Rotman School of Management
UNIVERSITY OF TORONTO

Rotman

Group Number:	1003463995 & 1003248844
Assignment Title:	Assignment 1
Course Code:	RSM 2307HS.2018-0103
Section #:	1 2 3 4 5 AM PM
Course Name:	Advanced Derivatives
Professor Name:	Redouane Elkamhi

<u>In submitting this **group** work for grading, we confirm:</u>

- That the work is original, and due credit is given to others where appropriate.
- That all members have contributed substantially and proportionally to each group assignment.
- That all members have sufficient familiarity with the entire contents of the group assignment so as to be able to sign off on them as original work.
- Acceptance and acknowledgement that assignments found to be plagiarized in any way will be subject to sanctions under the University's Code of Behaviour on Academic Matters.

Please **check the box and record your student number** below to indicate that you have read and abide by the statements above:

■	1003463995	☐	
■	1003248844	☐	
☐		☐	

Assignments are to be submitted using Student ID Numbers *only*; do not include your name.
Assignments that include names or that do not have the box above checked **will not be graded.**

Please pay attention to Course Outline for specific formatting requirements set by instructors.

If submitting this assignment to eDropBox, please use the following "Standard File Naming Convention":

Full Course Code *(including Section)*–Group Name or Number–Assignment Title

Example: RSM1234HS.2016-0101-Group1-Homework1

On this report, we outlined the general procedure used to solve all questions from the assignment. Detailed calculations for **Part1** problems are available on the attached **excel file**. Detailed codes used to answer **Part 2** are available on the **HTML file** that is also attached to this report.

Part 1

Question 1 – Part 1:

To estimate volatility we simulated different time periods. Using historical data, the 24 months annualized volatility is 13.8%. For 12 month and 6 months period, the annualized volatility is 10.17% and 8.06% respectively. A roll back of 3 years leads to 14.15% volatility. If we go even further back, volatility goes to 16.4% and 26.4% with 4 and 5 years respectively. Finally, with a 6-year, period volatility of the portfolio is 25.7%.

Given these, we chose a 24-month period with an annualized volatility of 13.8% combined with a sensitivity analysis (spreadsheet).

We are assuming a dividend yield of 3.86%, interest rate of 1.5%, period to maturity of 6 years (considering that the VBA code takes on account the 12-semi-annual average date). All information used was given either by the professor or found in the TD client flyer. The product description tells that if the returns are negative, the holder of the note won't incur losses. Also, that it will consider the average of returns on semi-annual intervals as the pay-off. Thus, because these assumptions slightly differ from the average option formulation, we ran Monte

Carlo to simulate a Brownian motion over the 6-year period and averaged the returns only over the semi-annual periods. The idea was basically to check the assumption we made about the weighted average option formulation.

A) If use the average option VBA formula provided by the professor, we priced the option embedded in the product at $4.13. The Monte Carlo simulation results in an option price of $3.95.

The bond was priced as strip bond with an interest rate of 1.5%. The price of the bond is 91.39. Therefore, the final product (bond + option) is worth $95.52 (using Monte Carlo it would be worth $95.35).

B) The effective interest rate is 0.76%, which is the rate to which when the principal is discounted we obtained the value of the note.

<u>Question 2 – Part 1:</u>

To replicate the pay-off of the note we need to build a portfolio with:

1) Long position: 3 calls at the money

2) Short position: 3 calls with strike price k = (1+0.4/3)*SPX

3) Short position: put with strike price k = (0.9)*SPX

4) Long position on a bond

The strategy consisted of buying 3 calls ATM ($885.28); selling 3 calls OTM with the strike price capped on 40% of the value of the note [885.28*(1+40%/3)]; selling 1 put OTM with strike 10% lower of the ATM value (885.28*0.9); and buying a bond with value of (=Principal*EXP(-Int_Rate*Life)).

The strike price of the positive side of the payoff is found by manipulating the following relationship: Payoff = $10\$ + 10\$ * 300\% *$ □□□□ $< 14\$$ If we isolate delta SPX, we find that the maximum change is $\frac{0.4}{3}\%$ ..

For all assets, we used the maturity of 2.083 years which is the difference between 1/7/2011 and 12/18/2008. To transform to years, we divided the delta days by 360, and we also assumed that all strike prices could be a fractional number.

