Numerical Methods for Visual Computing
M. Möller, V. Gandikota, University of Siegen
Winter Semester 19/20

Weekly Exercises 3

To be discussed on Friday, 01.11.2019, 10:15-11:45, in room H-C 6336
Submission deadline: Tuesday, 29.10.2019, 10:15, H-F 104/105

Programming

[8 points] We will practically consider the problem of determining a linear least-squares fit

$$\hat{u} = \arg\min_u \|Au - f\|_2^2$$

in the presence of noisy data f. For this first

- randomly draw 5 coefficients to consider the polynomial

  $$y(x) = c_1x^4 + c_2x^3 + c_3x^2 + c_4x + c_5,$$

  (use a *randn* function to compute random numbers and use or write a function *polyval* to evaluate the polynomial.

- pretend to have $n = 100$ measurements $f = y(x)$ of y, evenly spaced between -1 and 1.

- set 20% of your measurement values in f to $\pm\max(|f|)$ with a random sign, to simulate a noisy signal.

- Using f, determine a least-squares estimate of the coefficients c_i, (see *pinv* in Matlab, *numpy.linalg.pinv* in Python and a LAPACK variant in C++).

- Plot the polynomial curve you get with the estimated $\tilde{c}_i$.

- To get better results consider the following strategy. For *maxiter* many iterations

  - draw 5 indices/measurements from f at random, many languages have a *randperm* for this task.
  - determine the least-squares estimate of the coefficients c_i using only these 5 data points.
  - given the estimated $\tilde{c}_i$, count how many points of f lie within a distance of $0.05 \cdot \max(|f|)$ of the polynomial curve given by the estimated $\tilde{c}_i$.

- Among all iterations select the parameters that gave most inliers and plot the resulting polynomial – does it look better?