APPROXIMATION THEORY HOMEWORK 2

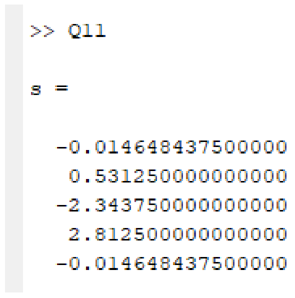
TOMMENIX YU
ID: 12370130
STAT 31220
DUE WED MAR 29ST, 2023, 11PM

Discussed with classmates.

Exercise 1.

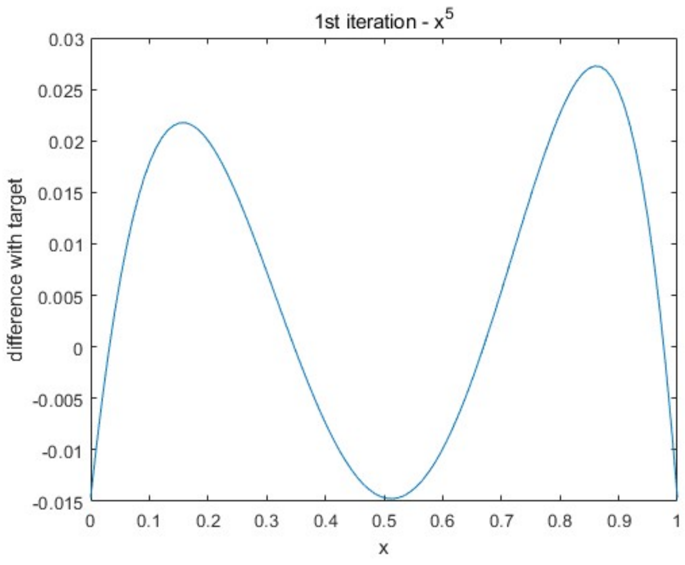
Proof.

(1): We create the matrix, do the linear solve and get that the final list of coefficients and error is:

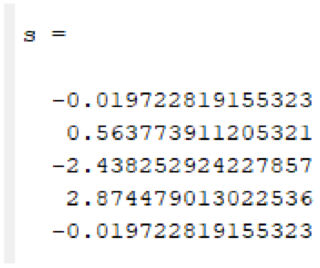


```
>> Q11

s =

  -0.014648437500000
   0.531250000000000
  -2.343750000000000
   2.812500000000000
  -0.014648437500000
```

and the error function is(generated by the file "TommenixYu_Q1a")

