.25}$$

tak, že jsme položili $p=1$ a zanedbali jsme chybu $O(h^2)$. Protože $\mathbf{y}'(x)=\mathbf{f}(x,\mathbf{y}(x))$, dostali jsme EE metodu $\mathbf{y}_{i+1}=\mathbf{y}_i+h\mathbf{f}(x_i,\mathbf{y}_i)$. Podobně můžeme pro $p=2$ dostat metodu řádu 2. Stačí jen vyjádřit druhou derivaci $\mathbf{y}''(x)$. Platí

$$\begin{aligned}\mathbf{y}''(x)=&\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{f}(x,\mathbf{y}(x))=\mathbf{f}_x(x,\mathbf{y}(x))+\mathbf{f}_{\mathbf{y}}(x,\mathbf{y}(x))\mathbf{y}'(x)=\\ &\mathbf{f}_x(x,\mathbf{y}(x))+\mathbf{f}_{\mathbf{y}}(x,\mathbf{y}(x))\mathbf{f}(x,\mathbf{y}(x))\,,\end{aligned}$$

kde $\mathbf{f}_{\mathbf{y}}(x,\mathbf{y})=\{\partial f_j(x,\mathbf{y})/\partial y_\ell\}_{j,\ell=1}^n$ je Jacobiova matice funkce $\mathbf{f}(x,\mathbf{y})$. Metoda

$$\mathbf{y}_{i+1}=\mathbf{y}_i+h\mathbf{f}(x_i,\mathbf{y}_i)+\tfrac{1}{2}h^2[\mathbf{f}_x(x_i,\mathbf{y}_i)+\mathbf{f}_{\mathbf{y}}(x_i,\mathbf{y}_i)\mathbf{f}(x_i,\mathbf{y}_i)] \tag{4.26}$$

je proto řádu 2. Za zvýšenou přesnost však platíme značnou daň: v každém kroku metody (4.26) musíme oproti EE metodě počítat navíc n parciálních derivací $\partial f_j(x_i,\mathbf{y}_i)/\partial x$ a n^2 parciálních derivací $\partial f_j(x,\mathbf{y}_i)/\partial y_\ell$. Metoda (4.26) se nazývá *metoda Taylorova rozvoje* řádu 2. Podobně lze zavést metodu Taylorova rozvoje řádu p předpisem

$$\mathbf{y}_{i+1}=\mathbf{y}_i+h\mathbf{f}(x_i,\mathbf{y}_i)+\tfrac{1}{2}h^2\mathbf{f}^{(1)}(x_i,\mathbf{y}_i)+\cdots+\tfrac{1}{p!}h^p\mathbf{f}^{(p-1)}(x_i,\mathbf{y}_i)\,, \tag{4.27}$$

kde $\mathbf{f}^{(j)}(x_i,\mathbf{y}_i)$ dostaneme tak, že do výrazu $\mathrm{d}^j\mathbf{f}(x,\mathbf{y}(x))/\mathrm{d}x^j$ dosadíme x_i místo x a $\mathbf{y}_i$ místo $\mathbf{y}(x)$. Výrazy pro vyšší derivace funkce $\mathbf{f}(x,\mathbf{y}(x))$ jsou obecně velmi složité. Pro některé zajímavé speciální případy však tyto výrazy mohou být docela jednoduché, takže pak lze metodu Taylorova rozvoje vysokého řádu použít velmi dobře. Jsou-li metody Taylorova rozvoje kvalitně implementovány, mohou být velmi efektivní i pro obecné problémy.

4.3.2. Rungovy-Kuttovy metody

V dalším půjdeme jinou cestou a odvodíme jednokrokové metody vyšších řádů, v nichž vystupují jen hodnoty funkce $\mathbf{f}$. Takové metody se nazývají *Rungovy-Kuttovy metody*. Obecný tvar *s-stupňové Rungovy-Kuttovy metody* je

$$\mathbf{y}_{i+1}=\mathbf{y}_i+h\sum_{j=1}^{s}b_j\mathbf{k}_j\,, \tag{4.28}$$

kde koeficienty $\mathbf{k}_j$ jsou určeny předpisem

$$\mathbf{k}_j=\mathbf{f}(x_i+hc_j,\mathbf{y}_i+h\sum_{\ell=1}^{s}a_{j\ell}\mathbf{k}_\ell)\,,\quad j=1,2,\ldots,s\,. \tag{4.29}$$

Abychom dostali konkrétní metodu, musíme určit stupeň s a dále reálné konstanty b_j, c_j a $a_{j\ell}$ pro $j,\ell=1,2,\ldots,s$. Tak například pro $s=1$, $b_1=1$, $c_1=0$, $a_{11}=0$ dostaneme EE metodu a pro $s=1$, $b_1=1$, $c_1=1$, $a_{11}=1$ dostaneme IE metodu. Označíme-li

$$X_j=x_i+hc_j\,,\quad \mathbf{Y}_j=\mathbf{y}_i+h\sum_{\ell=1}^{s}a_{j\ell}\mathbf{k}_\ell\,,\quad j=1,2,\ldots,s\,, \tag{4.30}$$

vidíme, že $\mathbf{k}_j=\mathbf{f}(X_j,\mathbf{Y}_j)$, takže

$$\mathbf{y}_{i+1}=\mathbf{y}_i+h\sum_{j=1}^{s}b_j\mathbf{f}(X_j,\mathbf{Y}_j) \tag{4.31}$$

určujeme pomocí s hodnot pravé strany $\mathbf{f}(x,\mathbf{y})$ ve vhodně vybraných bodech $[X_j,\mathbf{Y}_j]$. Rungovy-Kuttovy metody včleníme do obecného schématu (4.22) jednokrokových metod tak, že položíme $\Phi=\sum_{j=1}^{s}b_j\mathbf{f}(X_j,\mathbf{Y}_j)$.

Obecná Rungova-Kuttova metoda s koeficienty (4.29) je *implicitní*: abychom určili vektory $\mathbf{k}_j$, musíme řešit soustavu ns obecně nelineárních rovnic, neznámé jsou složky všech vektorů $\mathbf{k}_1,\mathbf{k}_2,\ldots,\mathbf{k}_s$. Dá se ukázat, že pro malé h má soustava (4.29) vždy jediné řešení. Situace se zjednoduší v případě *semi-implicitních* Rungových-Kuttových metod, kdy $a_{j\ell}=0$ pro $\ell>j$, takže

$$\mathbf{k}_j=\mathbf{f}(x_i+hc_j,\mathbf{y}_i+h\sum_{\ell=1}^{j}a_{j\ell}\mathbf{k}_\ell)\,,\quad j=1,2,\ldots,s\,. \tag{4.32}$$

V tomto případě lze nejdříve vyřešit soustavu n obecně nelineárních rovnic pro neznámé složky vektoru $\mathbf{k}_1$, pak další soustavu pro složky vektoru $\mathbf{k}_2$ a tak dále, až nakonec vyřešíme soustavu pro neznámé složky vektoru $\mathbf{k}_s$. Tedy místo abychom řešili jednu soustavu o ns neznámých, řešíme postupně s soustav pro n neznámých. Nejznámější Rungovy-Kuttovy metody jsou však metody *explicitní*, pro které $a_{j\ell}=0$ pro $\ell\geq j$, takže

$$\mathbf{k}_1=\mathbf{f}(x_i,\mathbf{y}_i),\quad \mathbf{k}_j=\mathbf{f}(x_i+hc_j,\mathbf{y}_i+h\sum_{\ell=1}^{j-1}a_{j\ell}\mathbf{k}_\ell),\quad j=2,3,\ldots,s\,. \tag{4.33}$$

Pak se totiž žádné soustavy rovnic řešit nemusí, stačí jen postupně počítat $\mathbf{k}_1,\mathbf{k}_2,\ldots,\mathbf{k}_s$. Poznamenejme, že pod pojmem Rungova-Kuttova metoda se v literatuře běžně rozumí právě metoda explicitní, zatímco skutečnost, že metoda je semi-implicitní nebo plně implicitní se zdůrazňuje označením semi-implicitní Rungova-Kuttova metoda nebo implicitní Rungova-Kuttova metoda.

Koeficienty Rungových-Kuttových metod je zvykem zapisovat do tabulky známé jako *Butcherova tabulka*:

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \dots & a_{1s} \\ c_2 & a_{21} & a_{22} & \dots & a_{2s} \\ \vdots & \vdots & \vdots & & \vdots \\ c_s & a_{s1} & a_{s2} & \dots & a_{ss} \\ \hline & b_1 & b_2 & \dots & b_s \end{array}$$

U explicitních metod se do tabulky zapisují koeficienty $a_{k\ell}$ jen pro $\ell < k$. Protože všechny prakticky používané metody splňují podmínku

$$c_j = \sum_{\ell=1}^{s} a_{j\ell}\,, \quad j = 1, 2, \dots, s\,, \tag{4.34}$$

budeme i my předpokládat, že podmínka (4.34) platí.

4.3.2.1. *Podmínky řádu*

Věnujme se nyní otázce, jak určit koeficienty b_j, c_j a $a_{j\ell}$ tak, aby metoda byla zvoleného řádu. Naše úvahy se zjednoduší, když problém (4.4) přepíšeme do *autonomního tvaru*

$$\begin{pmatrix} \mathbf{y}(t) \\ x(t) \end{pmatrix}' = \begin{pmatrix} \mathbf{f}(x(t), \mathbf{y}(t)) \\ 1 \end{pmatrix}, \quad \begin{pmatrix} \mathbf{y}(0) \\ x(0) \end{pmatrix} = \begin{pmatrix} \boldsymbol{\eta} \\ a \end{pmatrix}. \tag{4.35}$$

Aplikujeme-li na tento systém Rungovu-Kuttovu metodu, dostaneme

$$\begin{pmatrix} \mathbf{y}_{i+1} \\ x_{i+1} \end{pmatrix} = \begin{pmatrix} \mathbf{y}_i \\ x_i \end{pmatrix} + h \sum_{j=1}^{s} b_j \begin{pmatrix} \mathbf{f}(X_j, \mathbf{Y}_j) \\ 1 \end{pmatrix},$$

kde body $[\mathbf{Y}_j, X_j]$ splňují

$$\begin{pmatrix} \mathbf{Y}_j \\ X_j \end{pmatrix} = \begin{pmatrix} \mathbf{y}_i \\ x_i \end{pmatrix} + h \sum_{\ell=1}^{s} a_{j\ell} \begin{pmatrix} \mathbf{f}(X_\ell, \mathbf{Y}_\ell) \\ 1 \end{pmatrix}, \quad j = 1, 2, \dots, s.$$

Vzhledem k podmínce (4.34) je $X_j = x_i + hc_j$, takže body $[X_j, \mathbf{Y}_j]$ splňují (4.30). To ale znamená, že odvození Rungovy-Kuttovy metody pro autonomní systém (4.35) je ekvivalentní odvození Rungovy-Kuttovy metody pro obecný problém (4.4).

Předpokládejme tedy, že pravá strana $\mathbf{f}(x, \mathbf{y})$ je funkcí jen proměnné $\mathbf{y}$, řešíme tedy rovnici $\mathbf{y}' = \mathbf{f}(\mathbf{y})$. Rungova-Kuttova metoda je řádu p, jestliže pro její lokální diskterizační chybu platí

$$\begin{aligned} \mathbf{lte} &:= \mathbf{y}(x+h) - \mathbf{y}(x) - h \sum_{j=1}^{s} b_j \mathbf{k}_j = O(h^{p+1})\,, \\ \text{kde} \quad \mathbf{k}_j &= \mathbf{f}(\mathbf{y}(x) + h \sum_{\ell=1}^{s} a_{j\ell} \mathbf{k}_\ell)\,, \quad j = 1, 2, \dots, s\,. \end{aligned} \tag{4.36}$$

Podmínky nutné pro dosažení řádu p dostaneme tak, že ve výrazu **lte** rozvineme funkci $\mathbf{y}(x+h)$ okolo x, $\mathbf{k}_j$ okolo $\mathbf{y}(x)$ a anulujeme koeficienty u mocnin h^j pro $j \le p$.

Odvození podmínek řádu 3. Začneme tím, že v rozvoji (4.25) funkce $\mathbf{y}(x+h)$ vyjádříme derivace řešení $\mathbf{y}(x)$ pomocí pravé strany $\mathbf{f}(\mathbf{y})$. Derivováním rovnice $\mathbf{y}'(x) = \mathbf{f}(\mathbf{y}(x))$ po složkách dostaneme

$$y_r''(x) = \frac{\mathrm{d}}{\mathrm{d}x} f_r(\mathbf{y}(x)) = \sum_{\alpha=1}^{n} \frac{\partial f_r}{\partial y_\alpha}(\mathbf{y}(x)) y_\alpha'(x) = \sum_{\alpha=1}^{n} \frac{\partial f_r}{\partial y_\alpha}(\mathbf{y}(x)) f_\alpha(\mathbf{y}(x))\,.$$

Abychom zjednodušili zápis, r-tou složku vektoru označíme indexem r vpravo nahoře a parciální derivaci podle složky α označíme čárkou a indexem α vpravo dole. Tedy f^r je r-tá složka vektoru $\mathbf{f}$, y^α je složka α vektoru $\mathbf{y}$, $f^r_{,\alpha}$ je derivace r-té složky vektoru $\mathbf{f}$ podle složky y^α vektoru $\mathbf{y}$. Vyšší parciální derivace zapíšeme podobně, například druhou parciální derivaci r-té složky vektoru $\mathbf{f}$ podle proměnných y^α a y^β označíme $f^r_{,\alpha,\beta}$. Užitím tohoto označení dostaneme

$$(\mathbf{y}'')^r = \sum_{\alpha=1}^{n} f^r_{,\alpha} f^\alpha\,.$$

Elementární diferenciály. V dalším se nám bude hodit, když definujeme vektor $\mathbf{D}_1^1 = \mathbf{f}$ a vektor $\mathbf{D}_1^2$, jehož r-tá složka $(\mathbf{D}_1^2)^r = \sum_{\alpha=1}^{n} f^r_{,\alpha} f^\alpha$. Pak totiž $\mathbf{y}' = \mathbf{D}_1^1$ a $\mathbf{y}'' = \mathbf{D}_1^2$. Další derivace je

$$(\mathbf{y}''')^r = \sum_{\alpha=1}^{n} \left[\sum_{\beta=1}^{n} \frac{\partial f^r_{,\alpha}}{\partial y^\beta} \frac{\mathrm{d}y^\beta}{\mathrm{d}x} f^\alpha + f^r_{,\alpha} \sum_{\beta=1}^{n} \frac{\partial f^\alpha}{\partial y^\beta} \frac{\mathrm{d}y^\beta}{\mathrm{d}x} \right] = \sum_{\alpha,\beta=1}^{n} \left[f^r_{,\alpha\beta} f^\alpha f^\beta + f^r_{,\alpha} f^\alpha_{,\beta} f^\beta \right].$$

Položíme-li $(\mathbf{D}_1^3)^r = \sum_{\alpha,\beta=1}^{n} f^r_{,\alpha\beta} f^\alpha f^\beta$ a $(\mathbf{D}_2^3)^r = \sum_{\alpha,\beta=1}^{n} f^r_{,\alpha} f^\alpha_{,\beta} f^\beta$, pak $\mathbf{y}''' = \mathbf{D}_1^3 + \mathbf{D}_2^3$. Vektory $\mathbf{D}_1^1$, $\mathbf{D}_1^2$, $\mathbf{D}_1^3$, $\mathbf{D}_2^3$ jsou tzv. *elementární diferenciály.*

V tomto postupu můžeme pokračovat a vyjádřit obecně $\mathbf{y}^{(j)}$ pomocí elementárních diferenciálů $\mathbf{D}_\ell^j$, $\ell = 1, 2, \dots, \lambda_j$. S růstem j počet λ_j elementárních diferenciálů $\mathbf{D}_\ell^j$ rychle roste. Abychom výklad příliš nekomplikovali, spokojíme se s odvozením postačujících podmínek pro metody řádu $p \le 3$ a k tomu nám již odvozená vyjádření pro prvou, druhou a třetí derivaci řešení $\mathbf{y}(x)$ stačí.

Pokračujme proto rozvojem složek

$$k_j^r(\mathbf{y}) = f_r(\mathbf{y}(x) + h \sum_{\ell=1}^{s} a_{j\ell} \mathbf{k}_\ell)$$

koeficientu $\mathbf{k}_j$ okolo bodu $\mathbf{y}(x)$. Z Taylorova rozvoje dostaneme

$$k_j^r = f^r + h \sum_{\alpha=1}^{n} f^r_{,\alpha} \sum_{\ell=1}^{s} a_{j\ell} k_\ell^\alpha + \tfrac{1}{2} h^2 \sum_{\beta,\gamma=1}^{n} f^r_{,\beta\gamma} \sum_{\ell=1}^{s} a_{j\ell} k_\ell^\beta \sum_{\ell=1}^{s} a_{j\ell} k_\ell^\gamma + O(h^3)\,. \tag{4.37}$$

V tomto výrazu se opět vyskytují složky koeficientů $\mathbf{k}_\ell$. Za ně znovu dosadíme z Taylorova rozvoje. Protože člen k_ℓ^α je ve výrazu (4.37) násoben h, stačí použít zkrácený rozvoj

$$k_\ell^\alpha = f^\alpha + h \sum_{\delta=1}^{n} f^\alpha_{,\delta} \sum_{m=1}^{s} a_{\ell m} k_m^\delta + O(h^2)\,.$$

Dosadíme-li toto vyjádření do (4.37) zjistíme, že se v něm vyskytují členy k_m^δ, k_ℓ^β a k_ℓ^γ násobené h^2. Proto je nahradíme ještě kratším rozvojem

$$k_\ell^\beta = f^\beta + O(h)\,, \quad k_\ell^\gamma = f^\gamma + O(h), \quad k_m^\delta = f^\delta + O(h)\,.$$

Jestliže provedeme všechna naznačená dosazení, dostaneme z (4.37)

$$k_j^r = f^r + h\sum_{\alpha=1}^{n} f_{,\alpha}^r \sum_{\ell=1}^{s} a_{j\ell}\left\{ f^\alpha + h\sum_{\delta=1}^{n} f_{,\delta}^\alpha \sum_{m=1}^{s} a_{\ell m}[f^\delta + O(h)] + O(h^2)\right\} +$$
$$\tfrac{1}{2}h^2 \sum_{\beta,\gamma=1}^{n} f_{,\beta\gamma}^r \sum_{\ell=1}^{s} a_{j\ell}[f^\beta + O(h)] \sum_{\ell=1}^{s} a_{j\ell}[f^\gamma + O(h)] + O(h^3)\,.$$

Pomocí (4.34) poslední výraz upravíme na tvar

$$k_j^r = f^r + h\sum_{\alpha=1}^{n} f_{,\alpha}^r f^\alpha c_j + h^2 \sum_{\alpha,\delta=1}^{n} f_{,\alpha}^r f_{,\delta}^\alpha f^\delta \sum_{\ell=1}^{s} a_{j\ell}c_\ell + \tfrac{1}{2}h^2 \sum_{\beta,\gamma=1}^{n} f_{,\beta,\gamma}^r f^\beta f^\gamma c_j^2 + O(h^3)$$

a odtud pomocí elementárních diferenciálů

$$\mathbf{k}_j = \mathbf{D}_1^1 + hc_j\mathbf{D}_1^2 + h^2\left[\tfrac{1}{2}c_j^2\mathbf{D}_1^3 + \sum_{\ell=1}^{s} a_{j\ell}c_\ell \mathbf{D}_2^3\right] + O(h^3)\,.$$

Protože

$$\mathbf{y}(x+h) - \mathbf{y}(x) = h\mathbf{D}_1^1 + \tfrac{1}{2}h^2\mathbf{D}_1^2 + \tfrac{1}{6}h^3(\mathbf{D}_1^3 + \mathbf{D}_2^3) + O(h^4),$$

dosazením do (4.36) dostaneme

$$\mathbf{lte} = \mathbf{y}(x+h) - \mathbf{y}(x) - h\sum_{j=1}^{s} b_j\mathbf{k}_j = h\mathbf{D}_1^1\left[1 - \sum_{j=1}^{s} b_j\right] + h^2\mathbf{D}_1^2\left[\tfrac{1}{2} - \sum_{j=1}^{s} b_j c_j\right] + \tag{4.38}$$
$$h^3\mathbf{D}_1^3\left[\tfrac{1}{6} - \tfrac{1}{2}\sum_{j=1}^{s} b_j c_j^2\right] + h^3\mathbf{D}_2^3\left[\tfrac{1}{6} - \sum_{j,\ell=1}^{s} b_j a_{j\ell} c_\ell\right] + O(h^4)\,,$$

nebo stručněji $\mathbf{lte} = hT_1^1\mathbf{D}_1^1 + h^2T_1^2\mathbf{D}_1^2 + h^3[T_1^3\mathbf{D}_1^3 + T_2^3\mathbf{D}_2^3] + O(h^4)$, kde T_ℓ^j jsou tzv. *koeficienty lokální diskretizační chyby.*

Podmínky pro metody řádu 1 až 3. Aby Rungova-Kuttova metoda byla řádu $p \le 3$, musí být tzv. *podmínky řádu* $T_\ell^j = 0$ pro $j = 1, 2, \ldots, p$ a $\ell = 1, 2, \ldots, \lambda_j$, kde

$\lambda_1 = 1$, $\quad T_1^1 = 1 - \sum_{i=1}^{s} b_i$,

$\lambda_2 = 1$, $\quad T_1^2 = \frac{1}{2} - \sum_{i=1}^{s} b_i c_i$,

$\lambda_3 = 2$, $\quad T_1^3 = \frac{1}{2}\left[\frac{1}{3} - \sum_{i=1}^{s} b_i c_i^2\right]$, $\quad T_2^3 = \frac{1}{6} - \sum_{i,j=1}^{s} b_i a_{ij} c_j$.

Obecně, má-li funkce $\mathbf{f}(\mathbf{y})$ spojité derivace všech řádů, můžeme lokální diskretizační chybu Rungovy-Kuttovy metody vyjádřit ve tvaru

$$\mathbf{lte} = \sum_{j=1}^{\infty} h^j \sum_{\ell=1}^{\lambda_j} T_\ell^j \mathbf{D}_\ell^j\,. \tag{4.39}$$

Podmínky pro metodu řádu 4. Podmínky řádu $T_\ell^j = 0$, $\ell = 1, 2, \ldots, \lambda_j$, $j = 1, 2, \ldots, p$, nutné a postačující pro metody řádu p, lze najít ve většině učebnic numerické matematiky nejméně pro $p \le 4$. Uveďme si proto koeficienty T_ℓ^4 lokální diskretizační chyby pro $p = 4$ také my:

$$\lambda_4 = 4, \qquad T_1^4 = \tfrac{1}{6}\left[\tfrac{1}{4} - \textstyle\sum_{i=1}^{s} b_i c_i^3\right], \qquad T_2^4 = \tfrac{1}{8} - \textstyle\sum_{i,j=1}^{s} b_i c_i a_{ij} c_j\,,$$
$$T_3^4 = \tfrac{1}{2}\left[\tfrac{1}{12} - \textstyle\sum_{i,j=1}^{s} b_i a_{ij} c_j^2\right], \quad T_4^4 = \tfrac{1}{24} - \textstyle\sum_{i,j,k=1}^{s} b_i a_{ij} a_{jk} c_k\,.$$

V [13] je popsán algoritmus pro generování koeficientů T_ℓ^j a tyto koeficienty jsou tam explicitně uvedeny pro $p \le 7$.

Čtenáře možná napadlo, proč jsme podmínky řádu odvozovali s poměrně velkým úsilím pro soustavu $\mathbf{y}' = \mathbf{f}(x, \mathbf{y})$, když pro jednu rovnici $y' = f(x, y)$ by to jistě bylo jednodušší. Důvod je ten, že podmínky řádu pro soustavu se od podmínek řádu pro jednu rovnici mohou lišit. Lze ukázat, že pro $p \le 4$ tato situace nenastane, takže všechny Rungovy-Kuttovy metody, které jsou řádu nejvýše 4 pro jednu rovnici, jsou téhož řádu i pro soustavu rovnic. Pro $p \ge 5$ se však podmínky řádu pro soustavu a pro jednu rovnici už liší a existují metody, které jsou řádu p pro jednu rovnici a řádu menšího než p pro soustavu. Když byla tato skutečnost odhalena, vyvolala rozčarování a obavy, neboť v té době bylo známo několik metod vyšších řádů, odvozených pro jednu rovnici, ale používaných i pro systémy. Naštěstí se ukázalo, že řád těchto metod na počtu rovnic nezávisel.

Konvergence. Pro rychlost konvergence Rungovy-Kuttovy metody platí následující

Věta *(o rychlosti konvergence Rungovy-Kuttovy metody). Rungova-Kuttova metoda řádu* $p \ge 1$ *má globální diskretizační chybu* $\mathbf{e}_i = \mathbf{y}(x_i) - \mathbf{y}_i$ *řádu* $O(h^p)$.

Důkaz věty je uveden např. v [38], [17]. Předpokladem platnosti věty je dostatečná hladkost pravé strany $\mathbf{f}$, konkrétně je třeba, aby funkce $\mathbf{f}(x, \mathbf{y})$ měla spojité derivace až do řádu p včetně. Pokud $\mathbf{f}$ má spojité derivace jen do řádu $s \le p$, pak lze pro globální chybu dokázat pouze řád $O(h^s)$.

4.3.2.2. *Explicitní metody*

Označme $p(s)$ maximální dosažitelný řád s-stupňové explicitní Rungovy-Kuttovy metody abychom zdůraznili, že závisí na s. O funkci $p(s)$ je známo, že $p(s) \to \infty$ pro $s \to \infty$. Kromě toho platí

$p(s) = s$ pro $s = 1, 2, 3, 4$, $\qquad p(8) = 6$,
$p(5) = 4$, $\qquad p(9) = 7$
$p(6) = 5$, $\qquad p(s) \le s - 2$ pro $s = 10, 11, \ldots$
$p(7) = 6$,

Explicitní p-stupňové Rungovy-Kuttovy metody řádu p tedy existují jen pro $1 \le p \le 4$, tj. pouze pro tyto řády existují metody, které počítají pravou stranu jen p-krát. Uveďme si několik nejznámějších metod tohoto typu.

Metoda řádu 1. Pro $s = p = 1$ existuje jediná explicitní metoda a tou je nám již známá explicitní Eulerova metoda $\mathbf{y}_{i+1} = \mathbf{y}_i + h\mathbf{f}(x_i, \mathbf{y}_i)$.

Metody řádu 2. Pro $s = p = 2$ má explicitní metoda Butcherovu tabulku

0	
c_2	a_{21}
	b_1 b_2

Podmínky pro metodu řádu 2 stanoví

$$b_1 + b_2 = 1, \quad b_2 c_2 = \tfrac{1}{2},$$

a protože ve shodě s (4.34) předpokládáme $a_{21} = c_2$, dostáváme tabulku

0	
a	a
	$1-b$ b

kde $ab = \frac{1}{2}$. Parametry a, b jsou tedy svázány jednou podmínkou, takže zvolíme-li $a \neq 0$, je $b = 1/(2a)$.

Pro $a = \frac{1}{2}$ je $b = 1$ a dostáváme metodu

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h\mathbf{k}_2, \quad \text{kde} \quad \mathbf{k}_2 = \mathbf{f}(x_i + \tfrac{1}{2}h, \mathbf{y}_i + \tfrac{1}{2}h\mathbf{k}_1)\,, \ \mathbf{k}_1 = \mathbf{f}(x_i, \mathbf{y}_i)\,, \tag{4.40}$$

známou pod názvem *modifikovaná Eulerova metoda*. Budeme ji značit EM1 jako *první modifikace Eulerovy metody*. V anglicky psané literatuře je metoda (4.40) známa jako „midpoint Euler formula".

Pro $a = 1$ je $b = \frac{1}{2}$ a dostáváme metodu

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \tfrac{1}{2}h[\mathbf{k}_1 + \mathbf{k}_2], \quad \text{kde} \quad \mathbf{k}_1 = \mathbf{f}(x_i, \mathbf{y}_i),\ \mathbf{k}_2 = \mathbf{f}(x_i + h, \mathbf{y}_i + h\mathbf{k}_1)\,, \tag{4.41}$$

uváděnou též pod názvem *modifikovaná Eulerova metoda*. Budeme ji značit EM2 jako *druhou modifikaci Eulerovy metody*. Metoda (4.41) se také často uvádí pod názvem *Heunova metoda*.

