Parcialne diferencialne enačbe: robni problemi in relaksacija

5. domača naloga pri predmetu Modelska analiza II

Domen Vaupotič

1 Parcialne diferencialne enačbe

Ukvarjamo se s parcialnimi diferencialnimi enačbami (PDE), ki so zelo pomembne pri opisovanju fizikalnih pojavov. Sistem linearnih parcialnih diferencialnih enačb prvega reda lahko zapišemo v matrični obliki

$$A\boldsymbol{\nu}_x + B\boldsymbol{\nu}_y = \mathbf{c},$$

pri čemer sta A in B realni matriki velikosti $M \times M$, vektor $\boldsymbol{\nu} = (\nu_1, \nu_2, \ldots, \nu_M)^T$ predstavlja rešitev sistema, vektor $\mathbf{c} = (c_1, c_2, \ldots, c_M)^T$ pa nehomogene člene. Glede na karakteristični polinom $\rho(\lambda) = \det(A - \lambda B)$ delimo sisteme na *hiperbolične* ($\rho(\lambda)$ ima natanko M nedegeneriranih realnih ničel in $(A - \lambda B)\mathbf{u} = 0$ ima natanko M neodvisnih rešitev), *parabolične* ($\rho(\lambda)$ ima M realnih ničel, $(A - \lambda B)\mathbf{u} = 0$ nima M neodvisnih rešitev) ter *eliptične* ($\rho(\lambda)$ nima realnih ničel).

V fiziki pa pogosto nastopajo parcialne diferencialne enačbe drugega reda, ki jih v splošni obliki zapišemo kot v skupine:

$$a\nu_{xx} + b\nu_{xy} + c\nu_{yy} + d\nu_x + e\nu_y + f\nu = Q,$$

pri čemer iščemo skalarno rešitev $\nu(x, y)$ na neki domeni znotraj ravnine (x, y), ki ima podane začetne in robne pogoje. Klasificiramo jih glede na predznak diskriminante $\mathcal{D} = b^2 - 4ac$:

- ⋆ *hiperbolične* ($\mathcal{D} > 0$): časovnoodvisni konservativni procesi, ki se razvijajo proti stacionarnemu stanju

$$\nu_t \pm c\nu_x = 0 \qquad \text{advekcijska enačba}$$

$$\nu_{tt} = c^2\nu_{xx} \qquad \text{valovna enačba}$$

- ⋆ *parabolične* ($\mathcal{D} = 0$): časovnoodvisni disipativni procesi, ki se razvijajo proti stacionarnemu stanju

$$\nu_t = D\nabla^2\nu \qquad \text{difuzijska enačba}$$

- ⋆ *eliptične* ($\mathcal{D} < 0$): sistemi v ravnovesnih/stacionarnih stanjih, katerih rešitve običajno minimizirajo ali maksimizirajo ohranitvene integrale sistema (npr. energijo)

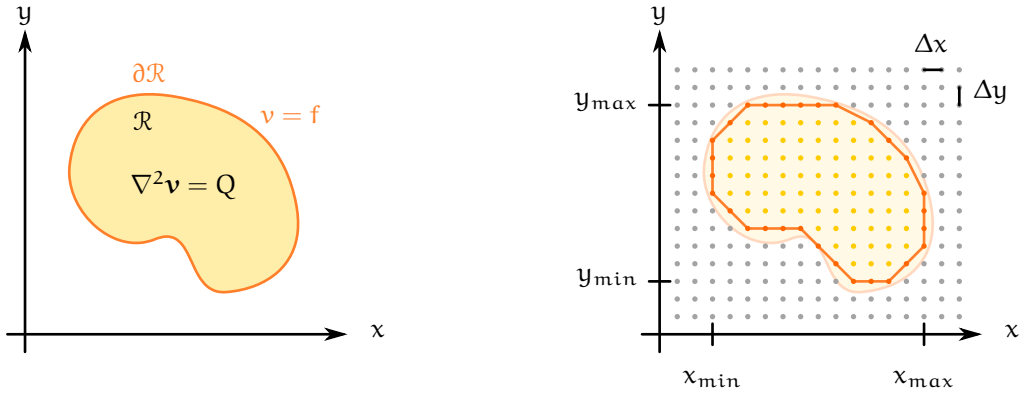
$$\nabla^2\nu = \rho \qquad \text{Poissonova enačba}$$

1.1 Diskretizacija

Parcialne diferencialne enačbe lahko rešimo z diferenčnimi metodami, pri katerih moramo domeno diskretizirati. Običajno jo diskretiziramo z ekvidistančno (pravokotno) mrežo, kot je prikazano na sliki 1. Zvezne spremenljivke tako postanejo diskretne:

$$\begin{aligned} x, y &\longrightarrow x_j, y_j \qquad j = 0, \ldots, N_x; k = 0, \ldots, N_y \\ \nu(x, y) &\longrightarrow \nu(x_j, y_k) = \nu_{jk} \\ \Delta x &= (x_{max} - x_{min})/N_x \\ \Delta y &= (y_{max} - y_{min})/N_y \end{aligned}$$

Na diskretizirani mreži lahko sedaj zapišemo diferenčno shemo. Pri tem uporabljamo različne izraze za končne diference:



Slika 1: Domena diferencialne enačbe, $\mathcal{R}$, in njen rob, $\partial\mathcal{R}$, (levo) ter diskretizacija območja (desno).

$$\frac{\partial u}{\partial x} \approx \frac{1}{\Delta x}\left(u_{j+1,k} - u_{j,k}\right) \qquad \text{diferenca naprej}$$

$$\frac{\partial u}{\partial x} \approx \frac{1}{\Delta x}\left(u_{j,k} - u_{j-1,k}\right) \qquad \text{diferenca nazaj}$$

$$\frac{\partial u}{\partial x} \approx \frac{1}{2\Delta x}\left(u_{j+1,k} - u_{j-1,k}\right) \qquad \text{centralna diferenca}$$

$$\frac{\partial u}{\partial x} \approx \frac{1}{2\Delta x}\left(-3u_{j,k} + 4u_{j+1,k} - u_{j+2,k}\right) \qquad \text{diferenca naprej drugega reda}$$