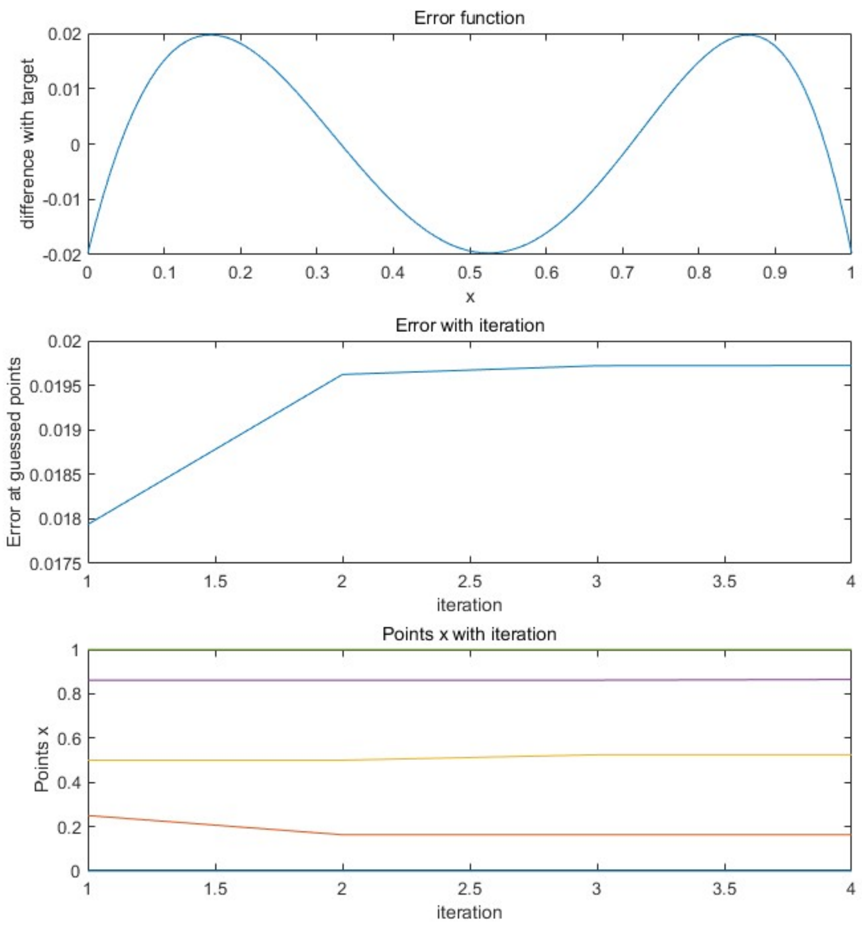


(2): The list of coefficients and error is (generated by the file "TommenixYu_Q1b"):

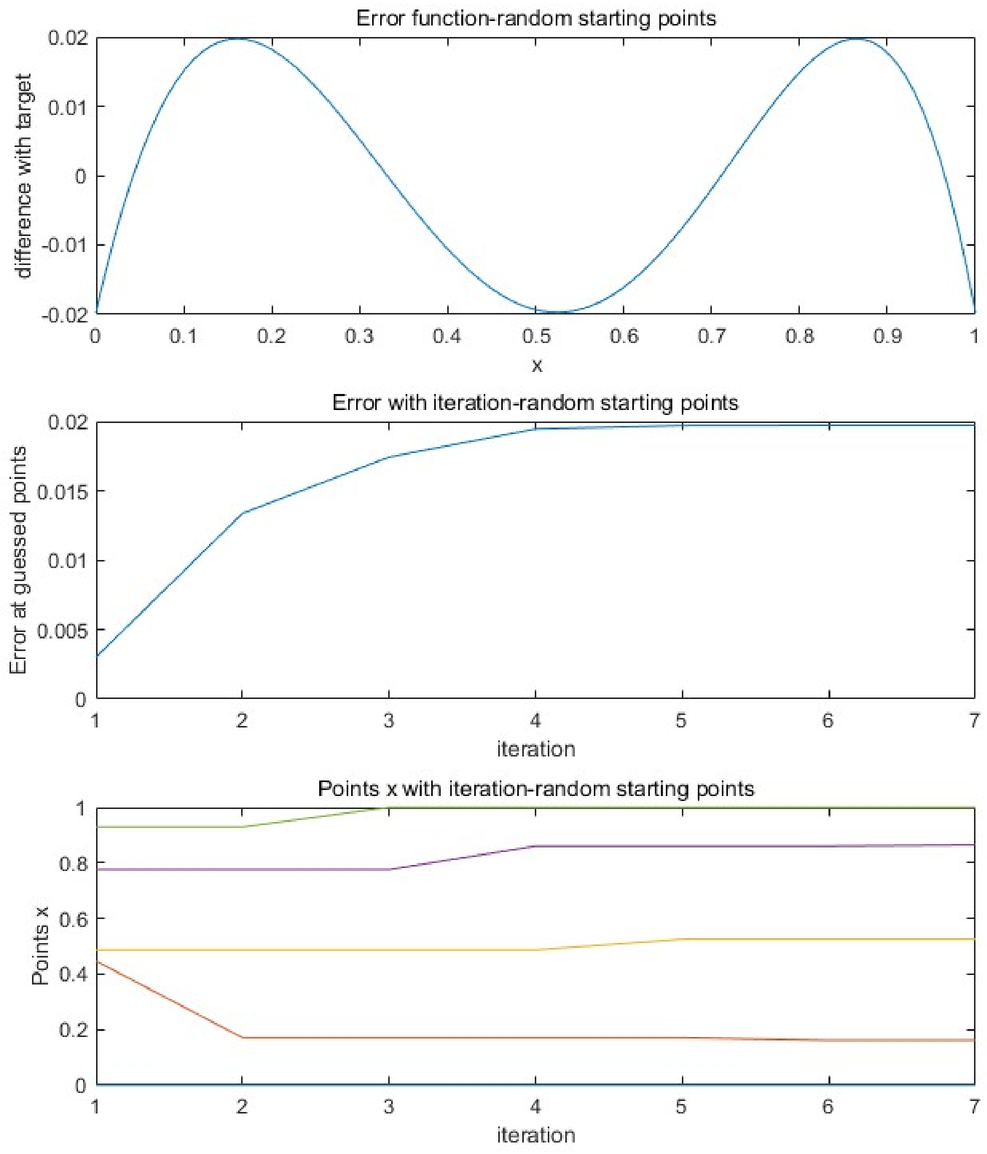


```
s =

  -0.019722819155323
   0.563773911205321
  -2.438252924227857
   2.874479013022536
  -0.019722819155323
```

