Problem Set 1

Skander Garchi Casal

September 28, 2018

1 Quantitative Macroeconomics

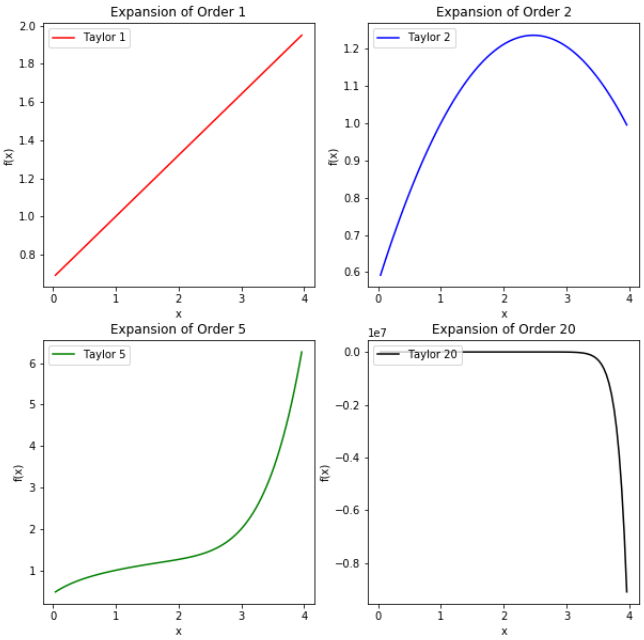
1.1 Problem set 1

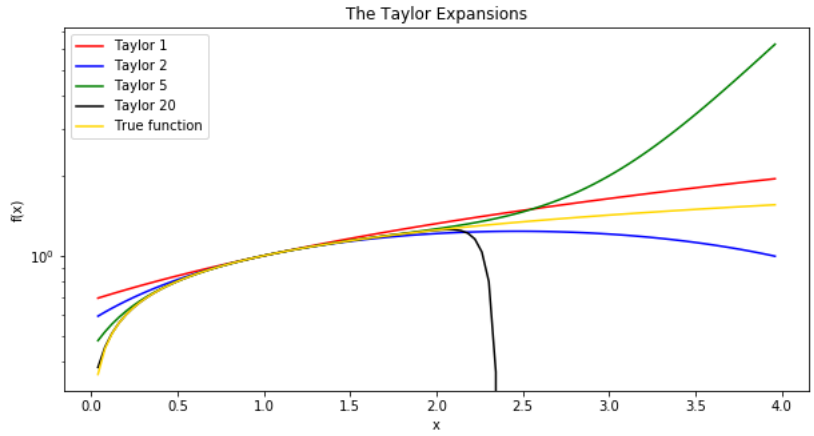
Before starting the problem set, let me tell you how my code is stuctured. I created several files that answer different questions of the problem set. I have given them names refering to the point or exercise they are answering. Most of the functions I use in the problem set were functions created by myself, you will find them all in the file useful-functions.py.

Exercise 1

1. Approximate $f(x) = x^{0.321}$ with a Taylor series around $\bar{x} = 1$. Compare your approximation over the domain (0,4). Compare when you use up to 1, 2, 5 and 20 order approximations. Discuss your results.

Here $f(x) = x^n$ where n = 0.321. To write the Taylor series I need all derivatives until order 20. I notice that $f^m(x) = x^{n-m} \prod_{i=0}^{m-1}(n-i)$ which helps me compute the expansion. The function that performs it is Taylor-pol.





We can notice from the first figure that the expansions seem to have standard shapes. To be able to compare them we refer to figure 2. As expected they are all very close to each other around 1. However, for the set of x's higher than 2 the functions separate from each other. Surprisingly, from all the functions it is the taylor 1 that gives the best fit and the Taylor 20 the worst one.

Exercise 2

2. Approximate the ramp function $f(x) = \frac{x+|x|}{2}$ with a Taylor series around $x = 2$. Compare your approximation over the domain (0,6). Compare when you use up to 1, 2, 5 and 20 order approximations. Discuss your results.

$f(x) = \frac{x+|x|}{2}$

$$f(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases}$$

$$f'(x) = \begin{cases} 0 & x < 0 \\ \emptyset & x = 0 \\ 1 & x > 0 \end{cases}$$

$$f^n(x) = \begin{cases} 0 & x < 0, \forall n > 1 \\ \emptyset & x = 0, \forall n > 1 \\ 0 & x > 0, \forall n > 1 \end{cases}$$

The Taylor expansion formula around $\bar{x}$ is:

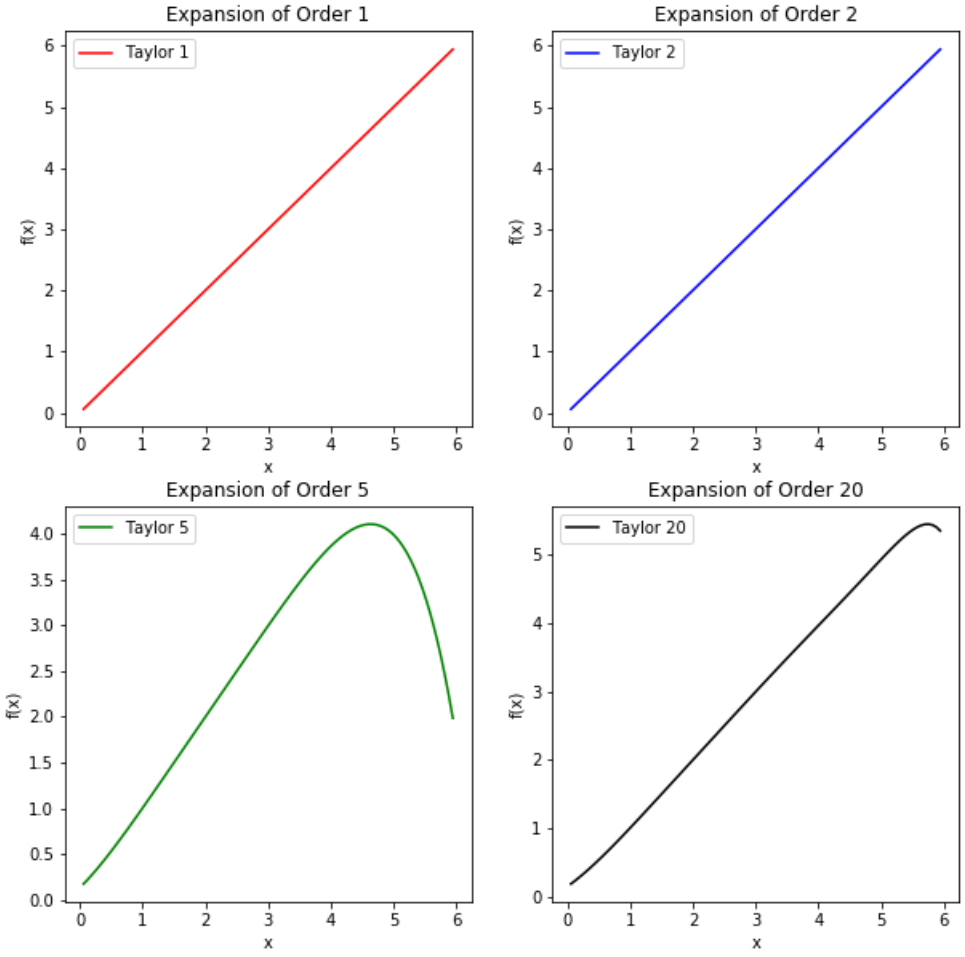
$T(f, \bar{x}) = f(\bar{x}) + \sum_{i=1}^{+\infty} \frac{(x-\bar{x})^i}{i!}$

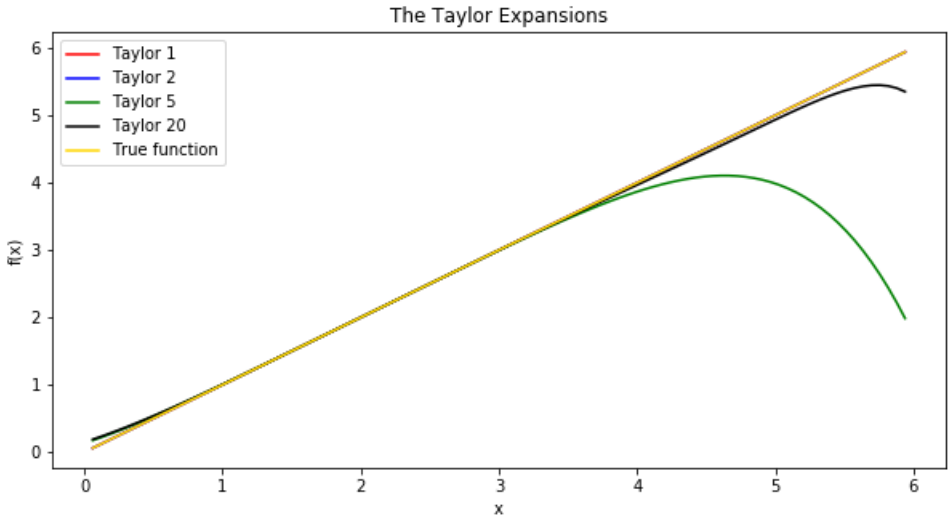
Since $f^n(\bar{x}) = 0, \forall n > 1$

This means that:

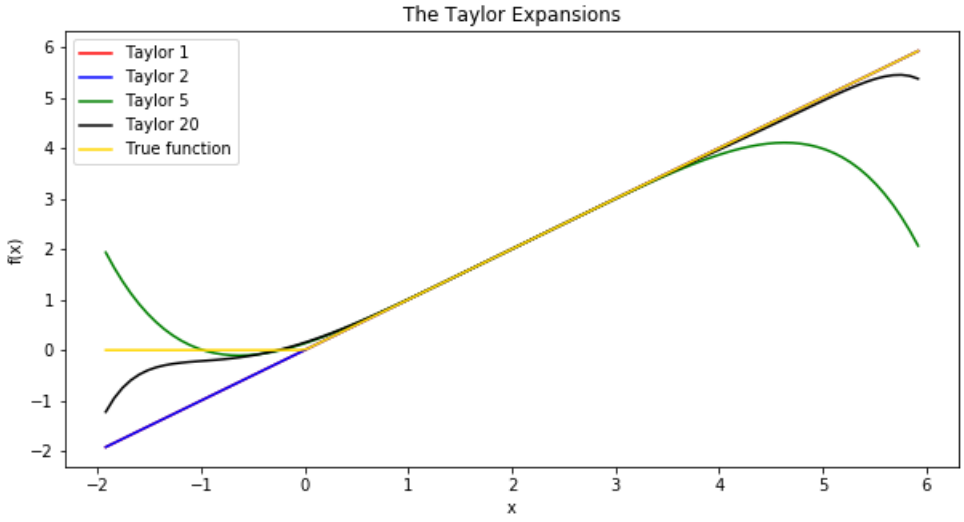
$T(f, \bar{x}) = f(\bar{x}) + (x - \bar{x})$

We should theoretically have the save Taylor expansion for all the orders. However, because of numerical errors in the computation of the derivative or because of the existance of a kink at $x = 0$, we obtain a slitly different result, described in the following graphs.


Expansion of Order 1
Taylor 1
f(x)
x
Expansion of Order 2
Taylor 2
f(x)
x
Expansion of Order 5
Taylor 5
f(x)
x
Expansion of Order 20
Taylor 20
f(x)
x



You can see here that we do not obtain the expected results. The higher order Taylor expansions do poorly in the approximation of the function when x approches 6.



If we extend the domain to -2, we can understand the origine of the problem. The kink located at $x = 0$ gives some curvature to the Taylor expansions of

higher order. Hence, that curvature also appears arround $x = 6$. A better alternative here to numerical differenciation might be to use a library that uses symbolic mathematics such as sympy.

Exercise 3

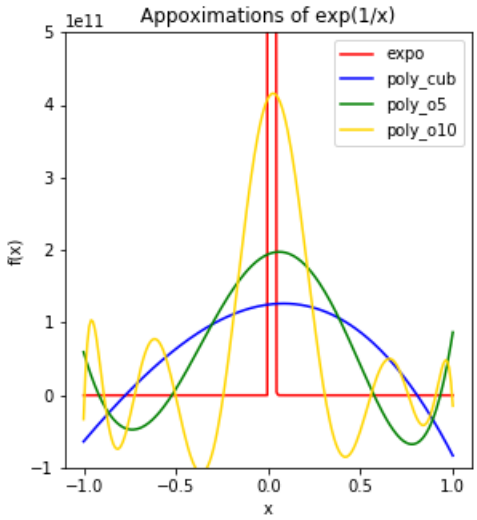
3. Approximate these three functions: $e^{\frac{1}{x}}$, the runge function $\frac{1}{1+25x^2}$, and the ramp function $f(x) = \frac{x+|x|}{2}$ for the domain $x \in [-1, 1]$ with:

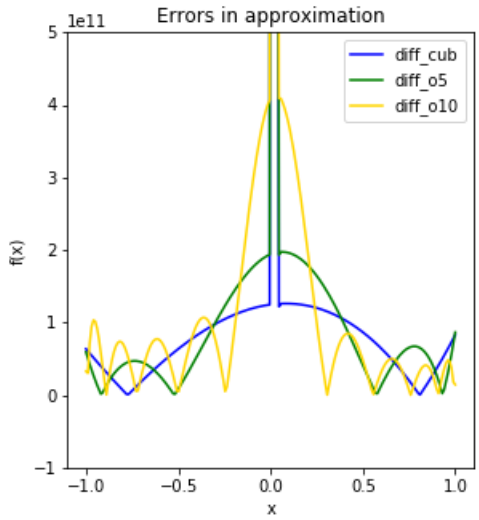
Before answering the points let me describe the procedure that follow several of the functions that I constructed for interpolation. Let $x = (x_1, x_2, ..., x_n)$ be a vector of dimension n, containing the different nodes. Let $F = \{1, z, z^2, ..., z^m\}$ a family of polynomials and $\Theta = (\theta_0, \theta_1, ..., \theta_i)$'.

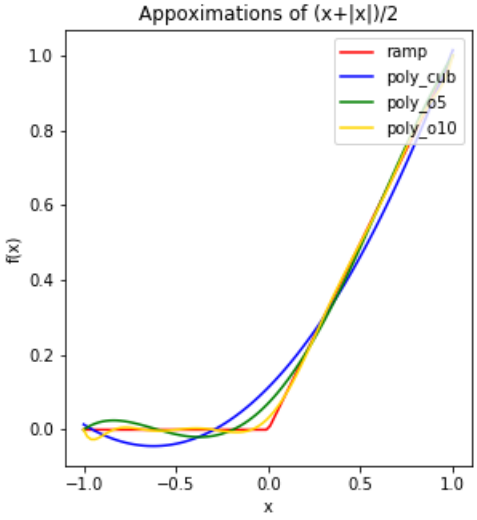
Let $F^* = \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \\ x_1^2 & x_2^2 & \dots & x_n^2 \\ \vdots & \vdots & \ddots & \\ x_1^m & x_2^m & \dots & x_n^m \end{bmatrix}$, and $f^* = (f(x1), f(x2), ..., f(xn))$

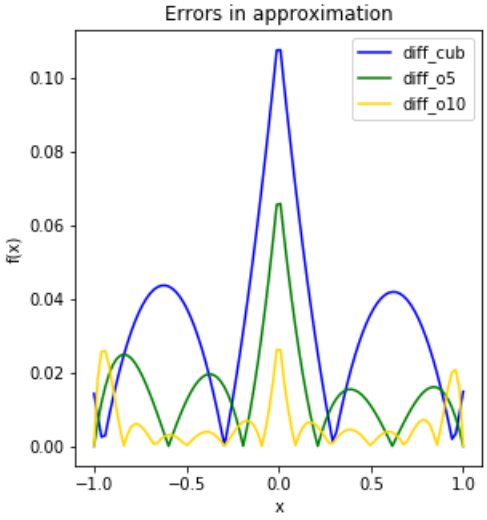
Where f(.) is the function we are trying to interpolate.
We need to solve the follwing system of equations:
$\theta' F^* = f^*$, F^* is not squared if the number of nodes is different than the number of functions in the family. So to isolate θ' we do the following.
$\theta' F^* F^{*\prime} = f^* F^{*\prime}$
$\theta' = f^* F^{*\prime} (F^* F^{*\prime})^{-1}$
These are the steps that a few of the functions I created follow. It will work if the system is just identified but also if it is overidentified.

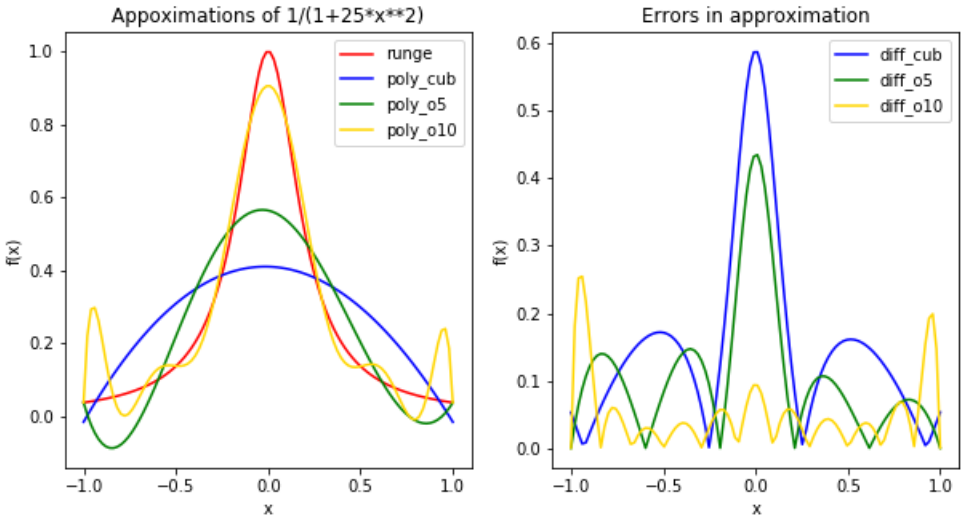
1. Evenly spaced interpolation nodes and a cubic polynomial. Redo with monomials of order 5 and 10. Plot the exact function and the three approximations in the same graph. Provide an additional plot that reports the errors as the distance between the exact function and the approximand.
For the cubic polynomial interpolation $F = \{1, z, z^2, z^3\}$. So we just need to follow the steps that I previously decribed.


