Znanstveno računanje 2

3. dio vježbi
Modeli s diferencijalnim jednadžbama

Nela Bosner

Inicijalni problem za običnu diferencijalnu jednadžbu

- Rješavamo običnu diferencijalnu jednadžbu (skraćeno ODJ) oblika

$$y'(x) = f(x, y(x)), \quad x \in \langle a, b \rangle, \tag{1}$$

  - uz zadani početni uvjet $y(a) = y_0$ — **inicijalni problem**
  - ili uz zadani rubni uvjet $r(y(a), y(b)) = 0$, gdje je r neka zadana funkcija — **rubni problem**.
- Sustav običnih diferencijalnih jednadžbi je općenitiji problem:

$$\begin{aligned} y_1' &= f_1(x, y_1, \ldots, y_n), \\ y_2' &= f_2(x, y_1, \ldots, y_n), \\ &\vdots \\ y_n' &= f_n(x, y_1, \ldots, y_n). \end{aligned}$$

- Koristeći vektorsku notaciju

$$\mathbf{y} = [y_1, \ldots, y_n]^T \quad \text{i} \quad \mathbf{f} = [f_1, \ldots, f_n]^T$$

sustav pišemo u obliku analognom jednadžbi (1):

$$\mathbf{y}'(x) = \mathbf{f}(x, \mathbf{y}(x)),$$

te primijenjujemo iste numeričke metode kao za rješavanje diferencijalne jednadžbe (1) vodeći računa o tome da se umjesto skalarnih funkcija y i f javljaju vektorske funkcije $\mathbf{y}$ i $\mathbf{f}$.

- Diferencijalne jednadžbe višeg reda

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

supstitucijama

$$y_1 = y, \quad y_2 = y', \quad \ldots, \quad y_n = y^{(n-1)}$$

svodimo na sustav jednadžbi prvog reda:

$$\begin{aligned} y_1' &= y' = y_2, \\ y_2' &= y'' = y_3, \\ &\vdots \\ y_{n-1}' &= y^{(n-1)} = y_n, \\ y_n' &= y^{(n)} = f(x, y, y', y'', \ldots, y^{(n-1)}) = f(x, y_1, y_2, y_3, \ldots, y_n), \end{aligned}$$

te i u ovom slučaju možemo koristiti metode razvijene za diferencijalnu jednadžbu (1).

Eulerova metoda

- Eulerova metoda je zasigurno najjednostavnija metoda za rješavanje inicijalnog problema za ODJ oblika

$$y' = f(x, y), \quad y(a) = y_0.$$

- Metoda se zasniva na ideji da se y' u gornjoj jednadžbi zamijeni s podijeljenom razlikom

$$y'(x) = \frac{y(x+h) - y(x)}{h} + \mathcal{O}(h),$$

pa rješenje diferencijalne jednadžbe zadovoljava

$$y(x+h) = y(x) + hy'(x) + \mathcal{O}(h^2) = y(x) + hf(x, y(x)) + \mathcal{O}(h^2)$$

- Interval $[a, b]$ podijelimo na n jednakih dijelova te stavimo

$$h = \frac{b-a}{n}, \quad x_i = a + ih, \quad i = 0, \ldots, n.$$

- Eulerova metoda, se tada može kraće zapisati rekurzijom

$$y_{i+1} = y_i + hf(x_i, y_i), \quad i = 1, \ldots, n,$$

gdje je početni uvjet y_0 zadan kao inicijalni uvjet diferencijalne jednadžbe.

- Dobivene vrijednosti y_i su aproksimacije rješenja diferencijalne jednadžbe u točkama x_i.

Algoritam (Eulerova metoda)

y_0, a, b i n *zadani;*
funkcija f *zadana;*
$h = \frac{b-a}{n}$;
$x(1) = a$;
$y(1) = y_0$;
for $i = 1 : n$
    $x(i+1) = a + i * h$;
    $y(i+1) = y(i) + h * f(x(i), y(i))$;
end

Runge–Kutta metode

- Koristeći sličnu ideju kao u Eulerovoj metodi, diferencijalnu jednadžbu

$$y' = f(x, y), \quad y(a) = y_0 \tag{2}$$

na intervalu $[a, b]$, možemo rješavati tako da podijelimo interval $[a, b]$ na n jednakih podintervala, označivši

$$h = \frac{b-a}{n}, \quad x_i = a + ih, \quad i = 0, \dots, n.$$

- Sada y_{i+1}, aproksimaciju rješenja u točki x_{i+1}, računamo iz y_i korištenjem aproksimacije oblika

$$y(x + h) \approx y(x) + h\Phi(x, y(x), h; f), \tag{3}$$

te dobivamo rekurziju:

$$y_{i+1} = y_i + h\Phi(x_i, y_i, h; f), \quad i = 0, \dots, n. \tag{4}$$

- Funkciju Φ nazivamo **funkcija prirasta**, a različit izbor te funkcije definira različite metode.
- Tako je npr. u Eulerovoj metodi

$$\Phi(x, y, h; f) = f(x, y).$$

- Metode oblika (4) zovemo **jednokoračne metode** (jer za aproksimaciju y_{i+1} koristimo samo vrijednost y_i u prethodnoj točki x_i).
- O odabiru funkcije Φ ovisi i točnost metode.
- Za očekivati je da ako izaberemo Φ tako da aproksimacija točnog rješenja $y(x + h)$ dana s (3) bude što točnija, da će točnija biti i aproksimacija y_i za $y(x_i)$ dana rekurzijom (4).

- Pogrešku aproksimacije (3):
$$\tau(x;h) = \Delta(x;h) - \Phi(x, y(x), h), \tag{5}$$
gdje je $y(x)$ točno rješenje diferencijalne jednadžbe (2) i
$$\Delta(x;h) = \frac{y(x+h) - y(x)}{h},$$
nazivamo **lokalna pogreška diskretizacije.**
- Za razumne jednokoračne metode zahtjevat ćemo da je
$$\lim_{h\to 0} \tau(x;h) = 0.$$
- U tom slučaju je
$$0 = \lim_{h\to 0} (\Delta(x;h) - \Phi(x, y(x), h)) = f(x,y) - \lim_{h\to 0} \Phi(x, y(x), h)$$
tj.
$$\lim_{h\to 0} \Phi(x, y(x), h) = f(x,y).$$

Definicija

Jednokoračnu metodu zovemo konzistentnom ako za svaki $f \in F_1(a, b)$, $x \in [a, b]$ i $y \in \mathbb{R}$ lokalna pogreška diskretizacije τ zadovoljava

$$\lim_{h \to 0} \tau(x; h) = 0.$$

Ukoliko je još $f \in F_p(a, b)$ i

$$\tau(x; h) = \mathcal{O}(h^p)$$

kažemo da je metoda reda p.

- Što je veći p metoda je točnija.

- Pod točnošću metode podrazumijevamo ponašanje pogreške
$$y(x_i) - y_i.$$
- Zbog jednostavnosti promatrat ćemo pogrešku u fiksiranoj točki b.
- Ako je jednokoračna metoda reda p, tada se može pokazati (teorem kasnije)
$$y(b) - y_n = \mathcal{O}(h^p).$$
tj. $\lim_{n\to\infty}(y(b) - y_n) = 0$ za sve $x \in [a, b]$ i $f \in F_1(a, b)$, te kažemo da je jednokoračna metoda **konvergentna.**
- Uočimo da je $h = (b - a)/n$ te da je y_n uvijek (za svaki n) aproksimacija za $y(b)$.

- Najpoznatije jednokoračne metode su **Runge–Kutta metode**.
- Kod njih je funkcija Φ oblika

$$\Phi(x, y, h) = \sum_{j=1}^{r} \omega_j k_j(x, y, h),$$

a k_j su zadani s

$$k_j(x, y, h) = f\left(x + c_j h, y + h \sum_{l=1}^{r} a_{jl} k_l(x, y, h, f)\right), \quad j=1,\ldots,r. \tag{6}$$

- Broj r zovemo broj stadija Runge–Kutta (RK) metode, i on označava koliko puta moramo računati funkciju f u svakom koraku.

- Različit izbor koeficijenata ω_j, c_j i a_{jl} definira različite metode.
- Ovi koeficijenti se najčešće biraju tako da red metode bude što je moguće veći.
    - Iz izraza (6) vidimo da se k_j nalazi na lijevoj i na desnoj strani jednadžbe, tj. zadan je implicitno te govorimo o **implicitnoj** Runge–Kutta metodi.
    - U praksi se najviše koriste metode gdje je $a_{jl} = 0$ za $l \geq j$. Tada k_j možemo izračunati preko $k_1, \ldots, k_{j-1}$, tj. funkcije k_j su zadane eksplicitno. Takve RK metode nazivamo **eksplicitnima**.
- Nadalje, obično se dodaje uvjet (teorem kasnije)

$$\sum_{l=1}^{r} a_{jl} = c_j.$$

- Odabrat ćemo koeficijente za RK metodu s dva stadija:

$$
\begin{aligned}
\Phi(x,y,h) &= \omega_1 k_1(x,y,h) + \omega_2 k_2(x,y,h),\\
k_1(x,y,h) &= f(x,y),\\
k_2(x,y,h) &= f(x+ah, y+ahk_1).
\end{aligned}
$$

- Razvojem rješenja diferencijalne jedadžbe u Taylorov red, i primjenom definicije lokalne pogreške diskretizacije dobivamo sljedeće uvjete na koeficijente:
  - Da bi metoda bila reda 1 koeficijente treba odabrati tako da je:
  $$1-\omega_1-\omega_2=0.$$
  - Da bi metoda bila reda 2 koeficijente treba odabrati tako da je još i:
  $$\frac{1}{2}-\omega_2 a=0$$

- Uvođenjem slobodnog koeficijenta t rješenje ove dvije jednadžbe možemo napisati u obliku:

$$\omega_2 = t \neq 0, \quad \omega_1 = 1 - t, \quad a = \frac{1}{2t}.$$

- t ne možemo odabrati tako da metoda bude reda 3.

Eulerova metoda $\omega_2 = t = 0$, metoda s jednim stadijem — metoda reda 1

Heunova metoda $t = \frac{1}{2}$ — metoda reda 2

$$\begin{aligned}\Phi &= \frac{1}{2}(k_1 + k_2),\\ k_1 &= f(x, y),\\ k_2 &= f(x + h, y + hk_1),\end{aligned}$$

modificirana Eulerova metoda $t = 1$ — metoda reda 2

$$\Phi = f\left(x + \frac{h}{2}, y + \frac{h}{2} f(x, y)\right).$$

- Najraširenije su metode sa četiri stadija.
- Odgovarajuće jednadžbe koje moraju zadovoljavati koeficijenti RK-4 metoda su:

$$\begin{aligned}
\omega_1 + \omega_2 + \omega_3 + \omega_4 &= 1, &(8)\\
\omega_2 c_2 + \omega_3 c_3 + \omega_4 c_4 &= \frac{1}{2}, &(9)\\
\omega_2 c_2^2 + \omega_3 c_3^2 + \omega_4 c_4^2 &= \frac{1}{3}, &(10)\\
\omega_3 c_2 a_{32} + \omega_4 (c_2 a_{42} + c_3 a_{43}) &= \frac{1}{6}, &(11)\\
\omega_2 c_2^3 + \omega_3 c_3^3 + \omega_4 c_4^3 &= \frac{1}{4}, &(12)\\
\omega_3 c_2^2 a_{32} + \omega_4 (c_2^2 a_{42} + c_3^2 a_{43}) &= \frac{1}{12}, &(13)\\
\omega_3 c_2 c_3 a_{32} + \omega_4 (c_2 a_{42} + c_3 a_{43}) c_4 &= \frac{1}{8}, &(14)\\
\omega_4 c_2 a_{32} a_{43} &= \frac{1}{24}, &(15)
\end{aligned}$$

gdje je

$$c_1 = 0, \qquad c_2 = a_{21},$$
$$c_3 = a_{31} + a_{32}, \qquad c_4 = a_{41} + a_{42} + a_{43}.$$

- Uvjet (8) treba biti zadovoljen da bi metoda bila reda 1,
- uvjet (9) za red 2,
- uvjeti (10)–(11) za red 3,
- dok za red 4 trebaju biti ispunjeni i uvjeti (12)–(15).

Ukupno imamo 10 koeficijenata i 8 jednadžbi ukoliko je metoda reda 4.

- Međutim, metoda s četiri stadija može postići najviše red četiri, tj. ne možemo dva stupnja slobode iz sustava jednadžbi iskoristiti da red metode podignemo na pet.

- Općenito vrijedi:
  - Za metode s 1, 2, 3 i 4 stadija najveći mogući red metode odgovara broju stadija.
  - Za metode s 5, 6 i 7 stadija najveći mogući red je 4, 5 i 6.
  - Za metode s 8 i više stadija najveći mogući red je barem za dva manji od broja stadija.
- To je razlog zašto su metode s 4 stadija najpopularnije (RK-4).
- Slijedi nekoliko primjera RK-4 metoda.

“Klasična” Runge–Kutta metoda (RK-4)

$$
\begin{aligned}
\Phi &= \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4),\\
k_1 &= f(x, y),\\
k_2 &= f\left(x + \frac{h}{2}, y + \frac{h}{2}k_1\right),\\
k_3 &= f\left(x + \frac{h}{2}, y + \frac{h}{2}k_2\right),\\
k_4 &= f\left(x + h, y + hk_3\right).
\end{aligned}
$$

3/8-ska metoda

$$
\begin{aligned}
\Phi &= \frac{1}{8}(k_1 + 3k_2 + 3k_3 + k_4),\\
k_1 &= f(x, y),\\
k_2 &= f\left(x + \frac{h}{3}, y + \frac{h}{3}k_1\right),\\
k_3 &= f\left(x + \frac{2}{3}h, y - \frac{h}{3}k_1 + hk_2\right),\\
k_4 &= f(x + h, y + h(k_1 - k_2 + k_3))
\end{aligned}
$$

Gillova metoda

$$
\begin{aligned}
\Phi &= \frac{1}{6}\Big(k_1 + (2-\sqrt{2})k_2 + (2+\sqrt{2})k_3 + k_4\Big),\\
k_1 &= f(x, y),\\
k_2 &= f\left(x + \frac{h}{2}, y + \frac{h}{2}k_1\right),\\
k_3 &= f\left(x + \frac{h}{2}, y + h\frac{\sqrt{2}-1}{2}k_1 + h\frac{2-\sqrt{2}}{2}k_2\right),\\
k_4 &= f\left(x + h, y - h\frac{\sqrt{2}}{2}k_2 + h\frac{2+\sqrt{2}}{2}k_3\right).
\end{aligned}
$$

Algoritam (Runge–Kutta metoda (RK-4))

y_0, a, b i n zadani;
funkcija f zadana;
$h = \frac{b-a}{n}$;
$x(1) = a$;
$y(1) = y_0$;
for $i = 1 : n$
$\quad x(i+1) = a + i * h$;
$\quad k1 = f(x(i), y(i))$;
$\quad k2 = f\left(x(i) + \frac{h}{2}, y(i) + \frac{h}{2}k_1\right)$;
$\quad k3 = f\left(x(i) + \frac{h}{2}, y(i) + \frac{h}{2}k_2\right)$;
$\quad k4 = f\left(x(i+1), y(i) + hk_3\right)$;
$\quad y(i+1) = y(i) + \frac{h}{6} * (k1 + 2 * k2 + 2 * k3 + k4)$;
end

Neka svojstva Runge–Kutta metoda

Teorem

Runge–Kutta metoda sa r stadija ima red konzistencije veći ili jednak 1 ako i samo ako je

$$\sum_{j=1}^{r} \omega_j = 1.$$

Teorem

Neka je RK metoda zadana s

$$y_{i+1} = y_i + h\sum_{j=1}^{s} \tilde{k}_j, \qquad \tilde{k}_j = f\left(x + \tilde{c}_j h, y + h\sum_{l=1}^{s} a_{jl}\tilde{k}_l\right)$$

reda konzistencije $\tilde{p}$, te neka je p red konzistencije metode

$$y_{i+1} = y_i + h\sum_{j=1}^{s} k_j, \qquad k_j = f\left(x + c_j h, y + h\sum_{l=1}^{s} a_{jl}k_l\right),$$

gdje je $c_j = \sum_{l=1}^{s} a_{jl}$. Tada je $p \geq \tilde{p}$.