B) To get the value of the note we used the =BackShcoles() formula to price all options. To price the bond, we used the formula =Principal*EXP(-Int_Rate*Life). To get the quantities of the options we divided the principal value over index value, remembering that to match the levered payoff we had to buy and sell 3x the call options. Finally, to price the note, we added the costs to get the final price of $9.06.

Question 3 – Part 1:

A) First, we calculated the payoff of the coupons. We annualized the period (dividing the difference of T by 360) and computed the PV of all coupons, then we added theses PV values ($0.9248).

We did a similar process on the dividends, annualizing the periods and getting the PV of the projected dividends. We needed the PV of dividends to value the option, thus we used the values from the note's issuance until the maturity of the call (we calculated the accrual for the period Jan/20/2009).

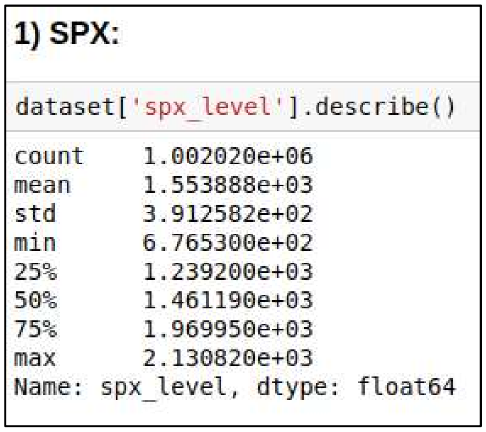
Then we calculated the option price using the =Black_Scholes() formula. We used the strike price of $11.6337*2 and used the dividend yield as the rate between the issue date and Jan/20/2009. For the spot price we added the PV of the coupons that are after the exercise date (4/20/2009 and 7/20/2009), and the PV of the dividends. We priced the options at $0.84, therefore the total cost of the note is equal to $10.47 (S0/2 - Option price + Bond payoff).

Part 2

We will post the results on this report, but the details regarding the implementation and concepts used to solve the problems are described in the jupyter-notebook attached to this file (html format).

Question 1 – Part 2:

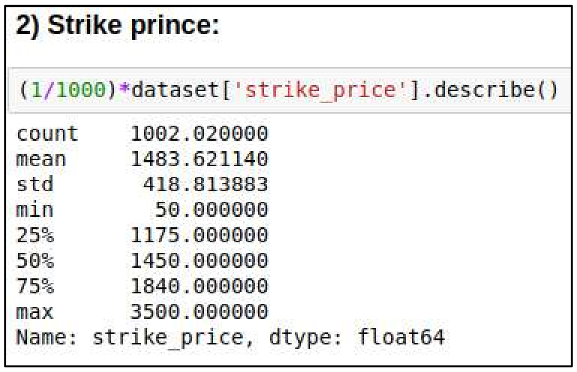
The descriptive statistics were as follows:



1) SPX:

```
dataset['spx_level'].describe()
```

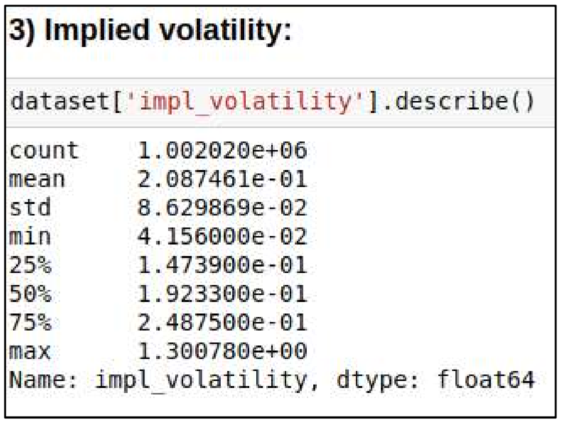
```
count    1.002020e+06
mean     1.553888e+03
std      3.912582e+02
min      6.765300e+02
25%      1.239200e+03
50%      1.461190e+03
75%      1.969950e+03
max      2.130820e+03
Name: spx_level, dtype: float64
```



2) Strike prince:

```
(1/1000)*dataset['strike_price'].describe()
```

```
count    1002.020000
mean     1483.621140
std       418.813883
min        50.000000
25%      1175.000000
50%      1450.000000
75%      1840.000000
max      3500.000000
Name: strike_price, dtype: float64
```

