205: Parcialne diferencialne enačbe: robni problemi in relaksacija

Peter Rupnik

27. marec 2024

Metoda pospešene relaksacije (SOR - Successive Over-Relaxation) je iterativna metoda, ki nadgradi Jacobijev in Gauss-Seidlov postopek tako, da poveča iteracijski korak za faktor $1 \leq \omega < 2$:

$$u_i^{(n+1)} = u_i^{(n)} + \omega \left(\tilde{u}_i^{(n)} - u_i^{(n)} \right)$$

kjer je $\tilde{u}_i^{(n)}$ korak običajne Jacobijeve ali Gauss-Seidlove iteracije, in je za Poissonovo enačbo $\nabla^2 u = q$ na ekvidistančni kartezični mreži v2D enak $\tilde{u}_i = \frac{1}{4} \left(\sum_{j \in \text{ soseedi}} u_j - q_i \Delta x^2 \right)$. Po zgledu Gauss-Seidlovega postopka sta dva soseda že popravljena, dva pa se iz prejšnje iteracije. Optimalna izbira faktorja pospešitve ω je odvisna od spektralnega radija iteracijske matrike ρ_{Jacobi} ,

$$\omega = \frac{2}{1 + \sqrt{1 - \rho_{\text{Jacobi}}^2}} \quad \text{ali približno} \quad \omega = \frac{2}{1 + \frac{\pi}{N}}$$

kjer je N število tock po enem robu. Ta ocena za ω velja za poln kvadrat; pri drugačnih oblikah dobis boljso oceno z nastavkom:

$$\omega = \frac{2}{1 + \alpha \frac{\pi}{N}}$$

Še boljša je metoda Čebiševega pospeška, pri kateri zgornji iteracijski postopek simetriziramo tako, da ga izvajamo izmenično po zgolj belih in zgolj črnih poljih šahovnice, tako da so vsi sosedi iz iste politeracije. Na vsake pol iteracije popravimo vrednost faktorja ω po postopku, ki asimptotično konvergira k zgornji vrednosti:

$$\omega^{(0)} = 1, \quad \omega^{(1/2)} = \frac{1}{1 - \frac{1}{2}\rho_{J^2_{\text{acobi}}}}, \quad \omega^{(n+1/2)} = \frac{1}{1 - \frac{1}{4}\rho^2_{\text{Jacobi}}\, \omega^{(n)}}$$

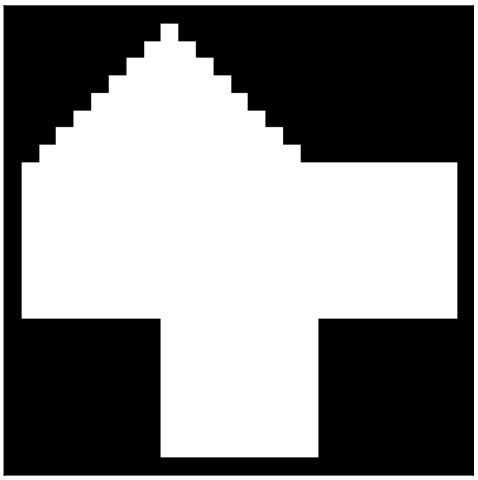
1. Z metodo pospešene relaksacije (SOR) določi Poiseuillov koeficient za pretok viskozne tekočine po cevi s prerezom, podanim s spodnjo shemo. Pazi na pravilno obravnavo primerov, ko je rob točno na mrežni točki in ko je rob med sosednjima točkama. Poišči optimalno vrednost parametra α. Preveri tudi uspešnost Čebisevega pospeška k relaksaciji.

2. Enakostraničen kovinski valj ($2r = h$) grejemo na spodnji ploskvi s konstantnim toplotnim tokom, plašč prerežemo horizontalno na dve enako široki polovici pri različnih temperaturah, zgornjo ploskev pa držimo na isti temperaturi kot spodnjo polovico plašča. Določi temperaturni profil! Kaj pa, če zgornjo polovico plašča toplotno izoliramo?

