Problem Set 1

Quantitative Macro

Pau Belda-i-Tortosa

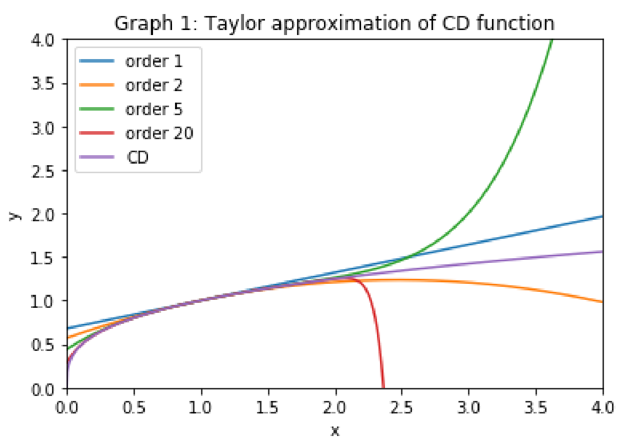
September 2018

1 Function Approximation: Univariate

1.1 Taylor approximation of a Cobb-Douglas function

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.

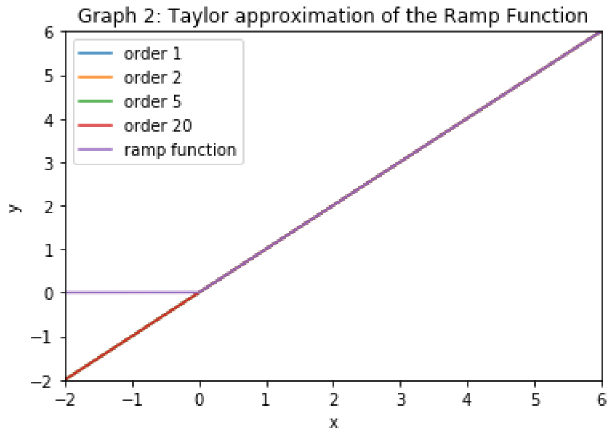
The Taylor approximation is a local method of interpolation. It represents a function as an infinite sum of terms that are calculated using the values of the function' derivatives at a single point. The Taylor expansion of the function $f(x) = x^{0.321}$ at $\bar{x} = 1$ got for different approximation orders (1,2,5,20) is plotted in the graph 1.



The approximation error (i.e., the gap between the exact value and a specific approximation) is minimum around $x = 1$ for all order approximations. However, it increases strongly for $x > 1$, the higher the approximation order the higher the error. This is because the function has a singular point at x=0 and since the approximation is around $\bar{x} = 1$, any point situated at a distance more than 1 w.r.t. the singular point is out of the radius of convergence.

1.2 Taylor approximation of the Ramp Function

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



The Ramp function (the analytical definition we use here is just one of the possible definitions) is a unary real function that looks like a ramp. The Taylor Series approximations fit perfectly the actual value over the non-negative domain. However, the image of the ramp function takes 0 for negative inputs; since all Taylor approximations become linear approximations (no matter the approximation order) all them are not able to capture the kink at $x = 0$ (singular point).

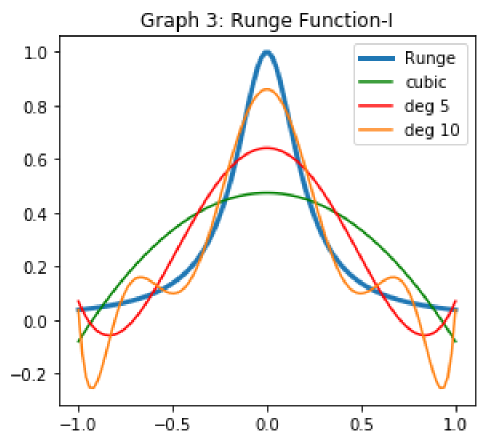
1.3 Some global methods

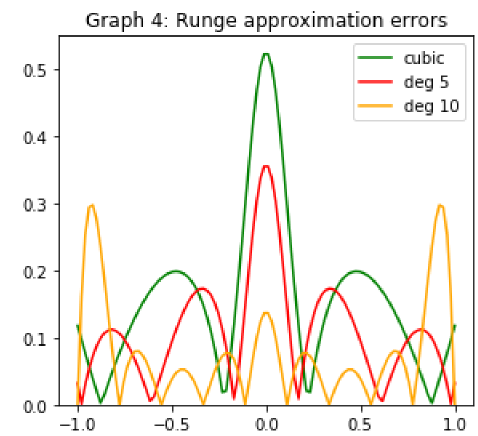
Approximate these three functions: $e^{\frac{1}{x}}$, the runge function $\frac{1}{1+25x^2}$, and the ramp function $f(x) = \frac{x+|x|}{2}$ for the domain $x \in [-1, 1]$ with:

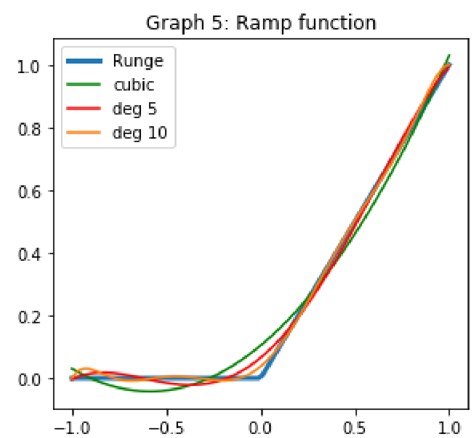
1.3.1. Evenly spaced interpolation nodes and a cubic polynomial. Redo with monomials of order 5 and 10.

