205: Parcialne diferencialne enačbe: robni problemi in relaksacija

Peter Rupnik

25. 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:

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

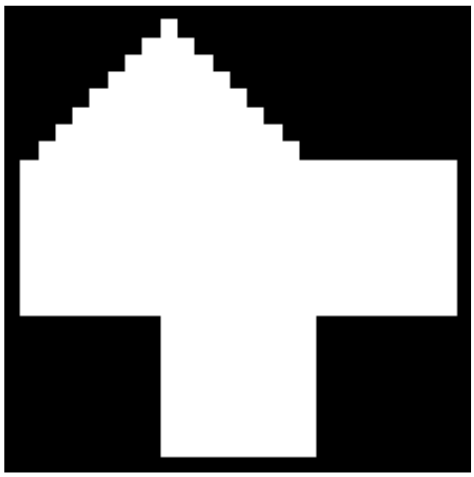
1. Z metodo pospešene relaksacije (SOR) določi Poiseuillov koeficient za pretok viskozne tekočine po cevi s prerezom, podanim s spodnjo shemo. Pazi na pravilno obravnavo primerov, ko je rob točno na mrežni točki in ko je rob med sosednjima točkama. Poišči optimalno vrednost parametra α. Preveri tudi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

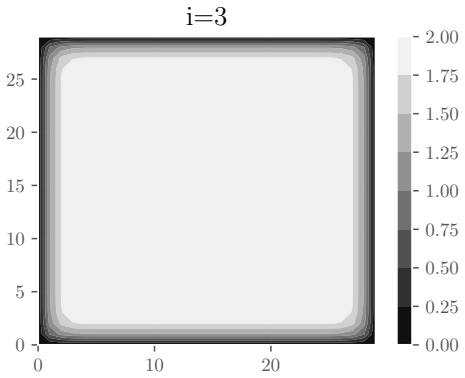
1. naloga

1.1 Tipanje problema

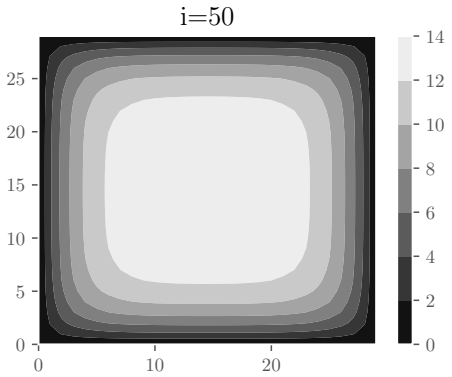
Obravnave se lotim z določanjem reprezentacije cevi v matrični obliki. Dobim spodnjo sliko. Poleg geometrijskih atributov sem na koncu zahteval še 'črn rob' okrog cevi, da verno opišem problem. Število točk v vsaki dimenziji znaša 27.



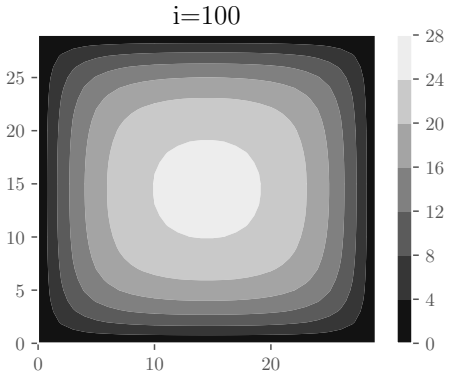
Namesto reševanja sistema diskretnih diferencialnih enačb v matrični obliki sem se lotil poskušanja z iterativnimi metodami. Najprej sem poskusil Jacobijevo metodo na kvadratni cevi, da se uverim, ali je problem zastavljen pravilno:



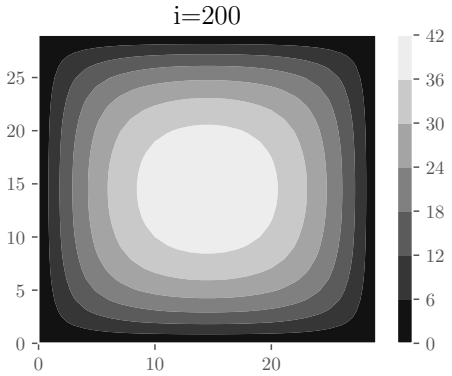
Hitrostno polje v kvadratni cevi po treh korakih iteracije.



