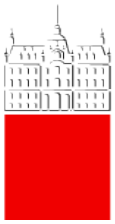
Univerza *v Ljubljani*
Fakulteta za *matematiko in fiziko*



Spektralne metode za začetne probleme PDE

9. naloga pri Matematično-fizikalnem praktikumu

Avtor: Marko Urbanč (28191096)
Predavatelj: prof. dr. Borut Paul Kerševan

1.9.2023

Kazalo

1 Uvod

Parcialne diferencialne enačbe (PDE) lahko rešujemo na več različnih načinov. Glavna razlika je v tem, kako diskretiziramo prostor in čas. V tej nalogi bomo reševali PDE z spektralnimi metodami.(Drugo možnost; diferenčne metode, bomo obravnavali v naslednji nalogi.) Pri spektralnih metodah diskretiziramo prostor s tem, začetni pogoj izrazimo z nekim naborom baznih funkcij in nato iščemo rešitev tako, da računamo, kako se koeficienti teh baznih funkcij spreminjajo s časom. V tej nalogi bomo preizkusili reševanje preko Fourierove metode in metode končnih elementov s kubičnimi B-zlepki.

1.1 Fourierova metoda

Fizikalno gledano rešujemo enodimenzionalno toplotno enačbo, torej difuzijsko enačbo, v homogeni neskončni plasti, s končno debelino a, brez izvirov in ponorov toplote.

$$D\frac{\partial^2 T}{\partial x^2} = \frac{\partial T}{\partial t} \quad 0 \leq x \leq a, \quad D = \frac{\lambda}{\rho c}\,. \tag{1}$$

Če Temperaturo $T(x, y)$ izrazimo kot Fourierovo vrsto dobimo

$$T(x,t) = \sum_{k=0}^{N-1} \hat{T}_k(t) \exp\left(\frac{-2\pi i k x}{a}\right). \tag{2}$$

Torej se PDE (1) zdaj zapiše kot

$$\sum_{k=0}^{N-1}\left(-\frac{4\pi^2 k^2 D}{a^2}\right)\hat{T}_k(t)\exp\left(-\frac{2\pi i k x}{a}\right) = \sum_{k=0}^{N-1}\left(\frac{\mathrm{d}\hat{T}_k(t)}{\mathrm{d}t}\right)\exp\left(-\frac{2\pi i k x}{a}\right). \tag{3}$$

Torej se naša naloga prevede na iskanje koeficientov $\hat{T}_k(t)$, ki jih dobimo preko **evolucijske enačbe**

$$\frac{\mathrm{d}\hat{T}_k(t)}{\mathrm{d}t} = -\frac{4\pi^2 k^2 D}{a^2}\hat{T}_k(t)\,. \tag{4}$$

Pogosto se uporabi spektralno reprezentacijo za krajevni odvod, časovni korak pa naredimo z neko ekspliticitno metodo. V našem primeru bomo uporabili **Eulerjevo metodo**.

$$\hat{T}_k(t+k) = \hat{T}_k(t) + \frac{4\pi^2 k^2 D}{a^2}\hat{T}_k(t)k\,. \tag{5}$$

Reprezentacijo $T(x, y)$ v običajnem prostoru dobimo z obratno Fourierovo transformacijo. Enačba (4) ima analitično rešitev

$$\hat{T}_k(t) = \hat{T}_k(0)\exp\left(-\frac{4\pi^2 k^2 D}{a^2}t\right). \tag{6}$$

To bo koristno za preverjanje numeričnih metod.

1.2 Metoda končnih elementov

Pri razvoju $T(x, y)$ nismo omejeni samo na trigonometrične funkcije. Lahko uporabimo tudi poljubne druge funkcije. V našem primeru bomo uporabili kubične B-zlepke. To so funkcije oblike

$$B_{i,k}(x) = \frac{x - x_i}{x_{i+k} - x_i} B_{i,k-1}(x) + \frac{x_{i+k+1} - x}{x_{i+k+1} - x_{i+1}} B_{i+1,k-1}(x) \,. \tag{7}$$

Začetni pogoj bomo izrazili kot linearno kombinacijo teh funkcij in nato iščemo rešitev v obliki

$$T(x,t) = \sum_{i=-1}^{N+1} c_k(t) B_{i,3}(x) \,. \tag{8}$$

Tako zasnujemo metodo končnih elementov, s kolokacijskim pogojem. To pomeni, da se začetni pogoj mora ujemati z rešitvijo na nekem končnem številu točk. Podobno kot pri Fourierovi metodi vstavimo razvoj v PDE in dobimo sistem enačb za koeficiente $\hat{T}_i(t)$

$$\sum_{i=-1}^{N+1} D c_k(t) B_{i,3}(x) = \sum_{i=-1}^{N+1} \left(\frac{\partial c_k(t)}{\partial t} \right) B_{i,3}(x) \,. \tag{9}$$

Uporabimo lastnosti B-zlepkov in dobimo sistem diferencialnih enačb za koeficiente $c_k(t)$

$$\dot{c}_{j-1}(t) + 4\dot{c}_j(t) + \dot{c}_{j+1}(t) = \frac{6D}{h^2}(c_{j-1}(t) - 2c_j(t) + c_{j+1}(t)) \,, \tag{10}$$

kjer je h razdalja med točkami, ki smo jih izbrali za kolokacijski pogoj. Iz robnega pogoja pri $x = 0$ ugotovimo, da je $c_{-1} = -4c_0 - c_1$. Podobno iz robnega pogoja pri $x = a$ sledi $c_0 = C_N = 0$ in $c_{-1} = -c_1$ ter $c_{N-1} = -c_{N+1}$. Reševanje smo torej prevedli na reševanje matričnega sistema

$$\mathbf{A} \frac{\mathrm{d}\vec{c}}{\mathrm{d}t} = \mathbf{B}\vec{c} \,, \tag{11}$$

kjer je $\mathbf{A}$ tridiagonalna matrika z 4 na diagonali in 1 na pod in nad diagonalo

$$\mathbf{A} = \begin{bmatrix} 4 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 1 & 4 & 1 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 4 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 4 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 4 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 4 \end{bmatrix} \,, \tag{12}$$

$\mathbf{B}$ tridiagonalna matrika z -2 na diagonali in 1 na pod in nad diagonalo pomnožena z $6D/h^2$

$$\mathbf{B} = \frac{6D}{h^2}\begin{bmatrix} -2 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 1 & -2 & 1 & \cdots & 0 & 0 & 0 \\ 0 & 1 & -2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -2 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 1 & -2 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 & -2 \end{bmatrix}, \tag{13}$$

in $\vec{c}$ vektor koeficientov $c_k(t)$. Začetni pogoj za PDE je $T(x,0) = f(x)$, torej je začetni približek za kolokacijsko aproksimacijo

$$\mathbf{A}\vec{c} = \vec{f}\,, \tag{14}$$

kjer je $\vec{f}$ vektor $f(x_i)$. To zdaj rešujemo z neko eksplitično metodo. V našem primeru bomo uporabili **Implicitno Eulerjevo metodo** zaradi njene stabilnosti. To pomeni, da za časovni korak k velja

$$\mathbf{A}\frac{\vec{c}_{k+1} - \vec{c}_k}{\Delta t} = \mathbf{B}\vec{c}_{k+1}\,. \tag{15}$$