Dosazením do (4.38) dostaneme pro metodu řádu 2 lokální diskretizační chybu

$$\mathbf{lte} = h^3[T_1^3\mathbf{D}_1^3 + T_2^3\mathbf{D}_2^3] + O(h^4) = h^3\left[\left(\tfrac{1}{6} - \tfrac{1}{2}ba^2\right)\mathbf{D}_1^3 + \tfrac{1}{6}\mathbf{D}_2^3\right] + O(h^4)\,.$$

Odtud předešlím vidíme, že metodu řádu 3 nemůžeme dostat pro žádnou volbu a, b, neboť člen $\frac{1}{6}\mathbf{D}_2^3$ není čím anulovat. Pomocí vhodné volby koeficientu a můžeme ovlivnit jen koeficient $T_1^3 = \frac{1}{6} - \frac{1}{2}ba^2 = \frac{1}{6} - \frac{1}{4}a$. Pro EM1 metodu je $T_1^3 = \frac{1}{24}$ a pro EM2 metodu $T_1^3 = -\frac{1}{12}$. Pokud bychom tedy mohli vliv druhého sčítance $\frac{1}{6}\mathbf{D}_2^3$ zanedbat, mohli bychom tvrdit, že EM1 metoda je dvakrát přesnější než EM2 metoda. Pro $a = \frac{2}{3}$ lze dokonce docílit anulování koeficientu T_1^3. Pak $b = 1/(2a) = \frac{3}{4}$, takže dostáváme metodu

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \tfrac{1}{4}h[\mathbf{k}_1 + 3\mathbf{k}_2]\,, \quad \text{kde}\ \mathbf{k}_1 = \mathbf{f}(x_i, \mathbf{y}_i),\ \mathbf{k}_2 = \mathbf{f}(x_i + \tfrac{2}{3}h, \mathbf{y}_i + \tfrac{2}{3}h\mathbf{k}_1)\,, \tag{4.42}$$

která je optimální v tom smyslu, že pro ni jsou absolutní hodnoty koeficientů T_1^3 a T_2^3 nejmenší možné. Taková metoda se nazývá *metoda s vyladěnou chybou*. I když metoda (4.42) příliš rozšířená není, nejlepší používané metody vyšších řádů mívají chybu vyladěnu, tj. jde-li o metodu řádu p, pak jsou absolutní hodnoty $|T_j^{p+1}|$ koeficientů T_j^{p+1} hlavního

členu $\sum_{j=1}^{\lambda_{p+1}} T_j^{p+1}\mathbf{D}_j^{p+1}$ lokální diskretizační chyby **lte** malé. Je však třeba uvést, že kvalitu metody ovlivňuje ještě řada dalších faktorů, které jsme doposud nezkoumali a o nichž se zmíníme později.

Metody řádu 3. Pro $s = p = 3$ dostáváme Butcherovu tabulku

0			
c_2	c_2		
c_3	$c_3 - a_{32}$	a_{32}	
	b_1	b_2	b_3

a 4 podmínky pro metodu řádu 3:

$$b_1 + b_2 + b_3 = 1\,, \quad b_2c_2 + b_3c_3 = \tfrac{1}{2}\,,$$
$$b_2c_2^2 + b_3c_3^2 = \tfrac{1}{3}\,, \quad b_3a_{32}c_2 = \tfrac{1}{6}\,.$$

Ukazuje se, že když zvolíme dva parametry $0 < c_2 < c_3$, jsou tím všechny koeficienty metody jednoznačně určeny. Volba $c_2 = \frac{1}{2}$, $c_3 = \frac{3}{4}$ vede na populární *Ralstonovu metodu*:

0			
$\frac{1}{2}$	$\frac{1}{2}$		
$\frac{3}{4}$	0	$\frac{3}{4}$	
	$\frac{2}{9}$	$\frac{1}{3}$	$\frac{4}{9}$

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \tfrac{1}{9}h\,[2\mathbf{k}_1 + 3\mathbf{k}_2 + 4\mathbf{k}_3]\,,$$
$$\mathbf{k}_1 = \mathbf{f}(x_i, \mathbf{y}_i)\,, \quad \mathbf{k}_2 = \mathbf{f}(x_i + \tfrac{1}{2}h, \mathbf{y}_i + \tfrac{1}{2}h\mathbf{k}_1)\,,$$
$$\mathbf{k}_3 = \mathbf{f}(x_i + \tfrac{3}{4}h, \mathbf{y}_i + \tfrac{3}{4}h\mathbf{k}_2)\,.$$

Ralstonova metoda je řádu 3 a má vyladěnou chybu: dva koeficienty T_2^4 a T_3^4 hlavního členu lokální diskretizační chyby jsou nulové, koeficient $T_4^4 = \frac{1}{24}$ pro jakoukoliv volbu parametrů c_1, c_2 a koeficient $T_1^4 = \frac{1}{288}$ je malý.

Metody řádu 4. Pro $s = p = 4$ je nejznámější *klasická Rungova-Kuttova metoda*

0				
$\frac{1}{2}$	$\frac{1}{2}$			
$\frac{1}{2}$	0	$\frac{1}{2}$		
1	0	0	1	
	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \tfrac{1}{6}h\,[\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4]\,,$$
$$\mathbf{k}_1 = \mathbf{f}(x_i, \mathbf{y}_i)\,, \qquad \mathbf{k}_2 = \mathbf{f}(x_i + \tfrac{1}{2}h, \mathbf{y}_i + \tfrac{1}{2}h\mathbf{k}_1)\,,$$
$$\mathbf{k}_3 = \mathbf{f}(x_i + \tfrac{1}{2}h, \mathbf{y}_i + \tfrac{1}{2}h\mathbf{k}_2)\,, \quad \mathbf{k}_4 = \mathbf{f}(x_i + h, \mathbf{y}_i + h\mathbf{k}_3)\,.$$

Klasická Rungova-Kuttova metoda byla velmi populární v době, kdy se ještě nepoužívaly samočinné počítače a kdy proto velmi významným kritériem byla jednoduchost metody. Toto hledisko však v současné době ztratilo na významu a proto se používají jiné metody. Jednu velice kvalitní dvojici metod řádu 4 a 5 uvedeme v odstavci 4.3.2.4.

4.3.2.3. *Řízení délky kroku*

Doposud jsme předpokládali, že délka h kroku metody je konstantní. Avšak ani z odvození metod ani z jejich tvaru nevyplývá nutnost takové volby. V každém kroku můžeme délku h měnit, otázkou zůstává jak. Ideální přání uživatele je donutit program, aby mu spočítal řešení s požadovanou přesností, tj. aby pro zadanou toleranci $\varepsilon > 0$ platilo $\|\mathbf{e}_i\| = \|\mathbf{y}_i - \mathbf{y}(x_i)\| \le \varepsilon$. To se však neumí, současné programy pro řešení ODR dovedou zajistit jen to, aby pro dostatečně malé ε byla globální chyba $\|\mathbf{e}_i\|$ dosti malá. Docilují toho tím, že délku kroku h vybírají tak, aby velikost $\|\mathbf{le}_i\|$ lokální chyby nabývala pořád přibližně stejné hodnoty ε popřípadě $h\varepsilon$. Velikost $\|\mathbf{e}_i\|$ globální chyby lze jen odhadnout (řídit ji zatím neumíme), v odstavci 4.3.2.6 si ukážeme, jak se to dělá. Předesíláme však, že monitorování velikosti globální chyby je nákladné a že většina programů tuto službu

nenabízí.

Strategie EPS a EPUS. Připomeňme si, že lokální chybou $\mathbf{le}_i$ rozumíme chybu, které se dopustíme v jednom kroku metody $\mathbf{y}_{i+1} = \mathbf{y}_i + h_i\mathbf{\Phi}(x_i, \mathbf{y}_i, \mathbf{y}_{i+1}, h_i; \mathbf{f})$, tj.

$$\mathbf{le}_i = \mathbf{u}(x_i+h_i) - \mathbf{y}_{i+1}\,, \quad \text{kde} \quad \mathbf{u}' = \mathbf{f}(x, \mathbf{u}),\ \mathbf{u}(x_i) = \mathbf{y}_i$$

je lokální řešení. Velikost lokální chyby se obvykle ovládá jedním ze dvou následujících způsobů. Podle kritéria *chyby v kroku*, zkráceně EPS podle anglického *error per step*, se krok z x_i do $x_{i+1} = x_i + h_i$ akceptuje, jen když

$$\|\mathbf{le}_i\| \le \varepsilon\,. \tag{4.43}$$

Podle kritéria *chyby v jednotkovém kroku*, zkráceně EPUS podle anglického *error per unit step*, se krok akceptuje, jen když

$$\|\mathbf{le}_i\| \le h_i\varepsilon\,. \tag{4.44}$$

Není-li podmínka (4.43) resp. (4.44) splněna, krok je odmítnut a je třeba ho opakovat pro menší délku h_i^* kroku takovou, aby tato podmínka chyby splněna byla. Délku h_i^* kroku však nevolíme ani zbytečně malou, naopak, vybereme ji co možná největší.

Když program provede krok z x_i do x_{i+1}, spočte odhad $\mathbf{est}_i$ vzniklé lokální chyby $\mathbf{le}_i$. Jak se to provede, to si ukážeme v odstavci 4.3.2.4. Nyní je pro nás podstatné, že lokální chyba Rungovy-Kuttovy metody řádu p je

$$\mathbf{le}_i = \boldsymbol{\psi}(x_i, \mathbf{y}_i)h_i^{p+1} + O(h_i^{p+2})\,, \tag{4.45}$$

kde funkce $\boldsymbol{\psi}(x, \mathbf{y})$ je tzv. *hlavní funkce chyby* (z (4.39) plyne $\boldsymbol{\psi} = \sum_{\ell=1}^{\lambda_{p+1}} T_\ell^{p+1}\mathbf{D}_\ell^{p+1}$), takže

$$\mathbf{est}_i \approx \mathbf{le}_i = h_i^{p+1}\boldsymbol{\psi}(x_i, \mathbf{y}_i) + O(h_i^{p+2})\,.$$

Pokud bychom z bodu x_i postupovali krokem délky h_i^*, dostali bychom lokální chybu

$$\mathbf{le}_i^* = \mathbf{u}(x_i+h_i^*) - \mathbf{y}_{i+1}^* = (h_i^*)^{p+1}\boldsymbol{\psi}(x_i, \mathbf{y}_i) + O((h_i^*)^{p+2})\,.$$

Srovnáním obou výrazů dospíváme k závěru, že lokální chyba

$$\mathbf{le}_i^* \approx (h_i^*/h_i)^{p+1}\mathbf{est}_i \tag{4.46}$$

je aproximována výrazem, který umíme spočítat. Jestliže je norma odhadu $\mathbf{est}_i$ lokální chyby $\mathbf{le}_i$ větší než tolerance ε, tedy platí-li $\|\mathbf{est}_i\| > \varepsilon$, použijeme přibližnou rovnost (4.46) pro určení *optimální délky kroku*, tj. maximální délky, pro kterou je zvolené kritérium chyby ještě splněno. Jestliže k řízení délky kroku použijeme kritérium EPS, vybereme h_i^* tak, aby

$$\varepsilon = \|\mathbf{le}_i^*\| \approx (h_i^*/h_i)^{p+1}\|\mathbf{est}_i\|\,. \tag{4.47}$$

Zvolíme tedy

$$h_i^* = h_i(\varepsilon/\|\mathbf{est}_i\|)^{1/(p+1)}\,,$$

a výpočet $\mathbf{y}_{i+1}$ opakujeme s krokem délky h_i^*. Nová délka h_i^* kroku je zřejmě menší než původní délka h_i, jde o to, jestli to stačí: při odvozování h_i^* jsme se totiž dopustili řady nepřesností, sice malých, řádu $O(h_i^{p+2})$, ale dopustili. Po výpočtu $\mathbf{y}_{i+1}$ s krokem délky h_i^* se proto může stát, že test chyby znovu neprojde, i když třeba jen těsně. Opakované krácení kroku je „drahé" a proto je na místě být opatrný a za h_i^* zvolit jen jistou část Θ optimální délky. V [34] se uvádí, že reprezentativní hodnota používaná v řadě programů je $\Theta = 0,9$. Proto zvolíme

$$h_i^* = \Theta h_i(\varepsilon/\|\mathbf{est}_i\|)^{1/(p+1)}\,.$$

Zatím jsme se zabývali jen tím, co máme dělat, když je krok zamítnut. Jestliže však je krok z x_i do x_{i+1} úspěšný, je třeba rozhodnout, jak vybrat délku h_{i+1} dalšího kroku. Pro lokální chybu platí

$$\mathbf{le}_{i+1} = \mathbf{u}(x_{i+1}+h_{i+1}) - \mathbf{y}_{i+2} = h_{i+1}^{p+1}\boldsymbol{\psi}(x_{i+1}, \mathbf{y}_{i+1}) + O(h_{i+1}^{p+2})\,.$$

Tuto chybu můžeme dobře aproximovat pomocí odhadu $\mathbf{est}_i$ lokální chyby ve stávajícím kroku. Vskutku, protože $x_{i+1} = x_i + h_i$ a $\mathbf{y}_{i+1} = \mathbf{y}_i + O(h_i)$, užitím Taylorova rozvoje dostaneme $\boldsymbol{\psi}(x_{i+1}, \mathbf{y}_{i+1}) = \boldsymbol{\psi}(x_i, \mathbf{y}_i) + O(h_i)$, takže výraz pro lokální chybu

$$\mathbf{le}_{i+1} = h_{i+1}^{p+1}\boldsymbol{\psi}(x_i, \mathbf{y}_i) + O(h_{i+1}^{p+2})$$

je formálně stejný jako výraz pro lokální chybu $\mathbf{le}_i$. Proto volíme délku

$$h_{i+1} = \Theta h_i(\varepsilon/\|\mathbf{est}_i\|)^{1/(p+1)}$$

kroku z x_{i+1} do x_{i+2} stejně jako délku h_i^* zamítnutého kroku.

Jestliže k řízení délky kroku použijeme kritérium EPUS, nahradíme (4.47) pomocí

$$h_i^*\varepsilon = \|\mathbf{le}_i^*\| \approx (h_i^*/h_i)^{p+1}\|\mathbf{est}_i\|\,.$$

Odtud vyjádříme h_i^* a dostaneme předpis

$$h_i^* = \Theta h_i(\varepsilon h_i/\|\mathbf{est}_i\|)^{1/p}\,.$$

Tentýž předpis použijeme také pro určení délky kroku h_{i+1}. Následuje hrubý

Algoritmus řízení délky kroku

Krok 1. Spočteme $\mathbf{y}_{i+1} = \mathbf{y}_i + h\mathbf{\Phi}(x_i, \mathbf{y}_i, \mathbf{y}_{i+1}, h; \mathbf{f})$.

Krok 2. Určíme $\mathbf{est}_i$ a položíme $h^* = \begin{cases} \Theta h(\varepsilon/\|\mathbf{est}_i\|)^{1/(p+1)} & \text{pro EPS, XEPS,} \\ \Theta h(\varepsilon h/\|\mathbf{est}_i\|)^{1/p} & \text{pro EPUS, XEPUS.} \end{cases}$

Je-li $\|\mathbf{est}_i\| > \varepsilon$, následuje krok 3, v opačném případě pokračujeme krokem 4.

Krok 3. Položíme $h := \mathcal{M}_3(h, h^*)$ a pokračujeme krokem 1.

Krok 4. Položíme $h := \mathcal{M}_4(h, h^*)$, zvýšíme $i := i + 1$ a pokračujeme krokem 1.

XEPS je modifikace kritéria EPS a podobně XEPUS je modifikace kritéria EPUS, obě modifikace jsou vysvětleny v následujícím odstavci 4.3.2.4. Zbývá vysvětlit význam příkazů $h := \mathcal{M}_3(h,h^*)$ v kroku 3 a $h := \mathcal{M}_4(h,h^*)$ v kroku 4. $\mathcal{M}_3(h,h^*)$ a $\mathcal{M}_4(h,h^*)$ jsou modifikační funkce, pro $\mathcal{M}_3(h,h^*) = \mathcal{M}_4(h,h^*) = h^*$ dostáváme algoritmus, který plně odpovídá předchozímu výkladu. V programech pro řešení ODR však bývá navržená délka h^* kroku ještě modifikována postupy, které jsme si formálně označili pomocí funkcí $\mathcal{M}_3$ a $\mathcal{M}_4$. Abychom byli konkrétní uveďme si, o jaké modifikace jde v implementacích metod BS32 resp. DP54 v programech ODE23 resp. ODE45 v MATLABU, viz [19]. Popis metod BS32 a DP54 uvedeme v odstavci 4.3.2.4. Následují poznámky specifikující zmiňované modifikace $\mathcal{M}_3$ a $\mathcal{M}_4$.

1. Označme h_{min} resp. h_{max} minimální resp. maximální povolenou délku kroku. Pak je třeba algoritmus modifikovat tak, aby bylo zajištěno splnění podmínky

   $$h_{min} \le h \le h_{max}\,.$$

   Pokud tedy v kroku 3 algoritmu nastane $h^* < h_{min}$, je třeba výpočet ukončit konstatováním, že danou diferenciální rovnici program neumí s požadovanou přesností vyřešit. Přitom $h_{min} = 16\varepsilon_m$, kde ε_m je tzv. strojová přesnost. Jestliže v kroku 4 algoritmu nastane $h^* > h_{max}$, položí se $h = h_{max}$. Použito je $h_{max} = 0,1(b-a)$.

2. Požadujeme, aby se nová délka $\mathcal{M}_3(h,h^*)$ v kroku 3 resp. $\mathcal{M}_4(h,h^*)$ v kroku 4 příliš nelišila od stávající délky h. Konkrétně, při prvém zkracování v kroku 3

   $$\mathcal{M}_3(h,h^*) = \max(h^*, q_{min}\,h)\,,$$

   kde

   $$q_{min} = \begin{cases} 0,5 & \text{v ODE23 (metoda řádu 3)}\,,\\ 0,1 & \text{v ODE45 (metoda řádu 5)}\,,\end{cases}$$

   při opakovaném zkracování téhož kroku pak $\mathcal{M}_3(h,h^*) = 0,5\,h$. Při prodlužování v kroku 4

   $$\mathcal{M}_4(h,h^*) = \min(h^*, q_{max}\,h)\,,$$

   přičemž $q_{max} = 5$.

3. Bezprostředně po zkrácení kroku nesmí následovat jeho prodloužení. □

Stranou našeho zájmu zatím zůstala otázka, jak smysluplně volit toleranci ε. V [34] se doporučuje zadávat ji ve tvaru $\varepsilon = \max\{\varepsilon_R N_i(\mathbf{Y}), \varepsilon_A\}$, kde ε_R je relativní tolerance, $N_i(\mathbf{y})$ reprezentuje velikost řešení, například $N_i(\mathbf{y}) = \max(\|\mathbf{y}_i\|, \|\mathbf{y}_{i+1}\|)$, ε_A je absolutní tolerance, jako standardní hodnota se doporučuje $\varepsilon_R = 10^{-3}$ a $\varepsilon_A = 10^{-6}$.

Na začátku výpočtu je třeba vybrat délku počátečního kroku. Spokojíme se s hrubým odhadem založeným na následující myšlence: metoda $\widehat{\mathbf{y}}_1 = \boldsymbol{\eta} \approx \mathbf{y}(a+h)$ dává lokální chybu $\widehat{\mathbf{le}} = \mathbf{y}(a+h) - \boldsymbol{\eta} \approx h\mathbf{f}(a,\boldsymbol{\eta})$ řádu $O(h)$ a my předpokládáme, že při výpočtu $\mathbf{y}_1$

metodou řádu p dostaneme lokální chybu $\mathbf{le} \approx (\widehat{\mathbf{le}})^{p+1}$. Odtud, například podle kritéria EPS, obdržíme $\|\mathbf{le}\| \approx (h\|\mathbf{f}(a,\boldsymbol{\eta})\|)^{p+1} \approx \varepsilon$ a proto klademe

$$h = \Theta\left[\max(\varepsilon_R\|\boldsymbol{\eta}\|, \varepsilon_A)\right]^{1/(p+1)} / \|\mathbf{f}(a,\boldsymbol{\eta})\|\,,$$

přičemž pro $h < h_{min}$ změníme h na h_{min} a pro $h > h_{max}$ změníme h na h_{max}.

Při programovaní algoritmu pro řízení délky kroku je třeba být opatrný. Příkazy programu je nutné uspořádat tak, aby nemohlo dojít k dělení nulou (v kroku 2 algoritmu a také při určování délky počátečního kroku).Podrobnější informace týkající se řízení délky kroku lze najít například v [34], [19], [36].

4.3.2.4. *Odhad lokální chyby*

je založen na použití dvou vhodně zvolených metod, z nichž jedna je řádu p a druhá řádu $p+1$. V dalším uvedeme dva nejčastěji používané postupy odhadu lokální chyby a sice metodu poloviční kroku a metodu vnořených odhadů chyby.

Metoda poloviční kroku. Vysvětleme si nejdříve hlavní myšlenku. Z výchozí hodnoty $\mathbf{y}_i$ spočteme v jednom kroku délky h přibližné řešení $\overline{\mathbf{y}}_{i+1}$ a kromě toho spočteme ze stejné výchozí hodnoty $\mathbf{y}_i$ pomocí dvou kroků poloviční délky $h/2$ přibližné řešení $\widetilde{\mathbf{y}}_{i+1}$. Odhad lokální chyby pak dostaneme vhodnou kombinací hodnot $\overline{\mathbf{y}}_{i+1}$ a $\widetilde{\mathbf{y}}_{i+1}$. Následuje podrobnější výklad.

Odvození. V kroku od x_i do $x_{i+1} = x_i + h$ spočteme $\overline{\mathbf{y}}_{i+1}$ metodou řádu p, takže pro lokální chybu platí

$$\mathbf{u}(x_{i+1}) - \overline{\mathbf{y}}_{i+1} = h^{p+1}\boldsymbol{\psi}(x_i,\mathbf{y}_i) + O(h^{p+2}) = \overline{\mathbf{le}}_i\,.$$

Postupujeme-li z x_i do $x_{i+1/2} := x_i + h/2$ krokem poloviční délky, dostaneme přibližné řešení, které označíme $\widetilde{\mathbf{y}}_{i+1/2}$. Při tom vznikne chyba

$$\mathbf{u}(x_{i+1/2}) - \widetilde{\mathbf{y}}_{i+1/2} = \left(\frac{h}{2}\right)^{p+1}\boldsymbol{\psi}(x_i,\mathbf{y}_i) + O(h^{p+2}) = \frac{1}{2^{p+1}}\,\overline{\mathbf{le}}_i + O(h^{p+2})\,.$$

V dalším půlkroku získáme z výchozí hodnoty $\widetilde{\mathbf{y}}_{i+1/2}$ přibližné řešení $\widetilde{\mathbf{y}}_{i+1}$ s chybou

$$\mathbf{v}(x_{i+1}) - \widetilde{\mathbf{y}}_{i+1} = \left(\frac{h}{2}\right)^{p+1}\boldsymbol{\psi}(x_{i+1/2},\widetilde{\mathbf{y}}_{i+1/2}) + O(h^{p+2})\,,$$

kde $\mathbf{v}(x)$ je lokální řešení procházející bodem $[x_{i+1/2}, \widetilde{\mathbf{y}}_{i+1/2}]$. Protože $\widetilde{\mathbf{y}}_{i+1/2} = \mathbf{y}_i + O(h)$, z Taylorova rozvoje dostaneme $\boldsymbol{\psi}(x_{i+1/2},\widetilde{\mathbf{y}}_{i+1/2}) = \boldsymbol{\psi}(x_i,\mathbf{y}_i) + O(h)$, takže

$$\mathbf{v}(x_{i+1}) - \widetilde{\mathbf{y}}_{i+1} = \left(\frac{h}{2}\right)^{p+1}\boldsymbol{\psi}(x_i,\mathbf{y}_i) + O(h^{p+2}) = \frac{1}{2^{p+1}}\,\overline{\mathbf{le}}_i + O(h^{p+2})\,.$$

Odečteme-li od rovnice $\mathbf{u}(x_{i+1}) - \overline{\mathbf{y}}_{i+1} = \overline{\mathbf{le}}_i$ předchozí rovnici, po malé úpravě obdržíme

$$\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1} = \left(1 - \frac{1}{2^{p+1}}\right)\overline{\mathbf{le}}_i - [\mathbf{u}(x_{i+1}) - \mathbf{v}(x_{i+1})] + O(h^{p+2})\,.$$

Dále upravujme rozdíl $\mathbf{u}(x_{i+1}) - \mathbf{v}(x_{i+1})$. Označíme-li

$$\boldsymbol{\omega} = \mathbf{u}(x_{i+1}) - \mathbf{v}(x_{i+1}) - [\mathbf{u}(x_{i+1/2}) - \mathbf{v}(x_{i+1/2})]\,,$$

pak

$$\mathbf{u}(x_{i+1}) - \mathbf{v}(x_{i+1}) = \mathbf{u}(x_{i+1/2}) - \mathbf{v}(x_{i+1/2}) + \boldsymbol{\omega}\,.$$

Protože $\mathbf{v}(x_{i+1/2}) = \widetilde{\mathbf{y}}_{i+1/2}$, je

$$\mathbf{u}(x_{i+1}) - \mathbf{v}(x_{i+1}) = \mathbf{u}(x_{i+1/2}) - \widetilde{\mathbf{y}}_{i+1/2} + \boldsymbol{\omega} = \frac{1}{2^{p+1}}\,\overline{\mathbf{le}}_i + \boldsymbol{\omega} + O(h^{p+2})\,.$$

Naším dalším cílem je dokázat, že $\boldsymbol{\omega} = O(h^{p+2})$. Protože

$$\boldsymbol{\omega} = \int_{x_{i+1/2}}^{x_{i+1}} [\,\mathbf{u}'(t) - \mathbf{v}'(t)\,]\,\mathrm{d}t = \int_{x_{i+1/2}}^{x_{i+1}} [\,\mathbf{f}(t, \mathbf{u}(t)) - \mathbf{f}(t, \mathbf{v}(t))\,]\,\mathrm{d}t\,,$$

užitím věty o střední hodnotě a Lipschitzovy podmínky dostaneme

$$\|\boldsymbol{\omega}\| \le \tfrac{1}{2} L h \max_{t \in I} \|\mathbf{u}(t) - \mathbf{v}(t)\|\,, \quad \text{kde} \quad I = \langle x_{i+1/2}, x_{i+1} \rangle\,.$$

Abychom postoupili dále, použijeme známý výsledek, který tvrdí, že pro řešení $\mathbf{u}(x)$ a $\mathbf{v}(x)$ diferenciální rovnice $\mathbf{y}' = \mathbf{f}(x, \mathbf{y})$ platí

$$\|\mathbf{u}(x) - \mathbf{v}(x)\| \le \|\mathbf{u}(a) - \mathbf{v}(a)\| e^{L(x-a)} \quad \text{pro } x \ge a\,,$$

důkaz viz např. [34]. Zvolíme-li v poslední nerovnosti $a := x_{i+1/2}$, $x := t \in I$ a dosadíme do předposlední nerovnosti, dostaneme

$$\|\boldsymbol{\omega}\| \le \tfrac{1}{2} L h e^{Lh/2} \|\mathbf{u}(x_{i+1/2}) - \mathbf{v}(x_{i+1/2})\| = \tfrac{1}{2} L h e^{Lh/2} \|\mathbf{u}(x_{i+1/2}) - \widetilde{\mathbf{y}}(x_{i+1/2})\|\,,$$

takže $\boldsymbol{\omega} = O(h^{p+2})$ a

$$\mathbf{u}(x_{i+1}) - \mathbf{v}(x_{i+1}) = \frac{1}{2^{p+1}}\,\overline{\mathbf{le}}_i + O(h^{p+2})\,.$$

Vrátíme-li se zpět a dosadíme do výrazu pro $\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1}$, obdržíme

$$\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1} = \left(1 - \frac{1}{2^{p+1}}\right) \overline{\mathbf{le}}_i - \frac{1}{2^{p+1}}\,\overline{\mathbf{le}}_i + O(h^{p+2}) = \frac{2^p - 1}{2^p}\,\overline{\mathbf{le}}_i + O(h^{p+2})\,.$$

Odtud dostáváme odhad chyby, který umíme spočítat, neboť

$$\mathbf{u}(x_{i+1}) - \overline{\mathbf{y}}_{i+1} = \overline{\mathbf{le}}_i = \frac{2^p}{2^p - 1}(\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1}) + O(h^{p+2})\,.$$