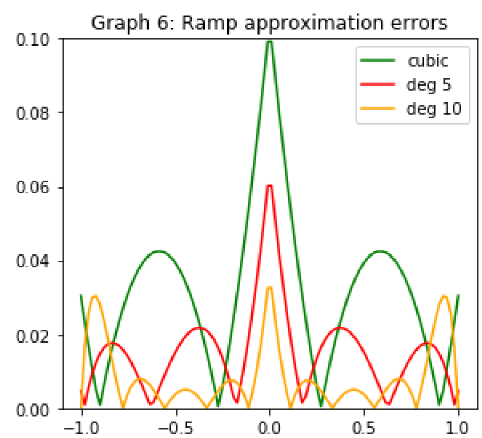
The reasoning we use is quite simple. We approximate $f(x)$ through $\tilde{f}(x)$, where the approximation function consists of a sum of monomials (as a basis functions) weighted somehow (with some coefficients). Then, we end up with a system of linear equations of the type Ax=b, where A is a nxn matrix of n monomials, x is the nx1 vector of coefficients and b is a nx1 vector containing the value of the function evaluated at the n nodes. Notice that for the system being identified we need number of coefficients$\leq$ number of nodes. In this section, we use equally spaced nodes over the domain of the function. Once we have the coefficients, we build a polynomial of type $a + bx + cx^2 + ...$ (up to order n) and evaluate it over the whole domain. This is our approximation. The graph 3 to 8 report the representation of the functions and their approximation using polynomials of order 3, 5 and 10.

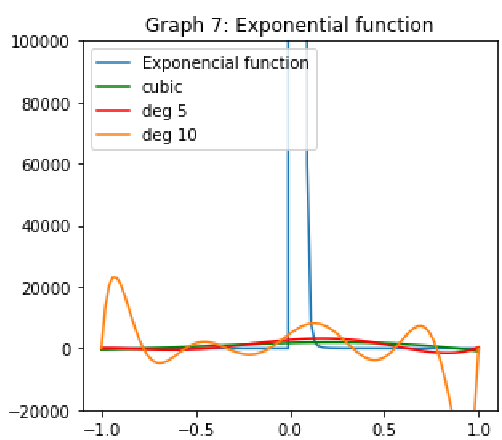
We include a representation of the approximation error. Some comments about it. First, we clearly observe the Runge's phenomenon: the magnitude of the oscillations increase with polynomials of higher order. However, the higher order polynomial fits better in the area close to the stationary point and worse at the extremes. Something similar happens with the Ramp function: polynomials of lower order fit worse the kink at the singular point $x = 0$. Second, polynomials are not capable of catch well the discontinuity of the exponential function at $x = 0$ (no matter the order of the polynomial). All these problems suggest that we should try other interpolation methods.

