Scientific Computing for DPhil Students I — Assignment 3

Due at lecture at 11:00 on Tuesday, 20 November 2018. This is the third of four assignments this term.

1. *Fitting ellipses via least-squares.* Suppose we have n data points $(x_1, y_1), \ldots, (x_n, y_n)$ in the plane and we want to find an ellipse that fits them well. Finding the geometrically closest fit is a nonlinear problem, but we can come close by a linear formulation. An equation for an ellipse centred at $(0, 0)$ is

$$bx^2 + cxy + dy^2 = 1.$$

Let us view b, c and d as unknowns and find them by solving a linear least-squares problem.

(a) Write down in matrix form an $n \times 3$ least-squares problem whose unknown vector is $(b, c, d)^T$.

(b) Write a Matlab function `[b,c,d] = ellipse(x,y)` which uses "\" to solve this problem. Write driver code to call `ellipse` for the data

$$(3, 3),\ (1, -2),\ (0, 3),\ (-1, 2),\ (-2, -2),\ (0, -4),\ (-2, 0),\ (2, 0).$$

and then print b, c, d and also plot the data points and the fitting ellipse. (Hint: to plot the ellipse you may find it helpful to take a range of angles θ and work with corresponding ratios $y/x = \tan\theta$.)

(c) Here's a little code to let you put in points interactively with the mouse. Try it for some data points of your own choosing and turn in the resulting plot of a fitted ellipse.

```
hold off, axis([-3 3 -3 3]), axis manual, hold on, grid on
x = []; y = []; button = 1;
disp('input points with mouse, button >= 2 for final point')
while button == 1
  [xx,yy,button] = ginput(1)
  x = [x; xx]; y = [y; yy]; plot(xx,yy,'x')
end
```

2. *Solution of the Laplace equation on a square.* Suppose $u(x, y)$ satisfies $\Delta u = 0$ on the unit square $-1 \le x, y \le 1$ with boundary data $u = 0$ on the left side, right side, and bottom and $u = 1 - x^2$ on the top. Our aim is to compute $u(0.99, 0.99)$ to 8 digits of accuracy. It will be convenient to use the following function that returns n Chebyshev points in $(-1, 1)$: `cheb = @(n) cos(pi*((1:n)-.5)/n);`.

(a) Solve the problem numerically by a polynomial least-squares method as follows, using complex arithmetic $z = x + iy$ for convenience. Given an integer $k \ge 0$, sample the $n = 2k + 1$ functions $1, \mathrm{Re}(z), \mathrm{Im}(z), \mathrm{Re}(z^2), \mathrm{Im}(z^2), \ldots, \mathrm{Re}(z^k), \mathrm{Im}(z^k)$ at $m = 8n$ points along the boundary, namely $2n$ Chebyshev points along each side, and construct the corresponding $m \times n$ matrix. Use Matlab backslash to find the n-vector c corresponding to the least-squares solution of $Ac \approx f$, where f the m-point discretization of the boundary data. Based on this computed vector c, make a Matlab function that evaluates $u(x, y)$. Make a table of the computed values $u(0.99, 0.99)$ for $k = 2, 4, 8, \ldots, 128$. What's your best estimate of the exact value?

(b) Now do the same again, but fitting the solution by rational functions instead of polynomials. Specifically, given an integer $k \ge 1$, use the $n = 8k + 1$ functions 1 and $\mathrm{Re}(d_j/(z - z_j)), \mathrm{Im}(d_j/(z - z_j))$, where the points z_j lie on rays extending from the four corners, as in the figure on the right for $k = 8$, at distances $d_j = 2\exp(-\sqrt{j - 1})$, $1 \le j \le k$. Everything else should be as before. Make a table of the computed values $u(0.99, 0.99)$ for $k = 1, 2, 4, \ldots, 64$. What's your best estimate of the exact value?