# IE523: Financial Computing
# Fall, 2019
## Programming Assignment 10: Influence of Memoization on Pricing European-Options using Linear Programs (LPs)
## Due Date: 24 November 2019



In the previous programming assignment you priced European- and American-Options using a memoized-Trinomial Models. You would have clearly seen the benefits, in terms of computation-time, of memoization. In this programming exercise, you are going to explore if memoization has similar benefits when it comes to pricing European-Options using Linear-Programs (LPs).

Section 8 of Lesson 6 of my notes contains the necessary theoretical background for pricing European Options using LPs. I want you to take the C++ code European Option Pricing via LPs.cpp on Compass, which prices an European-Option using LPs, and modify it to include memoization.

More specifically, in the un-memoized version of the code in European Option Pricing via LPs.cpp on Compass, we are being very inefficient by adding the same set of *self-financing-constraints* when we repeatedly visit the same Binomial Lattice State $(k, i)$ many times in course of the recursive functions create_LP_for_european_put_option(int k, int i) and *create_LP_for_european_call_option(int k, int i)*. In the memoized-version, which you are going to write as a part of this assignment, you will do the needful to ensure the constraints are not unnecessarily repeated when a state $(k, i)$ has been visited earlier.

I want you to compare the running times of your (memoized) code against European Option Pricing via LPs.cpp on Compass. From the programming assignment on Repeated-Squaring, you know how to measure the time it takes to compute the price of the options (i.e. you have to insert the appropriate code to do this).

The code you write will produce an output that looks like what is shown in figure 1, I have intentionally blanked-out the runtime information to make you think about what you are getting in your experiments. To keep the running-times reasonable, I suggest you try Number of Divisions to be in the set $\{10, \cdots, 15\}$. Do you see any benefits to memoization here? Explain your results (i.e. provide a justification for whatever you observe in course of your experiments).

What I need from you:

1. Upload the C++ code on Compass, and
2. Upload a short explanation for what you see in your experiments.

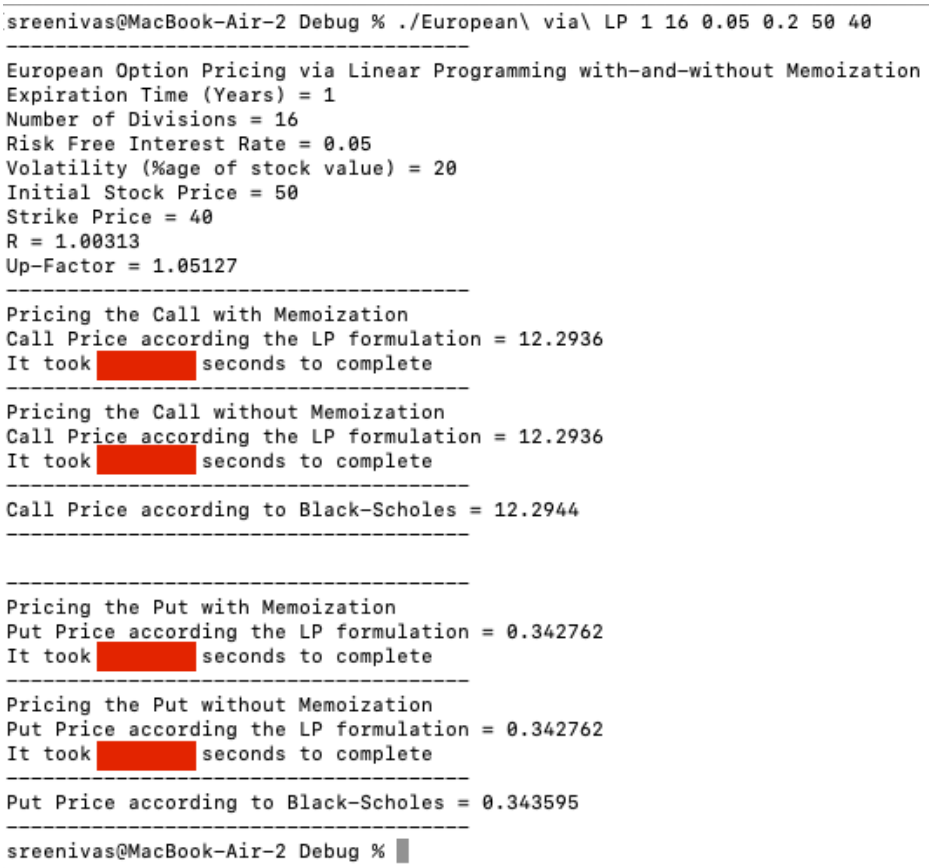


Figure 1: Comparing the Runtimes with-and-without memoization for an algorithm that prices European-Options using Linear Programs.