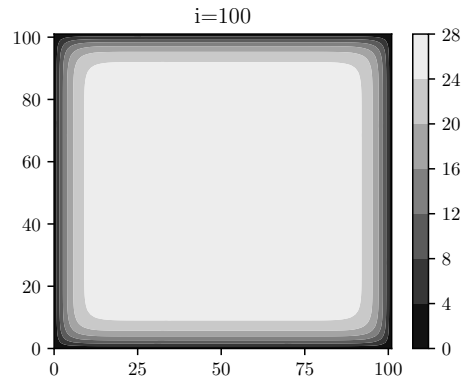
1. naloga

1.1 Tipanje problema

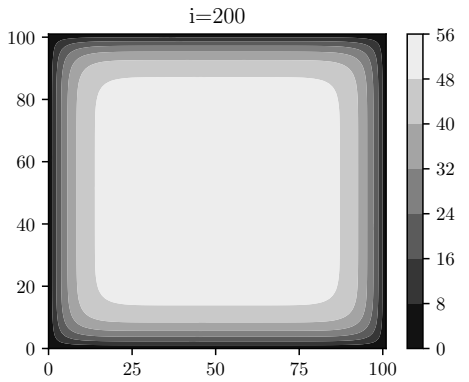
Obravnave se lotim z določanjem reprezentacije cevi v matrični obliki za n elementov v vsaki dimenziji (zahtevam še $n \mod 3 = 0$). Za $n = 27$ dobim spodnjo sliko. Poleg geometrijskih atributov sem na koncu zahteval še 'črn rob' okrog cevi, kjer je hitrost ničelna.



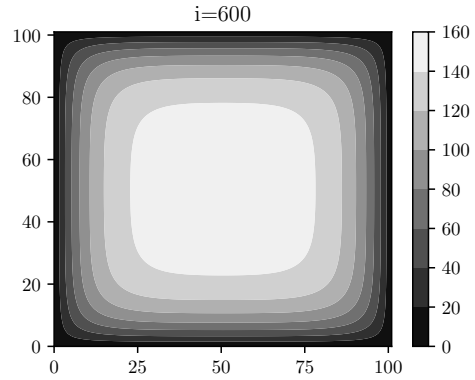
Najprej sem poskusil Jacobijevo metodo na kvadratni cevi, da se uverim, ali je problem zastavljen pravilno.



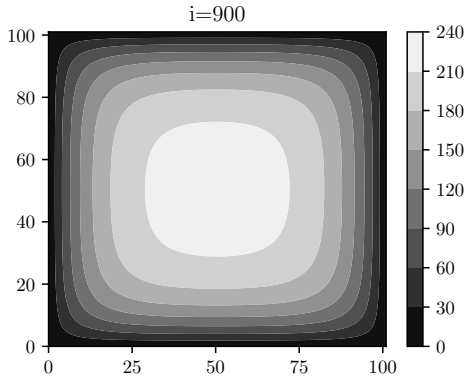
Hitrostno polje v kvadratni cevi po 100 korakih iteracije.



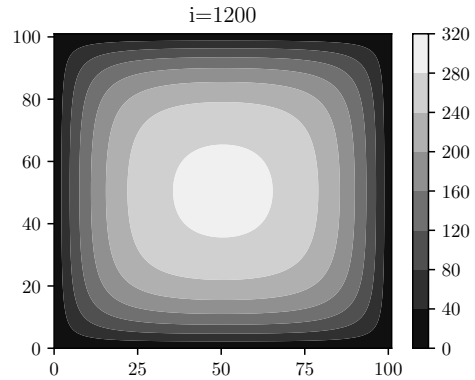
Hitrostno polje v kvadratni cevi po 200 korakih iteracije.



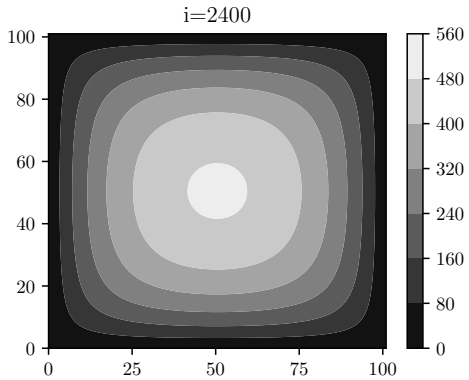
Hitrostno polje v kvadratni cevi po 600 korakih iteracije.



Hitrostno polje v kvadratni cevi po 900 korakih iteracije.



Hitrostno polje v kvadratni cevi po 1200 korakih iteracije.



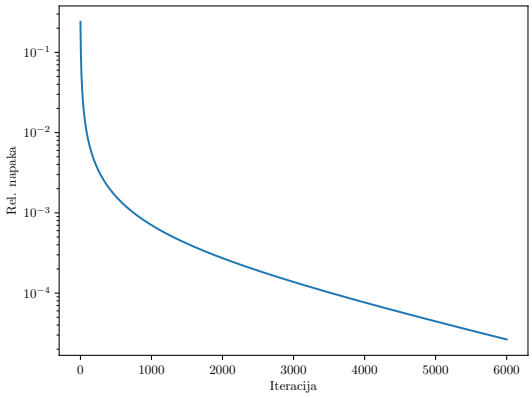
Hitrostno polje v kvadratni cevi po 2400 korakih iteracije.