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{1}{(\Delta x)^2}\left(u_{j+1,k} - 2u_{j,k} + u_{j-1,k}\right) \qquad \text{centralna diferenca za drugi odvod}$$

Za vsako točko v notranjosti diferencialno enačbo zapišemo z diferenčno shemo, za robne točke pa zapišemo enačbe robnih pogojev:

- *Dirichletov robni pogoj* zapišemo z enostavno enačbo, ki podaja vrednost rešitve na robu:

$$u_{j,k} \stackrel{!}{=} f_{j,k}.$$

- *Neumannov robni pogoj* je nekoliko zahtevnejši za implementacijo. Če želimo uporabiti centralno diferenco, moramo uvesti dodatne navidezne točke zunaj območja $\mathcal{R}$. Običajno se zato zadovoljimo z diferenco naprej (v območje):

$$u_{j+1,k} - u_{j,k} \stackrel{!}{=} f'_{j,k}\Delta x.$$

  Uporabimo lahko tudi diferenco višjega reda, pri kateri se upošteva tudi drugega najbližjega soseda:

$$-3u_{j,k} + 4u_{j+1,k} - u_{j+2,k} \stackrel{!}{=} 2f'_{j,k}\Delta x,$$

  vendar ta ni primerna za iterativne metode, ki računajo samo z najbližjimi sosedi.

1.2 Pretok tekočine – Pouisseuillov zakon

V prvem delu se bomo ukvarjali s pretokom tekočine skozi cev z zapleteno geometrijo preseka. Iskali bomo torej stacionarno rešitev enačbe za pretok, ki se v brezdimenzijskih količinah skupaj z Dirichletovim robnih pogojem glasi:

$$\nabla^2 u = 1, \qquad u\Big|_{\partial\mathcal{R}} = 0.$$

Zapišimo diferenčno shemo za kvadratno mrežo ($\Delta x = \Delta y = h$):

$$\frac{1}{h^2}\left(u_{j+1,k} + u_{j-1,k} + u_{j,k+1} + u_{j,k-1} - 4u_{j,k}\right) = 1.$$

1.2.1 Pouiseillov koeficient

Če izračunano hitrostno polje integriramo po celotni površini, dobimo celotni volumski pretok:

$$\iint u(x,y)\,dx\,dy = \Phi_V,$$

tega pa povežemo s Hagen-Pouisellovim zakonom, iz katerega izrazimo Poisseuilleov koeficient C:

$$\Phi_V = C\frac{S^2 p'}{8\pi\eta} \quad \overset{p'/\eta=1}{\Longrightarrow} \quad C = \frac{8\pi\Phi_V}{S^2}.$$

1.3 Prevajanje toplote – Fourierov zakon

V drugem delu bomo obravnavali prevajanje toplote skozi enakostranični ($2r = h$) kovinski valj. Zapišimo najprej zakon prevajanja:

$$T_t = D\nabla^2 T + \frac{q}{\rho c_p},$$

ki se v stacionarnih razmerah ($T_t = 0$) in brez notranjih izvorov ($q = 0$) poenostavi na enačbo:

$$\nabla^2 T = 0.$$

Enačbo bomo reševali v cilindričnih koordinatah ($T = T(r,z)$) in iskalni osnosimetrične rešitve (brez odvisnosti od polarnega kota). Parcialna diferencialna enačba je torej:

$$\nabla^2 T = \frac{\partial^2 T}{\partial r^2} + \frac{1}{r}\frac{\partial T}{\partial r} + \frac{\partial^2 T}{\partial z^2} = 0,$$

robni pogoji pa bodo mešani:

$$T\Big|_{\partial\mathcal{R}_1} = T,$$
$$\frac{\partial T}{\partial r}\Big|_{\partial\mathcal{R}_2} = -j/\lambda = -J.$$

S premeteno izbiro mreže ($N_r = N_x, \Delta z = 2\Delta r$) se diferenčna shema poenostavi v:

$$\left(1+\frac{1}{2j}\right)T_{j+1,k} + \left(1-\frac{1}{2j}\right)T_{j-1,k} + \frac{1}{4}T_{j,k+1} + \frac{1}{4}T_{j,k-1} - \frac{5}{2}T_{j,k} = 0.$$

2 Metode reševanja

Ko zapišemo vse diferenčne sheme in robne pogoje, pridelamo približno $N_x \times N_y$ enačb za približno $N_x \times N_y$ neznank (dejansko število je manjše, če je domena manjša od pravokotnika). Zaradi lažje implementacije vse točke zložimo po vrsti v en vektor:

$$\mathbf{u} = (u_{0,0}, u_{1,0}, \ldots, u_{N_x,0}, u_{0,1}, u_{1,1}, \ldots, u_{N_x,1}, \ldots, u_{N_x,N_y}),$$

sistem enačb pa zapišemo v matrični obliki:

$$A\mathbf{u} = \mathbf{q}.$$

Rešitev naše parcialne diferencialne enačbe smo torej prevedli na reševanje sistema linearnih enačb. Ker pa je matrika A v splošnem zelo velika in zelo redka, se je smiselno poslužiti premetenejših metod za reševanje.

2.1 Direktna metoda

Matrično enačbo lahko rešimo direktno z metodami za reševanje sistema enačb, kot je `numpy.linalg.solve`, ki uporabi razcep LU in sistem reši z zaporednimi direktnimi in obratnimi vstavljanji. Metoda je za velike mreže počasna.

