Mini-project

The goal of this mini-project is to study the polynomial and trigonometric interpolation, and to compare the noise sensitivity of these techniques.

We will work on the interval $[-1, 1]$. Let's introduce 17 equispaced points $x_0, x_1, ..., x_{16}$ on this interval so that $x_0 = -1$, $x_{16} = 1$. (Convenient MATLAB function to use is **linspace**).

Problem 1

Consider bell-shaped function $f(x) = e^{-10(x-0.2)^2}$. The first set of data (column vector $b^{(\text{exact})}$) to interpolate comes from discretizing $f(x)$ at the points x_j :

$$b_j^{(\text{exact})} = f(x_j), \quad j = 0, .., 16.$$

In order to construct an interpolating polynomial introduce (17×17) Vandermonde matrix A built using points x_j, $j = 0, ..., 16$. Then the coefficients $c^{(\text{exact})}$ of the interpolating polynomial are obtained by solving the linear system

$$Ac^{(\text{exact})} = b^{(\text{exact})}.$$

You can solve this system by using the backslash operator \ in MATLAB. To make a nice plot of the result, build a fine discretization grid on $[-1, 1]$ with 129 points. Compute the values of the interpolating polynomial (vector p) on the fine grid. Now on the same figure plot the values you interpolate (say, by using commands **plot(x,bexact,'o'); hold on;**) and the graph of the interpolating polynomial (i.e. **plot(xfine,p)**). Print the picture. Save values of $c^{(\text{exact})}$ for the future use.

Problem 2

Let us now perturb the right hand side of the system $b^{(\text{exact})}$. In order to do so, first generate vector n of normally distributed random numbers using command **randn** (e.g. **randn(size(bexact)).** Next, normalize n so that for the new n

$$||n||_2 = 0.01||b^{(\text{exact})}||_2$$

(Please, do not use the **norm** command in MATLAB – use the dot-product instead).
Next, compute the perturbed right hand side $b^{(\text{pert})}$ by adding n to $b^{(\text{exact})}$.

$$b^{(\text{pert})} = b^{(\text{exact})} + n.$$

Obviously, relative perturbation (or relative error) in b is 0.01 in the 2-norm.

Now you can solve the perturbed system to find $c^{(\text{pert})}$:

$$Ac^{(\text{pert})} = b^{(\text{pert})}.$$

As in the problem 1, compute values of the new interpolating polynomial $p^{(\text{pert})}$ on the fine grid and plot those values and the data $b^{(\text{pert})}$ that are being interpolated, in the same figure (Figure #2). Print this figure.

Finally, compute the 2-norm relative error in the polynomial coefficients c as

$$\text{Rel. error} = \frac{||c^{(\text{exact})} - c^{(\text{pert})}||_2}{||c^{(\text{exact})}||_2}.$$

What's the ratio of the relative error in c to the relative error in b? Print this ratio and the relative error in c.

Problems 3 and 4.

Repeat problems 1 and 2 replacing polynomial interpolation with the DFT-based trigonometric interpolation. For a fair comparison, throw away point x_{16} and the corresponding data (the trigonometric polynomial will be periodic). For the trigonometric interpolation there is no need to form the matrix. The Fourier coefficients can be obtained using the **FFT** and **FFTSHIFT** commands in MATLAB. The resulting interpolating polynomial can be evaluated on a fine grid (128 points) by adding some number of zeros (precisely 128-16=112) to the array of the Fourier coefficients.

To complete problems 3 and 4, plot in two new figures the data $b^{(\text{exact})}$ and $b^{(\text{pert})}$ with the graphs of the corresponding interpolating trigonometric polynomials, and print the figures. Also, compute the relative error in the coefficients of the interpolating trigonometric polynomial corresponing to 0.01 perturbation of the interpolated data b, and print these numbers.