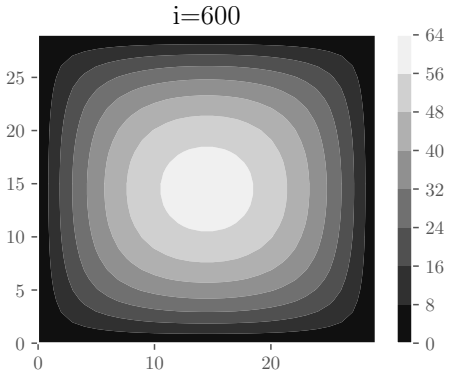
Hitrostno polje v kvadratni cevi po 50 korakih iteracije. Maksimalna hitrost se je povečala, kar lahko kaže na slabo zastavljen postopek.



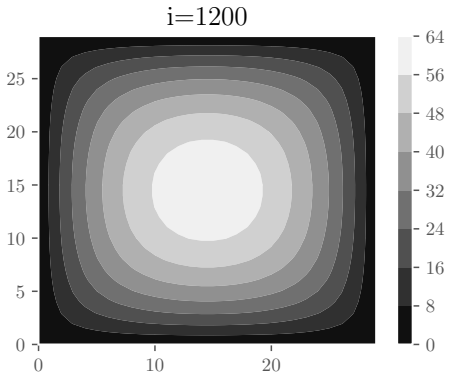
Hitrostno polje v kvadratni cevi po 100 korakih iteracije.



Hitrostno polje v kvadratni cevi po 200 korakih iteracije.



Hitrostno polje v kvadratni cevi po 600 korakih iteracije. Z metodo ostrega pogleda sem ocenil, da smo okrog iteracije 400 dosegli konvergenco hitrostnega polja.



Hitrostno polje v kvadratni cevi po 1200 korakih iteracije. Tudi po trikrat prekoračeni ocenjeni točki konvergence je hitrostno polje konstantno.

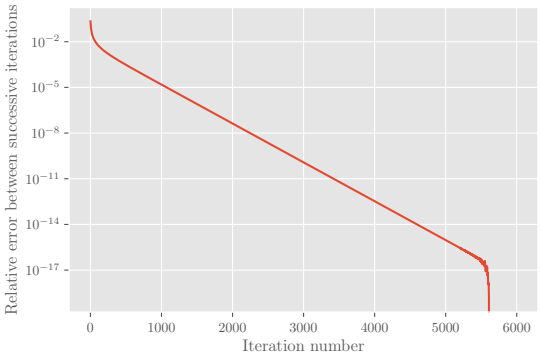
Validacija rezultatov je na tej točki predvsem optična, zato velja za primerjavo metod v nadaljevanju uvesti primerne metrike, denimo relativno razliko med iteracijami, ali pa kar Poiseuillov koeficient. Ker je slednja opcija zahtevnejša, sem implementiral preprosto

funkcijo:

```
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
                                                    )))
```

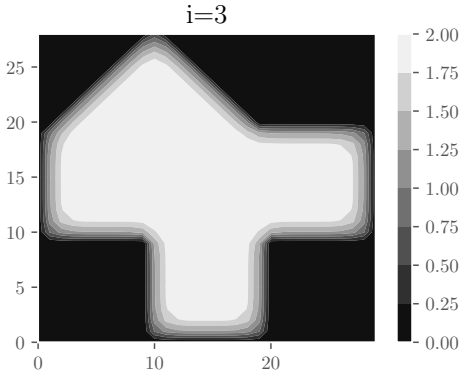
Opazim, da sprva hitro približevanje konvergenci preide v počasnejše, a monotono padanje vse do približno iteracije številka 600, ko postaneta zaporedna hitrostna polja do strojne natančnosti enaka.



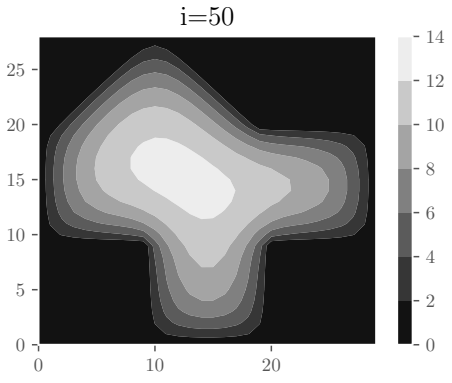
Nadaljujem lahko s podanim profilom cevi.

