Direktno reševanje Poissonove enačbe

Andrej Kolar-Požun, 28172042

17. maj 2018

1 Poves opne

V tej nalogi bomo izkoristali dejstvo, da znamo nekatere enačbe v določenih preprostih geometrijah in z preprostimi robnimi pogoji delno ali v celoti analitično rešiti. Začnimo z neobteženo kvadratno opno, rešujemo enačbo(v ustreznih enotah)

$$\nabla^2 u(x,y) = -p(x,y)$$

Z Dirichletovimi robnimi pogoji $u(x,y)=0$ na robu.

Z izbiro sinusne Fouriere transformacije bomo robne pogoje že takoj upoštevali:

$$G^{mn} = \sum_{j=1}^{J-1}\sum_{l=1}^{L-1} g_{jl} \sin\frac{\pi jm}{J} \sin\frac{\pi ln}{L}$$
$$U^{mn} = \sum_{j=1}^{J-1}\sum_{l=1}^{L-1} u_{jl} \sin\frac{\pi jm}{J} \sin\frac{\pi ln}{L}$$

Vstavimo v enačbo(Laplacian zapišemo v diskretni obliki) in dobimo:

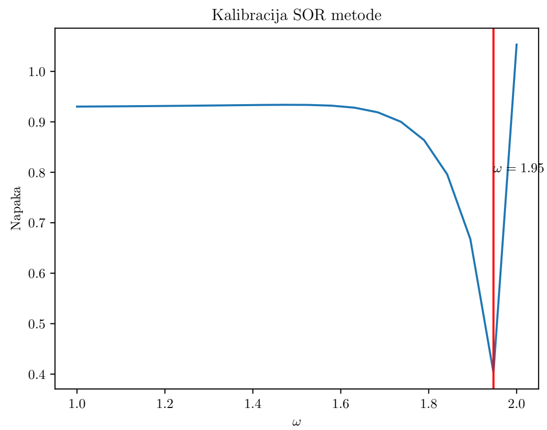
$$U^{mn} = h^2/2 \frac{G^{mn}}{\cos(\pi m/J) + \cos(\pi n/L) - 2}$$

Rešitev dobimo z inverzno sinusno transformacijo:

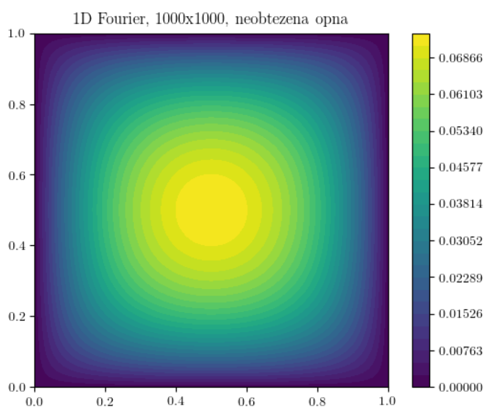
$$u_{jl} = \frac{2}{J}\frac{2}{L}\sum_{m=1}^{J-1}\sum_{n=1}^{L-1} U^{mn} \sin\frac{\pi jm}{J} \sin\frac{\pi ln}{L}$$

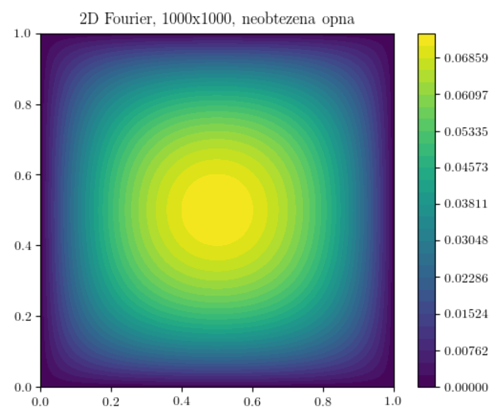
Še ena možnost je, da Fourierovo transformacijo napravimo le v eni izmed dimenzij(izbral si bom x), v drugi nam potem ostane preprosta diferenčna shema

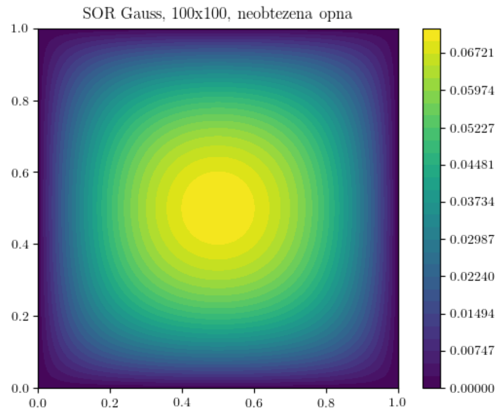
$$U_l^m = \sum_n u_{ln} \sin\frac{\pi nm}{N}$$
$$G_l^m = \sum_n g_{ln} \sin\frac{\pi nm}{N}$$
$$U_{l-1}^m - U_l^m(4 - 2\cos(m\pi/J)) + U_{l+1}^m = h^2 G_l^m$$



