Matematično-fizikalni praktikum 2025/26

6. naloga: Enačbe hoda

Samo Krejan, 28231092

Za opis najpreprostejših fizikalnih procesov uporabljamo navadne diferencialne enačbe, ki povezujejo vrednosti spremenljivk sistema z njihovimi časovnimi spremembami. Tak primer je na primer enačba za časovno odvisnost temperature v stanovanju, ki je obdano s stenami z neko toplotno prevodnostjo in določeno zunanjo temperaturo. V najpreprostejšem primeru ima enačba obliko

$$\frac{\mathrm{d}T}{\mathrm{d}t} = -k\left(T - T_{\mathrm{zun}}\right) \tag{1}$$

z analitično rešitvijo

$$T(t) = T_{\mathrm{zun}} + \mathrm{e}^{-kt}\left(T(0) - T_{\mathrm{zun}}\right) \ .$$

Enačbam, ki opisujejo razvoj spremenljivk sistema y po času ali drugi neodvisni spremenljivki x, pravimo *enačbe hoda*. Pri tej nalogi bomo proučili uporabnost različnih numeričnih metod za reševanje enačbe hoda oblike $\mathrm{d}y/\mathrm{d}x = f(x, y)$, kot na primer (1). Najbolj groba prva inačica, tako imenovana osnovna Eulerjeva metoda, je le prepisana aproksimacija za prvi odvod $y' \approx (y(x+h) - y(x))/h$, torej

$$y(x+h) = y(x) + h \left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_x \ . \tag{2}$$

Diferencialno enačbo smo prepisali v diferenčno: sistem spremljamo v ekvidistantnih korakih dolžine h. Metoda je večinoma stabilna, le groba: za večjo natančnost moramo ustrezno zmanjšati korak. Za red boljša ($\mathcal{O}\left(h^3\right)$, t.j. lokalna natančnost drugega reda) je simetrizirana Eulerjeva (ali sredinska) formula, ki sledi iz simetriziranega približka za prvi odvod, $y' \approx (y(x+h) - y(x-h))/2h$. Računamo po shemi

$$y(x+h) = y(x-h) + 2h \left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_x \ , \tag{3}$$

ki pa je praviloma nestabilna. Želeli bi si pravzaprav nekaj takega

$$y(x+h) = y(x) + \frac{h}{2}\left[\left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_x + \left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_{x+h}\right] \ , \tag{4}$$

le da to pot ne poznamo odvoda v končni točki intervala (shema je implicitna). Pomagamo si lahko z iteracijo. Zapišimo odvod kot:

$$\left.\frac{dy}{dx}\right|_x = f(x, y)$$

ter

$$x_{n+1} = x_n + h, \quad y_n = y(x_n)$$

Heunova metoda ($\mathcal{O}\left(h^3\right)$ lokalno) je približek idealne formule z:

$$\hat{y}_{n+1} = y_n + h \cdot f(x_n, y_n) \tag{5}$$

$$y_{n+1} = y_n + \frac{h}{2}\left[f(x_n, y_n) + f(x_{n+1}, \hat{y}_{n+1})\right] \tag{6}$$

Izvedenka tega je nato Midpoint metoda (tudi $\mathcal{O}\left(h^3\right)$ lokalno):

$$k_1 = f(x_n, y_n) \tag{7}$$
$$k_2 = f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}\,h\,k_1) \tag{8}$$
$$y_{n+1} = y_n + h\,k_2 \tag{9}$$

Le-to lahko potem izboljšamo kot modificirano Midpoint metodo itd...

V praksi zahtevamo natančnost in numerično učinkovitost, ki sta neprimerno boljši kot pri opisanih preprostih metodah. Uporabimo metode, zasnovane na algoritmih prediktor-korektor, metode višjih redov iz družine Runge-Kutta (z adaptivnimi koraki), ali ekstrapolacijske metode. Brez dvoma ena najbolj priljubljenih je metoda RK4,