2 Naloga

Naloga od nas zahteva da v eni razsežnosti rešimo PDE (1) z začetnim pogojem po plasti gaussovsko porazdeljene temperature

$$T(x,0) \propto \exp\left(-\frac{(x-a)^2}{2\sigma^2}\right). \tag{16}$$

Rešiti moramo za periodični robni pogoj $T(0,t) = T(a,t)$ in homogeni Dirichletov robni pogoj $T(0,t) = T(a,t) = 0$ po Fourierevi metodi. Kolokacijsko metodo nato uporabi za reševanje PDE z nehomogenim Dirichletovim robnim pogojem $T(0,t) = T(a,t) = 0$ in istim začetnim pogojem. in primerjamo obe metodi.

3 Opis reševanja

Po (pre)dolgem premoru sem se lotil reševanja naloge, spet na tradicionalen način, torej z uporabo Pythona in knjižnic `numpy` in `scipy`. Za risanje grafov sem uporabil knjižnico `matplotlib`. Zdaj imam tudi nekaj trikov v rokavu, ki sem jih uspel na hitro preizkusiti.

Po tem ko sem se lotil reševanja naloge sem ugotovil, da je bil moj prvoten pristop (opisan v datoteki `src.py`) popolnoma nepraktičen in napačen. Lotil sem se reševanja na novo in sicer z bolj sistematičnim pristopom. Za obe metodi sem napisal razred, ki vsebuje vse potrebne funkcije za reševanje. Poglejmo si to podrobneje.

3.1 SpectralSolver

Razred `SpectralSolver` je namenjen reševanju PDE po Fourierovi metodi. V konstruktorju razreda se inicializirajo vsi potrebni parametri, ki jih potrebujemo za reševanje. To so difuzijska konstanta D, debelina plasti a, število točk v prostoru N, začetni pogoj $f(x)$ in niz časov za katere računamo rešitev. Od tu naprej uporabnik pokliče eno od metod za reševanje PDE. Torej ali `solve_Analytically()` ali `solve_Numerically()`. Prva metoda uporablja analitično rešitev, ki je podana z enačbo (6). Druga metoda pa uporablja `scipy.integrate.odeint()` za reševanje sistema enačb (4).

Po opravljenem reševanju vsebuje razred tudi vse metode za izris raznih grafov. Zna risati te zvezne line plot-e, ki jih uporabljam za risanje temperaturnega profila v odvisnosti od časa. Zna risati "Heatmap"grafe in tudi animacije, ki jih žal ni smiselno dati v statično poročilo. Obstajajo tudi eksperimentalne metode za neke hexbin grafe in 3D grafe, ki jih na koncu nisem uporabil, ampak obstajajo pa, če bi jih kdo rad uporabil.

3.2 ColocationSolver

Podobno, obstaja razred `ColocationSolver`, ki je namenjen reševanju PDE po metodi končnih elementov. Tudi ta razred vsebuje vse potrebne parametre za reševanje, ki se incializira v konstruktorju. Parametri so enaki kot pri prejšnjem razredu. Razred ima tudi dve metode za reševanje PDE. Prva je `solve_Properly()`, ki uporablja `scipy.linalg.solve_banded()`, da reši sistem enačb. Druga metoda je `solve_Manually()`, ki reši sistem enačb z Thomasovim algoritmom. Ta metoda je načeloma precej hitrejša. Razred vsebuje tudi metode za risanje grafov, ki so enake kot pri prejšnjem razredu.

3.3 MPI_Node

Oče mi je dal nasvet, da naj poskusim pri zaključku 1. stopnje faksa biti pragmatičen in se ne ukvarjati z nekimi nepotrebnimi stvarmi. No jaz sem pač nekoliko glup in imam zdaj precej časovno stisko, kako naj bi uspel vse končati... ampak imam pa paralelizirane metode preko MPI-ja heh. (Pri tržni raziskavi te naloge se je izkazalo, da so tudi vsi ostali istega mnenja kot moj oče.)

Razred `MPI_Node` je namenjen paralelizaciji reševanja PDE. V konstruktorju se inicializira MPI komunikator, ki ga uporabljamo za komunikacijo med procesi. V osnovi je to samo ovoj (angl. wrapper) okoli prejšnjih dveh razredov. Vsebuje metode za delitev intervalov podatkov in za pošiljanje podatkov med procesi. Vsebuje tudi metode za risanje grafov, ki so enake kot pri prejšnjih dveh razredih, le da jih opravlja lahko samo "root"proces.

3.4 GatherStatistics

Zato da res pripeljem do konca idejo, da sem nepraktičen, sem napisal še en razred, pravzaprav še en ovoj okoli `MPI_Node`, ki je namenjen temu da zbira

statistiko o času izvajanja. Želel sem dobiti značilne premice v log-log skali, ki kažejo na uspešno paralelizacijo, ampak sem to le deloma uspel.

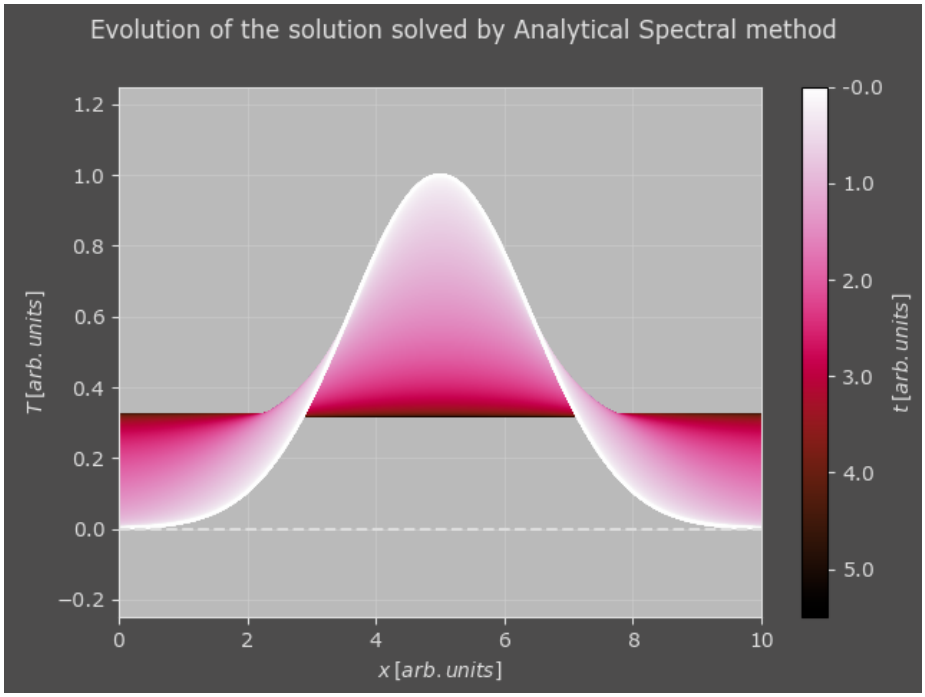
4 Rezultati

Sedaj pa končno rezultati. Ker žal nimam nobenih "grafov 3. reda", kompenziram za pomanjkanje s tem, da so grafi divje pisani.