Je přirozené, abychom pro další výpočet použili přesnější řešení $\widetilde{\mathbf{y}}_{i+1}$. V tom případě nás bude zajímat odhad chyby $\widetilde{\mathbf{le}}_i := \mathbf{u}(x_{i+1}) - \widetilde{\mathbf{y}}_{i+1}$. Ten dostaneme snadno, neboť

$$\mathbf{u}(x_{i+1}) - \widetilde{\mathbf{y}}_{i+1} = [\mathbf{u}(x_{i+1}) - \mathbf{v}(x_{i+1})] + [\mathbf{v}(x_{i+1}) - \widetilde{\mathbf{y}}_{i+1}]\,,$$

a protože výraz v každé ze složených závorek na pravé straně je roven $\overline{\mathbf{le}}_i/2^{p+1} + O(h^{p+2})$,

$$\mathbf{u}(x_{i+1}) - \widetilde{\mathbf{y}}_{i+1} = \widetilde{\mathbf{le}}_i = \frac{1}{2^p}\,\overline{\mathbf{le}}_i + O(h^{p+2}) = \frac{1}{2^p - 1}(\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1}) + O(h^{p+2})\,.$$

Pro chybu metody s krokem $h/2$ tedy použijeme odhad

$$\mathbf{est}_i = \frac{1}{2^p - 1}(\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1})\,.$$

Když definujeme

$$\widehat{\mathbf{y}}_{i+1} = \widetilde{\mathbf{y}}_{i+1} + \mathbf{est}_i = \widetilde{\mathbf{y}}_{i+1} + \frac{1}{2^p - 1}(\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1})$$

vidíme, že $\mathbf{u}(x_{i+1}) - \widehat{\mathbf{y}}_{i+1} = O(h^{p+2})$, tj. metoda výpočtu $\widehat{\mathbf{y}}_{i+1}$ je vyššího řádu $p+1$. Způsob, jakým jsme z přibližných řešení $\overline{\mathbf{y}}_{i+1}$ a $\widetilde{\mathbf{y}}_{i+1}$ řádu p zkonstruovali přibližné řešení $\widehat{\mathbf{y}}_{i+1}$ řádu $p+1$ není nic jiného než Richardsonova extrapolace. Vyšší přesnosti lze dosáhnout také v $x_{i+1/2}$. Stačí si uvědomit, že

$$\mathbf{u}(x_{i+1/2}) - \widetilde{\mathbf{y}}_{i+1/2} = \frac{1}{2^{p+1}}\,\overline{\mathbf{le}}_i + O(h^{p+2}) = \frac{1}{2}\,\frac{1}{2^p - 1}(\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1}) + O(h^{p+2})\,,$$

a položit

$$\widehat{\mathbf{y}}_{i+1/2} = \widetilde{\mathbf{y}}_{i+1/2} + \frac{1}{2}\,\mathbf{est}_i\,.$$

Přesnější řešení $\widehat{\mathbf{y}}_{i+1}$ použijeme jako výchozí v dalším kroku, tj. položíme $\mathbf{y}_{i+1} = \widehat{\mathbf{y}}_{i+1}$. Definujeme-li ještě $\mathbf{y}_{i+1/2} = \widehat{\mathbf{y}}_{i+1/2}$, dostaneme výsledný

Algoritmus metody poloviční kroku

$$\begin{aligned}
&\widetilde{\mathbf{y}}_{i+1/2} = \mathbf{y}_i + \tfrac{1}{2} h \boldsymbol{\Phi}(x_i, \mathbf{y}_i, \mathbf{y}_{i+1/2}, \tfrac{1}{2}h; \mathbf{f})\,,\\
&\widetilde{\mathbf{y}}_{i+1} = \widetilde{\mathbf{y}}_{i+1/2} + \tfrac{1}{2} h \boldsymbol{\Phi}(x_{i+1/2}, \widetilde{\mathbf{y}}_{i+1/2}, \widetilde{\mathbf{y}}_{i+1}, \tfrac{1}{2}h; \mathbf{f})\,,\\
&\overline{\mathbf{y}}_{i+1} = \mathbf{y}_i + h \boldsymbol{\Phi}(x_i, \mathbf{y}_i, \overline{\mathbf{y}}_{i+1}, h; \mathbf{f})\,,\\
&\mathbf{est}_i = \frac{1}{2^p - 1}(\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1})\,,\\
&\mathbf{y}_{i+1/2} = \widetilde{\mathbf{y}}_{i+1/2} + \tfrac{1}{2}\,\mathbf{est}_i\,, \quad \mathbf{y}_{i+1} = \widetilde{\mathbf{y}}_{i+1} + \mathbf{est}_i\,.
\end{aligned} \tag{4.48}$$

Lokální extrapolace. Metoda (4.48) je ukázkou univerzálního postupu, který bývá označován jako *lokální extrapolace*: k základní metodě řádu p, v našem případě jde o vzorec pro výpočet $\widetilde{\mathbf{y}}_{i+1}$, se sestaví předpis pro odhad její lokální chyby

$$\mathbf{est}_i = \widetilde{\mathbf{le}}_i + O(h^{p+2}) = \mathbf{u}(x_i + h) - \widetilde{\mathbf{y}}_{i+1} + O(h^{p+2})$$

a definuje se nové řešení $\mathbf{y}_{i+1} = \widetilde{\mathbf{y}}_{i+1} + \mathbf{est}_i$, které je aproximací řádu $p+1$, neboť

$$\mathbf{u}(x_i + h) - \mathbf{y}_{i+1} = O(h^{p+2})\,.$$

V algoritmu (4.48) první dva řádky představují výpočet základní metodou, další dva odhad lokální chyby základní metody a v posledním řádku je popsána lokální extrapolace. Výpočet bez lokální extrapolace bychom dostali, kdybychom poslední řádek v (4.48) nahradili příkazy

$$\mathbf{y}_{i+1/2} = \widetilde{\mathbf{y}}_{i+1/2}\,, \quad \mathbf{y}_{i+1} = \widetilde{\mathbf{y}}_{i+1}\,.$$

Odhad $\mathbf{est}_i$ lokální chyby základní metody použijeme v algoritmu pro řízení délky kroku. V případě lokální extrapolace tak vlastně řídíme délku kroku extrapolované metody řádu $p+1$ pomocí odhadu lokální chyby základní metody řádu p, postupujeme tedy podle modifikovaných kritérií, která značíme XEPS místo EPS a XEPUS místo EPUS. Původní označení EPS a EPUS se ponechává pro případ, kdy se lokální extrapolace nepoužije. Řízení délky kroku extrapolované metody podle kritérií XEPS nebo XEPUS se běžně používá, funguje velmi dobře a je také teoreticky zdůvodněno.

Zamysleme se ještě nad výpočtovou náročností metody (4.48). Omezíme se na explicitní Rungovy-Kuttovy metody, p značí řád a s je počet koeficientů $\mathbf{k}_j$, tj. v každém kroku metody se funkce $\mathbf{f}$ počítá s-krát. Pak výpočet $\widetilde{\mathbf{y}}_{i+1/2}$, $\widetilde{\mathbf{y}}_{i+1}$ a $\widehat{\mathbf{y}}_{i+1}$ vyžaduje $3s-1$ vyhodnocení funkce $\mathbf{f}$. Protože počítáme ve dvou krocích délky $h/2$ hodnoty $\mathbf{y}_{i+1/2}$ a $\mathbf{y}_{i+1}$, připadá na jeden krok $(3s-1)/2$ vyhodnocení funkce $\mathbf{f}$. To představuje oproti základní metodě, kdy se počítá jen $\widetilde{\mathbf{y}}_{i+1/2}$ a $\widetilde{\mathbf{y}}_{i+1}$, zvýšení nákladů o $100\,(s-1)/(2s)\,\%$, získáváme však odhad lokální chyby a navíc metodu vyššího řádu $p+1$. Uveďme si dva konkrétní příklady.

a) Uvažme jako základní Rungovu–Kuttovu metodu řádu $p=s=2$, například metodu EM1 nebo EM2. Pak metoda (4.48) řádu 3 potřebuje 5 vyhodnocení funkce $\mathbf{f}$, tedy na jeden krok připadá 2,5 vyhodnocení. Avšak pro Rungovu–Kuttovu metodu řádu 3 je $s \geq 3$.

b) Je-li základní metodou klasická Rungova-Kuttova metoda řádu 4, tj. $p=s=4$, pak metoda (4.48) řádu 5 potřebuje 11 vyhodnocení funkce $\mathbf{f}$, tedy na jeden krok připadá 5,5 vyhodnocení. Avšak pro Rungovu–Kuttovu metodu řádu 5 je $s \geq 6$.

K největším nedostatkům metody (4.48) patří ta skutečnost, že délku kroku lze měnit vždy až po provedení dvou půlkroků. To je neefektivní, řízení provádíme vlasně jen na 50%, velmi drahé je z toho důvodu také odmítnutí kroku. Proto se v současných programech dává přednost metodě, která bývá označována jako

Metoda vnořených odhadů chyby. Základní myšlenka je jednoduchá. Použijí se dvě metody, z nichž jedna je řádu p a druhá řádu $p+1$. Z výchozí hodnoty $\mathbf{y}_i$ spočteme $\mathbf{y}^{**}_{i+1}$ přesnější metodou a $\mathbf{y}^{*}_{i+1}$ méně přesnou metodou. Pak lokální chyba méně přesné metody splňuje

$$\mathbf{le}^*_i = \mathbf{u}(x_{i+1}) - \mathbf{y}^*_{i+1} = (\mathbf{y}^{**}_{i+1} - \mathbf{y}^*_{i+1}) + (\mathbf{u}(x_{i+1}) - \mathbf{y}^{**}_{i+1}) = (\mathbf{y}^{**}_{i+1} - \mathbf{y}^*_{i+1}) + O(h^{p+2})$$

a protože $\mathbf{le}^*_i = O(h^{p+1})$, je

$$\mathbf{est}_i = \mathbf{y}^{**}_{i+1} - \mathbf{y}^*_{i+1}$$

použitelný odhad lokální chyby méně přesné metody. O vnořeném odhadu chyby mluvíme tehdy, když obě metody sdílejí společnou množinu koeficientů $\{\mathbf{k}_j\}_{j=1}^s$. V tom případě je totiž získání odhadu laciné. Pokud ve výpočtu pokračujeme přesnější metodou, tj. pro $\mathbf{y}_{i+1} = \mathbf{y}^{**}_{i+1} = \mathbf{y}^{*}_{i+1} + \mathbf{est}_i$, říkáme, že jsme použili dvojici metod s lokální extrapolací. Tento postup se v současných programech upřednostňuje. Druhou možností je pokračovat méně přesnou metodou, tj. položit $\mathbf{y}_{i+1} = \mathbf{y}^{*}_{i+1}$. V tom případě se přesnější metoda použije jen pro získání odhadu chyby a říkáme, že jsme dvojici metod použili bez lokální extrapolace.

Dvojice Rungových-Kuttových metod se popisují pomocí rozšířené Butcherovy tabulky,

c_1	a_{11}	a_{12}	$\dots$	a_{1s}
c_2	a_{21}	a_{22}	$\dots$	a_{2s}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
c_s	a_{s1}	a_{s2}	$\dots$	a_{ss}
	b^*_1	b^*_2	$\dots$	b^*_s
	b^{**}_1	b^{**}_2	$\dots$	b^{**}_s
	E_1	E_2	$\dots$	E_s

ve které jako obvykle $\mathbf{k}_j = \mathbf{f}(x_i + hc_j, \mathbf{y}_i + h\sum_{\ell=1}^s a_{j\ell}\mathbf{k}_\ell)$, $j = 1, 2, \dots, s$, přičemž

$\mathbf{y}^*_{i+1} = \mathbf{y}_i + h\sum_{j=1}^s b^*_j \mathbf{k}_j$ je metoda řádu p,

$\mathbf{y}^{**}_{i+1} = \mathbf{y}_i + h\sum_{j=1}^s b^{**}_j \mathbf{k}_j$ je metoda řádu $p+1$ a

$\mathbf{est}_i = h\sum_{j=1}^s E_j \mathbf{k}_j$ je odhad lokální chyby,

takže $E_j = b^{**}_j - b^*_j$, $j = 1, 2, \dots, s$.

Jestliže se používá lokální extrapolace, tj. $\mathbf{y}_{i+1} = \mathbf{y}^{**}_{i+1}$, připojíme k názvu metody značku $(p+1, p)$ a pokud se lokální extrapolace nepoužívá, tj. $\mathbf{y}_{i+1} = \mathbf{y}^{*}_{i+1}$, připojíme značku $(p, p+1)$. Existuje celá řada osvědčených a používaných dvojic metod. Dvě takové si uvedeme.

Bogacki–Shampine (3,2) metoda, stručně BS32 metoda. Rozšířená Butcherova tabulka BS32 metody je

0				
$\frac{1}{2}$	$\frac{1}{2}$			
$\frac{3}{4}$	0	$\frac{3}{4}$		
1	$\frac{2}{9}$	$\frac{1}{3}$	$\frac{4}{9}$	0
	$\frac{7}{24}$	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{1}{8}$
	$\frac{2}{9}$	$\frac{1}{3}$	$\frac{4}{9}$	0
	$-\frac{5}{72}$	$\frac{1}{12}$	$\frac{1}{9}$	$-\frac{1}{8}$

Přesnější z obou metod páru je Ralstonova metoda

$$\mathbf{y}^{**}_{i+1} = \mathbf{y}_i + h\left[\tfrac{2}{9}\mathbf{k}_1 + \tfrac{1}{3}\mathbf{k}_2 + \tfrac{4}{9}\mathbf{k}_3\right] = \mathbf{y}_{i+1}$$

řádu 3 s vyladěnou chybou. Pomocná metoda

$$\mathbf{y}^{*}_{i+1} = \mathbf{y}_i + h\left[\tfrac{7}{24}\mathbf{k}_1 + \tfrac{1}{4}\mathbf{k}_2 + \tfrac{1}{3}\mathbf{k}_3 + \tfrac{1}{8}\mathbf{k}_4\right]$$

řádu 2 používá kromě koeficientů $\mathbf{k}_1$, $\mathbf{k}_2$ a $\mathbf{k}_3$ navíc koeficient $\mathbf{k}_4 = \mathbf{f}(x_{i+1}, \mathbf{y}_{i+1})$. V každém kroku se tedy počítají jen 3 nové hodnoty funkce $\mathbf{f}$, neboť koeficient $\mathbf{k}_1$ ve stávajícím kroku je roven koeficientu $\mathbf{k}_4$ z kroku předchozího, takže nově se počítají jen koeficienty $\mathbf{k}_2$, $\mathbf{k}_3$ a $\mathbf{k}_4$. Výjimkou je případ, kdy se hodnota $\mathbf{y}_{i+1}$ neakceptuje a krok se krátí. Tyto případy však nebývají časté. Zařazení koeficientu $\mathbf{k}_4$ do metody řádu 2 nás tedy nic nestojí, umožní však zlepšit vlastnosti této metody a tím i celého páru. Tento postup bývá označován FSAL podle anglického *First Same As Last.*

Hodnoty přibližného řešení pro $x \in \langle x_i, x_{i+1}\rangle$ spočteme dostatečně přesně pomocí

kubického Hermitova polynomu $\mathbf{H}_3(x)$ určeného podmínkami

$$\mathbf{H}_3(x_i) = \mathbf{y}_i, \qquad \mathbf{H}_3'(x_i) = \mathbf{k}_1\,,$$
$$\mathbf{H}_3(x_{i+1}) = \mathbf{y}_{i+1}, \quad \mathbf{H}_3'(x_{i+1}) = \mathbf{k}_4\,.$$

Metoda BS32 je tedy skvělá: je řádu 3, v každém úspěšném kroku se pravá strana $\mathbf{f}$ počítá jen 3-krát a to stačí jak na řízení délky kroku tak na výpočet řešení mezi uzly x_i a x_{i+1}.

Dormand-Prince (5,4) metoda, stručně DP54 metoda je definována rozšířenou Butcherovou tabulkou

0							
$\frac{1}{5}$	$\frac{1}{5}$						
$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$					
$\frac{4}{5}$	$\frac{44}{45}$	$-\frac{56}{15}$	$\frac{32}{9}$				
$\frac{8}{9}$	$\frac{19\,372}{6\,561}$	$-\frac{25\,360}{2\,187}$	$\frac{64\,448}{6\,561}$	$-\frac{212}{729}$			
1	$\frac{9\,017}{3\,168}$	$-\frac{355}{33}$	$\frac{46\,732}{5\,247}$	$\frac{49}{176}$	$-\frac{5\,103}{18\,656}$		
1	$\frac{35}{384}$	0	$\frac{500}{1\,113}$	$\frac{125}{192}$	$-\frac{2\,187}{6\,784}$	$\frac{11}{87}$	
	$\frac{5\,179}{57\,600}$	0	$\frac{7\,571}{16\,695}$	$\frac{393}{640}$	$-\frac{92\,097}{339\,200}$	$\frac{187}{2\,100}$	$\frac{1}{40}$
	$\frac{35}{384}$	0	$\frac{500}{1\,113}$	$\frac{125}{192}$	$-\frac{2\,187}{6\,784}$	$\frac{11}{84}$	0
	$\frac{71}{57\,600}$	0	$-\frac{71}{16\,695}$	$\frac{71}{1\,920}$	$-\frac{17\,253}{339\,200}$	$\frac{22}{525}$	$-\frac{1}{40}$

Metoda DP54 je typu FSAL, neboť $\mathbf{k}_7 = \mathbf{f}(x_{i+1}, \mathbf{y}_{i+1})$. Proto se v každém úspěšném kroku metody počítají jen koeficienty $\mathbf{k}_2, \ldots, \mathbf{k}_7$, koeficient $\mathbf{k}_1$ byl už spočítán jako koeficient $\mathbf{k}_7$ v předchozím kroku. Metoda DP54 má vyladěnou chybu.

Hodnoty přibližného řešení pro $x \in \langle x_i, x_{i+1}\rangle$ spočteme dostatečně přesně pomocí interpolačního polynomu $\mathbf{H}_4(s) = \mathbf{y}_i + h\mathbf{K}\mathbf{B}\mathbf{s}$, kde $\mathbf{K} = (\mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3, \mathbf{k}_4, \mathbf{k}_5, \mathbf{k}_6, \mathbf{k}_7)$,

$$\mathbf{B} = \begin{bmatrix} 1 & -\frac{183}{64} & \frac{37}{12} & -\frac{145}{128} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1\,500}{371} & -\frac{1\,000}{159} & \frac{1\,000}{371} \\ 0 & -\frac{125}{32} & \frac{125}{12} & -\frac{375}{64} \\ 0 & \frac{9\,477}{3\,392} & -\frac{729}{106} & \frac{25\,515}{6\,784} \\ 0 & -\frac{11}{7} & \frac{11}{3} & -\frac{55}{28} \\ 0 & \frac{3}{2} & -4 & \frac{5}{2} \end{bmatrix},$$

$\mathbf{s} = (s, s^2, s^3, s^4)^T$ a $s = (x - x_i)/h$.

Metoda DP54 je rovněž vynikající: je řádu 5, v každém úspěšném kroku se pravá strana počítá jen 6-krát a to stačí jak pro řízení délky kroku tak pro výpočet řešení v intervalu $\langle x_i, x_{i+1}\rangle$.

4.3.2.5. *Implicitní a semi-implicitní metody*

V úvodu kapitoly 4 jsme ukázali, proč má smysl používat implicitní Eulerovu metodu přesto, že je ve srovnání s explicitní Eulerovou metodou výpočetně velmi náročná. Jediným důvodem je to, že EE metoda má malou oblast absolutní stability zatímco IE metoda je A-stabilní. Ukážeme si, že u Rungových-Kuttových metod je to podobné: explicitní metody mají oblast absolutní stability omezenou, takže A-stabilní metody je třeba hledat jen mezi implicitními nebo semi-implicitními metodami. A-stabilní Rungovy-Kuttovy metody se používají prakticky výhradně k řešení tzv. *tuhých problémů*. Co to tuhé problémy jsou, s jakými obtížemi se při jejich řešení setkáváme a jaké metody se pro jejich řešení hodí, to vše jsou témata, se kterými se seznámíme podrobněji v samostatném odstavci 4.5. Zde si jen uvedeme několik známých a přitom poměrně jednoduchých A-stabilních Rungových-Kuttových metod.

Začneme tím, že si připomeneme, a případně nově zavedeme, potřebné pojmy. Uvažujme tedy testovací rovnici

$$y' = \lambda y\,, \quad \text{kde } \lambda \text{ je komplexní číslo se zápornou reálnou složkou, tj. } \mathrm{Re}(\lambda) < 0.$$

Aplikujeme-li na tuto rovnici obecnou Rungovu-Kuttovu metodu, dostaneme diferenční rovnici

$$y_{i+1} = R(\widehat{h})y_i\,, \quad \text{kde } \widehat{h} = h\lambda.$$

Funkce $R(\widehat{h})$ se nazývá *funkce stability* metody. $|y_{i+1}| < |y_i|$ zřejmě platí, právě když

$$|R(\widehat{h})| < 1\,, \tag{4.49}$$

tj. metoda je absolutně stabilní pro ty hodnoty $\widehat{h}$, pro které platí (4.49). Oblast absolutní stability metody je oblast $\mathscr{R}_A$ komplexní roviny taková, že pro $\widehat{h} \in \mathscr{R}_A$ platí (4.49). Jestliže $\mathscr{R}_A$ obsahuje celou zápornou polorovinu, metoda je A-stabilní. Na řešení velmi tuhých problémů ani A-stabilní metody nestačí a používají se metody ještě silnějšího kalibru, tzv. *L-stabilní metody*. Jsou to A-stabilní metody s vlastností $|R(\widehat{h})| \to 0$ pro $\mathrm{Re}(\widehat{h}) \to -\infty$. Průnik oblasti absolutní stability s reálnou osou je interval absolutní stability.

Snadno se ověří, že pro explicitní s-stupňovou Rungovu-Kuttovu metodu je $R(\widehat{h})$ polynom stupně s a proto oblast $\mathscr{R}_A$ nemůže být neomezená. Odtud plyne závěr: explicitní Rungova-Kuttova metoda není A-stabilní. Pro $s = p \le 4$ funkce stability $R(z)$ nezávisí na volbě konkrétní metody. Skutečně, lokální řešení testovací rovnice splňuje $u(x_i+h) = e^{h\lambda}y_i$ a protože $u(x_i+h) - y_{i+1} = [e^{h\lambda} - R(h\lambda)]y_i = O(h^{p+1})$, $R(z)$ je Taylorův polynom funkce e^z stupně p. Oblasti absolutní stability se zjišťují numericky a jejich obrázky lze najít v monografiích věnovaných numerickému řešení ODR, např. v [17], [34]. Intervaly absolutní stability lze prezentovat jednodušeji. Pro $s = p = 1, 2, 3, 4$ dostáváme postupně intervaly $(-2; 0)$, $(-2; 0)$, $(-2,51; 0)$, $(-2,78; 0)$ a pro DP54 metodu interval $(-3,30; 0)$.

Pro implicitní s-stupňové Rungovy-Kuttovy metody je $R(\widehat{h})$ racionální lomená funkce, čitatel stejně jako jmenovatel je polynom stupně nejvýše s. Tak například pro IE metodu

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h\mathbf{k}_1\,, \quad \text{kde} \quad \mathbf{k}_1 = \mathbf{f}(x_i+h, \mathbf{y}_i+h\mathbf{k}_1)\,, \quad \text{tj. pro} \quad \mathbf{y}_{i+1} = \mathbf{y}_i + h\mathbf{f}(x_{i+1}, \mathbf{y}_{i+1})\,,$$

dostaneme

$$R(z) = \frac{1}{1-z}, \quad \text{takže pro } z = a+ib,\ a<0, \text{ je } |R(z)|^2 = \frac{1}{(1+|a|)^2+b^2},$$

tj. $|R(z)| < 1$ a $|R(z)| \to 0$ pro $a \to -\infty$. IE metoda je tedy L-stabilní.

Lichoběžníková metoda. Velmi známou metodou, která nechybí v žádné učebnici věnované numerickému řešení ODR, je *lichoběžníková metoda*, stručně TR metoda podle aglického *trapezoidal rule*,

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \tfrac{1}{2}h[\mathbf{f}(x_i,\mathbf{y}_i) + \mathbf{f}(x_{i+1},\mathbf{y}_{i+1})]\,. \tag{4.50}$$

Za svůj název vděčí způsobu odvození: rovnici $\mathbf{u}'(x) = \mathbf{f}(x,\mathbf{u}(x))$ integrujeme v mezích od x_i do x_{i+1} a $\int_{x_i}^{x_{i+1}} \mathbf{f}(x,\mathbf{u}(x))\,dx$ spočteme přibližně lichoběžníkovou metodou, tj.

$$\mathbf{u}(x_{i+1}) - \mathbf{u}(x_i) = \tfrac{1}{2}h[\mathbf{f}(x_i,\mathbf{u}(x_i)) + \mathbf{f}(x_{i+1},\mathbf{u}(x_{i+1}))] + O(h^3)\,.$$

TR metoda je jednokroková implicitní metoda řádu 2, kterou lze zapsat také jako semi-implicitní Rungovu-Kuttovu metodu:

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \tfrac{1}{2}h(\mathbf{k}_1+\mathbf{k}_2)\,, \quad \text{kde} \quad \mathbf{k}_1 = \mathbf{f}(x_i,\mathbf{y}_i),\ \mathbf{k}_2 = \mathbf{f}(x_i+h, \mathbf{y}_i+\tfrac{1}{2}h(\mathbf{k}_1+\mathbf{k}_2))\,.$$

Funkce stability TR metody je

$$R(z) = \frac{2+z}{2-z}, \quad \text{takže pro } z = a+ib,\ a<0, \text{ je } |R(z)|^2 = \frac{(2-|a|)^2+b^2}{(2+|a|)^2+b^2},$$

tj. $|R(z)| < 1$, $|R(z)| \to 1$ pro $a \to -\infty$. TR metoda je tedy A-stabilní, L-stabilní však není. TR metoda je základem programu ODE23T MATLABu.

TRX2 metoda. Jestliže provedeme lichoběžníkovou metodou dva kroky stejné délky $h/2$, tj.

$$\mathbf{y}_{i+1/2} = \mathbf{y}_i + \tfrac{1}{4}h[\mathbf{f}(x_i,\mathbf{y}_i) + \mathbf{f}(x_{i+1/2},\mathbf{y}_{i+1/2})]\,,$$
$$\mathbf{y}_{i+1} = \mathbf{y}_{i+1/2} + \tfrac{1}{4}h[\mathbf{f}(x_{i+1/2},\mathbf{y}_{i+1/2}) + \mathbf{f}(x_{i+1},\mathbf{y}_{i+1})]\,,$$

dostaneme semi-implicitní Rungovu-Kuttovu metodu, známou jako TRX2 metoda, kterou popíšeme pomocí rozšířené Butcherovy tabulky a vzorců

0	0	0	0
$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	0
1	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{4}$
	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{4}$
	$\frac{1}{6}$	$\frac{2}{3}$	$\frac{1}{6}$
	$-\frac{1}{12}$	$\frac{1}{6}$	$-\frac{1}{12}$

$$\mathbf{k}_1 = \mathbf{f}(x_i,\mathbf{y}_i)\,,$$
$$\mathbf{k}_2 = \mathbf{f}(x_i+\tfrac{1}{2}h, \mathbf{y}_i+\tfrac{1}{4}h(\mathbf{k}_1+\mathbf{k}_2))\,,$$
$$\mathbf{k}_3 = \mathbf{f}(x_i+h, \mathbf{y}_i+\tfrac{1}{4}h(\mathbf{k}_1+2\mathbf{k}_2+\mathbf{k}_3))\,,$$
$$\mathbf{y}_{i+1} = \mathbf{y}_i + \tfrac{1}{4}h(\mathbf{k}_1+2\mathbf{k}_2+\mathbf{k}_3)\,,$$
$$\mathbf{est}_i = \tfrac{1}{12}h(-\mathbf{k}_1+2\mathbf{k}_2-\mathbf{k}_3)\,.$$

TRX2 metoda je A-stabilní metoda řádu 2, používá se jako (2, 3) metoda, tj. bez lokální extrapolace, přesnější metoda řádu 3 se použije jen k získání odhadu $\mathbf{est}_i$ lokální chyby, k výpočtu přibližného řešení ji použít nelze, neboť není A-stabilní. TRX2 metodu je třeba kvalitně implementovat, viz [14].