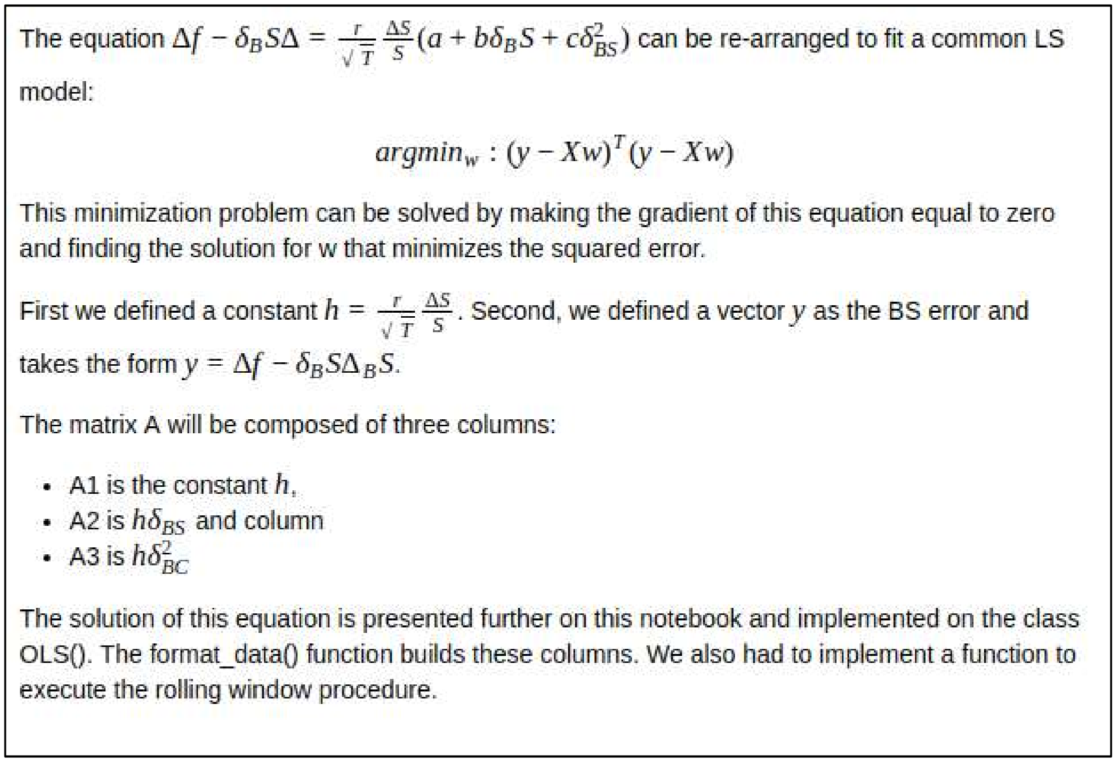


3) Implied volatility:

```
dataset['impl_volatility'].describe()
```

```
count    1.002020e+06
mean     2.087461e-01
std      8.629869e-02
min      4.156000e-02
25%      1.473900e-01
50%      1.923300e-01
75%      2.487500e-01
max      1.300780e+00
Name: impl_volatility, dtype: float64
```

Question 2 – Part 2:



The equation $\Delta f - \delta_B S\Delta = \frac{r}{\sqrt{T}}\frac{\Delta S}{S}(a + b\delta_B S + c\delta_{BS}^2)$ can be re-arranged to fit a common LS model:

$$argmin_w : (y - Xw)^T(y - Xw)$$

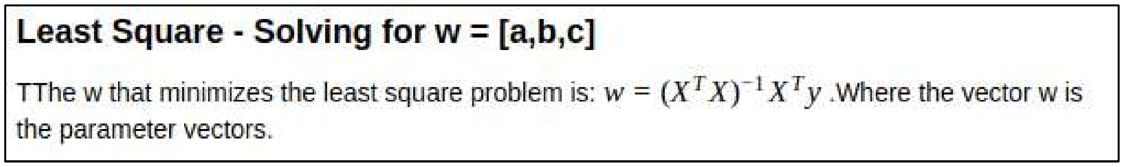
This minimization problem can be solved by making the gradient of this equation equal to zero and finding the solution for w that minimizes the squared error.

