MATH 449, HOMEWORK 5

DUE OCTOBER 27, 2017

Part I. Theory

Problem 1. Prove the following norm inequalities.

a. For all $v \in \mathbb{R}^n$, $\|v\|_2^2 \leq \|v\|_1 \|v\|_\infty$.

b. For any norm $\|\cdot\|$ on $\mathbb{R}^n$, if λ is an eigenvalue of $A \in \mathbb{R}^{n \times n}$, then $\|A\| \geq |\lambda|$ in the induced norm.

Problem 2. Consider the function $f \colon \mathbb{R}^2 \to \mathbb{R}^2$ defined by

$$f(x_1, x_2) = \begin{pmatrix} x_1^3 - 3x_1 x_2^2 - 1 \\ 3x_1^2 x_2 - x_2^3 \end{pmatrix}.$$

Show that $f(x_1, x_2) = 0$ has three solutions: $(1, 0)$, $(-\frac{1}{2}, \frac{\sqrt{3}}{2})$, and $(-\frac{1}{2}, -\frac{\sqrt{3}}{2})$. Hint: Start by solving the second equation for x_2 in terms of x_1, and then substitute this into the first equation.

Remark. The function in Problem 2 can be identified with the complex-valued function $z \mapsto z^3 - 1$, whose roots are the three complex cube roots of 1 (or "roots of unity"). Indeed, if $z = x_1 + ix_2$, then we have

$$\begin{aligned} (x_1 + ix_2)^3 - 1 &= x_1^3 + 3x_1^2(ix_2) + 3x_1(ix_2)^2 + (ix_2)^3 - 1 \\ &= x_1^3 + i3x_1^2 x_2 - 3x_1 x_2^2 - ix_2^3 - 1 \\ &= (x_1^3 - 3x_1 x_2^2 - 1) + i(3x_1^2 x_2 - x_2^3) \end{aligned}$$

whose real and imaginary parts are precisely the two components of f.

Part II. Programming

Instructions. For the programming portion of this assignment, you will be running and modifying the code in the provided file `hw5.py`. Hand in a printed copy of the modified file, as well as a printout of the IPython terminal session(s) containing the commands and output you used to get your answers.

Problem 3.

a. Write a function `hilbert(n)` which returns the $n \times n$ Hilbert matrix,

$$\begin{pmatrix} 1 & 1/2 & \cdots & 1/n \\ 1/2 & 1/3 & \cdots & 1/(n+1) \\ \vdots & \vdots & \ddots & \vdots \\ 1/n & 1/(n+1) & \cdots & 1/(2n-1) \end{pmatrix}.$$

(Hint: You may find it helpful to notice that `H[i,j] = 1/(i+j+1)`.) Print the output of `hilbert(5)`.

b. Let A be the 20×20 Hilbert matrix, $x = (1, \ldots, 20)$, and $b = Ax$. Find δx by comparing x with the numerical solution `solve(A,b)` (which uses NumPy's built-in solver), and then find δb by comparing $A(x + \delta x)$ with b. Using either one of the functions `norm` or `l2norm` for $\|\cdot\|_2$, compute and print the values of $\|\delta b\|_2/\|b\|_2$ and $\|\delta x\|_2/\|x\|_2$.

Problem 4. Newton's method does not always converge to the closest root to the starting point—in fact, the behavior can be very complex. In $\mathbb{R}^2$, we can obtain fractal patterns called "Newton fractals" by coloring in the regions that converge to each root. The function `newtonFractal` in `hw5.py` does exactly this.

a. The function f from Problem 2 is already coded in `hw5.py` as `f(x)`. Create a function `Jf(x)` which returns the Jacobian matrix $J_f(x)$. Produce the corresponding Newton fractal by running `newtonFractal(f, Jf)`, and print the resulting image.

b. Recall, from the remark following Problem 2, that f gives the real and imaginary components of $(x_1 + ix_2)^3 - 1$. Now, create a new function `g(x)`, corresponding to the real and imaginary parts of $(x_1 + ix_2)^4 - 1$, and its Jacobian matrix function `Jg(x)`. Run `newtonFractal(g, Jg)` and print the resulting Newton fractal.

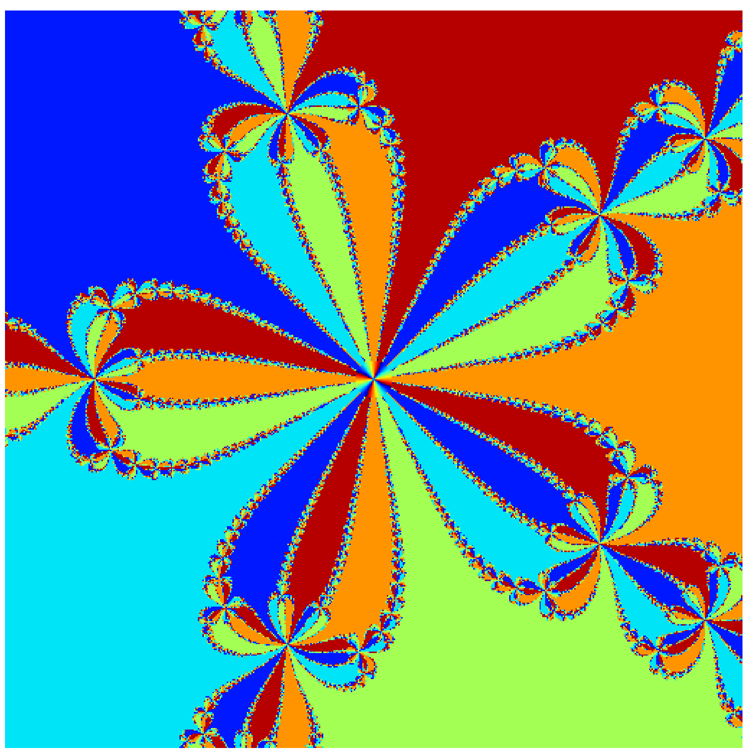

FIGURE 1. Newton fractal for $(x_1 + ix_2)^5 - 1$.