Metoda končnih elementov: Poissonova enačba

Miha Čančula

19. april 2012

1 Uporabljena orodja

Za izračun lastnih vrednosti in lastnih vektorjev matrike sem uporabil knjižnico `ARPACK`, za prikaz rešitev `MathGL`, ostalo pa sem napisal v programskem jeziku `C`. Metoda za reševanje posplošenega sistema lastnih vrednosti uporablja LU razcep. Za ekstrapolacijo lastnih vrednosti pri $n \to \infty$ sem uporabil `Gnuplot`.

2 Končni elementi

Najprej sem nalogo rešile z metodo končnih elementov. Celoten postopek je zelo podoben kot pri prejšnji nalogi, razlika je le v samem reševanju matričnega sistema.

Iskanje lastnih vrednosti matrike je bolj zahtevna operacija kot reševanje sistema, zato sem se moral omejiti na manjše matrike. Matriki A in B redki (vsaka ima približno $7n$ elementov, matriki pa sta $n \times n$), ko pa problem z razcepom matrike B prevedemo na iskanje lastnih vrednosti matrike C, pa ima ta matrika že več elementov.

3 Galerkinov nastavek

Podoben postopek lahko izvedemo tudi, če rešitev namesto po funkcijah w_i, ki so različne od 0 le na majhnem prostoru, razvijemo po zveznih funkcijah

$$u = \sum_{m=1}^{\infty} \sum_{k=0}^{\infty} a_k^m g_k^m \tag{1}$$

$$g_k^m = r^{m+k}(1-r)\sin(m\varphi) \tag{2}$$

Če usmerimo os y v smeri ravnega roba polkroga, je $\varphi \in [0, \pi]$ in so pri robnih pogojih prve vrste smiselne kotne odvisnosti le sinusi. Ti so za različne m med seboj ortogonalni, zato je matrika A bločno diagonalna in razpade na podmatrike A_m. Isto velja tudi za masno matriko B. Dovolj je torej, če izračunamo koeficiente a_k^m za vsak m posebej.

Matrične elemente A in B izrazimo kot integrale funkcij g_k^m in njihovih gradientov, ki jih lahko izračunamo analitično.

$$
\begin{aligned}
\langle g_k^m, g_l^m \rangle &= \int_0^\pi \int_0^1 r^{2m+k+l}(1-r)^2 \sin^2(m\varphi) r \mathrm{d}r\mathrm{d}\varphi \\
&= \frac{\pi}{2} \int_0^1 \left(r^{2m+k+l+1} - 2r^{2m+k+l+2} + r^{2m+k+l+3} \right) \mathrm{d}r \\
&= \frac{\pi}{2} \left(\frac{1}{2m+k+l+2} - \frac{2}{2m+k+l+3} + \frac{1}{2m+k+l+4} \right) \qquad (3) \\
\nabla g_k^m &= \left(\frac{\partial}{\partial r}, \frac{1}{r}\frac{\partial}{\partial \varphi} \right) \left(r^{m+k} - r^{m+k+1} \right) \sin(m\varphi) \\
&= \left((m+k) r^{m+k-1} - (m+k+1) r^{m+k} \right) \sin(m\varphi), \\
&\quad m \left(r^{m+k-1} - r^{m+k} \right) \cos(m\varphi) \qquad (4) \\
\langle \nabla g_k^m, \nabla g_l^m \rangle &= \frac{\pi}{2} \int_0^1 \Big[\left((m+k) r^{m+k-1} - (m+k+1) r^{m+k} \right) \left((m+l) r^{m+l-1} - (m+l+1) r^{m+l} \right) + \\
&\qquad + m^2 \left(r^{m+k-1} - r^{m+k} \right) \left(r^{m+l-1} - r^{m+l} \right) \Big] r \mathrm{d}r = \\
&= \frac{\pi}{2} \int_0^1 r^{2m+k+l} \Big\{ \left[(m+k)(m+l) + m^2 \right] r^{-1} + \left[(m+k+1)(m+l+1) + m^2 \right] r - \\
&\qquad - \left[(m+k)(m+l+1) + (m+k+1)(m+l) + 2m^2 \right] \Big\} \mathrm{d}r \\
&= \frac{\pi}{2} \left(m + \frac{kl}{2m+k+l} - 2m - \frac{k+l}{2m+k+l+1} + m + \frac{k+l+kl}{2m+k+l+2} \right) \\
&= \frac{\pi}{2} \left(\frac{kl}{2m+k+l} - \frac{2kl+k+l}{2m+k+l+1} + \frac{(k+1)(l+1)}{2m+k+l+2} \right) \qquad (5)
\end{aligned}
$$