First we defined a constant $h = \frac{r}{\sqrt{T}}\frac{\Delta S}{S}$. Second, we defined a vector y as the BS error and takes the form $y = \Delta f - \delta_B S\Delta_B S$.

The matrix A will be composed of three columns:

- A1 is the constant h,
- A2 is $h\delta_{BS}$ and column
- A3 is $h\delta_{BC}^2$

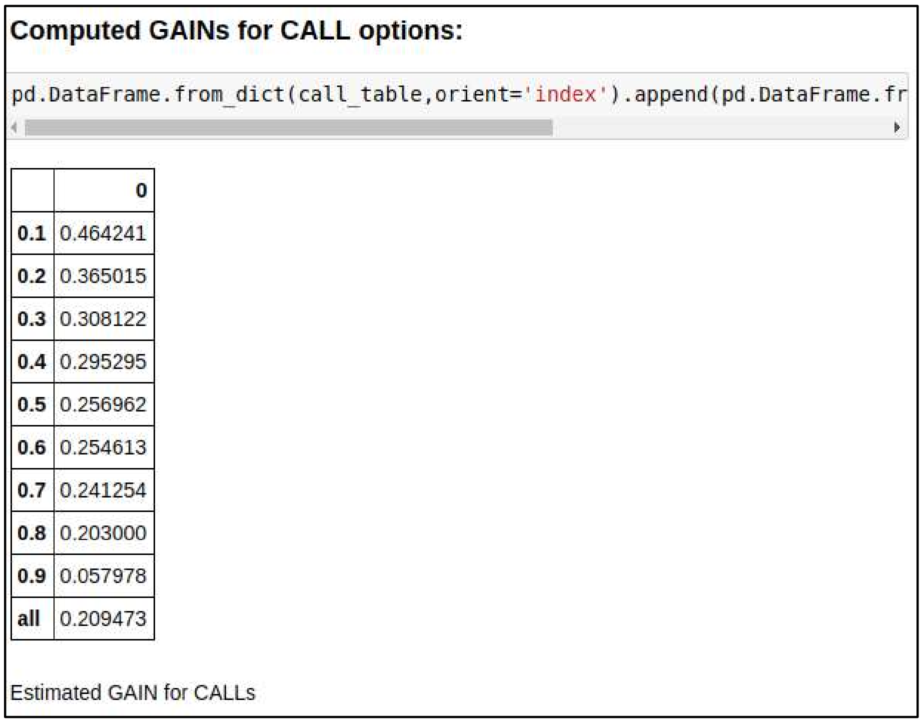
The solution of this equation is presented further on this notebook and implemented on the class OLS(). The format_data() function builds these columns. We also had to implement a function to execute the rolling window procedure.



Least Square - Solving for w = [a,b,c]

TThe w that minimizes the least square problem is: $w = (X^TX)^{-1}X^Ty$ .Where the vector w is the parameter vectors.

The simulations we developed returned GAINs very similar to what were presented by (J. Hull, 2017)