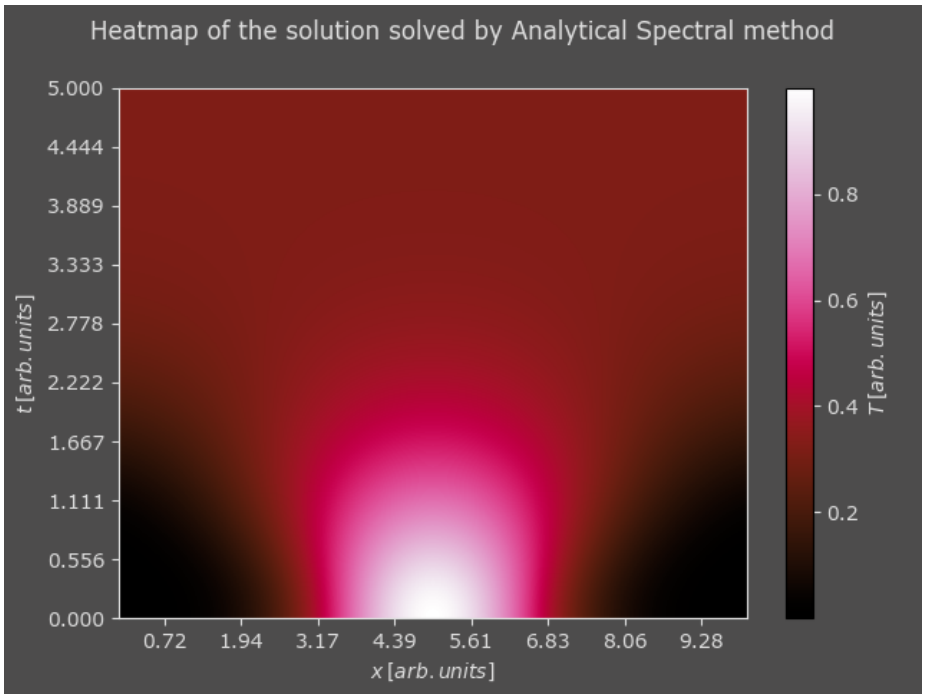
4.1 Fourierova metoda

4.1.1 Periodični robni pogoji

Najprej si poglejmo rezultate za periodične robne pogoje. Slike so rešene za mrežo točk $N = 10000$ in difuzijsko konstanto $D = 1e - 3$. Začetni pogoj je gaussovka z $\sigma = 1$ in $a = 5$, na intervalu $[0, 10]$. Začnimo z analitično rešitvijo. Na sliki 1