and the error function, along with the iteration caused changes are (generated by the file "TommenixYu_Q1b")



and if I changed to random starting points (to do this I use the same file as above, but there's a commented random starting point line which, if uncommented, yields the result (graph titles are different of course))

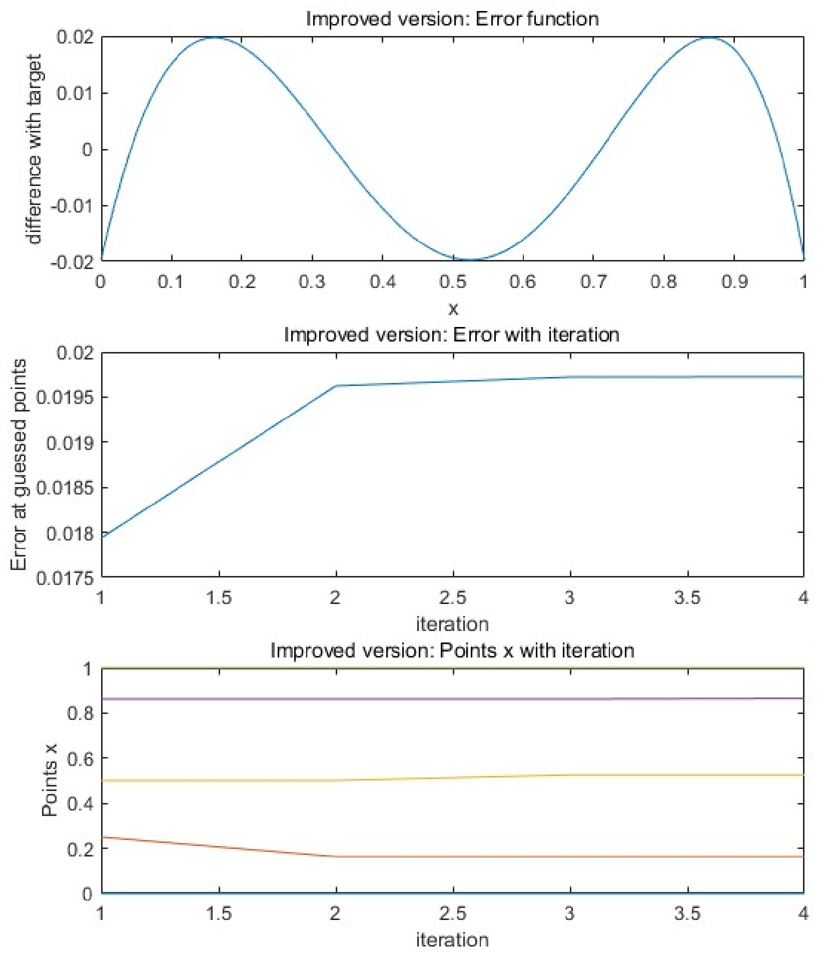


(3): So what I did is just to up date all points by finding the closest maximum point with the same sign. I updated the values such that x_i is moved to the maximum value (with the same sign as x_i) between $[x_{i-1}, x_{i+1}]$ and for the end points the range is $[0, x(1)]$ or $[x(n), 1]$.