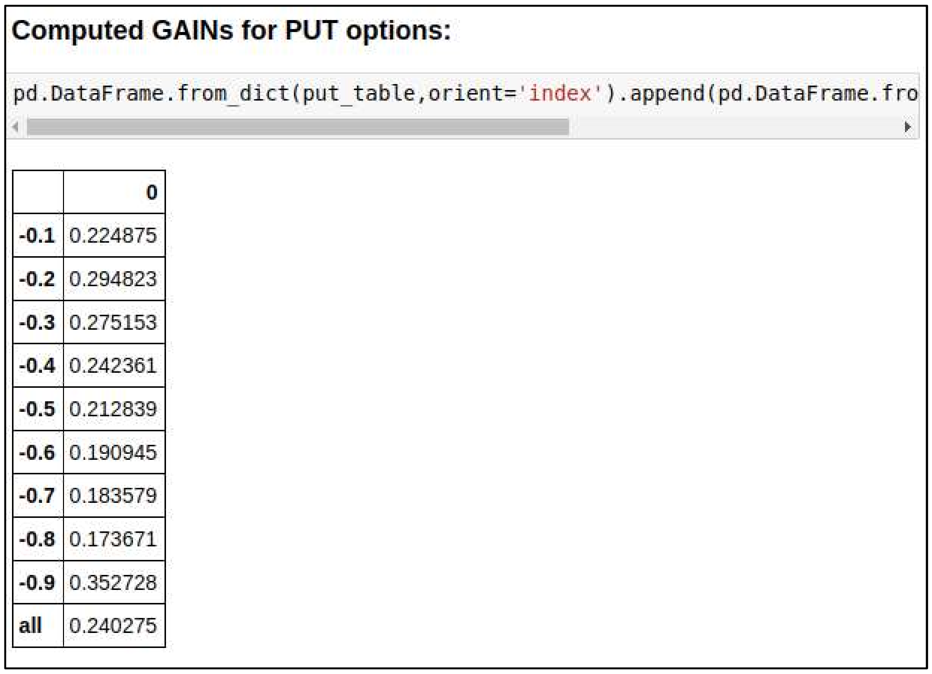


Computed GAINs for CALL options:

```
pd.DataFrame.from_dict(call_table,orient='index').append(pd.DataFrame.fr
```

	0
0.1	0.464241
0.2	0.365015
0.3	0.308122
0.4	0.295295
0.5	0.256962
0.6	0.254613
0.7	0.241254
0.8	0.203000
0.9	0.057978
all	0.209473

Estimated GAIN for CALLs



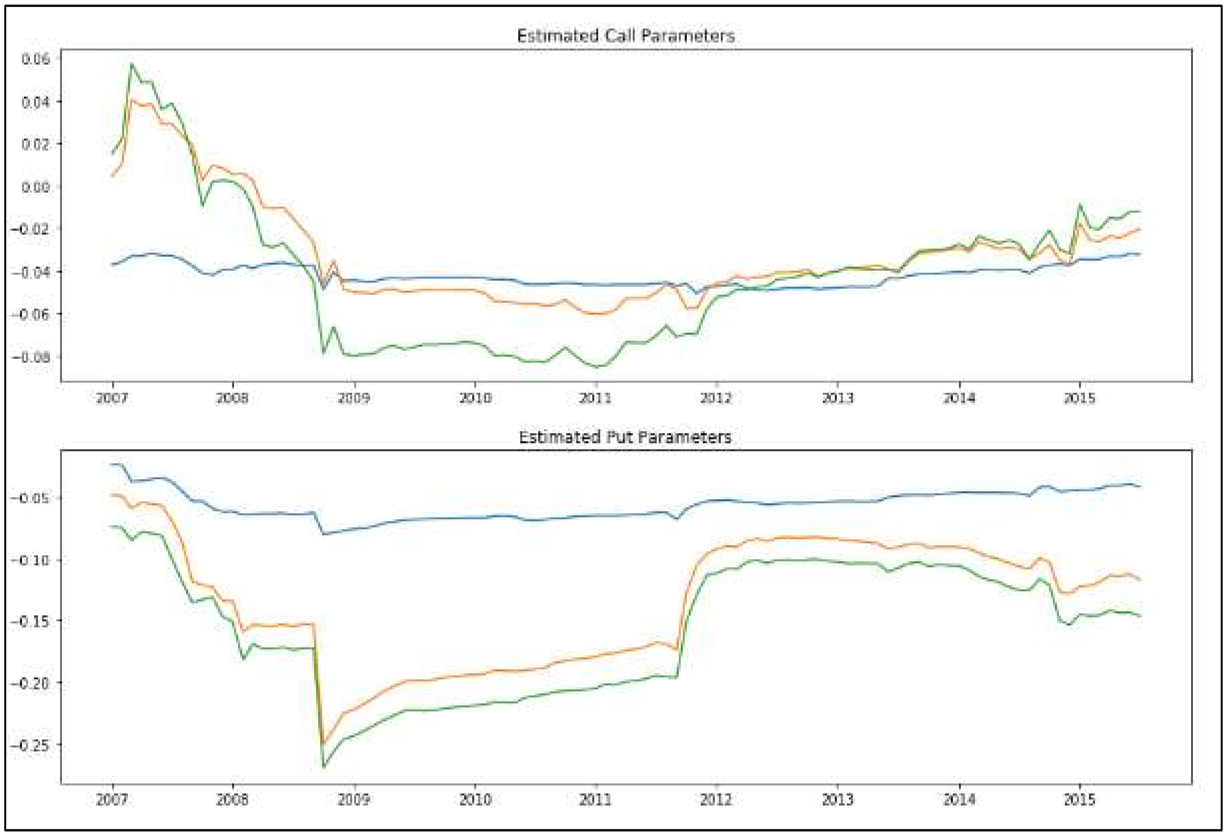
Computed GAINs for PUT options:

```
pd.DataFrame.from_dict(put_table,orient='index').append(pd.DataFrame.fro
```