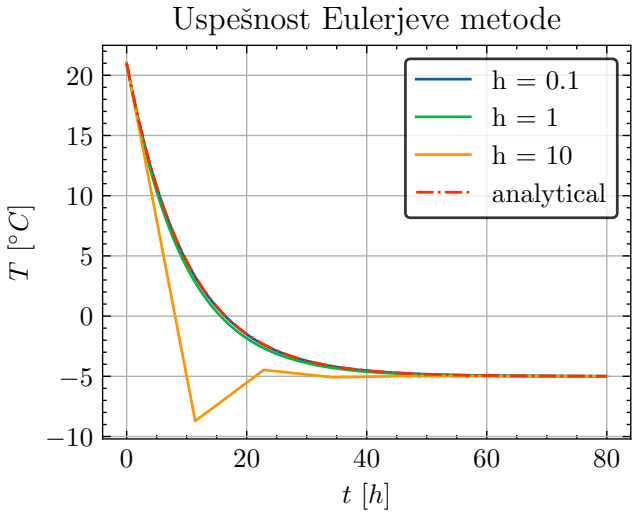
$$\begin{aligned} k_1 &= f\left(x,\, y(x)\right)\,, \\ k_2 &= f\left(x + \tfrac{1}{2}h,\, y(x) + \tfrac{h}{2}k_1\right)\,, \\ k_3 &= f\left(x + \tfrac{1}{2}h,\, y(x) + \tfrac{h}{2}k_2\right)\,, \\ k_4 &= f\left(x + h,\, y(x) + hk_3\right)\,, \\ y(x+h) &= y(x) + \tfrac{h}{6}\,\left(k_1 + 2k_2 + 2k_3 + k_4\right) + \mathcal{O}(h^5)\,. \end{aligned} \tag{10}$$

1 Naloga

Preizkusi preprosto Eulerjevo metodo ter nato še čim več naprednejših metod(Midpoint, Runge-Kutto 4. reda, Adams-Bashfort-Moultonov prediktor-korektor ...) na primeru z začetnima temperaturama $y(0) = 21$ ali $y(0) = -15$, zunanjo temperaturo $y_{\text{zun}} = -5$ in parametrom $k = 0.1$. Kako velik (ali majhen) korak h je potreben? Izberi metodo (in korak) za izračun družine rešitev pri različnih vrednostih parametra k.

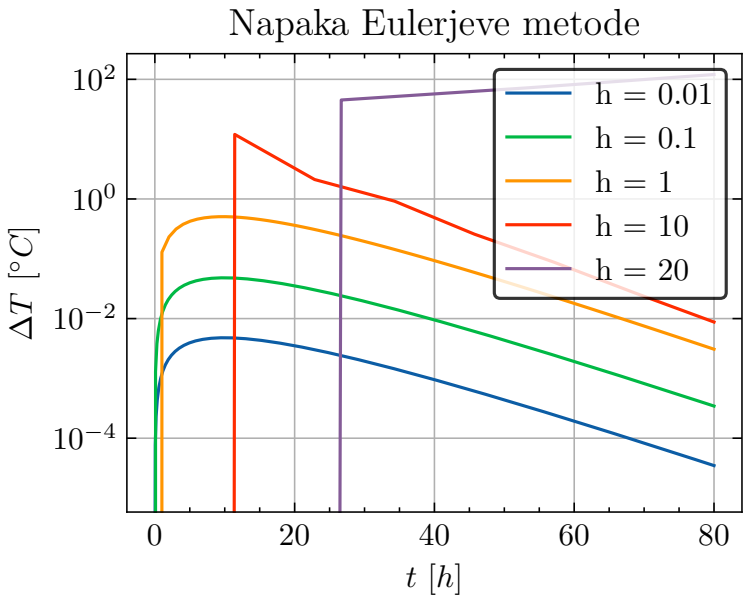
1.1 Eulerjeva metoda

Naloge sem se lotil tako, da sem najprej najbolj preprosto izmed metod - Eulerjevo metodo, implementiral v Rust jeziku in se nato lotil analize uspešnosti metode. Metoda je seveda bolša, tem manjši h (kot definiran v uvodu) vzamemo. Za občutek, kako majhnega je res treba vzeti, sem najprej zgrafiral rešitve naše enačbe za različne h in tudi analitično rešitev - glej sliko 1