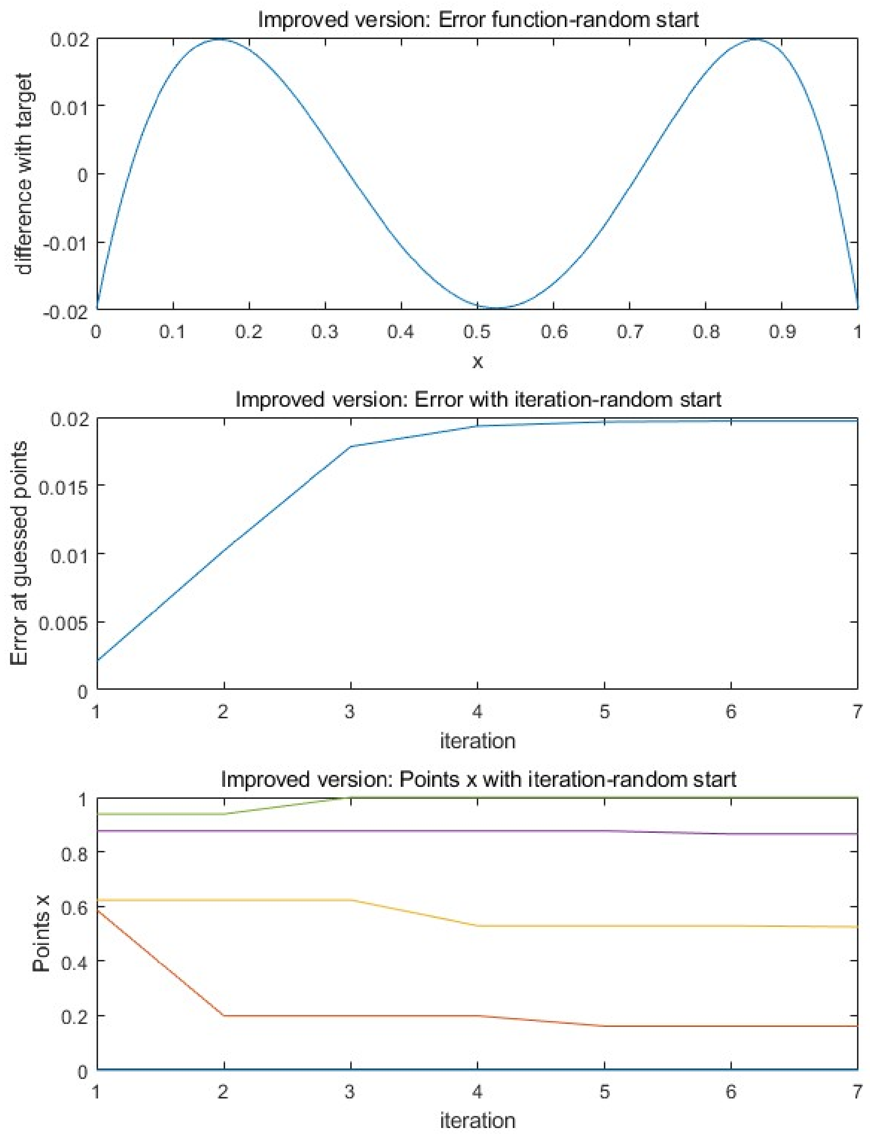
This is correct since these points are made oscillating by the matrix. Turns out that indeed this works the same. The list of coefficients and error is:

```
s =

  -0.019738429195426
   0.563739718790718
  -2.437961660042835
   2.874201814818812
  -0.019716233212612
```

and the error function is (generated by the file "TommenixYu_Q1c")



and if I changed to random starting points (to do this I use the same file as above, but there’s a commented random starting point line which, if uncommented, yields the result (graph titles are different of course))



□

Exercise 2.

Proof.