2.2 Iterativne metode

Matrično enačbo rešujemo z iterativnimi približki $\mathbf{w}$, ki se bližajo pravi rešitvi $\mathbf{u}$. Pri tem v vsakem koraku zmanjšujemo algebrajsko napako $\mathbf{e} = \mathbf{u} - \mathbf{w}$ in residualno napako $\mathbf{r} = \mathbf{q} - A\mathbf{w}$. Iterativni korak zapišemo kot:

$$\mathbf{w}^{n+1} = \mathbf{w}^n + B\mathbf{r}^n,$$

pri čemer je B aproksimacija matrike A. Matriko A dodatno razdelimo na spodnje trikotno, diagonalno in zgornje trikotno $A = L + D + U$. Glede na uporabljeni približek $B \approx A$ se iterativne metode delijo na slednje:

metoda	matrika B
Jacobi	D^{-1}
Gauss-Seidel	$(L + D)^{-1}$
SOR	$\omega(D + \omega L)^{-1}$

Preden zapišemo implementacijo iterativnih metod, zapišimo diferenčno shemo še v splošni obliki:

$$\alpha^1_{j,k} u_{j+1,k} + \alpha^2_{j,k} u_{j-1,k} + \alpha^3_{j,k} u_{j,k+1} + \alpha^4_{j,k} u_{j,k-1} - \alpha^0_{j,k} u_{j,k} = q_{j,k}.$$

Za Laplaceov operator na ekvidistančni kvadratni mreži s homogenimi robnimi pogoji so koeficienti:

$$\alpha^0_{j,k} = -\frac{4}{h}, \quad \alpha^1_{j,k} = \alpha^2_{j,k} = \alpha^3_{j,k} = \alpha^4_{j,k} = -\frac{1}{h}, \quad q_{j,k} = 0.$$

Iterativne metode delujejo v zanki po vseh aktivnih točkah mreže:

$$u^{n+1}_{j,k} = -\frac{1}{\alpha^0_{j,k}} \left[q_{j,k} - \alpha^1_{j,k} u^n_{j+1,k} - \alpha^2_{j,k} u^n_{j-1,k} - \alpha^3_{j,k} u^n_{j,k+1} - \alpha^4_{j,k} u^n_{j,k-1} \right] \quad \text{Jacobi},$$

$$u^{n+1}_{j,k} = -\frac{1}{\alpha^0_{j,k}} \left[q_{j,k} - \alpha^1_{j,k} u^n_{j+1,k} - \underline{\alpha^2_{j,k} u^{n+1}_{j-1,k}} - \alpha^3_{j,k} u^n_{j,k+1} - \underline{\alpha^4_{j,k} u^{n+1}_{j,k-1}} \right] \quad \text{Gauss-Seidel}.$$

Gauss-Seidelova metoda se od Jacobijeve razlikuje zgolj v tem, da uporablja dve vrednosti sosedov iz trenutne iteracije.

2.2.1 Metoda pospešene relaksacije (SOR)

Metoda SOR je nadgradnja navadnih iteracijskih metod, pri ktaeri v vsakem iteracijskem koraku sprva izvedemo en Gauss-Seidelov korak, s katerim izračunamo vmesno rešitev $u^\star_{jk}$, nato pa izračunamo uteženo povprečje med staro in novo vrednostjo:

$$u^{n+1}_{jk} = u^n_{jk} + \omega(u^\star_{jk} - u^n_{jk}),$$

pri čemer $0 < \omega < 2$ predstavlja prosti parameter, ki nadzoruje hitrost pospeševanja. S poznavanjem spektralnega radija $\rho_J = \rho(R_J)$ matrike $R_J = I - BA$ lahko določimo optimalni faktor ω, s katerim bo konvergenca metode najhitrejša:

$$\omega_{opt} = \frac{2}{1 + \sqrt{1 - \rho_J^2}}.$$

Ker je računanje spektralnega radija v splošnem zahtevno, lahko uporabimo nastavek:

$$\omega_{opt} = \frac{2}{1 + \alpha \frac{\pi}{N}},$$

pri čemer je α neznani parameter, N pa število točk po enem robu. Ugodno je torej, da na majhni mreži poiščemo najoptimalnejši α, nato pa izračunani ω_{opt} uporabimo pri reševanju na večji mreži.

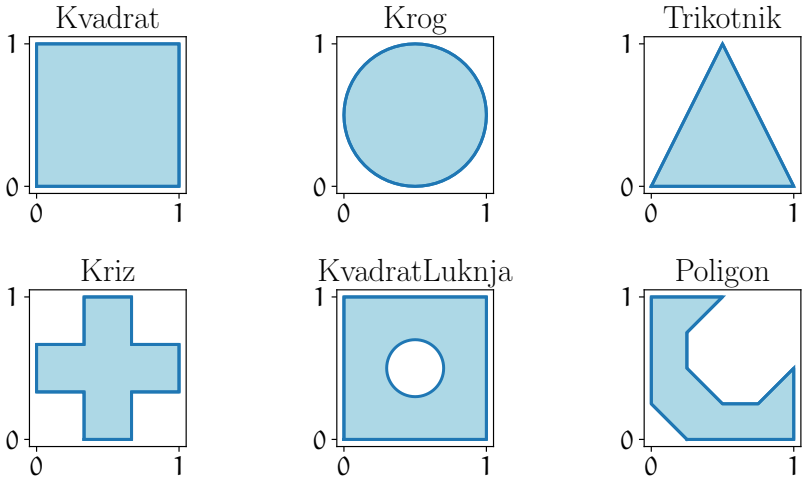
2.2.2 Pospeševanje Čebiševa

Metodo SOR lahko izboljšamo tako, da faktor pospeševanja postopoma povečujemo proti optimalnemu. Eno iteracijo razdelimo na dve politeraciji, tako da v vsaki politeraciji obravnavamo samo ene izmed točk šahovnice (izmenično črne/bele). V vsaki politeraciji uporabimo faktor pospeševanja $\omega^{(n)}$:

$$\omega^{(0)} = 1, \quad \omega^{(n+1/2)} = \frac{1}{1 - \frac{1}{4}\rho_J^2 \omega^{(n)}} \qquad \text{in velja} \qquad \lim_{n\to\infty} \omega^{(n)} = \omega_{opt}.$$

3 Implementacija

Reševanja problema sem se lotil v splošni geometriji, tako da sem uporabil knjižnico `shapely`. Ta omogoča, da definiramo večkotnik (`Polygon`) in točko (`Point`), nato pa ponuja metodo za preverjanje, ali točka leži znotraj večkotnika. V prvem delu sem obravnaval 6 različnih prerezov cevi, ki so prikazani na sliki 2.

