Міністерство освіти та науки України
Київський національний університет імені Тараса Шевченка
Факультет комп'ютерних наук та кібернетики
Кафедра обчислювальної математики

Звіт до лабораторної роботи №3 на тему: "Проблема знаходження власних чисел матриці'

Виконав студент групи ОМ-3
Скибицький Нікіта

Київ, 2018

1 Постановка задачі

Для матриці

$$a_{i,j} = \begin{cases} \frac{i+j-1}{2n}, & i \neq j \\ n + 10 + \frac{i+j-1}{2n}, & i = j \end{cases}$$

порядку $n = 5, 6, \ldots$ знайти

1. $\lambda_{\max} = \max\limits_i |\lambda_i(A)|$;
2. $\lambda_{\min} = \min\limits_i \lambda_i(A)$;
3. $\hat{\lambda}_{\max} = \min\limits_i |\lambda_i(A)|$.

методом скалярних добутків та всі власні значення – методом обертання Якобі.

2 Теоретична частина

2.1 Степеневий метод

1. *Знаходження* $\lambda_{\max} : |\lambda_1| \equiv \lambda_{\max} > |\lambda_2| \geq |\lambda_3| \geq \ldots$

   Нехай $\vec{x}^{(0)}$ – заданий вектор, будемо послідовно обчислювати вектори

   $$\vec{x}^{(k+1)} = A\vec{x}^{(k)}, \quad k = 0, 1, \ldots \tag{2.1}$$

   Тоді $\vec{x}^{(k)} = A^k\vec{x}^{(0)}$. Нехай $\{\vec{e}_i\}_{i=1}^n$ – система власних векторів. Представимо $\vec{x}^{(0)}$ у вигляді:

   $$\vec{x}^{(0)} = \sum_{i=1}^{n} c_i\vec{e}_i.$$

   Оскільки $A\vec{e}_i = \lambda_i\vec{e}_i$, то $\vec{x}^{(k)} = \sum\limits_{i=1}^{n} c_i\lambda_i^k\vec{e}_i$. При великих k: $\vec{x}^{(k)} \approx c_1\lambda_1^k\vec{e}_1$. Тому

   $$\mu_1^{(k)} = \frac{\vec{x}_m^{(k+1)}}{\vec{x}_m^{(k)}} = \lambda_1 + O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right).$$

   Значить $\mu_1^{(k)} \xrightarrow[k\to\infty]{} \lambda_1$.

   Якщо матриця $A = A^T$ симетрична, то існує ортонормована система векторів $(\vec{e}_i, \vec{e}_j) = \delta_{i,j}$. Тому

   $$\mu_1^{(k)} = \frac{(\vec{x}^{(k+1)}, \vec{x}^{(k)})}{(\vec{x}^{(k)}, \vec{x}^{(k)})} = \frac{\left(\sum\limits_{i=1}^{n} c_i\lambda_i^(k+1)\vec{e}_i, \sum\limits_{j=1}^{n} c_j\lambda_j^k\vec{e}_j\right)}{\left(\sum\limits_{i=1}^{n} c_i\lambda_i^k\vec{e}_i, \sum\limits_{j=1}^{n} c_j\lambda_j^k\vec{e}_j\right)} = \frac{\sum\limits_i c_i^2\lambda_i^{2k+1}}{\sum\limits_i c_i^2\lambda_i^{2k}} =$$

$$= \frac{c_1^2\lambda_1^{2k+1} + c_2^2\lambda_2^{2k+1} + \dots}{c_1^2\lambda_1^{2k} + c_2^2\lambda_2^{2k} + \dots} = \lambda_1 + O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^{2k}\right) \xrightarrow[k\to\infty]{} \lambda_1.$$

Це означає збіжність до максимального за модулем власного значення з квадратичною швидкістю.

Якщо $\lambda_1 > 1$, то при проведенні ітерацій відбувається зріст компонент вектора $\vec{x}^{(k)}$, що приводить до "переповнення" (overflow). Якщо ж $\lambda_1 < 1$, то це приводить до зменшення компонент (underflow). Позбутися негативу такого явища можна нормуючи вектори $\vec{x}^{(k)}$ на кожній ітерації.

Алгоритм степеневого методу знаходження максимального за модулем власного значення з точністю ε виглядає так:

(а) $\vec{x}^{(0)} \to \vec{e}_0 = \frac{\vec{x}^{(0)}}{\|\vec{x}^{(0)}\|}$;

(б) $\vec{x}^{(k+1)} = A\vec{x}^{(k)}$, $\mu_1^{(k)} = (\vec{x}^{(k+1)}, \vec{e}^{(k)})$, $\vec{e}^{(k+1)} = \frac{\vec{x}^{(k+1)}}{\|\vec{x}^{(k+1)}\|}$, $k = 0, 1, \dots$;

(в) $|\mu_1^{(k+1)} - \mu_1^{(k)}| \geq \varepsilon$ `goto` (б);

(г) $\lambda_1 \approx \mu_1^{(k+1)}$.