Validacija rezultatov je na tej točki predvsem optična, zato velja za primerjavo metod v nadaljevanju uvesti primerne metrike, denimo relativno razliko med iteracijami, ali pa kar Poiseuillov koeficient. Za začetek sem uvedel preprosto metriko, ki vrne razmerje med vsoto absolutnih razlik po elementih dveh matrik in vsoto po elementih prve matrike:

```
def rel_err(a1: numpy.ndarray, a2: numpy.ndarray) -> float:

    return np.sum(np.abs(a1-a2).reshape(-1))/np.sum(np.abs(a1.reshape(-1
                                                    )))
```

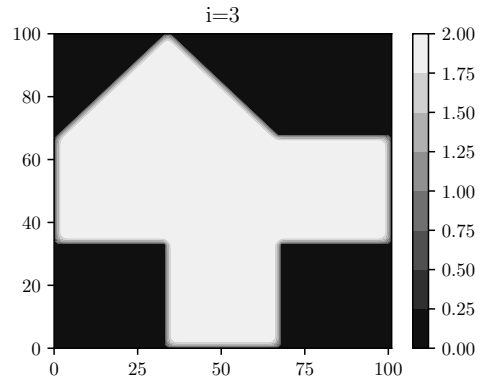
Opazim, da sprva hitro približevanje konvergenci preide v počasnejše, a monotono padanje.



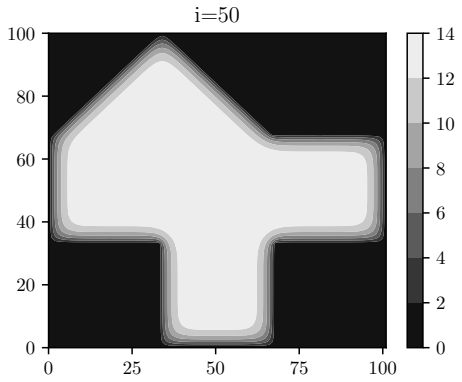
Nadaljujem lahko s podanim profilom cevi.

1.2 Rešitev za podan profil cevi

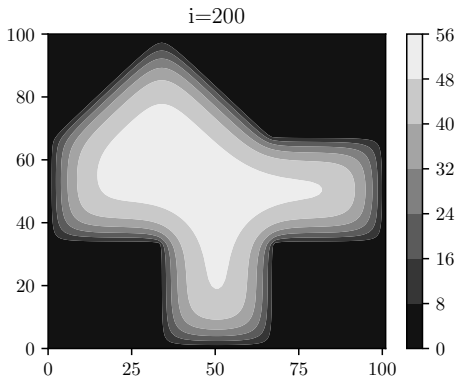
S pripravljeno matriko za profil cevi lahko postopek za kvadratno cev hitro popravim za dano cev. Nadaljnja metodologija je enaka.



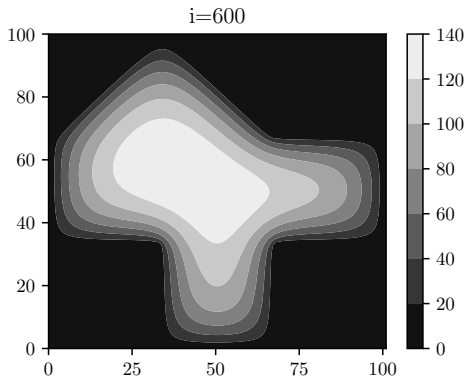
Hitrostno polje v podani cevi po treh korakih iteracije.



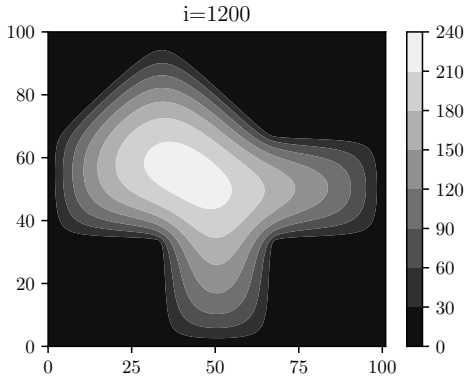
Hitrostno polje v podani cevi po 50 korakih iteracije. Opazimo enake tendence povečevanja hitrostnega polja kot pri okrogli cevi.