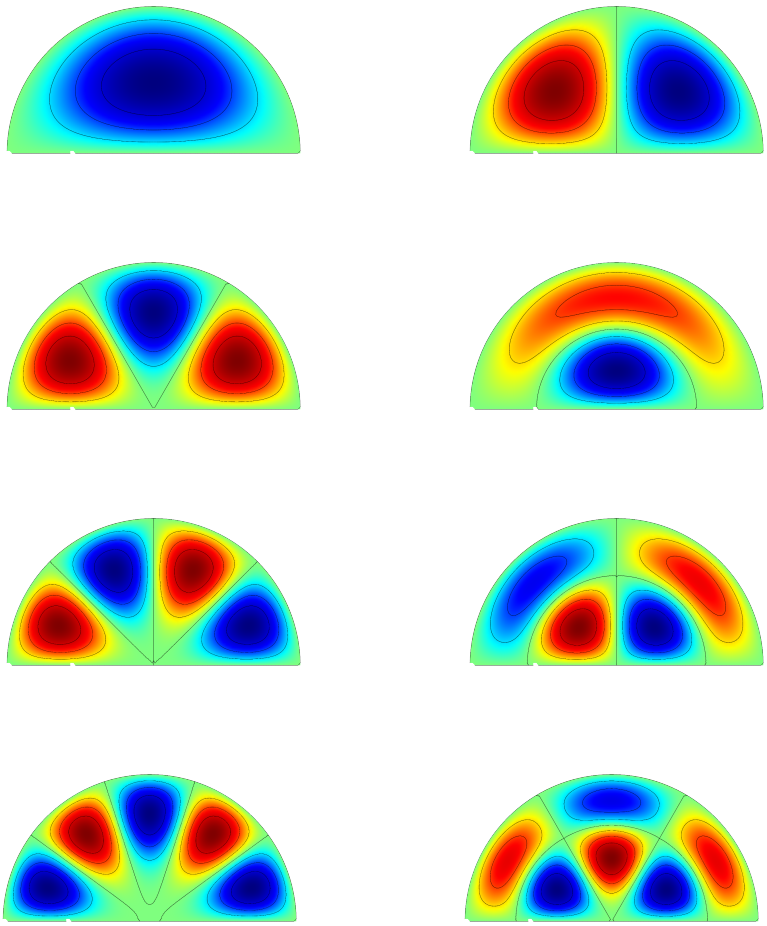
Lastne vrednosti in lastni vektorji se ne spremenijo, če matriki A in B pomnožimo s konstantnim faktorjem, zato sem pri računih izpustil množenje s $\pi/2$.

3.1 Reševanje

Matriki A in B sedaj nista več redki, pri velikih k pa funkcije g postajajo vse bolj linearno odvisne, zato smo omejeni na majhne k in s tem tudi majhne matrike. Zaradi majhnih dimenzij matrik ne potrebujemo posebej optimiziranega postopka za iskanje lastnih vrednosti in vektorjev, v mojem primeru sem uporabil `GSL`. Obe matriki sta še vedno simetrični, zato lahko uporabimo funkcijo za simetrične matrike, lastne vrednosti pa so realne.

Število testnih funkcij sem izbral tako, da je bila bil največji eksponent $m+k \leq 15$. Pri večjem številu sem imel težave z izračunom, saj matrika ni bila več pozitivno definitna.