(a): Show the algorithm holds:

By exponential form of cos function we get $T_n = 2xT_{n-1} - T_{n-2}$ which is equivalent to

$$T_{n-2} = 2xT_{n-1} - T_n \tag{0.1}$$

for $n \geq 0$, and we try to employ this relation, using in the end the fact that $T_0 = 1, T_1 = s$.

Denote the partial sum

$$S_n := \sum_{k=0}^{n} \alpha_k T_k$$

and u_i are defined as in the problem. Now we can compute (remember $u_{n+1} = 0$ so we insert it in) repeated using 0.1

$$\begin{aligned}
S_n &= \alpha_n T_n + \alpha_{n-1}T_{n-1} + \alpha_{n-2}T_{n-2} + S_{n-3} = u_n(2sT_{n-1} - T_{n-2}) + \alpha_{n-1}T_{n-1} + \alpha_{n-2}T_{n-2} + S_{n-3} \\
&= (2su_n - u_{n+1} + \alpha_{n-1})T_{n-1} + (\alpha_{n-2} - u_n)T_{n-2} + S_{n-3} \\
&= u_{n-1}T_{n-1} + (\alpha_{n-2} - u_n)T_{n-2} + S_{n-3} \\
&\overset{similarly}{=} u_{n-2}T_{n-2} + (\alpha_{n-3} - u_{n-1})T_{n-3} + S_{n-4} \\
&\vdots \\
&= u_1T_1 + (\alpha_0 - u_2)T_0 = u_1 s + \alpha_0 - u_2 = \frac{1}{2}(u_0 + \alpha_0 - u_2)
\end{aligned}$$

where each time we reduce S_n into a polynomial-valued-coefficient polynomial of T_0 through T_k, where the leading polynomial-valued-coefficient is exactly u_k. Hence we are done.

(b): T_n satisfies the ODE:

Just compute

$$T_n' = \frac{n\sin(n\arccos(x))}{\sqrt{1-x^2}}$$

$$T_n'' = \frac{nx\sin(n\arccos(x))}{(1-x^2)^{3/2}} - \frac{n^2 \cdot T_n}{1-x^2}$$

which means

$$(1-x^2)T_n'' = xT_n' - n^2T_n$$

which is equivalent to the given ODE.

□

Exercise 3.

Proof.

Simplify the problem:

Let's say the best approximation to f is p, then we know that the best approximation to $f-p$ is 0 and $g-p$ is close enough to f. Thus, we WLOG suppose Pf, the best approximation to f, is 0.

Moreover, we can assume by scaling that $||f|| = 1$. Hence, f should be equi-oscillating between -1 and 1.

Let's assume that $||f-g|| \leq \delta$, then $||g|| \leq 1+\delta$, hence we have

$$||f-(Pg-Pf)|| \leq ||f-g|| + ||g-(Pg-Pf)|| \leq 1+2\delta$$

and if we can show that $||Pg|| = ||Pg-Pf|| \leq \varepsilon = O(\delta)$ then we know given $||f-g|| \leq \delta$, $||Pf-Pg|| \leq \varepsilon$ hence the operator from f to its best approximation is continuous.

For the last result we use the source `https://www.jstor.org/stable/pdf/2039182.pdf?refreqid=excelsior` (originally result by Freud) to show the following lemma:

Lemma 0.1. *For* $||f|| = 1$ *whose best approximation is* 0*, if polynomial* q *has that* $||f-q|| < 1+\varepsilon$*, then* $||q|| = O(\varepsilon)$*.*

Proof.

We first note again the fact that since f has best approximation 0, f is the error so it oscillates at $n+2$ points, call them $x_0, \dots, x_{n+1}$. WLOG let's assume $f(x_0) = 1, f(x_1) = -1, \dots$. Now, for the purpose of bounding Lagrange interpolation terms later, we define

$$w(x) = \prod_{i=0}^{n+1}(x-x_i)$$

and denote

$$M := \max w'(x); \qquad m := \min w'(x).$$

Now we show that for all x_i we have

$$|q(x_i)| \leq \frac{M(n+1)\varepsilon}{m}.$$

Assume the contrary that for some j we have $q(x_n) \geq \dfrac{M(n+1)\varepsilon}{m}$, where the sign is WLOG chosen. Now write q out as it's Lagrange interpolation form we get