Konvergencija jednokoračnih metoda

- Za fiksirani $x \in [a, b]$ definiramo korak
$$h_n = \frac{x - x_0}{n}$$
i **globalnu pogrešku diskretizacije**
$$e(x; h_n) = y_n - y(x).$$
- Sada za svaki $n \in \mathbb{N}$ vrijedi $x_n = x$, te možemo promatrati globalnu pogrešku diskretizacije kada $n \to \infty$.

Definicija

*Jednokoračna metoda je **konvergentna** ako*
$$\lim_{n\to\infty} e(x; h_n) = 0$$
za sve $x \in [a, b]$ i sve $f \in F_1(a, b)$.

Teorem

Za $x_0 \in [a, b]$, $y_0 \in \mathbb{R}$, *promatramo inicijalni problem*

$$y' = f(x, y), \quad y(x_0) = y_0,$$

koji ima jedinstveno rješenje $y(x)$. *Neka je funkcija* Φ *neprekidna na*

$$G = \{(x, y, h) \mid x \in [a, b],\ |y - y(x)| \le \gamma,\ |h| \le h_0\},$$

za $h_0 > 0$, $\gamma > 0$ *i neka postoje pozitivne konstante M i N takve da je*

$$|\Phi(x, y_1; h) - \Phi(x, y_2; h)| \le M|y_1 - y_2|$$

za sve $(x, y_i, h) \in G$, $i = 1, 2$, *i*

Teorem (nastavak)

$$|\tau(x;h)| = |\Delta(x;h) - \Phi(x,y(x);h)| \leq N|h|^p, \quad p > 0$$

za sve $x \in [a,b]$, $h \leq h_0$. *Tada postoji* $\bar{h}$, $0 < \bar{h} \leq h_0$, *takav da globalna pogreška diskretizacije zadovoljava*

$$|e(x;h_n)| \leq |h_n|^p N \frac{e^{M|x-x_0|} - 1}{M}$$

za sve $x \in [a,b]$ *i* $h_n = (x - x_0)/n$, $n = 1,2,\ldots$, *uz* $|h_n| \leq \bar{h}$. *Ako je* $\gamma = \infty$, *tada je* $\bar{h} = h_0$.

Zadaci

Zadatak

Napišite M-file funkciju `odj_euler()` *koja implementira Eulerovu metodu za rješavanje inicijalnog problema za ODJ. Funkcija neka ima ulazne parametre*

- *pokazivač na funkciju* f
- *rubove segmenta* a *i* b
- *početni uvjet* y_0
- *broj podintervala* n

i izlazne parametre

- *vektor* x *duljine* $n+1$ *s vrijednostima* x_i
- *vektor* y *duljine* $n+1$ *s vrijednostima* y_i

Zadatak

Svoju funkciju `odj_euler()` *primijenite na sljedeći inicijalni problem*

$$\begin{aligned} y'(x) &= -y(x) - 5e^{-x}\sin(5x), \quad x \in [0,3], \\ y(0) &= 1 \end{aligned}$$

pri čemu je $n = 30$.

- f *uzmite kao pokazivač na anonimnu funkciju.*
- *Nacrtajte graf vašeg aproksimativnog rješenja, tako da točke* (x_i, y_i) *označite sa zvjezdicama, a točke međusobno povežete isprekidanom linijom.*
- *Označite osi sa 'x' i 'y'.*

Zadatak (nastavak)

- *Na istoj slici nacrtajte i graf egzaktnog rješenja*

$$y(x) = e^{-x}\cos(5x),$$

pomoću naredbe za crtanje funkcija

```
fplot('exp(-x)*cos(5*x)',[0 3],'r-');
```

- *Napišite prigodnu legendu.*
- *Izračunajte maksimalnu grešku* $\max_i |y(x_i) - y_i|$.

Zadatak

Napišite M-file funkciju `odj_rk4()` *koja implementira Runge–Kutta metodu (RK-4) za rješavanje inicijalnog problema za ODJ. Funkcija neka ima ulazne parametre*

- *pokazivač na funkciju* f
- *rubove segmenta* a *i* b
- *početni uvjet* y_0
- *broj podintervala* n

i izlazne parametre

- *vektor* x *duljine* $n+1$ *s vrijednostima* x_i
- *vektor* y *duljine* $n+1$ *s vrijednostima* y_i

Zadatak

Svoju funkciju `odj_rk4()` *primijenite na prethodni primjer, na isti način prikažite aproksimativno i egzaktno rješenje, te izračunajte maksimalnu grešku.*

Runge–Kutta–Fehlbergove metode

- Kod prethodnih jednokoračnih metoda pretpostavili smo da je korak integracije h konstantan tijekom cijelog postupka rješavanja diferencijalne jednadžbe.
- Ali, h se može mijenjati u svakom koraku integracije, pa jednokoračnu metodu možemo pisati u obliku:

$$y_{i+1} = y_i + h_i \Phi(x_i, y_i, h_i).$$

- Želimo odrediti duljinu koraka h_i tako da bude postignuta neka unaprijed zadana točnost ε.

- Neka su s Φ i $\bar{\Phi}$ zadane dvije metode reda p i $p+1$. Tada računamo aproksimacije

$$
\begin{aligned}
y_{i+1} &= y_i + h_i \Phi(x_i, y_i, h_i), \\
\bar{y}_{i+1} &= y_i + h_i \bar{\Phi}(x_i, y_i, h_i).
\end{aligned}
$$

- Znamo da vrijedi:

$$
\begin{aligned}
y(x_i + h_i) &= y(x_i) + h_i \Phi(x_i, y(x_i), h_i) + C(x_i) h_i^{p+1} + \mathcal{O}(h_i^{p+2}), \\
y(x_i + h_i) &= y(x_i) + h_i \bar{\Phi}(x_i, y(x_i), h_i) + \mathcal{O}(h_i^{p+2}).
\end{aligned}
$$

- Cilj je da pogreška u i-tom koraku bude manja od ε.
- Stoga ćemo pretpostaviti da je aproksimacija y_i za $y(x_i)$ točna za male h, tj. $y_i \approx y(x_i)$.

- Iz prve dvije jednakosti oduzimanjem slijedi

$$\bar{y}_{i+1} - y_{i+1} = h_i[\bar{\Phi}(x_i, y_i, h_i) - \Phi(x_i, y_i, h_i)],$$

- a iz druge dvije

$$h_i[\bar{\Phi}(x_i, y_i, h_i) - \Phi(x_i, y_i, h_i)] \approx C(x_i)h_i^{p+1} + \mathcal{O}(h_i^{p+2}).$$

- Kombiniranjem prethodnih izraza dobit ćemo

$$\begin{aligned} \bar{y}_{i+1} - y_{i+1} &\approx C(x_i)h_i^{p+1} + \mathcal{O}(h_i^{p+2}) \\ y(x_{i+1}) - y_{i+1} &\approx C(x_i)h_i^{p+1} + \mathcal{O}(h_i^{p+2}) \end{aligned}$$

- Zanemarivanjem viših članova u razvoju pogreške, dobivamo

$$\bar{y}_{i+1} - y_{i+1} \approx C(x_i)h_i^{p+1}$$

i

$$\bar{y}_i - y_i \approx C(x_{i-1})h_{i-1}^{p+1}.$$

- Uz pretpostavku da se član $C(x)$ u pogrešci ne mijenja brzo, tj. $C(x_i) \approx C(x_{i-1})$, uvjet da pogreška u i-tom koraku bude manja od ε sada glasi:

$$\varepsilon \geq |y(x_{i+1}) - y_{i+1}| \approx |C(x_i) h_i^{p+1}| \approx |C(x_{i-1}) h_i^{p+1}| \approx \left| \frac{\bar{y}_i - y_i}{h_{i-1}^{p+1}} \right| h_i^{p+1}$$

  odnosno

$$h_i^{p+1} \leq h_{i-1}^{p+1} \frac{\varepsilon}{|\bar{y}_i - y_i|}.$$

- Iz ovoga slijedi da za novi korak trebamo izabrati

$$h_i = h_{i-1} \sqrt[p+1]{\frac{\varepsilon}{|\bar{y}_i - y_i|}}.$$

- Ukoliko je prethodni korak bio uspješan, tada je zadovoljeno

$$|\bar{y}_i - y_i| \leq \varepsilon,$$

  te je stoga $h_i \geq h_{i-1}$.

- Ako prethodna nejednakost ne vrijedi, $(i-1)$-vi korak treba ponoviti uz manji h_{i-1}.
- Mnogi iz prakse preporučaju izbor koraka

$$h_i = \alpha h_{i-1} \sqrt[p+1]{\frac{\varepsilon}{|\bar{y}_i - y_i|}}.$$

gdje je α pogodno odabran korigirajući faktor, koji služi da ispravi pogrešku nastalu odbacivanjem viših članova u ocjeni pogreške.
- Obično je $\alpha = 0.9$.
- Čim izračunamo h_i najbolje je odmah provjeriti da li će biti ispunjeno $|\bar{y}_{i+1} - y_{i+1}| \leq \varepsilon$, i ukoliko to nije ispunjeno izračunati novi h_i^{novi} iz starog $h_i^{stari} = h_i$

$$h_i^{novi} = \alpha h_i^{stari} \sqrt[p+1]{\frac{\varepsilon}{|\bar{y}_{i+1} - y_{i+1}|}}.$$

na isti način kao što se h_i računa iz h_{i-1}.

- Prikazani izbor koraka vrijedi za bilo koji par jednokoračnih metoda reda p i $p+1$.
- Primjena Runge–Kutta metoda zahtijevala bi jednu metodu sa s stadija i jednu sa $s+1$ stadija, što općenito znači da bismo u svakom koraku funkciju f iz diferencijalne jednadžbe trebali računati $2s+1$ puta.
- Postupak se može pojednostavniti ako prvih s stadija $k_1, \ldots, k_s, k_{s+1}$ korištenih za računanje funkcije prirasta $\bar{\Phi}$ iskoristimo za računanje funkcije Φ:

$$\Phi(x, y, h) = \sum_{i=1}^{s} \omega_i k_i(x, y, h),$$

$$\bar{\Phi}(x, y, h) = \sum_{i=1}^{s+1} \bar{\omega}_i k_i(x, y, h).$$

- Sada u svakom koraku funkciju f računamo $s+1$ puta.

- Ovu ideju ćemo ilustrirati na paru Runge–Kutta metoda reda 2 i 3.
- Promatramo metode s 3 i 4 stadija:

$$\Phi(x, y, h) = \sum_{i=1}^{3} \omega_i k_i(x, y, h),$$

$$\bar{\Phi}(x, y, h) = \sum_{i=1}^{4} \bar{\omega}_i k_i(x, y, h).$$

- Da bi metoda definirana s $\bar{\Phi}$ bila reda 3 trebaju biti zadovoljeni uvjeti (8)–(11), što je ukupno 4 jednadžbe i 10 koeficijenata.
- Nadalje, metoda reda 2 s 3 stadija ima 3 dodatna koeficijenta (ω_1, ω_2 i ω_3) i treba zadovoljavati 2 dodatna uvjeta (8)–(9).

- Sveukupno, 13 koeficijenata u ovom paru metoda treba zadovoljavati 6 uvjeta.
- Preostalih 7 stupnjeva slobode iskoristit ćemo za smanjivanje broja računanja funkcije f.
- Zahtijevat ćemo da $k_4(x_i, y_i, h_i, f)$, zadnji stadij iz računanja $\bar{\Phi}$ u i-tom koraku, iskoristimo kao $k_1(x_{i+1}, y_{i+1}, h_{i+1}, f)$, prvi stadij u $(i+1)$-om koraku:

$$\begin{aligned} f(x_i + c_4 h_i, y_i + h_i(a_{41}k_1 + a_{42}k_2 + a_{43}k_3) &= f(x_{i+1}, y_{i+1}) \\ &= f(x_i + h_i, y_i + h_i\Phi(x_i, y_i, h_i, f)) \\ &= f(x_i + h_i, y_i + h_i(\omega_1 k_1 + \omega_2 k_2 + \omega_3 k_3)) \end{aligned}$$

- Odavde slijede dodatna 3 uvjeta:

$$a_{41} = \omega_1, \quad a_{42} = \omega_2, \quad a_{43} = \omega_3.$$

- Uvjet $c_4 = 1$ automatski je ispunjen zbog

$$\omega_1 + \omega_2 + \omega_3 = 1.$$

- Jedno od rješenja ovog sustava jednadžbi je prikazano u sljedećoj tablici.

i	c_i	a_{ij}: $j=1$	$j=2$	$j=3$	ω_i	$\bar{\omega}_i$
1	0				$\frac{214}{891}$	$\frac{533}{2106}$
2	$\frac{1}{4}$	$\frac{1}{4}$			$\frac{1}{33}$	0
3	$\frac{27}{40}$	$-\frac{189}{800}$	$\frac{729}{800}$		$\frac{650}{891}$	$\frac{800}{1053}$
4	1	$\frac{214}{891}$	$\frac{1}{33}$	$\frac{650}{891}$		$-\frac{1}{78}$

Zadaci

Zadatak

Napišite M-file funkciju `odj_rk23()` *koja implementira Runge–Kutta–Fehlbergovu metodu za rješavanje inicijalnog problema za ODJ. Metoda je kombinacija Runge–Kutta metoda reda 2 i 3. Funkcija neka ima ulazne parametre*

- *pokazivač na funkciju* f
- *rubove segmenta* a *i* b
- *početni uvjet* y_0
- *toleranciju na grešku* tol

i izlazne parametre

- *vektor* x *duljine* $n+1$ *s vrijednostima* x_i
- *vektor* y *duljine* $n+1$ *s vrijednostima* y_i

gdje je n *broj podintervala.*

Zadatak (nastavak)

- *Početni korak je* $h_0 = b - a$.
- *Za svaku novu iteraciju najprije izračunajte* h_i, *i provjerite da li će biti* $|\bar{y}_{i+1} - y_{i+1}| \leq tol$. *Ukoliko to nije ispunjeno novi* h_i *računajte po formuli*

$$h_i^{novi} = 0.9 h_i^{stari} \sqrt[p+1]{\frac{tol}{|y_{i+1} - \bar{y}_{i+1}|}}.$$

Analogno računajte i početni h_{i+1}.
- *Obratite pozornost na slučaj kada je* $x_i + h_i > b$. *U tom slučaju uzmite* $h_i = b - x_i$ *i* $x_{i+1} = b$.

Zadatak

- *Svoju funkciju* `odj_rk23()` *primijenite na problem*

$$y'(x) = \begin{cases} 1 - y(x) & x \in [0, \pi] \\ -5y(x) & x \in \langle \pi, 4] \end{cases}, \quad y(0) = 0$$

  za koji je točno rješenje

$$y(x) = \begin{cases} 1 - e^{-x} & x \in [0, \pi] \\ (1 - e^{-\pi})e^{-5(x-\pi)} & x \in \langle \pi, 4] \end{cases}.$$

- *Uzmite tol* $= 10^{-5}$, *prikažite aproksimativno i egzaktno rješenje, te na x-osi označite korake koje je metoda napravila. Izračunajte maksimalnu grešku.*
- *Usporedite sa rješenjem dobivenim samo Runge–Kutta metodom reda 2 iz Runge–Kutta–Fehlberg metode sa istim brojem koraka.*

Implicitna trapezna metoda

- Jednokoračne metode možemo shvatiti kao primjenu kvadraturnih formula na integraciju ODJ

$$y'(x) = f(x, y(x)), \qquad y(x_0) = y_0.$$

- Integracijom prethodne jednadžbe slijedi

$$y(x_{i+1}) - y(x_i) = \int_{x_i}^{x_{i+1}} f(x, y(x))dx = \mathcal{I}_i.$$

- Dakle, vrijednost $y(x_{i+1})$ možemo izračunati iz stare vrijednosti $y(x_i)$ ako znamo izračunati integral $\mathcal{I}_i$.
- Korištenjem kvadraturnih formula dobivamo:
  - *formula lijevog ruba* $\longrightarrow$ **Eulerova metoda**

$$\mathcal{I}_i \approx hf(x_i, y(x_i))$$

  - *formula desnog ruba* $\longrightarrow$ **implicitna Eulerova metoda**

$$\mathcal{I}_i \approx hf(x_{i+1}, y(x_{i+1}))$$

- *formula srednje točke* $\longrightarrow$ **modificirana Eulerova metoda**

$$\mathcal{I}_i \approx hf\left(x_i + \frac{h}{2}, y\left(x_i + \frac{h}{2}\right)\right),$$

pri čemu se koristi aproksimacija

$$y(x_i + h/2) \approx y(x_i) + \frac{h}{2}y'(x_i) = y(x_i) + \frac{h}{2}f(x_i, y(x_i)).$$

- *trapezna formula* $\longrightarrow$ **Imlicitna trapezna metoda** ili **Crank–Nicolsonova metoda**

$$\mathcal{I}_i \approx \frac{h}{2}(f(x_i, y(x_i)) + f(x_{i+1}, y(x_{i+1})))$$

- Dakle, primjenom trapezne formule na integraciju ODJ dobit ćemo metodu oblika

$$y_{i+1} = y_i + \frac{h}{2}(f(x_i, y_i) + f(x_{i+1}, y_{i+1})).$$

- Budući da se y_{i+1} javlja i na lijevoj i na desnoj strani jednadžbe, radi se o implicitnoj metodi.
- U svakoj iteraciji rješava se gornji problem nekom od metoda za numeričko rješavanje nelinearnih jednadžbi, npr. Newtonovom metodom sa fiksnim brojem iteracija.
- Implicitna trapezna metoda je reda 2, budući da je takva točnost i trapezne kvadraturne formule.