4 Rezultati



Slika 1: Prvih 8 nihanj polkrožne opne, izračunanih po metodi končnih elementov

Po obeh metodah sem izračunal 8 najnižnih lastnih frekvence polkrožne opne. Rezultati ter primerjava z vrednosti po metodi relaksacije in tabeliranimi ničlami Besselovih funkcij so v tabeli 1.

m	n	Relaksacija	FEM		Galerkin	Tabeliran
		8192 točk	9401 točk	Ekstrapolacija	10 funkcij	
1	1	3.8403	3.8321	3.8316	3.8317	3.8317
2	1	5.1374	5.1365	5.1354	5.1356	5.1356
3	1	6.3800	6.3822	6.3797	6.3802	6.3802
1	2	7.0284	7.0180	7.0150	7.0156	7.0156
4	1	7.5864	7.5914	7.5876	7.5883	7.5883
2	2	8.4178	8.4214	8.4162	8.4172	8.4172
5	1	8.7673	8.7761	8.7704	8.7715	8.7715
3	2	9.7580	9.7681	9.7594	9.7610	9.7610

Tabela 1: Primerjava lastnih frekvenc, izračunanih po različnih metodah

S tabele vidimo, da je izmed vseh treh preizkušanih metoda uporaba nastavkov daleč najboljša. Tudi pri uporabi majhnega števila poskusnih funkcij se prvih 8 lastnih vrednosti ujema s pravimi

vrednostmi na vsaj 4 decimalke. Poleg najboljše natančnosti je ta metoda tudi mnogo hitrejša, saj ima matrika 100 elementov namesto nekaj milijonov.

Seveda za takšno izboljšavo potrebujemo nastavke, ki so dobra baza za iskane nihajne načine. V našem primeru smo privzeli celotno kotno odvisnost rešitve, približno pa smo poznali vsaj obliko radialnega dela. V poljubni geometriji takšnih predpostav ne moremo narediti, zato je težko uganiti prave nastavke.

Metodi relaksacije in končnih elementov se pri fiksnem številu točk izkažeta za podobno natančni. Lastne vrednosti so pri računu s približno 9000 točkami natančne na 1 do 2 decimalki. Tudi v hitrost se ne razlikujeta, saj v obeh primerih računamo lastne vrednosti in vektorje redke matrike. Prednost FEM pa je, da lastne vrednosti konvergirajo monotono, torej se z večanjem števila točk le manjšajo. Če to odvisnost ekstrapoliramo, dosežemo večjo natančnost (na 3 decimalke).

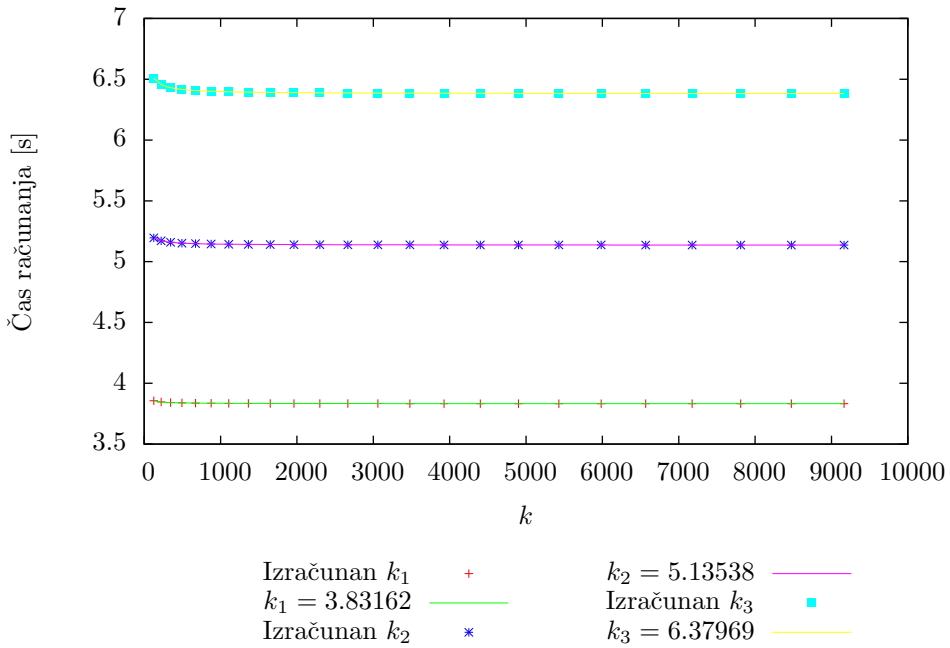
Odvisnosti i-te lastne frekvence od števila točk sem prilagodil funkcijo