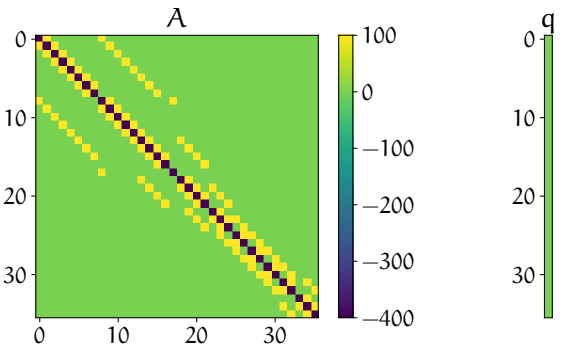


Slika 2: Različni prerezi cevi.

4 Rezultati

4.1 Pretok tekočine

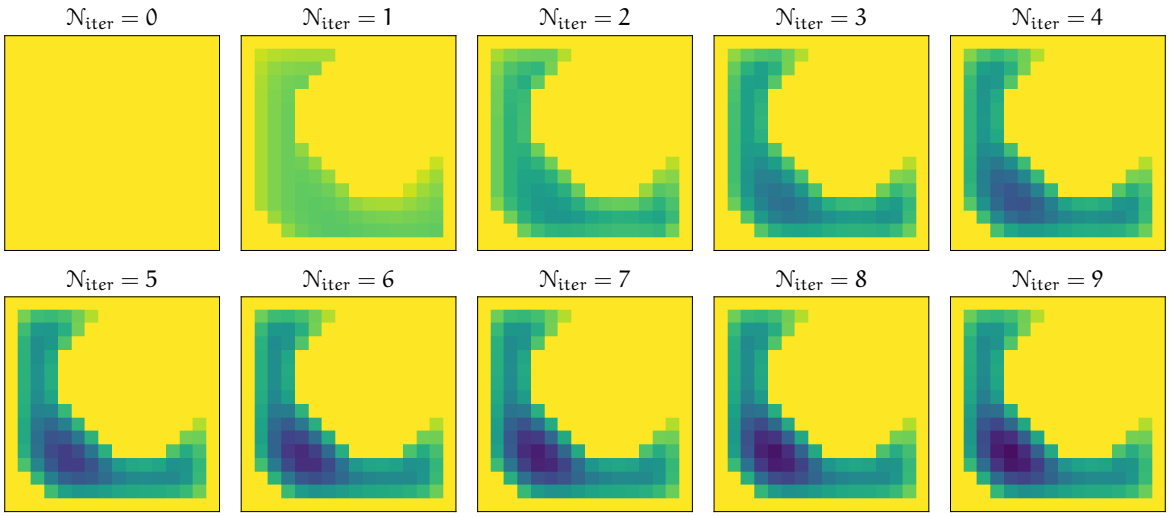
Za začetek si oglejmo, kakšen matrični sistem rešujemo. Na sliki 3 sta prikazana matrika A in vektor q na manjši mreži ($N_x = N_y = 10$). Matrika A je zares zelo redka, neničelne vrednosti pa so zbrane okoli



Slika 3: Matrika A in vektor q za poligon.

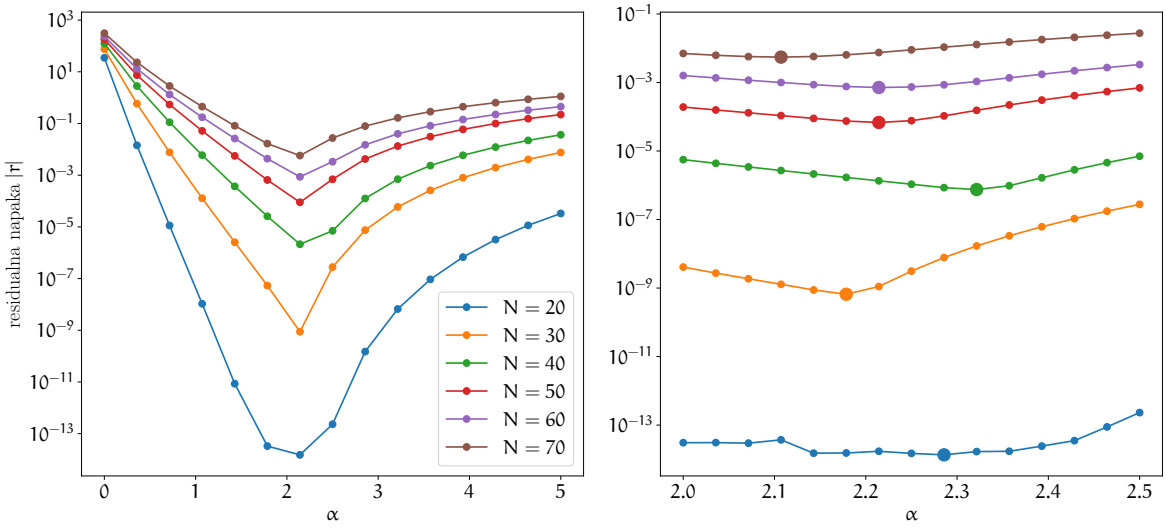
glavne diagonale. Vektor q je v tem primeru kar konstanten, saj imamo homogene Dirichletove robne pogoje.