	0
-0.1	0.224875
-0.2	0.294823
-0.3	0.275153
-0.4	0.242361
-0.5	0.212839
-0.6	0.190945
-0.7	0.183579
-0.8	0.173671
-0.9	0.352728
all	0.240275

Question 3 – Part 2:

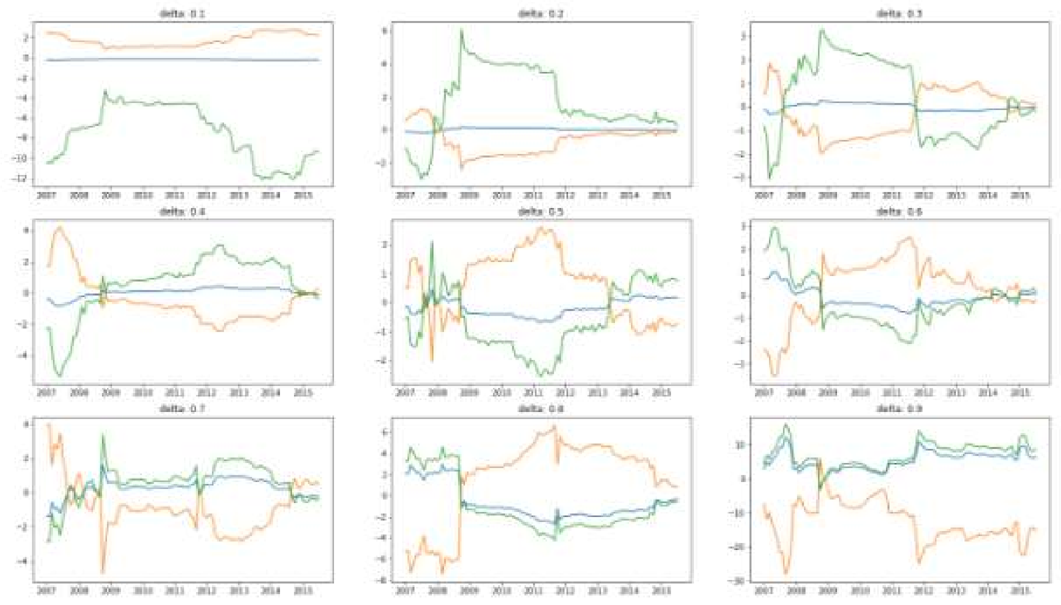
Next, we present the parameter plot for the cases in which no bucket was used for either puts or calls.



We also present here the parameter plots for individual cases: puts and calls and different delta buckets:

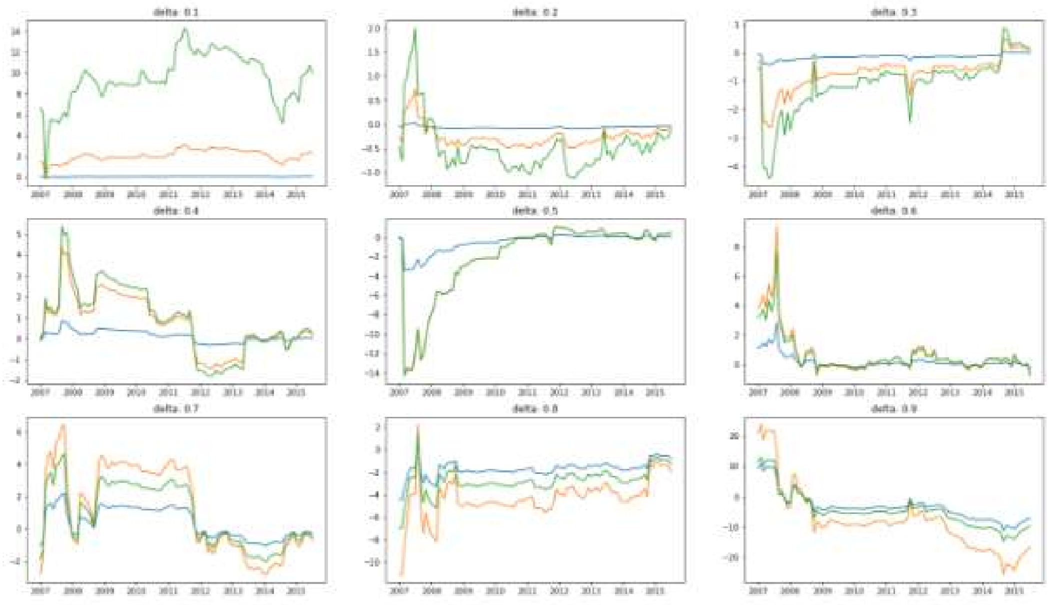
Coeficients for different deltas/buckets (Calls)

```
plot_reg(call_reg)
```