$$q(x) = w(x)\sum_{k=0}^{n+1}\frac{q(x_k)}{w'(x_k)(x-x_k)} = \left(\sum_{k=0}^{n+1}\frac{q(x_k)}{w'(x_k)}\right)x^{n+1} + O(x_n)$$

but q is a polynomial of degree n so the first coefficient must be 0, that is

$$\sum_{k=0}^{n+1} \frac{q(x_k)}{w'(x_k)} = 0$$

but this means that even in the worst case where the other $n+1$ terms x_i are all evenly negative (thus smallest max value), the following still holds: for some $l \neq j$ we have

$$q(x_l) \leq -\frac{w'(x_l)}{w'(x_j)} \frac{q(x_j)}{n+1} \leq \varepsilon$$

and

$$\operatorname{sgn}\left[\frac{q(x_j)}{w'(x_j)}\right] = -\operatorname{sgn}\left[\frac{q(x_l)}{w'(x_l)}\right].$$

Now since $|f - q| < 1 + \varepsilon$ and $|q(x_j)| \geq \varepsilon$, $|q(x_l)| \geq \varepsilon$, we know that at the point x_j, x_l q and f has the same sign. Thus, by equi-oscillation of the choice of x_i we know that

$$\operatorname{sgn}[q(x_j)] = \operatorname{sgn}[(-1)^{l-j} q(x_l)]$$

and since just knowing what $w(x)$ is, $w'(x)$ must oscillate at the points x_i too. So

$$\operatorname{sgn}\left[\frac{q(x_j)}{w'(x_j)}\right] = \operatorname{sgn}\left[\frac{q(x_l)}{w'(x_l)}\right]$$

contradiction! So we know at all x_i we have the bound

$$|q(x_i)| \leq \frac{M(n+1)\varepsilon}{m}$$

plugging this into the Lagrange interpolation form of q we see that

$$|q(x)| = \left|\sum_{k=0}^{n+1} \frac{q(x_k)w(x)}{w'(x_k)(x-x_k)}\right| \leq \sum_{k=0}^{n+1} \left|\frac{q(x_k)w(x)}{w'(x_k)(x-x_k)}\right| \leq (n+2)\frac{(n+1)M\varepsilon}{m^2} \cdot \left\|\frac{w(x)}{(x-x_k)}\right\|$$

but $\dfrac{w(x)}{(x-x_k)} = \dfrac{1}{(x-x_k)} \prod_{i=0}^{n+1} (x - x_i)$ is continuous over a compact set, thus it's infinity norm is bounded by L. So we have

$$|q(x)| \leq (n+2)\frac{(n+1)ML\varepsilon}{m^2} = O(\varepsilon).$$

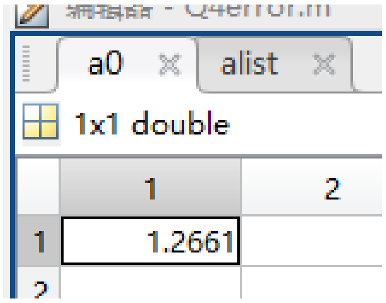
□

Now going through the above argument we get that the best polynomial approximation operator is continuous with respect to the infinity norm. □

Exercise 4.

Proof.

We use the formula in chap 3 to compute the coefficients with a_0 being

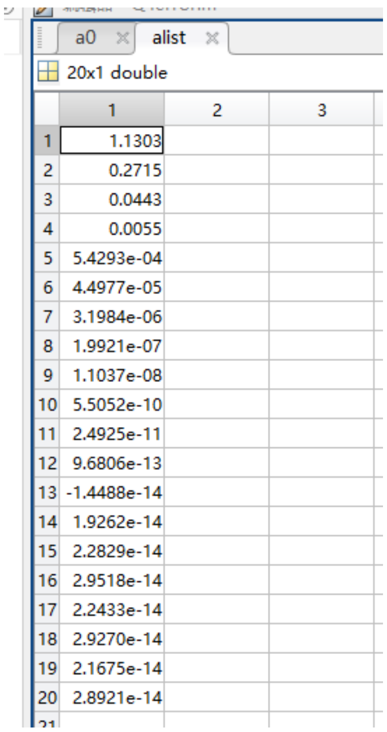


a0 | alist

1x1 double

	1	2
1	1.2661	
2		

and the rest coefficients are (up to 20):

