Iterativne numerične metode v linearni algebri 2017/2018

1. domača naloga

Rešitve oddajte do **petka, 22. decembra 2017**, do 23. ure. Navodila:

- Programe in poročilo stisnite v ZIP datoteko z imenom `ime-priimek-vpisna-1.zip`, ki jo oddate preko spletne učilnice (`http://ucilnica.fmf.uni-lj.si`).
- V poročilo ni potrebno stisniti testnih datotek z matrikami, ki so na voljo na spletni učilnici. Če pa imate kakšne druge testne podatke, jih priložite.
- V poročilu za vsako nalogo opišite postopek reševanja, zapišite rešitev in komentirajte rezultat. Če poročilo skenirate, mora biti oddano v pdf obliki.
- Rešitvi priložite izjavo, da ste naloge reševali samostojno.
- Navedite, s katerim programom ste reševali naloge, namesto Matlaba lahko uporabite Octave. Programi naj bodo smiselno poimenovani in razporejeni v mapah, ki so poimenovane nal1, nal2, ... K vsaki nalogi spada glavna skripta, ki izpiše rešitve naloge (nal1.m, ..., nal4.m). Preverite, da se skripte res izvedejo v ukazni vrstici (npr. klic nal1 se mora izvesti brez napak), v nasprotnem boste izgubili polovico točk pri konkretni nalogi.
- Z vprašanji o nalogah ali Matlabu se lahko obrnete name. Če menite, da je vprašanje zanimivo tudi za ostale, uporabite forum. Vprašanja so dobrodošla.

Vsaka naloga je vredna 5%, ena naloga je za bonus, skupno lahko torej zberete do 20%.

1. Brezdimenzijski geometrijski faktor F pokončne prizme, katere presek je območje $\Omega \subset \mathbb{R}^2$ z robom $\partial\Omega$, je

   $$F = \frac{4\pi}{S^2} \int_\Omega \psi dS,$$

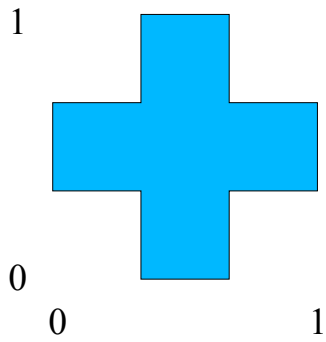
   kjer je ψ (torzijska funkcija preseka) rešitev parcialne diferencialne enačbe

   $$-\Delta\psi = 2$$

   pri robnem pogoju

   $$\psi|_{\partial\Omega} = 0.$$

   Na 6 decimalk oziroma čim bolj natančno izračunajte F za prizmo s presekom, ki ga dobite, če enotski kvadrat razdelite na 3×3 enake dele in odstranite vogalne dele.



Diferencialno enačbo aproksimirajte s pettočkovno shemo in rešite dobljeni razpršeni sistem. Dobljene točke iz mreže vstavite v dvodimenzionalno trapezno kvadraturno pravilo. Kako velik sistem potrebujete, da pridete (oz. da bi prišli) do 6 točnih decimalk? Sestavite program v Matlabu, ki reši nalogo.

Nasveti:

- Bistveni del te naloge je konstrukcija sistema, saj lahko v Matlabu dobljene sisteme (dokler niso preveliki) učinkovito rešite kar direktno z `x=A\b`.
- Uporaba trapezne metode in dejstvo, da so napake, ki jih dobimo z diferenčno metodo, reda $\mathcal{O}(h^2)$, nam omogoča, da iz rezultatov, ki jih dobite za razmike $h, h/2, h/4, \ldots$ z Rombergovo ekstrapolacijo dobimo še boljše približke. Pri tem pazite, da mrežo sestavite tako, da bo večja matrika res ustrezala razpolovljenem razmiku h.

2. Na predavanjih smo obravnavali metodo D-Lanczos, ki jo najdete tudi v knjigi [1] na strani 156. V originalnem algoritmu se izvaja LU algoritem brez pivotiranja, zaradi česar se lahko algoritem kritično zaustavi. Možna rešitev je uporaba LU z delnim pivotiranjem.

   Razvijte varianto D-Lanczos z delnim pivotiranjem in jo implementirajte v Matlabu. Na manjših primerih najprej preverite, da pivotiranje deluje v redu. Metodo potem uporabite na primerih `DL1`, `DL2` in `DL3` iz spletne učilnice s toleranco 10^{-10} in maksimalnim številom korakov 5000. Narišite grafe konvergence.