Zašto koristimo implicitne metode, kada kod njih u svakom koraku moramo rješavati nelinearnu jednadžbu?

- Kod Runge–Kutta–Fehlbergove metode vidjeli smo da red konzistencije utječe na izbor koraka h_i.
- S druge strane, kod rješavanja npr. inicijalnih problema oblika $y' = \lambda y$, $y(x_0) = y_0$ znamo da je egzaktno rješenje oblika $y = y_0 e^{\lambda x}$.
- Za $\lambda < 0$ za rješenje vrijedi $\lim_{x\to\infty} y(x) = 0$, što bi trebalo vrijediti i za aproksimaciju rješenja y_i.
- Za standardne eksplicitne metode sa fiksnim korakom h to nije uvijek tako.
- Produkt λh mora biti u određenom intervalu da bi to vrijedilo i za y_i, a izvan tog intervala se pojavljuju rastuće oscilacije u aproksimativnom rješenju.

Definicija

Interval apsolutne stabilnosti numeričke metode je interval produkta λh za koji aproksimacija y_i rješenja $y(x_i) = y_0 e^{\lambda x_i}$ inicijalnog problema $y' = \lambda y$, $y(x_0) = y_0$ teži ka nuli kada $i \to \infty$.

Intervali apsolutne stabilnosti za metode koje smo do sada radili su:

Euler. m.	RK-1	RK-2	RK-3	RK-4	Impl. trap.
$\langle -2,0 \rangle$	$\langle -2,0 \rangle$	$\langle -2,0 \rangle$	$\langle -2.51,0 \rangle$	$\langle -2.78,0 \rangle$	$\langle -\infty,0 \rangle$

prema tome implicitna trapezna metoda je **apsolutno stabilna.**

Implicitne metode su puno efikasnije za rješavanje **krutih ("stiff") jednadžbi.**

- Primjenom numeričke metode na krutu jednadžbu veći utjecaj na veličinu koraka h_i ima interval apsolutne stabilnosti nego uvjet na održavanje male lokalne pogreške diskretizacije.
- Takve jednadžbe se teško rješavaju pomoću eksplicitnih Runge–Kutta metoda jer zahtijevaju puno vrlo sitnih koraka, dok za implicitnu trapeznu metodu to nije slučaj.

Zadaci

Zadatak

Napišite M-file funkciju `odj_impl_trapez()` *koja implementira implicitnu trapeznu metodu za rješavanje inicijalnog problema za ODJ. Funkcija neka ima ulazne parametre*

- *pokazivač na funkciju f*
- *rubove segmenta a i b*
- *početni uvjet y_0*
- *broj podintervala n*

i izlazne parametre

- *vektor x duljine $n+1$ s vrijednostima x_i*
- *vektor y duljine $n+1$ s vrijednostima y_i*

Zadatak (nastavak)

Svoju funkciju implementirajte na sljedeći način.

- *U svakom koraku metode definirajte funkciju $g(z)$ čija nultočka je y_{i+1} pomoću pokazivaća na anonimnu funkciju.*
- *Budući da nam za Newtonovu metodu treba i $g'(z)$, koristit ćemo MATLAB-ove simboličke izraze:*

```
w=sym('w');
g=@(z) ...;
gs=g(w);
dgs=diff(gs);
dg=@(z) double(subs(dgs,z));
```

Zadatak (nastavak)

- *Objašnjenja:*

`sym()` `w=sym('w')`	*kreira simboličke brojeve, varijable i objekte* `w` *je simbolička varijabla*
`g(w)`	*simbolički izraz*
`diff()`	*derivira simbolički izraz*
`subs()`	*zamjena simboličke varijable sa novom vrjednosti*
`double()`	*konverzija simboličkog izraza u numerički oblik*

- *Napišite pomoćnu M-file funkciju* `odj_newton` *koja implementira Newtonovu metodu za rješavanje nelinearne jedadžbe.*
- `odj_newton` *neka ima ulazne parametre*
    - *pokazivač na funkciju f*
    - *pokazivač na derivaciju funkcije df*
    - *početnu iteraciju* z_0
    - *broj iteracija k*

Zadatak (nastavak)

- *Funkcija* `odj_newton` *neka vraća aproksimaciju rješenja.*
- *U svakom koraku implicitne trapezne metode računajte* y_{i+1} *pomoću* `odj_newton`, *i uzmite da je* $z_0 = y_i$ *ili* $z_0 = y_i + hf(x_i, y_i)$ *i* $k = 5$.

Zadatak

Svoju funkciju `odj_impl_trapez()` *primijenite na problem*

$$\begin{aligned} y'(x) &= -100(y(x) - \cos x) - \sin x, \quad x \in [0,1], \\ y(0) &= 1, \end{aligned}$$

za koji je točno rješenje

$$y(x) = \cos x.$$

- *Za $h = 1/n$, $n = 20, \ldots, 50$ računajte vrijednost aproksimacije u zadnjoj točci (y_{n+1}) pomoću implicitne trapezne metode i Runge–Kutta metode reda 2 iz Runge–Kutta–Fehlberg metode. Usporedite greške u odnosu na točno rješenje.*

Linearne višekoračne metode

- Kod jednokoračnih metoda je za aproksimaciju y_{i+1} u točki x_{i+1} bilo potrebno poznavanje samo aproksimacije y_i u točki x_i.
- Promatrajmo ponovno diferencijalnu jednadžbu

$$y' = f(x, y), \quad y(x_0) = y_0.$$

- Integracijom, te primjenom formule srednje točke za aproksimaciju integrala, slijedi da je

$$\begin{aligned} y(x_{i+1}) - y(x_{i-1}) = \int_{x_{i-1}}^{x_{i+1}} y'(x)\, dx &= \int_{x_{i-1}}^{x_{i+1}} f(x, y(x))\, dx \\ &\approx 2hf(x_i, y(x_i)). \end{aligned}$$

- Gornja formula vodi na rekurzivno definiranu aproksimaciju

$$y_{i+1} = y_{i-1} + 2hf(x_i, y_i).$$

- U ovoj metodi za određivanje vrijednosti y_{i+1} trebamo poznavati prethodne vrijednosti y_i i y_{i-1} — **dvokoračna metoda**
- Aproksimacija y_{i+1} zadana je eksplicitno izrazom na desnoj strani — **eksplicitna metoda**
- Prethodno opisana metoda zove se **metoda preskoka** i reda je 2.

- Ako umjesto formule srednje točke pri računanju integrala primijenimo Simpsonovu formulu, dobivamo drugu aproksimaciju:

$$
\begin{aligned}
y(x_{i+1}) - y(x_{i-1}) &= \int_{x_{i-1}}^{x_{i+1}} f(x, y(x))\, dx \\
&\approx \frac{h}{3}\,[f(x_{i-1}, y(x_{i-1})) + 4f(x_i, y(x_i)) + \\
&\quad + f(x_{i+1}, y(x_{i+1}))],
\end{aligned}
$$

- Prethodna formula vodi na metodu

$$
y_{i+1} = y_i + \frac{h}{3}\,[f(x_{i-1}, y_{i-1}) + 4f(x_i, y_i) + f(x_{i+1}, y_{i+1})].
$$

- Ovdje se y_{i+1} javlja i na lijevoj strani i na desnoj strani kao argument, općenito nelinearne, funkcije f.
- Dakle, y_{i+1} je zadan implicitno — **implicitna dvokoračna metoda**

Napomena

- *Uočimo da gornjim formulama ne možemo odrediti y_1, pa za njegovo određivanje treba upotrebiti jednu od jednokoračnih metoda.*
- *Višekoračne metode se koriste kada je funkcija f vrlo "skupa" za računanje, jer se u iteracijama koriste već izračunate vrijednosti $f(x_j, y_j)$ a ne neke nove vrijednosti kao kod Runge–Kutta metoda.*

- Općenito, linearne višekoračne metode su oblika

$$\sum_{j=0}^{r} \alpha_j y_{i+1-j} = h \sum_{j=0}^{r} \beta_j f_{i+1-j},$$

gdje je $f_k = f(x_k, y_k)$, $\alpha_0 \neq 0$ i $|\alpha_r| + |\beta_r| \neq 0$.
- Ovu metodu zovemo ***r*-koračna metoda**:
  - za $\beta_0 = 0$ metoda je još i **eksplicitna**,
  - a za $\beta_0 \neq 0$ metoda je **implicitna**.
- Prikaz višekoračne metode pomoću koeficijenata α_j i β_j nije jedinstven jer npr. koeficijenti $2\alpha_0, \ldots, 2\alpha_r$ i $2\beta_0, \ldots, 2\beta_r$ definiraju istu metodu.
- Često se koristi normalizacija $\alpha_0 = 1$ te je zapis metode oblika:

$$y_{i+1} + \sum_{j=1}^{r} \alpha_j y_{i+1-j} = h\beta_0 f(x_{i+1}, y_{i+1}) + h \sum_{j=1}^{r} \beta_j f_{i+1-j}.$$

- Primjenom različitih integracijskih metoda možemo dobiti cijeli niz višekoračnih metoda.
- Integracijom jednadžbe $y'(x) = f(x, y(x))$ na nekom zadanom intervalu $[x_{p-j}, x_{p+k}]$ dobivamo

$$y(x_{p+k}) - y(x_{p-j}) = \int_{x_{p-j}}^{x_{p+k}} f(t, y(t))\, dt.$$

- Ukoliko podintegralnu funkciju $f(t, y(t))$ zamijenimo interpolacijskim polinomom P_q stupnja q koji interpolira $f(t, y(t))$ u točkama x_i, tj. ako je

$$P_q(x_i) = y'(x_i) = f(x_i, y(x_i)), \quad i = p, p-1, \ldots, p-q,$$

korištenjem interpolacijskog polinoma u Lagrangeovoj formi

$$P_q(x) = \sum_{i=0}^{q} f(x_{p-i}, y(x_{p-i}))\ell_i(x), \quad \ell_i(x) = \prod_{\substack{l=0 \\ l\neq i}}^{q} \frac{x - x_{p-l}}{x_{p-i} - x_{p-l}},$$

dobivamo

$$\begin{aligned} y(x_{p+k}) - y(x_{p-j}) \approx & \sum_{i=0}^{q} f(x_{p-i}, y(x_{p-i})) \int_{x_{p-j}}^{x_{p+k}} \ell_i(t)\, dt \\ = & h \sum_{i=0}^{q} \beta_{qi} f(x_{p-i}, y(x_{p-i})), \end{aligned}$$

gdje smo s β_{qi} označili

$$\beta_{qi} = \frac{1}{h} \int_{x_{p-j}}^{x_{p+k}} \ell_i(t)\, dt = \int_{-j}^{k} \prod_{\substack{l=0 \\ l \neq i}}^{q} \frac{s+l}{-i+l}\, ds, \quad i = 0, \ldots, q.$$

- Zamjenom vrijednosti $y(x_{p-i})$ aproksimacijama y_{p-i} dobivamo višekoračnu metodu oblika

$$y_{p+k} = y_{p-j} + h \sum_{i=0}^{q} \beta_{qi} f_{p-i}.$$

- U **Adams–Bashforthovoj metodi** je $k = 1$ i $j = 0$, pa je ona eksplicitna, $(q+1)$-koračna i glasi:

$$y_{p+1} = y_p + h(\beta_{q0} f_p + \beta_{q1} f_{p-1} + \cdots + \beta_{qq} f_{p-q}).$$

- Sljedeća tablica prikazuje koeficijente β_{qi} za ovu metodu.

	i				
β_{qi}	0	1	2	3	4
β_{0i}	1				
$2\beta_{1i}$	3	-1			
$12\beta_{2i}$	23	-16	5		
$24\beta_{3i}$	55	-59	37	-9	
$720\beta_{4i}$	1901	-2774	2616	-1274	251

- Izborom $k = 0$ i $j = 1$ dobivamo **Adams–Moultonovu metodu**, koja je implicitna, q-koračna i glasi:

$$y_{p+1} = y_p + h(\beta_{q0} f_{p+1} + \beta_{q1} f_p + \cdots + \beta_{qq} f_{p+1-q}).$$

- Koeficijenti su im prikazani u sljedećoj tablici.

β_{qi}	i = 0	1	2	3	4
β_{0i}	1				
$2\beta_{1i}$	1	1			
$12\beta_{2i}$	5	8	-1		
$24\beta_{3i}$	9	19	-5	1	
$720\beta_{4i}$	251	646	-264	106	-19

- Sada ćemo definirati **lokalnu pogrešku diskretizacije** za višekoračnu metodu:

$$\tau(x; h) = \frac{1}{h}\sum_{j=0}^{r} \alpha_j y(x - jh) - \sum_{j=0}^{r} \beta_j y'(x - jh),$$

gdje je y egzaktno rješenje ODJ.

Definicija

Višekoračnu metodu zovemo konzistentnom ako za svaki $f \in F_1(a, b)$, $x \in [a, b]$ i $y \in \mathbb{R}$ lokalna pogreška diskretizacije τ zadovoljava

$$\lim_{h\to 0} \tau(x; h) = 0.$$

Ukoliko je još $f \in F_p(a, b)$ i

$$\tau(x; h) = \mathcal{O}(h^p)$$

Primjer

- *Red $(r+1)$-koračne Adams–Bashforthove metode jednak je $r+1$.*
- *Red r-koračne Adams–Moultonove metode za jedan je veći od broja koraka i iznosi $r+1$.*

- Koeficijenti α_j i β_j za dani r određuju se tako da red metode bude što je moguće veći.
- Kod jednokoračnih metoda:

konzistentnost $\Longrightarrow$ konvergencija

kod višekoračnih metoda:

konzistentnost + stabilnost $\Longrightarrow$ konvergencija

Konvergencija linearnih višekoračnih metoda

Višekoračna metoda

$$\sum_{j=0}^{r} \alpha_j y_{i+1-j} = h \sum_{j=0}^{r} \beta_j f(x_{i+1-j}, y_{i+1-j})$$

definira dva polinoma

$$\rho(z) = \sum_{j=0}^{r} \alpha_j z^{r-j} \qquad \text{i} \qquad \sigma(z) = \sum_{j=0}^{r} \beta_j z^{r-j}.$$

Definicija

Za višekoračnu metodu kažemo da je **stabilna** *ako nultočke z_j polinoma $\rho(z)$ zadovoljavaju*

1. *Sve nultočke su po apsolutnoj vrijednosti manje od 1 ($|z_j| \leq 1$).*
2. *Ako je $|z_j| = 1$ tada je z_j jednostruka nultočka ($\rho'(z_j) \neq 0$).*

Zajedno uvjete 1 i 2 zovemo **uvjet stabilnosti***.*

Teorem

Linearna višekoračna metoda je konvergentna ako i samo ako je konzistentna i stabilna.

Napomena

Može se pokazati da stabilna r-koračna metoda ima red

$$p \leq \begin{cases} r+1, & \textit{ako je } r \textit{ neparan,} \\ r+2, & \textit{ako je } r \textit{ paran.} \end{cases}$$

- Konvergencija metode ponovo ovisi o ponašanju **globalne pogreške diskretizacije**:

  $$e(x; h) = y_n - y(x),$$

  gdje je $x \in \langle a, b \rangle$, $h = h_n = (x - a)/n$.
- Globalna pogreška diskretizacije ovisi o
  1. lokalnoj pogrešci diskretizacije
  2. r izračunatih početnih vrijednosti $y_0, \ldots, y_{r-1}$.