За цим алгоритмом для симетричної матриці $A^T = A$ швидкість прямування $\mu_1^{(k)}$ до $\lambda_{\max}$ – квадратична.

2. *Знаходження* $\lambda_2 : |\lambda_1| > |\lambda_2| > |\lambda_3| \geq \dots$. Нехай λ_1, $\vec{e}_1$ відомі.

Якщо $|\lambda_1| > |\lambda_2| > |\lambda_3| \geq \dots$, то

$$\mu_2^{(k)} = \frac{\vec{x}_m^{(k+1)} - \lambda_1 \vec{x}_m^{(k)}}{\vec{x}_m^{(k)} - \lambda_1 \vec{x}_m^{(k-1)}} \xrightarrow[k\to\infty]{} \lambda_2, \quad \text{де } \vec{x}^{(k+1)} = A\vec{x}^{(k)}.$$

$x_m^{(k)}$ – m-та компонента $\vec{x}^{(k)}$.

Є алгоритм обчислення λ_2, $\vec{e}_2$, використовуючи нормування векторів та скалярні добутки для обчислення $\mu_2^{(k)}$).

3. *Знаходження мінімального власного числа* $\lambda_{\min}(A) = \min_i |\lambda_i(A)|$.

Припустимо , що $\lambda_i(A) > 0$ та відоме $\lambda_{\max}$. Розглянемо матрицю $B = \lambda_{\max}E - A$. Маємо

$$\forall i : \lambda_i(B) = \lambda_{\max} - \lambda_i(A).$$

Тому $\max\limits_i \lambda_i(B) = \lambda_{\max} - \min\limits_i \lambda_i(A)$. Звідси $\lambda_{\min}(A) = \lambda_{\max}(A) - \lambda_{\max}(B)$.

Якщо $\exists i : \lambda_i(A) < 0$, то будуємо матрицю $\overline{A} = \sigma E + A$, $\sigma > 0$, $\overline{A} > 0$ і для неї попередній розгляд дає необхідний результат. Замість $\lambda_{\max}$ в матриці B можна використовувати $\|A\|$.

Ще один спосіб обчислення мінімального власного значення полягає в використання обернених ітерацій:

$$A\vec{x}^{(k+1)} = \vec{x}^k, \quad k = 0, 1, \dots \tag{2.2}$$

Але цей метод вимагає більшої кількості арифметичних операцій: складність методу на основі формули (2.1) $Q = O(n^2)$, а на основі (2.2) – $Q = O(n^3)$, оскільки треба розв'язувати СЛАР, але збігається метод (2.2) швидче.

2.2 Ітераційний метод обертання

Це метод розв'язання повної проблеми власних значень для симетричних матриць $A^T = A$. Існує матриця U, що приводить матрицю A до діагонального виду:

$$A = U\Lambda U^T, \tag{2.3}$$

де Λ – діагональна матриця, по діагоналі якої стоять власні значення λ_i; U – унітарна матриця, тобто: $U^{-1} = U^T$.

З (2.3) маємо

$$\Lambda = U^T\Lambda U, \tag{2.4}$$

Нехай $\exists\tilde{U}$ – матриця, така що $\tilde{\Lambda} = \tilde{U}^T A\tilde{U}$ і $\tilde{\Lambda} = (\tilde{\lambda}_{i,j})_{i,j=1}^n$, $|\tilde{\lambda}_{i,j}| < \delta \ll 1$, $i \neq j$.

Тоді діагональні елементи мало відрізняються від власних значень

$$|\tilde{\lambda}_{i,i} - \lambda_i(A)| < \varepsilon = \varepsilon(\delta).$$

Введемо $t(A) = \sum\limits_{\substack{i,j=1\\ i\neq j}}^{n} a_{i,j}^2$. З малості величини $t(A)$ витікає, що діагональні елементи малі. По $A = A_0$ за допомогою матриць обертання що повертають систему векторів на кут φ, побудуємо послідовність $\{A_k\}$ таку, що $A_k \to \Lambda$ при $k \to \infty$.

Матриця обертання U_k є унітарною: $U_k^{-1} = U_k^T$.