Appoximations of exp(1/x)
1e11
expo
poly_cub
poly_o5
poly_o10
f(x)
x


Errors in approximation
1e11
diff_cub
diff_o5
diff_o10
f(x)
x

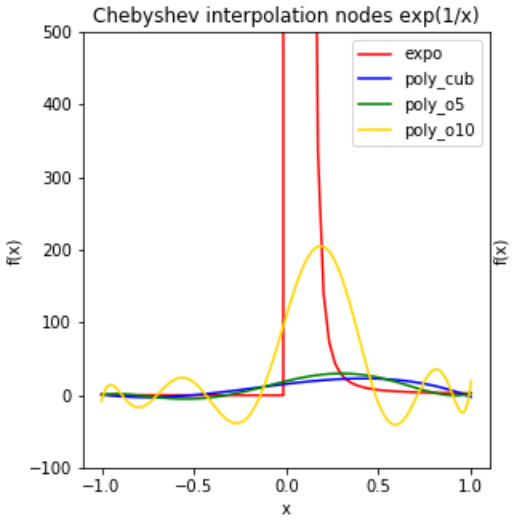

Appoximations of (x+|x|)/2
ramp
poly_cub
poly_o5
poly_o10
f(x)
x

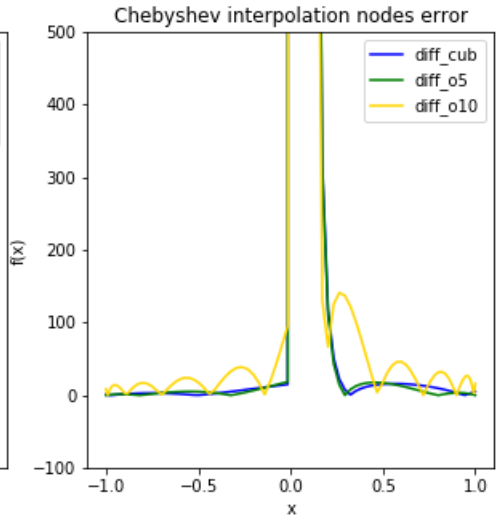

Errors in approximation
diff_cub
diff_o5
diff_o10
f(x)
x

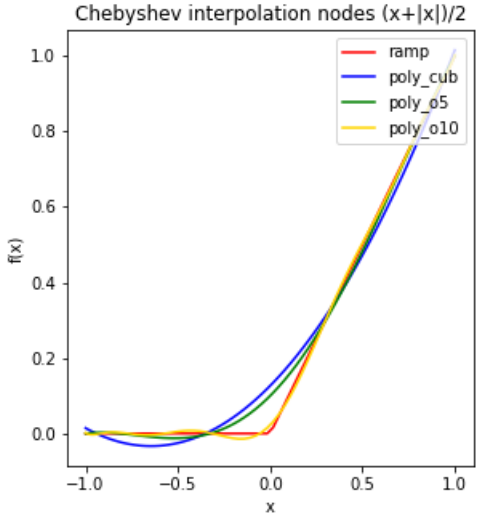


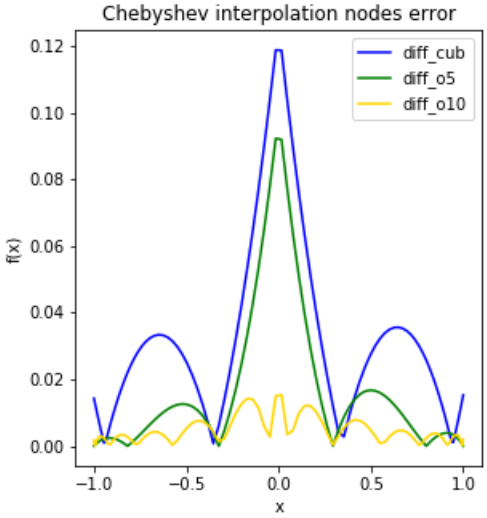
2. Chebyshev interpolation nodes and a cubic polynomial. Redo with monomials of order 5 and 10. Plot the exact function and the three approximations in the same graph. Provide an additional plot that reports the errors as the distance between the exact function and the approximand.

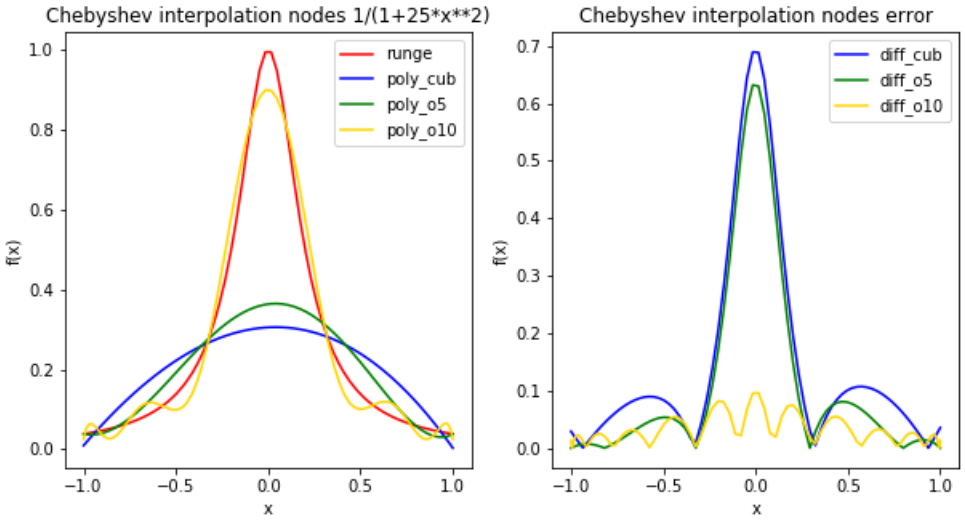
The chebychev interpolation nodes work as follow. Let say that we have a lower bound a, an upper bound b and a number of elements to our grid. The set of Chebychev nodes C_n is such that $C_n = \{\frac{a+b}{2} + \frac{b-a}{2} cos(\frac{i-1}{N}\pi) : i = \{1, 2, ..., N\}\}$. Basically, Cn will contain the set of cosines of the uniformly separated points on the circle of radius b-a. This will have as effect to make the points of the grid closer if they are at the edges of the domain. However, if the points are in the middle of the interval they will be more separated. If the function we are trying to interpolate has more curvature around the edges it will help us make a better interpolation as we can see comparing the following graphs to the previous ones. If you look at the file useful-functions.py you will notice that the function cheb-node that I created perfoms this routine.