- Da bismo startali r-koračnu metodu, prvo je potrebno izračunati r početnih vrijednosti $y_0, \ldots, y_{r-1}$.
- Dok y_0 možemo odrediti iz početnog uvjeta diferencijalne jednadžbe, ostale vrijednosti moramo odrediti nekom drugom, najčešće jednokoračnom, metodom.
- Pri njihovom određivanju javit će se pogreška ε_i:

$$y(x_i) = y_i + \varepsilon_i, \quad i = 0, \ldots, r-1.$$

- Ova pogreška ne ovisi o promatranoj višekoračnoj metodi, već o načinu na koji određujemo početne vrijednosti.
- Očito je, da ako želimo da globalna pogreška diskretizacije teži nuli kada $n \to \infty$, pogreške početnih vrijednosti trebaju zadovoljavati

$$\lim_{n\to\infty} \varepsilon_i = 0, \quad i = 0, \ldots, r-1.$$

Definicija

Višekoračnu metodu zovemo konvergentnom ako je

$$\lim_{n\to\infty} e(x; h_n) = 0, \quad h_n = \frac{x-a}{n}, \quad n = 1, 2, \ldots$$

za sve $x \in [a, b]$, *sve* $f \in F_1(a, b)$ *i sve* y_i, $i = 0, \ldots, r-1$ *za koje je*

$$\lim_{n\to\infty} (y(x_i) - y_i) = 0, \quad i = 0, \ldots, r-1.$$

Korolar

Neka je linearna višekoračna metoda stabilna i konzistentna reda p, te $f \in F_p(a, b)$. Tada globalna pogreška diskretizacije zadovoljava

$$e(x; h_n) = \mathcal{O}(h_n^p)$$

za sve $h_n = (x - x_0)/n$ čim pogreške ε_i zadovoljavaju

$$|\varepsilon_i| \leq \varepsilon(h_n), \quad i = 0, \ldots, r-1$$

uz $\varepsilon(h_n) = \mathcal{O}(h_n^p)$.

- Ovaj korolar ujedno kazuje koju metodu moramo izabrati za određivanje početnih vrijednosti $y_0, \ldots, y_{r-1}$.
- Iz teorema o konvergenciji jednokoračnih metoda slijedi da možemo izabrati jednokoračnu metodu reda p da bi se pogreška višekoračne metode ponašala kao $\mathcal{O}(h^p)$.

Prediktor–korektor par

- Dosad je ostalo otvoreno pitanje kako izračunati y_{i+1} u implicitnoj metodi (korektor)

$$y_{i+1} + \sum_{j=1}^{k} \alpha_j^* y_{i+1-j} = \beta_0^* h f(x_{i+1}, y_{i+1}) + h \sum_{j=1}^{k} \beta_j^* f_{i+1-j}.$$

- Ako označimo

$$c = -\sum_{j=1}^{k} \alpha_j^* y_{i+1-j} + h \sum_{j=1}^{k} \beta_j^* f_{i+1-j}, \quad \varphi(y) = \beta_0^* h f(x_{i+1}, y) + c,$$

y_{i+1} je rješenje nelinearne jednadžbe $y = \varphi(y)$.

- Budući da možemo izabrati dovoljno malen korak integracije h takav da je nejednakost

$$|\varphi'(y)| = h|\beta_0^*| \left| \frac{\partial f(x_{i+1}, y)}{\partial y} \right| < 1$$

zadovoljena,

- slijedi da jednostavne iteracije

$$y^{[m+1]} = \varphi(y^{[m]}), \quad m = 0, 1, 2, \ldots$$

  konvergiraju prema rješenju jednadžbe.
- Za odabir početne aproksimacije $y^{[0]}$ koristi se neka od eksplicitnih metoda (prediktor)

$$y_{i+1} + \sum_{j=1}^{\bar{k}} \alpha_j y_{i+1-j} = h \sum_{j=1}^{\bar{k}} \beta_j f_{i+1-j}.$$

- Sada možemo zapisati cijeli algoritam:

$$
\begin{aligned}
y_{i+1}^{[0]} &= -\sum_{j=1}^{\bar{k}} \alpha_j y_{i+1-j} + h \sum_{j=1}^{\bar{k}} \beta_j f_{i+1-j}, \\
y_{i+1}^{[m+1]} &= \beta_0^* h f(x_{i+1}, y_{i+1}^{[m]}) - \sum_{j=1}^{k} \alpha_j^* y_{i+1-j} + h \sum_{j=1}^{k} \beta_j^* f_{i+1-j}, \\
& \qquad m = 0, \ldots, M-1, \\
y_{i+1} &= y_{i+1}^{[M]}.
\end{aligned}
$$

- Broj iteracija (M) može biti unaprijed zadan ili se iteracije provode dok se jednadžba ne riješi do na neku unaprijed zadanu točnost.
- U primjeni, broj iteracije nije velik, uvijek se radi o nekoliko iteracija.

- Kako odabrati korektor–prediktor par?
- Točnost metode definirana je s točnošću korektora, tj. implicitne metode.
- Uobičajeno je da se za prediktor–korektor par uzimaju eksplicitna i implicitna metoda istoga reda.
- Često korišten par je k-koračna Adams–Bashforthova metoda kao prediktor i $(k-1)$-koračna Adams–Moultonova metoda kao korektor.
- Uz ovakav odabir prediktor-korektor para govorimo o Adams–Bashforth–Moultonovim metodama.

Zadaci

Zadatak

Napišite M-file funkciju `odj_pred_kor()` *koja implementira par prediktor–korektor za rješavanje inicijalnog problema za ODJ. Funkcija neka ima ulazne parametre*

- *pokazivač na funkciju* f
- *rubove segmenta* a *i* b
- *početni uvjet* y_0
- *broj podintervala* n
- *broj iteracija korektora* m

i izlazne parametre

- *vektor* x *duljine* $n+1$ *s vrijednostima* x_i
- *vektor* y *duljine* $n+1$ *s vrijednostima* y_i

Zadatak (nastavak)

Svoju funkciju implementirajte na sljedeći način.

- *Za prediktor uzmite 4-koračnu Adams–Bashforthovu metodu.*
- *Za korektor uzmite 3-koračnu Adams–Moultonovu metodu.*
- *Za određivanje y_1, y_2 i y_3 iskoristite Runge–Kutta metodu RK-4.*

Zadatak

Svoju funkciju `odj_pred_kor()` *primijenite na prethodni primjer sa problemom*

$$\begin{aligned} y'(x) &= -y(x) - 5e^{-x}\sin(5x), \quad x \in [0,3], \\ y(0) &= 1 \end{aligned}$$

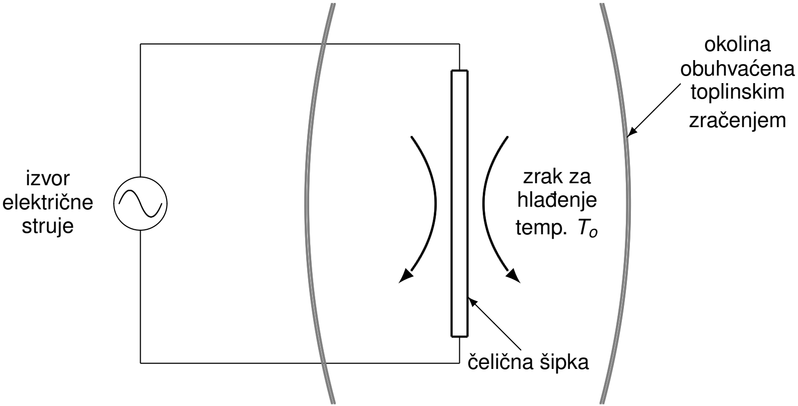
Na isti način prikažite aproksimativno i egzaktno rješenje, te izračunajte maksimalnu grešku.

- *Varirajte parametre n i m.*

Primjeri iz primjene: Termička obrada metalne šipke

Primjer

- *Čvrstoća metala kontrolira se njegovim kemijskim sastavom i postupcima mehaničkog oblikovanja koji su korišteni za njegovo dobivanje.*
- *Nakon što se rastopljeni metal skrutio, ali prije nego što se potpuno ohladio, oblikuje se u poluge, ploče ili šipke.*
- *Zatim, metal se ostavlja da se ohladi i nakon toga se podvrgava pažljivo kontroliranim promjenama temperature u procesu koji se zove* ***termička obrada****.*
- *Jednostavan model termičke obrade je dan nelinearnom ODJ 1. reda za temperaturu metala kao funkcije o vremenu.*



Slika: Termička obrada čelične šipke.

Primjer (nastavak)

- *Čelik se oblikuje u duge šipke, koje se zagrijavaju kako bi se oslobodile naprezanja nastalih za vrijeme postupka oblikovanja.*
- *Šipke se zagrijavaju propuštanjem električne struje kroz njih.*
- *Nakon što šipke dostignu određenu temperaturu, struja se isključuje i uključuju se ventilatori koji ih hlade.*
- *U obje faze grijanja i hlađenja šipke razmjenjuju toplinu s okolinom pomoću konvekcije i zračenja.*
- *Prijenos topline konvekcijom događa se zbog micanja zraka oko šipaka i modeliran je jednadžbom*

$$Q_k = HA_p(T - T_o),$$

Primjer (nastavak)

gdje je

- Q_k *snaga hlađenja konvekcijom*
- H *empirička konstanta nazvana koeficijent prijenosa topline*
- A_p *površina plašta šipke*
- T *temperatura šipke (uz pretpostavku da je ona uniformna)*
- T_o *temperatura okolnog zraka*

- *Prijenos topline zračenjem događa se posredstvom elektormagnetskih valova u infracrvenom spektru.*
- *Šipke razmjenjuju toplinu sa bilo kojom plohom koje su u vidljivom dosegu.*
- *Pretpostavit ćemo da su sve plohe koje okružuju šipke temperature* T_o*.*

Primjer (nastavak)

- *Model u tom slučaju glasi*

$$Q_z = \epsilon\sigma A_p(T^4 - T_o^4),$$

gdje je

  - Q_z *snaga hlađenja zračenjem*
  - ϵ *relativna snaga zračenja topline koju ima površina šipke* ($\epsilon \in [0, 1]$)
  - $\sigma = 5.67 \cdot 10^{-8} W/(m^2 K^4)$ *Stefan–Boltzmannova konstanta*

- *Kod prijenosa topline zračenjem temperatura mora biti izražena u* $^\circ K$, *a poslije se može prebaciti u* $^\circ C$

$$T(^\circ K) = T(^\circ C) + 273.15$$

Primjer (nastavak)

- *U svakom trenutku vremena, ravnoteža energije izražena je sa*

$$mc\frac{dT}{dt} = Q_e - Q_k - Q_z,$$

gdje je

  - t *vrijeme*
  - m *masa šipke*
  - c *specifični toplinski kapacitet*
  - Q_e *snaga topline generirane električnom strujom.*

- *Uvrštavanjem izraza za* Q_k *i* Q_z *u prethodnu jednadžbu dobit ćemo*

$$\frac{dT}{dt} = \frac{1}{mc}\{Q_e - A_p[H(T - T_o) + \epsilon\sigma(T^4 - T_o^4)]\}.$$

Primjer (nastavak)

- *Budući da se termička obrada odvija u dvije faze:*
  1. *Uključeno je grijanje električnom strujom i ventilatori su isključeni: Q_k je smanjen jer se hlađenje odvija slobodnom konvekcijom tj. prirodnim strujanjem zraka.*
  2. *Isključena je električna struja i uključeni su ventilatori: Q_k je povećan zbog prisilne konvekcije.*
- *Zbog toga imamo sljedeću situacija*

$$H(t) = \begin{cases} H_1 & \text{za } t < t_h \quad \text{(slobodna konvekcija)} \\ H_2 & \text{za } t \geq t_h \quad \text{(prisilna konvekcija)} \end{cases}$$

gdje je t_h trenutak u kojem je isključeno grijanje i uključeno hlađenje ventilatorima.

Zadatak

Izračunjte temeperaturu u ovisnosti o vremenu u modelu toplinske obrade metalne šipke. Parametri modela su sljedeći:

duljina šipke	$l = 1m$
promjer šipke	$\Phi = 1cm$
gustoća čelika	$\rho = 7822\frac{kg}{m^3}$
specifični toplinski kapacitet čelika	$c = 444\frac{J}{kg\cdot^\circ K}$
relativna snaga zračenja topline površine	$\epsilon = 0.7$
koeficijenti prijenosa topline	$H_1 = 15\frac{W}{m^2\cdot^\circ K}$
	$H_2 = 100\frac{W}{m^2\cdot^\circ K}$
temperatura okolnog zraka	$T_o = 21^\circ C$
vrijeme isključenja grijanja i uključenja hlađenja	$t_h = 70s$
ukupno vrijeme simulacije	$210s$
snaga topline generirane električnom strujom	$Q_e = 3000W$

Zadatak (nastavak)

- *Napišite M-file funkciju* `term_obrada_funkcija(t,T)` *koja treba implementirati funkciju desne strane u ODJ modela*

$$f(t,T) = \frac{1}{mc}\{Q_e - A_p[H(t)(T - T_o) + \epsilon\sigma(T^4 - T_o^4)]\}$$

  sa zadanim parametrima.
- *Kod računanja A_p ne trebate uračunati krajnje kružne plohe.*
- *Problem riješite vašom funkcijom* `odj_rk23()` *za tol $= 10^{-3}$.*
- *Nađite rješenje i sa funkcijom* `odj_rk4()` *pri čemu broj podintervala neka bude isti kao i kod* `odj_rk23()`.

Zadatak (nastavak)

- *Nacrtajte oba rješenja na istom grafu sa linijama u različitim bojama, i sa različitim oznakama točaka.*
- *Adekvatno označite osi i legendu.*
- *Koje je rješenje točnije?*
- *U točnost svoga odgovora uvjerite se tako da udvostručite broj podintervala za* `odj_rk4()` *funkciju.*

Model grabežljivca i plijena

Primjer

- *Promatramo dinamiku populacija dviju međusobno povezanih životinjskih vrsta:*
  1. *plijen (zec) je primarni izvor hrane druge vrste*
  2. *grabežljivac (vuk)*
- *Označimo sa $p_1(t)$ populaciju plijena, a sa $p_2(t)$ populaciju grabežljivca.*
- *Stope prirasta kod obiju populacija možemo prikazati modelom*

$$\begin{aligned} \frac{dp_1}{dt} &= \alpha_1 p_1 - \delta_1 p_1 p_2 \\ \frac{dp_2}{dt} &= \alpha_2 p_1 p_2 - \delta_2 p_2, \end{aligned}$$

gdje su α_1 i α_2 koeficijenti stope rasta, a δ_1 i δ_2 koeficijenti stope mortaliteta.

Primjer (nastavak)

- *U 1. jednadžbi:*
  - *Iraz $\alpha_1 p_1$ opisuje plodnost plijena, za koju se pretpostavlja da ovisi samo o dostupnosti partnera.*
  - *Pretpostavlja se da plijen uvijek ima dostupnu dovoljnu količinu hrane.*
  - *Izraz $\delta_1 p_1 p_2$ je stopa mortaliteta plijena, koja raste s rastom populacija grabežljivca (jer ih jedu) i rastom same populacije plijena.*
  - *Koeficijent δ_1 predstavlja efikasnost u lovu grabežljivaca.*

Primjer (nastavak)

- *U 2. jednadžbi:*
  - *Grabežljivci se reproduciraju u ovisnosti o njihovom broju, ali njihova reprodukcija je ograničena dostupnom hranom (p_1).*
  - *Koeficijent α_2 opisuje i plodnost grabežljivaca, ali i hranjivu vrijednost plijena.*
  - *Mortalitet grabežljivaca raste s rastom populacije grabežljivaca (jer se hrana brže troši).*
- *Početni uvjeti $p_1(0)$ i $p_2(0)$, kao i koeficijenti α_1, δ_1, α_2 i δ_2 moraju doći iz bioloških studija.*

Zadatak

- *Napišite M-file funkcije* `odj_rk4v()` *i* `odj_rk23v()` *koje će biti vektorske verzije funkcija* `odj_rk4()` *i* `odj_rk23()`*, pogodne za rješavanje sustava od m ODJ.*
- *Parametar $y0$ je sada stupčani m-dimenzionirani vektor a y je $m \times (n+1)$ matrica kod koje je i-ti stupac*

$$y(1:m, i) \approx y(x(i)).$$

- *Napišite M-file funkciju* `grab_plijen_funkcija(t,p)` *koja treba implementirati funkciju desne strane u ODJ modela, i koja vraća stupčani vektor od dva elementa $\frac{dp_1}{dt}$ i $\frac{dp_2}{dt}$.*

Zadatak (nastavak)

- *Nađite rješenje sustava modela grabežljivca i plijena sa parametrima*

α_1	*2*
δ_1	*0.02*
α_2	*0.0002*
δ_2	*0.8*
$p_1(0)$	*5000*
$p_2(0)$	*100*

  pomoću metoda `odj_rk4v()` *i* `odj_rk23v()` *u periodu od 30 vremenskih jedinica.*

- *Za* `odj_rk4v()` *uzmite* $n = 300$, *a za* `odj_rk23v()` *uzmite* $tol = 10^{-2}$.