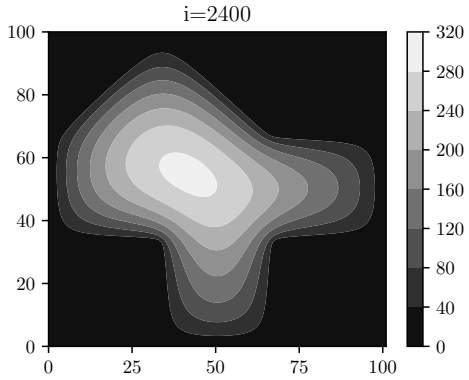
Hitrostno polje v podani cevi po 200 korakih iteracije.



Hitrostno polje v podani cevi po 600 korakih iteracije.

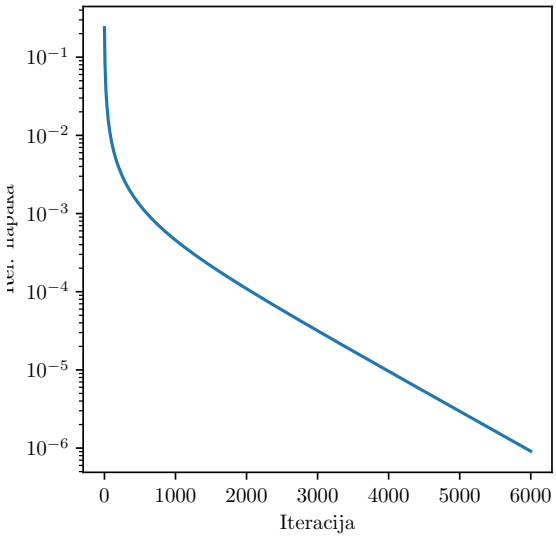


Hitrostno polje v podani cevi po 1200 korakih iteracije.



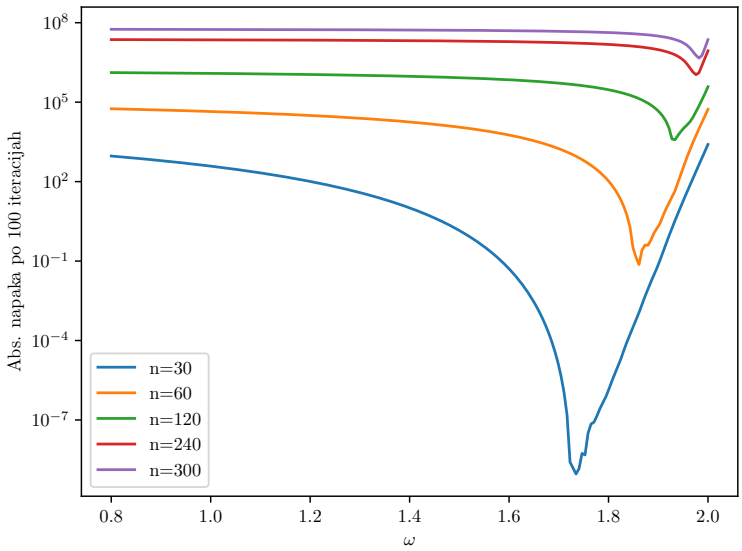
Hitrostno polje v kvadratni cevi po 2400 korakih iteracije.

Spet lahko pogledamo hitrost konvergence. Kot je razvidno iz točke, ko razlika med zaporednima iteracijama pade na 0, je v trenutnem primeru konvergenca hitrejša.



1.3 Implementacija pospešene iteracijske sheme

Nadaljeval sem z implementacijo metode SOR. Popravljena koda je delovala odlično, evalvacija pa mi je povzročala nekaj težav. Končno sem se odločil za sledeč postopek: generiral sem 'pravo' hitrostno polje s 1000 koraki iteracije, nato pa sem pri posameznih vrednostih ω z metodo SOR iteriral le 100 korakov. Rezultata sem primerjal z vsoto absolutnih razlik po elementih.