Slika 1: Osnovna Eulerjeva metoda za različno dolge korake

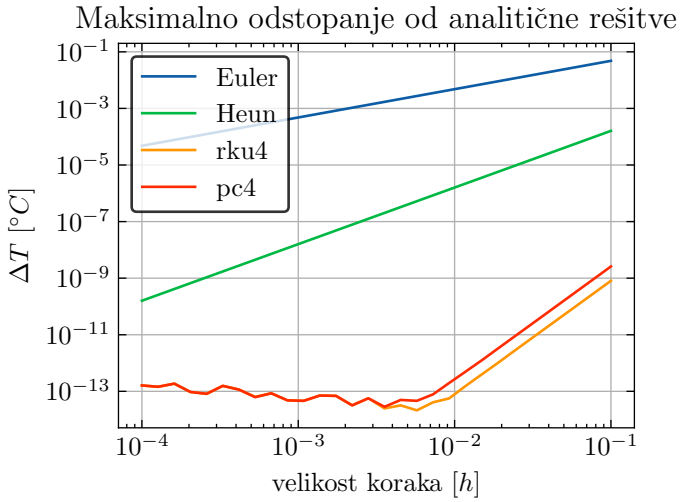
Hitro vidimo, da se rešitev enačbe s 6 minutnim korakom na oko že odlično sklada z analitično rešitev. Ker imamo to srečo, da imamo na voljo analitično rešitev problema, lahko grafiramo tudi napako metode pri različnih korakih - glej sliko 2



Slika 2: Absolutna napaka Eulerjeve metode za različno dolge korake, opazimo da metoda za 20 urni korak ni stabilna, in se napaka metode čez čas le nabira in veča

1.2 Ostale metode

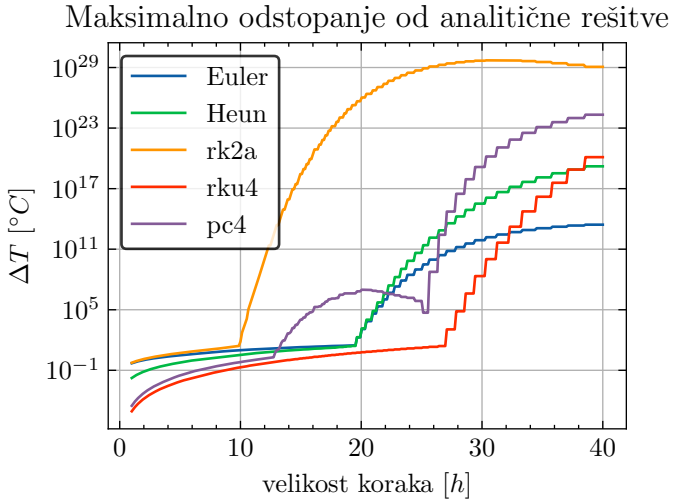
Za dokaj preprosto implementacijo eulerjeve metode sem se lotil implementacije tudi drugih metod. Tu sem si močno pomagal z že narejeno implementacijo metod v Pythonu, ki je objavljena v spletni učilnico. Pravzaprav sem jo le prepisal v Rust in preuredil v željeno obliko. Najprej sem se na hitro prepričal, da so metode delovale pravilno, nato pa sem se lotil raziskovanja, katera je najbolj natančna v odvisnosti od dolžine koraka. Vsako izmed metod sem uporabil na enakem časovnem intervalu in nato iskal največjo razliko med metodo in analitično rešitvijo. Na ta način sem dobil graf 3



Slika 3: Maksimalno odstopanje različnih metod v odvisnosti od koraka

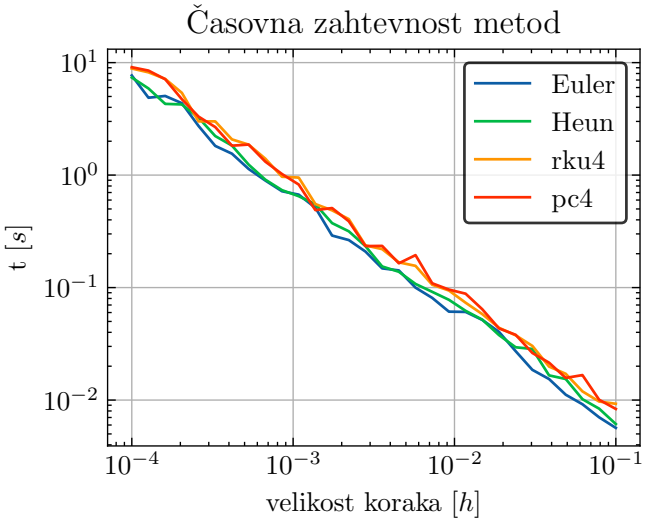
Vidimo, da sta daleč najnatančnejši metodi Runge-Kutta 4. reda (rku4) in Adams - Bashfort - Moultonov prediktor-korektor (pc4), ki oba že pri koraku dolžine $10^{-2}h$ dosegata tevretične limite natančnosti. Runge-Kutta se izkaže celo za malo bolj natančno. Heunova metoda pa po natančnosti zadeva ravno nekje vmes med omenjenima metodama in med najosnovnejšo Eulerjevo metodo. Runge-Kutta 2. reda nisem narisal, saj se je njena natančnost na danem intervalu skoraj popolnoma skladala z Eulerjevo.