Na sliki 4 je prikazano, kako se pri iterativnih metodah rešitev z vsako iteracijo izboljšuje.



Slika 4: Potek rešitve med iteracijami.

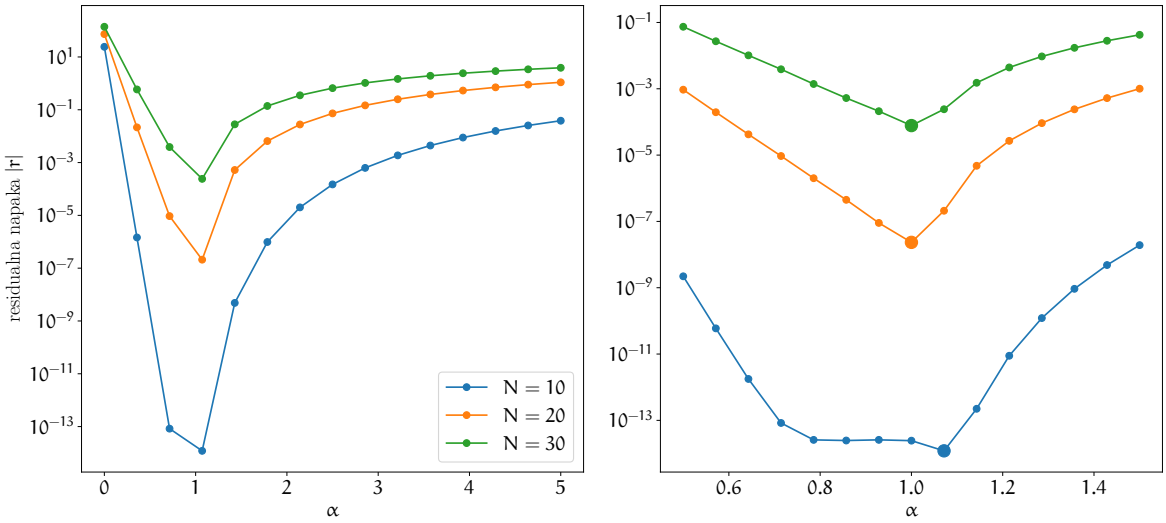
Izračunajmo sedaj optimalni α in s tem optimalni ω za metodo SOR. Na sliki 5 je prikazana residualna napaka pri različnih α in različnih velikostih mreže N. Opazimo, da smo po 75 iteracijah dobili najbolj točno rešitev pri $\alpha_{\text{opt}} \approx 2{,}2$. Podobno je na sliki 6 prikazana residualna napaka za krog. Opazimo, da je ta napaka v splošnem večja, kar je posledica oblike kroga. Motnja iz enega roba kroga namreč potrebuje veliko več časa (iteracij), da prepotuje do drugega roba, medtem ko je pri poligonu zaradi njegove oblike ta čas krajši.



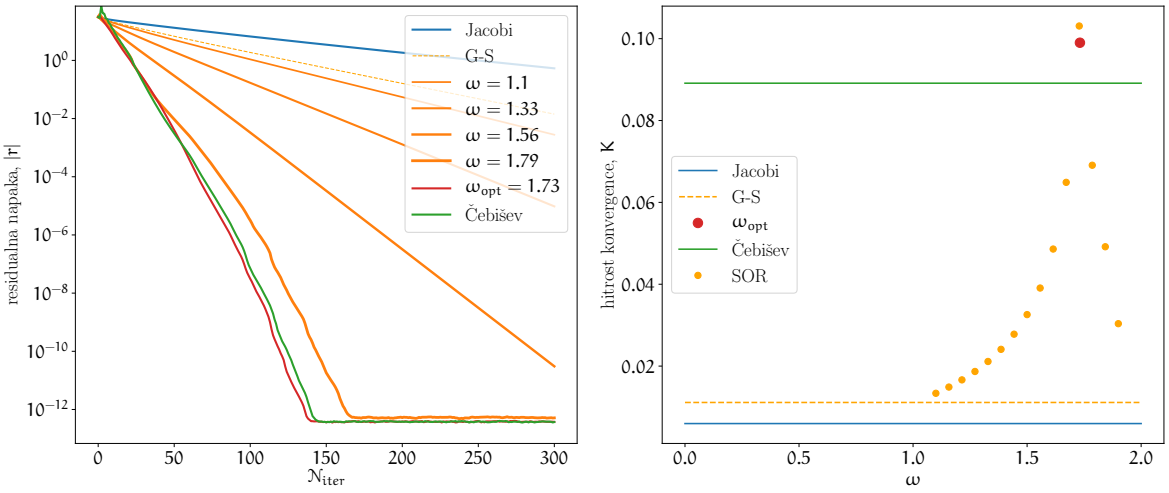
Slika 5: Residualna napaka za poligon pri različnih vrednostih α in različnih velikostih mreže N po 75 iteracijah (levo) in povečan prikaz na območju med $\alpha = 2$ in $\alpha = 2{,}5$ (desno).

Primerjajmo sedaj hitrost različnih metod med seboj. Na sliki 7 je prikazan potek residualne napake med iteracijami za različne metode, izračunana pa je tudi hitrost konvergence K s prileganjem premice na residualne napake med 50. in 100. iteracijo:

$$\log_{10} |r| = K\mathcal{N}_{\text{iter}} + a.$$



Slika 6: Residualna napaka za krog pri različnih vrednostih α in različnih velikostih mreže N po 75 iteracijah (levo) in povečan prikaz na območju med $\alpha = .5$ in $\alpha = 1{,}5$ (desno).



Slika 7: Residualna napaka med iteracijami pri različnih metodah (levo) in izračunana hitrost konvergence (desno).

