Univerza *v Ljubljani*
Fakulteta za *matematiko in fiziko*

Reševanje PDE z metodo Galerkina

11. naloga pri Matematično-fizikalnem praktikumu

Avtor: Marko Urbanč (28191096)
Predavatelj: prof. dr. Borut Paul Kerševan

8.9.2023

Kazalo

1 Uvod

Če poznamo lastne funkcije diferencialnega operatorja za določeno geometrijo, se reševanje parcialnih diferencialnih enačb včasih lahko prevede na razvoj po lastnih funkcijah. V tem primeru se lahko diferencialni operator zapiše kot matrika in enačbo potem rešujemo kot sistem linearnih enačb. Tega lahko računamo kot vemo in znamo. Zdaj smo to počeli že parkrat.

V našem primeru bo fizikalna inspiracija Navier-Stokesova enačba, ki je pravzaprav drugi Newtonov zakon za tekočine. Vendar pa je ta enačba zelo zapletena in je še vedno odprt problem, ali sploh obstajajo rešitve v splošnem. Zato se bomo omejili na preprostejši primer, kjer privzamemo, da imamo enakomeren laminaren tok nestisljive tekočine v dolgi ravni cevi pod vplivom stalnega tlačnega gradienta p'. V tem primeru se Navier-Stokesova enačba poenostavi na Poissonovo enačbo

$$\nabla^2 \vec{v} = -\frac{p'}{\eta} \,, \tag{1}$$

kjer je $\vec{v}$ hitrost tekočine in η njena viskoznost. Enačbo rešujemo v notranjosti preseka cevi, medtem ko je ob stenah hitrost enaka nič. Za pretok velja Poiseuillov zakon

$$\Phi = \int_S v \, \mathrm{d}S = C \frac{p' S^2}{8\pi\eta} \,, \tag{2}$$

kjer je S presek cevi in C konstanta, ki je odvisna od oblike preseka. Konstanta znaša $C = 1$ za krožni presek. V našem primeru pa bomo določili konstanto C pa polkrožni presek. Uvedemo nove spremenljivke $\xi = r/R$ in $u = v\eta/(p'R^2)$ in nato se problem glasi

$$\Delta u(\xi,\, \phi) = -1, \qquad u(1,\, \phi) = u(\xi,\, 0) = u(\xi,\, \pi) = 0 \,, \tag{3}$$

$$C = 8\pi \iint \frac{u(\xi,\, \phi)\xi \, \mathrm{d}\xi \, \mathrm{d}\phi}{(\pi/2)^2} \,. \tag{4}$$

Da se izognemo računanju lastnih funkcij (v temu primeru Besselovih) in njihovih ničel, lahko zapišemo rešitev v obliki razvoja po neki poskuni bazi. V našem primeru bomo vzeli bazo

$$\psi_{nm}(\xi,\, \phi) = \xi^{2m+1}(1-\xi)^n \sin((2m+1)\phi) \,. \tag{5}$$

Z njo lahko zapišemo aproksimacijo rešitve kot linearno kombinacijo

$$\tilde{u}(\xi,\, \phi) = \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} a_{nm} \psi_{nm}(\xi,\, \phi) \,. \tag{6}$$

Za te funkcije niti ni nujno da so prava baza v smislu, da bi bile ortogonalne druga na drugo. Potrebno je da zadoščajo robnim pogojem tako, da jim bo avtomatično zadoščala tudi linearna kombinacija. Približna funkcija $\tilde{u}$ seveda ne zadosti Poissonovi enačbi za res, ampak ji preostane majhna napaka ε

$$\Delta \tilde{u}(\xi,\, \phi) = -1 + \varepsilon(\xi,\, \phi) \,. \tag{7}$$

Pri metodi Galerkina zahtevamo, da je napaka ortogonalna na vse poskusne funkcije ψ_{nm}, torej

$$\langle \psi_{nm},\, \varepsilon \rangle = 0 \qquad \forall n, m \, . \tag{8}$$

V splošnem bi lahko zahtevali še ortogonalnost napake na nek drug sistem utežnih funkcij. Metoda Galerkina je poseben primer takih metod (angl. *Methods of Weighted Residuals*), kjer je utežna funkcija kar poskusna funkcija sama. Omenjena izbira vodi do sistema enačb za koeficiente a_{nm}

$$A_{nm,n'm'} a_{n'm'} = b_{nm} \, . \tag{9}$$

Koeficiente b_{nm} dobimo iz skalarnega produkta

$$b_{nm} = \langle -1,\, \psi_{nm} \rangle \, , \tag{10}$$

ki se zaradi ortogonalnosti poskusnih funkcij poenostavi v

$$b_{nm} = -\frac{2}{2m+1} \mathrm{B}(2m+3, n+1) \, , \tag{11}$$

kjer je B Eulerjeva beta funkcija. Matrika A pa je definirana kot

$$A_{nm,n'm'} = \langle \nabla^2 \psi_{nm},\, \psi_{n'm'} \rangle \, , \tag{12}$$

kar se po upoštevanju ortogonalnosti poskusnih funkcij poenostavi v

$$A_{nm,n'm'} = -\frac{\pi}{2} \frac{nn'(3+4m)}{2+4m+n+n'} \mathrm{B}(n+n'-1, 3+4m)\, \delta_{mm'} \, . \tag{13}$$

Končno, se naša enačba za koeficient za pretok C glasi

$$C = -\frac{32}{\pi} \sum_{mn,m'n'} b_{nm} A^{-1}_{nm,n'm'} b_{m'n'} \, . \tag{14}$$

2 Naloga

Naloga od nas zahteva, da rešimo Poissonovo enačbo (3) in izračunamo koeficient za pretok C v primeru polkrožnega preseka cevi. Naj si tudi pogledamo kako je odvisna natančnost rešitve od števila členov v indeksih m in n.

3 Opis reševanja

Za reševanje sem prižgal svoj trusty IBM PC/XT 5160 (*beri:* v resnici sem uporabil svoj domači računalnik) in preko božjega čudeža uspel uporabiti Python 3.11.0 kljub temu, da je bil ta napisan šele leta 2023. Uporabil sem standardni nabor paketov za znanstveno računanje, torej `numpy`, `scipy` in `matplotlib`.

Naloge sem se lotil kot zadnjih dveh nalog, kjer smo tudi reševali parcialne diferencialne enačbe, le na druge načine. Ne vem če je res smiselno ponavljati iste razloge zakaj uberem takšno metodo reševanja. Več naj bi bilo eventually dostopno na spletu na moji strani. Napisal sem razred `GalerkinObject` (da ni stalno Solver) v katerem je vsebovano vse kar potrebujemo za reševanje. Ob kreaciji novega `GalerkinObjekt`-a uporabnik poda, kolikšno naj bo število členov v indeksih m in n. Ostalo pa se ob klicanju metode `solve()` izračuna samodejno. Imel sem izbiro, da bi lahko uporabil za matrične sisteme kar `numpy`-jeve funkcije, ampak se mi je zdelo dovolj preprosto, da sem kar sam napisal funkcije za reševanje sistema.

4 Rezultati

Glede na to, kako se mi, zaradi lastnih napak, mudi z oddajo se mi zdi najbolj smiselno, da gremo kar takoj na rezultate.

4.1 Bazne funkcije

Najprej si poglejmo kako izgledajo bazne funkcije.

5 Komentarji in izboljšave

Literatura