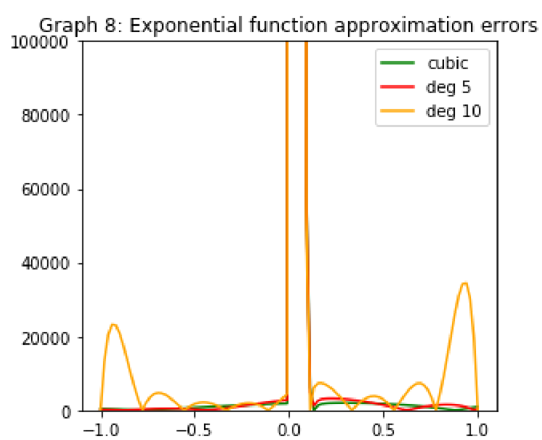






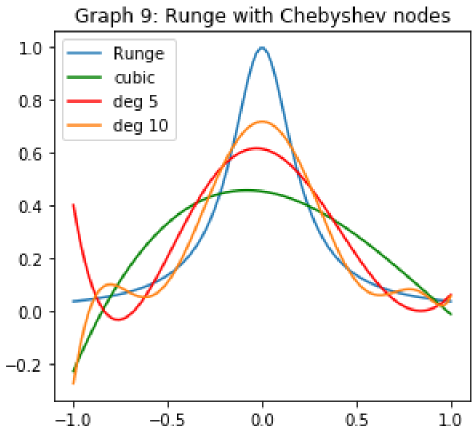


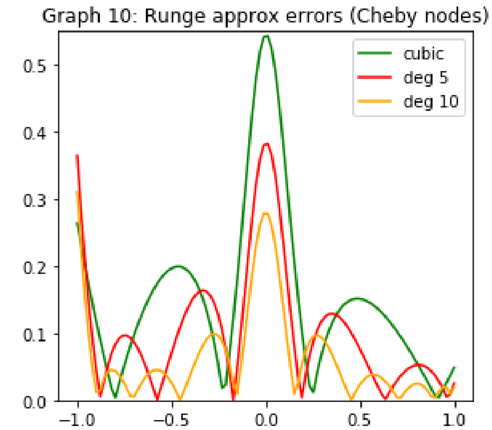


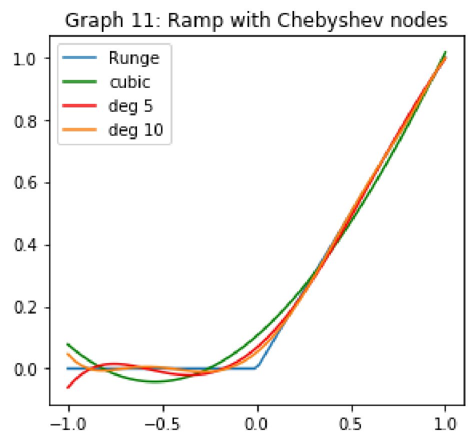


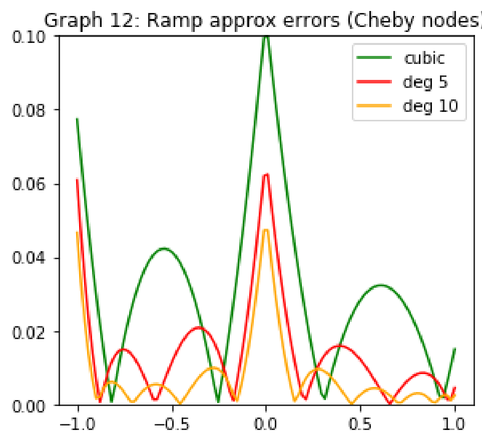
1.3.2. Chebyshev interpolation nodes and a cubic polynomial. Redo with monomials of order 5 and 10.

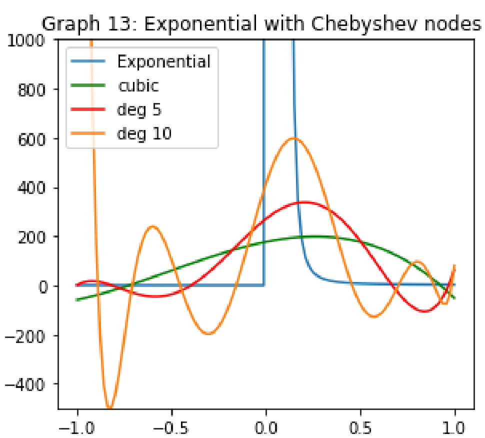
The algorithm is just as before, with the difference that now we substitute the equally spaced nodes by the Chebyshev nodes (see the code). Chebyshev nodes are more closely spaced near the limit of the domain and less near the center. This way we correct one of the problems we detected before: the equally spaced nodes leads to a constant that increases quickly when n increases, generating the increasing oscillations.

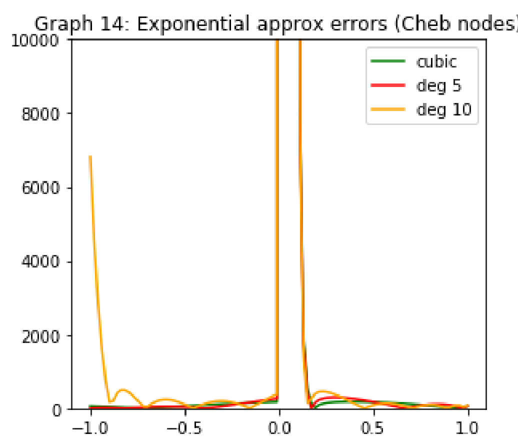










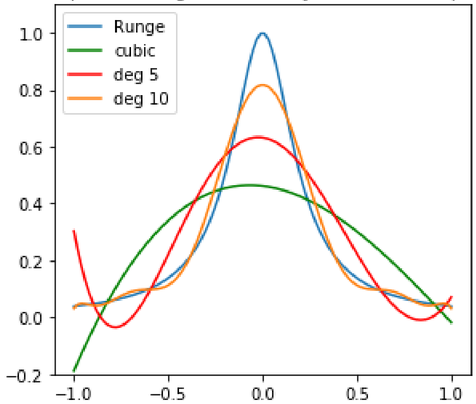


Now, the polynomial of order 10 is capable to replicate better the stationary point of the Runge function and the singular point of the Ramp function without showing more oscillations in the extremes. Regarding the exponential function, we change the y-axis to zoom in and see what happens more accurately. As before, none of the polynomials are able to capture the discontinuity at $x = 0$.

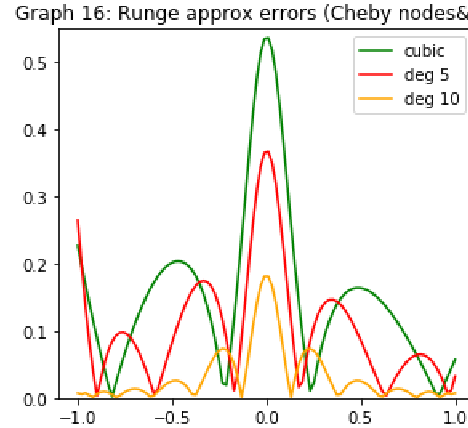
1.3.3. Chebyshev interpolation nodes and Chebyshev polynomial of order 3, 5 and 10.