TR-BDF2 metoda. TRX2 metoda není L-stabilní. Existuje však podobná metoda řádu 2, která L-stabilní je. Je to metoda označovaná jako TR-BDF2: tato metoda používá nejdříve lichoběžníkovou metodu, odtud TR v jejím názvu, a pak metodu zpětného derivování řádu 2, zkráceně BDF2 metodu, viz odstavec 4.4.3. Výpočet je popsán vzorci

$$\text{TR:} \quad \mathbf{y}_{i+\gamma} = \mathbf{y}_i + \tfrac{1}{2}\gamma h[\mathbf{f}(x_i,\mathbf{y}_i) + \mathbf{f}(x_{i+\gamma},\mathbf{y}_{i+\gamma})]\,,$$
$$\text{BDF2:} \quad \frac{2-\gamma}{1-\gamma}\mathbf{y}_{i+1} - \frac{1}{\gamma(1-\gamma)}\mathbf{y}_{i+\gamma} + \frac{1-\gamma}{\gamma}\mathbf{y}_i = h\mathbf{f}(x_{i+1},\mathbf{y}_{i+1})\,,$$

kde $\gamma = 2-\sqrt{2}$ a $x_{i+\gamma} = x_i + h\gamma$. TR-BDF2 metodu lze zapsat jako semi-implicitní Rungovu-Kuttovu metodu pomocí rozšířené Butcherovy tabulky a vzorců

0	0	0	0
γ	d	d	0
1	w	w	d
	w	w	d
	$\frac{1}{3}(1-w)$	$\frac{1}{3}(3w+1)$	$\frac{1}{3}d$
	$\frac{1}{3}(1-4w)$	$\frac{1}{3}$	$-\frac{2}{3}d$

$$\mathbf{k}_1 = \mathbf{f}(x_i,\mathbf{y}_i)\,,$$
$$\mathbf{k}_2 = \mathbf{f}(x_i+\gamma h, \mathbf{y}_i + hd(\mathbf{k}_1+\mathbf{k}_2))\,,$$
$$\mathbf{k}_3 = \mathbf{f}(x_i+h, \mathbf{y}_i + h(w\mathbf{k}_1 + w\mathbf{k}_2 + d\mathbf{k}_3))\,,$$
$$\mathbf{y}_{i+1} = \mathbf{y}_i + h(w\mathbf{k}_1 + w\mathbf{k}_2 + d\mathbf{k}_3)\,,$$
$$\mathbf{est}_i = \tfrac{1}{3}h[(1-4w)\mathbf{k}_1 + \mathbf{k}_2 - 2d\mathbf{k}_3]\,,$$

kde $d = \gamma/2$ a $w = \sqrt{2}/4$. Metoda řádu 3 se opět používá jen pro odhad $\mathbf{est}_i$ lokální chyby. Podrobnosti o TR-BDF2 metodě lze najít v [14]. Metoda je implementována v MATLABu jako program ODE23TB.

Je známo, že existují implicitní a semi-implicitní Rungovy-Kuttovy metody, které jsou A-stabilní nebo i L-stabilní a jsou přitom libovolně vysokého řádu, viz například [17]. To je zajímavé z teoretického hlediska, v praxi se však takové metody vyšších řádů příliš nepoužívají, neboť algoritmy na nich založené jsou výpočtově velmi náročné.

4.3.2.6. *Odhad globální chyby*

je založen na obdobné myšlence jako odhad lokální chyby metodou poloviční kroku, viz odstavec 4.3.2.4. Stejné je to, že $\overline{\mathbf{y}}_{i+1}$ spočteme v jednom kroku délky h z výchozí hodnoty $\mathbf{y}_i$. Stejné je také to, že $\widetilde{\mathbf{y}}_{i+1}$ spočteme pomocí dvou kroků poloviční délky $h/2$, jediný rozdíl je v tom, že tentokrát vyjdeme z výchozí hodnoty $\widetilde{\mathbf{y}}_i$. Položíme tedy $\mathbf{y}_0 = \widetilde{\mathbf{y}}_0 = \boldsymbol{\eta}$ a počítáme

$$\overline{\mathbf{y}}_{i+1} = \mathbf{y}_i + h\Phi(x_i,\mathbf{y}_i,\overline{\mathbf{y}}_{i+1},h;\mathbf{f})\,,$$
$$\widetilde{\mathbf{y}}_{i+1/2} = \widetilde{\mathbf{y}}_i + \tfrac{1}{2}h\Phi(x_i,\widetilde{\mathbf{y}}_i,\widetilde{\mathbf{y}}_{i+1/2},\tfrac{1}{2}h;\mathbf{f})\,,$$
$$\widetilde{\mathbf{y}}_{i+1} = \widetilde{\mathbf{y}}_{i+1/2} + \tfrac{1}{2}h\Phi(x_{i+1/2},\widetilde{\mathbf{y}}_{i+1/2},\widetilde{\mathbf{y}}_{i+1},\tfrac{1}{2}h;\mathbf{f})\,.$$

Dále využijeme toho, že pro metodu řádu p platí

$$\overline{\mathbf{y}}_{i+1} = \mathbf{y}(x_{i+1}) + c_i H^p + O(H^{p+1}),$$

$$\widetilde{\mathbf{y}}_{i+1} = \mathbf{y}(x_{i+1}) + c_i \left(\frac{H}{2}\right)^p + O(H^{p+1}),$$

kde $H = \max h$ je nejdelší krok a c_i je konstanta nezávislá na H (jde o hluboký teoretický výsledek, jehož důkaz lze najít např. v [34]). Odečtením dvou posledních výrazů obdržíme

$$\overline{\mathbf{y}}_{i+1} - \widetilde{\mathbf{y}}_{i+1} = (2^p - 1) c_i \left(\frac{H}{2}\right)^p + O(H^{p+1})$$

a odtud pro globální chybu $\widetilde{\mathbf{e}}_{i+1} := \mathbf{y}(x_{i+1}) - \widetilde{\mathbf{y}}_{i+1}$ přesnější metody dostaneme

$$\widetilde{\mathbf{e}}_{i+1} = \mathbf{y}(x_{i+1}) - \widetilde{\mathbf{y}}_{i+1} = -c_i \left(\frac{H}{2}\right)^p + O(H^{p+1}) = \frac{1}{2^p - 1}(\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1}) + O(H^{p+1}),$$

takže jako odhad globální chyby $\widetilde{\mathbf{e}}_{i+1}$ lze použít výraz

$$\Delta_{i+1} := \frac{1}{2^p - 1}(\widetilde{\mathbf{y}}_{i+1} - \overline{\mathbf{y}}_{i+1}).$$

Považujeme-li za výsledky přibližné hodnoty $\widetilde{\mathbf{y}}_{i+1/2}$, $\widetilde{\mathbf{y}}_{i+1}$ vidíme, že odhad globální chyby zvyšuje výpočtové náklady přibližně o 50%. Postup, který jsme použili ke stanovení odhadu Δ_{i+1} globální chyby, se nazývá *globální extrapolace.*

Na závěr připojíme dvě poznámky:

1) Přibližné řešení $\widetilde{\mathbf{y}}_{i+1}$ lze zpřesnit: položíme-li

   $$\widehat{\mathbf{y}}_{i+1} = \widetilde{\mathbf{y}}_{i+1} + \Delta_{i+1},$$

   dostaneme $\mathbf{y}(x_{i+1}) - \widehat{\mathbf{y}}_{i+1} = O(H^{p+1})$.

2) Délku kroku řídíme pomocí odhadu $\mathbf{est}_i$ lokální chyby $\overline{\mathbf{le}}_i = \mathbf{u}(x_{i+1}) - \overline{\mathbf{y}}_{i+1}$ méně přesné metody.

4.4. Lineární mnohokrokové metody

V této kapitole se budeme zabývat metodami, které počítají přibližné řešení $\mathbf{y}_{i+1}$ v uzlu x_{i+1} pomocí dříve spočtených hodnot $\mathbf{y}_i, \mathbf{y}_{i-1}, \mathbf{y}_{i-2}, \ldots$ a odpovídajících hodnot $\mathbf{f}(x_i, \mathbf{y}_i)$, $\mathbf{f}(x_{i-1}, \mathbf{y}_{i-1}), \mathbf{f}(x_{i-2}, \mathbf{y}_{i-2}), \ldots$ pravé strany diferenciální rovnice. Tyto hodnoty jsou znovu použity tak, abychom získali $\mathbf{y}_{i+1}$ s vysokou přesností pomocí jen několika málo nových vyhodnocení funkce $\mathbf{f}(x, \mathbf{y})$. Nejznámějšími metodami tohoto typu jsou *Adamsovy metody* a *metody zpětného derivování.* Obě skupiny metod patří do obecné třídy metod známých jako *lineární mnohokrokové metody*, stručně LMM.

4.4.1. Obecná lineární mnohokroková metoda

Pro konstantní délku h kroku je lineární mnohokroková metoda předpis

$$\alpha_0 \mathbf{y}_{i+1} + \alpha_1 \mathbf{y}_i + \cdots + \alpha_k \mathbf{y}_{i+1-k} = h[\beta_0 \mathbf{f}(x_{i+1}, \mathbf{y}_{i+1}) + \beta_1 \mathbf{f}_i + \cdots + \beta_k \mathbf{f}_{i+1-k}], \tag{4.51}$$

ze kterého počítáme $\mathbf{y}_{i+1}$. Přitom α_j a β_j jsou reálné koeficienty, které formuli jednoznačně určují, a $\mathbf{f}_j$ je zkrácený zápis pro $\mathbf{f}(x_j, \mathbf{y}_j)$. V dalším budeme předpokládat, že $\alpha_0 \neq 0$. Jestliže alespoň jeden z koeficientů α_k nebo β_k je různý od nuly, *metoda* je *k-kroková.*

Pro $\beta_0 \neq 0$ je nová hodnota $\mathbf{y}_{i+1}$ určena implicitně, hovoříme proto o *implicitní metodě*, pro $\beta_0 = 0$ máme *metodu explicitní.* Abychom v implicitní metodě určili $\mathbf{y}_{i+1}$, musíme vyřešit rovnici

$$\mathbf{y}_{i+1} = \boldsymbol{\varphi}(\mathbf{y}_{i+1}), \quad \text{kde} \quad \boldsymbol{\varphi}(\mathbf{z}) = h\frac{\beta_0}{\alpha_0}\mathbf{f}(x_{i+1}, \mathbf{z}) + \frac{1}{\alpha_0}\sum_{j=1}^{k}[h\beta_j \mathbf{f}_{i+1-j} - \alpha_j \mathbf{y}_{i+1-j}].$$

Pro dostatečně malé h má tato rovnice jediné řešení. Skutečně, užitím Lipschitzovy podmínky (4.4) dostaneme $\|\boldsymbol{\varphi}(\mathbf{u}) - \boldsymbol{\varphi}(\mathbf{v})\| \leq h(\beta_0/\alpha_0)L\|\mathbf{u} - \mathbf{v}\|$, takže $\boldsymbol{\varphi}(\mathbf{z})$ je pro $h(\beta_0/\alpha_0)L < 1$ kontrakce a jednoznačná existence $\mathbf{y}_{i+1}$ je důsledkem věty o pevném bodu.

LMM lze použít, jen když jsou zadány *startovací hodnoty* $\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{k-1}$. $\mathbf{y}_0$ určíme z počáteční podmínky, zbývající startovací hodnoty je však třeba získat jinou vhodnou metodou, $\mathbf{y}_r$ metodou nejvýše r-krokovou.

Všimněte si, že výsledky získané formulí se nezmění, když formuli normalizujeme tak, že všechny koeficienty vydělíme nenulovým číslem. Používají se různé typy normalizací. Často se volí $\alpha_0 = 1$. My použijeme jinou normalizaci zahrnující všechny koeficienty α_j. K tomu účelu se nám bude hodit *(první) charakteristický polynom* $\varrho(\Theta)$ definovaný předpisem

$$\varrho(\Theta) = \alpha_0 \Theta^k + \alpha_1 \Theta^{k-1} + \cdots + \alpha_k.$$

V dalším budeme za *normalizovanou* považovat takovou formuli, pro kterou $\varrho'(1) = 1$.

Lokální diskretizační chyba. Chybu, které se dopustíme v jednom kroku, měříme pomocí lokální diskterizační chyby $\mathbf{lte}_i$ definované rovnicí

$$\sum_{j=0}^{k} \alpha_j \mathbf{y}(x_{i+1-j}) = h\sum_{j=0}^{k} \beta_j \mathbf{f}(x_{i+1-j}, \mathbf{y}(x_{i+1-j})) + \mathbf{lte}_i,$$

nebo ekvivalentně

$$\sum_{j=0}^{k} \alpha_j \mathbf{y}(x_{i+1-j}) = h\sum_{j=0}^{k} \beta_j \mathbf{y}'(x_{i+1-j}) + \mathbf{lte}_i.$$

Lokální diskretizační chyba je tedy chyba, která vznikne, když do normalizované formule (4.51) dosadíme místo přibližného řešení $\mathbf{y}_{i+1-j}$ přesné řešení $\mathbf{y}(x_{i+1-j})$.

Chybu $\mathbf{lte}_i$ rozvineme do mocnin h tak, že provedeme Taylorův rozvoj všech funkcí okolo bodu x_{i+1}. Pro řešení samotné dostaneme

$$\begin{aligned}\mathbf{y}(x_{i+1-j}) &= \mathbf{y}(x_{i+1}) - jh\mathbf{y}'(x_{i+1}) + \frac{(-jh)^2}{2!}\mathbf{y}''(x_{i+1}) + \dots \\ &= \mathbf{y}(x_{i+1}) + \sum_{s=1}^{p+1} \frac{(-jh)^s}{s!}\mathbf{y}^{(s)}(x_{i+1}) + O(h^{p+2})\end{aligned}$$

a podobně pro derivaci

$$\mathbf{y}'(x_{i+1-j}) = \mathbf{y}'(x_{i+1}) - jh\mathbf{y}''(x_{i+1}) + \dots = \sum_{s=1}^{p+1} \frac{(-jh)^{s-1}}{(s-1)!}\mathbf{y}^{(s)}(x_{i+1}) + O(h^{p+2}).$$

Když tyto rozvoje dosadíme do vzorce pro lokální diskretizační chybu, obdržíme

$$\mathbf{lte}_i = \sum_{s=0}^{p+1} C_s h^s \mathbf{y}^{(s)}(x_{i+1}) + O(h^{p+2}),$$

kde

$$C_0 = \sum_{j=0}^{k} \alpha_j \quad \text{a} \quad C_s = (-1)^s \sum_{j=0}^{k} \left[\frac{j^s\alpha_j}{s!} + \frac{j^{s-1}\beta_j}{(s-1)!}\right] \quad \text{pro } s = 1, 2, \dots, p+1.$$

Řád metody. Stejně jako u jednokrokových metod řekneme, že LMM je řádu p, jestliže $\mathbf{lte}_i = O(h^{p+1})$, tj. pokud $C_0 = C_1 = \dots = C_p = 0$ a $C_{p+1} \neq 0$. Všimněte si, že výraz pro $\mathbf{lte}_i$ zůstane v platnosti, když uzel x_{i+1} nahradíme bodem $x^* = x_{i+1} + O(h)$, například uzlem x_i. Pro metodu řádu p se člen $C_{p+1}h^{p+1}\mathbf{y}^{(p+1)}(x_i)$ nazývá *hlavní člen lokální diskretizační chyby* a konstanta C_{p+1} je tzv. *chybová konstanta*. Ta závisí na normalizaci LMM, takže pokud chceme srovnávat přesnost dvou formulí, musíme si být jisti, že jsou stejně normalizovány.

Podmínky $C_s = 0$ pro $s \leq p$ jsou ekvivalentní podmínkám pro dosažení řádu p u Rungových - Kuttových metod. Zatímco u Rungových - Kuttových metod představovaly podmínky řádu nelineární rovnice pro koeficienty metody, u LMM dostáváme v důsledku linearity jen rovnice lineární.

Pro LMM lze definovat také *lokální chybu* $\mathbf{le}_i$ jako lokální diskretizační chybu lokálního řešení $\mathbf{u}_i(x)$ splňujícího $\mathbf{u}_i' = \mathbf{f}(x, \mathbf{u}_i)$, $\mathbf{u}_i(x_i) = \mathbf{y}_i$. Význam takto definovaného lokálního řešení je však omezen tím, že nelze předpokládat $\mathbf{u}_i(x_{i+1-j}) = \mathbf{y}_{i+1-j}$, $j = 2, 3, \dots, k$. Na rozdíl od jednokrokových metod tedy lokální chyba $\mathbf{le}_i$ v případě LMM nemá bezprostřední názornou interpretaci a proto ji nebudeme používat.

Metoda konzistententní s ODR. Dá se ukázat, že každá *konvergentní LMM musí být řádu* $p \geq 1$, viz např. [38]. LMM řádu alespoň 1 se nazývá *metoda konzistentní (s diferenciální rovnicí)*. V dalším budeme uvažovat jen konzistentní LMM. Všimněte si, že podmínku $C_0 = 0$ lze ekvivalentně zapsat jako $\varrho(1) = 0$. Abychom podobně vyjádřili také druhou podmínku konzistence, tj. podmínku $C_1 = 0$, definujeme *druhý charakteristický polynom*

$$\sigma(\Theta) = \beta_0\Theta^k + \beta_1\Theta^{k-1} + \dots + \beta_k.$$

Z podmínky

$$C_1 = -\sum_{j=0}^{k}[j\alpha_j + \beta_j] = 0 \quad \text{dostaneme} \quad \sigma(1) = \sum_{j=0}^{p}\beta_j = -\sum_{j=0}^{k} j\alpha_j,$$

a dále

$$\varrho'(\Theta) = \sum_{j=0}^{k}(k-j)\alpha_j\Theta^{k-j-1} \quad \text{a odtud} \quad \varrho'(1) = k\sum_{j=0}^{k}\alpha_j - \sum_{j=1}^{k} j\alpha_j = k\varrho(1) + \sigma(1).$$

Konzistentní LMM je tedy ekvivalentně charakterizována dvěma podmínkami na charakteristické polynomy: $\varrho(1) = 0$ a $\varrho'(1) = \sigma(1)$.

D-stabilita. Jednokroková metoda řádu $p \geq 1$ je vždy konvergentní, pro LMM metodu to však neplatí. Abychom si objasnili v čem je problém, zabývejme se řešením triviální úlohy

$$y'(x) = 0 \quad \text{pro } x \in \langle 0, 1\rangle \text{ s počáteční podmínkou } y(0) = 0.$$

Pokud startovací hodnoty zvolíme přesně, tj. $y_r = 0$ pro $0 \leq r < k$, pak $y_i = 0$ pro všechna i. Prozkoumejme co se stane, když startovací hodnoty nepatrně pozměníme, tj. řešíme-li úlohu

$$\begin{aligned} z_r &= \Delta_r \qquad \text{pro} \quad 0 \leq r < k, \\ \sum_{j=0}^{k}\alpha_j z_{i+1-j} &= 0 \qquad \text{pro} \quad i \geq k-1, \end{aligned}$$

kde Δ_r jsou malé poruchy. z_i je řešení diferenční rovnice s konstantními koeficienty. Taková rovnice se řeší podobně jako diferenciální rovnice s konstantními koeficienty: řešení navrhneme ve tvaru $z_i = \gamma\zeta^i$, kde γ a ζ jsou zatím neurčené konstanty, a dosadíme ho do diferenční rovnice. Tak dostaneme

$$\sum_{j=0}^{k}\alpha_j\gamma\zeta^{i+1-j} = \gamma\zeta^{i+1-k}\sum_{j=0}^{k}\alpha_j\zeta^{k-j} = \gamma\zeta^{i+1-k}\varrho(\zeta) = 0,$$

takže je-li ζ kořen charakteristického polynomu $\varrho(\Theta)$, je z_i řešení diferenční rovnice.

Vybereme-li startovací hodnoty $\Delta_r = \gamma\zeta^r$, $r = 0, 1, \dots, k-1$, pak $z_i = \gamma\zeta^i$ je řešení pozměněné úlohy. Nechť N je počet stejných dílků, na které je interval $\langle 0, 1\rangle$ rozdělen. z_N je tedy aproximace přesné hodnoty $y(1) = 0$, která by měla být tím lepší, čím je N větší. Avšak $|y_N - z_N| = |z_N| = \gamma|\zeta|^N$, takže pro $|\zeta| > 1$ je $\lim_{N\to\infty}|y_N - z_N| = \infty$. To je ale zcela nepřijatelné: z malých počátečních poruch vznikla koncová porucha, jejíž velikost s rostoucím N neomezeně roste.

Je-li $\zeta = re^{I\varphi}$, $I = \sqrt{-1}$, komplexní kořen velikosti $r > 1$, předchozí analýza zůstává v platnosti, pokud bychom se však chtěli vyhnout počítání s komplexními čísly, stačí uvažovat $z_i = \gamma r^i\cos(i\varphi)$. Vyloučit je třeba také případ, kdy $\zeta = e^{I\varphi}$ je násobný kořen velikosti 1, pak totiž $z_i = \gamma i\cos(i\varphi)$ je řešením diferenční rovnice a opět $\lim_{N\to\infty}|z_N| = \infty$.

Předchozí analýza nás opravňuje zavést nový pojem: řekneme, že LMM je *stabilní ve smyslu Dahlquista* nebo stručně *D-stabilní*, jestliže pro všechny kořeny ζ_s charakteristického polynomu $\varrho(\Theta)$ platí $|\zeta_s| \le 1$, a pokud $|\zeta_s| = 1$, pak je kořen ζ_s jednoduchý.

Konvergence. Nyní již můžeme vyslovit nutnou a postačující podmínku zaručující konvergenci LMM: *LMM je konvergentní, tj.* $\max_{0\le i\le N} \|\mathbf{y}(x_i) - \mathbf{y}_i\| \to 0$ *pro* $h \to 0$, *právě když je D-stabilní řádu alespoň 1.* Důkaz viz např. [38]. Toto tvrzení lze stručně vyjádřit schématem

$$\text{konzistence} + \text{D-stabilita} \iff \text{konvergence}$$

Pro rychlost konvergence LMM platí následující

Věta *(o rychlosti konvergence LMM). Uvažujme D-stabilní LMM řádu* $p \ge 1$. *Jestliže startovací hodnoty zadáme s chybou řádu* $O(h^p)$, *pak metoda je řádu p, tj. pro globální diskretizační chybu* $\mathbf{e}_i = \mathbf{y}(x_i) - \mathbf{y}_i$ *platí* $\mathbf{e}_i = O(h^p)$.

Precizní formulaci a důkaz této věty lze najít např. v [38], [17]. Předpokladem její platnosti je dostatečná hladkost pravé strany $\mathbf{f}$, konkrétně je třeba, aby funkce $\mathbf{f}(x, \mathbf{y})$ měla spojité derivace až do řádu p včetně. Pokud $\mathbf{f}$ má spojité derivace jen do řádu $s \le p$, pak lze pro globální chybu dokázat pouze řád $O(h^s)$.

Absolutní stabilita. D-stabilita souvisí s konvergencí a tedy s délkou kroku $h \to 0$, v anglicky psané literatuře se proto místo D-stabilita často používá ekvivalentní termín „zero-stability". Při praktickém výpočtu však počítáme s konkrétní nenulovou délkou kroku a proto je třeba požadovat od numerické metody více než jen D-stabilitu. Vhodným nástrojem pro formulování takového požadavku je testovací úloha

$$y' = \lambda y \quad \text{pro } x \in (0, \infty),\ y(0) \text{ je libovolné, } \lambda \text{ je komplexní číslo, } \mathrm{Re}(\lambda) < 0\,.$$

V tom případě totiž $y(x) \to 0$ pro $x \to +\infty$ a proto je žádoucí, aby při pevně zvolené délce h kroku $y_i \to 0$ pro $x_i = ih \to \infty$. Testovací úlohu řešíme LMM a dostaneme lineární diferenční rovnici

$$\sum_{j=0}^{k} (\alpha_j - h\lambda\beta_j) y_{i+1-j} = 0\,.$$

Předpokládáme $y_i = \gamma\zeta^i$, dosadíme do diferenční rovnice a obdržíme

$$\sum_{j=0}^{k} (\alpha_j - h\lambda\beta_j)\gamma\zeta^{i+1-j} = \gamma\zeta^{i+1-k} \sum_{j=0}^{k} (\alpha_j - h\lambda\beta_j)\zeta^{k-j} = \gamma\zeta^{i+1-k}\pi(\zeta, \widehat{h}) = 0\,,$$

kde $\widehat{h} = h\lambda$ a

$$\pi(z, \widehat{h}) = \sum_{j=0}^{k} (\alpha_j - \widehat{h}\beta_j) z^{k-j} = \varrho(z) - \widehat{h}\sigma(z)$$

je charakteristický polynom diferenční rovnice, označovaný také jako *polynom stability* LMM. Je-li ζ_s kořen polynomu stability $\pi(z, \widehat{h})$, pak ζ_s^i je zřejmě řešení diferenční rovnice.

Je-li kořen ζ_s násobný, pak další řešení diferenční rovnice jsou $i^n\zeta_s^i$ pro $n = 1, 2, \ldots, m_s - 1$, kde m_s je násobnost kořene ζ_s.

Vidíme tedy, že $y_i \to 0$ pro $i \to \infty$, právě když $|\zeta_s| < 1$. Označme $\mathscr{R}_A$ množinu těch bodů $\widehat{h}$ komplexní roviny, pro které všechny kořeny ζ_s polynomu stability $\pi(z, \widehat{h})$ leží uvnitř jednotkového kruhu $|z| < 1$. Pak $\widehat{h} \in \mathscr{R}_A \Rightarrow y_i \to 0$ pro $i \to \infty$. Oblast $\mathscr{R}_A$ nazýváme oblastí absolutní stability LMM. Jsou-li všechny kořeny polynomu stability $\pi(z, \widehat{h})$ jednoduché, pak zřejmě $|y_{i+1}| < |y_i|$, takže definice absolutní stability LMM je ve shodě s definicí absolutní stability zavedené pro jednokrokové metody, viz (4.18) a (4.49).

Velikost oblasti $\mathscr{R}_A$ absolutní stability je významnou charakteristikou LMM. Jak uvidíme v odstavci 4.5, metody s velkou oblastí absolutní stability se používají při řešení tuhých problémů. Pro řešení tuhých problémů se výtečně hodí metody zpětného derivování, seznámíme se s nimi v odstavci 4.4.3. Metody prediktor-korektor založené na Adamsových metodách, viz následující odstavec 4.4.2, mají poměrně malou oblast absolutní stability a proto se pro řešení tuhých problémů nehodí. To ale vůbec neznamená, že to jsou metody špatné, naopak, jsou to velmi dobré a hojně používané metody pro řešení problémů, které tuhé nejsou.

Přibližné řešení mezi uzly x_i a x_{i+1} získáme interpolací z hodnot $\mathbf{y}_{i+1}, \mathbf{y}_i, \ldots, \mathbf{y}_{i+1-k}$.

4.4.2. Adamsovy metody

Integrací diferenciální rovnice (4.4) od x_i do $x_{i+1} = x_i + h_i$ dostaneme

$$\mathbf{y}(x_{i+1}) - \mathbf{y}(x_i) = \int_{x_i}^{x_{i+1}} \mathbf{f}(x, \mathbf{y}(x))\,\mathrm{d}x.$$

Funkci $\mathbf{f}(x, \mathbf{y}(x))$ aproximujeme pomocí interpolačního polynomu $\mathbf{P}_{\nu-1}(x)$ stupně $\nu - 1$ a dostaneme metodu

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \int_{x_i}^{x_{i+1}} \mathbf{P}_{\nu-1}(t)\,\mathrm{d}t\,. \tag{4.52}$$

4.4.2.1. *Adamsovy-Bashforthovy metody*

Jestliže interpolační polynom určíme podmínkami

$$\mathbf{P}_{\nu-1}(x_{i+1-j}) = \mathbf{f}_{i+1-j}\,, \quad j = 1, 2, \ldots, \nu\,,$$

dostaneme ν-krokovou *Adamsovu-Bashforthovu metodu*, stručně ABν metodu. Když vyjádříme interpolační polynom v Lagrangeově tvaru

$$\mathbf{P}_{\nu-1}(x) = \sum_{j=1}^{\nu} \mathbf{f}_{i+1-j}\, l_j(x)\,, \quad \text{kde} \quad l_j(x) = \prod_{\substack{k=1 \\ k\ne j}}^{\nu} \frac{x - x_{i+1-k}}{x_{i+1-j} - x_{i+1-k}}\,, \ j = 1, 2, \ldots, \nu\,,$$

obdržíme ABν metodu ve tvaru

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h_i \sum_{j=1}^{\nu} \beta^*_{\nu,j}\, \mathbf{f}_{i+1-j}\,, \tag{4.53}$$

kde

$$\beta^*_{\nu,j} = \frac{1}{h_i}\int_{x_i}^{x_{i+1}} \prod_{\substack{k=1\\k\neq j}}^{\nu} \frac{t - x_{i+1-k}}{x_{i+1-j} - x_{i+1-k}}\,\mathrm{d}t\,, \quad j = 1,2,\ldots,\nu\,.$$

Koeficienty $\beta^*_{\nu,j}$ závisejí na vzájemných vzdálenostech uzlů x_{i+1-j} a proto se musejí počítat v každém kroku. Jsou však známy postupy, jak tyto koeficienty počítat efektivně, navíc výpočtové náklady lze snížit tím, že délka kroku se změní, jen když se vyplatí spočítat novou sadu koeficientů. Druhou, tradiční možností je pracovat s krokem konstantní délky h,

$$x_{i+1-j} = x_{i+1} - jh, \quad j = 1,2,\ldots,\nu,$$

pak koeficienty $\beta^*_{\nu,j}$ na h nezávisejí,

$$\beta^*_{\nu,j} = \int_0^1 \prod_{\substack{k=1\\k\neq j}}^{\nu} \frac{k-s}{k-j}\,\mathrm{d}s, \quad j = 1,2,\ldots,\nu.$$

To ale neznamená, že bychom měli celý výpočet provádět s pevnou délkou kroku. Jestliže se rozhodneme pokračovat z x_i do $x_i + h^*$ s novou délkou kroku h^*, pak přibližné hodnoty $\mathbf{f}^*_{i+1-k}$ pravé strany $\mathbf{f}(x,\mathbf{y}(x))$ v uzlech $x_i - h^*, x_i - 2h^*, \ldots, x_i - (\nu-1)h^*$ stačí spočítat interpolací z původních hodnot $\mathbf{f}_{i+1-k}$, tj. určíme

$$\mathbf{f}^*_{i+1-k} = \mathbf{P}_{\nu-1}(x_i - (k-1)h^*) = \sum_{j=1}^{\nu} \mathbf{f}_{i+1-j}\, l_j(x_i - (k-1)h^*)\,, \quad k = 2,3,\ldots,\nu.$$

Jeden krok Adamsovy-Bashforthovy metody vyžaduje jen jediné vyčíslení funkce $\mathbf{f}$. Skutečně, předpokládejme, že jsme v uzlu x_i, právě jsme spočetli $\mathbf{y}_i$ a máme k dispozici $\mathbf{y}_{i-1}, \mathbf{y}_{i-2}, \ldots$ v předchozích uzlech $x_{i-1}, x_{i-2}, \ldots$ a dále také hodnoty $\mathbf{f}_{i-1}, \mathbf{f}_{i-2}, \ldots$. Abychom mohli provést další krok, musíme spočítat $\mathbf{f}(x_i,\mathbf{y}_i)$, pokud je to zapotřebí tak také koeficienty $\beta^*_{\nu,j}$, a nakonec užitím metody (4.53) vypočteme $\mathbf{y}_{i+1}$.