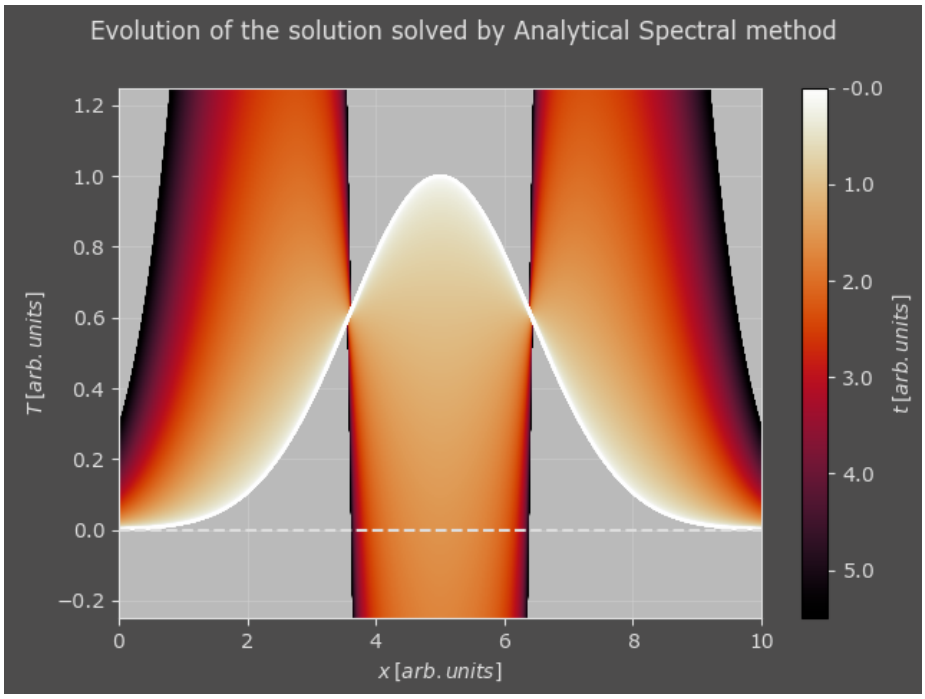


Slika 1: Analitična rešitev

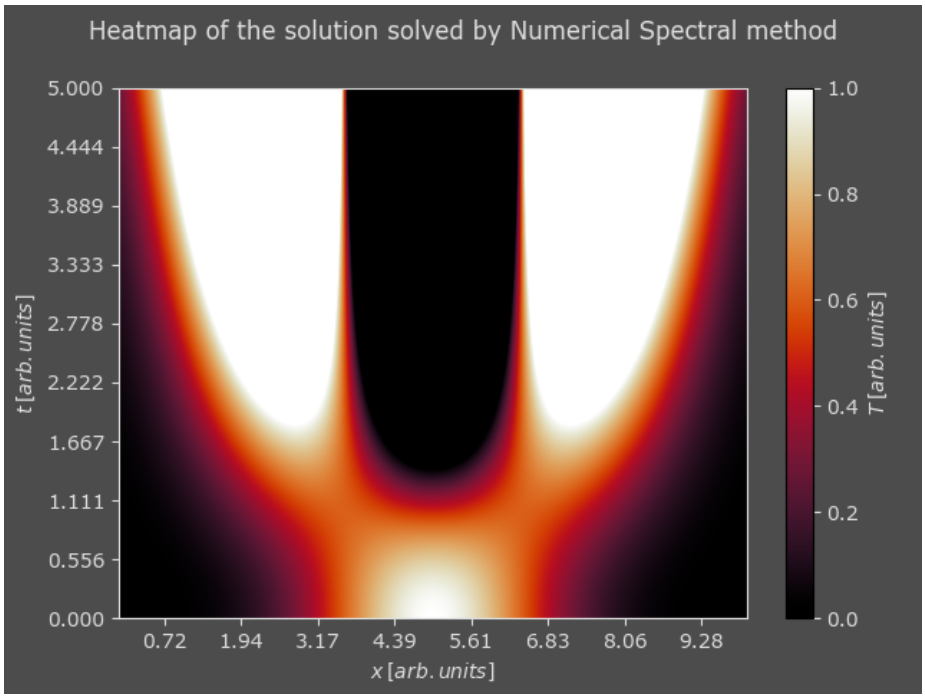


Slika 2: Še heatmap analitične rešitve

Kot pričakovano se temperaturna porazdelitev z časom razširi in zgladi. Zaradi periodičnih robnih pogojev se temperatura ustali pri neki konstantni vrednosti okoli 0.36 arbitrarnih enot. Super zdaj vemo proti čemu tekmujemo z numeričnimi metodami. Poglejmo si rezultate za `solve_Numerically()` na sliki 3, ki bodo seveda skočili na naslednjo stran (ampak je to vseeno boljše kot slike na koncu poročila).



Slika 3: Numerična rešitev



Slika 4: Še heatmap numerične rešitve

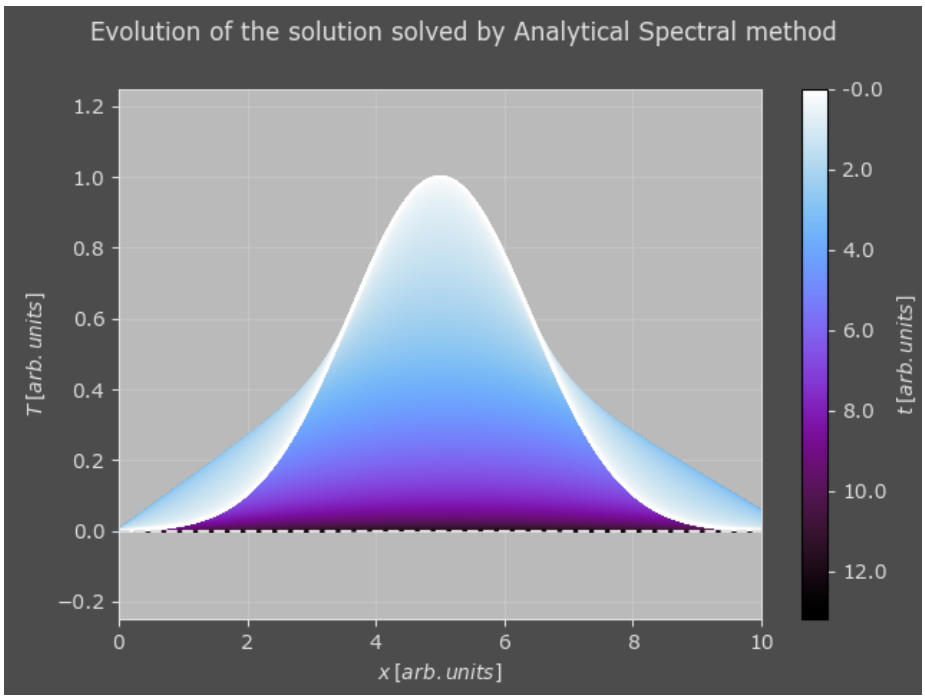
Kot vidimo, se numerična rešitev absolutno NE ujema z analitično in iskreno ne vem zakaj. Poleg vgrajenega integratorja iz `scipy` sem poskusil tudi Runge-Kutta 8 integrator, ki sem ga napisal sam (guess kdo je spet zapravil preveč časa na tem) po NASA dokumentih [1], Eulerjevo metodo, RK45 ipd. ampak tudi to ni pomagalo. I tried I suppose.

4.1.2 Dirichletovi robni pogoji

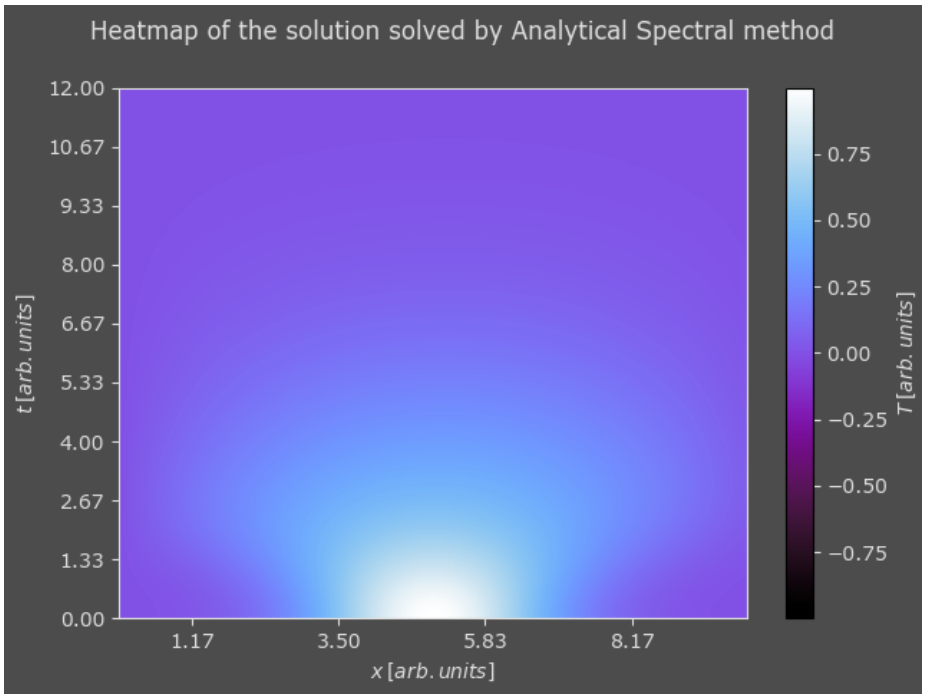
Zdaj pa poglejmo še rezultate za Dirichletove robne pogoje. Začetni pogoj je enak kot prej, ampak to le "v teoriji". V resnici moramo narediti liho razširitev funkcije in intervala na katerem integriramo. Torej je začetni pogoj

$$T(x,0) \propto \exp\left(-\frac{(x-a)^2}{2\sigma^2}\right) - \exp\left(-\frac{(x+a)^2}{2\sigma^2}\right). \tag{17}$$

Rišemo še vedno za isti interval, le da smo zdaj s tem trikom dobili robni pogoj $T(0,t) = T(a,t) = 0$. Dobimo sliko 5.