Now we use the Chebyshev nodes plus Chebyshev polynomials as a basis functions (instead of monomials). Chebyshev polynomials offer us a set of advantages (more robust, smooth and bounded between $x \in [-1, 1]$, errors bounded according to several theorems, etc.). Graphs 15 to 20 show us the approximation using this method. The difference with previous method is clear: now higher order approximations fit much better both the Runge and the Ramp function, reducing the errors in the center and reducing the oscillations at the extremes. However, it does not perform better the exponential function approximation.

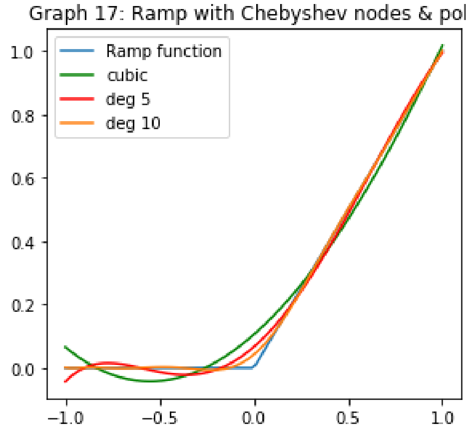
Graph 15: Runge with Chebyshev nodes & poly



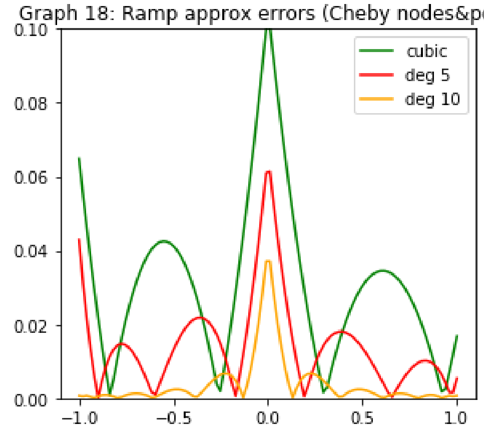
Graph 16: Runge approx errors (Cheby nodes&poly)



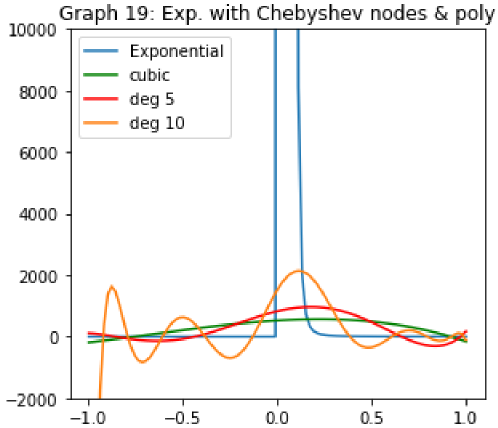
Graph 17: Ramp with Chebyshev nodes & poly



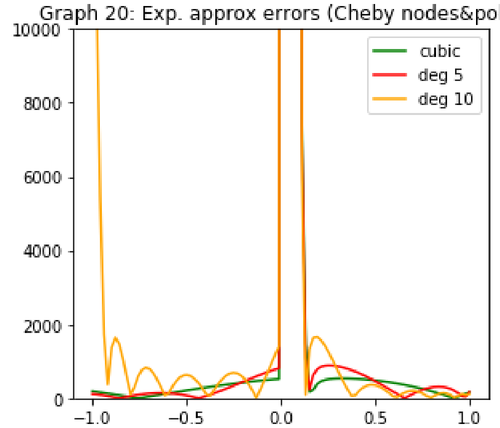
Graph 18: Ramp approx errors (Cheby nodes&poly)



Graph 19: Exp. with Chebyshev nodes & poly



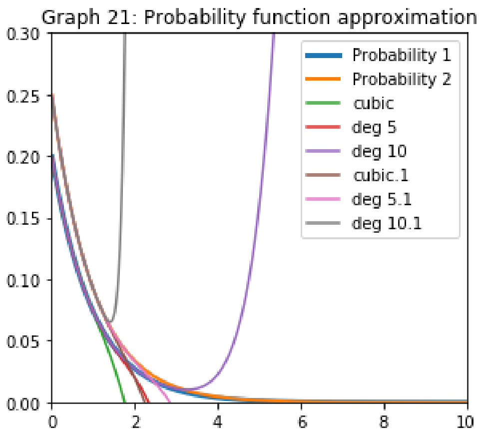
Graph 20: Exp. approx errors (Cheby nodes&poly)

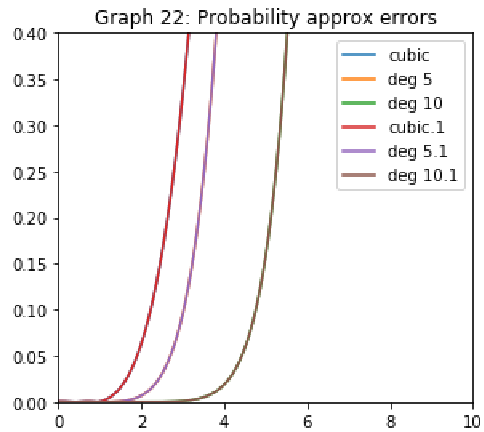


1.4 Probability function approximation

Approximate the 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* $\alpha = 1$, $\rho_1 = 5$ *and* $\rho_2 = 0.01$. *Redo for* $\rho_1 = 4$.