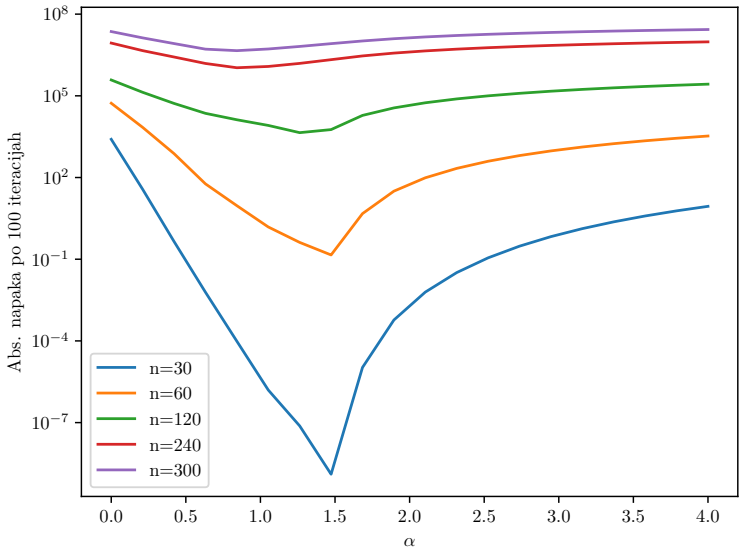


Kot smo pričakovali s predavanj, je optimalna ω odvisna od dimenzije problema. Naraščanje napake (neodvisno od ω) lahko pojasnimo s tem, da je matrika hitrostnega polja večja, zaradi česar je tudi razlika med 'pravim' in trenutnim hitrostnim poljem večja.

Nadaljujem z uvedbo parametra α, kot je definiran v navodilih:

$$\omega = \frac{2}{1 + \alpha \frac{\pi}{N}}.$$

Kot razvidno na sliki spodaj, je zdaj optimalen pospešek invarianten od števila točk na mreži.



1.4 Poiseuillov koeficient

Zdaj lahko z optimalnim pospeškom iščem Poiseuillov koeficient za dano cev. Zanj velja:

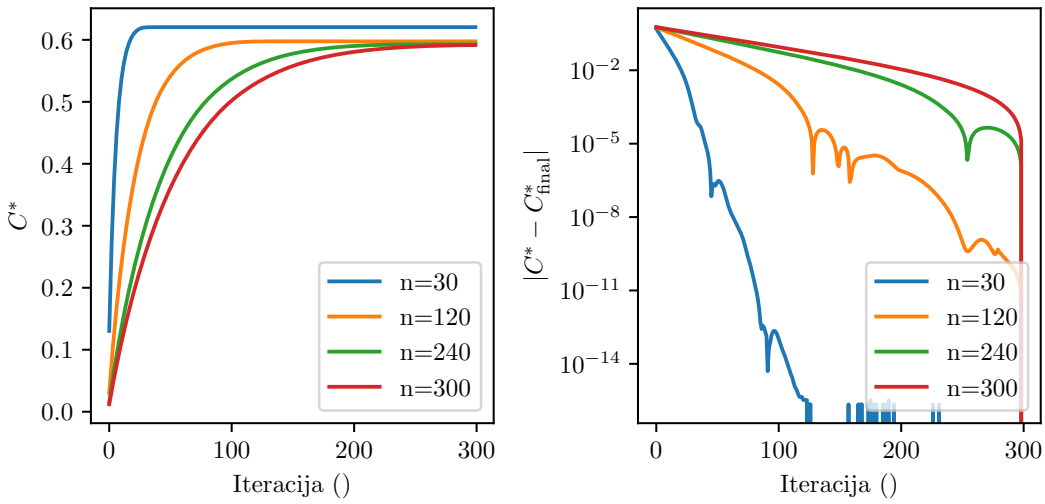
$$\Phi_V = \iint u(x,y)\,\mathrm{d}S = \frac{CS^2}{8\pi\eta}\frac{\partial p}{\partial z}.$$

Izrazim lahko

$$C^* = C\eta^{-1}\frac{\partial p}{\partial z},$$

za presek cevi pa vzamem analitično vrednost $S = \frac{5}{9}$.

Za izračun Φ sem poskusil integracijo z dvodimenzionalnim Simposonom, s svojo implementacijo dvodimenzionalne verzije metode za trapezijsko integracijo `numpy.trapz`, pa tudi z rudimentarno sumacijo. Medsebojno so se vse metode razlikovale pod 0.1%. Pri iteraciji sem uporabil zgoraj določen optimalen parameter α za pospešek iteracije.



Vrednost modificiranega Poiseuillovega koeficienta C^*med iteriranjem in absolutne razlike med končnim in trenutnim C^* za različna števila točk v vsaki dimenziji n.