Slika 5: Analitična rešitev, pri Dirichletovih robnih pogojih

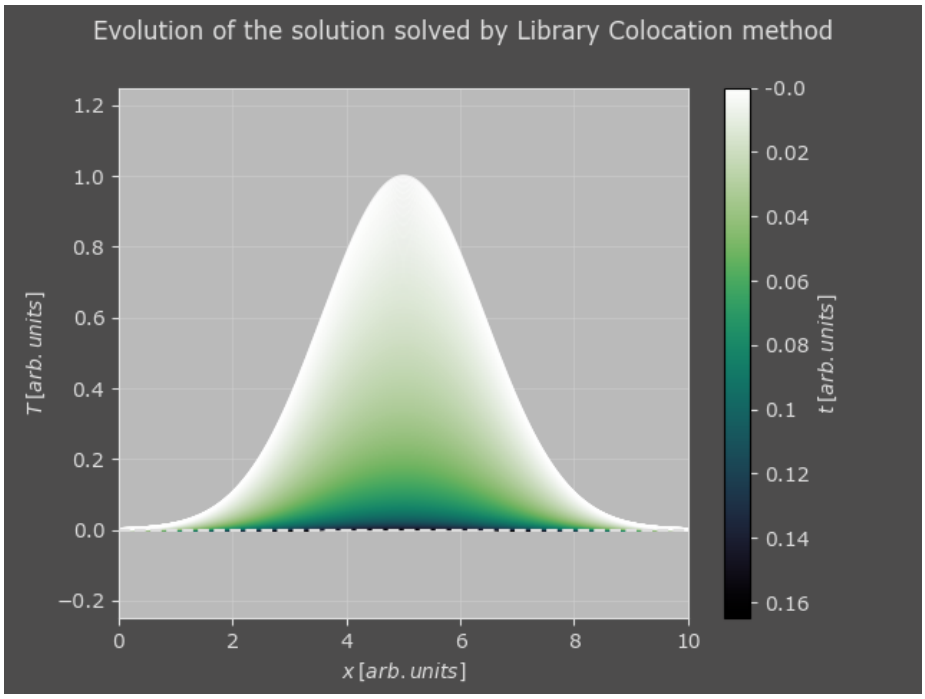


Slika 6: Še heatmap analitične rešitve, pri Dirichletovih robnih pogojih

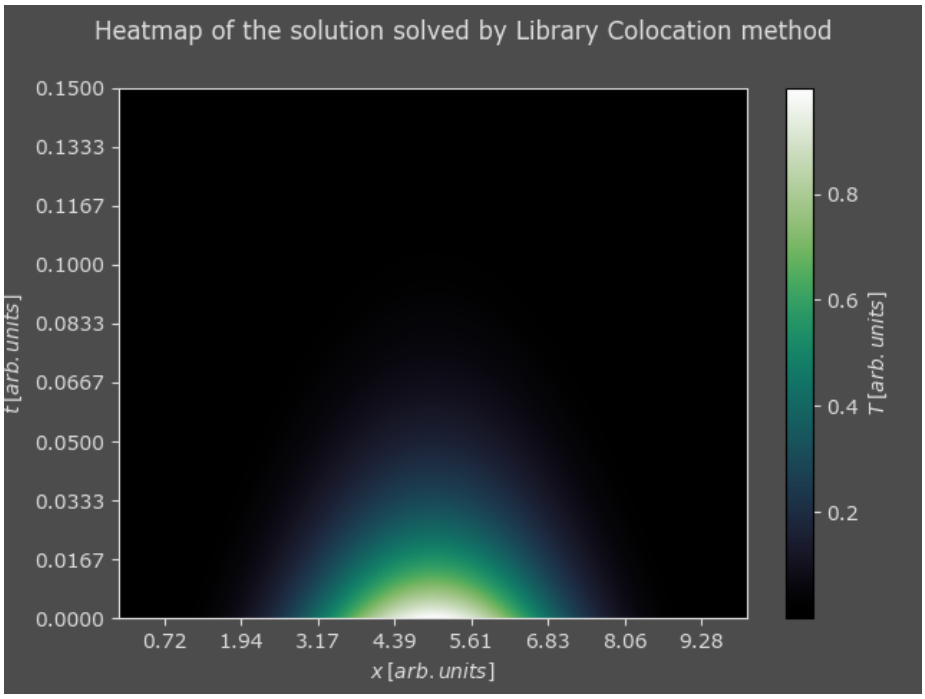
Tukaj se temperatura zmanjšuje in se ustali na 0 (kar je seveda pričakovano). Z numeričnim izračunom se nisem ukvarjal, saj sem bil že od peridičnih robnih pogojev dovolj razočaran.

4.2 Metoda končnih elementov

Zdaj pa poglejmo še rezultate za metodo končnih elementov, oz. kolokacijskim pogojem. Poglejmo si najprej rezultate za metodo kjer sem uporabil vgrajene knjižnice