We use the previous method to approximate a probability function now. What we see is that from 0 to 1 is well approximated, but from then on the approximation errors grow exponentially no matter what combination of parameters and what order of Chebyshev polynomial we consider.





2 Function approximation: Multivariate

2.1 Elasticity of substitution

Consider the following CES production function:

$$f(k,h) = \left[(1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}}\right]^{\frac{\sigma}{\sigma-1}} \tag{1}$$

The concept of elasticity of substitution is a measure of the ease with which some input can be substituted for others. More technically, it measures the change in the capital-labor ratio relative to the change in their marginal productivity. With competitive markets, it stands for the change in the k/h ratio caused by the change in the r/w ratio. Mathematically it can be expressed as follows:

$$ES = \frac{\partial log\left[\frac{k}{h}\right]}{\partial log\left[\frac{F_h}{F_k}\right]} \tag{2}$$

First, compute the marginal productivities:

$$f_h = \left[(1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}}\right]^{\frac{1}{\sigma}} \alpha h^{-\frac{1}{\sigma}} \tag{3}$$

$$f_k = \left[(1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}}\right]^{\frac{1}{\sigma}} (1-\alpha)k^{-\frac{1}{\sigma}} \tag{4}$$

Now, we proceed as follows:

$$log\left[\frac{F_h}{F_k}\right] = log\frac{\alpha h^{-\frac{1}{\sigma}}}{(1-\alpha)k^{-\frac{1}{\sigma}}} = log\left[\frac{\alpha}{1-\alpha}\right] + \frac{1}{\sigma}log\left[\frac{k}{h}\right] \tag{5}$$

Rearrange the previous equation to get:

$$log\left[\frac{k}{h}\right] = \sigma log\left[\frac{F_h}{F_k}\right] - \alpha log\left[\frac{F_h}{F_k}\right] \tag{6}$$

Finally, take the derivative of the RHS w.r.t. the log of the marginal productivities ratio:

$$\frac{\partial log\left[\frac{k}{h}\right]}{\partial log\left[\frac{F_h}{F_k}\right]} = \sigma = ES \tag{7}$$

2.2 Wage Share

Assume that we have perfect markets, so that the wage rate is equal to the marginal productivity of labor. In this virtual world, the wage share (call it β) can be computed as follows:

$$\beta \equiv \frac{wh}{Y} = \frac{F_h h}{f(k,h)} \tag{8}$$

Plugging our previous results in, we get the labor share corresponding to a CES economy with perfect markets:

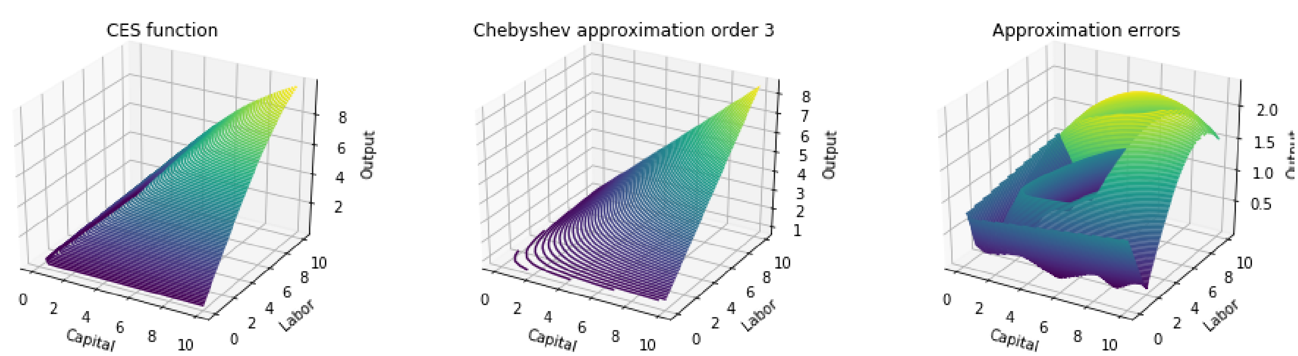
$$\beta = \frac{\left[(1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}}\right]^{\frac{1}{\sigma}}}{\left[(1-\alpha)k^{\frac{\sigma-1}{\sigma}} + \alpha h^{\frac{\sigma-1}{\sigma}}\right]^{\frac{\sigma}{\sigma-1}}} \alpha h^{\frac{\sigma-1}{\sigma}} \tag{9}$$