Chebyshev interpolation nodes exp(1/x)
expo
poly_cub
poly_o5
poly_o10
f(x)
x


Chebyshev interpolation nodes error
diff_cub
diff_o5
diff_o10
f(x)
x


Chebyshev interpolation nodes (x+|x|)/2
ramp
poly_cub
poly_o5
poly_o10
f(x)
x

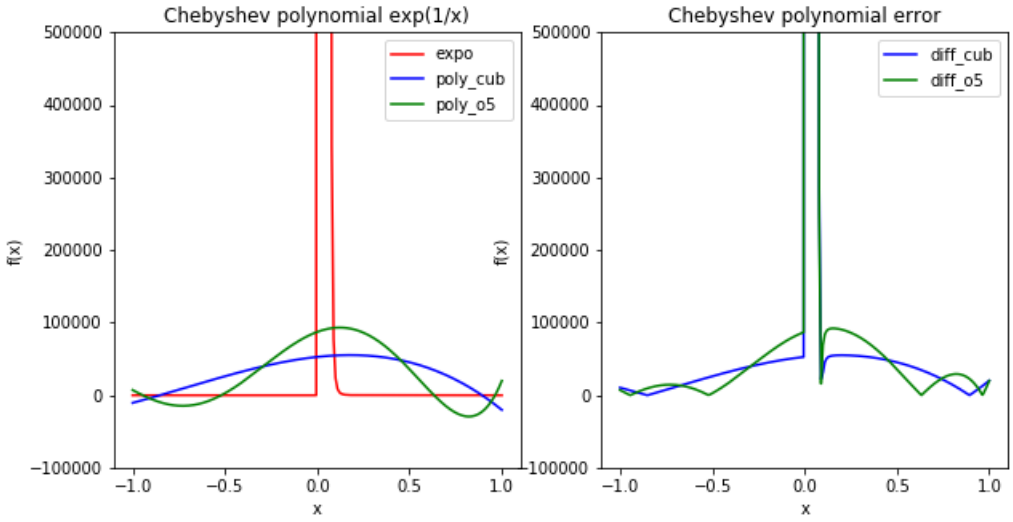

Chebyshev interpolation nodes error
diff_cub
diff_o5
diff_o10
f(x)
x

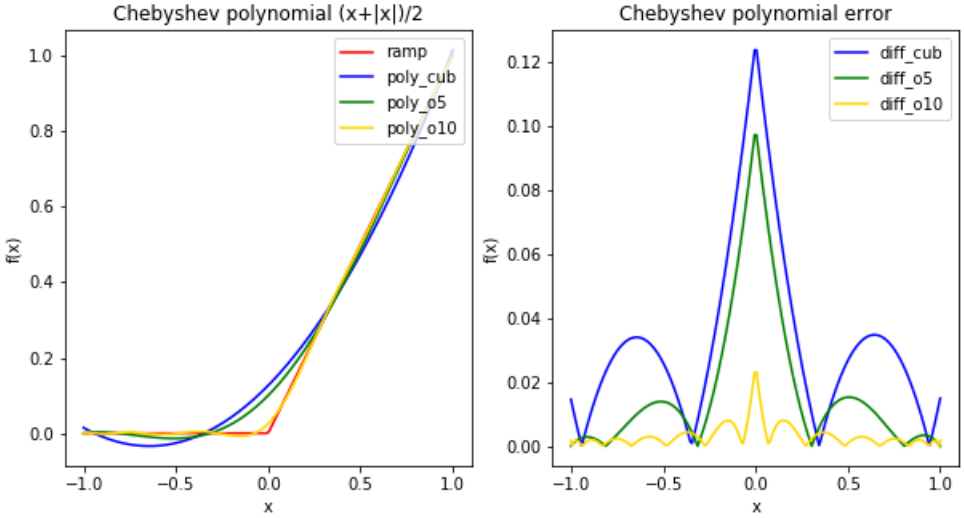


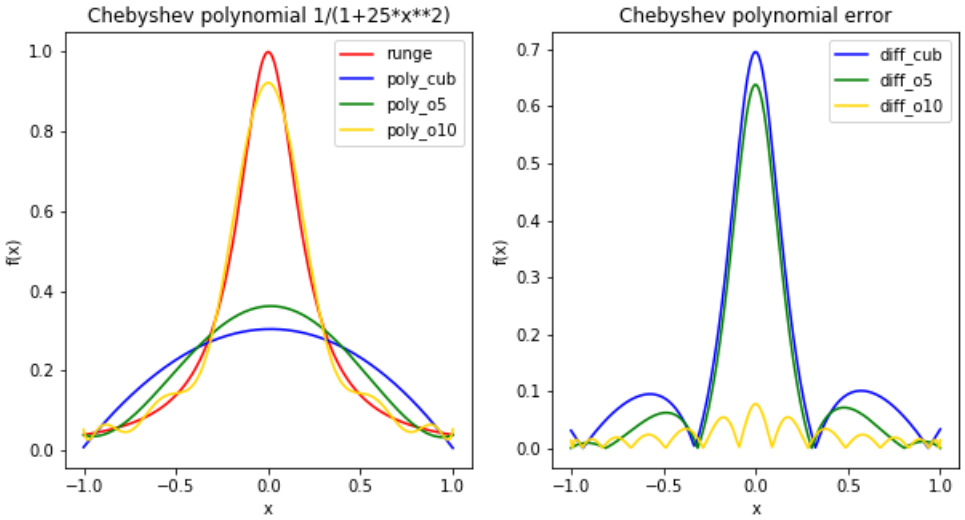
3. Chebyshev interpolation nodes and Chebyshev polynomial of order 3, 5 and 10. How does it compare to the previous results? Report your approximation and errors.

The Chebychev family of polynomial is defined recurcively, such that:

$\Psi = \{\Psi_i(x) = 2x\Psi_{i-1}(x) - \Psi_{i-2}(x) : \forall i \geq 2, \Psi_0(x) = 1, \Psi_0(x) = 1\}$
The procedure to estimate the coefficents is the same as performed previously. I just changed the family of polynomials with the Chebychev family Ψ. This improves the results significantly as you will see in the following graphs.


Chebyshev polynomial exp(1/x)
expo
poly_cub
poly_o5
f(x)
x
Chebyshev polynomial error
diff_cub
diff_o5
f(x)
x

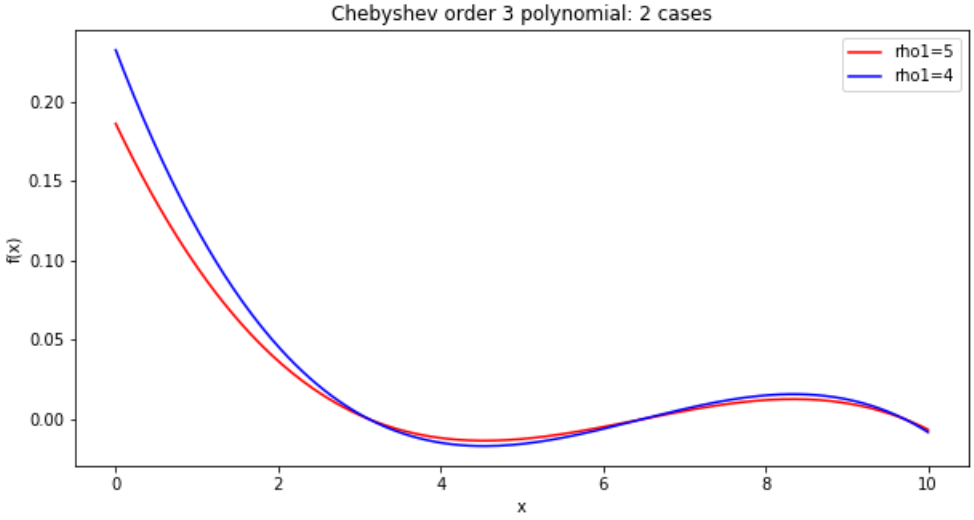

Chebyshev polynomial (x+|x|)/2
ramp
poly_cub
poly_o5
poly_o10
f(x)
x
Chebyshev polynomial error
diff_cub
diff_o5
diff_o10
f(x)
x

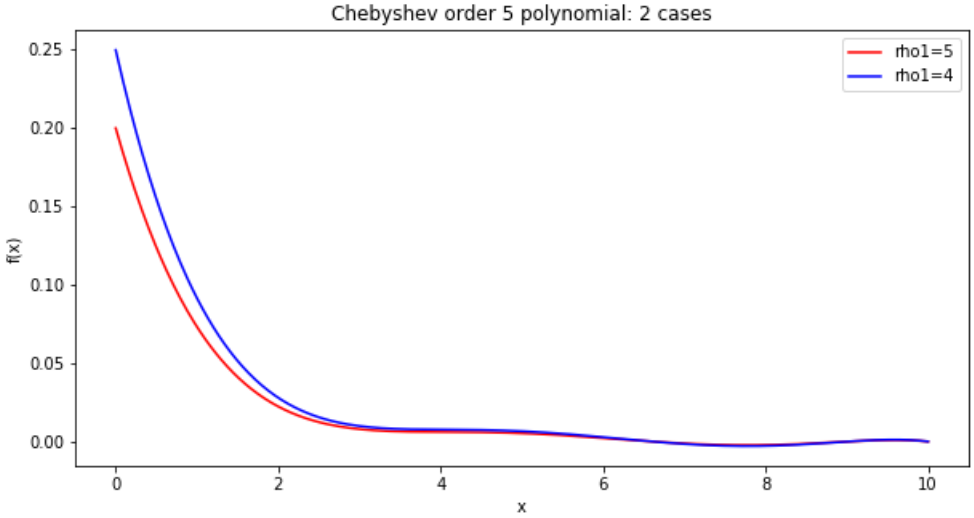


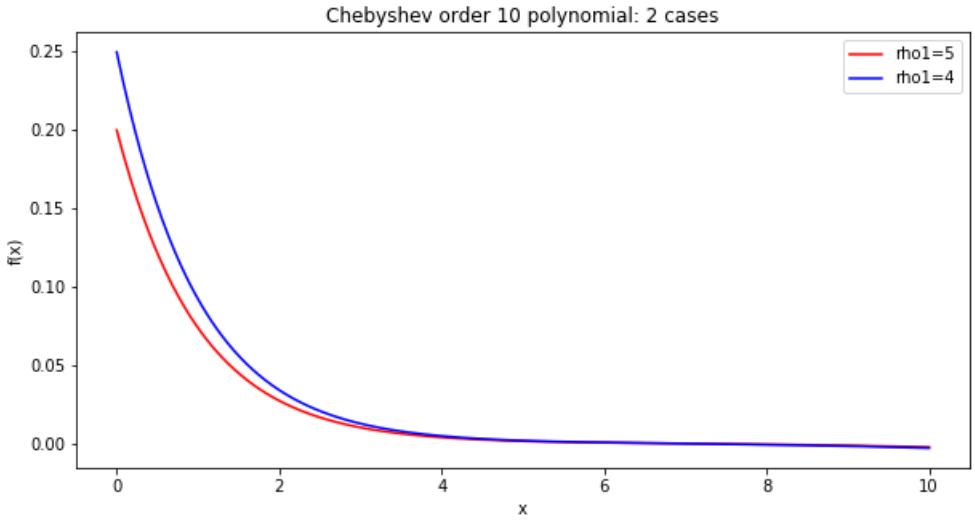
Comments. The higher order polynomial is in general more accurate than the lower order polynomial. The Chebichev nodes capture the right curvature of the function in the edges of the domain. However, it does worst than the evenly separated nodes in center of the interval.
Finally, it is very clear that Chebychev polynomials combined with Chebychev nodes improve significantly the interpolation. Which was expected because the Chebychev family of polynomials form an orthogonal basis.