Slika 7: Knjižnična rešitev, pri periodičnih robnih pogojih

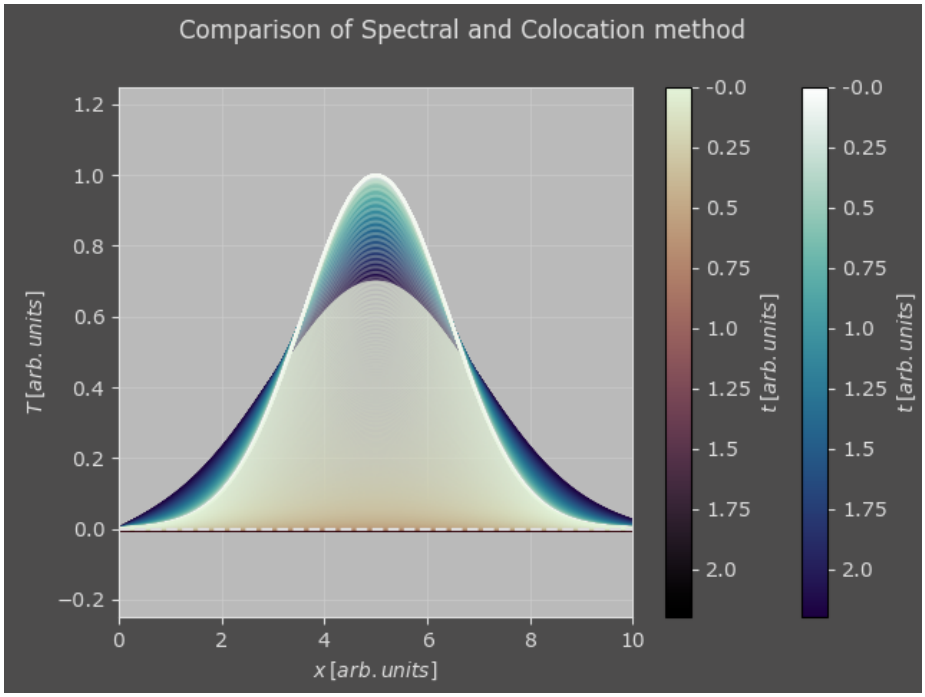


Slika 8: Še heatmap knjižnične rešitve, pri periodičnih robnih pogojih

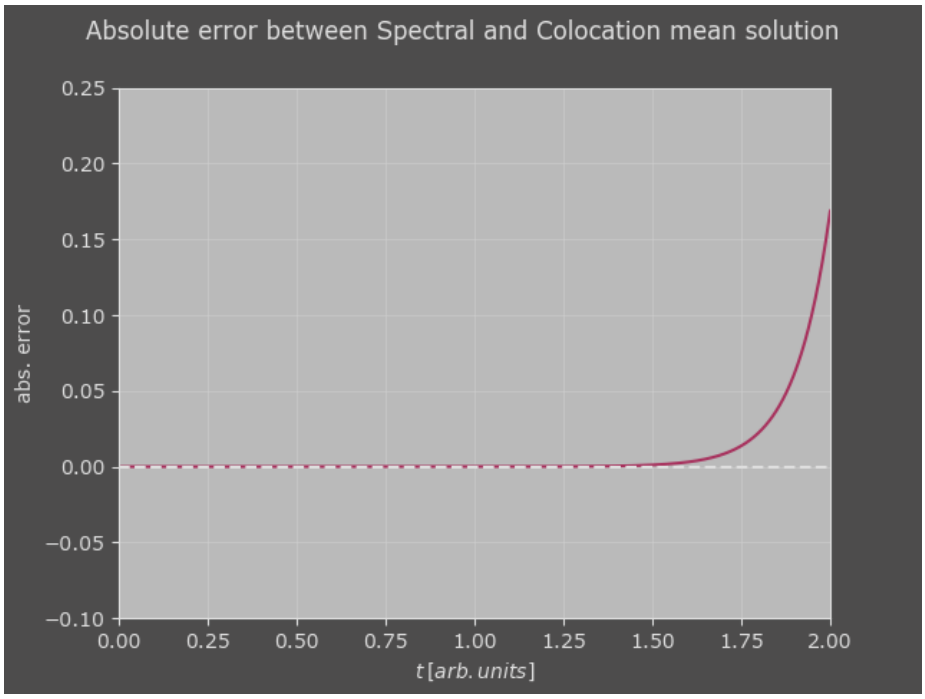
4.3 Primerjava metod

Želim si, da bi lahko bolj natančno primerjal obe metodi, ampak žal mi to ni uspelo. Najbolj me je presenetilo to, da je metoda s kolokacijo prišla do rezultatov, ki so praktično identični analitični rešitvi z spektralno metodo, a s ključno razliko. Pri kolokacijski metodi se rešitev ustali po okoli 0.16 arbitrarnih enot, pri spektralni metodi pa pri okoli 12.0 arbitrarnih enot. To je več redov velikosti razlike in to pri istih parametrih. Želim si, da bi to čudnost lahko raziskal in mogoče celo znal pojasnit, ampak se moram žal spet sklicat na to, da nimam časa.

Vseeno sem poskusil narisati neko primerjavo za neke srednje parametre $N = 1000$, $D = 1e-3$, $a = 5$, $\sigma = 1$ in $t = 2$. Rezultati so na sliki 9.



Slika 9: Primerjava metod (modra, spektralna metoda, oranžna, kolokacijska metoda)

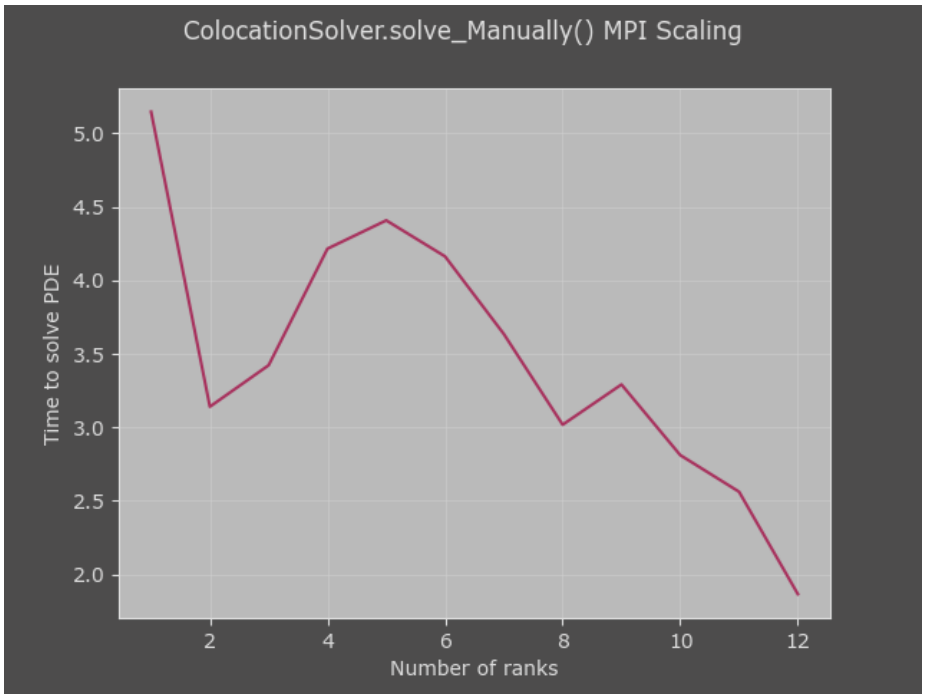


Slika 10: Absolutna napaka mediane rešitve obeh metod

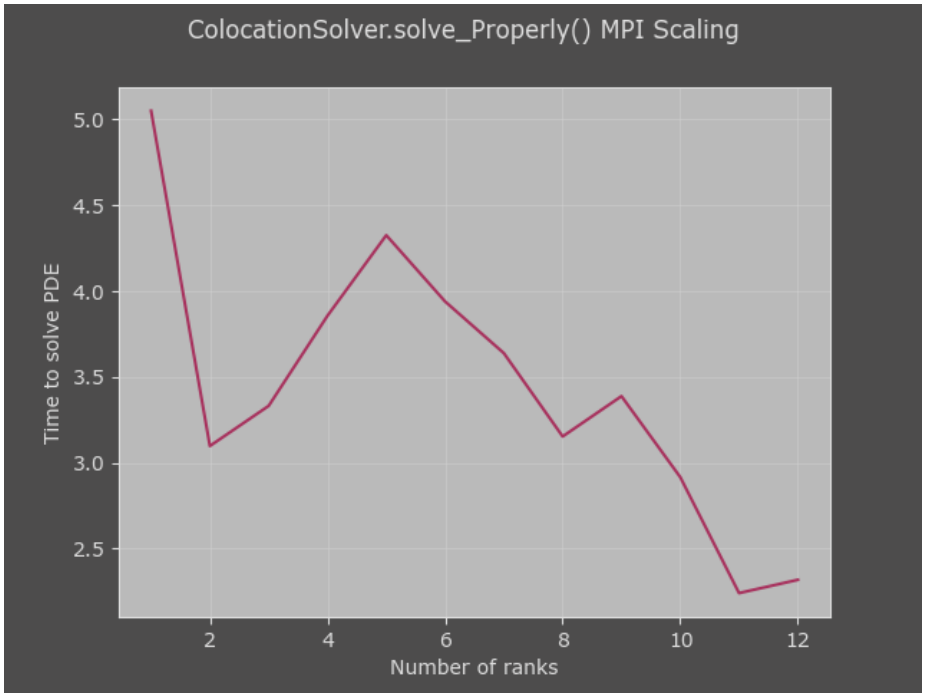
Ne vem če je to že preveč fabricated parameter, ampak če pogledamo absolutno napako mediane rešitve obeh metod, vidimo, da se rešitvi pravzaprav ujemata kar precej dobro in šele proti koncu pride do razcepa.