$$k_i(n) = k_i + \alpha_i n^{-\beta_i} \tag{6}$$

Za prvih osem lastnih frekvence je ekstrapolirana vrednost manjša od prave, zato domnevam, da bi nastavek (6) lahko izboljšali in s tem povečali natančnost celotnega izračuna.

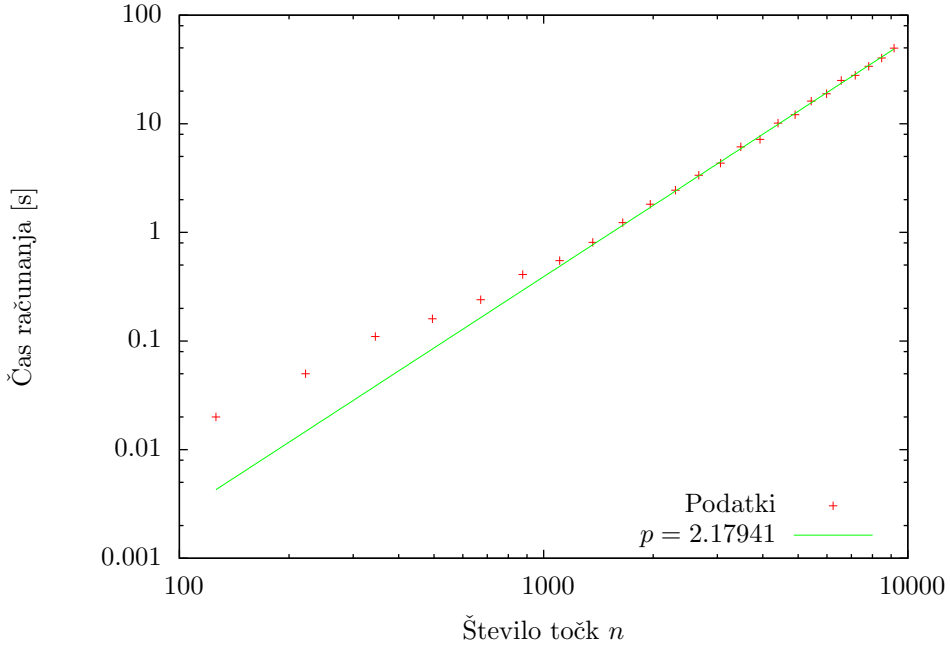
5 Hitrost in konvergenca

Pri metodi končnih elementov sem opazoval, kako sta trajanje in natančnost računanja odvisna od števila točk. Pri nastavku Galerkina to početje ni bilo smiselno, saj sem bil zaradi linearne odvisnosti funkcij omejen na majhne matrike (največ 10×10). Rezultati so na slikah 2 in 3.



Slika 2: Konvergenca prvih nekaj lastnih vrednostih pri večanju števila točk

Za razliko od reševanja Poissonove enačbe je iskanje lastnih nihanj reda $\mathcal{O}(n^2)$. Zaradi te odvisnosti mi je uspelo rešiti le sisteme z do 10000 točkami.



Slika 3: Odvisnost časa računanja lastnih frekvenc in nihanj od števila točk

Ker metoda izhaja iz variacijskega problema, sem preveril, da tudi izračunane lastne frekvence monotono konvergirajo k pravi vrednosti. Za razliko od računanja pretoka po cevi se frekvence približujejo pravi vrednosti od zgoraj.

Za prva tri lastna nihanja so vrednosti na sliki 2, v vseh treh primerih je β med 0.9 in 1. S podvojitvijo števila točk se čas računanja poveča za štirikrat, napaka lastnih frekvenc pa se le prepolovi.