1.2 Rešitev za podan profil cevi

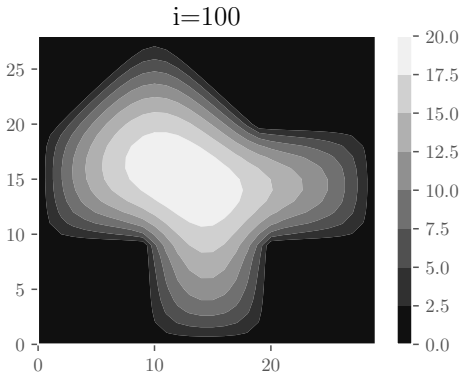
S pripravljeno matriko za profil cevi lahko postopek za kvadratno cev hitro popravim za dano cev. Nadaljnja metodologija je enaka.



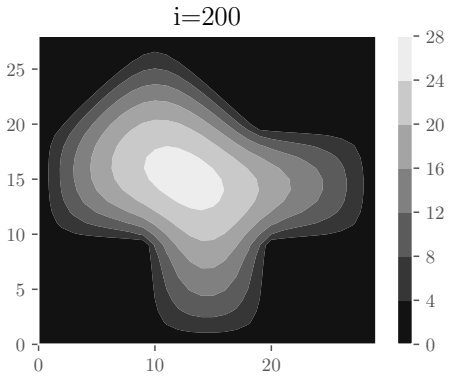
Hitrostno polje v podani cevi po treh korakih iteracije.



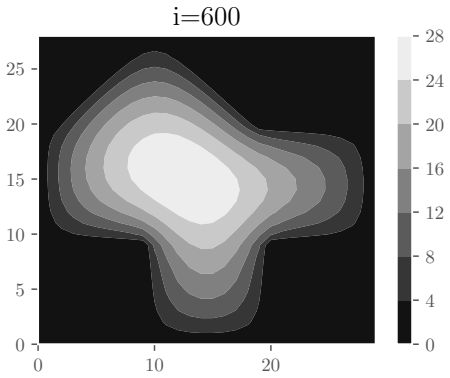
Hitrostno polje v podani cevi po 50 korakih iteracije. Opazimo enake tendence povečevanja hitrostnega polja kot pri okrogli cevi.



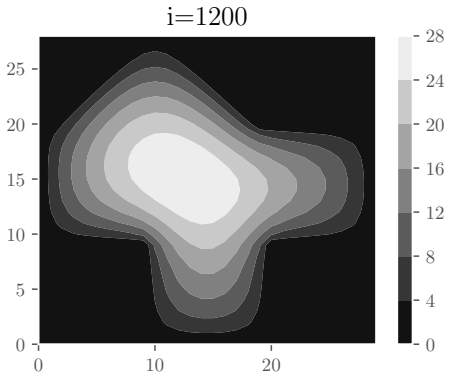
Hitrostno polje v podani cevi po 100 korakih iteracije.



Hitrostno polje v podani cevi po 200 korakih iteracije.

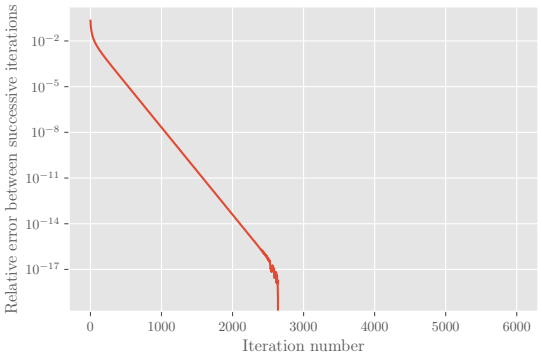


Hitrostno polje v podani cevi po 600 korakih iteracije.



Hitrostno polje v kvadratni cevi po 1200 korakih iteracije.

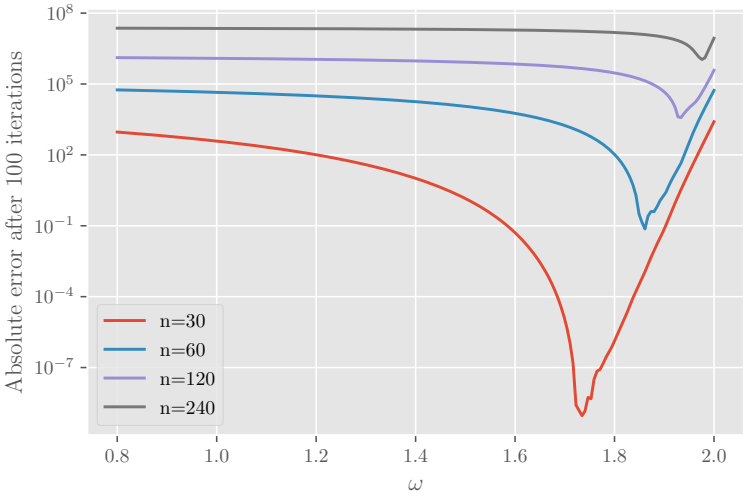
Spet lahko pogledamo hitrost konvergence. Kot je razvidno iz točke, ko razlika med zaporednima iteracijama pade na 0, je v trenutnem primeru konvergenca hitrejša.