4.4 Paralelizacija

Za konec pa še paralelizacija. Želel sem dobiti neko značilno premico v log-log skali, ki bi pokazala na uspešno paralelizacijo. Dobil sem slednje slike 11 in 12.



Slika 11: Čas izvajanja v odvisnosti od števila procesov

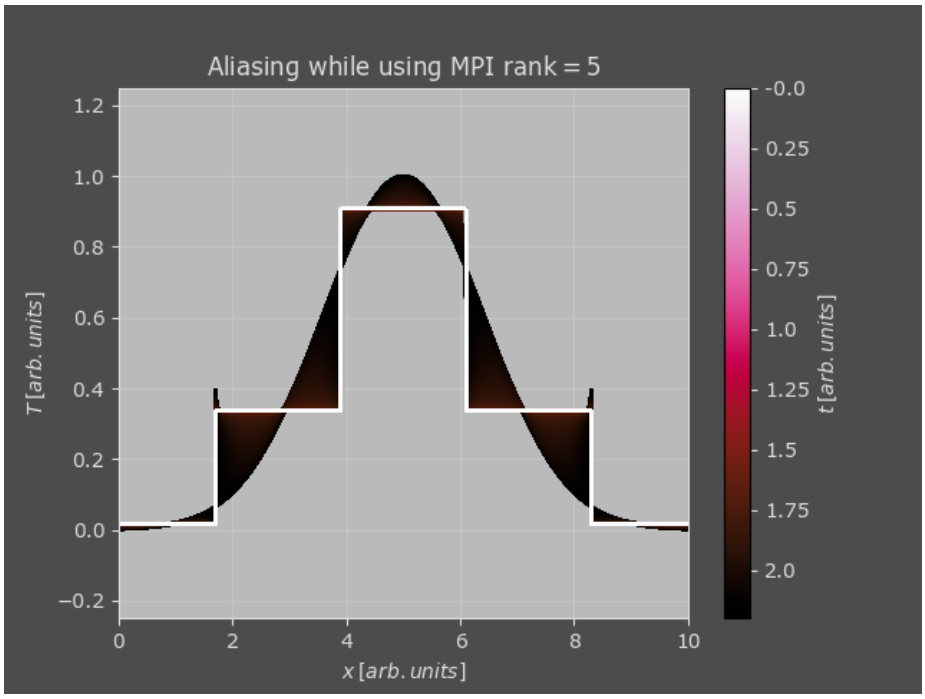


Slika 12: Čas izvajanja v odvisnosti od števila procesov

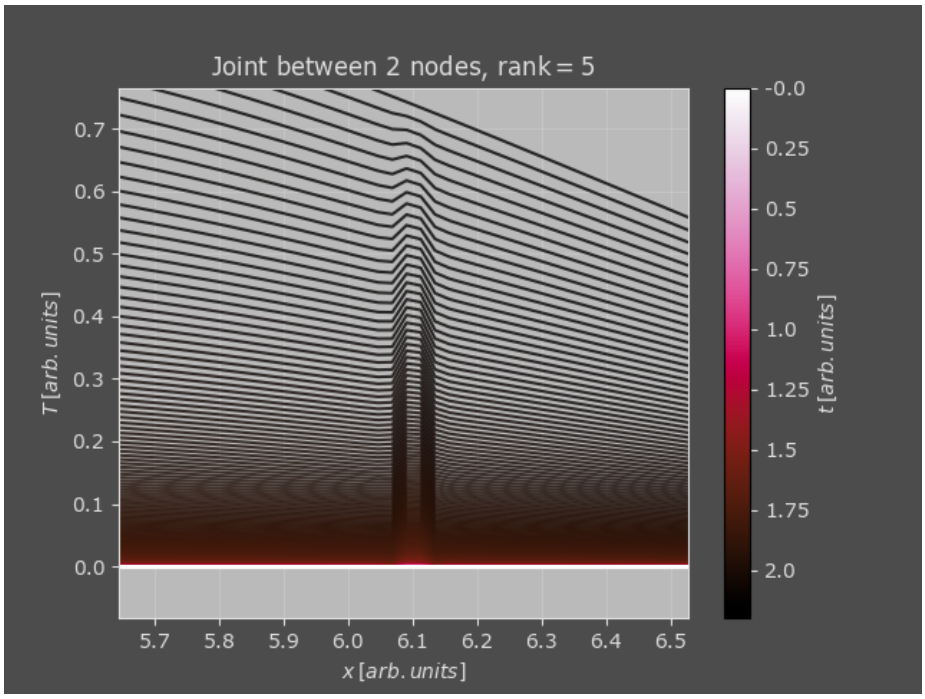
Za vsak primer sem naredil 10 meritev in izračunal mediano. Kot vidimo, je čas izvajanja v obeh primerih podobno odvisen, a žal ne tako, da bi bilo linearno v log skali. Sprva sem mislil, da gre za neke sorte statističen pojav, ampak zdaj sem precej prepričan, da je nekaj še bolj kompliciranо, kot sem pričakoval. Zanimivo je, da je rank = 5 tako časovno potraten. Lahko da je to le značilnost mojega procesorja. Ključno pa je to, da se čas izvajanja z večanjem števila procesov načeloma zmanjšuje. Unfortunatelly, mi je moj domači cluster Astrum delal težave in seveda si nisem smel dovoliti, da bi se še s tem ubadal. Drugače bi imeli na voljo veliko več jeder in bi mogoče lahko dobili lepši graf.