Coeficients for different deltas/buckets (Puts)

```
plot_reg(put_reg)
```



Question 4 - Part 2

Controling for additional factors:

Starting from the noise model:

$$\Delta f - \delta_{MV} \Delta S = \epsilon$$

We break down the expression for δ_{MV} and introduce three new factors, adding new columns to the matrix A:

- A4: Fator d, controls for the spx level
- A5: Factor e, controls for the percent change in the SPX
- A6: Factor f, controls for the percent change in the option price

The "brazuca" modified δ_{MV} becomes:

$$\delta_{MV} = \delta_{BS} + \frac{\upsilon_{BS}}{S\sqrt{\ T}}(a + b\delta_{BS} + c\delta^2) + dS_0 + e\frac{\Delta S}{S_0} + f\frac{\Delta f}{f_0}$$

Which leads to:

$$\epsilon_{MV} = \Delta f - \delta_{BS}\Delta S - \frac{\upsilon_{BS}}{\sqrt{\ T}}\frac{\Delta S}{S}(a + b\delta_{BS} + c\delta_{BS}^2) - d\Delta S S_0 - e\frac{\Delta S^2}{S_0} - f\frac{\Delta S \Delta f}{f_0}$$

The model continues linear with respect to the factors, thus it can be solved using LS as previously. So we take the deltaMV error

$$\epsilon_{MV} = \Delta f - \delta_{MV}\Delta S$$

adjust the formulation for deltaMV and solve for:

$$argmin_w : \frac{1}{2}(y - Ax)^2$$

The matrix A will now have six columns:

- A1 is the constant h,
- A2 is $h\delta_{BS}$ and column
- A3 is $h\delta_{BC}^2$
- A4 is S_0
- A5 is $\frac{\Delta S}{S_0}$
- A6 is $\frac{\Delta f}{f_0}$

Finally, the parameter vector is $w = [a, b, c, d, e, f]$ and the vector $y = \Delta f - \delta_B S \Delta_B S$, which is the BS error ϵ_{BS}

Introducing regularization (Ridge Regression)

The least square solution for the linear model we are discussing have a probabilistic interpretation because it leads to the same optimial solution as the maximum likelihood estimate of a normal distribution with mean Ax and standar deviation σ. We choose to introduce a prior to the model, leading to a Bayesan model whise MLE is the same as the solution of a Ridge Regression.

$$argmin_w : (y - Xw)^T (y - Xw) + \lambda X^T X$$

And the solution obtained from taking the gradient and making it equal to zero is:

$$w = (X^T X + \lambda I)^{-1} X^T y$$

Because lambda now is a hyper parameter that needs to be choosen we used a k-fold validation procedure with random search at each iteration to select the lambda, using proper validation data, that would generate the minimum error. The 3-fold procedure splits the roll_back data into k folds and alternate the process of training and validating the data using each k split. The results SSE for a set of three trials is recorded for each random parameter. The parameter whose average is the best will be selected to calculate the SSE on the forward data.

We used implementations from SKlearn to minimize coding...