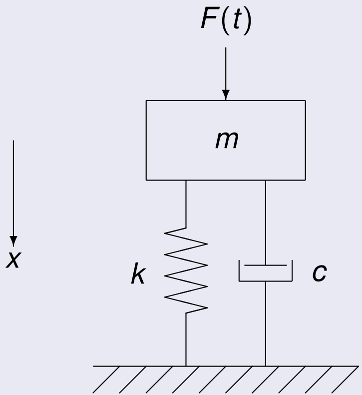
Zadatak (nastavak)

- *Nacrtajte dva grafa na jednoj slici koristeći MATLAB-ovu funkciju* `subplot()`.
- *U gornjem dijelu nacrtajte graf rješenja za plijen (p_1), a u donjem graf rješenja za grabežljivce (p_2).*
- *Kod oba grafa pravilno označite osi, i stavite adekvatne naslove.*

Mehanički sustav

Primjer

Promatramo mehanički sustav sastavljen od utega, opruge i prigušivača.



Primjer (nastavak)

- *Vanjska sila koja ovisi o vremenu primijenjena je na objekt mase m uzrokujući njegovo gibanje.*
- *Opruga djeluje povratnom silom, a prigušivač troši energiju.*
- *Ravnoteža sila koje djeluju na objekt izražena je relacijom*

$$\sum F = ma,$$

gdje je $\sum F$ *suma svih sila koje djeluju na objekt, a* a *je akceleracija objekta.*

- *Sila opruge djeluje u negativnom* x *smjeru, i proporcionalna je skračenju opruge:*

$$F_{opruga} = -kx,$$

gdje je k *konstanta opruge.*

Primjer (nastavak)

- *Prigušivač se opire kretanju objekta, i sila prigušivača se povećava s brzinom objekta:*

$$F_{prigušivač} = -c\dot{x},$$

gdje je $\dot{x} = dx/dt$ *trenutna brzina mase.*

- *Uvrštavajući dvije prethodne jednadžbe u jednadžbu ravnoteže sila dobivamo ODJ drugog reda*

$$F(t) - kx - c\dot{x} = m\ddot{x},$$

gdje je $\ddot{x} = d^2x/dt^2$ *trenutna akceleracija mase.*

- *Prethodna jednadžba se obično piše u obliku*

$$\ddot{x} + 2\zeta\omega_n\dot{x} + \omega_n^2 x = \frac{F}{m},$$

Primjer (nastavak)

gdje je ζ bezdimenzionalni koeficijent prigušivanja a ω_n je prirodna frekvencija:

$$\zeta = \frac{c}{2\sqrt{km}}, \qquad \omega_n = \sqrt{\frac{k}{m}}.$$

- *Jednadžbu drugog reda, zapisujemo sada kao ekvivalentni sustav ODJ prvog reda pomoću transformacija:*

$$y_1 = x, \qquad y_2 = \dot{x}.$$

- *Dakle, imamo*

$$\begin{aligned}\frac{dy_1}{dt} &= y_2\\ \frac{dy_2}{dt} &= \frac{F}{m} - 2\zeta\omega_n y_2 - \omega_n^2 y_1.\end{aligned}$$

Primjer (nastavak)

- *Postoje mnoge moguće funkcije sile $F(t)$.*
- *Zbog jednostavnosti, koristit ćemo tzv. "step" funkciju koja je ekvivalentna postavljanju dodatne mase m_s na vrh objekta mase m u jednom trenutku u vremenu.*
- *U tom slučaju iznos sile primijenjene na objekt je gm_s, gdje je $g = 9.8m/s^2$ akceleracija gravitacije.*
- *Matematički opis step funkcije primijenjene u trenutku t_0 je*

$$F(t) = \begin{cases} 0, & za\ t < t_0 \\ F_0 & za\ t \geq t_0. \end{cases}$$

- *U našem slučaju je $F_0 = ma_0$, gdje je $a_0 = g$.*

Zadatak

- *Napišite M-file funkciju* `meh_sustav_funkcija(t,y)` *koja treba implementirati funkciju desne strane u sustavu ODJ modela, i koja vraća stupčani vektor od dva elementa* $\frac{dy_1}{dt}$ *i* $\frac{dy_2}{dt}$.
- *Nađite rješenje sustava modela mehaničkog sustava s oprugom i prigušivačem*

a_0	$9.8\,\frac{m}{s^2}$
ζ	0.1
ω_n	$35\,\frac{rad}{s}$
$y_1(0) = x(0)$	$0m$
$y_2(0) = \dot{x}(0)$	$0\,\frac{m}{s}$

pomoću metode `odj_rk23v()` *u periodu od 0 do 1.5 sekundi* ($t_0 = 0$). *Uzmite* $tol = 10^{-5}$.

Zadatak (nastavak)

- *Nacrtajte dva grafa na jednoj slici koristeći MATLAB-ovu funkciju* `subplot()`.
- *U gornjem dijelu nacrtajte graf rješenje za* x, *a u donjem graf rješenja za* $\dot{x}$.
- *Kod oba grafa pravilno označite osi, i stavite adekvatne naslove.*

Rubni problem za običnu diferencijalnu jednadžbu

- Rubni problemi su općenitiji od inicijalnih problema.
- Kod rubnog problema traži se rješenje $y(x)$ sustava običnih diferencijalnih jednadžbi

$$y' = f(x, y), \quad x \in \langle a, b \rangle,$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \quad f(x, y) = \begin{bmatrix} f_1(x, y_1, \ldots, y_n) \\ \vdots \\ f_n(x, y_1, \ldots, y_n) \end{bmatrix},$$

koje zadovoljava rubne uvjete

$$r(y(a), y(b)) = 0,$$

gdje je

$$r(u, v) = \begin{bmatrix} r_1(u_1, \ldots, u_n, v_1, \ldots, v_n) \\ \vdots \\ r_n(u_1, \ldots, u_n, v_1, \ldots, v_n) \end{bmatrix}.$$

- Specijalni slučaj su *linearni rubni uvjeti* oblika

$$Ay(a) + By(b) = c,$$

gdje su $A, B \in \mathbb{R}^{n\times n}$, i $c \in \mathbb{R}^n$. Oni su linearni (odnosno afini) po $y(a)$ i $y(b)$.
- U primjeni su često rubni uvjeti *razdvojeni*:

$$A_1 y(a) = c_1, \quad B_2 y(b) = c_2,$$

tj. reci matrica općenitih linearnih rubnih uvjeta A, B i c mogu se tako izpermutirati da budu oblika

$$\bar{A} = \begin{bmatrix} A_1 \\ 0 \end{bmatrix}, \quad \bar{B} = \begin{bmatrix} 0 \\ B_1 \end{bmatrix}, \quad \bar{c} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}.$$

- Prema tome, inicijalni problem može se smatrati specijalnim slučajem rubnog problema za

$$A = I, \quad a = x_0, \quad c = y_0, \quad B = 0.$$

Napomena

Dok za inicijalni problem obično postoji jedinstveno rješenje, rubni problem može imati više rješenja ili niti jedno.

Jednostavna metoda gađanja

- Metodu ćemo objasniti na primjeru rubnog problema

$$y'' = f(x, y, y'),$$

$$y(a) = \alpha, \quad y(b) = \beta,$$

sa razdvojenim rubnim uvjetima.

- Za njega znamo da je ekvivalentan 2×2 sustavu $\mathbf{y}' = \mathbf{f}(x, \mathbf{y}(x))$, gdje je $y_1 = y$ a $y_2 = y'$.
- S druge strane, inicijalni problem

$$y'' = f(x, y, y'),$$

$$y(a) = \alpha, \quad y'(a) = s$$

općenito ima jedinstveno rješenje $y(x) = y(x; s)$ koje ovisi o izboru inicijalne vrijednosti s za $y'(a)$.

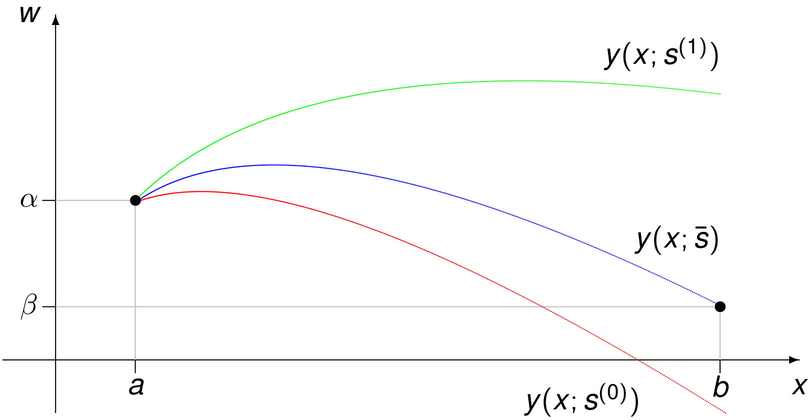
- Kako bismo riješili rubni problem moramo odrediti $s = \bar{s}$ takav da je zadovoljen drugi rubni uvjet

$$y(b) = y(b; \bar{s}) = \beta.$$

- Drugim riječima, moramo naći nultočku $\bar{s}$ funkcije

$$F(s) = y(b; s) - \beta.$$

- Za računanje vrijednosti $F(s)$ za svaki s potrebno je naći rješenje inicijalnog problema.
- Za računanje nultočke $\bar{s}$ od $F(s)$ možemo koristiti bilo koju numeričku metodu:
    - metodu bisekcije
    - Newtonovu metodu



Slika: Jednostavna metoda gađanja. Ako znamo vrijednosti $s^{(0)}$ i $s^{(1)}$ za koje je $F(s^{(0)}) < 0$ i $F(s^{(1)}) > 0$, $\bar{s}$ možemo izračunti pomoću **metode bisekcije**.

- Kako je $y(b; s)$, a onda i $F(s)$ općenito neprekidno diferencijabilna funkcija od s, možemo također koristiti i **Newtonovu metodu** za određivanje $\bar{s}$.
- Počevši od početne aproksimacije $s^{(0)}$, iterativno računamo

$$s^{(i+1)} = s^{(i)} - \frac{F(s^{(i)})}{F'(s^{(i)})}.$$

- $y(b, s^{(i)})$ i $F(s^{(i)})$ mogu se odrediti rješavanjem inicijalnog problema

$$y'' = f(x, y, y'),$$

$$y(a) = \alpha, \quad y'(a) = s^{(i)}.$$

- Derivaciju $F'(s^{(i)})$ aproksimiramo kvocijentom diferencija

$$\Delta F(s^{(i)}) = \frac{F(s^{(i)} + \Delta s^{(i)}) - F(s^{(i)})}{\Delta s^{(i)}},$$

gdje je $\Delta s^{(i)}$ “dovoljno mali”.

- $F(s^{(i)} + \Delta s^{(i)})$ se ponovo dobiva rješavanjem inicijalnog problema.
- Zbog mogućih numeričkih problema, često se uzima

$$\Delta s^{(i)} = \sqrt{\text{eps}} \cdot s^{(i)}.$$

- U ovom konkretnom slučaju kada imamo samo **jednu** diferencijalnu jednadžbu 2. reda, Newtonovu metodu možemo izvesti i **egzaktno**.
- Parcijalno derivirajmo cijelu jednadžbu inicijalnog problema po s, pri čemu dobivamo:

$$\begin{aligned}\left(\frac{\partial y}{\partial s}\right)'' = {} & \frac{\partial f}{\partial y}(x, y(x;s), y'(x;s))\frac{\partial y}{\partial s}(x;s) + \\ & + \frac{\partial f}{\partial y'}(x, y(x;s), y'(x;s))\left(\frac{\partial y}{\partial s}\right)'(x;s)\end{aligned}$$

$$\frac{\partial y}{\partial s}(a;s) = 0, \qquad \left(\frac{\partial y}{\partial s}\right)'(a;s) = 1.$$

- Ovime smo dobili sustav od dvije diferencijalne jedadžbe 2. reda, koje ćemo prikazati kao ekvivalentan sustav od 4 jednadžbe 1.reda, s nepoznanicama $y_1 = y(x;s)$, $y_2 = y'(x;s)$, $y_3 = \frac{\partial y}{\partial s}(x;s)$ i $y_4 = \left(\frac{\partial y}{\partial s}\right)'(x;s)$:

$$\begin{aligned}
y_1' &= y_2 \\
y_2' &= f(x, y, y') \\
y_3' &= y_4 \\
y_4' &= \frac{\partial f}{\partial y}(x, y_1, y_2)y_3 + \frac{\partial f}{\partial y'}(x, y_1, y_2)y_4,
\end{aligned}$$

s inicijalnim uvjetima

$$y_1(a) = \alpha, \quad y_2(a) = s, \quad y_3(a) = 0, \quad y_4(a) = 1.$$

- Dobivene vrijednosti $y_1(b)$ i $y_3(b)$ su nam potrebne u iteraciji Newtonove metode:

$$\begin{aligned} s^{(i+1)} &= s^{(i)} - \frac{F(s^{(i)})}{F'(s^{(i)})} \\ &= s^{(i)} - \frac{y(b; s^{(i)}) - \beta}{\frac{\partial y}{\partial s}(b; s^{(i)})} \end{aligned}$$

Primjer

- *Razmatramo rubni problem*

$$y'' = \frac{3}{2}y^2,$$

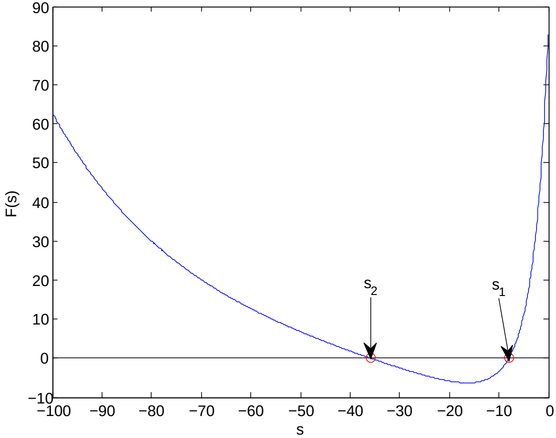
$$y(0) = 4, \quad y(1) = 1.$$

- *Rješavamo li ovaj problem metodom gađanja, tada moramo naći rješenja inicijalnih problema*

$$y'' = \frac{3}{2}y^2,$$

$$y(0; s) = 4, \quad y'(0; s) = s.$$

- *Graf funkcije $F(s) = y(1; s) - 1$ prikazan je na sljedećoj slici.*

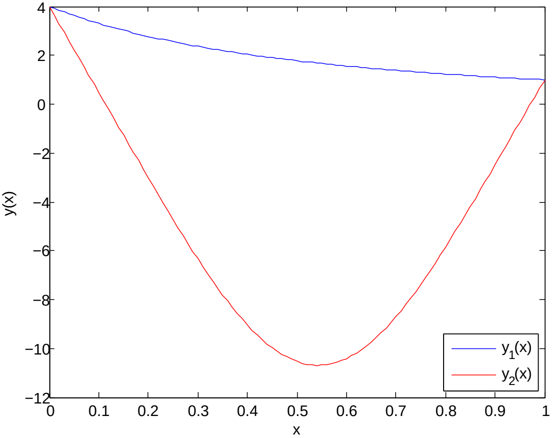

90
80
70
60
50
40
30
20
10
0
−10
F(s)
−100
−90
−80
−70
−60
−50
−40
−30
−20
−10
0
s
s_2
s_1

Primjer (nastavak)

- *Može se pokazati da funkcija $F(s)$ ima dvije nultočke $\bar{s}_1$ i $\bar{s}_2$.*
- *Iteracije metoda gađanja, pri kojima inicijalni problem rješavamo egzaktno, dat će*

$$\bar{s}_1 = -8.000\,000\,0000,$$
$$\bar{s}_2 = -35.858\,548\,7278,$$

a odgovarajuća rješenja rubnog problema označit ćemo sa $y_1(x)$ i $y_2(x)$.