Na tej točki sem se seveda osredotočil zgolj na sorazmerno majhne korake, a seveda veliko o uspešnosti metode pove tudi, kakšen je največji korak, pri katerem je metoda še stabilna. S tem namenom je nastal graf 4



Slika 4: Maksimalno odstopanje različnih metod v odvisnosti od koraka. Na prelomih krivulj pride do meje med stabilno in nestabilno metodo. Vidimo, da je v tem smislu najstabilnejša metoda Runge-Kutta 4. reda, medtem ko me je Eulerjeva metoda pozitivno presenetila s svojo stabilnostjo. Opazimo tudi zanimivo obliko na grafu pc4, ki je ne znam pojasniti

Seveda pa metoda ni dobra, če porabi izračun veliko časa. Tako sem se odločil narisati tudi koliko časa porabi določena metoda (slika 5), kjer sem meril čas, kot povprečje izvedbe metode 100 krat, kar je kar močno segrelo moj računalnik.

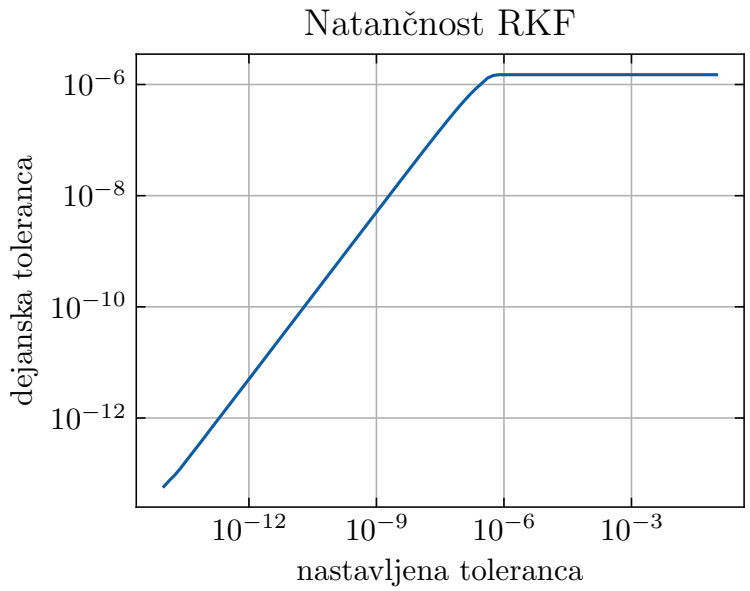


Slika 5: Čas izvedbe metod v odvisnosti od dolžine koraka

Po pričakovanjih so seveda preprostejše metode hitrejše pri enaki dolžini koraka, a v splošnem sem preveril da (približno) velja, da metoda rku4 doseže enako natančnost v manjšem času, tako da izmed naštetih metod, mislim da rku4 žmaga"na vseh področjih, je najnatančnejša in pri poljubni natančnosti tudi najhitrejša.