Послідовно будуємо:

$$A_{k+1} = U_K^T A_k U_k, \tag{2.5}$$

Процес (2.5) називається *монотонним*, якщо: $t(A_{k+1}) < t(A_k)$.
Для матриці (2.5) виконується:

$$a_{i,j}^{(k+1)} = a_{i,j}^{(k)}\cos 2\varphi + \frac{1}{2}(a_{j,j}^{(k)} - a_{i,i}^{(k)})\sin 2\varphi, \tag{2.6}$$

А також $t(A_{k+1}) = t(A_k) - 2(a_{i,j}^{(k)})^2$, якщо вибирати φ з умови $a_{i,j}^{(k+1)} = 0$.

Звідси $\varphi = \varphi_k = \frac{1}{2}\arctan(p^{(k)})$, $p^{(k)} = \frac{2a_{i,j}^{(k)}}{a_{i,i}^{(k)} - a_{j,j}^{(k)}}$, де $|a_{i,j}^{(k)}| = \max\limits_{\substack{m,l\\ m\neq l}} |a_{m,l}^{(k)}|$. Тоді $t(A_k) \to 0$, $\to \infty$. Чим більше n тим більше ітерацій необхідно для зведення A до Λ.

3 Практична частина

Покажемо наочні результати для значення $n = 5$ та для заданої точності $\varepsilon = 10^{-6}$, хоча запрограмований алгоритм дає змогу отримати результати $\forall n \in \mathbb{N}$:

$$A = \begin{pmatrix} 15.1 & 0.2 & 0.3 & 0.4 & 0.5 \\ 0.2 & 15.3 & 0.4 & 0.5 & 0.6 \\ 0.3 & 0.4 & 15.5 & 0.6 & 0.7 \\ 0.4 & 0.5 & 0.6 & 15.7 & 0.8 \\ 0.5 & 0.6 & 0.7 & 0.8 & 15.9 \end{pmatrix}.$$

3.1 Метод скалярних добутків

Почнемо з $\vec{x}^{(0)} = (1, 0, 0, 0, 0)$ щоб довго не думати.

1. В процесі розв'язання задачі, були отримані відповідні результати для $\lambda_{\max} = \max\limits_i |\lambda_i(A)|$ та $\lambda_{\min} = \min\limits_i |\lambda_i(A)|$:

$$\lambda_{\max} = 17.686140661634397$$

2. Для знаходження $\lambda_{\min}$ була використана допоміжна матриця, $B = \lambda_{\max} \cdot E - A$:

$$B = \begin{pmatrix} 2.586140 & -0.2 & -0.3 & -0.4 & -0.5 \\ -0.2 & 2.486140 & -0.4 & -0.5 & -0.6 \\ -0.3 & -0.4 & 2.386140 & -0.6 & -0.7 \\ -0.4 & -0.5 & -0.6 & 2.286140 & -0.8 \\ -0.5 & -0.6 & -0.7 & -0.8 & 2.186140 \end{pmatrix}.$$

   Для якої знаходимо $\lambda_{\min}(A) = \lambda_{\max}(A) - \lambda_{\max}(B)$:

$$\lambda_{\min} = 17.686140661634397 - 2.8722810782394337 = 14.813859583394963.$$

3. Для знаходження $\hat{\lambda}_{\min}$ була використана допоміжна матриця $C = E - \frac{A^2}{\lambda^2_{\max}}$:

$$C = \begin{pmatrix} -11.92254791 & -0.3788277 & -0.55693326 & -0.73503882 & -0.91314438 \\ -0.3788277 & -12.2815861 & -0.74069297 & -0.9216256 & -1.10255823 \\ -0.55693326 & -0.74069297 & -12.64627844 & -1.10821238 & -1.29197208 \\ -0.73503882 & -0.9216256 & -1.10821238 & -13.01662492 & -1.48138593 \\ -0.91314438 & -1.10255823 & -1.29197208 & -1.48138593 & -13.39262555 \end{pmatrix}$$

$$\hat{\lambda}_{\min} = \min_i |\lambda_i(A)| = \sqrt{\lambda^2_{\max}(A)(1 - \lambda_{\max}(C))} =$$
$$= \sqrt{17.686140661634397^2(1 - 0.2984311005453412))} = 14.813859583394963.$$

3.2 Метод обертання Якобі

Враховуючи теоретичний арсенал та застосовуючи заданий алгоритм отримаємо:

$$\lambda(A) = (14.813859, 14.999999, 14.999999, 14.999999, 17.686140)$$

Ітерації виконуємо допоки не буде виконана умова зупинки:

$$t(A_k) = \sum_{\substack{i,j=1 \\ i \neq j}}^{n} a^2_{i,j} < \varepsilon, \quad A_{k+1} = U_k^T \cdot A_k \cdot U_k$$

На останній ітерації: $t(A_N) = 1.1086685175654626 \cdot 10^{-7}$. Кількість ітерацій $N = 11$