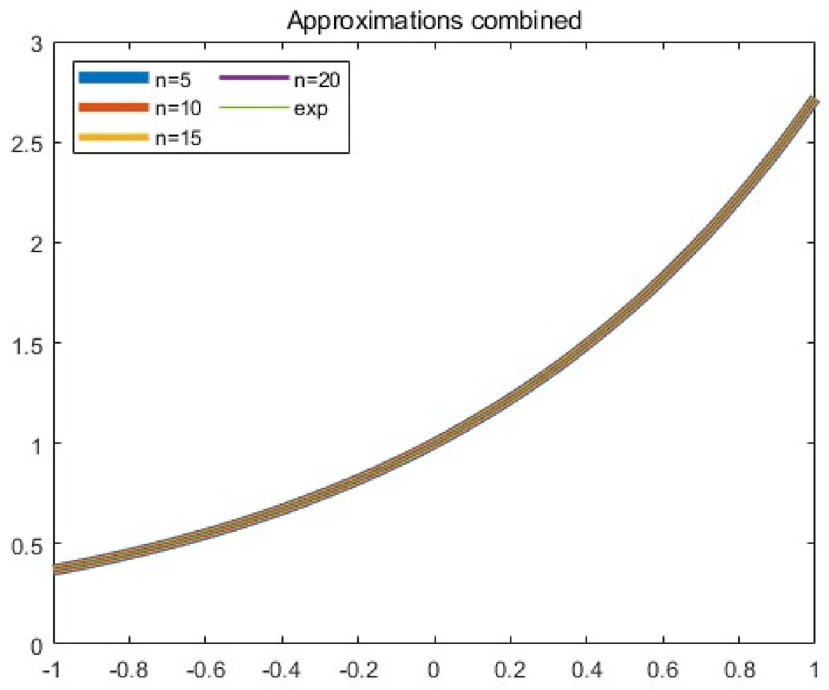


a0 | alist

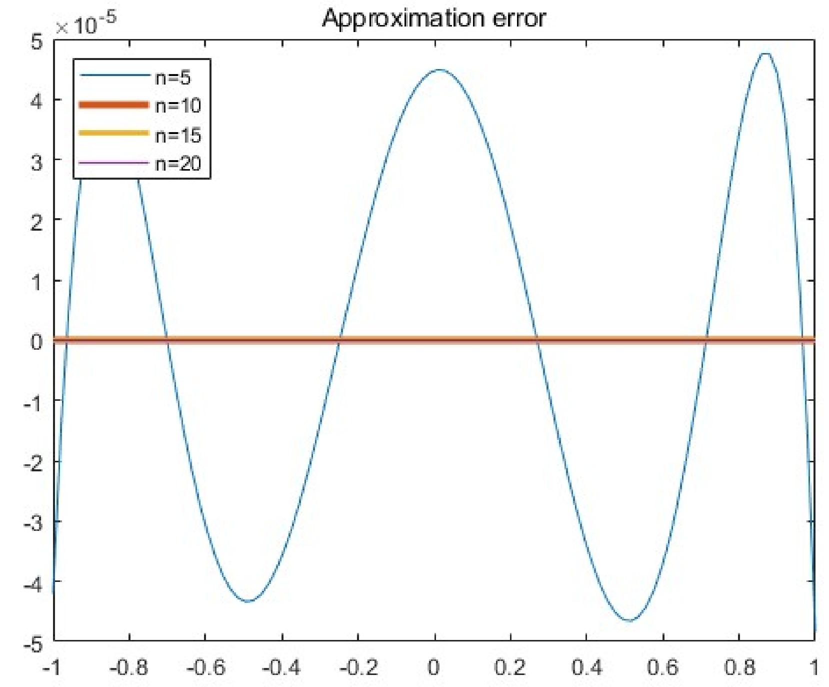
20x1 double

	1	2	3
1	1.1303		
2	0.2715		
3	0.0443		
4	0.0055		
5	5.4293e-04		
6	4.4977e-05		
7	3.1984e-06		
8	1.9921e-07		
9	1.1037e-08		
10	5.5052e-10		
11	2.4925e-11		
12	9.6806e-13		
13	-1.4488e-14		
14	1.9262e-14		
15	2.2829e-14		
16	2.9518e-14		
17	2.2433e-14		
18	2.9270e-14		
19	2.1675e-14		
20	2.8921e-14		

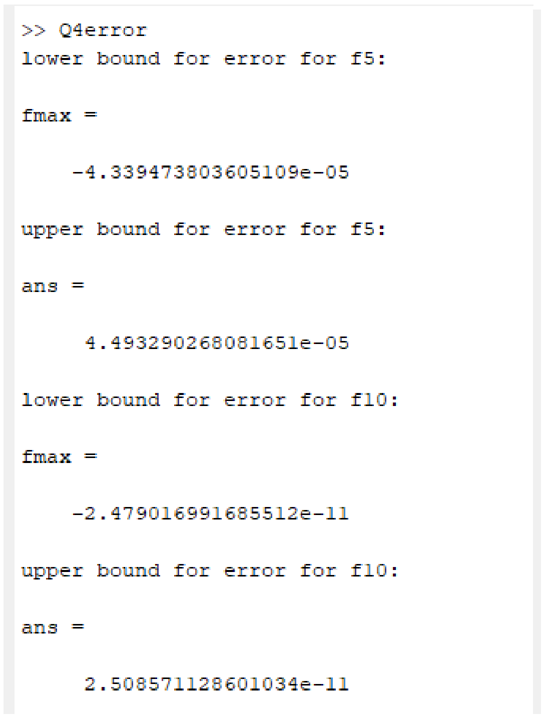
Note that when $n \geq 13$ or so the coefficient is already at machine precision, thus non of further matters. But I've printed the approximations and the errors so we can see that it is indeed so. The approximation is (generated by the file "TommenixYu_Q4"):



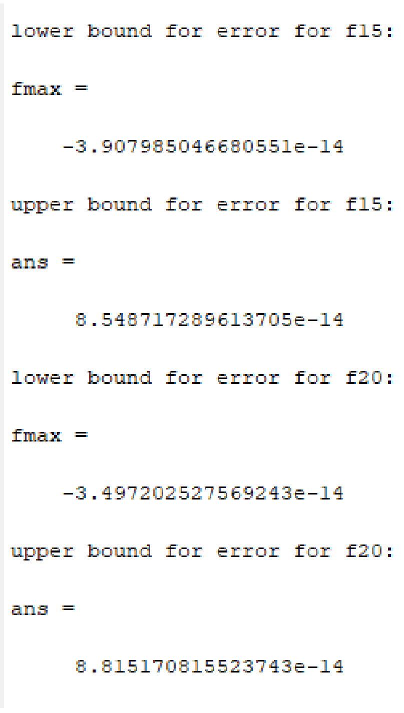
and the error graph is (generated by the file "TommenixYu_Q4", change the plotted function to f5(t)-exp(t),etc):



To compute the upper and lower bound I just used fminbnd to get: (generated by the file "Q4 error")



```
>> Q4error
lower bound for error for f5:

fmax =

   -4.339473803605109e-05

upper bound for error for f5:

ans =

    4.493290268081651e-05

lower bound for error for f10:

fmax =

   -2.479016991685512e-11

upper bound for error for f10:

ans =

    2.508571128601034e-11
```



```
lower bound for error for f15:

fmax =

   -3.907985046680551e-14

upper bound for error for f15:

ans =

    8.548717289613705e-14

lower bound for error for f20:

fmax =

   -3.497202527569243e-14

upper bound for error for f20:

ans =

    8.815170815523743e-14
```

□