Snadno ověříme, že nejjednodušší Adamsova-Bashforthova metoda, kterou dostaneme pro $\nu = 1$, tedy AB1 metoda, není nic jiného než nám dobře známá jednokroková explicitní Eulerova metoda. První vícekrokovou metodou je AB2 metoda. V tom případě

$$\beta^*_{2,1} = \frac{1}{h_i}\int_{x_i}^{x_i+h_i} \frac{t - x_{i-1}}{x_i - x_{i-1}}\,\mathrm{d}t = 1 + \frac{1}{2}\frac{h_i}{h_{i-1}}, \quad \beta^*_{2,2} = \frac{1}{h_i}\int_{x_i}^{x_i+h_i} \frac{t - x_i}{x_{i-1} - x_i}\,\mathrm{d}t = -\frac{1}{2}\frac{h_i}{h_{i-1}},$$

kde $h_i = x_{i+1} - x_i$ a $h_{i-1} = x_i - x_{i-1}$. Pro konstantní délku kroku nabývá AB2 metoda tvaru

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h\left[\frac{3}{2}\mathbf{f}_i - \frac{1}{2}\mathbf{f}_{i-1}\right].$$

Formule pro konstantní délku kroku. V následující tabulce uvádíme koeficienty ABν metody pro $\nu = 1,2,\ldots,6$ a pro konstantní délku kroku:

ν	$\beta^*_{\nu,1}$	$\beta^*_{\nu,2}$	$\beta^*_{\nu,3}$	$\beta^*_{\nu,4}$	$\beta^*_{\nu,5}$	$\beta^*_{\nu,6}$
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	$\frac{1\,901}{720}$	$-\frac{2\,774}{720}$	$\frac{2\,616}{720}$	$-\frac{1\,274}{720}$	$\frac{251}{720}$	
6	$\frac{4\,277}{1\,440}$	$-\frac{7\,923}{1\,440}$	$\frac{9\,982}{1\,440}$	$-\frac{7\,298}{1\,440}$	$\frac{2\,877}{1\,440}$	$-\frac{475}{1\,440}$

Tvar (4.53) AB metody je důsledkem použitého Lagrangeova tvaru interpolačního polynomu. Pokud používáme krok konstantní délky h, je účelné použít Newtonův tvar interpolačního polynomu

$$\mathbf{P}_{\nu-1}(x) = \mathbf{f}_i + \frac{x - x_i}{h}\nabla\mathbf{f}_i + \cdots + \frac{(x-x_i)(x-x_{i-1})\ldots(x-x_{i+2-\nu})}{h^{\nu-1}(\nu-1)!}\nabla^{\nu-1}\mathbf{f}_i\,,$$

kde operátor zpětné diference je

$$\nabla^0\mathbf{f}_i = \mathbf{f}_i, \quad \nabla\mathbf{f}_i = \mathbf{f}_i - \mathbf{f}_{i-1}, \quad \text{a obecně} \quad \nabla^{j+1}\mathbf{f}_i = \nabla(\nabla^j\mathbf{f}_i)\,.$$

Adamsova-Bashforthova metoda, která vznikne integrací Newtonova tvaru interpolačního polynomu, je tvaru

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h\sum_{j=1}^{\nu} \gamma^*_{j-1}\nabla^{j-1}\mathbf{f}_i\,, \tag{4.54}$$

kde $\gamma^*_0 = 1$ a pro $j \geq 1$

$$\gamma^*_j = \int_{x_i}^{x_i+h} \frac{(t-x_i)\ldots(t-x_{i+1-j})}{h^j j!}\,\mathrm{d}t = \frac{1}{j!}\int_0^1 s(s+1)\ldots(s+j-1)\,\mathrm{d}s\,.$$

Všimněte si, že γ^*_j nezávisí na ν. Toho lze využít: označíme-li $\mathbf{y}^{\nu}_{i+1}$ řešení získané ν-krokovou metodou a $\mathbf{y}^{\nu+1}_{i+1}$ řešení získané metodou $(\nu+1)$-krokovou, pak

$$\mathbf{y}^{\nu+1}_{i+1} = \mathbf{y}^{\nu}_{i+1} + h\gamma^*_{\nu}\nabla^{\nu}\mathbf{f}_i\,.$$

Dá se ukázat, že koeficienty γ^*_j lze počítat pomocí rekurze

$$\gamma^*_0 = 1, \quad \gamma^*_j + \frac{1}{2}\gamma^*_{j-1} + \frac{1}{3}\gamma^*_{j-2} + \cdots + \frac{1}{j+1}\gamma_0 = 1\,, \quad j = 1,2,\ldots.$$

Z ní snadno spočteme několik prvních koeficientů

j	0	1	2	3	4	5	6
γ^*_j	1	$\frac{1}{2}$	$\frac{5}{12}$	$\frac{3}{8}$	$\frac{251}{720}$	$\frac{95}{288}$	$\frac{19\,087}{60\,480}$

Adamsova-Bashforthova formule (4.53) resp. (4.54) je normalizovaná. Skutečně, charakteristický polynom $\varrho(\Theta) = \Theta^\nu - \Theta^{\nu-1}$, takže $\varrho'(1) = 1$.

Lokální diskretizační chyba. Přesnost metody měříme pomocí lokální diskretizační chyby $\mathbf{lte}_i^*$ definované rovností

$$\mathbf{y}(x_{i+1}) - \mathbf{y}(x_i) = h_i \sum_{j=1}^{\nu} \beta_{\nu,j}^* \mathbf{f}(x_{i+1-j}, \mathbf{y}(x_{i+1-j})) + \mathbf{lte}_i^* .$$

Jestliže do polynomu $\mathbf{P}_{\nu-1}(x)$ dosadíme $\mathbf{f}(x_{i+1-j}, \mathbf{y}(x_{i+1-j}))$ místo $\mathbf{f}_{i+1-j}$ vidíme, že

$$\mathbf{P}_{\nu-1}(x_{i+1-j}) = \mathbf{f}(x_{i+1-j}, \mathbf{y}(x_{i+1-j})) = \mathbf{y}'(x_{i+1-j}) , \quad j = 1, 2, \ldots, \nu .$$

Ze známé věty o Lagrangeově tvaru zbytku interpolačního polynomu plyne, že pro každé t a každou složku m existuje číslo $\xi_m(t) \in (x_{i+1-\nu}, x_i)$ takové, že

$$y'_{,m}(t) = P_{\nu-1,m}(t) + \frac{1}{\nu!} y_{,m}^{(\nu+1)}(\xi_m(t)) \prod_{k=1}^{\nu} (t - x_{i+1-k}) .$$

To zapíšeme stručněji ve vektorové formě

$$\mathbf{y}'(t) = \mathbf{P}_{\nu-1}(t) + \frac{1}{\nu!} \mathbf{y}^{(\nu+1)}(*) \prod_{k=1}^{\nu} (t - x_{i+1-k}) ,$$

hvězdička přitom vyjadřuje, že každá složka vektoru $\mathbf{y}^{(\nu+1)}$ je vyhodnocena pro jiný argument. Pak

$$\mathbf{y}(x_{i+1}) = \mathbf{y}(x_i) + \int_{x_i}^{x_{i+1}} \mathbf{P}_{\nu-1}(t)\,\mathrm{d}t + \int_{x_i}^{x_{i+1}} \frac{1}{\nu!} \mathbf{y}^{(\nu+1)}(*) \prod_{k=1}^{\nu} (t - x_{i+1-k})\,\mathrm{d}t .$$

Protože člen $\prod_{k=1}^{\nu}(t - x_{i+1-k})$ nemění v intervalu (x_i, x_{i+1}) znaménko, užitím první věty o střední hodnotě integrálu dostaneme

$$\mathbf{lte}_i^* = \frac{1}{\nu!} \mathbf{y}^{(\nu+1)}(*) \int_{x_i}^{x_{i+1}} \prod_{k=1}^{\nu} (t - x_{i+1-k})\,\mathrm{d}t .$$

Jestliže H je nejdelší z kroků vystupujících ve formuli, pak pro lokální diskretizační chybu platí odhad

$$\|\mathbf{lte}_i^*\| \le \|\mathbf{y}^{(\nu+1)}\| H^{\nu+1} ,$$

takže ABν metoda je řádu ν a protože $\nu \ge 1$, je ABν metoda konzistentní. Je-li délka kroku konstanta h, výraz pro lokální diskretizační chybu nabude tvaru

$$\mathbf{lte}_i^* = \mathbf{y}^{(\nu+1)}(*) h^{\nu+1} \frac{1}{\nu!} \int_0^1 \prod_{k=0}^{\nu-1} (t+k)\,\mathrm{d}t = \gamma_\nu^* \mathbf{y}^{(\nu+1)}(*) h^{\nu+1} .$$

Protože $\mathbf{y}^{(\nu+1)}(*) = \mathbf{y}^{(\nu+1)}(x_i) + O(h)$, je $\gamma_\nu^* \mathbf{y}^{(\nu+1)}(x_i) h^{\nu+1}$ hlavní člen lokální diskretizační chyby a γ_ν^* je chybová konstanta. Lokální chybu lze snadno odhadnout. Přidáme-li ještě uzel $x_{i-\nu}$ a sestrojíme polynom $\mathbf{P}_\nu(x)$, dostaneme chybu

$$\mathbf{y}(x_{i+1}) = \mathbf{y}(x_i) + \int_{x_i}^{x_{i+1}} \mathbf{P}_\nu(t)\,\mathrm{d}t + O(H^{\nu+2}) .$$

Odečteme-li tuto rovnici od rovnice $\mathbf{y}(x_{i+1}) = \mathbf{y}(x_i) + \int_{x_i}^{x_{i+1}} \mathbf{P}_{\nu-1}(t)\,\mathrm{d}t + \mathbf{lte}_i^*$, dostaneme

$$\mathbf{lte}_i^* = \int_{x_i}^{x_{i+1}} \mathbf{P}_\nu(t)\,\mathrm{d}t - \int_{x_i}^{x_{i+1}} \mathbf{P}_{\nu-1}(t)\,\mathrm{d}t + O(H^{\nu+2}) .$$

Tyto výrazy se spočtou snadno, když délka kroku je konstanta h a interpolační polynom je vyjádřen pomocí zpětných diferencí. V tom případě

$$\mathbf{lte}_i^* = h\gamma_\nu^* \nabla^\nu \mathbf{f}(x_i, \mathbf{y}(x_i)) + O(h^{\nu+2}) .$$

Proto je přirozené odhadnout lokální diskretizační chybu výrazem $h\gamma_\nu^* \nabla^\nu \mathbf{f}_i$.

Další vlastnosti ABν metod. Interpolační polynom vyjádřený pomocí zpětných diferencí je zřejmě neobyčejně vhodný jak pro sledování lokální chyby, a následně k řízení délky kroku, tak ke změně řádu ν metody. Tyto přednosti se neomezují jen na konstantní délku kroku. Na obecné síti lze použít formálně stejný Newtonův interpolační polynom, rozdíl spočívá jen v tom, že zpětné diference se nahradí zpětnými poměrnými diferencemi, viz např. [35]. Integrační koeficienty, ekvivalenty γ_j^*, závisejí nyní na vzájemné poloze uzlů $x_i, x_{i-1}, \ldots$, a proto se musejí v každém kroku počítat znovu. Výpočet na nerovnoměrné síti je proto v detailech jiný, podstata však zůstává stejná. Příslušné algoritmy jsou důkladně propracovány a úspěšně implementovány.

Adamsova-Bashforthova metoda je D-stabilní: charakteristický polynom ABν metody $\varrho(\Theta) = \Theta^\nu - \Theta^{\nu-1}$ má $(\nu-1)$-násobný kořen 0 a jednoduchý kořen 1. ABν metoda je tedy konvergentní.

Adamsova-Bashfortova metoda je explicitní: do pravé strany formule (4.53) dosadíme známé hodnoty a přibližné řešení $\mathbf{y}_{i+1}$ je určeno. Jednu věc jsme však zatím zamlčeli, totiž jak určit *startovací hodnoty* $\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{\nu-1}$. Z počáteční podmínky vždy máme $\mathbf{y}_0 = \boldsymbol{\eta}$. Zbývající startovací hodnoty můžeme určit třeba tak, že je spočteme dostatečně přesnou Rungovovou-Kuttovou metodou, teoretické výsledky říkají, že by měla být řádu alespoň $O(h^\nu)$. To je však spíše klasický přístup, který měl oprávnění v dobách, kdy programy pracovaly s pevnou délkou kroku. Jestliže však délku kroku řídíme tak, aby lokální chyba nepřesáhla zadanou toleranci, můžeme dostatečně přesné startovací hodnoty získat i pomocí méně přesné metody, pokud délky kroků volíme dostatečně malé. Tomuto přístupu se v moderních programech dává přednost a používají se tzv. *samostartující algoritmy*: startovací hodnoty pro ν-krokovou Adamsovu-Bashforthovu metodu se spočtou opět pomocí Adamsových-Bashforthových metod, $\mathbf{y}_1$ pomocí AB1, $\mathbf{y}_2$ pomocí AB2 až nakonec $\mathbf{y}_{\nu-1}$ pomocí AB$(\nu-1)$.

Adamsovy-Bashforthovy metody se samostatně téměř nepoužívají. To však neznamená, že by se nepoužívaly vůbec, nejvýznamnější uplatnění nacházejí, spolu s Adamsovými-Moultonovými metodami, v algoritmu známém jako prediktor-korektor. Kvalitní

implementace Adamsových metod technikou prediktor-korektor přitom patří v současnosti k základnímu programovému vybavení pro řešení ODR. V následujícím textu si proto nejdříve zavedeme třídu Adamsových-Moultonových metod a pak si vysvětlíme, jak pracuje algoritmus prediktor-korektor.

4.4.2.2. *Adamsovy-Moultonovy metody*

jsou metody implicitní. Odvodí se podobně jako Adamsovy-Bashforhovy metody, rozdíl je jen v tom, že interpolační polynom $\mathbf{P}_{\nu-1}(x)$ stupně $\nu-1$ je nyní určen podmínkami

$$\mathbf{P}_{\nu-1}(x_{i+1-j}) = \mathbf{f}_{i+1-j}\,, \quad j = 0, 1, \ldots, \nu-1\,.$$

Dosazením Lagrangeova tvaru polynomu $\mathbf{P}_{\nu-1}$ do (4.52) dostaneme Adamsovu-Moultonovu metodu, stručně AMν metodu

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h_i\beta_{\nu,0}\,\mathbf{f}(x_{i+1}, \mathbf{y}_{i+1}) + h_i \sum_{j=1}^{\nu-1} \beta_{\nu,j}\,\mathbf{f}_{i+1-j}\,. \tag{4.55}$$

Člen $h_i\beta_{\nu,0}\,\mathbf{f}(x_{i+1}, \mathbf{y}_{i+1})$ jsme úmyslně vypsali zvlášť mimo sumu, neboť právě tento člen „je zodpovědný“ za to, že metoda je implicitní: počítaná hodnota $\mathbf{y}_{i+1}$ je nejen vlevo, ale také vpravo jako druhý argument funkce $\mathbf{f}$.

Pro $\nu = 1$ dostáváme implicitní Eulerovu metodu a pro $\nu = 2$ lichoběžníkovou metodu. Obě tyto metody jsou jednokrokové, vícekrokovou metodu dostaneme až pro $\nu \geq 3$, AM3 je už dvoukroková. Obecně, AMν metoda je $\max(1, \nu-1)$-kroková.

Formule pro konstantní délku kroku. V následující tabulce uvádíme koeficienty AMν metody pro $\nu = 1, 2, \ldots, 6$ a pro konstantní délku kroku:

ν	$\beta_{\nu,0}$	$\beta_{\nu,1}$	$\beta_{\nu,2}$	$\beta_{\nu,3}$	$\beta_{\nu,4}$	$\beta_{\nu,5}$
1	1					
2	$\frac{1}{2}$	$\frac{1}{2}$				
3	$\frac{5}{12}$	$\frac{8}{12}$	$-\frac{1}{12}$			
4	$\frac{9}{24}$	$\frac{19}{24}$	$-\frac{5}{24}$	$\frac{1}{24}$		
5	$\frac{251}{720}$	$\frac{646}{720}$	$-\frac{264}{720}$	$\frac{106}{720}$	$-\frac{19}{720}$	
6	$\frac{475}{1\,440}$	$\frac{1\,427}{1\,440}$	$-\frac{798}{1\,440}$	$\frac{482}{1\,440}$	$-\frac{173}{1\,440}$	$\frac{27}{1\,440}$

Jestliže interpolační polynom $\mathbf{P}_{\nu-1}$ vyjádříme v Newtonově tvaru

$$\mathbf{P}_{\nu-1}(x) = \mathbf{f}_{i+1} + \frac{x - x_{i+1}}{h}\,\nabla\mathbf{f}_{i+1} + \cdots + \frac{(x - x_{i+1})(x - x_i)\ldots(x - x_{i+3-\nu})}{h^{\nu-1}(\nu-1)!}\,\nabla^{\nu-1}\mathbf{f}_{i+1}$$

a dosadíme do (4.52), dostaneme AMν formuli ve tvaru vhodném pro odhad chyby a změnu řádu:

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h\sum_{j=1}^{\nu} \gamma_{j-1}\nabla^{j-1}\mathbf{f}_{i+1}\,, \tag{4.56}$$

kde $\gamma_0 = 1$ a pro $j \geq 1$

$$\gamma_j = \int_{x_i}^{x_{i+1}} \frac{(t - x_{i+1})\ldots(t - x_{i+2-j})}{h^j j!}\,\mathrm{d}t = \frac{1}{j!}\int_0^1 (s-1)s\ldots(s+j-2)\,\mathrm{d}s\,.$$

Koeficienty γ_j lze opět počítat pomocí rekurze

$$\gamma_0 = 1, \quad \gamma_j + \frac{1}{2}\gamma_{j-1} + \frac{1}{3}\gamma_{j-2} + \cdots + \frac{1}{j+1}\gamma_0 = 0\,, \quad j = 1, 2, \ldots,$$

několik prvních koeficientů uvádí následující tabulka

j	0	1	2	3	4	5	6
γ_j	1	$-\frac{1}{2}$	$-\frac{1}{12}$	$-\frac{1}{24}$	$-\frac{19}{720}$	$-\frac{3}{160}$	$-\frac{863}{60\,480}$

Z definice koeficientů γ_j a γ_j^* odvodíme, že pro jejich vzájemný vztah platí

$$\gamma_j = \gamma_j^* - \gamma_{j-1}^* \quad \text{pro} \quad j \geq 1\,. \tag{4.57}$$

Lokální diskretizační chyba $\mathbf{lte}_i$ Adamsovy-Moultonovy metody se definuje předpisem

$$\mathbf{y}(x_{i+1}) - \mathbf{y}(x_i) = h_i \sum_{j=0}^{\nu-1} \beta_{\nu,j}\,\mathbf{f}(x_{i+1-j}, \mathbf{y}(x_{i+1-j})) + \mathbf{lte}_i$$

a podobně jako dříve pro Adamsovu-Bashforthovu metodu odvodíme

$$\mathbf{lte}_i = \frac{1}{\nu!}\,\mathbf{y}^{(\nu+1)}(*)\int_{x_i}^{x_{i+1}} \prod_{k=0}^{\nu-1}(t - x_{i+1-k})\,\mathrm{d}t\,.$$

Jestliže H je nejdelší z kroků vystupujících ve formuli, pak pro lokální diskretizační chybu platí odhad

$$\|\mathbf{lte}_i\| \leq \|\mathbf{y}^{(\nu+1)}\|H^{\nu+1}\,,$$

takže AMν metoda je řádu ν. Je-li délka kroku konstanta h,

$$\mathbf{lte}_i = \mathbf{y}^{(\nu+1)}(*)h^{\nu+1}\frac{1}{\nu!}\int_0^1 \prod_{k=0}^{\nu-1}(t + k - 1)\,\mathrm{d}t = \gamma_\nu\,\mathbf{y}^{(\nu+1)}(*)h^{\nu+1}\,,$$

takže $\gamma_\nu\,\mathbf{y}^{(\nu+1)}(x_i)h^{\nu+1}$ je hlavní člen lokální diskretizační chyby a γ_ν je chybová konstanta. Pomocí zpětných diferencí lze, podobně jako jsme to provedli pro Adamsovu-Bashforthovu metodu, odvodit

$$\mathbf{lte}_i = h\gamma_\nu\,\nabla^\nu\mathbf{f}(x_{i+1}, \mathbf{y}(x_{i+1})) + O(h^{\nu+2})\,,$$

takže lokální diskretizační chybu můžeme aproximovat výrazem $\mathbf{est}_i = h\gamma_\nu\,\nabla^\nu\mathbf{f}_{i+1}$.

Snadno se ověří, že AMν metoda je normalizovaná, D-stabilní, konzistentní a tedy konvergentní.

4.4.2.3. *Metody prediktor-korektor*

Adamsova-Bashforthova a Adamsova-Moultonova formule je pro zvolené ν téhož řádu ν. Pro $\nu = 1$ je $\gamma_1 = -\gamma_1^*$ a tedy implicitní Eulerova metoda IE $\equiv$ AM1 je přibližně stejně přesná jako explicitní Eulerova metoda EE $\equiv$ AB1. Pro $\nu > 1$ je však $|\gamma_\nu|$ značně menší než $|\gamma_\nu^*|$ a proto je Adamsova-Moultonova metoda výrazně přesnější než Adamsova-Bashforthova metoda téhož řádu $\nu > 1$.

Abychom určili $\mathbf{y}_{i+1}$, musíme řešit rovnici (4.55), tj. rovnici

$$\mathbf{y}_{i+1} = \boldsymbol{\varphi}(\mathbf{y}_{i+1}), \quad \text{kde} \quad \boldsymbol{\varphi}(\mathbf{z}) = \mathbf{y}_i + h\beta_{\nu,0}\,\mathbf{f}(x_{i+1}, \mathbf{z}) + h\sum_{j=1}^{\nu-1} \beta_{\nu,j}\,\mathbf{f}_{i+1-j}\,.$$

Použít můžeme metodu prosté iterace: zvolíme počáteční aproximaci $\mathbf{y}_{i+1}^{(0)}$ a postupně počítáme $\mathbf{y}_{i+1}^{(s)} = \boldsymbol{\varphi}(\mathbf{y}_{i+1}^{(s-1)})$, $s = 1, 2, \ldots$. Pro dostatečně malé h metoda konverguje, jak jsme ukázali v odstavci 4.4.1. Jestliže počáteční aproximaci $\mathbf{y}_{i+1}^{(0)}$ určíme pomocí Adamsovy-Bashforthovy metody, provedeme jen několik málo iterací

$$\mathbf{y}_{i+1}^{(s)} = \boldsymbol{\varphi}(\mathbf{y}_{i+1}^{(s-1)}), \quad s = 1, 2, \ldots, S,$$

a nakonec položíme

$$\mathbf{y}_{i+1} = \mathbf{y}_{i+1}^{(S)}, \quad \mathbf{f}_{i+1} = \mathbf{f}(x_{i+1}, \mathbf{y}_{i+1}),$$

dostaneme *metodu prediktor-korektor*, kterou schématicky označujeme P(EC)^{S}E. Přitom P značí předpověď počáteční aproximace explicitní Adamsovou-Bashforthovou metodou, C korekci implicitní Adamsovou-Moultonovou metodou a E vyhodnocení pravé strany $\mathbf{f}(x_{i+1}, \mathbf{y}_{i+1}^{(s)})$. Někdy se klade $\mathbf{f}_{i+1} = \mathbf{f}(x_{i+1}, \mathbf{y}_{i+1}^{(S-1)})$, takové schéma se označuje jako P(EC)^S. Nejčastěji se používá schéma PECE, kdy se korekce provede jen jednou a pravá strana se počítá dvakrát. Konkrétně pro dvojici AB2-AM2 a PECE schéma organizujeme výpočet následovně:

P	AB2	$\mathbf{y}_{i+1}^* = \mathbf{y}_i + \frac{1}{2}h(3\mathbf{f}_i - \mathbf{f}_{i-1})$
E		$\mathbf{f}_{i+1}^* = \mathbf{f}(x_{i+1}, \mathbf{y}_{i+1}^*)$
C	AM2	$\mathbf{y}_{i+1} = \mathbf{y}_i + \frac{1}{2}h(\mathbf{f}_{i+1}^* + \mathbf{f}_i)$
E		$\mathbf{f}_{i+1} = \mathbf{f}(x_{i+1}, \mathbf{y}_{i+1})$

Efektivní tvar metody prediktor-korektor. Zabývejme se dále obecným případem, kdy prediktor je ABν metoda

$$\mathbf{y}_{i+1}^* = \mathbf{y}_i + h\sum_{j=1}^{\nu} \gamma_{j-1}^* \nabla^{j-1}\mathbf{f}_i \tag{4.58}$$

a korektor je AMν metoda

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h\sum_{j=1}^{\nu} \gamma_{j-1} \nabla^{j-1}\mathbf{f}_{i+1}\,.$$

Diference v Adamsově-Moultonově metodě obsahují členy $\mathbf{f}_{i+1} = \mathbf{f}(x_{i+1}, \mathbf{y}_{i+1})$, které jsou v PECE schématu nahrazeny členy $\mathbf{f}_{i+1}^* = \mathbf{f}(x_{i+1}, \mathbf{y}_{i+1}^*)$, takže

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h\sum_{j=1}^{\nu} \gamma_{j-1} \nabla^{j-1}\mathbf{f}_{i+1}^*\,.$$

Diference $\nabla^{j-1}\mathbf{f}_{i+1}^*$ se snadno spočtou pomocí diferencí $\nabla^{j-1}\mathbf{f}_i$, které máme uloženy z předchozího kroku:

$$\begin{aligned}
\nabla^0\mathbf{f}_{i+1}^* &= \mathbf{f}_{i+1}^*\,,\\
\nabla^1\mathbf{f}_{i+1}^* &= \mathbf{f}_{i+1}^* - \mathbf{f}_i = \nabla^0\mathbf{f}_{i+1}^* - \nabla^0\mathbf{f}_i\,,\\
&\vdots\\
\nabla^{j+1}\mathbf{f}_{i+1}^* &= \nabla^j\mathbf{f}_{i+1}^* - \nabla^j\mathbf{f}_i\,.
\end{aligned}$$

Tyto diference se použijí jen pro výpočet $\mathbf{y}_{i+1}$ a k odhadu lokální chyby, takže se nemusejí schovávat. Až spočteme $\mathbf{y}_{i+1}$ a $\mathbf{f}_{i+1}$, vytvoříme diference

$$\begin{aligned}
\nabla^0\mathbf{f}_{i+1} &= \mathbf{f}_{i+1}\,,\\
\nabla^1\mathbf{f}_{i+1} &= \mathbf{f}_{i+1} - \mathbf{f}_i = \nabla^0\mathbf{f}_{i+1} - \nabla^0\mathbf{f}_i\,,\\
&\vdots\\
\nabla^{j+1}\mathbf{f}_{i+1} &= \nabla^j\mathbf{f}_{i+1} - \nabla^j\mathbf{f}_i\,.
\end{aligned}$$

Nové diference $\nabla^j\mathbf{f}_{i+1}$ mohou být uloženy na místo starých diferencí $\nabla^j\mathbf{f}_i$, čímž se ušetří paměťové místo.