Exercise 4

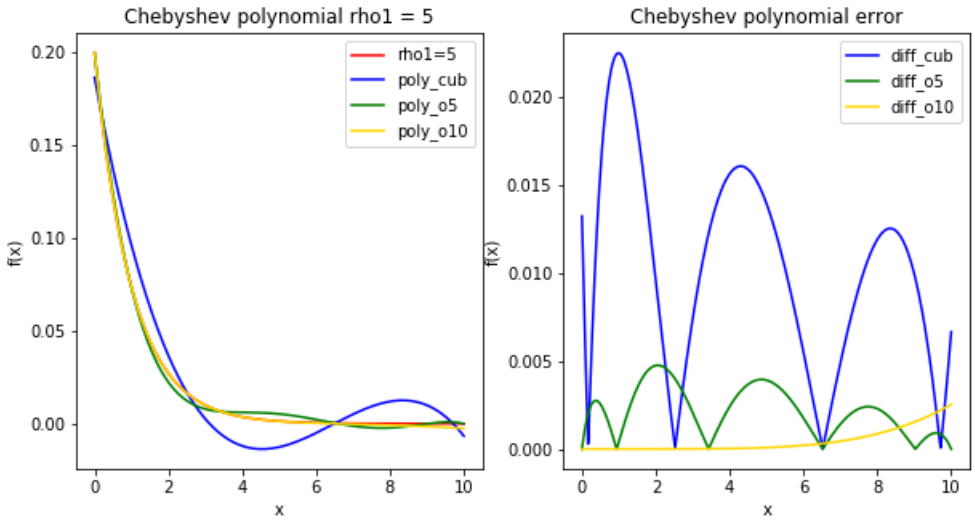
Approximate the following probability function:
$p(x) = \frac{e^{-\alpha x}}{\rho_1 + \rho_2 e^{-\alpha x}}$ for the domain x $\in$ (0,10] using Chebyshev interpolation nodes and Chebyshev polynomial of order 3, 5 and 10. Report your approximation and errors. Do this for two combinations of paramters $\alpha = 1.0$, $\rho_2 = \frac{1}{100}$ and $\rho_1 = 5$. Redo for $\rho_1 = 4$. Plot the results for both combinations in the same graph.

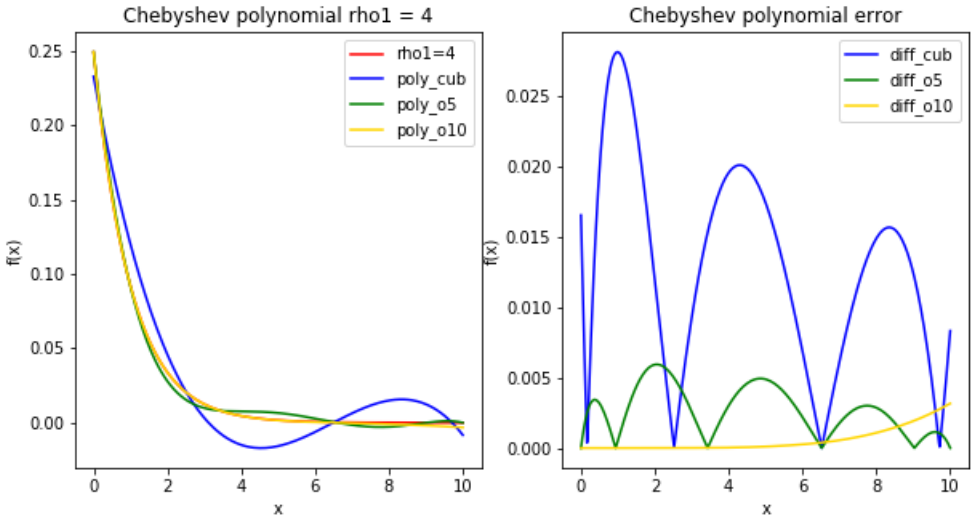

Chebyshev order 3 polynomial: 2 cases
rho1=5
rho1=4
f(x)
x
0.20
0.15
0.10
0.05
0.00
0
2
4
6
8
10


Chebyshev order 5 polynomial: 2 cases
rho1=5
rho1=4
f(x)
x
0.25
0.20
0.15
0.10
0.05
0.00
0
2
4
6
8
10



I added 2 figures representing the error of interpolation of each interpolation.





Question 2

Consider the following CES function $f(k,h) = ((1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}})^{\frac{\sigma}{\sigma-1}}$ where σ is the elasticity of substitution (ES) between capital and labor and α is a relative input share parameter. Set $\alpha = 0.5$, $\sigma = 0.25$, k $\in$ [0,10] and h $\in$ [0,10]. Do the following items:

1. Show that σ is the ES (hint: show this analytically).

$$ES = \frac{dlog(\frac{k}{h})}{dlog(\frac{f_h(k,h)}{f_k(k,h)})}$$

$$f_h(k,h) = \alpha h^{-\frac{1}{\sigma}}((1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}})^{\frac{1}{\sigma-1}}$$

$$f_k(k,h) = (1-\alpha)k^{-\frac{1}{\sigma}}((1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}})^{\frac{1}{\sigma-1}}$$

$$\frac{f_h(k,h)}{f_h(k,h)} = \frac{\alpha}{1-\alpha}(\frac{k}{h})^{\frac{1}{\sigma}} \equiv \theta$$
$$\frac{k}{h} = (\theta\frac{1-\alpha}{\alpha})^{\sigma}$$

$$ES = \frac{dlog((\theta\frac{1-\alpha}{\alpha})^{\sigma})}{dlog(\theta)}$$

$$ES = \frac{d\sigma log(\theta)}{dlog(\theta)} + \frac{d\sigma log(\frac{1-\alpha}{\alpha})}{dlog(\theta)} = \sigma$$

2. Compute labor share for an economy with that CES production function

assuming factor inputs face competitive markets (hint: compute this analytically as well).
$LS = \frac{wh}{f(k,h)}$

From the first order condition of the firm.
$w = f_h(k,h)$

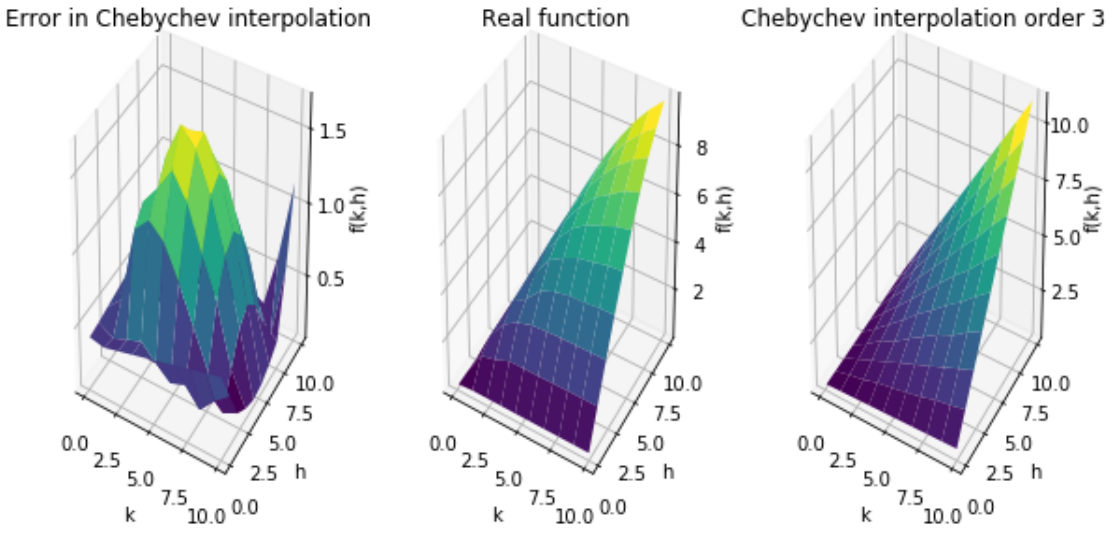
$$LS = \frac{h\alpha h^{-\frac{1}{\sigma}}((1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}})^{\frac{1}{\sigma-1}}}{((1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}})^{\frac{\sigma}{\sigma-1}}}$$