4
2
0
−2
−4
−6
−8
−10
−12
y(x)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
x
y1(x)
y2(x)

- Za računanje rješenja općenitog rubnog problema sa n nepoznatih funkcija $y_i(x)$, $i = 1, \ldots, n$

$$y' = f(x, y), \quad y = \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix}^T,$$

$$r(y(a), y(b)) = 0,$$

gdje su $f(x, y)$ i $r(u, v)$ vektori od n funkcija, postupak je sličan kao u prethodnom primjeru.

- Trebamo naći početni vektor $s \in \mathbb{R}^n$ za inicijalni problem

$$y' = f(x, y), \quad y(a) = s,$$

takav da rješenje inicijalnog problema $y(x) = y(x; s)$ zadovoljava rubne uvjete

$$r(y(a; s), y(b; s)) = r(s, y(b; s)) = 0.$$

- Znači, moramo naći rješenje $s = [\ s_1 \quad s_2 \quad \cdots \quad s_n\ ]^T$ jednadžbe

$$F(s) = 0, \quad F(s) = r(s, y(b; s)).$$

- Rješenje jednadžbe može se izračunati Newtonovom metodom

$$s^{(i+1)} = s^{(i)} - DF(s^{(i)})^{-1} F(s^{(i)}).$$

- U svakoj iteraciji Newtonove metode moramo
  - naći rješenje $y(x, s^{(i)})$ inicijalnog problema $y' = f(x, y)$, $y(a) = s^{(i)}$,
  - izračunati $F(s^{(i)}) = r(s^{(i)}, y(b; s^{(i)}))$,
  - izračunati $DF(s^{(i)})$ ili neku njegovu aproksimaciju,
  - riješiti sustav linearnih jednadžbi

$$DF(s^{(i)}) d^{(i)} = F(s^{(i)}), \quad \text{gdje je } d^{(i)} = s^{(i)} - s^{(i+1)}.$$

- $DF(s^{(i)})$ aproksimiramo pomoću kvocijenata diferencija

$$\Delta F(s) = [\Delta_1 F(s), \ldots, \Delta_n F(s)],$$

gdje su

$$\Delta_j F(s) = \frac{F(s_1, \ldots, s_j + \Delta s_j, \ldots, s_n) - F(s_1, \ldots, s_j, \ldots, s_n)}{\Delta s_j}$$

- Računanje $\Delta_j F(s)$ zahtijeva računanje $y(b; s) = y(b; s_1, \ldots, s_n)$ i $y(b; s_1, \ldots, s_j + \Delta s_j, \ldots, s_n)$ za koje trebamo naći rješenja odgovarajućih inicijalnih problema.

- Za linearne rubne uvjete

$$r(u, v) = Au + Bv - c,$$

imamo sljedeću situaciju

$$\begin{aligned} F(s) =& As + By(b; s) - c, \\ DF(s) =& A + BZ(b; s), \end{aligned}$$

gdje je $Z(b; s) = D_s y(b; s)$ matrica sa elementima

$$Z(b; s) = \left[\frac{\partial y_i(b; s)}{\partial s_j}\right].$$

- U ovom slučaju parcijalnu derivaciju od $y(b; s)$ aproksimiramo kvocijentom diferencija

$$\Delta_j y(b; s) = \frac{y(b; s_1, \ldots, s_j + \Delta s_j, \ldots, s_n) - y(b; s_1, \ldots, s_j, \ldots, s_n)}{\Delta s_j}$$

$$\Delta F(s) = A + B\Delta y(b; s), \quad \Delta y(b; s) = [\Delta_1 y(b; s), \ldots, \Delta_n y(b; s)].$$

- Dakle, za izvođenje aproksimativne Newtonove metode

$$s^{(i+1)} = s^{(i)} - \Delta F(s^{(i)})^{-1} F(s^{(i)}),$$

moraju se izvršiti sljedeći koraci:

Algoritam

Izaberi početni vektor $s^{(0)}$
Za $i = 0, 1, 2, \ldots$

1. *nađi* $y(b; s^{(i)})$ *rješavanjem inicijalnog problema* $y' = f(x, y)$, $y(a) = s^{(i)}$, *i izračunaj* $F(s^{(i)}) = r(s^{(i)}, y(b; s^{(i)}))$,
2. *izaberi dovoljno male brojeve* $\Delta s_j \neq 0$, $j = 1, \ldots, n$ *i nađi* $y(b; s^{(i)} + \Delta s_j e_j)$ *rješavanjem* n *inicijalnih problema* $y' = f(x, y)$, $y(a) = s^{(i)} + \Delta s_j e_j$, $j = 1, \ldots, n$,
3. *izračunaj* $\Delta F(s^{(i)})$, *nađi rješenje* $d^{(i)}$ *sustava linearnih jednadžbi* $\Delta F(s^{(i)}) d^{(i)} = F(s^{(i)})$ *i definiraj* $s^{(i+1)} = s^{(i)} - d^{(i)}$.

- Dakle, u svakom koraku prethodno opisane metode potrebno je riješiti $n+1$ inicijalni problem i jedan sustav linearnih jednadžbi reda n.
- Za ovu metodu $s^{(0)}$ mora biti dosta blizu rješenju $\bar{s}$ od $F(s) = 0$, inače metoda divergira.
- Još k tome može konvergirati i sporo.
- Još jedan nedostatak metode gađanja je taj da je rješenje inicijalnog problema jako osjetljivo na male promjene u inicijalnom uvjetu, do kojih može doći uslijed grešaka zaokruživanja.

Jednostavna metoda gađanja za linearan rubni problem

- Zamjenom $DF(s^{(i)})$ sa $\Delta F(s^{(i)})$ obično se gubi kvadratna konvergencija Newtonove metode.
- U specijalnom slučaju *linearnog rubnog problema* vrijedi $DF(s^{(i)}) = \Delta F(s^{(i)})$ za sve s i Δs_j.
- *Linearno* znači da je $f(x, y)$ afina funkcija od y, a rubni uvjeti su linearni:

$$y' = T(x)y + g(x),$$

$$Ay(a) + By(b) = c,$$

sa $n \times n$ matricom $T(x)$, funkcijom $g : \mathbb{R} \to \mathbb{R}^n$, $c \in \mathbb{R}^n$, i konstantnim $n \times n$ matricama A i B.
- Nadalje pretpostavljamo da su $T(x)$ i $g(x)$ neprekidne funkcije na $[a, b]$.

- Ponovo sa $y(x; s)$ označimo rješenje inicijalnog problema

$$y' = T(x)y + g(x), \quad y(a; s) = s.$$

- Za $y(x; s)$ postoji eksplicitna formula

$$y(x; s) = Y(x)s + y(x; 0),$$

gdje je $n \times n$ matrica $Y(x)$ rješenje inicijalnog problema

$$Y' = T(x)Y, \quad Y(a) = I.$$

- Ako označimo $u(x; s) = Y(x)s + y(x; 0)$, tada imamo

$$\begin{aligned} u(a; s) &= Y(a)s + y(a; 0) = Is + 0 = s \\ D_x u(x; s) &= u'(x; s) = Y'(x)s + y'(x; 0) \\ &= T(x)Y(x)s + T(x)y(x; 0) + g(x) \\ &= T(x)u(x; s) + g(x), \end{aligned}$$

pa je $u(x; s)$ rješenje inicijalnog problema.

- Budući da, zbog gornjih pretpostavki o $T(x)$ i $g(x)$, inicijalni problem ima jedinstveno rješenje, slijedi da je $u(x; s) = y(x; s)$.
- Za funkciju $F(s)$ imamo

$$F(s) = As + By(b; s) - c = [A + BY(b)]s + By(b; 0) - c.$$

- Zbog toga je i $F(s)$ afina funkcija od s.
- Dalje vrijedi

$$DF(s) = \Delta F(S) = A + BY(b) = \Delta F(0).$$

- Rješenje $\bar{s}$ od $F(s) = 0$ (uz pretpostavku da $[\Delta F(0)]^{-1}$ postoji) je dano sa

$$\begin{aligned}\bar{s} =& -[A + BY(b)]^{-1}[By(b; 0) - c] \\ =& 0 - [\Delta F(0)]^{-1} F(0).\end{aligned}$$

- Ili malo općenitije

$$\bar{s} = s^{(0)} - [\Delta F(s^{(0)})]^{-1} F(s^{(0)}),$$

gdje je $s^{(0)} \in \mathbb{R}^{n \times n}$ proizvoljan.

- Drugim riječima, rješenje $\bar{s}$ od $F(s) = 0$ a i rješenje linearnog rubnog problema, izračunat će se metodom gađanja u jednoj jedinoj iteraciji sa proizvoljnim početnim vektorom $s^{(0)}$.

Zadaci

Zadatak

Metodom gađanja riješite rubni problem iz prethodnog primjera:

$$y'' = \frac{3}{2}y^2,$$

$$y(0) = 4, \quad y(1) = 1.$$

- *Napišite M-file funkciju* `f=f_2rru(x,y)` *koja implementira funkciju desne strane ekvivalentnog sustava* $\mathbf{y}'(x) = \mathbf{f}(x, \mathbf{y})$ *sa dvije ODJ.*
- *Napišite M-file funkciju* `odj_gadjanje_2rru()` *koja implementira samu metodu pomoću RK-4 metode za sustave ODJ inicijalnog problema, i koristi aproksimaciju* $\Delta F(s^{(i)})$ *za* $F'(s^{(i)})$ *u Newtonovim iteracijama.*

Zadatak (nastavak)

Funkcija `odj_gadjanje_2rru()` *neka ima ulazne parametre*

- *pokazivač na funkciju f*
- *rubove segmenta a i b*
- *rubne uvjete α i β*
- *početnu aproksimaciju inicijalnog uvjeta $s^{(0)}$*
- *broj ekvidistantnih podintervala n za inicijalni problem.*

Izlazni parametri neka su

- *vektor x duljine $n+1$ s vrijednostima x_i*
- *vektor y duljine $n+1$ s vrijednostima $y_i \approx y(x_i)$*
- *inicijalni uvjet $\bar{s}$, rješenje jednadžbe $F(s) = 0$.*

Zadatak

Prethodni zadatak riješite metodom gađanja sa egzaktnim Newtonovim iteracijama.

- *Napišite M-file funkciju* `odj_gadjanje_2rru_en()` *koja implementira samu metodu pomoću RK-4 metode za sustave ODJ inicijalnog problema.*
- *Ova funkcija neka ima sve iste ulazne i izlazne parametre kao* `odj_gadjanje_2rru()` *uz još dva dodatna ulazna parametra:*
  - *pokazivač na funkciju* dfy, *koja za* $f(x, y, y') = \mathbf{f}_2(x, \mathbf{y})$ *funkciju desne strane originalne diferencijalne jednadžbe 2. reda predstavlja* $\frac{\partial f}{\partial y}$,
  - *pokazivač na funkciju* $dfdy$, *koja predstavlja* $\frac{\partial f}{\partial y'}$.
- *Usporedite broj Newtonovih iteracija u obje metode iz ovog i prethodnog zadatka, za postizanje iste točnosti.*

Zadatak

Metodom gađanja riješite linearan rubni problem

$$y' = T(x)y + g(x),$$

$$Ay(a) + By(b) = c,$$

zadan sljedećim parametrima:

$$T(x) = \begin{bmatrix} x & 0 & -x^2 \\ 2x^3 - x & 3x^2 - 2x + 1 & -4x - 2 \\ -3x^3 + 2 & 2x^3 + x^2 - 3x & 2x \end{bmatrix},$$

$$g(x) = \begin{bmatrix} -2x^2 + 5x - 3 \\ x + 3 \\ -x^3 + 1 \end{bmatrix},$$

Zadatak (nastavak)

$$a = 0, \qquad b = 2,$$

$$A = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 3 & 1 \\ 4 & 5 & -2 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} 3 \\ -8 \\ 5 \end{bmatrix}.$$

Napišite M-file funkciju `odj_gadjanje_linrp()` *koja implementira metodu gadjanja pomoću RK-4 metode za sustave ODJ inicijalnog problema. Funkcija neka ima ulazne parametre*

- *pokazivač na funkciju T koja implementira $T(x)$*
- *pokazivač na funkciju g koja implementira $g(x)$*
- *rubove segmenta a i b*
- *matrice iz rubnih uvjeta A, B i c*
- *broj ekvidistantnih podintervala n za inicijalni problem.*

Zadatak (nastavak)

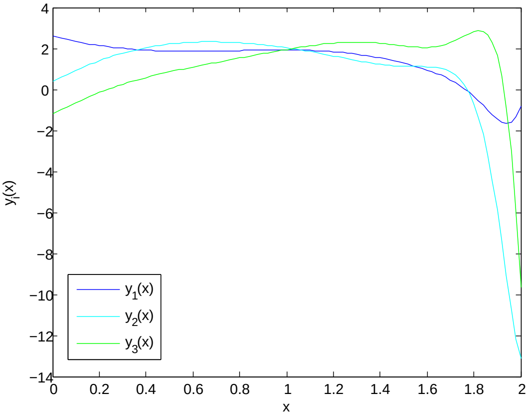
Izlazni parametri neka su

- *vektor x duljine $n+1$ s vrijednostima x_j*
- *$m \times (n+1)$ matrica y čiji (i,j)-ti element sadrži vrijednost $y_{i,j} \approx y_i(x_j)$, gdje je $y = [\ y_1 \quad \dots \quad y_m\ ]^T$,*
- *inicijalni uvjet $\bar{s}$, rješenje jednadžbe $F(s) = 0$.*

Za rješavanje inicijalnog problema

$$Y' = T(x)Y, \quad Y(a) = I,$$

također koristite RK-4 metodu za sustave ODJ inicijalnog problema, s time da rješavate stupac po stupac.



Slika: Komponente rješenja prethodnog zadatka.

Metoda kolokacije

- Rješavamo jedan važan tip rubnog problema za funkciju $u : [a, b] \to \mathbb{R}$

$$-(p(x)u'(x))' + q(x)u(x) = g(x, u(x)),$$
$$u(a) = \alpha, \quad u(b) = \beta.$$

- Pod pretpostavkama

$$\begin{aligned} &p \in C^1[a, b], && p(x) \geq p_0 > 0, \\ &q \in C[a, b], && q(x) \geq 0, \\ &g \in C^1([a, b] \times \mathbb{R}), && \frac{\partial g}{\partial u}(x, u) \leq \lambda_0, \end{aligned}$$

gdje je λ_0 najmanja svojstvena vrijednost svojstvenog problema

$$-(pz')' - (\lambda - q)z = 0, \quad z(a) = z(b) = 0,$$

gornji rubni problem uvijek ima jedinstveno rješenje.

- Ako je $u(x)$ rješenje danog rubnog problema, tada je $y(x) = u(x) - \ell(x)$, gdje

$$\ell(x) = \alpha\frac{b-x}{b-a} + \beta\frac{a-x}{a-b}, \quad \ell(a) = \alpha, \; \ell(b) = \beta,$$

rješenje rubnog problema oblika

$$-(py')' + qy = f,$$

$$y(a) = 0, \quad y(b) = 0,$$

sa trivijalnim rubnim uvjetima.

- Bez smanjenja općenitosti možemo promatrati rubne probleme sa trivijalnim rubnim uvjetima.
- Dalje možemo definirati diferencijalni operator

$$L(v) = -(pv')' + qv.$$

- L preslikava skup

$$D_L = \{v \in C^2[a,b] : v(a) = 0, v(b) = 0\}$$

svih realnih funkcija dva puta neprekidno diferencijabilnih na $[a,b]$ i koje zadovoljavaju rubne uvjete $v(a) = v(b) = 0$ u skup $C[a,b]$ neprekidnih funkcija na $[a,b]$.

- Rubni problem sa trivijalnim rubnim uvjetima je stoga ekvivalentan traženju rješenja problema

$$L(y) = f, \quad y \in D_L.$$

- Lako se provjeri da je D_L realni vektorski prostor, a L je linearni operator na D_L.