1.2.1 Runge-Kutta-Fehlbergova metoda,

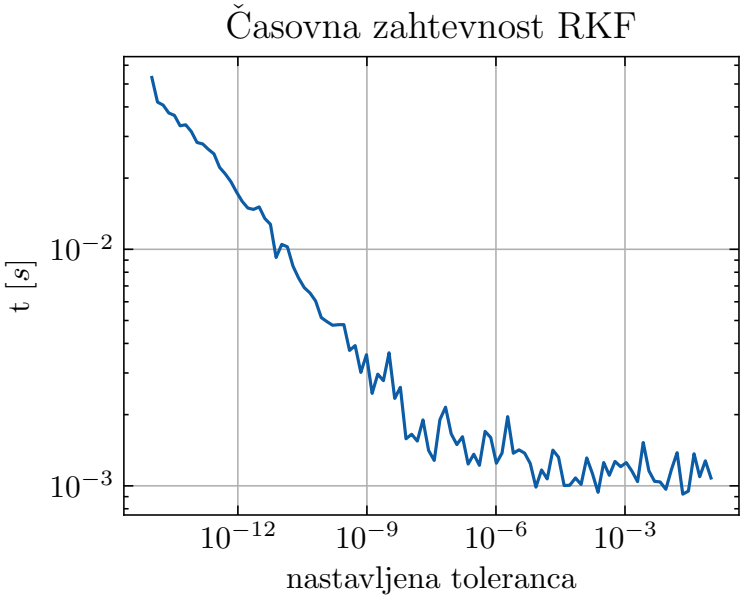
je metoda za katero se mi je zdelo primerno porabiti še nekaj dodatnega časa, saj deluje po nekoliko drugačnem principu kot ostale. Namreč, namesto da ročno nastavimo dolžino koraka, se ta avtomatično prilagaja tako, da doseže željeno natančnost. Prvo vprašanje, ki se mi je porodilo je bilo seveda, če to dela pravilno. Tako sem na enak način kot v prejšnem poglavju meril natančnost, a tokrat v odvisnosti od nastavljene natančnosti. Tako sem dobil graf 6



Slika 6: Dejanska natančnost v odvisnosti od nastavljene / željene natančnosti.

Vidimo, da je dejanska natančnost nekoliko slabša od nastavljene, a se giblje v željenem velikostnem redu. Opazimo tudi, da se na neki točki natančnost ne slabša več, razlog za čemer pa je ta, da ima metoda nastavljen maksimalno dolžino koraka, in ko zahtevana natančnost ne zahteva več manjših korakov, RKF metoda postane pravzaprav rku4.

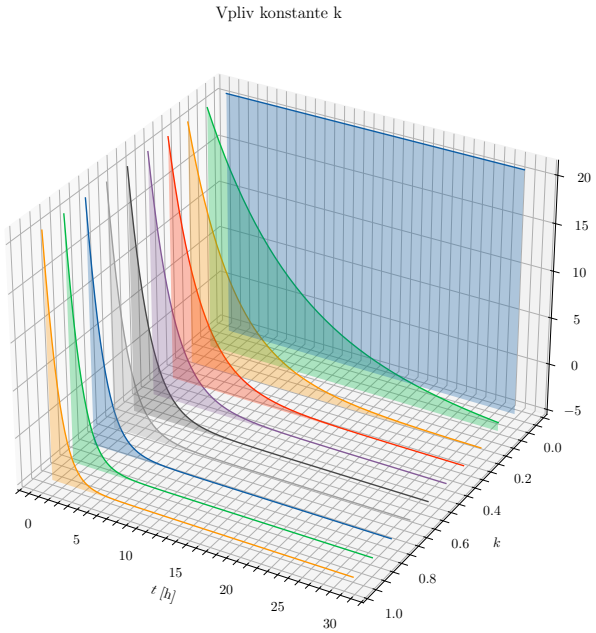
Kot prej nas seveda zanima tudi časovna zahtevnost metode, ki sem jo prikazal na grafu 7. Izkaže se, da metoda ni le natančna temveč tudi hitra, kar je smiselno, saj dinamično prilagaja natančnost, s čimer poskrbi, da ne porabi časa na nepotrebnih mestih.