1.3 Implementacija pospešene iteracijske sheme

S predavanj vemo, da bi Gauss-Seidlov postopek izboljšal hitrost konvergence le za konstanten faktor, zato sem raje nadaljeval 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.

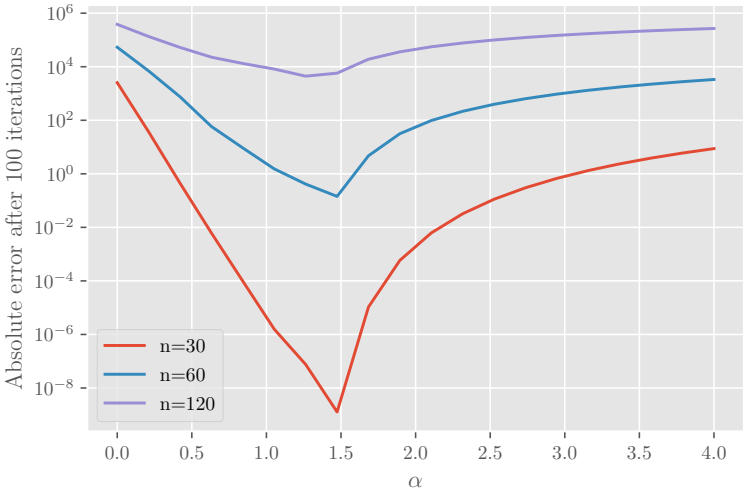


Kot smo pričakovali s predavanj, je optimalna ω odvisna od dimenzije problema. Naraščanje napake (neodvisno od ω) lahko pojasnimo s tem, da je matrika hitrostnega polja večja, zaradi česar je tudi razlika med 'pravim' in trenutnim hitrostnim poljem večja.

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:

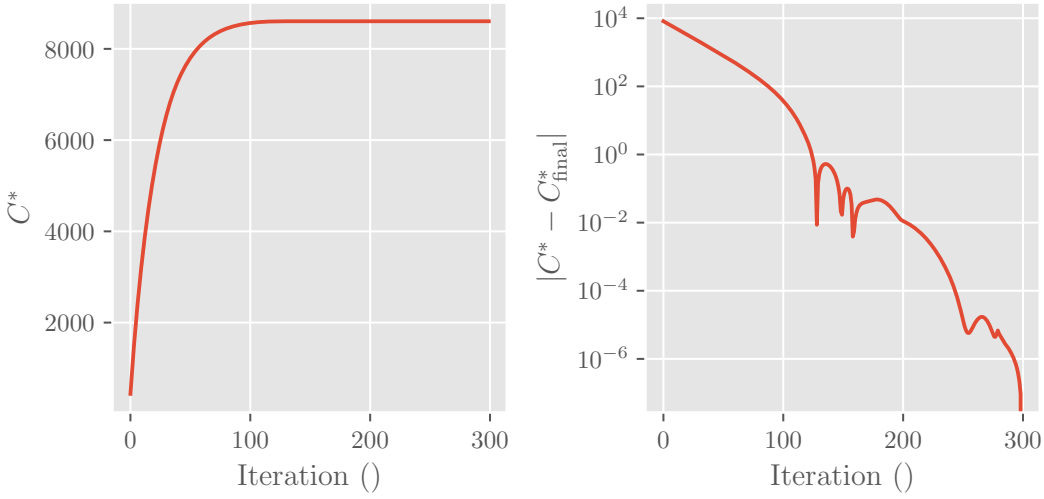
$$\Phi_V = \iint u(x,y)\,\mathrm{d}S = \frac{CS^2}{8\pi\eta}\frac{\partial p}{\partial z}.$$

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 najbolj rudimentarno sumacijo. Medsebojno so vse metode dosegle natančnosti pod 0.1%. Pri iteraciji sem uporabil zgoraj določen optimalen parameter α za pospešek iteracije.



Razvoj Poiseuillovega koeficienta skozi iteracijo. Opazimo hitro stabilizacijo vrednosti, po kateri so popravki venomer manjši. Diskretizacija mreže je v vsaki dimenziji znašala 120 točk.

2. naloga