2. naloga

2.1 Implementacija

Zaradi simetrije problema bom delala v cilindričnih koordinatah. Za Laplaceov operator tako velja

$$\nabla^2 u = \frac{1}{r}\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial u}{\partial r}\right) + \frac{\partial^2 u}{\partial z^2} = \frac{1}{r}\frac{\partial u}{\partial r} + \frac{\partial^2 u}{\partial r^2} + \frac{\partial^2 u}{\partial z^2},$$

oziroma diskretizirano:

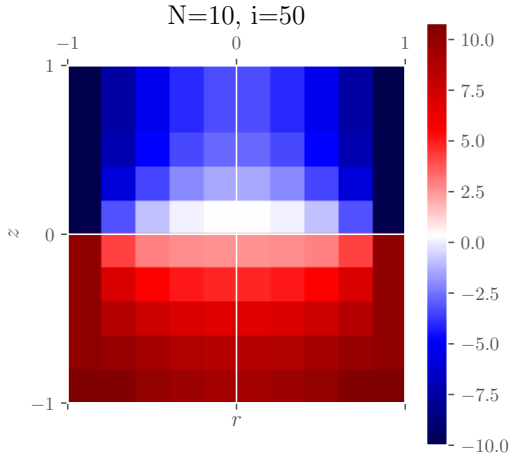
$$\frac{\partial^2 u(r,z)}{\partial z^2} = \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{h^2}$$
$$\frac{\partial^2 u(r,z)}{\partial r^2} = \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{h^2}.$$

Zaradi simetrije problema bi zadostovalo, da modeliramo samo pozitivne r, bi pa morali bržčas poskrbeti tudi za dodaten pogoj, da je pri $r = 0$ toplotni tok v radialni smeri ničeln, zato sem raje modeliral cel interval $r \in [-1, 1]$.

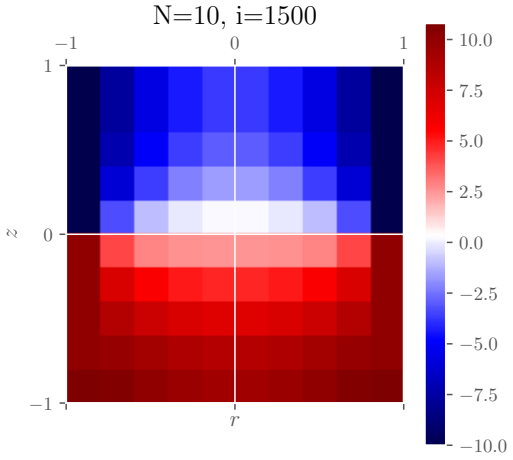
Robne pogoje na plašču je lahko definirati: tam matrične elemente le postavimo na fiksni temperaturi. Na zgornji osnovni ploskvi je valj toplotno izoliran, zato tam toplotnega toka ne sme biti, kar je ekvivalentno pogoju, da je prvi odvod v navpični smeri ničeln. Analogno na spodnji osnovni ploskvi zahtevamo nek končen toploten tok, kar pomeni enak odvod v navpični smeri za vse matrične elemente pri $z = 0$.

Obe podnalogi sem implementiral hkrati, saj je prva podnaloga le poseben primer druge. Zgornjo polovico sem držal na $T_1 = -10$, spodnjo pa na $T_2 = 10$, za spodnjo osnovno ploskev sem predpisal toplotni tok v valj z vrednostjo 15.

Najprej sem pogledal Jacobijevo metodo z diskretizacijo na 10 točk v vsaki dimenziji. Uveril sem se, da metoda deluje tudi pri $r = 0$ (ali bolje, da sem se točki pri $r = 0$ uspešno izognil z izborom diskretizacije). Kot vidimo spodaj metoda tudi hitro konvergira.

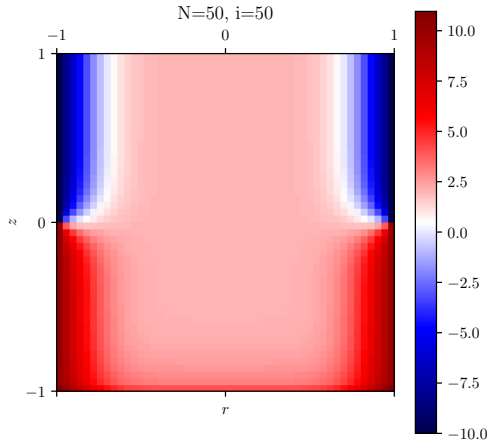


Temperaturno polje v prerezu valja po 50 korakih iteracije.

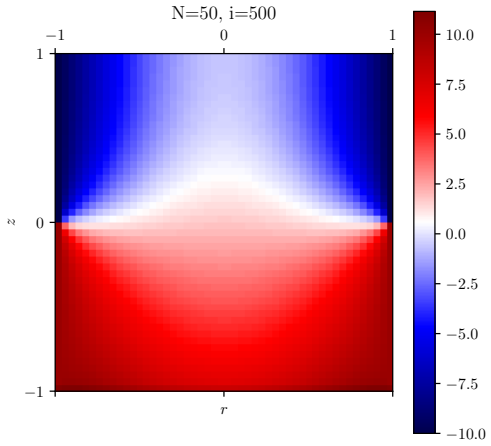


Temperaturno polje v prerezu valja po 1500 korakih iteracije.