Opazimo, da vse metode konvergirajo enakomerno eksponentno, pri čemer je Jacobijeva najbolj počasna, nekoliko boljša je Gauss-Seidelova. SOR je znatno hitrejši, pri čemer pa moramo izbrati ustrezni ω. Vidimo, da je bil naš približek za optimalni ω z izračunanim α_{opt} zelo dober. Metoda Čebiševa se ni odrezala bistveno bolje, kar je verjetno posledica napačne implementacije, saj reševanje nisem razdelil na politeracije, temveč le uporabil spreminjajoči se parameter ω.

Na spodnjih slikah so prikazana hitrostna polja tekočine v šestih različnih prerezih. Rezultati so izračunani z metodo SOR na kvadratni mreži 75×75 in 400 iteracijami. Vse residualne napake so bile manjše kot 1×10^{-9}. Izračunani Pouisseuilovi koeficienti so prikazani v tabeli 1. Točnost izračunanih Pouisseuilovih koeficientov najlažje uvidimo, če si ogledamo koeficient za krožni presek – njegova točna vrednost namreč znaša 1. Napaka izračunenega koeficienta je tako 3 %.

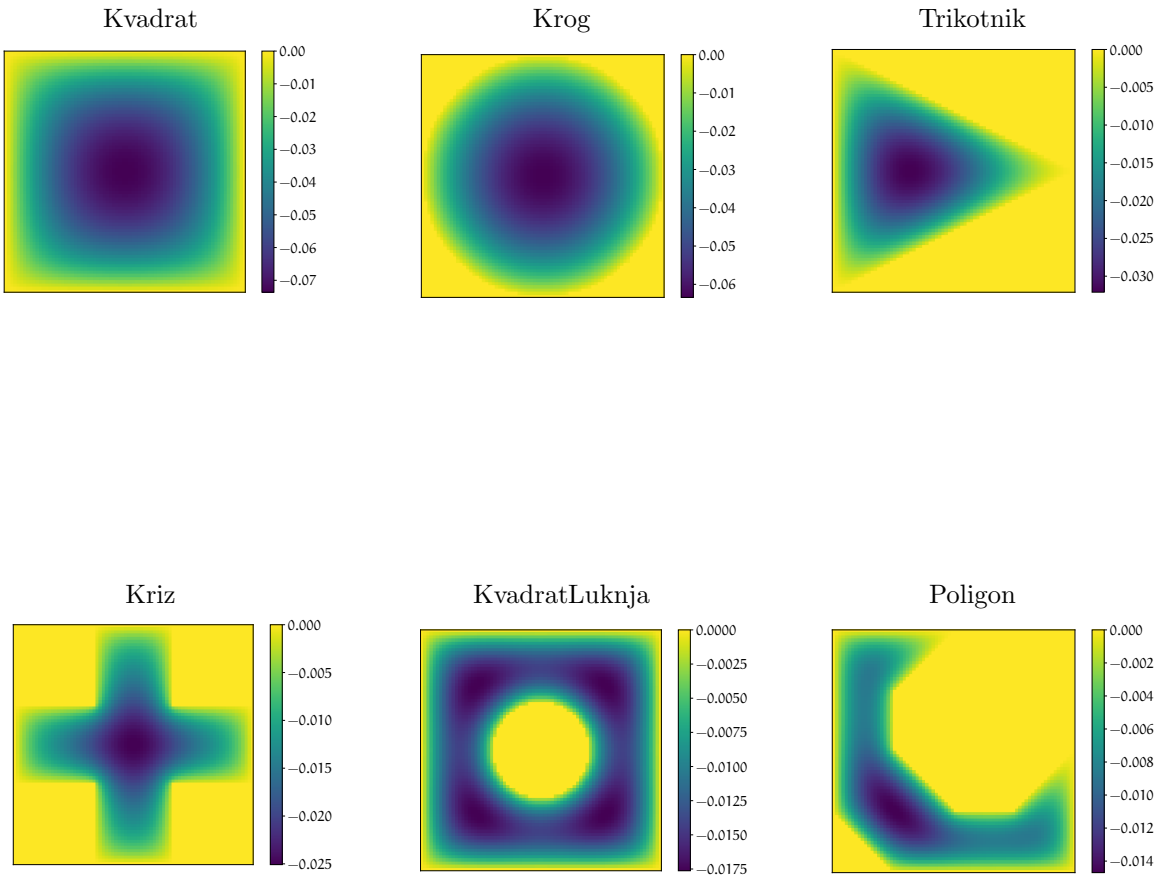
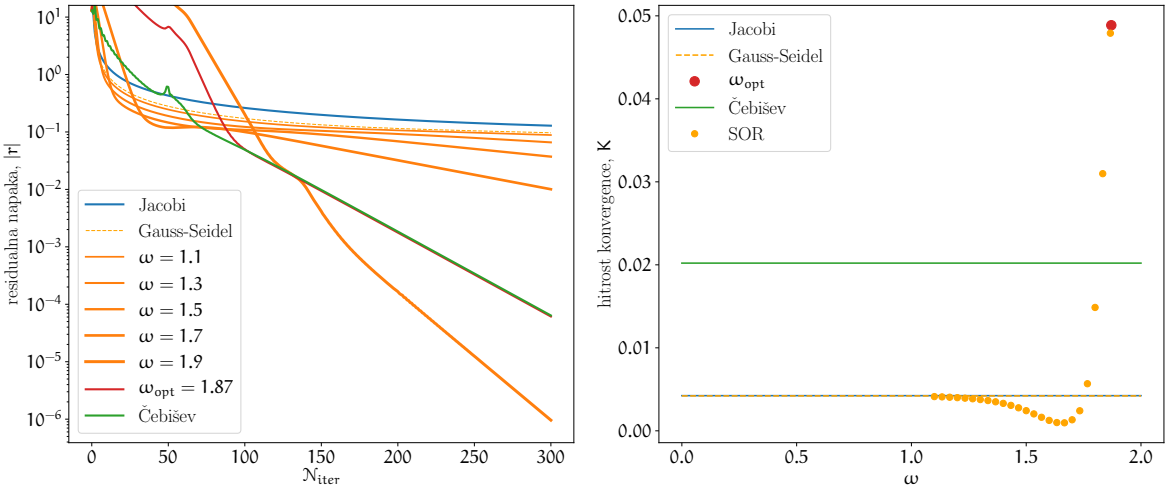


Tabela 1: Tabela izračunanih Pouisseuilovih koeficientov.