   Nasveti:

   - Pazite, da v spominu hranite le tiste vektorje, ki jih potrebujete.
   - Pri eksaktnem računanju bi se morali približki ujemati s približki, ki jih dobimo z metodo FOM. Tako lahko z obema metodama naredite le nekaj korakov in preverite, če se rešitvi ujemata (do vpliva zaokrožitvenih napak), da testirate vašo metodo.
   - Za test uporabite tako testno matriko (lahko majhno), pri kateri vidite, da je prišlo do pivotiranja. Kljub izvedenemu pivotiranju se morajo približki ujemati s približki iz FOM.
   - Če ne gre, poskusite najprej ločeno sestaviti algoritem, ki z LU razcepom z delnim pivotiranjem reši dani sistem s tridiagonalno matriko.

- Primer DL4 je sestavljen tako, da pri LU razcepu brez pivotiranja pivotni element na mestu $(3,3)$ pride 0, torej D-Lanczos ne deluje. Pazite na to, da v tem primeru ne obstaja približek iz $\mathcal{K}_3(A,b)$, ki bi zadoščal Ritzovemu pogoju, saj je matrika T_3 singularna. Algoritem mora znati preskočiti tak problem.

3. Iz zbirke SSMC naložite naslednje matrike (na voljo so tudi v spletni učilnici): `c-63` (1224), `gridgena` (1311) in `rajat25` (1374). Za vsako izmed matrik poskusite rešiti linearni sistem $Ax = b$ z iterativnimi metodami, ki so na voljo v Matlabu: `gmres, minres, pcg, bicg, qmr, symmlq, bicgstab`. Če desna stran ni podana zraven problema, vzemite $b = Ae$, kjer je e vektor samih enic. Za začetni približek izberite $x_0 = 0$. Na SSMC si oglejte, kakšne so lastnosti teh treh matrik, da ne boste npr. uporabljali metode konjugiranih gradientov za matriko, ki ni s.p.d.

   - Ugotovite, katere metode so primernejše za katero matriko in to obrazložite.
   - Pri metodah, kjer je možno uporabiti ponoven zagon (kot npr. GMRES), ugotovite optimalno izbiro maksimalne velikosti podprostorov in raziščite vpliv izbire velikosti na konvergenco.
   - Če je konvergenca počasna, poskusite poiskati ustrezno predpogojevanje, ki izboljša konvergenco. Poskusite npr. z nepopolnim LU razcepom ali nepopolnim razcepom Choleskega.
   - Ali lahko z iterativnimi metodami dobite še uporaben približek (npr. da bo izračunani x točen na vsaj tri decimalke) hitreje kot Matlab z direktno metodo `x=A\b`.

4. Naj bodo $c_1c_2c_3c_4$ zadnje 4 števke vaše vpisne številke in $V = 4 * c_1c_2 + c_3c_4$.

   Metodo konjugiranih gradientov lahko uporabimo tudi za reševanje nedoločenega sistema $Ax = b$. Za reševanje tega sistem lahko modificirate kar Matlabovo funkcijo `pcg`.

   Algoritem preizkusite na realnem problemu, ki nastane pri računanju približka Mahalanobis razdalje pri ocenjevanju kvalitete rudarjenja podatkov. Radi bi izračunali

   $$X_{\text{test}}^T (X_{\text{train}} X_{\text{train}}^T)^\dagger X_{\text{test}}.$$

   Obe matriki X imata veliko večje število vrstic kot stolpcev in sta razpršeni. Ker je sistem z matriko $X_{\text{train}}X_{\text{train}}^T$ poddoločen, namesto psevdoinverza uporabimo regularizacijo. Tako namesto z $(X_{\text{train}}X_{\text{train}}^T)^\dagger$ računamo z regularizirano matriko $A = (1-\alpha)X_{\text{train}}X_{\text{train}}^T + \alpha I$. Množenje z matriko A je potrebno implementirati implicitno, saj je matrika $X_{\text{train}}X_{\text{train}}^T$ prevelika, da bi jo shranili v pomnilnik.

   Narišite graf števila korakov potrebnih za konvergenco metode konjugiranih gradientov za sistem $Ax = b$ v odvisnosti od $\alpha \in [0,1]$, kjer je b V-ti stolpec matrike X_{test}. Poiščite najmanjši α pri katerem dobite zadovoljivo konvergenco in nato za ta α izračunajte celoten produkt $X_{\text{test}}^T A^{-1} X_{\text{test}}$. Pazite. Ko povečujemo α se rešitev oddaljuje od prave, možen kriterij za kvaliteto je, da izračunamo ostanek za originalno matriko in vektor izračunan s pomočjo regularizacije.

Matriki naložite v Matlab z ukazom `load mahalanobis`, kjer datoteko `mahalanobis` dobite na spletni učilnici.

Veliko uspeha pri reševanju!

Literatura

[1] Y.Saad, Iterative methods for sparse linear systems. Na voljo na `http://www-users.cs.umn.edu/~saad/PS/all_pdf.zip`

[2] T. Davis, Suite Sparse Matrix Collection. Na voljo na `https://sparse.tamu.edu/`