Pokud použijeme dvojici ABν-AMν, dostaneme pro určení $\mathbf{y}_{i+1}$ sympatický vztah

$$\mathbf{y}_{i+1} = \mathbf{y}_{i+1}^* + h\gamma_{\nu-1}^*\nabla^\nu\mathbf{f}_{i+1}^*\,. \tag{4.59}$$

Abychom to dokázali, odečteme formuli prediktoru od formule korektoru a obdržíme

$$\mathbf{y}_{i+1} - \mathbf{y}_{i+1}^* = h\sum_{j=1}^{\nu} \gamma_{j-1}\nabla^{j-1}\mathbf{f}_{i+1}^* - h\sum_{j=1}^{\nu} \gamma_{j-1}^*\nabla^{j-1}\mathbf{f}_i\,.$$

Prvé členy z obou sum sloučíme do tvaru

$$h\gamma_0\nabla^0\mathbf{f}_{i+1}^* - h\gamma_0^*\nabla^0\mathbf{f}_i = h\gamma_0^*\nabla^1\mathbf{f}_{i+1}^*\,,$$

přičemž jsme využili toho, že $\gamma_0 = \gamma_0^*$. Když k tomu přidáme druhé členy z každé sumy, dostaneme

$$\begin{aligned}
&h\gamma_0^*\nabla^1\mathbf{f}_{i+1}^* + h\gamma_1\nabla^1\mathbf{f}_{i+1}^* - h\gamma_1^*\nabla^1\mathbf{f}_i = h(\gamma_0^* + \gamma_1)\nabla^1\mathbf{f}_{i+1}^* - h\gamma_1^*\nabla^1\mathbf{f}_i =\\
&h\gamma_1^*(\nabla^1\mathbf{f}_{i+1}^* - \nabla^1\mathbf{f}_i) = h\gamma_1^*\nabla^2\mathbf{f}_{i+1}^*\,.
\end{aligned}$$

Zde jsme použili rovnost (4.57), tj. $\gamma_j^* = \gamma_{j-1}^* + \gamma_j$. Opakováním tohoto postupu dostaneme nakonec formuli (4.59).

Adamsova-Moultonova metoda řádu ν nepoužívá hodnotu $\mathbf{f}_{i+1-\nu}$, kterou Adamsova-Bashforthova metoda řádu ν používá. To nás přivádí k myšlence použít Adamsovu-Moultonovu metodu řádu $\nu+1$ jako korektor pro Adamsovy-Bashforthovu metodu řádu ν. Výraz pro $\mathbf{y}_{i+1}$ dostaneme, když k pravé straně rovnosti (4.59) přičteme člen $h\gamma_\nu \nabla^\nu \mathbf{f}_{i+1}^*$. Užitím vztahu $\gamma_\nu^* = \gamma_{\nu-1}^* + \gamma_\nu$ dostaneme

$$\mathbf{y}_{i+1} = \mathbf{y}_{i+1}^* + h\gamma_\nu^* \nabla^\nu \mathbf{f}_{i+1}^* \,. \tag{4.60}$$

Ze srovnání vzorců (4.60) a (4.59) je patrné, že použití korektoru vyššího řádu nás nic nestojí a proto je výpočet $\mathbf{y}_{i+1}$ podle vzorce (4.60) dosti populární.

Lokální diskretizační chyba. V dalším prozkoumáme lokální diskretizační chybu metody prediktor-korektor pro dvojici ABp*-AMp a schéma PECE. Připomeňme si, že

$$\mathbf{lte}_i^p = \mathbf{y}(x_{i+1}) - \mathbf{y}(x_i) - h\sum_{j=1}^{p^*} \beta_{p^*,j}^* \mathbf{f}(x_{i+1-j}, \mathbf{y}(x_{i+1-j}))$$

je lokální diskretizační chyba prediktoru ABp* a

$$\mathbf{lte}_i^c = \mathbf{y}(x_{i+1}) - \mathbf{y}(x_i) - h\beta_{p,0}\mathbf{f}(x_{i+1}, \mathbf{y}(x_{i+1})) - h\sum_{j=1}^{p-1} \beta_{p,j} \mathbf{f}(x_{i+1-j}, \mathbf{y}(x_{i+1-j}))$$

je lokální diskretizační chyba korektoru ABp. Lokální diskretizační chyba $\mathbf{lte}_i$ prediktoru-korektoru je proto určena rovností

$$\begin{aligned}\mathbf{lte}_i &= \mathbf{y}(x_{i+1}) - \mathbf{y}(x_i) - h\beta_{p,0}\mathbf{f}(x_{i+1}, \mathbf{y}(x_{i+1}) - \mathbf{lte}_i^p) - h\sum_{j=1}^{p-1} \beta_{p,j} \mathbf{f}(x_{i+1-j}, \mathbf{y}(x_{i+1-j})) = \\ &= \mathbf{lte}_i^c + h\beta_{p,0}[\mathbf{f}(x_{i+1}, \mathbf{y}(x_{i+1})) - \mathbf{f}(x_{i+1}, \mathbf{y}(x_{i+1}) - \mathbf{lte}_i^p)] \,.\end{aligned}$$

Odtud užitím věty o střední hodnotě dostaneme

$$\mathbf{lte}_i = \mathbf{lte}_i^c + h\beta_{p,0}\frac{\partial \mathbf{f}(x_{i+1}, *)}{\partial \mathbf{y}} \mathbf{lte}_i^p \,, \tag{4.61}$$

přičemž hvězdička vyjadřuje, že prvky Jacobiovy matice jsou vyhodnoceny pro různé hodnoty argumentů.

Protože prediktor je řádu p^* a korektor řádu p, je $\mathbf{lte}_i^p = O(h^{p^*+1})$ a $\mathbf{lte}_i^c = O(h^{p+1})$, takže $\mathbf{lte}_i = O(h^{q+1})$, kde $q = \min(p, p^*+1)$, tj. formule prediktor-korektor je řádu q. To je důvod, proč se nejčastěji volí prediktor i korektor stejného řádu $p^* = p$: pak je totiž řád q páru prediktor-korektor roven řádu p korektoru a navíc, hlavní člen lokální diskretizační chyby páru je stejný jako hlavní člen lokální diskretizační chyby korektoru. To je skvělé, stačí jediná korekce a máme přesnost korektoru! Je-li řád prediktoru o jedničku nižší než řád korektoru,tj. $p^* = p - 1$, je řád páru stále roven řádu korektoru, hlavní člen lokální diskretizační chyby páru je však již jiný než hlavní člen lokální diskretizační chyby korektoru.

Všechny úvahy jsme prováděli pro PECE schéma, které se v praxi zdaleka nejvíc používá. Není těžké zjistit, že každá korekce redukuje vliv prediktoru o jeden řád. Tak například pro prediktor řádu $p^* = p - 1$ a dvě korekce, tj. výpočet podle schématu $\mathrm{P(EC)^2E}$, je hlavní člen lokální diskretizační chyby páru opět stejný jako hlavní člen lokální diskretizační chyby korektoru.

Odhad lokální diskretizační chyby. Vyjádření lokální diskretizační chyby $\mathbf{lte}_i$ ve tvaru (4.61) má význam teoretický, pro praktický výpočet však potřebujeme odhad $\mathbf{est}_i$ lokální diskretizační chyby, který se dá snadno spočítat. Ukažme si proto, jak se takový odhad dá zkonstruovat.

Pro odvození odhadu $\mathbf{est}_i$ je účelné přijmout *lokalizační předpoklad*, totiž že dříve spočtené hodnoty $\mathbf{y}_i, \ldots, \mathbf{y}_{i+1-\nu}$ jsou přesné, tj. $\mathbf{y}_i = \mathbf{y}(x_i), \ldots, \mathbf{y}_{i+1-\nu} = \mathbf{y}(x_{i+1-\nu})$. Tento předpoklad je (pro $\nu > 1$) zcela nerealistický, umožní nám však uhodnout, jak odhad $\mathbf{est}_i$ může vypadat. Dodatečně se pak ukáže, že jde o odhad korektní, plně použitelný pro praktické výpočty.

Nechť $\widetilde{\mathbf{y}}_{i+1}$ je řešení získané formulí řádu $\nu+1$ a $\mathbf{y}_{i+1}$ je řešení získané formulí řádu ν, pak lokální diskretizační chybu $\mathbf{y}(x_{i+1}) - \mathbf{y}_{i+1}$ můžeme přibližně odhadnout pomocí

$$\widetilde{\mathbf{y}}_{i+1} - \mathbf{y}_{i+1} = (\mathbf{y}(x_{i+1}) - \mathbf{y}_{i+1}) - (\mathbf{y}(x_{i+1}) - \widetilde{\mathbf{y}}_{i+1}) = \mathbf{y}(x_{i+1}) - \mathbf{y}_{i+1} + O(h^{\nu+2}) \,,$$

neboť lokální diskretizační chyba $\mathbf{y}(x_{i+1}) - \widetilde{\mathbf{y}}_{i+1}$ je $O(h^{\nu+2})$. Přesnější řešení $\widetilde{\mathbf{y}}_{i+1}$ můžeme spočítat pomocí Adamsova - Moultonova korektoru řádu $\nu+1$ podle formule (4.60), tj.

$$\widetilde{\mathbf{y}}_{i+1} = \mathbf{y}_{i+1}^* + h\gamma_\nu^* \nabla^\nu \mathbf{f}_{i+1}^* \,.$$

Jestliže $\mathbf{y}_{i+1}$ spočteme pomocí ABν-AMν páru metodou PECE, dostaneme pomocí (4.59) a (4.57)

$$\widetilde{\mathbf{y}}_{i+1} - \mathbf{y}_{i+1} = h\gamma_\nu^* \nabla^\nu \mathbf{f}_{i+1}^* - h\gamma_{\nu-1}^* \nabla^\nu \mathbf{f}_{i+1}^* = h\gamma_\nu \nabla^\nu \mathbf{f}_{i+1}^* \,.$$

Odhad $\mathbf{est}_i$ lokální diskretizační chyby proto volíme jako

$$\mathbf{est}_i = h\gamma_\nu \nabla^\nu \mathbf{f}_{i+1}^* \,. \tag{4.62}$$

V [34] je dokázáno, že tento odhad je správný i bez lokalizačního předpokladu.

Jestliže jako přibližné řešení v x_{i+1} vezmeme přesnější hodnotu $\widetilde{\mathbf{y}}_{i+1} = \mathbf{y}_{i+1} + \mathbf{est}_i$ říkáme, že jsme použili metodu prediktor-korektor s *lokální extrapolací*. Tedy výpočet podle (4.58), (4.59) je bez lokální extrapolace a výpočet podle (4.58), (4.60) je s lokální extrapolací.

Řízení délky kroku a řádu metody. Kvalitní programy založené na metodách prediktor-korektor mění jak délku kroku tak řád metody. Tak například program ODE113 v MATLABu používá ABν-AMν prediktor-korektor typu PECE s lokální extrapolací pro $\nu = 1, 2, \ldots, 12$.

Změna řádu se provádí současně se změnou délky kroku. Algoritmy tohoto typu se se označují jako VSVO algoritmy (variable step variable order). Základní myšlenka je jednoduchá. Předpokládejme, že jsme spočetli $\mathbf{y}_{i+1}$ metodou řádu ν s lokální extrapolací a krokem délky h. Určíme odhad $\mathbf{E}_\nu := h\gamma_\nu \nabla^\nu \mathbf{f}_{i+1}^*$ lokální diskretizační chyby. Jestliže

$\|\mathbf{E}_\nu\| > \varepsilon$, jde o neúspěch a výpočet $\mathbf{y}_{i+1}$ je třeba opakovat, v opačném případě pokračujeme výpočtem přibližného řešení $\mathbf{y}_{i+2}$. V každém případě je však třeba stanovit novou délku kroku a nový řád. Pokud jde o řád, připustíme jen řády $\nu-1$, ν nebo $\nu+1$. Dá se ukázat, že chybu každé z těchto metod lze odhadnout pomocí $\mathbf{E}_{\nu-1}$, $\mathbf{E}_\nu$ nebo $\mathbf{E}_{\nu+1}$. Novou délku kroku stanovíme podobně jako pro jednokrokovou metodu. Podle kriteria XEPS dostaneme pro každou ze tří kandidujících metod

$$h_k^* = \Theta h(\varepsilon/\|\mathbf{E}_k\|)^{1/(k+1)} \quad \text{pro } k=\nu-1,\ \nu \text{ a } \nu+1\,,$$

a největší z čísel $h_{\nu-1}^*$, h_ν^*, $h_{\nu+1}^*$ určí jak novou délku kroku tak nový řád. Tedy, je-li největší h_ν^*, řád zůstává a jako novou délku kroku vezmeme $h^*=h_\nu^*$, je-li největší $h_{\nu+1}^*$, zvětšíme řád o jedničku a pokračujeme s krokem délky $h^*=h_{\nu+1}^*$ a je-li největší $h_{\nu-1}^*$, řád o jedničku snížíme a pokračujeme s krokem délky $h^*=h_{\nu-1}^*$.

Znovu upozorňujeme, že jsme zde vyložili jen základní ideu algoritmu VSVO, detaily je třeba hledat v dokumentaci úspěšných implementací, odkazy na ně lze najít např. v [17], [34].

Start metody není žádný problém: začneme metodou řádu 1, počáteční délku kroku určíme např. tak, jak je to popsáno v odstavci 4.3.2.3, a algoritmus VSVO se už sám rychle vyladí na správné hodnoty jak délky kroku tak řádu metody.

Absolutní stabilita. Doposud jsme se nezmínili o absolutní stabilitě Adamsových formulí. A-stabilní jsou jen první dvě Adamsovy-Moultonovy metody, totiž AM1, což je implicitní Eulerova metoda, a AM2, což je metoda lichoběžníková. Adamsovy-Moultonovy metody mají výrazně větší oblast absolutní stability než Adamsovy-Bashforthovy metody téhož řádu. S rostoucím řádem se oblasti absolutní stability zmenšují. Pro ilustraci uvádíme v následující tabulce dolní meze α intervalů absolutní stability $(\alpha,0)$ pro ABν a AMν, $\nu=1,2,3,4$:

ν	1	2	3	4
ABν	-2	-1	$-\frac{6}{11}$	$-\frac{3}{10}$
AMν	$-\infty$	$-\infty$	-6	-3

Pro metody prediktor-korektor typu P(EC)$^S E$ oblast absolutní stability roste, když zvětšujeme počet S iterací: pro $S=0$ je stabilita nejmenší, odpovídá stabilitě ABν metody, a pro $S\to\infty$ se stabilita blíží svému maximu, tj. stabilitě AMν metody. Tak například interval absolutní stability pro AB4-AM4 PECE metodu je $(-1,25;0)$, tedy větší než pro AB4 metodu, avšak menší než pro AM4 metodu. Interval absolutní stability AB4-AM4 PECE metody je více než čtyřikrát větší než interval absolutní stability AB4 metody, proto si (za jistých okolností) můžeme dovolit až čtyřikrát delší krok, a vzhledem k tomu, že PECE metoda v každém kroku vyhodnocuje pravou stranu dvakrát, můžeme oproti AB4 metodě docílit až 50 % úspory v počtu vyhodnocení pravých stran.

Oblasti absolutní stability pro metody ABν-AMν PECE metody pro $\nu\le 4$ jsou uvedeny např. v [17]. Obecně lze konstatovat, že oblasti absolutní stability ABν-AMν PECE metod nejsou příliš veliké a s rostoucím řádem ν se dále zmenšují. Proto se programy založené ABν-AMν PECE metodách, jako třeba program ODE113 v MATLABu, nehodí

pro řešení úloh se silným tlumením (stiff problémů), viz odstavec 4.5. Existuje však celá řada jiných úloh, které nejsou stiff, a pro ně jsou Adamsovy metody vhodné.

4.4.3. Metody zpětného derivování

Při řešení tuhých problémů je třeba pracovat s metodami, které se vyznačují velkou oblastí absolutní stability. Metody zpětného derivování (stručně BDF podle anglického backward differentiation formulas) tuto vlastnost mají. Pro označení ν-krokové metody zpětného derivování použijeme zkrácený zápis BDFν metoda. BDFν metodu odvodíme tak, že v rovnici

$$\mathbf{y}'(x_{i+1}) = \mathbf{f}(x_{i+1}, \mathbf{y}(x_{i+1}))$$

nahradíme derivaci $\mathbf{y}'(x_{i+1})$ pomocí derivace $\mathbf{P}'_\nu(x_{i+1})$ interpolačního polynomu $\mathbf{P}_\nu(x)$ stupně ν procházejícího body $[x_{i+1},\mathbf{y}_{i+1}]$, $[x_i,\mathbf{y}_i]$, $\ldots$, $[x_{i+1-\nu},\mathbf{y}_{i+1-\nu}]$. Tak dostaneme metodu

$$\mathbf{P}'_\nu(x_{i+1}) = \mathbf{f}(x_{i+1},\mathbf{y}_{i+1})\,.$$

Vyjádříme-li interpolační polynom v Lagrangeově tvaru

$$\mathbf{P}_\nu(x) = \sum_{j=0}^{\nu} \mathbf{y}_{i+1-j}\, l_j(x)\,, \quad \text{kde} \quad l_j(x) = \prod_{\substack{k=1\\k\neq j}}^{\nu} \frac{x-x_{i+1-k}}{x_{i+1-j}-x_{i+1-k}}\,,\ j=1,2,\ldots,\nu\,,$$

dostaneme BDFν metodu jako předpis

$$\alpha_{\nu,0}\mathbf{y}_{i+1}+\alpha_{\nu,1}\mathbf{y}_i+\cdots+\alpha_{\nu,\nu}\mathbf{y}_{i+1-\nu} = h_i\mathbf{f}(x_{i+1},\mathbf{y}_{i+1})\,, \quad \text{kde} \quad \alpha_{\nu,j}=h_i l'_j(x_{i+1})\,.$$

Metoda BDFν je implicitní ν-kroková metoda. Speciálně BDF1 je jednokroková implicitní Eulerova metoda, jak se snadno ověří.

Formule pro konstantní délku kroku. V následující tabulce uvádíme koeficienty BDFν metody pro ν=1,2,...,6 a pro konstantní délku kroku:

ν	$\alpha_{\nu,0}$	$\alpha_{\nu,1}$	$\alpha_{\nu,2}$	$\alpha_{\nu,3}$	$\alpha_{\nu,4}$	$\alpha_{\nu,5}$	$\alpha_{\nu,6}$
1	1	-1					
2	$\frac{3}{2}$	-2	$\frac{1}{2}$				
3	$\frac{11}{6}$	-3	$\frac{3}{2}$	$-\frac{1}{3}$			
4	$\frac{25}{12}$	-4	3	$-\frac{4}{3}$	$\frac{1}{4}$		
5	$\frac{137}{60}$	-5	5	$-\frac{10}{3}$	$\frac{5}{4}$	$-\frac{1}{5}$	
6	$\frac{147}{60}$	-6	$\frac{15}{2}$	$-\frac{20}{3}$	$\frac{15}{4}$	$-\frac{6}{5}$	$\frac{1}{6}$

Interpolační polynom $\mathbf{P}_\nu(x)$ můžeme zapsat také v Newtonově tvaru

$$\mathbf{P}_\nu(x) = \mathbf{y}_{i+1}+\frac{x-x_{i+1}}{h}\nabla\mathbf{y}_{i+1}+\cdots+\frac{(x-x_{i+1})(x-x_i)\ldots(x-x_{i+2-\nu})}{h^\nu\,\nu!}\nabla^\nu\mathbf{y}_{i+1}\,.$$

Pro konstantní délku kroku

$$\frac{\mathrm{d}}{\mathrm{d}x}\left.\frac{(x-x_{i+1})(x-x_i)\ldots(x-x_{i+2-j})}{h^j\, j!}\right|_{x_{i+1}} = \frac{1}{hj},$$

takže BDFν metoda zapsána pomocí zpětných diferencí je tvaru

$$\sum_{j=1}^{\nu}\frac{1}{j}\nabla^j \mathbf{y}_{i+1} = h\,\mathbf{f}(x_{i+1},\mathbf{y}_{i+1}). \tag{4.63}$$

Tento zápis je sice formálně velmi atraktivní, pro výpočty se však příliš nehodí. $\mathbf{y}_{i+1}$ se totiž počítá iterací, tj. vyjde se z počáteční aproximace $\mathbf{y}_{i+1}^{(0)}$ a postupně se počítají aproximace $\mathbf{y}_{i+1}^{(s)}$. To ale znamená, že se v každé iteraci musejí přepočítat zpětné diference $\nabla^j \mathbf{y}_{i+1}^{(s)}$ pro všechna $j = 1, 2, \ldots, \nu$. Ukážeme si, jak se tomu můžeme vyhnout.

Efektivní tvar BDFν metody. Začneme počáteční aproximací $\mathbf{y}_{i+1}^{*} \equiv \mathbf{y}_{i+1}^{(0)}$. Určíme ji extrapolací z předchozích hodnot $\mathbf{y}_{i+1-j}$ v uzlech x_{i+1-j}, $j = 1, 2, \ldots, \nu+1$, do uzlu x_{i+1}. Newtonův tvar interpolačního polynomu je

$$\mathbf{Q}_\nu(x) = \mathbf{y}_i + \frac{x-x_i}{h}\nabla\mathbf{y}_i + \cdots + \frac{(x-x_i)(x-x_{i-1})\ldots(x-x_{i+1-\nu})}{h^\nu\,\nu!}\nabla^\nu\mathbf{y}_i,$$

takže předpovězená hodnota

$$\mathbf{y}_{i+1}^{*} = \mathbf{Q}_\nu(x_{i+1}) = \mathbf{y}_i + \frac{h}{h}\nabla\mathbf{y}_i + \ldots\frac{h(2h)\ldots(\nu h)}{h^\nu\nu!}\nabla^\nu\mathbf{y}_i = \sum_{j=0}^{\nu}\nabla^j\mathbf{y}_i.$$

To je velmi vhodná počáteční aproximace, neboť zpětné diference máme schovány z předchozího kroku. V dalším se nám bude hodit vyjádření rozdílu počítané hodnoty $\mathbf{y}_{i+1}$ a její počáteční aproximace $\mathbf{y}_{i+1}^{*}$:

$$\begin{aligned}\mathbf{y}_{i+1} - \mathbf{y}_{i+1}^{*} &= (\mathbf{y}_{i+1} - \mathbf{y}_i) - \nabla\mathbf{y}_i - \nabla^2\mathbf{y}_i - \cdots - \nabla^\nu\mathbf{y}_i = \\ &= (\nabla\mathbf{y}_{i+1} - \nabla\mathbf{y}_i) - \nabla^2\mathbf{y}_i - \cdots - \nabla^\nu\mathbf{y}_i = \\ &= (\nabla^2\mathbf{y}_{i+1} - \nabla^2\mathbf{y}_i) - \cdots - \nabla^\nu\mathbf{y}_i = \cdots = \nabla^{\nu+1}\mathbf{y}_{i+1}.\end{aligned}$$

Je účelné stanovit také počáteční aproximaci $\mathbf{f}_{i+1}^{*}$ pro hodnotu pravé strany $\mathbf{f}(x_{i+1},\mathbf{y}_{i+1})$ jako $\mathbf{Q}_\nu'(x_{i+1})$. Po jednoduché úpravě odvodíme

$$\mathbf{f}_{i+1}^{*} = \mathbf{Q}_\nu'(x_{i+1}) = \frac{1}{h}\sum_{j=1}^{\nu}\delta_j\nabla^j\mathbf{y}_i, \quad \text{kde nové konstanty} \quad \delta_j = \sum_{k=1}^{j}\frac{1}{k}.$$

Všimněte si, že $\delta_1 = 1$ a že $\delta_j - \delta_{j-1} = \frac{1}{j}$ pro $j > 1$. Užitím tohoto vztahu obdržíme

$$\begin{aligned}\mathbf{f}(x_{i+1},\mathbf{y}_{i+1}) &= \frac{1}{h}\sum_{j=1}^{\nu}\frac{1}{j}\nabla^j\mathbf{y}_{i+1} = \frac{1}{h}\nabla\mathbf{y}_{i+1} + \frac{1}{h}\sum_{j=2}^{\nu}(\delta_j-\delta_{j-1})\nabla^j\mathbf{y}_{i+1} = \\ &= \frac{1}{h}\sum_{j=1}^{\nu}\delta_j\nabla^j\mathbf{y}_{i+1} - \frac{1}{h}\sum_{j=2}^{\nu}\delta_{j-1}(\nabla^{j-1}\mathbf{y}_{i+1} - \nabla^{j-1}\mathbf{y}_i) = \\ &= \frac{1}{h}\delta_\nu\nabla^\nu\mathbf{y}_{i+1} + \frac{1}{h}\sum_{j=1}^{\nu-1}\delta_j\nabla^j\mathbf{y}_i.\end{aligned}$$

Kombinací tohoto a předchozích výsledků dostaneme

$$\mathbf{f}(x_{i+1},\mathbf{y}_{i+1}) - \mathbf{f}_{i+1}^{*} = \frac{1}{h}\delta_\nu\nabla^\nu\mathbf{y}_{i+1} - \frac{1}{h}\delta_\nu\nabla^\nu\mathbf{y}_i = \frac{1}{h}\delta_\nu\nabla^{\nu+1}\mathbf{y}_{i+1} = \frac{1}{h}\delta_\nu(\mathbf{y}_{i+1} - \mathbf{y}_{i+1}^{*}).$$

Výsledná algebraická rovnice pro určení $\mathbf{y}_{i+1}$ proto může být zapsána ve tvaru

$$\mathbf{f}(x_{i+1},\mathbf{y}_{i+1}) - \mathbf{f}_{i+1}^{*} - \frac{1}{h}\delta_\nu(\mathbf{y}_{i+1} - \mathbf{y}_{i+1}^{*}) = \mathbf{0}. \tag{4.64}$$

BDF metody se používají prakticky výhradně pro řešení tuhých problémů a jak uvidíme v odstavci 4.5.2, $\mathbf{y}_{i+1}$ se počítá řešením rovnice

$$\mathbf{g}(\mathbf{z}) := \mathbf{f}(x_{i+1},\mathbf{z}) - \mathbf{f}_{i+1}^{*} - \frac{1}{h}\delta_\nu(\mathbf{z} - \mathbf{y}_{i+1}^{*}) = \mathbf{0}$$

pomocí zjednodušené Newtonovy metody. Pro takový výpočet se rovnice $\mathbf{g}(\mathbf{z}) = \mathbf{0}$ skvěle hodí: $\mathbf{z}_0 = \mathbf{y}_{i+1}^{*}$ je dobrá počáteční aproximace, $\mathbf{y}_{i+1}^{*}$ a $\mathbf{f}_{i+1}^{*}$ se určí snadno pomocí zpětných diferencí uchovaných z předchozího kroku, v rovnici se nevyskytují zpětné diference s neznámou $\mathbf{z}$.