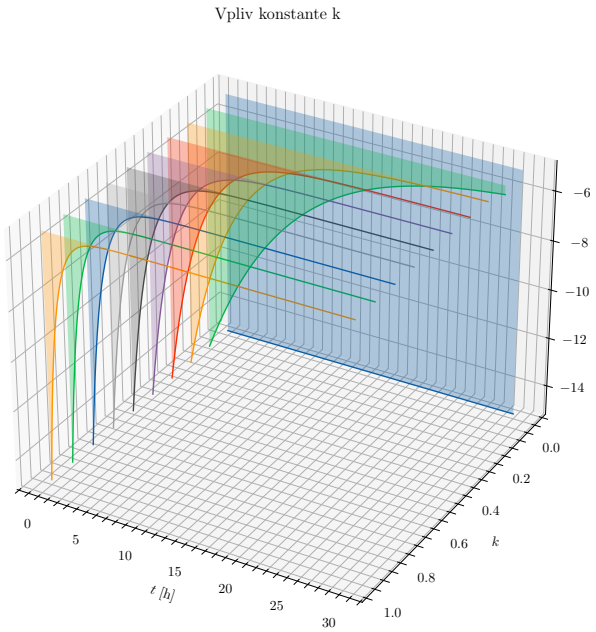
Slika 7: Časovna zahtevnost RKF

1.3 Družina rešitev

Ko sem se končno prepričal, da mi je najbolj všeč RKF metoda sem z njo izračunal družino rešitev našega problema pri različnih vrednostih parametra k. Dobljeno sem prikazal na grafih 8 in 9



Slika 8: Rešitve problema za različne vrednosti parametra k pri začetni temperaturi $21°C$



Slika 9: Rešitve problema za različne vrednosti parametra k pri začetni temperaturi $-15°C$

2 Dodatna naloga

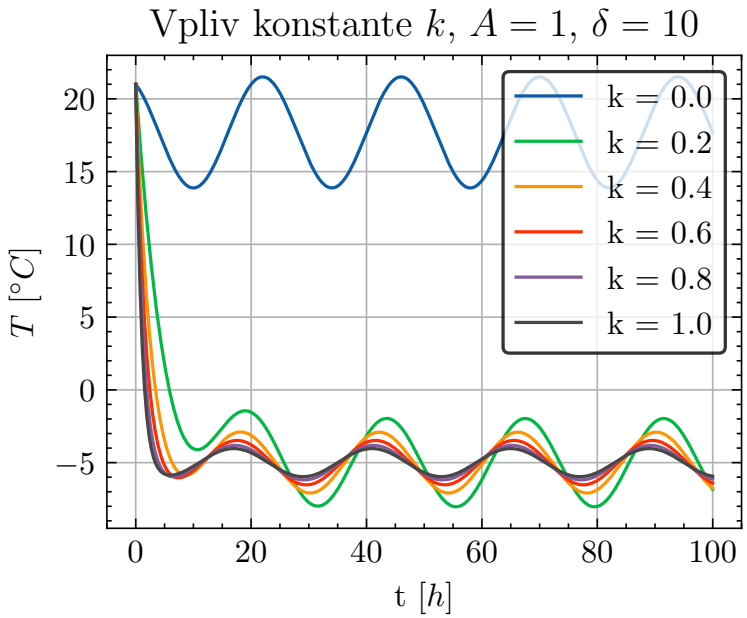
Temperatura prostora se lahko še dodatno spreminja zaradi denimo sončevega segrevanja skozi okna, s 24-urno periodo in nekim faznim zamikom δ, kar opišemo z dva- ali triparametrično enačbo

$$\frac{\mathrm{d}T}{\mathrm{d}t} = -k\,(T - T_{\mathrm{zun}}) + A \sin\left(\frac{2\pi}{24}(t-\delta)\right) . \tag{11}$$

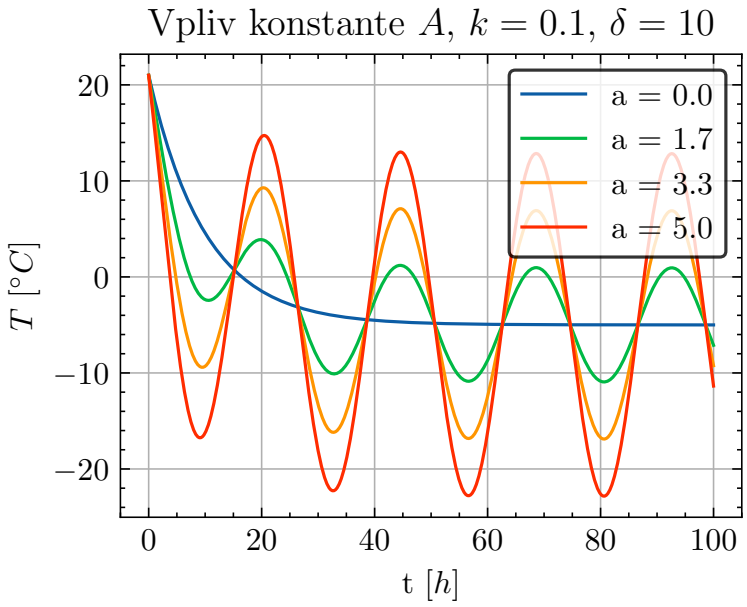
Poišči še družino rešitev te enačbe pri $k = 0.1$ in $\delta = 10$! Začni z $A = 1$, kasneje spreminjaj tudi to vrednost. V premislek: kakšno metodo bi uporabil, če bi posebej natančno želel določiti maksimalne temperature in trenutke, ko nastopijo?