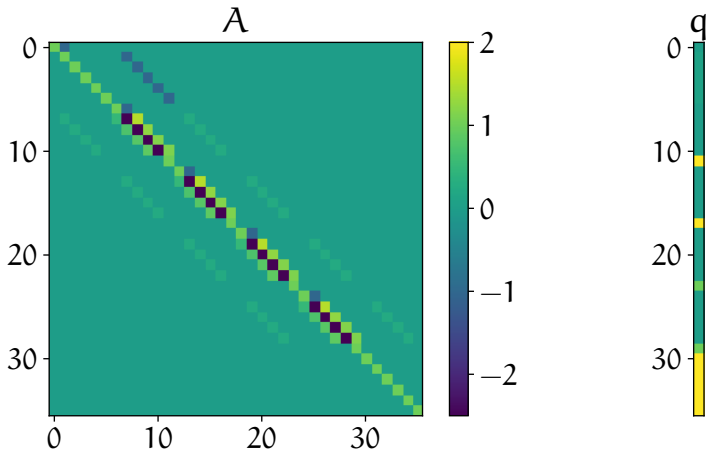
Oblika	S	Φ	C
Kvadrat	1,000 000	−0,035 124	0,882 759
Krog	0,784 137	−0,025 278	1,033 246
Trikotnik	0,500 000	−0,007 271	0,730 964
Kriz	0,555 556	−0,005 733	0,466 870
KvadratLuknja	0,874 538	−0,009 053	0,297 476
Poligon	0,500 000	−0,003 190	0,320 690

5 Prevajanje toplote

Posvetimo se sedaj prevajanju toplote v valju. Plašč valja razdelimo na zgornjo in spodnjo polovico. Spodnjo polovico držimo pri konstantni temperaturi T_1, zgornjo pa pri T_2. Na spodnji osnovni ploskvi segrevamo s toplotnim tokom J, zgornjo osnovno ploskev pa držimo pri temperaturi T_1. Za začetek si oglejmo hitrost konvergence, ki je prikazana na sliki 8. Konvergenca je znatno počasnejša, zato moramo uporabiti več iteracij. Na sliki 9 je prikazan primer matrike A in vektorja q; ta tokrat ni konstanten, saj robni pogoji niso več homogeni.



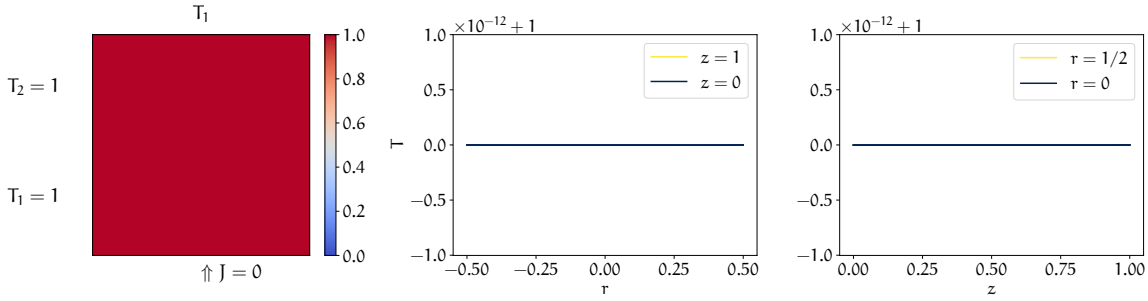
Slika 8: Hitrost konvergence za problem prevajanja toplote.



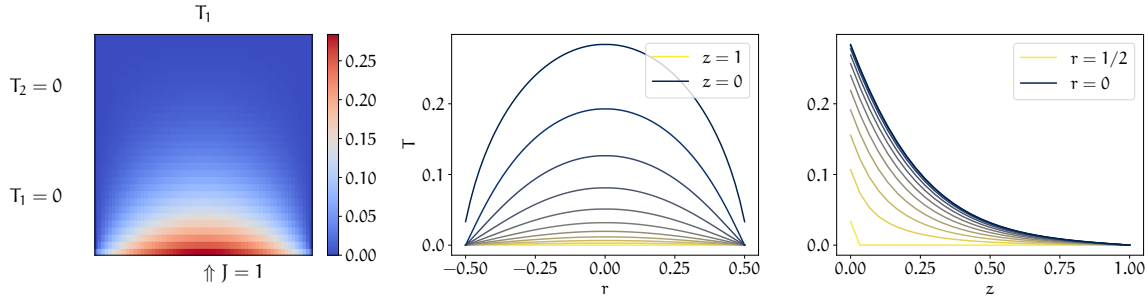
Slika 9: Matrika A in vektor q za prevajanje toplote v valju.

Na spodnjih slikah so prikazani temperaturni profili pri različnih robnih pogojih. Rezultati so izračunani na mreži velikosti 30×30 z metodo SOR. Residualne napake so bile manjše kot 1×10^{-13}. Na profilu 2 se lahko prepričamo, da metoda dobro deluje, saj je odvod profila na spodnji ploskvi ($\frac{\partial T}{\partial z}\big|_{z=0}$) res enak po celotnem r in enak toplotnemu toku J. Na profilu 2 vidimo tudi paraboličen profil, ki je značilen za cilindrične geometrije.