Lokální diskretizační chyba $\mathbf{lte}_i$ je definována vztahem

$$\mathbf{lte}_i = \sum_{j=0}^{\nu}\alpha_j\mathbf{y}(x_{i+1-j}) - h_i\mathbf{f}(x_{i+1},\mathbf{y}_{i+1}) = h_i(\mathbf{P}_\nu'(x_{i+1}) - \mathbf{y}'(x_{i+1})),$$

kde $\mathbf{P}_\nu(x)$ interpoluje $\mathbf{y}(x)$: $\mathbf{P}_\nu(x_{i+1-j}) = \mathbf{y}(x_{i+1-j})$, $j = 0, 1, \ldots, \nu$. Podle (2.4) pro derivaci Lagrangeova interpolačního polynomu $\mathbf{P}_\nu(x)$ v uzlu x_{i+1} platí

$$\mathbf{y}'(x_{i+1}) = \mathbf{P}_\nu'(x_{i+1}) + \frac{1}{(\nu+1)!}\mathbf{y}^{(\nu+1)}(*)\prod_{k=1}^{\nu}(x_{i+1} - x_{i+1-k}).$$

Jestliže H je maximální délka kroku, pak zřejmě

$$\|\mathbf{P}_\nu'(x_{i+1}) - \mathbf{y}'(x_{i+1})\| \le \frac{1}{\nu+1}\|\mathbf{y}^{(\nu+1)}\|H^\nu.$$

Dosadíme-li odtud do výrazu pro $\mathbf{lte}_i$ vidíme, že BDFν je metoda řádu ν. Speciálně pro konstantní délku h kroku dostaneme

$$\mathbf{lte}_i = -\frac{1}{\nu+1}\mathbf{y}^{(\nu+1)}(*)h^{\nu+1}.$$

Lokální diskterizační chybu můžeme také určit pomocí vztahu (4.63), do kterého místo $\mathbf{y}_{i+1-j}$ dosadíme $\mathbf{y}(x_{i+1-j})$, tj.

$$\sum_{j=1}^{\nu} \frac{1}{j}\nabla^j \mathbf{y}(x_{i+1}) = h\,\mathbf{f}(x_{i+1}, \mathbf{y}(x_{i+1})) + \mathbf{lte}_i\,.$$

Jestliže horní mez v sumě zvětšíme o jedničku, dostaneme metodu řádu $\nu+1$, takže

$$\sum_{j=1}^{\nu+1} \frac{1}{j}\nabla^j \mathbf{y}(x_{i+1}) = h\,\mathbf{f}(x_{i+1}, \mathbf{y}(x_{i+1})) + O(h^{\nu+2})\,.$$

Odečtením obou rovnic pak získáme

$$\mathbf{lte}_i = -\frac{1}{\nu+1}\nabla^{\nu+1}\mathbf{y}(x_{i+1}) + O(h^{\nu+2})\,.$$

Proto můžeme jako odhad $\mathbf{est}_i$ lokální diskretizační chyby $\mathbf{lte}_i$ použít výraz

$$\mathbf{est}_i = -\frac{1}{\nu+1}\nabla^{\nu+1}\mathbf{y}_{i+1}\,. \tag{4.65}$$

Další vlastnosti BDFν metod. Formule zpětného derivování jsou normalizované. Skutečně, druhý charakteristický polynom $\sigma(\Theta) = \Theta^\nu$ a protože BDFν metoda je řádu $\nu \geq 1$, je $\varrho'(1) = \sigma(1) = 1$. Formule zpětného derivování jsou D-stabilní jen pro $\nu \leq 6$, pro $\nu \geq 7$ jsou BDFν metody nepoužitelné. V dalším se proto omezíme na BDFν metody pro $\nu \leq 6$. Ty jsou vždy konvergentní.

Všimněte si, že chybová konstanta $-1/(\nu+1)$ je pro $\nu > 1$ podstatně větší než u Adamsových metod stejného řádu, tj. BDFν metody jsou značně méně přesné než Adamsovy metody. Metody zpětného derivování však mají jednu ohromnou přednost, která plně ospravedlňuje jejich používání, a tou je skutečnost, že mají neomezenou oblast $\mathscr{R}_A$ absolutní stability. Pro metody BDF1 a BDF2 oblast $\mathscr{R}_A$ absolutní stability obsahuje celou zápornou komplexní polorovinu, $\mathscr{R}_A \supseteq \{z \,|\, \mathrm{Re}\, z < 0\}$, tj. tyto metody jsou A-stabilní. Abychom mohli popsat oblast absolutní stability zbývajících BDFν metod, zavedeme si jeden nový pojem. Řekneme, že numerická metoda je *$A(\alpha)$-stabilní*, $\alpha \in (0, \pi/2\rangle$, jestliže oblast $\mathscr{R}_A$ její absolutní stability obsahuje segment $\{z \,|\, \pi - \alpha < \arg z < \pi + \alpha\}$ komplexní roviny. BDFν metody (pro $\nu \leq 6$) jsou A(α)-stabilní, příslušné úhly ve stupních udává následující tabulka

ν	1	2	3	4	5	6
α	90°	90°	88°	73°	52°	18°

Úhel 18° metody BDF6 je příliš malý a proto se tato metoda obvykle nepoužívá. V programu ODE15S v MATLABu jsou implementovány metody zpětného derivování řádů 1 až 5. Program ODE15S je typu VSVO, tj. volí optimální délku kroku i řád metody. Protože stabilita s růstem ν klesá, uživatel může určit, že se budou používat jen metody řádů $\nu \leq \nu_{max} \leq 5$. Výběr délky kroku a řádu metody vychází z odhadu (4.65) lokální

chyby, významnou roli ale hraje také rychlost konvergence zjednodušené Newtonovy metody použité k řešení nelineární rovnice (4.64).

A_0-stabilní metoda. Metoda, jejíž interval absolutní stability je interval $(-\infty, 0)$, se nazývá A_0-stabilní. $A(\alpha)$-stabilní metoda je A_0-stabilní pro každé α, tj. BDFν metody jsou A_0-stabilní. A_0-stabilní metody se používají např. pro řešení soustav ODR, které vzniknou po provedení prostorové diskretizace nestacionární úlohy vedení tepla, viz odstavec 6.2.

4.5. Tuhé problémy

Při řešení mnoha praktických úloh vznikají soustavy diferenciálních rovnic, které vykazují jev označovaný v anglicky psané literatuře jako „stiffness", samotné soustavy (problémy) se opatřují přívlastkem „stiff", tj. hovoří se o „stiff systems (problems)". Doslovný překlad termínu „stiffness" je „tuhost", budeme tedy hovořit o tuhosti a o tuhých soustavách (problémech), někdy ale použijeme „originální" označení a budeme hovořit o stiff soustavách (problémech). V česky psaných učebnicích [37], [38] se pro tuhé soustavy používá opisné označení „úlohy se silným tlumením", proto příležitostně použijme také toto označení.

Slovo „stiffness" vstoupilo do literatury pravděpodobně z teorie řízení: řídicí mechanismus, jehož chování může být popsáno stiff soustavou ODR, má tendenci přejít do ustáleného stavu a setrvat v něm jako tuhý mechanismus.

4.5.1. Co je to tuhost?

Kvalitní programy založené na explicitních Rungových-Kuttových nebo Adamsových metodách jsou obecně velmi efektivní. Jestliže však pomocí nich chceme řešit tuhý problém, jsou najednou značně neefektivní, dokonce natolik, že jejich použití je zcela nepraktické: pro pevnou délku kroku dostáváme numerické řešení, které (často) osciluje a (vždy) explozivně roste; pokud použijeme automatické řízení délky kroku, lze sice tuhý problém explicitní metodou vyřešit, ale jen za cenu toho, že budou použity extrémně malé kroky. Takové chování tuhých problémů se zdá být záhadné a matoucí. Abychom pochopili o co jde, je třeba tuhou soustavu nějak popsat. Než se k tomu dostaneme, řekněme si, co budeme chápat pod pojmem stabilní (počáteční) problém.

Stabilní počáteční problém. Zhruba řečeno, problém je *stabilní vzhledem k počáteční podmínce*, když malá změna v počáteční podmínce způsobí malou změnu řešení. Podobně řekneme, že problém je *stabilní vzhledem k pravé straně*, když malá změna pravé strany (diferenciální rovnice) způsobí malou změnu řešení. A souhrnně řekneme, že *problém je stabilní*, když je stabilní jak vzhledem k počáteční podmínce tak vzhledem k pravé straně.

Abychom mohli popsat stabilitu přesněji, zaveďme si jeden nový pojem: řekneme, že funkce $\mathbf{f}(x, \mathbf{y})$ splňuje *jednostrannou Lipschitzovu podmínku*, jestliže platí

$$(\mathbf{f}(x, \mathbf{u}) - \mathbf{f}(x, \mathbf{v}), \mathbf{u} - \mathbf{v}) \leq \mathscr{L}\|\mathbf{u} - \mathbf{v}\|^2\,,$$

kde $(\cdot,\cdot)$ je nějaký skalární součin generující normu $\|\cdot\|$, tj. $\|\cdot\|^2 = (\cdot,\cdot)$. Je-li L Lipschitzova konstanta, pak vždy můžeme položit $\mathscr{L} = L > 0$. Zajímavá situace však nastane, když existuje $\mathscr{L} \leq 0$, elementární příklad je funkce $f(x, y) = \mathscr{L}y$ pro $\mathscr{L} \leq 0$. Diferenciální

rovnice, jejíž pravá strana $\mathbf{f}$ splňuje jednostranou Lipschitzovu podmínku s konstantou $\mathscr{L} \le 0$, se nazývá *disipativní*.

Stabilita řeší otázku, do jaké míry se řešení úlohy

$$\mathbf{u}' = \mathbf{f}(x, \mathbf{u}), \qquad \mathbf{u}(a) \text{ dané},$$

může lišit od řešení úlohy

$$\mathbf{v}' = \mathbf{f}(x, \mathbf{v}) + \mathbf{g}(x), \qquad \mathbf{v}(a) \text{ dané},$$

když $\mathbf{f}$ splňuje jednostrannou Lipschitzovu podmínku s konstantou $\mathscr{L}$. Odpověď poskytuje následující nerovnost

$$\|\mathbf{v}(x)-\mathbf{u}(x)\| \le \|\mathbf{v}(a)-\mathbf{u}(a)\| e^{\mathscr{L}(x-a)} + \int_a^x \|\mathbf{g}(t)\| e^{\mathscr{L}(x-t)}\,\mathrm{d}t, \tag{4.66}$$

jejíž důkaz lze najít např. v [34]. Uvedený odhad je zvláště pěkný pro disipativní problémy ($\mathscr{L} \le 0$), neboť v tom případě jsou exponenciály ohraničeny 1, takže jde nepochybně o stabilní problém a to na libovolně dlouhém intervalu $\langle a, b\rangle$. Speciálně pro $\mathbf{g} = \mathbf{0}$ ze (4.66) plyne

$$\|\mathbf{v}(x_2)-\mathbf{u}(x_2)\| \le \|\mathbf{v}(x_1)-\mathbf{u}(x_1)\|, \qquad \text{kde } a \le x_1 \le x_2 \text{ je libovolné}. \tag{4.67}$$

Stabilita numerické metody. Měli bychom si také říci, co rozumíme stabilitou numerické metody. Zhruba řečeno, *numerická metoda je stabilní*, jestliže, aplikována na (libovolný) stabilní počáteční problém, má tuto vlastnost: malá změna startovacích hodnot způsobí malou změnu numerického řešení (pro k-krokovou metodu jsou startovací hodnoty $\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{k-1}$). Zpřesňujících definic je celá řada, pro disipativní počáteční problém lze stabilitu numerické metody definovat (s přihlédnutím k (4.67)) např. (viz [17]) takto :

Definice. *Nechť $\{\mathbf{u}_i\}$ a $\{\mathbf{v}_i\}$ jsou dvě numerická řešení téhož (ale jinak libovolného) disipativního počátečního problému, spočtená zvolenou k-krokovou metodou, avšak pro různé (ale jinak libovolné) startovací hodnoty, tj. $\{\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{k-1}\} \ne \{\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{k-1}\}$. Pak řekneme, že numerická metoda je stabilní, pokud*

$$\|\mathbf{U}_{i+1}-\mathbf{V}_{i+1}\| \le \|\mathbf{U}_i-\mathbf{V}_i\|, \qquad i = k-1, k, \ldots$$

kde $\mathbf{U}_i = (\mathbf{u}_i^T, \mathbf{u}_{i-1}^T, \ldots, \mathbf{u}_{i+1-k}^T)^T$, $\mathbf{V}_i = (\mathbf{v}_i^T, \mathbf{v}_{i-1}^T, \ldots, \mathbf{v}_{i+1-k}^T)^T$.

V dalším se budeme podrobně zabývat stabilitou speciálního modelového problému, který popisuje následující

Věta *(o stabilitě). Uvažujme diferenciální rovnici*

$$\mathbf{y}' = \mathbf{J}\mathbf{y} + \mathbf{g}(x), \tag{4.68}$$

kde $\mathbf{J}$ je konstantní matice, která má úplný systém vlastních vektorů. Nechť $\mathbf{u}(x)$ a $\mathbf{v}(x)$ jsou dvě řešení soustavy (4.68) s počátečními podmínkami $\mathbf{u}(a) = \boldsymbol{\eta}$ a $\mathbf{v}(a) = \boldsymbol{\eta} + \boldsymbol{\delta}$. Nechť vlastní čísla matice $\mathbf{J}$ jsou $\{\lambda_j\}$. Jestliže $\mathrm{Re}(\lambda_j) > 0$ pro nějaké λ_j, pak existuje $\boldsymbol{\delta}$ takové, že $\|\mathbf{u}(x) - \mathbf{v}(x)\| \to \infty$ exponenciálně pro $x \to \infty$. Jestliže $\mathrm{Re}(\lambda_j) \le 0$ pro

všechna j, pak pro každé $\boldsymbol{\delta}$ je $\|\mathbf{u}(x) - \mathbf{v}(x)\|$ stejnoměrně omezená vzhledem k x. Navíc, pokud $\mathrm{Re}(\lambda_j) < 0$ pro všechna j, pak pro každé $\boldsymbol{\delta}$ platí $\|\mathbf{u}(x) - \mathbf{v}(x)\| \to 0$ exponenciálně pro $x \to \infty$.

Důkaz. Jestliže matice $\mathbf{J}$ řádu n má úplný systém vlastních vektorů znamená to, že matice $\mathbf{J}$ má n lineárně nezávislých vlastních vektorů $\mathbf{v}_j$, $j = 1, 2, \ldots, n$, (příslušných k vlastním číslům λ_j) a tedy matice $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n)$ je regulární. Pak $\mathbf{V}^{-1}\mathbf{J}\mathbf{V} = \mathbf{D} = \mathrm{diag}\{\lambda_j\}$ je diagonální matice. Rozdíl $\mathbf{z}(x) = \mathbf{v}(x) - \mathbf{u}(x)$ splňuje $\mathbf{z}' = \mathbf{J}\mathbf{z}$. V nové proměnné $\mathbf{w} = \mathbf{V}^{-1}\mathbf{z}$ máme

$$\mathbf{w}' = \mathbf{V}^{-1}\mathbf{z}' = \mathbf{V}^{-1}\mathbf{J}\mathbf{V}\mathbf{V}^{-1}\mathbf{z} = \mathbf{D}\mathbf{w}.$$

Odtud dostaneme

$$w_j(x) = e^{\lambda_j(x-a)} w_j(a), \qquad j = 1, 2, \ldots, n.$$

Jestliže $\mathrm{Re}(\lambda_j) > 0$ a $w_j(a) \ne 0$, pak $w_j(x)$ exponenciálně roste. Jestliže $\mathrm{Re}(\lambda_j) \le 0$, pak $|w_j(x)| \le |w_j(a)|$ pro všechna x, a když $\mathrm{Re}(\lambda_j) < 0$, pak $w_j(x) \to 0$ exponenciálně. Přechodem k původním proměnným $\mathbf{z} = \mathbf{V}\mathbf{w}$ obdržíme tvrzení věty. □

Úloha (4.68) je tedy stabilní vzhledem k počáteční podmínce, právě když všechna vlastní čísla matice $\mathbf{J}$ mají nekladnou reálnou složku. Dá se ukázat, že když $\mathbf{J}$ je symetrická matice s nekladnými vlastními čísly, pak úloha (4.68) je disipativní (pro skalární součin $(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T\mathbf{v}$ dostaneme $\mathscr{L} = \max \lambda_j \le 0$).

Nyní se již můžeme pokusit o charakteristiku tuhého problému ODR. Přesná definice tuhého systému k dispozici není, proto postupně uvedeme několik tvrzení, která tento pojem spíše objasňují než definují. Dosti výstižné a přitom poměrně obecné je

Tvrzení 1. *Tuhý problém je stabilní problém s Lipschitzovou konstantou L takovou, že $L(b-a) \gg 1$.*

Pro lineární problém (4.68) je $L \ge \varrho(\mathbf{J})$, kde $\varrho(\mathbf{J}) = \max_j |\lambda_j|$ je spektrální poloměr matice $\mathbf{J}$. Skutečně, je-li $\mathbf{w} = \mathbf{u} - \mathbf{v}$ vlastní vektor příslušný vlastnímu číslu λ, pak

$$\|(\mathbf{J}\mathbf{u}+\mathbf{g})-(\mathbf{J}\mathbf{v}+\mathbf{g})\| = \|\mathbf{J}\mathbf{w}\| = \|\lambda\mathbf{w}\| = |\lambda|\|\mathbf{w}\| \le L\|\mathbf{u}-\mathbf{v}\|.$$

Jestliže má matice $\mathbf{J}$ navzájem různá vlastní čísla, můžeme řešení úlohy (4.68) zapsat ve tvaru

$$\mathbf{y}(x) = \sum_{j=1}^{n} c_j e^{\lambda_j x}\mathbf{v}_j + \mathbf{p}(x),$$

kde λ_j jsou vlastní čísla matice $\mathbf{J}$, $\mathbf{v}_j$ jsou odpovídající vlastní vektory, $\mathbf{p}(x)$ je partikulární řešení nehomogenního systému a c_j jsou konstanty, které určíme z počátečních podmínek. Předpokládejme, že $\mathrm{Re}(\lambda_j) < 0$, $j = 1, 2, \ldots, n$. Pak pro obecné řešení $\mathbf{w}(x)$ homogenního systému $\mathbf{w}' = \mathbf{J}\mathbf{w}$ platí

$$\mathbf{w}(x) = \sum_{j=1}^{n} c_j e^{\lambda_j x}\mathbf{v}_j \to \mathbf{0} \qquad \text{pro } x \to \infty.$$

Z toho důvodu mluvíme o funkci $\mathbf{w}(x)$ jako o *přechodovém řešení* a o funkci $\mathbf{p}(x)$ jako o *ustáleném řešení*. Označme dále λ_{max} a λ_{min} taková vlastní čísla, že

$$|\mathrm{Re}(\lambda_{max})| \geq |\mathrm{Re}(\lambda_j)| \geq |\mathrm{Re}(\lambda_{min})|\,, \quad j = 1, 2, \ldots, n\,.$$

Je-li naším úkolem spočítat ustálené řešení, musíme řešit soustavu alespoň do toho bodu, v němž je nejpomaleji klesající exponenciála v přechodovém řešení už zanedbatelná. Čím menší je tedy $|\mathrm{Re}(\lambda_{min})|$, tím na delším intervalu je třeba soustavu řešit, takže délka $b-a$ je přímo úměrná číslu $|1/\mathrm{Re}(\lambda_{min})|$. Jestliže je číslo $|\mathrm{Re}(\lambda_{max})|$ velké, je velká také Lipschitzova konstanta L. Proto je (v souladu s trvzením 1) přirozené považovat podíl

$$S = \frac{|\mathrm{Re}(\lambda_{max})|}{|\mathrm{Re}(\lambda_{min})|}\,,$$

nazývaný *poloměr tuhosti*, za měřítko tuhosti soustavy (4.68). Můžeme proto vyslovit další

Tvrzení 2. *Problém (4.68) je tuhý, jestliže všechna vlastní čísla matice* $\mathbf{J}$ *mají zápornou reálnou část a poloměr tuhosti* S *je velký .*

Tvrzení 2 poněkud zužuje obecnější tvrzení 1. Podle tvrzení 1 lze totiž problém (4.68) považovat za tuhý také v případě, že vlastní čísla matice $\mathbf{J}$ jsou velká ryze imaginární čísla. Tak přicházíme k definici nového pojmu: řekneme, že problém (4.68) je *oscilatoricky tuhý*, jestliže vlastní čísla $\{\lambda_j\}$ matice $\mathbf{J}$ splňují podmínky

$$\mathrm{Re}(\lambda_j) \leq 0, \qquad |\mathrm{Re}(\lambda_j)| \ll 1, \qquad (b-a)\max_j |\lambda_j| \gg 1\,.$$

Věnujme se nyní vztahu mezi tuhým problémem a numerickou metodou jeho řešení. Uvažujme homogenní problém (4.68), tj. problém

$$\mathbf{y}' = \mathbf{J}\mathbf{y}\,, \qquad \mathbf{y}(a) = \boldsymbol{\eta}\,. \tag{4.69}$$

Pak $\mathbf{w} = \mathbf{V}^{-1}\mathbf{y}$ je řešením ekvivalentního problému

$$\mathbf{w}' = \mathbf{D}\mathbf{w}\,, \qquad \mathbf{w}(a) = \mathbf{V}^{-1}\boldsymbol{\eta}\,, \tag{4.70}$$

kde (stejně jako v důkazu výše uvedené věty) $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n)$ je matice vlastních vektorů matice $\mathbf{J}$ a $\mathbf{D} = \mathrm{diag}\{\lambda_j\}$ je diagonální matice odpovídajících vlastních čísel. Nechť $\mathbf{y}_i$, $i = 0, 1, \ldots,$ je numerické řešení problému (4.69) a $\mathbf{w}_i$, $i = 0, 1, \ldots,$ je numerické řešení problému (4.70), přičemž obě řešení jsou získána stejným numerickým výpočtem, tj. stejnou metodou pro stejné délky kroků. Pak není těžké prověřit, že $\mathbf{w}_i = \mathbf{V}^{-1}\mathbf{y}_i$ jak pro Rungovy-Kuttovy metody tak pro LMM.

Je-li $\mathrm{Re}(\lambda_j) < 0$, pak $\mathbf{y}(x) \to \mathbf{0}$ pro $x \to \infty$, tj. úloha (4.69) je stabilní (vzhledem k počáteční podmínce). Stabilitu numerické metody stačí studovat na formálně jednodušším problému (4.70). Ten se rozpadá na samostatné úlohy

$$w_j' = \lambda_j w_j\,, \qquad w_j(a) = [\mathbf{V}^{-1}\boldsymbol{\eta}]_j\,, \qquad j = 1, 2, \ldots, n\,,$$

přičemž $w_j(x) = w_j(a)e^{\lambda_j(x-a)} \to 0$ pro $x \to \infty$. Je proto přirozené požadovat, aby pro přibližné řešení $w_{j,i} \approx w_j(x_i)$ platilo $w_{j,i} \to 0$ pro $i \to \infty$, a to pro každé $j = 1, 2, \ldots, n$.

Když si vzpomeneme na definici absolutní stability zjistíme, že náš požadavek je splněn, právě když (obecně komplexní) čísla $\widehat{h} = h\lambda_j$ leží v oblasti $\in \mathscr{R}_A$ absolutní stability užité numerické metody. Jestliže je oblast $\mathscr{R}_A$ omezená, je omezení, které zmíněný požadavek klade na velikost integračního kroku tím přísnější, čím delší je interval integrace $\langle a, b\rangle$. Je-li $|\mathrm{Re}(\lambda_{max})|$ podstatně větší než $|\mathrm{Re}(\lambda_{min})|$, můžeme se dostat do velmi nepříjemné situace vyžadující řešit na dlouhém intervalu diferenciální rovnici a užít při tom integrační krok malé délky splňující $h\lambda_j \in \mathscr{R}_A\ \forall j$. Potíže, které vznikají při numerickém řešení tuhých problémů ilustruje následující

Příklad 4.2. Uvažujme počáteční úlohu

$$\begin{pmatrix} y_1' \\ y_2' \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1000 & -1001 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \quad x \in (0, \ell)\,, \quad \begin{pmatrix} y_1(0) \\ y_2(0) \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}. \tag{4.71}$$

Řešení je $y_1 = e^{-x}$, $y_2 = -e^{-x}$. Matice soustavy

$$\mathbf{J} = \begin{pmatrix} 0 & 1 \\ -1000 & -1001 \end{pmatrix}$$

má vlastní čísla $\lambda_1 = -1$, $\lambda_2 = -1\,000$, spektrální poloměr $\varrho(\mathbf{J}) = 1\,000$, v $\|\cdot\|_1$ normě platí $\|\mathbf{J}(\mathbf{u}-\mathbf{v})\|_1 \leq \|\mathbf{J}\|_1\|\mathbf{u}-\mathbf{v}\|_1$, přičemž $\|\mathbf{J}\|_1 = 1002$, takže Lipschitzovu konstantu L lze vybrat tak, že $1\,000 \leq L \leq 1\,002$. Připomeňme si, že podle tvrzení 1 tuhost problému reprezentuje číslo $L\ell$.

Úlohu (4.71) jsme řešili explicitní BS32 metodou (program ODE23 v MATLABu) a implicitní TR metodou (program ODE23T v MATLABu). Oba programy jsme použili se stejným požadavkem na přesnost (konkrétně $\varepsilon_R = 10^{-3}$ a $\varepsilon_A = 10^{-6}$). BS32 je (explicitní) Rungova-Kuttova metoda řádu 3, která má interval absolutní stability $(-2,51; 0)$. Proto délka h kroku musí splňovat podmínku $h\lambda_2 \leq 2,51$, tj. $h \leq 2,51 \cdot 10^{-3}$. TR metoda je implicitní Rungova-Kuttova metoda řádu 2 a je A-stabilní, takže délka kroku z důvodu stability omezena není.

Efektivnost obou metod lze přibližně porovnat podle počtu úspěšně provedených kroků `pk` nebo lépe, podle počtu `pf` vyhodnocení pravé strany. V následující tabulce jsou uvedeny hodnoty `pk/pf` pro několik délek ℓ intervalu integrace.

ℓ	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
BS32	10/32	40/128	399/1 211	3 982/11 960	39 799/119 411
TR	10/15	14/21	16/24	67/79	86/108

Pro větší ℓ je problém (4.71) tuhý a to je důvod, proč k jeho efektivnímu řešení nelze použít explicitní metodu BS32 s omezeným intervalem absolutní stability.

Délka kroku BS32 metody je určena „přísnou“ podmínkou stability. Proto při zvětšení intervalu integrace z $\ell = 10$ na $\ell = 100$ počet vyhodnocení pravé strany vzroste přibližně 10-krát. Jinak je tomu u A-stabilní TR metody. Tam je délka kroku řízena velikostí lokální chyby a ta se s rostoucím x zmenšuje. O tom svědčí to, že pro $\ell = 10$ se provedlo 67 kroků a pro $\ell = 100$ celkem 86 kroků, tj. na interval $\langle 10, 100\rangle$ připadlo jen 19 dalších kroků. Na intervalu $\langle 10, 100\rangle$ lze tedy pozorovat paradoxní situaci: přesnější metoda BS32 vyžaduje

výrazně menší časový krok než méně přesná TR metoda. Délku kroku zde totiž omezuje stabilita metody a ne její přesnost.

Nenechte se zmást tím, že nestabilita se projevuje prostřednictvím lokální chyby, tj. pomocí nástroje pro měření přesnosti metody: jestliže délka kroku nesplňuje podmínku stability, dojde v několika málo krocích k takovému nárustu lokální chyby, že na to zareaguje mechanismus automatického řízení délky kroku a krok patřičně zkrátí. □

Právě uvedený příklad vystihuje charakteristikou vlastnost tuhých soustav ODR. Tuto vlastnost sformulujeme jako

Tvrzení 3. *Tuhost soustavy ODR se projevuje tím, že při jejím numerickém řešení délku kroku omezuje spíše stabilita metody než její přesnost.*

Až dosud jsme naše úvahy o stabilitě numerické metody zakládali na analýze jejího chování při řešení lineární soustavy (4.68) s konstantní maticí $\mathbf{J}$. Je jistě rozumné požadovat, aby metoda fungovala dobře na speciální třídě rovnic (4.68), je však třeba mít na paměti, že tak dostáváme jen hrubou informaci o možném chování metody při řešení problémů složitějších. Ukažme si nyní, proč se můžeme domnívat, že taková informace má vůbec nějakou vypovídací hodnotu.