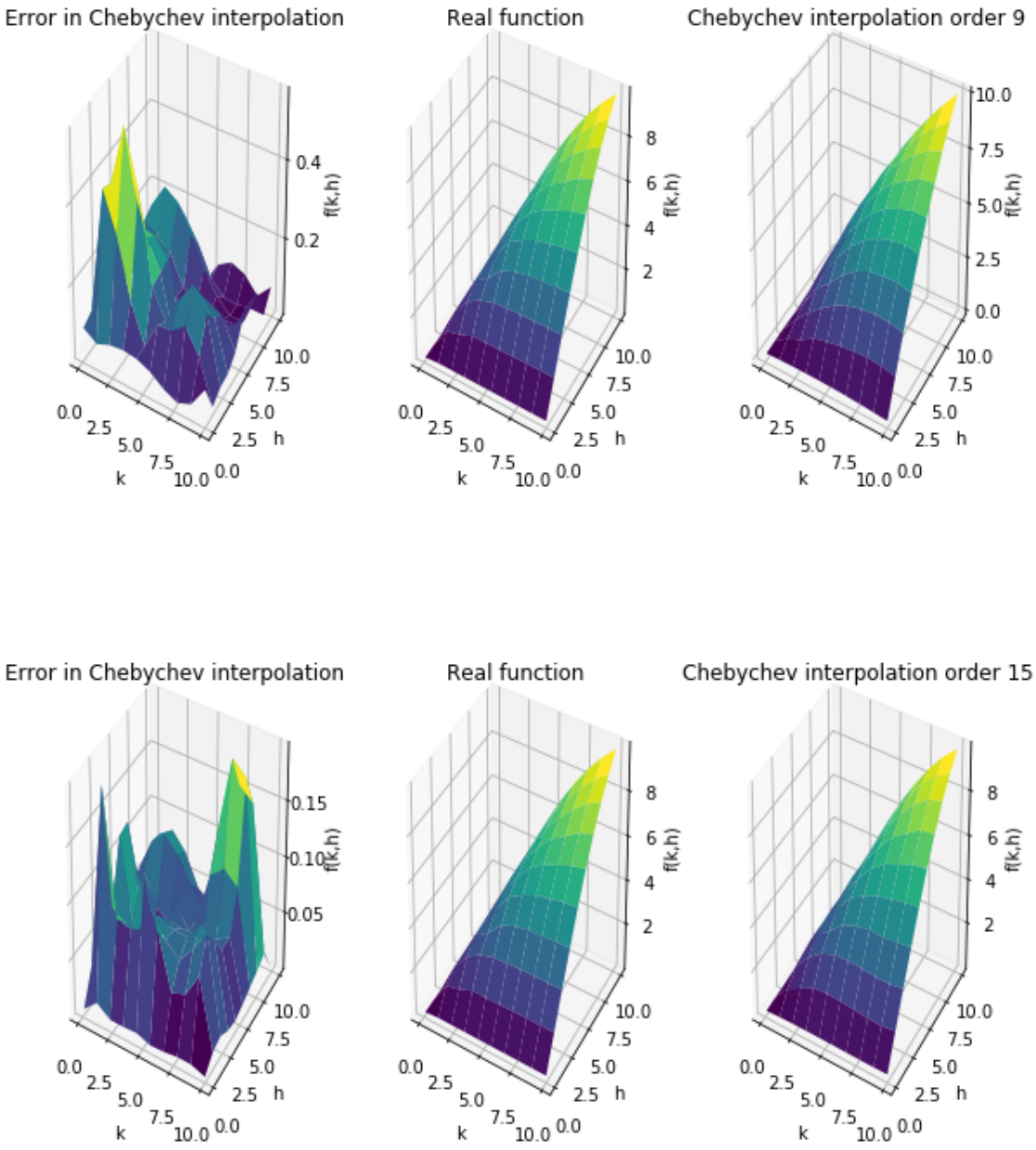
$$LS = \frac{\alpha h^{\frac{\sigma-1}{\sigma}}}{(1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}}}$$

3. Approximate f(k,h) using a 2-dimensional Chebyshev regression algorithm. Fix the number of nodes to be 20 and try Cheby polynomials that go from degree 3 to 15. For each case, plot the exact function and the approximation (vertical axis) in the (k,h) space.

To construct the 2 dimensional Chebychev interpolation I use the following family of polynomials. $F_{2d} = \{\Psi_0(x)\Psi_0(y), \Psi_0(x)\Psi_1(y), ..., \Psi_0(x)\Psi_m(y), ..., \Psi_m(x)\Psi_m(y)\}$ and associate a parameter θ_i to every element of the set. The routine is performed by the function find-theta-cheb-2dim in the file useful-functions.

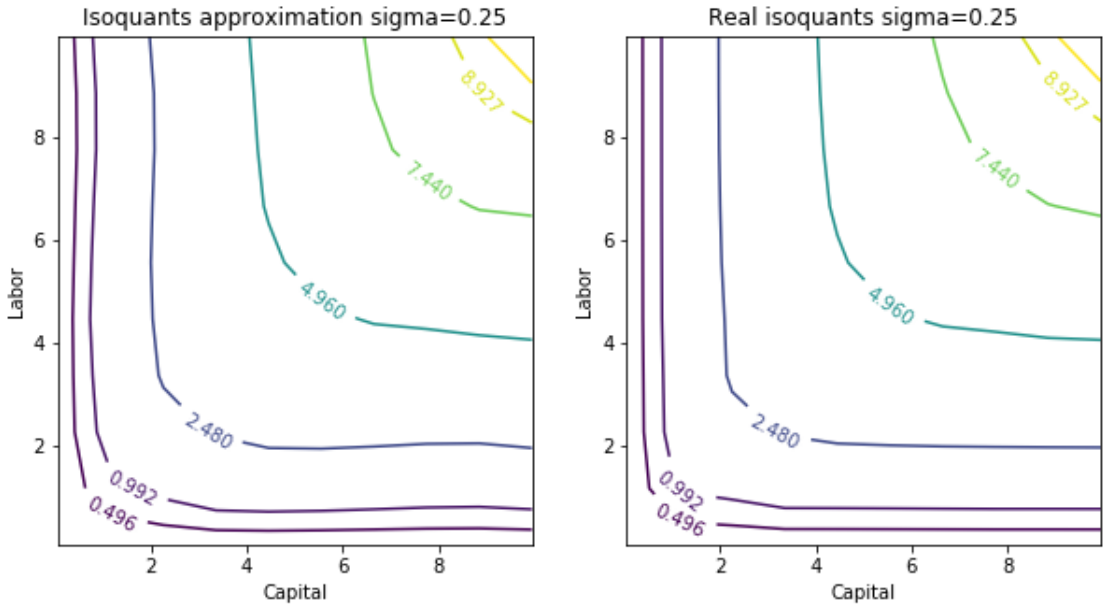
These are the cases of $\sigma = 0.25$

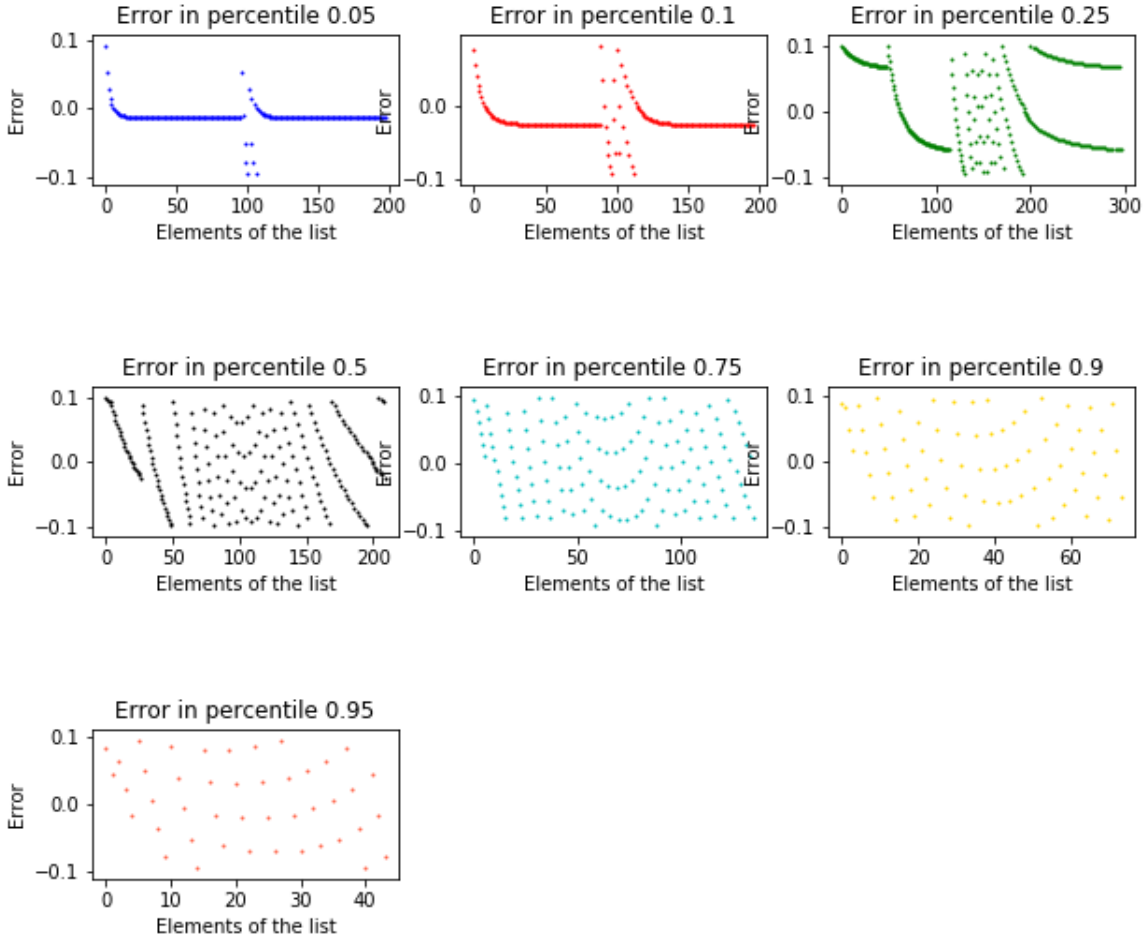



Error in Chebychev interpolation
Real function
Chebychev interpolation order 9
Error in Chebychev interpolation
Real function
Chebychev interpolation order 15
f(k,h)
k
h
0.0
2.5
5.0
7.5
10.0
0.2
0.4
2
4
6
8
0.05
0.10
0.15

4. Plot the exact isoquants associated with the percentiles 5, 10, 25, 50, 75, 90 and 95 of output. Use your approximation to plot the isoquants of the your approximation. Plot the associated errors per each of these isoquant.