2.3 2D Chebyshev regression algorithm

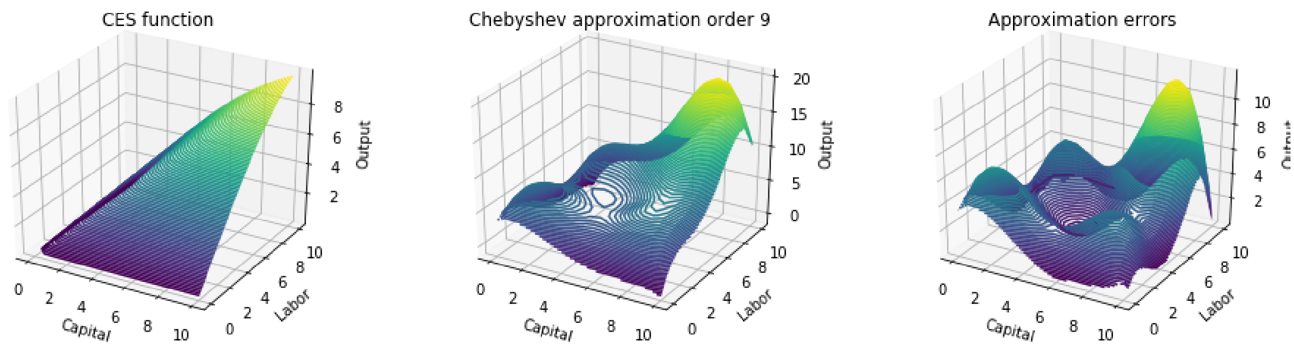
Approximate f(k,h) using a 2-dimensional Chebyshev regression algorithm. Fix the number of nodes to be 20 and try Cheby polynomials that go from degree 3 to 15. For each case, show the associated approximation errors (vertical axis) in the (k, h) space.

For this exercise we use the Chebyshev algorithm for R^2 described in Judd notes. See the code for the procedure and the reference. We show here a subset of the polynomials asked (order 3, 9, 15). This subset gives us a substantiated idea of what happens with the approximation when we increase the order of the polynomials.

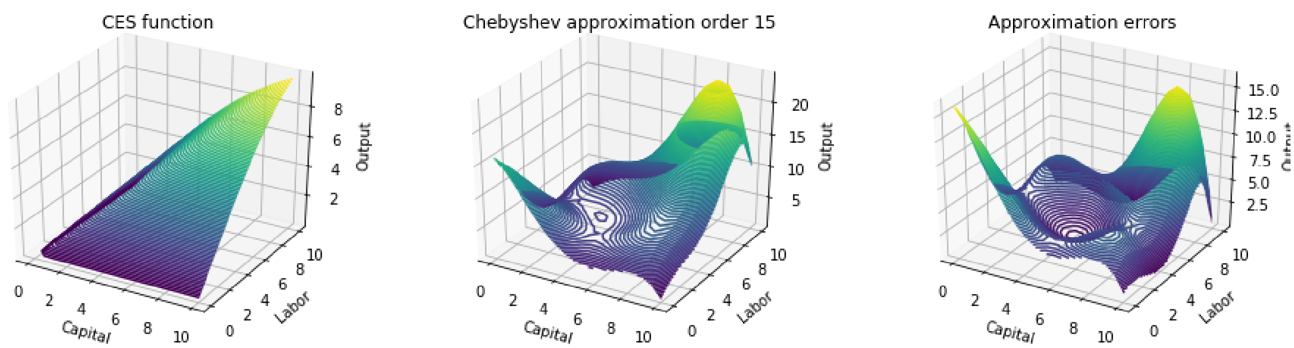
Graph 23: Chebyshev approx order 3 for σ=0.25