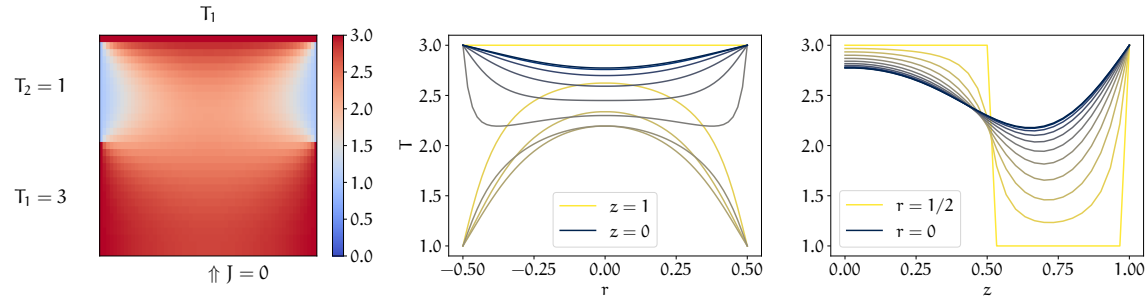
Sedaj izolirajmo zgornjo polovico plašča valja. Rezultat je prikazan na sliki 16. Zares, ničelni odvod temperature na zgornji polovici potrdi, da je plašč tam izoliran.



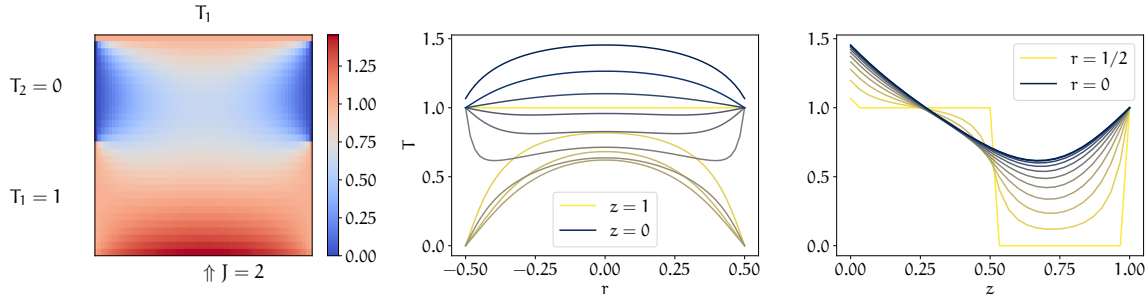
Slika 10: Temperaturni profil 1.



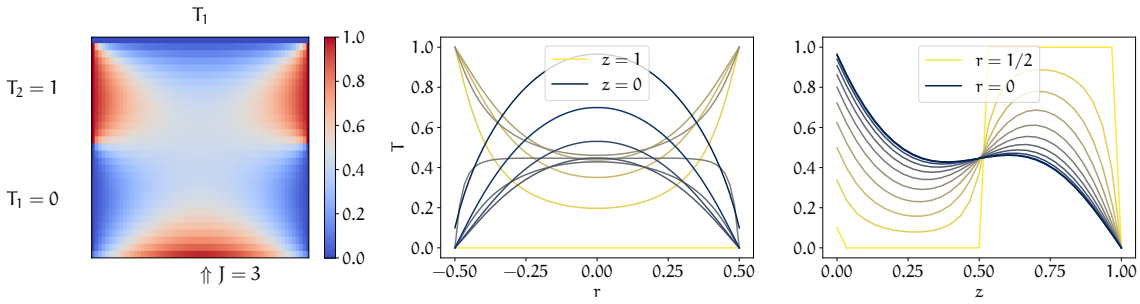
Slika 11: Temperaturni profil 2.



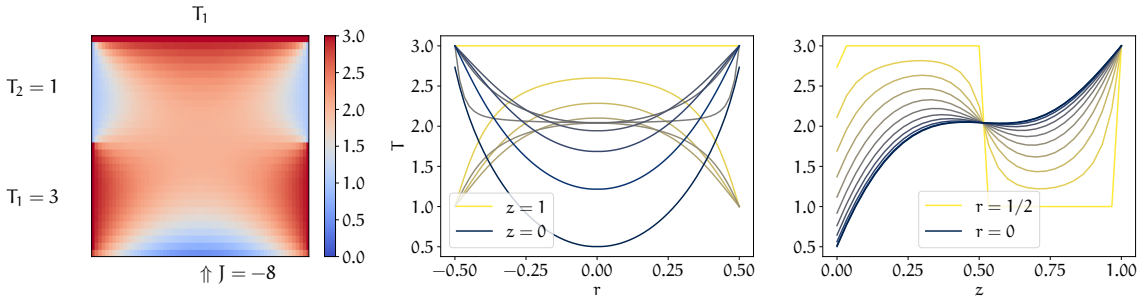
Slika 12: Temperaturni profil 3.



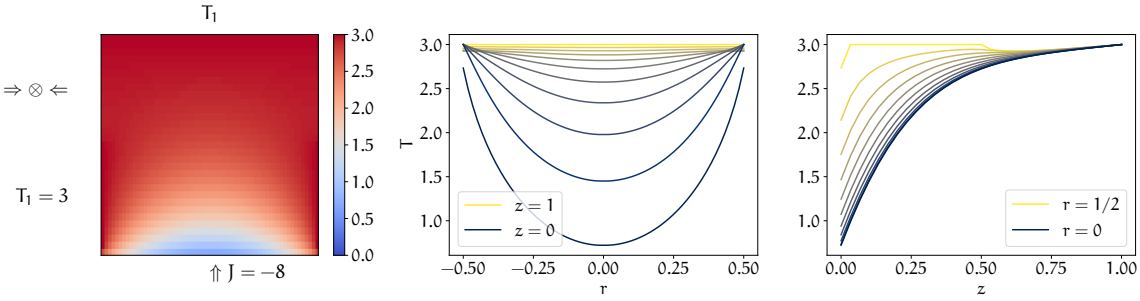
Slika 13: Temperaturni profil 4.



Slika 14: Temperaturni profil 5.



Slika 15: Temperaturni profil 6.



Slika 16: Temperaturni profil 7.