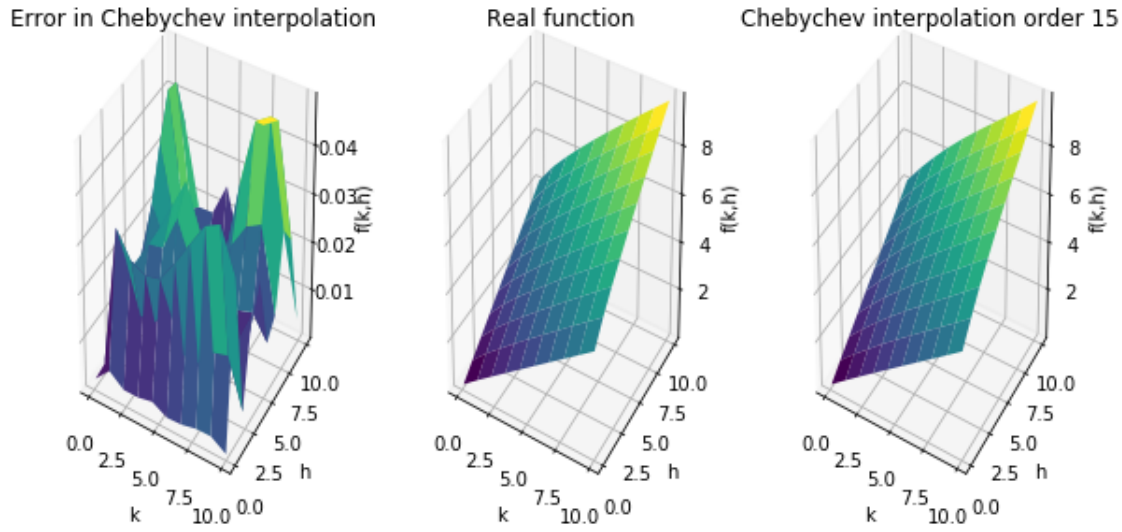
Let me describe the routine that computes the contour line and the errors. Let $\bar{F}$ the level that I fix for the countour line. I create 2 grids one for h another one for k, for every combination of h and k I compute a residual $R = f(k,h) - \bar{F}$, if $R \leq \epsilon$ then we store k and h and nothing happens otherwise. let $C_o = \{(k,h) \in \mathbf{R}^2_+ : R \leq \epsilon\}$.
For all $(k,h) \in C_o$ I compute $\widetilde{f}(k,h)$. The errors in the following graphs are actually $\widetilde{f}(k,h) - \bar{F}$. The x-axis refered to as elements of the list represents the set of elements contained in C_o. It is important to notice that there is no relevant order to the list since the elements are vectors. I found relevent to sort them in the order the loop detected them. The routine I constructed is called Iso-quants and you can find it in useful-functions.

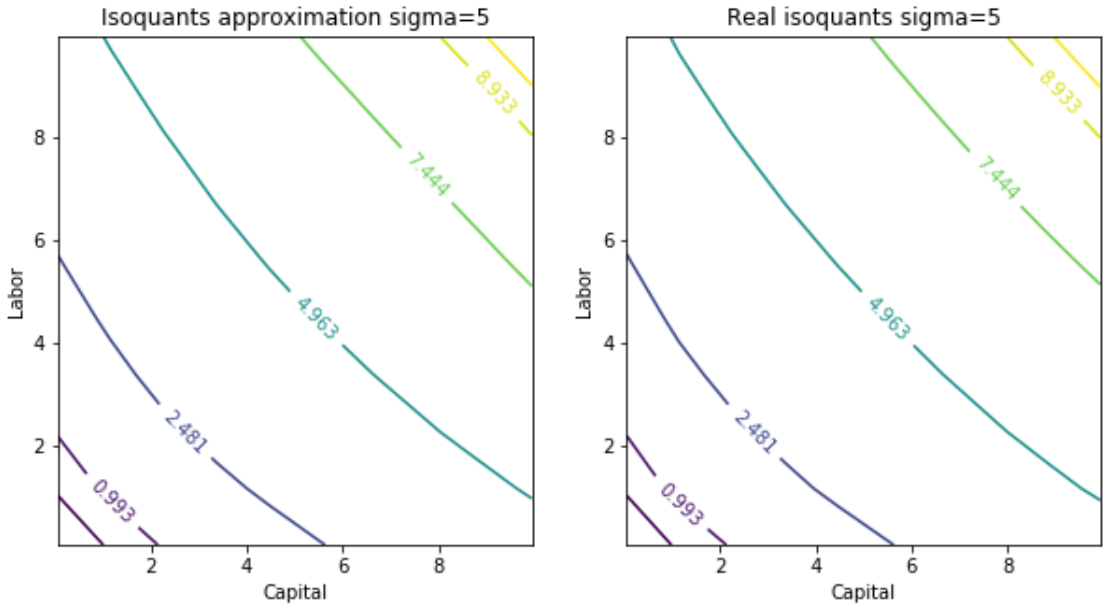


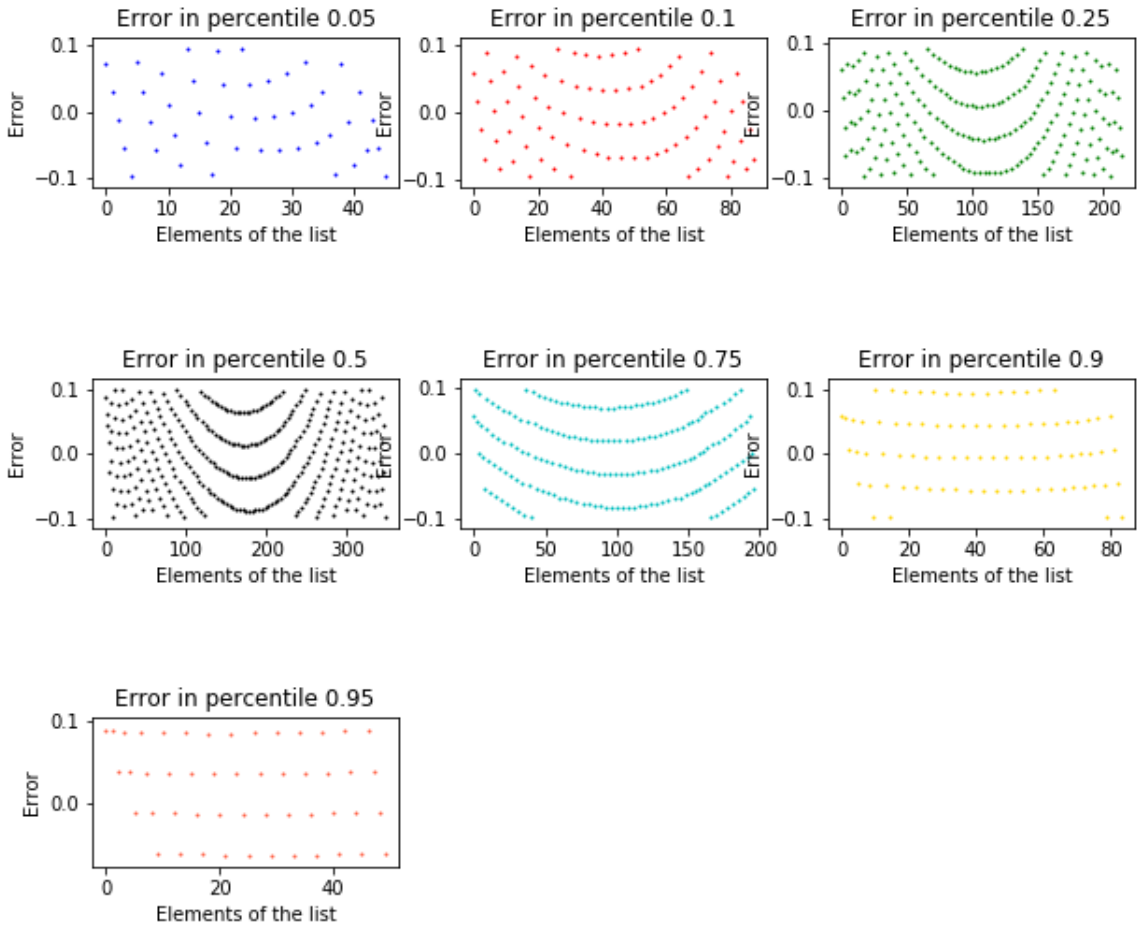


5. For each case, show the associated approximation errors (vertical axis) in the (k,h) space.
6. Redo using $\sigma = 5.00$ and $\sigma = 10$