Graph 24: Chebyshev approx order 9 for σ=0.25



Graph 25: Chebyshev approx order 15 for σ=0.25

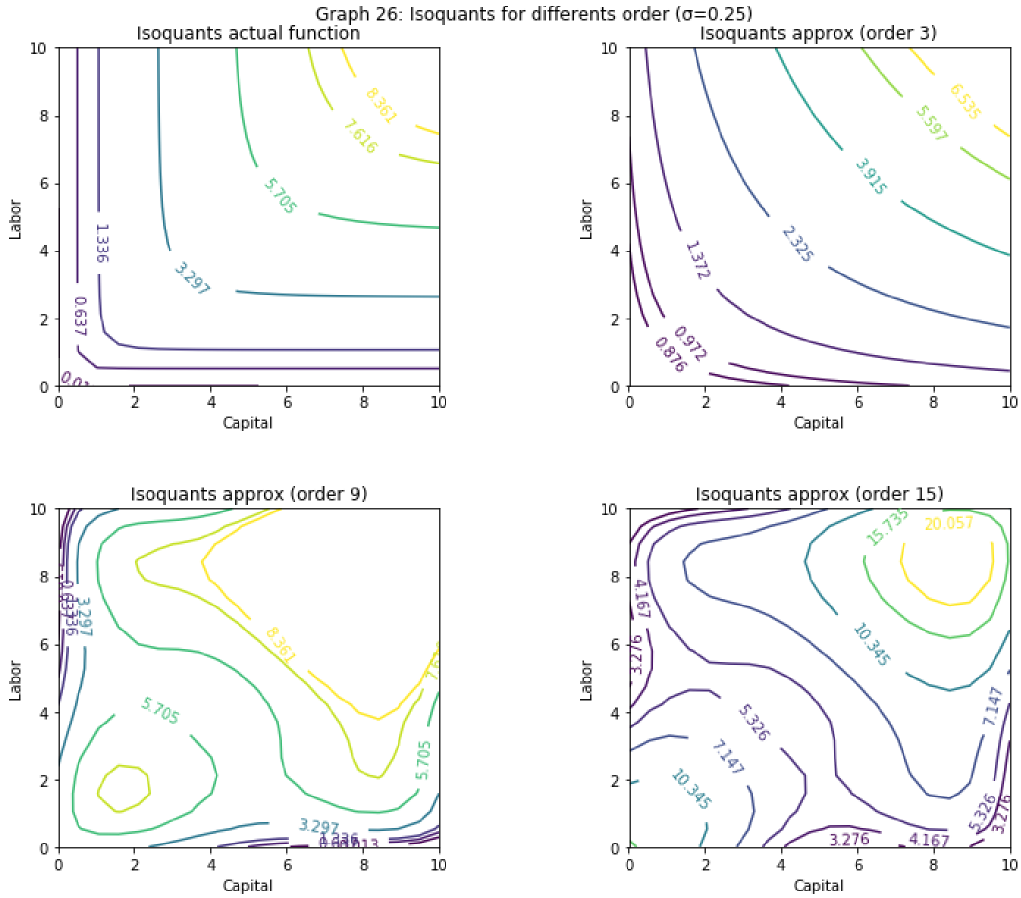


The main result is that our approximation error strongly increases with the degree of the polynomials (for the approximation of order 15 the error is more than 100%!). It suggests that either we made some mistakes in the algorithm or that higher order Cheby polynomials are not suitable for deal with the CES function (since it is bounded above).

2.4 Isoquants

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 your approximation.

Following the instructions, we have plotted the isoquants associated with these percentiles of output. We are able to plot the isoquants associated to different polynomials in the same plot, so that the error becomes the distance between the actual and approximated isoquants. However, it becomes a visual mess! Instead, we can visually evaluate this errors: the 50% of the actual output is below 5.705 units whereas there is only 3.915 units according to the approximated output. In this case, the approximation error would be of 1.790 units. Etc. The main result is same as before: the approximation error increases hugely with higher order approximations.



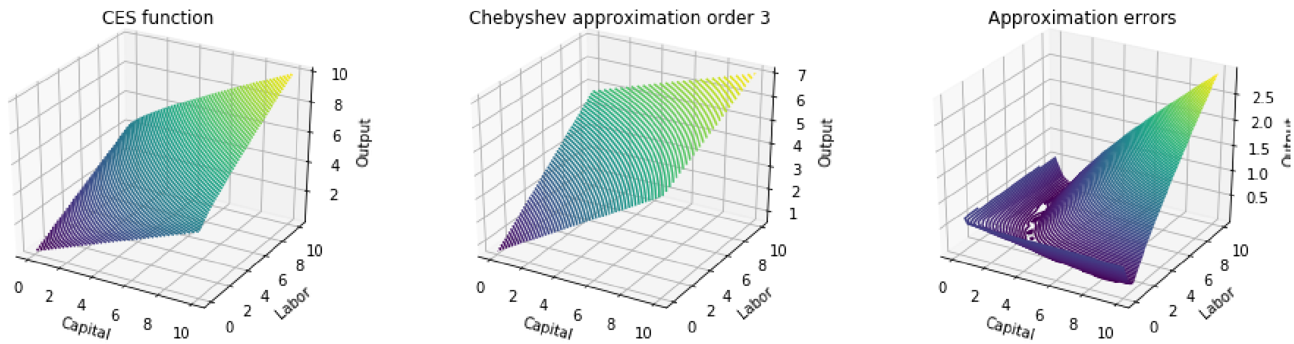
2.5 Change of parameters

2.5.1 $\sigma = 5.00$

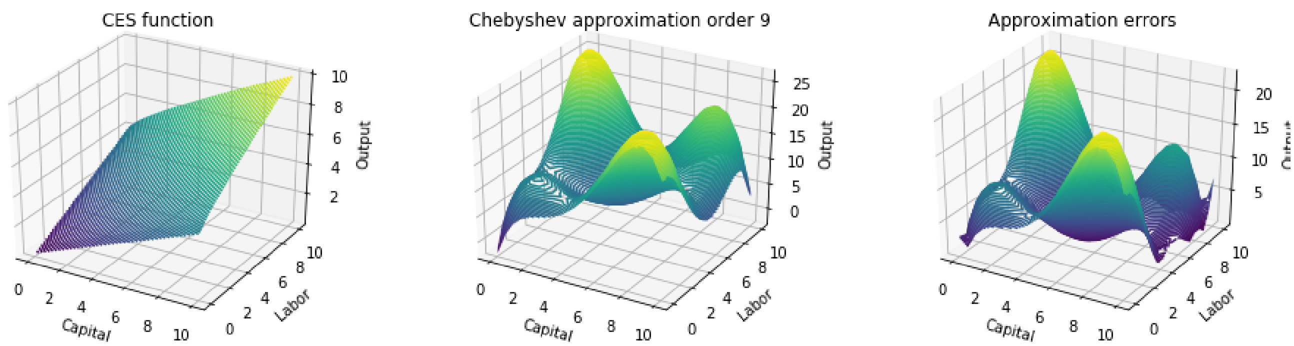
We repeat all the procedures just changing a new σ. Remind that the σ of the CES function stands for the elasticity of substitution. This way, the higher the σ the higher the possibilities of changes labor by capital to get the same output. In this section the elasticity is extremely high; we are in kind of a Matrix world where machines can do all the workers jobs much better.

The main result now is that the function is approximated much better for lower levels of labor in the case of the order 3 approximation; in contrast, the better approximation for higher order polynomials is at higher levels of capital and labor. However, in general, higher order approximation reach much higher errors.