2.1 Družine rešitev

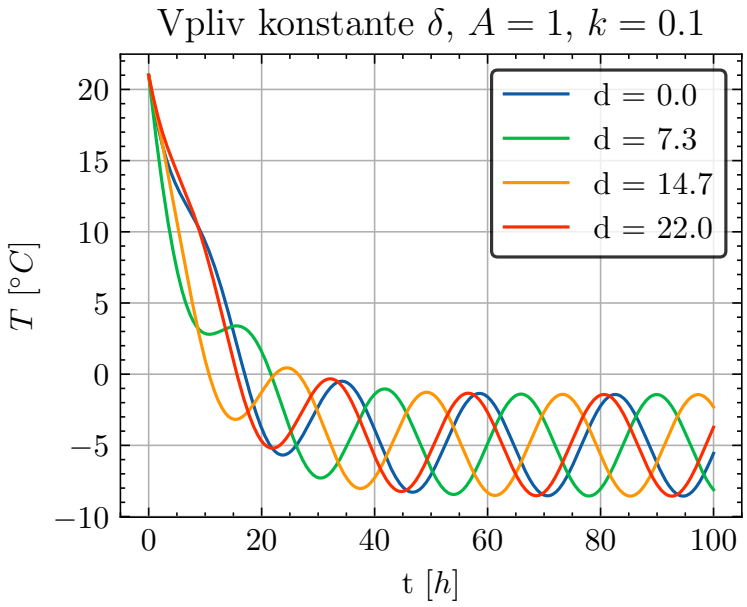
Za določanje maksimumov temperatur, mislim da se najbolj splača uporabiti prav RKF metodo, ki okoli točk z visokimi drugimi odvodi (torej v našem primeru prav okoli maksimumov) prilagodi interval, tako da je le ta na teh točkah krajši. S tem v mislih, sem tako izbral prav to metodo, za iskanje družine rešitev našega dodatno zakompliciranega problema. Po vrsti sem najprej spreminjal parameter k nato pa še a in d, medtem ko sem vse ostale parametre držal konstantne. Na ta način sem dobil slike 10, 11 in 12. Mislim, da se na grafu opazi marsikatero lastnost, ki je z nekaj fizikalnega razmisleka lahko pojasnljiva.



Slika 10: Temperatura skozi čas



Slika 11: Temperatura skozi čas



Slika 12: Temperatura skozi čas

3 Zaključek

Rokovanje z različnimi metodami reševanja diferencialnih enačb se je izkazala za presenetljivo prijetno izkušnjo. Sicer sem o numeričnih metodah reševanja diferencialnih enačb nekaj vedel že od lani, so bile nekatere metode zame nove in so tako ponudile novo navdušenje nad programiranjem.

Poleg novih metod pa sem se tekom naloge spoprijel tudi z drugimi izivi. Najprej mi je mnogo časa vzela bolezen, ki je efektivno uničila možnost dela čez vikend, tako da sem moral z delom nekoliko hiteti (verjetno se pozna v času oddaje). Poleg tega pa sem se končno odločil uporabiti Jupitrov

zvezek namesto preprostega Python dokumenta, kar je omogočilo precej bolj prijetno delo s kodo a je s seboj seveda prineslo nekaj preglavic učenja novega orodja.

Vseskupaj mislim, da je bilo reševanje uspešno tako iz perspektive novih znanj, kot tudi v kvaliteti končnega izdelka. Upam, da se bo bralec strinjal s tem :)