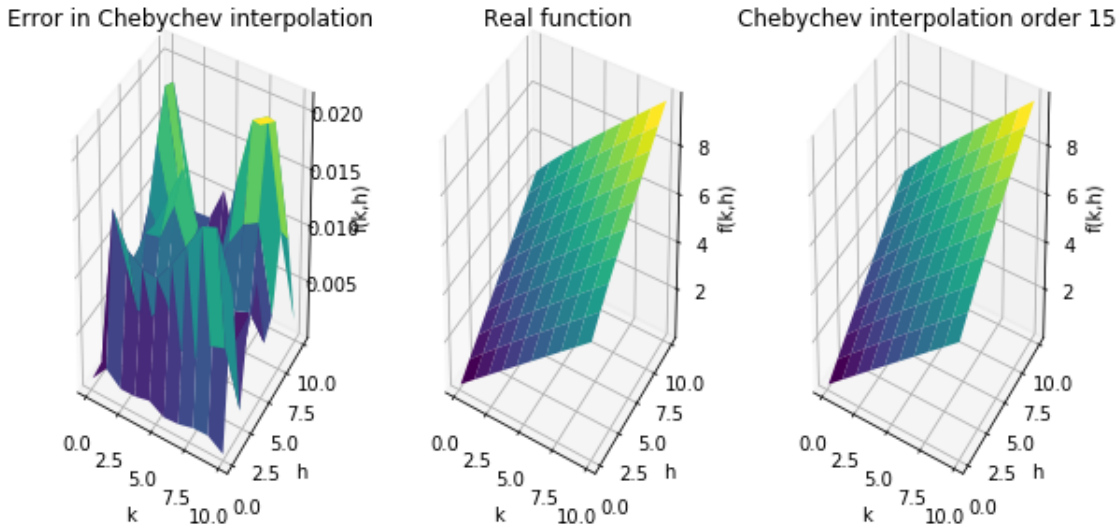
These are the cases of $\sigma = 5$

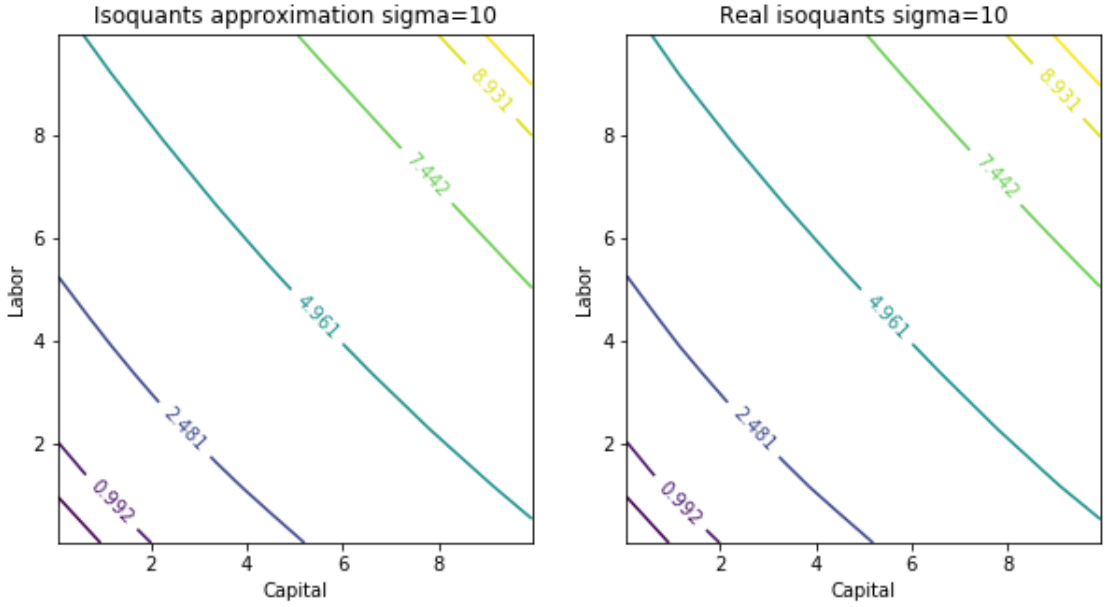

Error in Chebychev interpolation
Real function
Chebychev interpolation order 15
f(k,h)
k
h
0.0
2.5
5.0
7.5
10.0
0.01
0.02
0.03
0.04
2
4
6
8

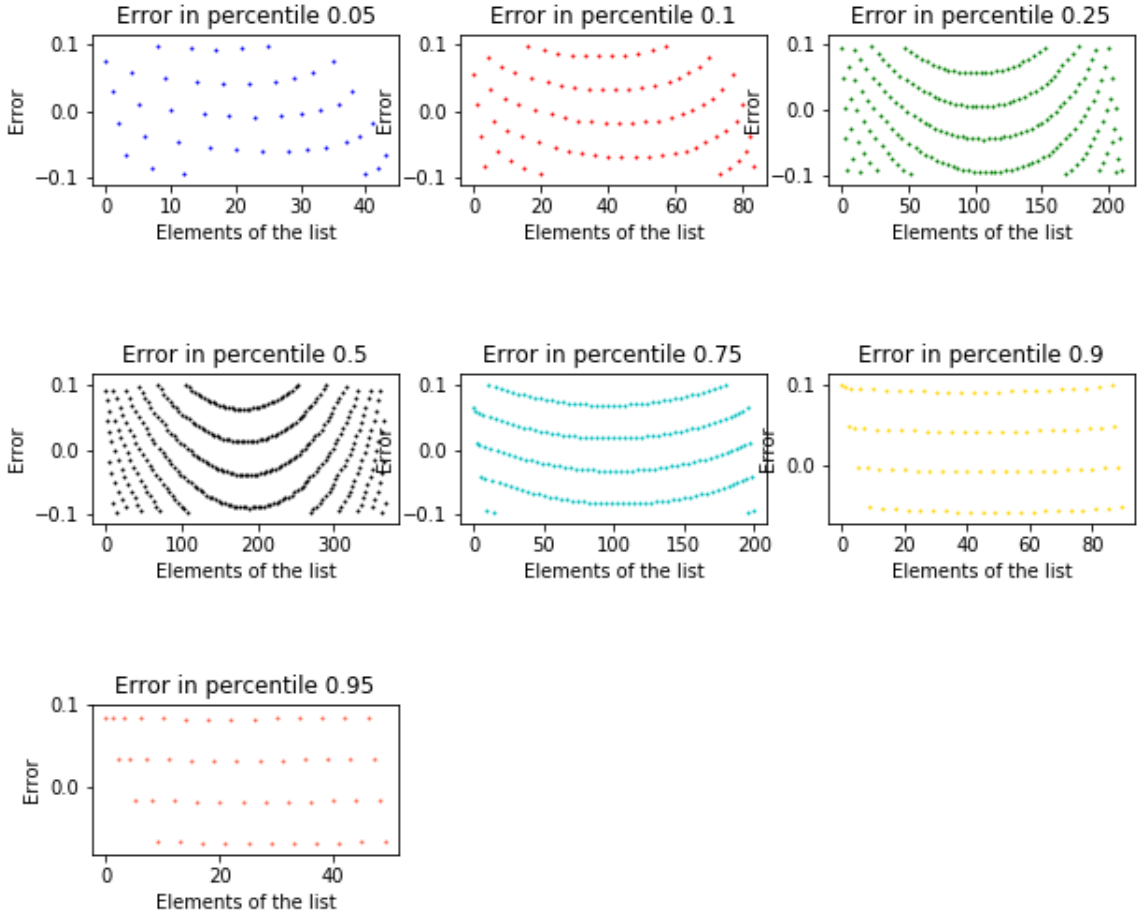

Isoquants approximation sigma=5
Real isoquants sigma=5
Labor
Capital
0.993
2.481
4.963
7.444
8.933
2
4
6
8



comment. I believe there is a typo in the question you asked to do it for $\sigma = 1$ but the function is not defined for that value.
These are the cases of $\sigma = 10$


Error in Chebychev interpolation
Real function
Chebychev interpolation order 15
f(k,h)
k
h
0.0
2.5
5.0
7.5
10.0
0.005
0.010
0.015
0.020
2
4
6
8


Isoquants approximation sigma=10
Real isoquants sigma=10
Labor
Capital
8.931
7.442
4.961
2.481
0.992
2
4
6
8


Error in percentile 0.05
Error in percentile 0.1
Error in percentile 0.25
Error in percentile 0.5
Error in percentile 0.75
Error in percentile 0.9
Error in percentile 0.95
Error
Elements of the list