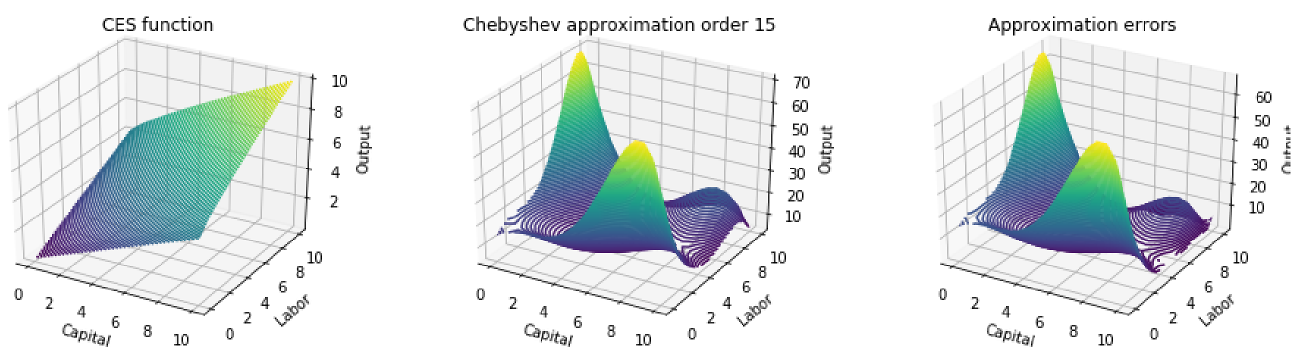
Graph 27: Chebyshev approx order 3 for σ=5



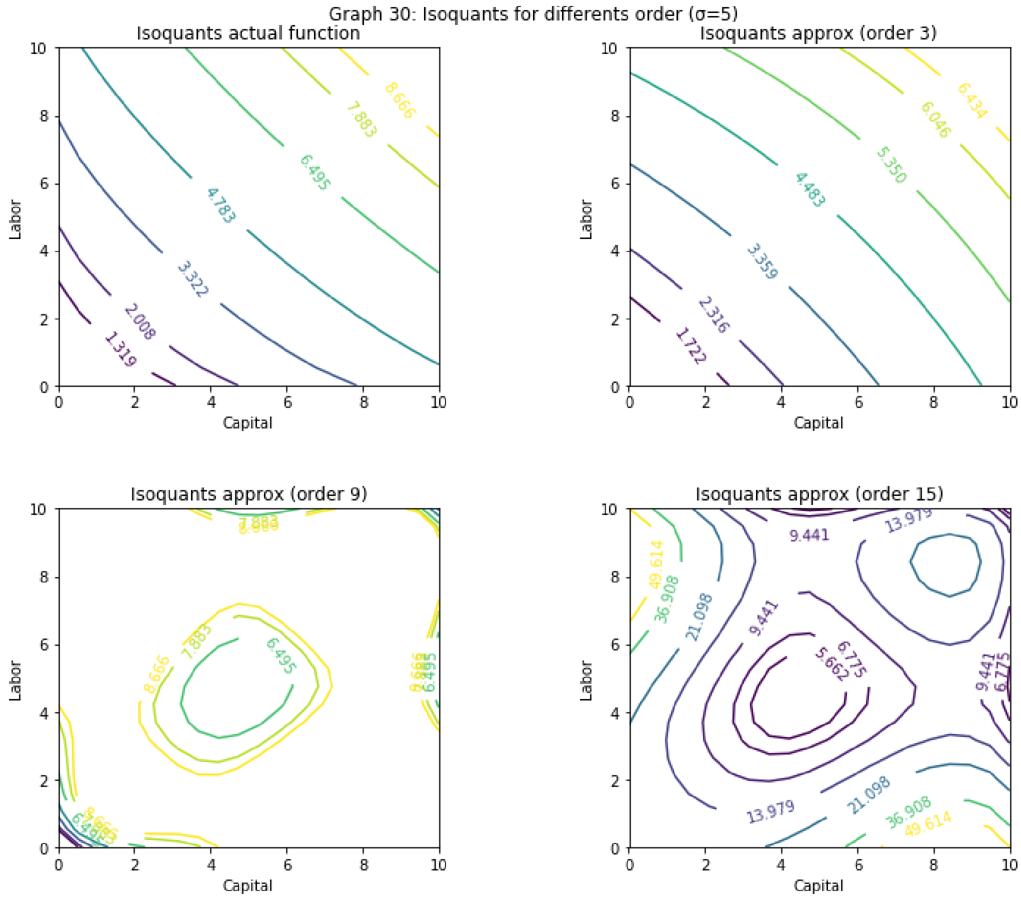
Graph 29: Chebyshev approx order 9 for σ=5



Graph 28: Chebyshev approx order 15 for σ=5



Now, take a look at the isoquants. Again, we can guess the errors visually. For instance, we realize that the approx of order 3 did relatively well: the approximation error is only of 300 units of output (over 4.783, that is, only about 6%) at the percentile 50.



2.5.2 $\sigma = 1.00$

For $\sigma = 1.00$ we would be in a world of perfect one-to-one substitution of workers and machines. However, the function is not well defined (the exponent takes $\frac{1}{0} = \infty$)!