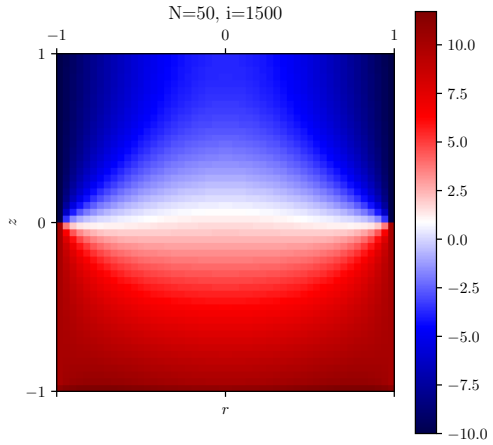
Nadaljeval sem s 50 točkami v vsaki dimenziji. Tu je sicer konvergenca počasnejša, se pa lepše opazijo atributi modela, predvsem pričakovano obnašanje pri zgornji osnovni ploskvi, kjer se izoterme sekajo s ploskvijo pod pravim kotom, kar pomeni pričakovano obnašanje, če naj tam ne bo toplotnega toka v ploskev.



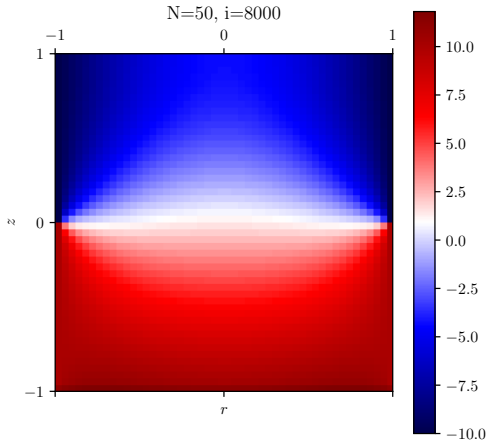
Temperaturno polje v prerezu valja po 50 korakih iteracije.



Temperaturno polje v prerezu valja po 500 korakih iteracije.

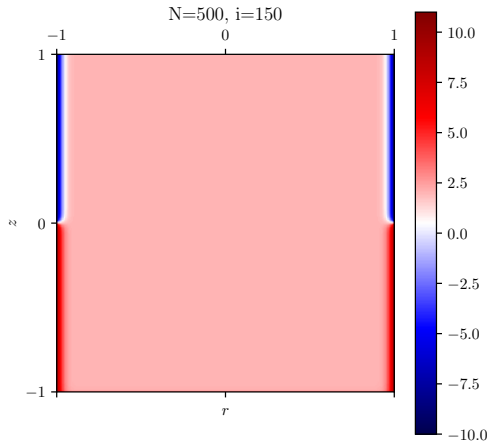


Temperaturno polje v prerezu valja po 1500 korakih iteracije.

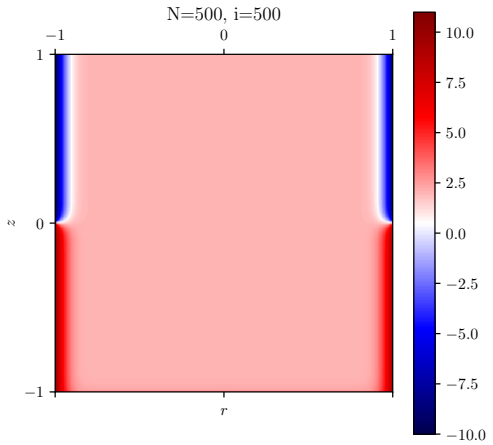


Temperaturno polje v prerezu valja po 3000 korakih iteracije.

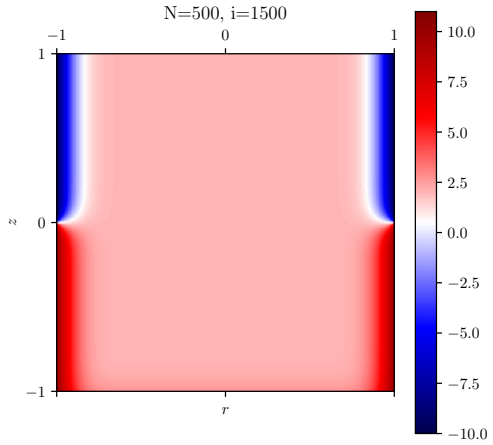
Nadaljeval sem s povečevanjem števila točk in opazovanjem konvergence. Kot je vidno na spodnjih slikah, pri 500 točkah v vsaki dimenziji tudi po 8000 iteracijah še ne dosežemo konvergence.



