COL726 Assignment 4

2–16 April, 2021

Note: All answers should be accompanied by a rigorous justification, unless the question explicitly states that a justification is not necessary.

1. Consider a nonsymmetric real matrix $\mathbf{A} \in \mathbb{R}^{m\times m}$ and a vector $\mathbf{v} \in \mathbb{R}^m$. For a small perturbation $\delta\mathbf{v} \in \mathbb{R}^m$, find the change in Rayleigh quotient $\delta r = r(\mathbf{v} + \delta\mathbf{v}) - r(\mathbf{v})$ to first order in $\delta\mathbf{v}$, i.e. ignoring higher-order terms of size $O(\|\delta\mathbf{v}\|^2)$. When is $\delta r = 0$, to first order in $\delta\mathbf{v}$?

2. Let $\mathbf{A} \in \mathbb{R}^{m\times m}$ be a real symmetric matrix.

   (a) Give an iterative algorithm for finding the smallest n eigenvalues of $\mathbf{A}$ and their corresponding eigenvectors. Your algorithm should take only $O(n^2m)$ flops per iteration, though it may do $O(m^3)$ extra work before starting the iterations.

   (b) Implement your algorithm as a Python function `(lambda, Qhat) = eigs(A, n)` which returns a vector of eigenvalues $\boldsymbol{\lambda} = \begin{bmatrix}\lambda_{m-n+1} & \cdots & \lambda_m\end{bmatrix}^T$ and a matrix of eigenvectors $\hat{\mathbf{Q}} = \begin{bmatrix}\mathbf{q}_{m-n+1} & \cdots & \mathbf{q}_m\end{bmatrix}$. Terminate when $\|\mathbf{A}\mathbf{q}_j - \lambda_j\mathbf{q}_j\|_2 < 10^{-6}$ for all computed eigenpairs, or after the 50th iteration.

3. Assume $\mathbf{A} \in \mathbb{R}^{m\times m}$ is a real upper Hessenberg matrix. Consider the QR algorithm with shifts:

   $\mathbf{A}^{(0)} = \mathbf{A}$
   for $k = 1, 2, \ldots$
   Choose a shift $\mu^{(k)}$
   $\mathbf{Q}^{(k)}\mathbf{R}^{(k)} = \mathbf{A}^{(k-1)} - \mu^{(k)}\mathbf{I}$
   $\mathbf{A}^{(k)} = \mathbf{R}^{(k)}\mathbf{Q}^{(k)} + \mu^{(k)}\mathbf{I}$

   (a) Show that $\mathbf{A}^{(k)}$ is upper Hessenberg and has the same eigenvalues as $\mathbf{A}$.

   (b) Show that if $\mu^{(k)}$ is an eigenvalue of $\mathbf{A}^{(k-1)}$, then $\mathbf{A}^{(k)}$ has at least one zero on its first subdiagonal. In other words, it is of the form $\mathbf{A}^{(k)} = \begin{bmatrix}\mathbf{A}_{11} & \mathbf{A}_{12} \\ & \mathbf{A}_{22}\end{bmatrix}$ where $\mathbf{A}_{11}$ and $\mathbf{A}_{22}$ are upper Hessenberg.

4. Let $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^n$ be a function with fixed point $\mathbf{x}^*$ and Jacobian $\mathbf{G}$. Assuming $\mathbf{G}(\mathbf{x})$ is diagonalizable for all $\mathbf{x}$, prove the equivalence of the following statements: (i) the fixed-point iteration converges to $\mathbf{x}^*$ when started sufficiently close to it, (ii) $\rho(\mathbf{G}(\mathbf{x}^*)) < 1$, (iii) there exists an induced norm in which $\|\mathbf{G}(\mathbf{x}^*)\| < 1$.

5. The Jacobian update in Broyden's (first) method can be expressed as $\mathbf{B} \leftarrow \mathbf{B} + \Delta\mathbf{B}$ where $\Delta\mathbf{B}$ has rank 1. There are many rank-1 matrices $\Delta\mathbf{B} = \mathbf{u}\mathbf{v}^T$ such that $(\mathbf{B} + \Delta\mathbf{B})\mathbf{s} = \mathbf{y}$; show that the one used by Broyden has the smallest Frobenius norm among all of them.

6. A complex polynomial $p(z) = a_n z^n + \cdots + a_1 z + a_0$ can be interpreted as a function $\mathbf{f} : \mathbb{R}^2 \to \mathbb{R}^2$ by defining $\mathbf{f}(x_1, x_2) = (y_1, y_2)$ if and only if $p(x_1 + ix_2) = y_1 + iy_2$. For simplicity, here we will assume the coefficients $a_0, \ldots, a_n$ to be real although the roots may be complex.

   (a) Give a formula for the Jacobian of $\mathbf{f}$.

   (b) Implement a Python function `xstar = newton(a, x0)` which performs Newton's method to find a root of $\mathbf{f}$, given the coefficients of the polynomial in the vector $\mathbf{a} = [a_0, \ldots, a_n]$, and the initial guess $\mathbf{x}^{(0)} \in \mathbb{R}^2$. Terminate when $\|\mathbf{f}(\mathbf{x}^{(k)})\|_2 < 10^{-6}$ or $k \geq 50$.

   (c) Choose a polynomial with degree ≥ 3 and some complex roots (e.g. $p(z) = z^3 - z - 1$). Which initial guesses converge to which roots? To find out, take each point in a 2D grid (e.g. $x_1 = -1.00, -0.95, -0.90, \ldots, 2.00$ and $x_2 = -1.00, -0.95, -0.90, \ldots, 1.00$) as an initial guess and record which root Newton's method converges to. Visualize this data as a 2D plot (e.g. using `matplotlib.pyplot.pcolormesh`), in which points that converge to the same root have the same colour. You should find that the resulting regions have a complicated fractal structure; explain why informally.

Collaboration policy: Refer to the policy on the course webpage.

If you collaborated with others to solve any question(s) of this assignment, give their names in your submission. If you found part of a solution using some online resource, give its URL.

Submission: Submit a PDF of your answers for all questions to Gradescope. Submit the code for Questions 2 and 6 to Moodle. Both submissions must be uploaded before the assignment deadline.

Code submissions should contain a single .py file which contains all the necessary functions. Functions are permitted but not required to produce any side-effects like printing out values or drawing plots. Any results you are asked to show should go in the PDF.