- Za metodu kolokacije biramo konačnodimenzionalan podskup $S \subset D_L$ funkcija koje zadovoljavaju trivijalne rubne uvjete.
- Zatim pokušavamo aproksimirati rješenje y pomoću funkcije $v(x) \in S$ reprezentirane kao

$$v = \alpha_1 v_1 + \cdots + \alpha_m v_m,$$

gdje je $\{v_1, \ldots, v_m\}$ baza od S.
- U tu svrhu biramo m različitih *kolokacijskih točaka* $x_i \in \langle a, b \rangle$, $i = 1, \ldots, m$ i tražimo $v \in S$ takav da je

$$(Lv)(x_i) = f(x_i), \quad i = 1, \ldots, m.$$

- To je ekvivalentno rješavanju sljedećeg sustava linearnih jednadžbi za koeficijente α_j, $j = 1, \ldots, m$:

$$\sum_{j=1}^{m} L(v_j)(x_i)\alpha_j = f(x_i), \quad j = 1, \ldots, m.$$

- Dakle rješavamo sustav $Ax = b$ gdje su

$$A = \begin{bmatrix} L(v_1)(x_1) & L(v_2)(x_1) & \cdots & L(v_m)(x_1) \\ L(v_1)(x_2) & L(v_2)(x_2) & \cdots & L(v_m)(x_2) \\ \vdots & \vdots & & \vdots \\ L(v_1)(x_m) & L(v_2)(x_m) & \cdots & L(v_m)(x_m) \end{bmatrix},$$

$$x = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix}, \quad b = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_m) \end{bmatrix}.$$

- Postoje razni izbori baza za S i kolokacijski točaka kod implementacije ove metode:
    - B-splajnovi
    - trigonometrijski polinomi — *spektralna metoda*

Metoda kolokacije sa kubičnim B-splajnovima

- U ovom slučaju biramo podjelu intervala $[a, b]$ na podintervale određene točkama

$$a = s_0 < s_1 < \cdots < s_{n-1} < s_n = b,$$

pri čemu označavamo $\mathbf{s} = \{s_0, \ldots, s_n\}$.

- Najprije definirajmo skup

$$\mathbb{PP}_{3,\mathbf{s}} = \{v \in C^2[a, b] : v|_{\langle s_i, s_{i+1}\rangle} \in \mathbb{P}_3(\langle s_i, s_{i+1}\rangle), i = 0, \ldots, n-1\},$$

po dijelovima polinomnih funkcija stupnja 3, dva puta neprekidno diferencijabilnih.

- Mi ćemo birati skup S kao

$$S = \{v \in \mathbb{PP}_{3,\mathbf{s}} : v(a) = 0, v(b) = 0\}.$$

- Definirajmo sada **čvorove** B-splajnova u našem slučaju kao

$$t_0 = t_1 = t_2 = t_3 = s_0,$$

$$t_4 = s_1,\ t_5 = s_2,\ \ldots\ , t_{n+2} = s_{n-1},$$

$$t_{n+3} = t_{n+4} = t_{n+5} = t_{n+6} = s_n,$$

pri čemu označavamo sa $N = n + 6$ i $\mathbf{t} = \{t_0, \ldots, t_N\}$,

$$\mathbf{t} = \{s_0, s_0, s_0, s_0, s_1, s_2, \ldots, s_{n-1}, s_n, s_n, s_n, s_n\}.$$

- Može se pokazati da je $\mathbb{PP}_{3,\mathbf{s}}$ vektorski prostor dimenzije

$$d + 1 = \dim(\mathbb{PP}_{3,\mathbf{s}}) = N - 3 = n + 3,$$

i da je njegovu bazu čine B-splajnovi $B_{0,3}, \ldots, B_{d,3}$ određeni čvorovima $\mathbf{t}$.

Definicija (de Boor – Coxova rekurzija)

- *Neka je* $t_0 \leq t_1 \leq \cdots \leq t_N$ *niz realnih brojeva.*
- *Za* $k = 0, \ldots, N-1$ *i* $i = 0, \ldots, N-k-1$

definiramo i-ti (normalizirani) B-splajn stupnja k kao

$$B_{i,0}(t) = \begin{cases} 1, & \text{za } t \in [t_i, t_{i+1}\rangle \\ 0, & \text{inače} \end{cases}$$

a za $k > 0$

$$B_{i,k}(t) = \begin{cases} \frac{t-t_i}{t_{i+k}-t_i} B_{i,k-1}(t) + \frac{t_{i+k+1}-t}{t_{i+k+1}-t_{i+1}} B_{i+1,k-1}(t), & \text{za } t_i < t_{i+k+1} \\ 0, & \text{inače} \end{cases}$$

- U slučaju da je $t_{i+k} - t_i = 0$ ili $t_{i+k+1} - t_{i+1} = 0$ odgovarajući izraz u rekurziji se uopće ne računa.

- Ako želimo neprekidnost i na desnom rubu domene moramo modificirati de Boor – Coxovu rekurziju.

Definicija

Neka je zadan vektor čvorova $\{t_i\}_{i=0}^{N}$, *i pretpostavimo da je* $j = \max\{i : t_i < t_{i+1}, i \in \{0, \ldots, N-1\}\}$. *Definiramo*

$$B_{i,0}(t) = \begin{cases} 1, & \text{za } t \in [t_i, t_{i+1}\rangle, \; i < j \\ 1, & \text{za } t \in [t_j, t_{j+1}], \; i = j \\ 0, & \text{inače} \end{cases}$$

Ova modifikacija utječe samo na zadnji netrivijalni B-splajn svih stupnjeva, koji time postaje neprekidan u desnom rubu domene. $B_{N-k-1,k}(t_{N-k}) = 1$.

Svojstva B-splajnova

Lema

Ako je $t < t_i$ ili $t \geq t_{i+k+1}$, tada je $B_{i,k}(t) = 0$, tj. $B_{i,k}(t)$ može biti različit od 0 samo na intervalu $[t_i, t_{i+k+1}\rangle$.

Lema

$$B_{i,k}(t) > 0 \quad za \quad t \in \langle t_i, t_{i+k+1}\rangle.$$

Konkretinije, $B_{i,k}(t_{i+k+1}) = 0$, a $B_{i,k}(t_i)$ može biti jednak ili različit 0, ovisno o čvorovima.

Korolar

Ako je $t \in \langle t_j, t_{j+1}\rangle$, za $t_j < t_{j+1}$, i $i \in \{j-k, j-k+1, \ldots, j\}$ tada je $B_{i,k}(t) > 0$ za njih ukupno $k+1$

- Možemo zaključiti da B-splajnovi imaju ograničene nosače, što predstavlja poželjno svojstvo jer će matrica sustava kolokacije biti vrpčasta.

Lema

Za vektor čvorova $\mathbf{t} = \{t_i\}_{i=0}^{N}$ *i za* $t \in [t_k, t_{N-k}\rangle$ *vrijedi*

$$\sum_{i=0}^{N-k-1} B_{i,k}(t) = 1, \quad \text{za sve } k \geq 0.$$

Dalje, za $t < t_k$ *ili* $t > t_{N-k}$ *je*

$$\sum_{i=0}^{N-k-1} B_{i,k}(t) < 1.$$

Teorem

Ako je $B_{i,k}(t)$ B-splajn stupnja k definiran na vektoru čvorova $\mathbf{t}$, tamo gdje derivacija postoji (između čvorova i ponekad u samim čvorovima) ona se može izračunati pomoću izraza

$$B'_{i,k}(t) = k\left(\frac{B_{i,k-1}(t)}{t_{i+k}-t_i} - \frac{B_{i+1,k-1}(t)}{t_{i+k+1}-t_{i+1}}\right).$$

Korolar

Ako je zadan vektor čvorova $\{t_i, t_{i+1}, \ldots, t_{i+k+1}\}$ koji ima različite rastuće vrijednosti $\{s_{i_1}, \ldots, s_{i_n}\}$, pri čemu se svaka od njih ponavlja sa multiplicitetima $\{m_{i_1}, \ldots, m_{i_n}\}$, tada je $B_{i,k} \in C^{(k-m_{i_j})}$ u s_{i_j}.

Korolar

Za vektor čvorova $\mathbf{t} = \{t_i\}_{i=0}^{d+k+1}$ *takav da je* $t_0 = \cdots = t_k$ *i* $t_{d+1} = \cdots = t_{d+k+1}$ *funkcija*

$$v(t) = \sum_{i=0}^{d} \alpha_i B_{i,k}(t)$$

interpolira $v(t_k) = \alpha_0$ *i* $v(t_{d+1}) = \alpha_d$*.*

Teorem

Ako je $k \geq 0$ *cijeli broj, a* $\mathbf{t}$ *je vektor čvorova sa* $d + k + 2$ *elemenata,* $k \leq d$*, tada je* $\{B_{i,k,\mathbf{t}}, i = 0, \ldots, d\}$ *linearno nezavisan skup u* $\mathbb{PP}_{k,\mathbf{s},\mathbf{m}}$ *i čini njegovu bazu.*
($N = d + k + 1$*,* $\mathbf{m} = \{m_0, \ldots, m_n\}$*)*

- Prethodno opisana svojstva B-splajnova koristit će nam za računanje koeficijenata α_j u rastavu funkcije v:

$$v(x) = \sum_{j=0}^{d} \alpha_j B_{j,k}(x).$$

- Samo izvrednjavanje funkcije v u nekoj točci x može se tada pojednostavniti rekurzivnim de Boorovim algoritmom.

Algoritam (de Boorov rekurzivni algoritam za B-splajnove)

Za definiranje po dijelovima polinomne funkcije $v(x)$ stupnja k, potrebno je da je domena krivulje jednaka $[t_k, t_{N-k}\rangle$.

1. $\alpha_i^{[0]} = \alpha_i$.
2. *Za dani $x \geq t_k$, nađi j takav da je $x \in [t_j, t_{j+1}\rangle$.*
3. *Za $\ell = 1, \ldots, k$*

   za $i = j - k + \ell, \ldots, j$

$$\alpha_i^{[\ell]} = \frac{x - t_i}{t_{i+k+1-\ell} - t_i}\alpha_i^{[\ell-1]} + \frac{t_{i+k+1-\ell} - x}{t_{i+k+1-\ell} - t_i}\alpha_{i-1}^{[\ell-1]}.$$

4. $v(x) = \alpha_j^{[k]}$.

Upotreba kubičnih splajnova u metodi kolokacije

- Budući da je nosač kubičnog B-splajna $B_{3,i}$ sadržan unutar $[t_i, t_{i+4}\rangle$, to isto vrijedi i za $L(B_{3,i})$.
- Dalje, u svakoj točci $x \in \langle a, b\rangle$ najviše 4 B-splajna poprima netrivijalnu vrijednost, to znači da matrica A ima najviše 4 netrivijalna elementa u jednom retku.
- Dimenzija od $\mathbb{PP}_{3,\mathbf{s}}$ je $d+1 = n+3$, što znači da imamo $n+3$ koeficijenta α_j, $j = 0, \ldots, n+2 = d$ za odrediti.
- Za to su nam potrebne $n+3$ kolokacijske točke.
- S druge strane, za zadane čvorove $\mathbf{t}$, znamo da vrijedi $t_3 = a$ i $t_{d+1} = t_{n+3} = b$ i
$$v(a) = \alpha_0, \quad v(b) = \alpha_d,$$
što možemo odrediti iz rubnih uvjeta.
- Dakle, preostaje nam odrediti još $n+1$ koeficijenata u $n+1$ kolokacijski točaka unutar $\langle a, b\rangle$.

- Ako izaberemo kolokacijske točke

$$x_1 < x_2 < \cdots < x_{n-1} < x_{n+1} \in \langle a, b \rangle,$$

  onda je za regularnost matrice A nužno da svaki B-splajn $B_{j,3}$ sadrži unutar nosača bar jednu kolokacijsku točku x_{i_j} i da su sve x_{i_j} međusobno različite za $j = 0, \ldots, d$ (kako A ne bi imala nul-stupac).
- Ako rubni problem ima netrivijalne rubne uvjete $v(a) = \alpha$ i $v(b) = \beta$, tada rješavamo $(n+3) \times (n+3)$ sustav $Ax = b$ sa

$$A = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ L(B_{0,3})(x_1) & L(B_{1,3})(x_1) & \cdots & L(B_{n+1,3})(x_1) & L(B_{n+2,3})(x_1) \\ L(B_{0,3})(x_2) & L(B_{1,3})(x_2) & \cdots & L(B_{n+1,3})(x_2) & L(B_{n+2,3})(x_2) \\ \vdots & \vdots & & \vdots & \vdots \\ L(B_{0,3})(x_{n+1}) & L(B_{1,3})(x_{n+1}) & \cdots & L(B_{n+1,3})(x_{n+1}) & L(B_{n+2,3})(x_{n+1}) \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

$$x = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{n+1} \\ \alpha_{n+2} \end{bmatrix}, \quad b = \begin{bmatrix} \alpha \\ f(x_1) \\ \vdots \\ f(x_{n+1}) \\ \beta \end{bmatrix}.$$

- Ako rubni problem ima trivijalne rubne uvjete $v(a) = v(b) = 0$, tada je

$$0 = v(a) = \alpha_0, \quad 0 = v(b) = \alpha_d,$$

  što znači da je skup S potprostor od $\mathbb{PP}_{3,\mathbf{s}}$ i njegova dimenzija je

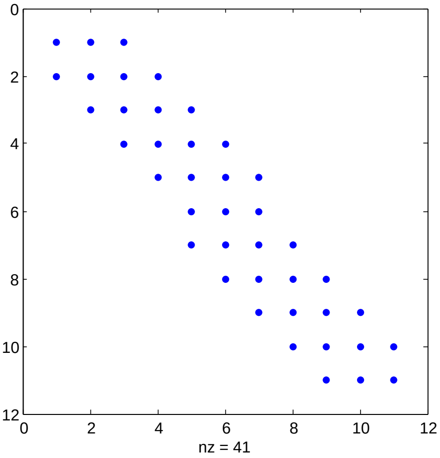
$$\dim(S) = \dim(\mathbb{PP}_{3,\mathbf{s}}) - 2 = n + 1,$$

  a upravo toliko ima i kolokacijskih točaka.
- Bazu prostora S čine $B_{j,3}$, $j = 1, \ldots, n+1$.

- U tom slučaju rješavamo $(n+1)\times(n+1)$ sustav $Ax=b$ sa

$$A=\begin{bmatrix} L(B_{1,3})(x_1) & L(B_{2,3})(x_1) & \cdots & L(B_{n+1,3})(x_1) \\ L(B_{1,3})(x_2) & L(B_{2,3})(x_2) & \cdots & L(B_{n+1,3})(x_2) \\ \vdots & \vdots & & \vdots \\ L(B_{1,3})(x_{n+1}) & L(B_{2,3})(x_{n+1}) & \cdots & L(B_{n+1,3})(x_{n+1}) \end{bmatrix},$$

$$x=\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{n+1} \end{bmatrix}, \quad b=\begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_{n+1}) \end{bmatrix}.$$



Slika: Raspored netrivijalnih elemenata u matrici A za jedan primjer rubnog problema sa trivijalnim rubnim uvjetima.