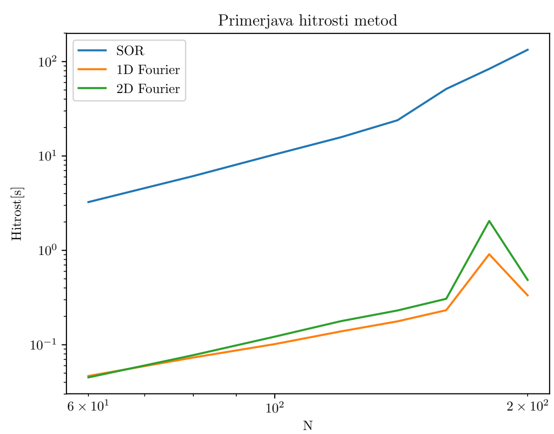
Najprej na hitro preverimo kakšen je optimalen omega za SOR v tem primeru. Rezultat po 50 iteracijah primerjamo s tistim, ki smo ga dobili tako, da smo iterirali dokler ni bil rezidum 0.0001(pribl 500 iteracij)





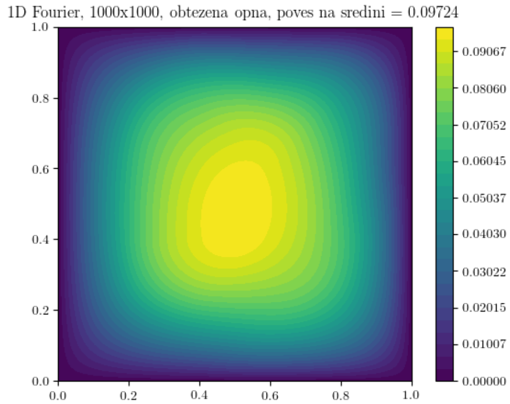


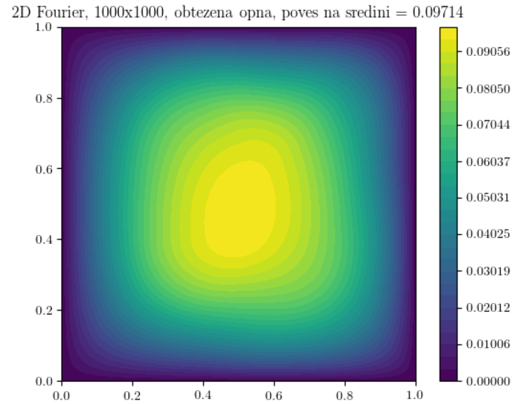
SOR je bil tako počasen, da sem ga raje izvedel na manjši mreži. Vidimo, da naše 3 metode dajo smiselen rezultat.

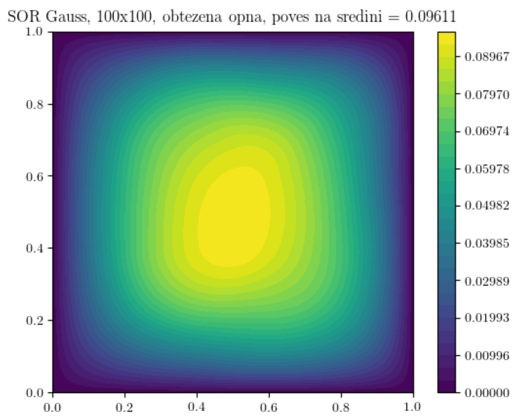


Tukaj vidimo, zakaj se splača uporabiti Fourierovo metodo, kljub temu, da nam SOR da prave rezultate. Veliko hitrejša je, pa tukaj je bil zaustavitveni pogoj za iteracijo še majhen - norma 0.1 vektorja reziduala.

Poglejmo si še obteženo opno, kjer je ta obtežena po porazdelitvi mase kot v 6. nalogi(glej 6. nalogo za sliko)







2 Temperaturni profil

Spet rešujemo temperaturni profil v valju. Če zapišemo Laplacian v cilindričnih koordinatah dobimo:

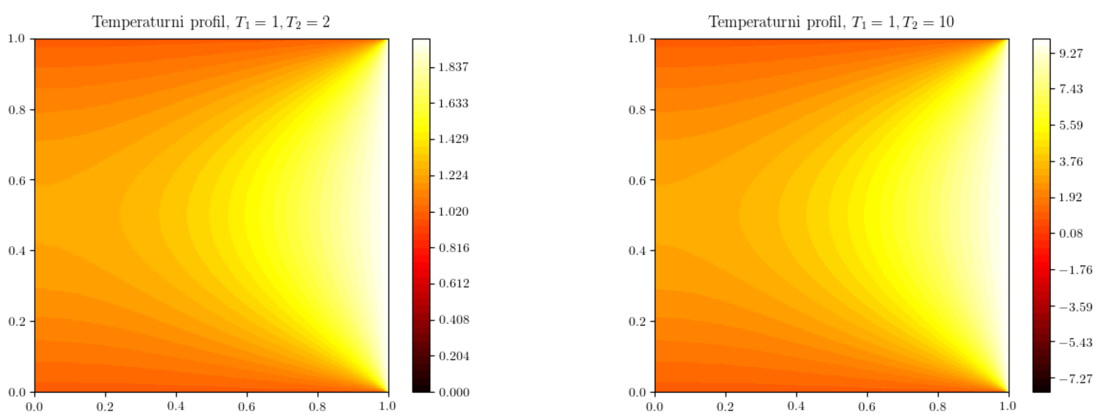
$$\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} + \frac{\partial^2 u}{\partial z^2} = 0$$

Tisto deljenje z radijem je grdo, če bi se zadeve lotili z Fourierovo transformacijo v r smeri. Zato jo bomo delali le v z smeri, v r smeri pa obdržali diferenčno shemo.

Naši robni pogoji so, da imamo določeno temperaturo T_1 na osnovnih ploskvah valja, ter temperaturo T_2 na plašču. Za toplotni profil rešujemo Laplacevo enačbo $\nabla^2 T = 0$. Lahko definiramo novo spremenljivko $u = T - T_1$, za katero se enačba spet glasi $\nabla^2 u = 0$, s tem da imam sedaj za u homogeni robni pogoj v z smeri in lahko spet uporabimo sinusno transformacijo. Na plašlu bomo potem zahtevali temperaturo $T_2 - T_1$ na koncu pa spet prešli na spremenljivko T $T = u + T_1$

Dobimo:

$$
\begin{aligned}
&\frac{u_{k+1}^j + u_{k-1}^j - 2u_k^j}{h^2} + \frac{1}{kh}\frac{u_{k+1}^j - u_{k-1}^j}{2h} + \frac{u_k^j}{h^2}\left(2\cos(\pi * j/N) - 2\right) = 0 \\
&\left(2\cos(\pi j/N) - 4\right) u_k^j + \left(1 - \frac{1}{2k}\right) u_{k-1}^j + \left(1 + \frac{1}{2k}\right) u_{k+1}^j = 0 \\
&\tilde{u}_k^j = \frac{1}{2\cos(\pi j/N) - 4}\left(-\left(1 - \frac{1}{2k}\right) u_{k-1}^j - \left(1 + \frac{1}{2k}\right) u_{k+1}^j\right) \\
&\tilde{u}_k^j = \frac{1}{2\cos(\pi j/N) - 4}\left(-\left(1 - \frac{1}{2k}\right) u_{k-1}^j - \left(1 + \frac{1}{2k}\right) u_{k+1}^j\right) \\
&u_k^{j(n+1)} = u_k^{j(n)} + \omega\left(\tilde{u}_k^{j(n)} - u_k^{j(n)}\right)
\end{aligned}
$$



Slike pridejo takšne kot jih pričakujemo

Na hitro preverimo še če bi šlo z Neumannovim robnim pogojem. V tem primeru bi fiksiral tok na ploksvi in uporabil kosinusno transformacijo. Nova spremenljivka bi se glasila $T = u + z * j$, kjer ima u ničelni odvod na robu.

3 Sinusna transformacija

Scipyjeva funkcija za večdimenzionalno sinusno transformacijo mi je nekako dajala napačen rezultat in se zdela nekako čudno definirana.

Zato sem jo raje sprogramiral sam z učinkovitim algoritmom iz knjige Numerical Recipes, ki ga bom na kratko napisal tu:

Iz naših podatkov f_j želimo dobiti sinusno transformiranko F_k. Najprej tvorimo novo zaporedje:

$$
\begin{aligned}
&y_0 = 0 \\
&y_j = \sin(j * \pi/N)(f_j + f_{N-j}) + 0.5(f_j - f_{N-j})
\end{aligned}
$$

Na zaporedju y uporabimo algoritem za realen FFT. Izkaže se, da je realni del te transformiranke enak $F_{2k+1} - F_{2k-1}$, imaginarni pa F_{2k}. Tako takoj dobimo sode člene naše sinusne transformacije, za lihe pa prvega dobimo z uporabo dejstva $F_1 = -F_{-1}$ ostale pa iz le-tega rekurzivno.