4.4.1 Paralelizacija II (Electric Boogaloo)

"Ampak to si paraleliziral samo kolokacijsko metodo Marko? Kaj pa Fouriereva?"pravite in ja to je res, ampak zdaj je treba razkrit eno skrivnost. Paralelizacija lahko sama po sebi vključi potujevanje rešitve, kar še sploh velja, če ne uporabljaš kakšnih bolj naprednih trikov. V mojem primeru načeloma samo delim intervale podatkov in jih razporejam. Posledica tega je, da se jasno vidi, da je rešitev sestavljena iz nekih "blokov". Sklepam da se ti pojavijo zaradi divergence numeričnih metod na robovih intervala, in je posledično tam vrednost rešitve malo večja oz. drugačna. Torej bi se to tudi hitro lahko rešilo, ampak spet beri: ni časa. Zakaj sem paraleliziral samo kolokacijsko metodo? Ker zaradi meni trenutno neznanega razloga paralelizacija Fouriereve metode vodi v absolutno absurdne stopničaste rešitve. Verjetno bo precej bolj očitno iz slik. Na sliki 13 je prikazan primer, kjer sem paraleliziral Fourierevo metodo. Na sliki 14 pa je prikazana ena rob med blokoma oz. intervaloma.

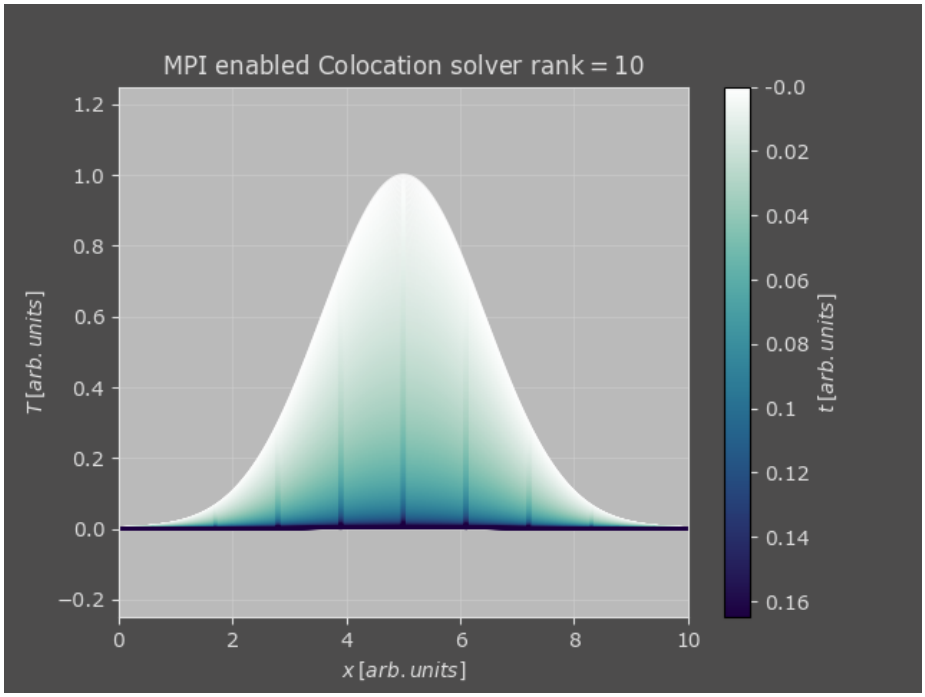


Slika 13: Fourierova metoda, paralelizirana

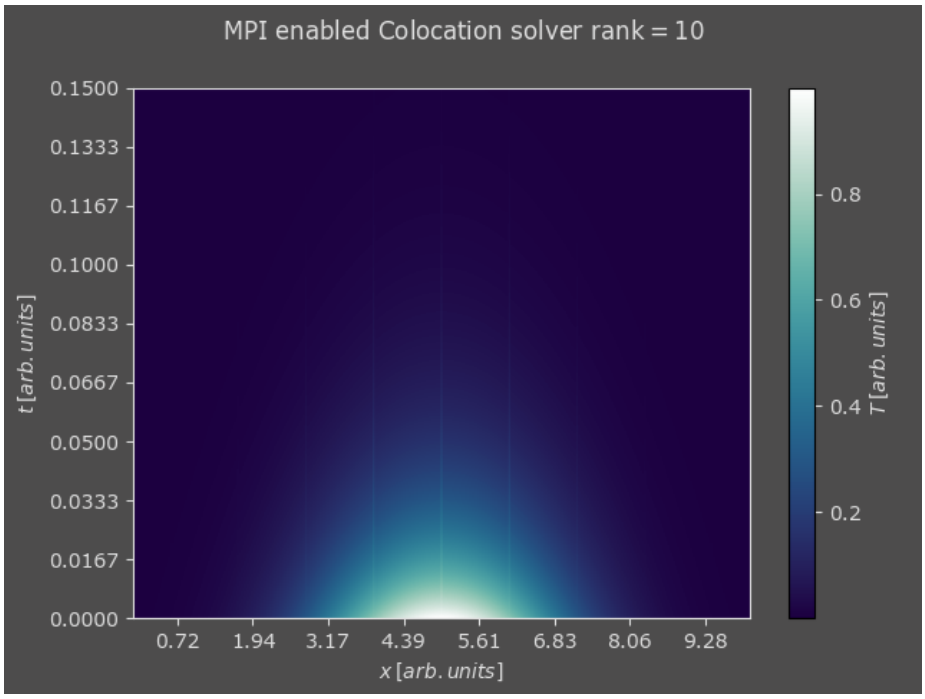


Slika 14: Fourierova metoda, paralelizirana, ena rob med blokoma

Zaradi bizarnega razloga dodatno tudi obrne časovno os pri sliki 13. Ne vem zakaj. Paralelizacija kolokacijske metode pa je v primerjavi s tem precej bolj preprosta. Na sliki 15 je prikazan primer paralelizacije kolokacijske metode. Na sliki 16 pa je prikazan še Heatmap. Iz obeh slik je očitno razvidno, da je rešitev sestavljena iz posameznih blokov oz. v 1D podintervalov.



Slika 15: Kolokacijska metoda, paralelizirana



Slika 16: Še heatmap za kolokacijsko metodo, paralelizirana

5 Komentarji in izboljšave

Res nevem, kako bom opravil še ostale 3 manjkajoče naloge, ampak upam, da mi bo uspelo (brez da bi moral prodati svojo dušo Satanu). Dvomim sicer. Zdi se mi, da sem vseeno ustrezno sproti stvari pokomentiral albeit v bistveno bolj sproščenem tonu, ker sem nekako defeated. Želim si, da bi dejansko lahko naredil vse kar sem si zamislil, ampak žal mi to ni uspelo. Res bi poročilu koristilo še kak malo bolj napreden graf.

Today no funny image. I'm too tired for that.

Literatura

[1] Erwin Fehlberg. *Classical fifth-, sixth-, seventh-, and eighth-order Runge-Kutta formulas with Stepsize Control.* National Aeronautics and Space Administration; for sale by the Clearinghouse for Federal Scientific and Technical Information, Springfield, Va, 1968.