Zadaci

Zadatak

Napišite M-file funkciju `deBoor_Cox()` *koja implementira de Boor–Coxovu rekurziju za izvrednjavanje B-splajna $B_{i,k}$ u točci* x. *Funkcija neka ima ulazne parametre*

- *točku* x
- *čvorove B-splajnova* t
- *stupanj B-splajnova* k
- *indeks* i

i izlazne parametre

- y *vrijednost B-splajna* $B_{i,k}(x)$
- dy *vrijednost derivacije B-splajna* $B'_{i,k}(x)$
- $d2y$ *vrijednost 2. derivacije B-splajna* $B''_{i,k}(x)$

Zadatak

Napišite M-file funkciju `odj_kolokacija_kBs_Ab()` *koja računa matricu A i vektor b za metodu kolokacije sa kubičnim B-splajnovima, uz pomoć funkcije* `deBoor_Cox()`. *A i b neka budu definirani za rubni problem sa trivijalnim rubnim uvjetima. Funkcija neka ima ulazne parametre*

- *čvorove B-splajnova t*
- *kolokacijske točke x*
- *pokazivač na funkciju* $L = L(x, y, y', y'')$ *koja implementira diferencijalni operator*
- *pokazivač na funkciju* $f = f(x)$ *desne strane ODJ*

i izlazne parametre

- *matricu A*
- *vektor b*

Zadatak

Napišite M-file funkciju `deBoor()` *koja implementira de Boorov algoritam za izvrednjavanje po dijelovima polinomne funkcije* $v(x) = \sum_{j=0}^{d} \alpha_j B_{j,k}(x)$ *u točci* `x`. *Funkcija neka ima ulazne parametre*

- *točku* `x`
- *čvorove B-splajnova* `t`
- *stupanj B-splajnova* `k`
- *parametre* α_j *spremljene u* $d+1$*-dimenzionalno polje* `alpha`

i izlazni parametar

- `v` *vrijednost funkcije* $v(x)$

Zadatak

Pomoću metode kolokacije sa kubičnim B-splajnovima riješite sljedeći rubni problem

$$-y'' + 400y = -400\cos^2(\pi x) - 2\pi^2\cos(2\pi x),$$

$$y(0) = y(1) = 0.$$

- *Rubovi podintervala neka su zadani sa* $s_i = 0.1 \cdot i$, *za* $i = 0, 1 \ldots, 10$.
- *Kolokacijske točke neka su* $x_i = i/12$, *za* $i = 1, \ldots, 11$.
- *Pomoću funkcije* `odj_kolokacija_kBs_Ab()` *izračunajte* A *i* b.
- *Riješite sustav* $Ax = b$, *pri čemu je* $alpha = A^{-1}b$.

Zadatak (nastavak)

- *Vektoru alpha dodajte $\alpha_0 = 0$ na početak, i $\alpha_{n+2} = 0$ na kraj (da bi funkcionira de Boorov algoritam).*
- *Definirajte funkciju*

$$v(x) = \mathtt{deBoor}(x, t, 3, alpha).$$

- *Definirajte funkciju*

$$y(x) = \frac{e^{-20}}{1 + e^{-20}} e^{20x} + \frac{1}{1 + e^{-20}} e^{-20x} - \cos^2(\pi x),$$

koja predstavlja egzaktno rješenje danog rubnog problema.

Zadatak (nastavak)

- *Na istoj slici nacrtajte grafove funkcija $y(x)$ i $v(x)$ u različitim bojama, sa prikladnom legendom. Koristite MATLAB-ovu funkciju `fplot()` u oba slučaja.*
- *Izračunajte maksimalnu grešku u kolokacijskim točkama.*

Metoda konačnih diferencija

- Osnovna ideja ove metode je zamjena derivacija u ODJ pogodnim kvocijentima diferencija, i rješavanje dobivenih diskretnih jednadžbi.
- Metodu ćemo ilustrirati na sljedećem rubnom problemu drugog reda za $y : [a, b] \to \mathbb{R}$:

$$-y'' + q(x)y = g(x)$$

$$y(a) = \alpha, \quad y(b) = \beta.$$

- Pod pretpostavkama da

$$\begin{aligned} &q, g \in C[a, b] \\ &q(x) \geq 0, \qquad\qquad x \in [a, b] \end{aligned}$$

može se pokazati da gornji rubni problem ima jedinstveno rješenje $y(x)$.

- Kako bismo diskretizirali ODJ, segment $[a, b]$ dijelimo u $n+1$ jednakih podintervala

$$a = x_0 < x_1 < \cdots < x_n < x_{n+1} = b, \quad x_j = a+jh, \; h = \frac{b-a}{n+1},$$

  i označavamo $y_j = y(x_j)$, $y'_j = y'(x_j)$ i $y''_j = y''(x_j)$, za $j = 1, \ldots, n$.
- 1. derivaciju možmo aproksimirati na dva načina:
  - **diferencijom unazad**

$$y'_j \approx \Delta_- y_j = \frac{y_j - y_{j-1}}{h},$$

  - **diferencijom unaprijed**

$$y'_j \approx \Delta_+ y_j = \frac{y_{j+1} - y_j}{h}.$$

- Druga derivacija se najčešće aproksimira kombinacijom ovih dviju diferencija, i tako dobivena aproksimacija zove se **centralna diferencija**:

$$y_j'' \approx \Delta^2 y_j = \frac{\frac{y_{j+1}-y_j}{h} - \frac{y_j-y_{j-1}}{h}}{h} =$$
$$= \frac{y_{j+1} - 2y_j + y_{j-1}}{h^2}.$$

- Sada ćemo procijeniti grešku

$$\tau_j(y) = y''(x_j) - \Delta^2 y_j.$$

- Pretpostavit ćemo da je $y \in C^4[a,b]$.
- Tada iz Taylorovog razvoja od $y(x_j \pm h)$ oko x_j dobivamo

$$y_{j\pm 1} = y_j \pm h y_j' + \frac{h^2}{2!} y_j'' \pm \frac{h^3}{3!} y_j''' + \frac{h^4}{4!} y^{(4)}(x_j \pm \theta_j^{\pm} h),$$
$$0 < \theta_j^{\pm} < 1.$$

- Zbog toga je

$$\Delta^2 y_j = y_j'' + \frac{h^2}{24}[y^{(4)}(x_j + \theta_j^+ h) + y^{(4)}(x_j - \theta_j^- h)].$$

- Budući da je $y^{(4)}$ još uvijek neprekidna, slijedi da je

$$\tau_j(y) = y''(x_j) - \Delta^2 y_j = -\frac{h^2}{12} y^{(4)}(x_j + \theta_j h),$$

za neki $|\theta_j| < 1$.

- Iz danog rubnog problema slijedi da $y_j = y(x_j)$ zadovoljavaju jednadžbe

$$\begin{aligned} y_0 &= \alpha, \\ \frac{-y_{j-1} + 2y_j - y_{j+1}}{h^2} + q(x_j)y_j &= g(x_j) + \tau_j(y), \quad j = 1, \ldots, n, \\ y_{n+1} &= \beta. \end{aligned}$$

- Uz oznake $q_j = q(x_j)$ i $g_j = g(x_j)$, i definicije vektora

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \boldsymbol{\tau}(y) = \begin{bmatrix} \tau_1(y) \\ \tau_2(y) \\ \vdots \\ \tau_n(y) \end{bmatrix}, \quad c = \begin{bmatrix} g_1 + \frac{\alpha}{h^2} \\ g_2 \\ \vdots \\ g_{n-1} \\ g_n + \frac{\beta}{h^2} \end{bmatrix},$$

i simetrične $n \times n$ tridijagonalne matrice

$$A = \frac{1}{h^2} \begin{bmatrix} 2 + q_1 h^2 & -1 & & 0 \\ -1 & 2 + q_2 h^2 & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 + q_n h^2 \end{bmatrix}$$

prethodne jednadžbe ekvivalentne su matričnoj jednadžbi

$$A\mathbf{y} = c + \tau(y).$$

- Metoda konačnih diferencija se sastoji u tome da se iz gornje jednadžbe izbaci izraz $\tau(y)$, pa se prema tome traži rješenje $u = [\ u_1 \ \cdots u_n\ ]^T$ sustava linearnih jednadžbi

$$Au = c,$$

kao aproksimacija za **y**.

Teorem

Ako je $q_j \geq 0$ za $j = 1, \ldots, n$ tada je matrica A pozitivno definitna, i vrijedi $0 \leq A^{-1} \leq A_0^{-1}$ (nejednakost je po elementima), gdje je A_0 pozitivno definitna $n \times n$ matrica oblika

$$A_0 = \frac{1}{h^2} \begin{bmatrix} 2 & -1 & & 0 \\ -1 & 2 & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{bmatrix}.$$

- Iz ovog teorema slijedi da sustav $Au = c$ ima jedinstveno rješenje za $q(x) \geq 0$ i $x \in [a, b]$, koje se jednostavno može izračunati metodom Choleskog sa malim brojem operacija ili metodom konjugiranih gradijenata.

Teorem

Neka dani rubni problem ima rješenje $y(x) \in C^4[a,b]$, i neka je $|y^{(4)}(x)| \leq M$ za $x \in [a,b]$. Neka je još $q(x) \geq 0$ za $x \in [a,b]$, i $u = [\ u_1 \ \ \cdots u_n\ ]^T$ neka je rješenje sustava $Au = c$. Tada za $i = 1, \ldots, n$ vrijedi

$$|y(x_i) - u_i| \leq \frac{Mh^2}{24}(x_i - a)(b - x_i).$$

- Uz pretpostavke teorema greška metode teži ka 0 kao h^2, dakle metoda je reda 2.

Zadaci

Zadatak

Napišite M-file funkciju `odj_diferencije_Ac()` *koja računa matricu A i vektor c za metodu konačnih diferencija. Funkcija neka ima ulazne parametre*

- *pokazivače na funkcije q i g*
- *rubove segmenta a i b*
- *rubne uvjete α i β*
- *vektor x koji sadrži rubove podsegmenata x_i $i = 0, \ldots, n+1$*

i izlazne parametre

- *matricu A*
- *vektor c*

Zadatak

Pomoću metode konačnih diferencija riješite sljedeći rubni problem

$$-y'' + 400y = -400\cos^2(\pi x) - 2\pi^2\cos(2\pi x),$$

$$y(0) = y(1) = 0.$$

- *Rubovi podintervala neka su zadani sa* $x_i = 0.1 \cdot i$, *za* $i = 0, 1 \ldots, 10$.
- *Pomoću funkcije* `odj_diferencije_Ac()` *izračunajte A i c.*
- *Riješite sustav* $Au = c$.

Zadatak (nastavak)

- *Vektoru u dodajte* α *na početak, i* β *na kraj.*
- *Definirajte funkciju*

$$y(x) = \frac{e^{-20}}{1+e^{-20}} e^{20x} + \frac{1}{1+e^{-20}} e^{-20x} - \cos^2(\pi x),$$

  koja predstavlja egzaktno rješenje danog rubnog problema.
- *Na istoj slici nacrtajte graf funkcije* $y(x)$ *i točke* u_i*,* $i = 0, \ldots, 10$ *u različitim bojama, sa prikladnom legendom.*
- *Izračunajte maksimalnu grešku* $\max_i |u_i - y(x_i)|$.

Primjeri iz primjene: Distribucija temperature unutar cilindra

Primjer

- *Temperatura stacionarnog stanja unutar cilindra radijusa 1 opisana je kao rješenje $y(x;\alpha)$ nelinearnog rubnog problema*

$$y'' = -\frac{y'}{x} - \alpha e^y,$$

$$y'(0) = y(1) = 0.$$

- *Ovdje je α "prirodni" parametar definiran kao*

$$\alpha = \frac{\text{generiranje topline}}{\text{konduktivitet}}, \quad 0 < \alpha \leq 0.8.$$

Zadatak

Riješite ovaj primjer metodom gađanja sa $s_0 = 1$. Potrebno je samo malo preraditi metodu `odj_gadjanje_2rru()`.

Primjer (nastavak)

- *Ovaj zadatak, ovako zadan, u principu ne da se riješiti metodom gađanja jer funkcija desne strane ODJ ima singularitet u* $x = 0$.
- *Iako je rješenje* $y(x; \alpha)$ *analitička funkcija na cijelom segmentu* $[0, 1]$, *kod rješavanja rubnog problema dolazi do problema u konvergenciji.*
- *Razlog gubitka konvergencije metode gađanja nije u samoj metodi već u polaznom rubnom problemu.*
- *Međutim, sa malo lukavstva ovaj problem se može izbjeći tako da rješenje* $y(x)$ *razvijemo u Taylorov red oko* $x = 0$:

$$y(x) = y(0) + \frac{x^2}{2!}y^{(2)}(0) + \frac{x^3}{3!}y^{(3)}(0) + \frac{x^4}{4!}y^{(4)}(0) + \cdots ,$$

jer je $y'(0) = 0$.

Primjer (nastavak)

- *Koeficijenti $y^{(i)}(0)$, $i = 2, 3, 4, \ldots$ mogu se izraziti preko izraza $\lambda = y(0)$ koji predstavlja nepoznatu konstantu.*
- *Iz ODJ rubnog problema slijedi*

$$y^{(2)}(x) = -\left(y^{(2)}(0) + \frac{x}{2!}y^{(3)}(0) + \frac{x^2}{3!}y^{(4)}(0) + \cdots\right) - \alpha e^{y(x)},$$

a kada pustimo $x \to 0$ dobivamo

$$y^{(2)}(0) = -y^{(2)}(0) - \alpha e^{y(0)},$$

odnosno

$$y^{(2)}(0) = -\frac{1}{2}\alpha e^{\lambda}.$$

Primjer (nastavak)

- *Daljnje deriviranje daje*

$$y^{(3)}(x) = -\left(\frac{1}{2}y^{(3)}(0) + \frac{x}{3}y^{(4)}(0) + \cdots\right) - \alpha y'(x)e^{y(x)},$$

tako da je

$$y^{(3)}(0) = 0.$$

- *Još jedanputa deriviramo i ostaje nam*

$$y^{(4)}(x) = -\left(\frac{1}{3}y^{(4)}(0) + x(\cdots)\right) - \alpha\left[\left(y'(x)\right)^2 + y^{(2)}(x)\right] e^{y(x)},$$

odakle je

$$y^{(4)}(0) = \frac{3}{8}\alpha^2 e^{2\lambda}.$$

Primjer (nastavak)

- *Može se pokazati da je* $y^{(5)}(0) = 0$*, i općenito* $y^{(2i+1)}(0) = 0$.
- *Polazni rubni problem možemo sada preformulirati na sljedeći način:*
  - *u nekoj okolini od* $x = 0$ *koristi se reprezentacija Taylorovim redom,*
  - *a na dovoljnoj udaljenosti od singulariteta* $x = 0$ *koristi se polazna ODJ samog problema.*
- *Kako iz Taylorovog reda slijedi*

$$y''(x) = y^{(2)}(0) + xy^{(3)}(0) + \frac{x^2}{2}y^{(4)}(0) + \cdots ,$$

polazni rubni problem sada možemo aproksimirati problemom

Primjer (nastavak)

$$y''(x) = \begin{cases} \frac{1}{2}\alpha e^{\lambda}\left(1 - \frac{3}{8}x^2\alpha e^{\lambda}\right), & \textit{ako je } 0 \le x \le 10^{-2}, \\ -\frac{y'(x)}{x} - \alpha e^{y(x)}, & \textit{ako je } 10^{-2} \le x \le 1. \end{cases}$$

- *Za $x \le 10^{-2}$ greška u funkciji desne strane ODJ je reda veličine 10^{-8}.*
- *Sada desna strana još uvijek sadrži nepoznati parametar $\lambda = y(0)$.*
- *Može se pokazati da se ovaj problem može interpretirati kao prošireni rubni problem s 3 jednadžbe:*
  - *za supstituciju*

$$\begin{aligned} y_1(x) &= y(x), \\ y_2(x) &= y'(x), \\ y_3(x) &= y(0) = \lambda, \end{aligned}$$

Primjer (nastavak)

dobivamo sustav ODJ za $0 \leq x \leq 1$

$$
\begin{aligned}
y_1' &= y_2, \\
y_2' &= \begin{cases} \frac{1}{2}\alpha e^{y_3}\left(1 - \frac{3}{8}x^2\alpha e^{y_3}\right), & \textit{ako je } 0 \leq x \leq 10^{-2}, \\ -\frac{y_2}{x} - \alpha e^{y_1}, & \textit{ako je } 10^{-2} \leq x \leq 1, \end{cases} \\
y_3' &= 0,
\end{aligned}
$$

sa rubnim uvjetima

$$
r = \begin{bmatrix} y_2(0) \\ y_1(1) \\ y_3(0) - y_1(0) \end{bmatrix} = 0.
$$

Zadatak

Riješite rubni problem na prethodno opisani način za $\alpha = 0.8$.

- *Napišite M-file funkciju* `odj_primjer_distr_temp_f3()` *koja implementira funkciju desne strane ODJ.*
- *Riješite rubni problem pomoću metode gađanja, ali tako da s bude aproksimacija samo za $y(0)$, pri tome će inicijalni uvjet za Runge-Kutta metodu biti* $\begin{bmatrix} s & 0 & s \end{bmatrix}^T$.
- *$F(s)$ je samo druga komponenta od $r(s, y(b; s))$, jer ona jedina definira uvjet u krajnjem rubu $x = 1$.*
- *Ispišite optimalni s i nacrtajte graf rješenja.*