Exercise 2 – Approximation

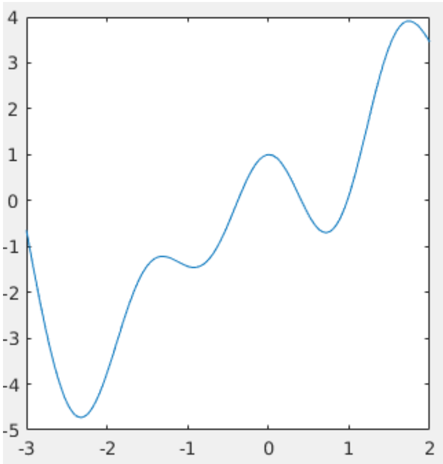
(Thu 10-12, PA)

Task 1.

Please plot the following function:

$$f(x) = x^2 \sin(x) + \mathrm{e}^{0.1x} \cos(4x)$$

for x∈ < -3,2>.
You should expect plot similar to:



Task 2.

Generate a new set of equidistant points within the given range of size n=30. (Just generate a new vector of data which will be used for later interpolation as input data.)

Task 3.

Implement the polynomial approximation as a MATLAB function and plot the approximating function with the given range from Task. 1 for order of polynomial equal 7 and 17. The approximating function should return vector of coefficients and it should have the following form:

```
function a = find_coefficients(xa, ya, n)
        ...
end
```

Where x,y are the approximation input data points, and n is the order of the desired polynomial.

Result: On a single figure please show four plots: the given function f(x) presented as a smooth curve, the approximating input data points (points from Task.2. Presented as circles) and two smooth numerical approximations p(x) for n = 7 and n = 17.

Task 4.

Find an average and maximum error of the approximation within the given range of x using N=80 probes.

Error definition:

$$e(x) = |f(x) - p(x)|$$

Average error definition:

$$\bar{e} = \frac{1}{b-a}\int_a^b e(x)\mathrm{d}x \approx \frac{1}{b-a}\frac{b-a}{N-1}\sum_{i=1}^{N-1}\frac{e(x_i)+e(x_{i+1})}{2} = \frac{1}{N-1}\sum_{i=1}^{N-1}\frac{e(x_i)+e(x_{i+1})}{2}$$

Task 5.

Fill in the experimental table which presents the average and maximum errors depending on the number of interpolating input data. Use the following template. Please probe the given interval with N=220 probes.

Order of the polynomial	**n = 7**		**n = 17**	
	Average error	**Maximum error**	**Average error**	**Maximum error**
6				
7				
9				
11				
13				
15				
17				

Task 6.

Implement the least squares approximation for the data from Task 2. based on the following base functions set:

$\phi_0(x) = sin(x)$
$\phi_1(x) = cos(x)$
$\phi_2(x) = sin(4x)$
$\phi_3(x) = cos(4x)$
$\phi_4(x) = x \cdot sin(x)$
$\phi_5(x) = \mathrm{e}^{0.1x} \cdot cos(x)$
$\phi_6(x) = x^2 \cdot sin(x)$
$\phi_7(x) = x^2 \cdot cos(x)$

You should implement these functions as a separate MATLAB function and follow the example from the lecture having an array of pointers to base functions (another name is a reference to a function). Below is presented an idea of the function, and it is only the idea. It is not a working piece of code!!! Please notice, that it must be corrected to work with given number of base functions.

```
function main
….
p = {@f0, @f1, @f2, @f3, @f4, @f5, ....}
for r = 1:3
   for c = 1:3
      A(r,c) = sumuj(xps, p{r}, p{c});
   end
   b(r) = sumujb(xps, yp, p{r});
end
a = A\b
...
End
function y = f1(x)
         y = sin(x);
end

function y = f2(x)
         y = cos(x);
end
...
```