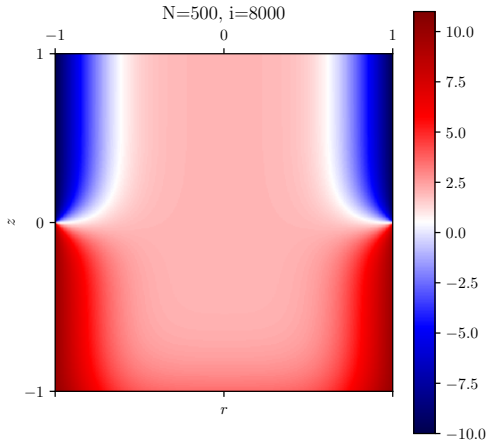
Temperaturno polje v prerezu valja po 50 korakih iteracije.



Temperaturno polje v prerezu valja po 500 korakih iteracije.



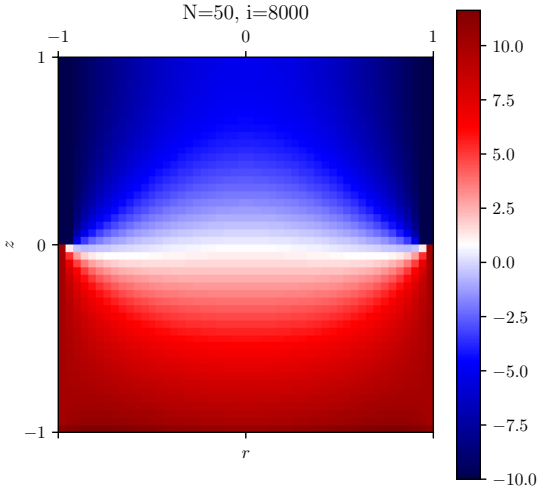
Temperaturno polje v prerezu valja po 1500 korakih iteracije.



Temperaturno polje v prerezu valja po 8000 korakih iteracije.

2.2 Toplotna izolacija zgornje polovice valja

Toplotno izolacijo modeliram tako, da zahtevam $\frac{\partial T}{\partial r} = 0$ na rzgornji polovici plašča, kar dobim z izenačitvijo temperature drugega in predzadnjega elementa v vsaki vrstici s temperaturo prvega in zadnjega elementa. Kvalitativno se temperaturni profil ne spremeni drastično.

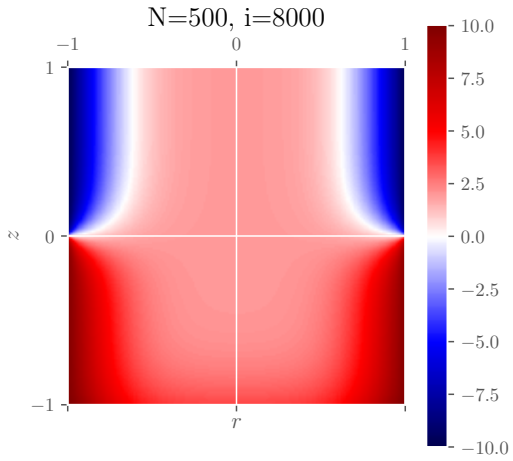


Valj z izolirano zgornjo polovico plašča.

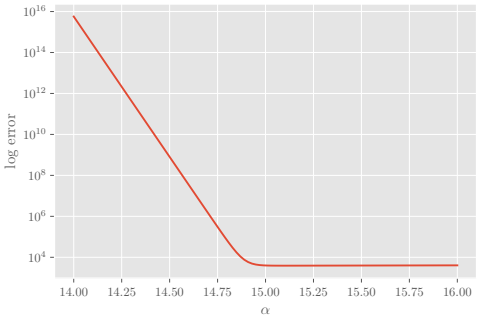
2.3 Pospešena relaksacija

Pri vpeljavi pospešene iteracijske sheme sem imel tokrat več težav kot pri prejšnji nalogi. Iz neznanega razloga skaliranje parametra α s številom točk ni uspelo, kar je pomenilo

več ugibanja in ročne optimizacije. Kot vidimo na sliki spodaj tudi rezultati niso bistveno pospešeni.



Končni rezultat po 8000 iteracijah mreže s 500 točkami. Stacionarnega stanja kljub pospešitvi nismo dosegli.



Obnašanje pri iskanju optimalnega parametra α.