Pro obecný problém $\mathbf{y}' = \mathbf{f}(x, \mathbf{y})$ aproximujeme funkci $\mathbf{f}$ v blízkosti bodu $(x_i, \mathbf{y}(x_i))$ lineární funkcí v proměnné $\mathbf{y}$,

$$\mathbf{f}(x,\mathbf{y}) \approx \mathbf{f}(x_i,\mathbf{y}(x_i)) + \frac{\partial \mathbf{f}}{\partial x}(x_i,\mathbf{y}(x_i))(x-x_i) + \frac{\partial \mathbf{f}}{\partial \mathbf{y}}(x_i,\mathbf{y}(x_i))(\mathbf{y}-\mathbf{y}(x_i))\,,$$

a doufáme, že chování numerického řešení původního problému lze v blízkosti bodu $(x_i, \mathbf{y}(x_i))$ objasnit z chování numerického řešení aproximujícího problému

$$\mathbf{u}' = \mathbf{J}^i\mathbf{u} + \mathbf{g}^i(x)$$

s konstantní Jacobiovou maticí $\quad \mathbf{J}^i = \mathbf{f}_{\mathbf{y}}(x_i, \mathbf{y}(x_i))$

a s funkcí $\quad \mathbf{g}^i(x) = \mathbf{f}(x_i, \mathbf{y}(x_i)) + \mathbf{f}_x(x_i, \mathbf{y}(x_i))(x - x_i) - \mathbf{J}^i\mathbf{y}(x_i)\,.$

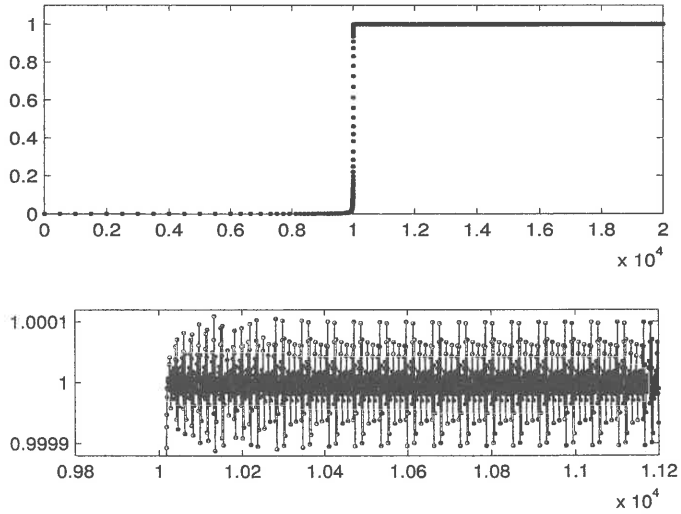
Aproximující problém je téhož typu jako problém (4.68), takže stabilitu numerické metody můžeme posoudit známým způsobem pomocí délky kroku h a vlastních čísel $\{\lambda_j\}$ konstantní Jacobiovy matice $\mathbf{J}^i$. Jestliže to uděláme, tak vlastně tiše předpokládáme, že stabilita metody aplikované na aproximující problém je stejná jako stabilita metody aplikované na problém obecný. To není vždy pravda, přesto však tento přístup může být pro praxi přínosem, nesmíme ale zapomenout, že případná podmínka stability omezující délku kroku má lokální charakter, tj. platí jen v okolí zvoleného bodu $(x_i, \mathbf{y}(x_i))$.

Právě uvedený postup odpovídá klasickému postupu, který se používá v teorii diferenciálních rovnic, kdy se stabilita řešení nelineárního problému zkoumá prostřednictvím stability aproximujícího lineárního problému. V našem případě je třeba analyzovat navíc ještě stabilitu numerické metody aplikované na aproximující problém. Z teorie obyčejných diferenciálních rovnic je známo, že analýza lineární stability je užitečná, je však třeba mít na zřeteli její omezenou působnost. Totéž platí tím spíše i pro analýzu lineární stability numerických metod. Dokonalejší nástroje pro posuzování stability numerických metod při řešení nelineárních problémů vycházejí z teorie nelineární stability ODR, tím se však my zde zabývat nebudeme, základní informace k tomuto tématu lze najít např. v [17].

Příklad 4.3. Uvažujme počáteční problém

$$y' = y^2 - y^3\,, \qquad x \in (0, 2/\delta)\,, \qquad y(0) = \delta\,. \tag{4.72}$$

Diferenciální rovnice má dvě konstantní řešení $y = 0$ a $y = 1$. Zvolíme-li počáteční podmínku $y(0) \le 0$, pak $y(x) \to 0$ pro $x \to \infty$, zatímco pro $y(0) > 0$ dostaneme $y(x) \to 1$ pro $x \to \infty$. Zvolíme-li $\delta > 0$ velmi malé, pak pravá strana $y^2 - y^3$ diferenciální rovnice nabývá malých kladných hodnot, tj. funkce $y(x)$ velmi pomalu roste a na poměrně dlouhém intervalu zůstávají její hodnoty blízké k 0. Konkrétně pro $\delta = 10^{-4}$ je $y(x) < 10^{-2}$ ještě pro $x = 9\,900$, pak $y(x)$ začíná prudce růst a pro $x > 10\,020$ je už $y(x)$ prakticky rovno 1. Roli Lipschitzovy konstanty L v nelineární úloze může do jisté míry reprezentovat velikost derivace pravé strany diferenciální rovnice, tj. výraz $|J| := |2y - 3y^2|$. Pro malé $y \approx 0$ je $|J| \approx 0$, pro y blízké 1 je však $|J| \approx 1$. Na intervalu $(0, 9\,900)$ je výraz $9\,900|J|$ (charakterizující podle tvrzení 1 tuhost) poměrně malý, nejde zde proto o tuhý problém, takže délka kroku se řídí především přesností metody. V intervalu $(9\,900; 10\,020)$ se řešení prudce mění, to mechanismus automatického řízení délky kroku zachytí a krok zkrátí. Důvodem zkrácení kroku je zde spíše prudká změna řešení (což se někdy nepřesně označuje jako snížená hladkost) než narůstající tuhost. Poté, co řešení nabude hodnotu rovnou přibližně 1, je délka kroku metody řízena stabilitou.



Obr. 4.2. Příklad 4.3 řešený DP54 metodou

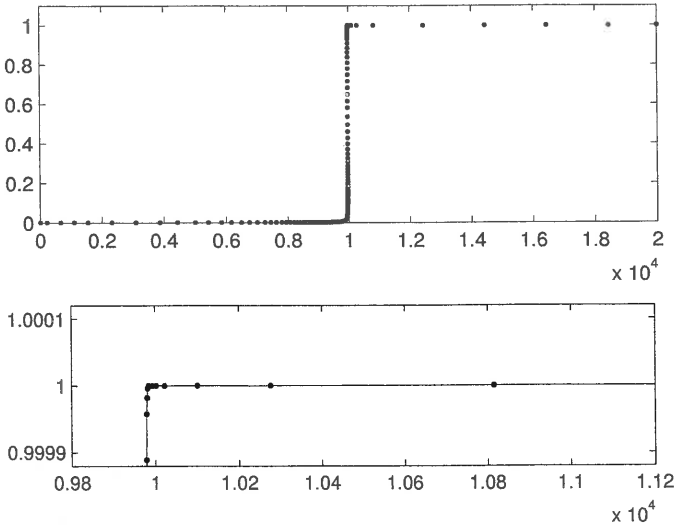
Úlohu (4.72) jsme řešili explicitní metodou DP54 (program ODE45 v MATLABu) a implicitní TR metodou (program ODE23T v MATLABu). Oba programy jsme použili se stejnou přesností ($\varepsilon_R = 10^{-4}$, $\varepsilon_A = 10^{-7}$). DP54 je explicitní Rungova-Kuttova metoda řádu 5 s intervalem absolutní stability $(-3,30; 0)$. Délka kroku proto musí splňovat podmínku stability $h|J| \le 3,3$, což pro $x > 10\,020$ znamená volit $h \doteq 3,3$. TR metoda

(nebo-li lichoběžníková metoda) je A-stabilní metoda řádu 2, takže tato metoda délku kroku z důvodu stability nijak neomezuje.

Průběh výpočtu znázorňuje pro každou z metod dvojice obrázků: horní zachycuje celý výpočet, dolní pak zvětšený výřez pro $t \in \langle 9\,800, 11\,200\rangle$. V následující tabulce uvádíme pro intervaly $(0, \ell)$, kde ℓ je postupně rovno 9 900, 10 020 a 20 000, údaj `pk/pf`, kde `pk` je počet (úspěšně) provedených kroků a `pf` je celkový počet vyhodnocení pravé stany:

ℓ	9900	10 020	20 000
DP54	17/151	36/331	3 041/20 245
TR	85/170	184/385	192/399

Vidíme, že v intervalu (10 020, 20 000), kde přesné řešení je prakticky rovno 1, je TR-metoda velmi efektivní, na zdolání intervalu (10 020, 20 000) potřebovala jen 8 kroků. Zato metoda DP54 na tomto intervalu provedla 3 005 kroků délky $h \doteq 3,3$, takže její použití rozhodně vhodné není. □



Obr. 4.2. Příklad 4.3 řešený TR metodou

Výsledky příkladu 2 umožňují vyslovit praktické kritérium, podle něhož lze vcelku spolehlivě zjistit, zda uvažovaný problém je tuhý.

Tvrzení 4. *Jestliže numerická metoda s omezenou oblastí absolutní stability, aplikovaná na počáteční problém, je nucena v jistém intervalu integrace používat krok, jehož délka je nepřiměřeně malá vzhledem k hladkosti přesného řešení v tomto intervalu, pak to znamená, že problém je v tomto intervalu tuhý.*

Velmi zjednodušený postup řešení počáteční úlohy, o níž nevíme, zda je či není tuhá, je tedy tento: zkusíme úlohu vyřešit přesnou explicitní metodou (v MATLABu třeba programem ODE113 založeným na Adamsových formulích řádů 1 až 12) a pokud řešení bude trvat příliš dlouho v důsledku volby extrémně krátkých kroků, výpočet přerušíme a zkusíme program pro řešení tuhých úloh (v MATLABu třeba program ODE15S založený na metodách zpětného derivování řádů 1 až 5). Pokud pravá strana diferenciální rovnice není příliš hladká, je vhodnější použít metody nižších řádů, třeba explicitní metodu BS32 (v MATLABu program ODE23) a v případě neúspěchu A-stabilní implicitní metodu, třeba TR-BDF2 (v MATLABu program ODE23TB).

Pokud dopředu víme, že problém je resp. není tuhý, neexperimentujeme zbytečně a přímo použijeme program určený pro řešení příslušného typu úloh.

Chceme-li efektivně využít všechny možnosti, které nám dostupné programy nabízejí, je třeba se seznámit s jejich vlastnostmi, vybrat správný program a vhodně nastavit jeho řídicí parametry (v kolekci programů v MATLABu jsou to například parametry ovlivňující řízení délky kroku, výstup výsledků, u implicitních metod je třeba určit způsob výpočtu Jacobiovy matice pravé strany, a jsou ještě další parametry, které zde neuvádíme a o nichž se lze poučit z programové dokumentace).

4.5.2. Řešení soustav nelineárních rovnic

Numerické metody pro řešení tuhých soustav jsou vždy implicitní. Jako příklad si uveďme TR metodu

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \frac{1}{2}h[\mathbf{f}(x_{i+1}, \mathbf{y}_{i+1}) + \mathbf{f}(x_i, \mathbf{y}_i)]\,.$$

Rovnice pro $\mathbf{y}_{i+1}$ má tvar

$$\mathbf{y} = h\gamma\mathbf{f}(\mathbf{y}) + \boldsymbol{\psi}\,. \tag{4.73}$$

Pro větší přehlednost jsme v tomto zápisu potlačili závislost $\mathbf{f}(x, \mathbf{y})$ na x, neboť $x = x_{i+1}$ se při výpočtu $\mathbf{y}$ nemění. γ je charakteristická konstanta metody a $\boldsymbol{\psi}$ je konstantní vektor reprezentující dříve spočtená data. Konkrétně pro TR metodu

$$\gamma = \frac{1}{2}\,, \qquad \boldsymbol{\psi} = \mathbf{y}_i + \frac{1}{2}h\mathbf{f}(x_i, \mathbf{y}_i)\,.$$

Rovnice (4.73) je vhodným modelem pro každou implicitní LMM, zejména pro BDF metody, ale také např. pro semi-implicitní Rungovy-Kuttovy metody TRX2 a TR-BDF2.

Nejdříve si objasníme, proč rovnici (4.73) nemůžeme řešit metodou prosté iterace v případě, když problém je tuhý. Předpokládejme, že $\mathbf{y}^*$ je řešení a $\mathbf{y}^{(s)}$ je už spočtená aproximace v $s-$té iteraci. Pak další aproximace je

$$\mathbf{y}^{(s+1)} = h\gamma\mathbf{f}(\mathbf{y}^{(s)}) + \boldsymbol{\psi}\,.$$

Odečteme-li rovnici, kterou splňuje $\mathbf{y}^*$, dostaneme rovnici pro chybu:

$$\mathbf{y}^{(s+1)} - \mathbf{y}^* = h\gamma[\mathbf{f}(\mathbf{y}^{(s)}) - \mathbf{f}(\mathbf{y}^*)]\,.$$

Speciálně pro diferenciální rovnici (4.68) proto platí

$$\mathbf{y}^{(s+1)} - \mathbf{y}^* = h\gamma\mathbf{J}(\mathbf{y}^{(s)} - \mathbf{y}^*) .$$

Jestliže jako počáteční aproximaci $\mathbf{y}^{(0)}$ zvolíme vlastní vektor $\mathbf{v}$ matice $\mathbf{J}$ příslušný vlastnímu číslu λ, pak chyba $\mathbf{y}^{(s)} - \mathbf{y}^* = (h\gamma\lambda)^s\mathbf{v}$. Konvergence tedy nastane, jen když $|h\gamma\lambda| < 1$ pro všechna vlastní čísla matice $\mathbf{J}$. Takové omezení délky kroku je však pro tuhý systém nepřijatelné.

Rovnici (4.73) budeme řešit iteračně, jinak to obecně nejde, víme však již, že nepůjde o iteraci prostou. Počáteční aproximace $\mathbf{y}^{(0)}$ se obvykle určí dostatečně přesnou interpolací z hodnot $\mathbf{y}_i, \mathbf{y}_{i-1}, \ldots$. Funkci $\mathbf{f}$ linearizujeme okolo stávající aproximace $\mathbf{y}^{(s)}$,

$$\mathbf{f}(\mathbf{y}) \approx \mathbf{f}(\mathbf{y}^{(s)}) + \mathbf{J}(\mathbf{y} - \mathbf{y}^{(s)}) ,$$

a další aproximaci $\mathbf{y}^{(s+1)}$ dostaneme řešením rovnice

$$\mathbf{y}^{(s+1)} = \boldsymbol{\psi} + h\gamma\mathbf{f}(\mathbf{y}^{(s)}) + h\gamma\mathbf{J}(\mathbf{y}^{(s+1)} - \mathbf{y}^{(s)}) . \tag{4.74}$$

Jestliže zvolíme $\mathbf{J} = \mathbf{f}_\mathbf{y}(\mathbf{y}^{(s)})$, dostáváme známou Newtonovu metodu. Ta vzhledem k dobré počáteční aproximaci konverguje velmi rychle. Problém je ale v tom, že takový postup je výpočetně příliš nákladný. V každé iteraci se totiž musí sestavit Jacobiova matice a řešit nová soustava lineárních rovnic. Proto všechny moderní programy pracují tak, že matici $\mathbf{J}$ drží co nejdéle konstantní. Aby iterace $\mathbf{y}^{(s)}$ konvergovaly, měla by matice $\mathbf{J}$ být dobrou aproximací Jacobiovy matice v řešení $\mathbf{y}^*$, tj. $\mathbf{J} \approx \mathbf{f}_\mathbf{y}(\mathbf{y}^*)$. Prakticky se postupuje tak, že na začátku řešení spoustavy ODR, tj. pro $x = a$, se za $\mathbf{J}$ vezme Jacobiova matice funkce $\mathbf{f}$ v bodě $[a, \boldsymbol{\eta}]$, a pokud se při řešení rovnice (4.73) zjistí, že iterační metoda (4.74) konverguje příliš pomalu, Jacobiova matice se přepočítá, tj. za $\mathbf{J}$ se vezme Jacobiova matice funkce $\mathbf{f}$ v bodě $[x_{i+1}, \mathbf{y}^{(s)}]$. Takto upravená Newtonova metoda se nazývá *zjednodušená Newtonova metoda* nebo také *metoda tětiv*.

Výpočet je dobré organizovat tak, že počítáme $\boldsymbol{\Delta} = \mathbf{y}^{(s+1)} - \mathbf{y}^{(s)}$ řešením soustavy rovnic

$$\left(\frac{1}{h\gamma}\mathbf{I} - \mathbf{J}\right)\boldsymbol{\Delta} = \mathbf{f}(\mathbf{y}^{(s)}) + \frac{1}{h\gamma}(\boldsymbol{\psi} - \mathbf{y}^{(s)}) , \tag{4.75}$$

a pak položíme $\mathbf{y}^{(s+1)} = \mathbf{y}^{(s)} + \boldsymbol{\Delta}$. Pro malé h je matice soustavy dobře podmíněná. Navíc, protože pravá strana „je malá“ (jde to reziduum rovnice (4.73) dělené číslem $h\gamma$), jsou malé také zaokrouhlovací chyby vznikající při řešení soustavy rovnic. Soustavu rovnic (4.75) můžeme řešit buď to pomocí vhodné iterační metody nebo metodou přímou. Všimněme si blíže druhého případu (použitého např. v MATLABu). Matice

$$\mathbf{G} := \frac{1}{h\gamma}\mathbf{I} - \mathbf{J}$$

soustavy (4.75) obsahuje tři členy, které se mohou měnit: h při změně délky kroku, γ při změně metody (třeba ve VSVO implementaci metod zpětného derivování) a $\mathbf{J}$ při přepočítání Jacobiovy matice. Pokud se žádný z těchto členů nezmění, zůstává matice $\mathbf{G}$ stejná.

Toho je třeba využít: pouze při změně $\mathbf{G}$ provedeme výpočetně náročný LU rozklad matice soustavy $\mathbf{G} = \mathbf{LU}$, kde $\mathbf{L}$ je dolní trojúhelníková matice a $\mathbf{U}$ horní trojúhelníková matice. V následujících iteracích, kdy se matice soustavy $\mathbf{G}$ nemění, provádíme výpočtově nenáročná řešení dvou soustav rovnic s trojúhelníkovou maticí soustavy, tj. $\mathbf{L}\boldsymbol{\delta} = \mathbf{b}$ a pak $\mathbf{U}\boldsymbol{\Delta} = \boldsymbol{\delta}$, kde $\mathbf{b}$ je pravá strana rovnice (4.75). Program, který má pracovat efektivně, musí mít promyšlenou strategii, podle níž rozhodne, kdy změní $\mathbf{J}$, což je výpočetně nejnáročnější, a kdy jen h nebo γ. Do detailů zde zacházet nebudeme, zájemce odkazujeme na skvělou monografii [34]. Neocenitelným zdrojem poučení je také studium kódů programů pro řešení ODR v MATLABu, viz [19].

Výpočet Jacobiovy matice. Časově velmi náročnou součástí algoritmu řešení rovnice (4.73) je výpočet Jacobiovy matice $\mathbf{J}$. Je žádoucí, aby uživatel poskytl maximum informací pro její výpočet. Řada programů umožňuje zadávat přímo předpis pro výpočet prvků Jacobiovy matice. To ale většinou uživatel neumí a proto Jacobiovu matici program počítá numericky. Postupuje se obvykle tak, že prvky $\partial\mathbf{f}(x, \mathbf{y})/\partial y_\ell$ jejího ℓ-tého sloupce se spočtou přibližně užitím nejjednoduššího vzorce numerického derivování

$$\frac{\mathbf{f}(x, \mathbf{y} + \delta\mathbf{e}_\ell) - \mathbf{f}(x, \mathbf{y})}{\delta} ,$$

kde $\delta > 0$ je malé kladné číslo a $\mathbf{e}_\ell = (0, \ldots, 0, 1, 0, \ldots, 0)^T$ je sloupcový vektor, který obsahuje jedničku v ℓ-té pozici a ostatní složky má nulové. Numerický výpočet lze podstatně urychlit, pokud uživatel popíše strukturu nenulových prvků Jacobiovy matice (tj. pokud programu dodá indexy (j, k) všech pozic, v nichž prvky Jacobiovy matice mohou nabývat nenulových hodnot).

Poměrně častý je případ, kdy Jacobiova matice je pásová. Jestliže j-tá pravá strana $f_j(x, \mathbf{y})$ neobsahuje proměnné y_k pro $|k - j| > s$, pak Jacobiova matice může mít nenulové prvky jedině v pásu kolem hlavní diagonaly. Číslo $s + 1$ se nazývá šíře polopásu. V tomto případě lze numerický výpočet Jacobiovy matice zorganizovat tak, aby se funkce $\mathbf{f}$ počítala jen $2(s + 1)$-krát, viz [34].

Přesnější výpočet derivace. Řada programů pro řešení tuhých problémů považuje za neznámou $\mathbf{z}_{i+1} = h\mathbf{f}(x_{i+1}, \mathbf{y}_{i+1})$ místo $\mathbf{y}_{i+1}$. Jestliže v rovnici (4.73) položíme $\mathbf{z} = h\mathbf{f}(\mathbf{y})$, dostaneme $\mathbf{y} = \gamma\mathbf{z} + \boldsymbol{\psi}$ a odtud

$$\mathbf{z} = h\mathbf{f}(\mathbf{y}) = h\mathbf{f}(\gamma\mathbf{z} + \boldsymbol{\psi}) .$$

Linearizace okolo $\gamma\mathbf{z} + \boldsymbol{\psi}$ vede na iterační předpis

$$\mathbf{z}^{(s+1)} = h\mathbf{f}(\gamma\mathbf{z}^{(s)} + \boldsymbol{\psi}) + h\gamma\mathbf{J}(\mathbf{z}^{(s+1)} - \mathbf{z}^{(s)}) ,$$

a odtud dostaneme pro $\boldsymbol{\delta} = \mathbf{z}^{(s+1)} - \mathbf{z}^{(s)}$ rovnici

$$\left(\frac{1}{h\gamma}\mathbf{I} - \mathbf{J}\right)\boldsymbol{\delta} = \frac{1}{\gamma}\mathbf{f}(\gamma\mathbf{z}^{(s)} + \boldsymbol{\psi}) - \frac{1}{h\gamma}\mathbf{z}^{(s)} ,$$

jejíž matice soustavy je stejná jako matice soustavy (4.75). Zřejmě $\mathbf{z}^{(s+1)} = \mathbf{z}^{(s)} + \boldsymbol{\delta}$. Pokud zvolíme počáteční aproximaci $\mathbf{z}^{(0)} = (\mathbf{y}^{(0)} - \boldsymbol{\psi})/\gamma$, pak $\mathbf{y}^{(s+1)} = \gamma\mathbf{z}^{(s+1)} + \boldsymbol{\psi}$ lze vyjádřit ve

tvaru $\mathbf{y}^{(s+1)} = \mathbf{y}^{(s)} + \gamma\boldsymbol{\delta}$, jak lze snadno ověřit. Výpočet tedy můžeme organizovat takto:

$$\begin{aligned}
&\text{určíme } \mathbf{y}^{(0)},\ \text{položíme } \mathbf{z}^{(0)} = (\mathbf{y}^{(0)} - \boldsymbol{\psi})/\gamma\,,\\
&\text{z rovnice } \left(\frac{1}{h\gamma}\mathbf{I} - \mathbf{J}\right)\boldsymbol{\delta} = \frac{1}{\gamma}\mathbf{f}(\mathbf{y}^{(s)}) - \frac{1}{h\gamma}\mathbf{z}^{(s)} \quad \text{vypočteme } \boldsymbol{\delta}\,, \qquad (4.76)\\
&\text{určíme } \mathbf{z}^{(s+1)} = \mathbf{z}^{(s)} + \boldsymbol{\delta}\,, \quad \mathbf{y}^{(s+1)} = \mathbf{y}^{(s)} + \gamma\boldsymbol{\delta}\,.
\end{aligned}$$

Objasněme si přínos výpočtu podle algoritmu (4.76). Nechť $\mathbf{z}^* = h\mathbf{f}(\mathbf{y}^*)$. Jestliže spočteme $\mathbf{y}^{(s)}$ podle (4.75) a položíme $\mathbf{z}^{(s)} = h\mathbf{f}(\mathbf{y}^{(s)})$, pak pro chybu $\mathbf{z}^* - \mathbf{z}^{(s)}$ dostaneme

$$\mathbf{z}^* - \mathbf{z}^{(s)} = h\mathbf{f}(\mathbf{z}^*) - h\mathbf{f}(\mathbf{z}^{(s)}) = h\mathbf{J}(*)(\mathbf{y}^* - \mathbf{y}^{(s)})\,,$$

kde $\mathbf{J}(*)$ je Jacobiova matice funkce $\mathbf{f}$, jejíž prvky jsou vyhodnoceny v (různých) bodech úsečky spojující body $\mathbf{y}^*$ a $\mathbf{y}^{(s)}$. Pokud však $\mathbf{z}^{(s)}$ počítáme podle (4.76), pak

$$\mathbf{z}^* - \mathbf{z}^{(s)} = \frac{1}{\gamma}(\mathbf{y}^* - \boldsymbol{\psi}) - \frac{1}{\gamma}(\mathbf{y}^{(s)} - \boldsymbol{\psi}) = \frac{1}{\gamma}(\mathbf{y}^* - \mathbf{y}^{(s)})\,.$$

Pro tuhý systém lze očekávat, že $\|h\mathbf{J}(*)\| \gg 1$, a v tom případě je přibližný výpočet derivace podle (4.76) výrazně přesnější.

Statistika o činnosti programů pro řešení ODR. Kvalitní programy uživateli vždy poskytují informaci o tom, jak úspěšně si při řešení konkrétního poblému počínaly. Tak třeba v MATLABu se dodávají tato čísla:

- `pk` počet úspěšných kroků
- `pn` počet neúspěšných kroků
- `pf` počet vyhodnocení pravé strany $\mathbf{f}$
- `pj` počet výpočtů Jacobiovy matice $\mathbf{J}$
- `pr` počet rozkladů $\mathbf{G} = \mathbf{LU}$
- `ps` počet řešených soustav $\mathbf{LU}\boldsymbol{\delta} = \mathbf{b}$

Pro ilustraci uvádíme

Příklad 4.4. Robertsonův problém

$$\begin{aligned}
y_1' &= -0,04\,y_1 + 10^4\,y_2y_3\,, & y_1(0) &= 1\,,\\
y_2' &= 0,04\,y_1 - 10^4\,y_2y_3 - 3\cdot 10^7\,y_2^2\,, & y_2(0) &= 0\,,\\
y_3' &= 3\cdot 10^7\,y_2^2\,, & y_3(0) &= 0
\end{aligned}$$

popisuje koncentrace tří příměsí v chemické reakci, tj. $0 \le y_1, y_2, y_3 \le 1$, nezávisle proměnná t je čas, blíže viz [34]. Jacobiova matice této soustavy je

$$\mathbf{J} = \begin{pmatrix} -0,04 & 10^4\,y_3 & 10^4\,y_2 \\ 0,04 & -10^4\,y_3 - 6\cdot 10^7 y_2 & -10^4 y_2 \\ 0 & 6\cdot 10^7 y_2 & 0 \end{pmatrix}.$$

V čase $t = 0$, tj. pro $y_1 = 1$, $y_2 = y_3 = 0$, má Jacobiova matice vlastní čísla $\{-0,04; 0; 0\}$. Z fyzikálních úvah plyne, že $y_1, y_2 \to 0$ a $y_3 \to 1$ pro $t \to \infty$. Vlastní čísla Jacobiovy

matice pro $y_1 = y_2 = 0$, $y_3 = 1$, jsou $\{-10\,000,04; 0; 0\}$. Při řešení na intervalu $(0, 10^{10})$ je Robertsonův problém tuhý. O tom se lze ostatně snadno přesvědčit experimentálně: explicitní metody selhávají, metody pro řešení tuhých problémů zabírají. Numerickým výpočtem lze zjistit, že již pro $t > 0,01$ je spektrální poloměr $\varrho(\mathbf{J}) > 2\cdot 10^3$, takže Robertsonův problém lze považovat za tuhý již na nepoměrně kratším intervalu délky řádově v jednotkách.

Úlohu jsme řešili v MATLABu třemi programy určenými pro tuhé problémy: programem ODE23T (metoda TR), programem ODE23TB (metoda TR-BDF2) a programem ODE15S (metody BDF1-5). Jacobiova matice se počítala přesně, délku kroku jsme řídili pomocí tolerancí $\varepsilon_R = 10^{-3}$, $\varepsilon_A = 10^{-6}$, činnost programu ODE15S jsme omezili tak, aby pracoval jen s BDF metodami řádů 1,2 a 3. Do následující tabulky jsme zapsali „statistiku" výpočtu, tj. čísla `pk`, `pn`, `pf`, `pj`, `pr`, `ps`:

	pk	pn	pf	pj	pr	ps
ODE23T	238	74	794	37	188	644
ODE23TB	140	13	630	10	93	728
ODE15S	245	15	504	11	67	458

Pro úplnost uvádíme, že MATLAB nabízí pro řešení tuhých problémů ještě program ODE23S (založený na modifikované Rosenbrockově metodě). Kromě toho, program ODE15S lze použít ve variantě založené na jisté optimalizaci BDF metod (jde o tzv. metody numerického derivování), podrobnosti viz [19], [36]. □