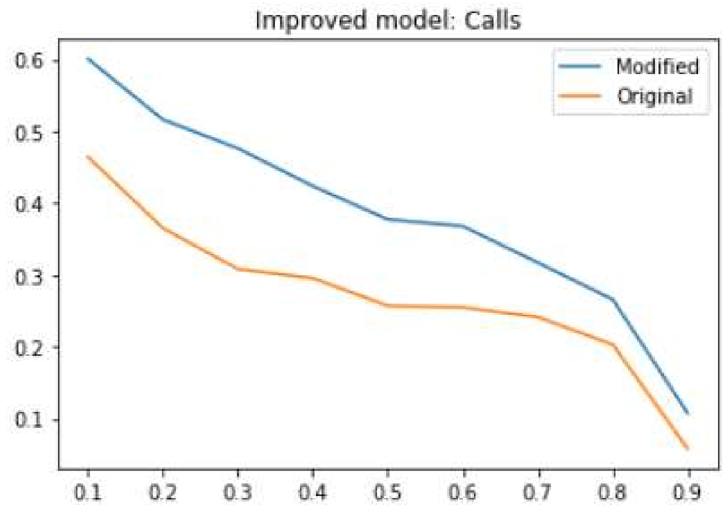


Figure 1 GAIN (y-axis) versus delta bucket (x-axis) for calls after introduction of new factors (modified delta MV)

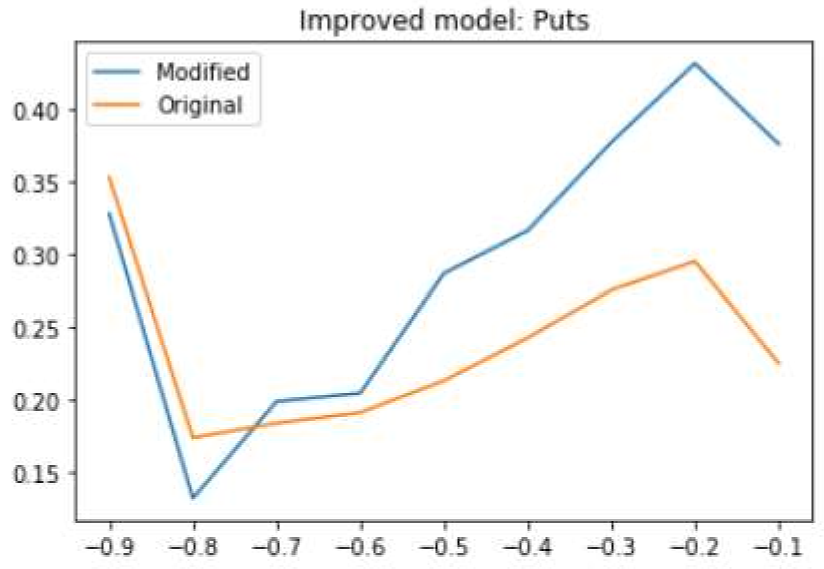


Figure 2 GAIN (y-axis) versus delta bucket (x-axis) for puts after introduction of new factors (modified delta MV)

Changing the Rolling Window:

In addition to changing the model, we performed a test to select a rolling window that could also improve the gain. The image bellow is a surface plot of the GAIN for a set of rolling window parameters, rolling forward (measured in days) and backward (measured in months).

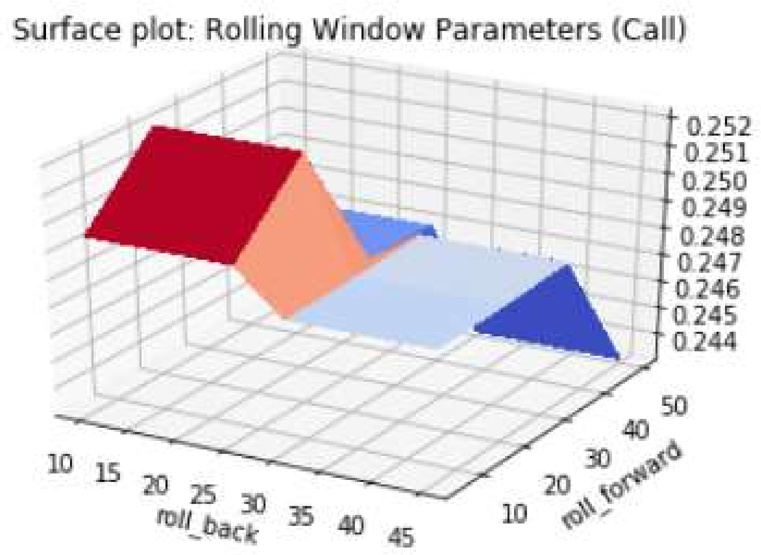


Figure 3 Surface plot of the rolling window parameters (backward and forward) versus the GAIN (calls)

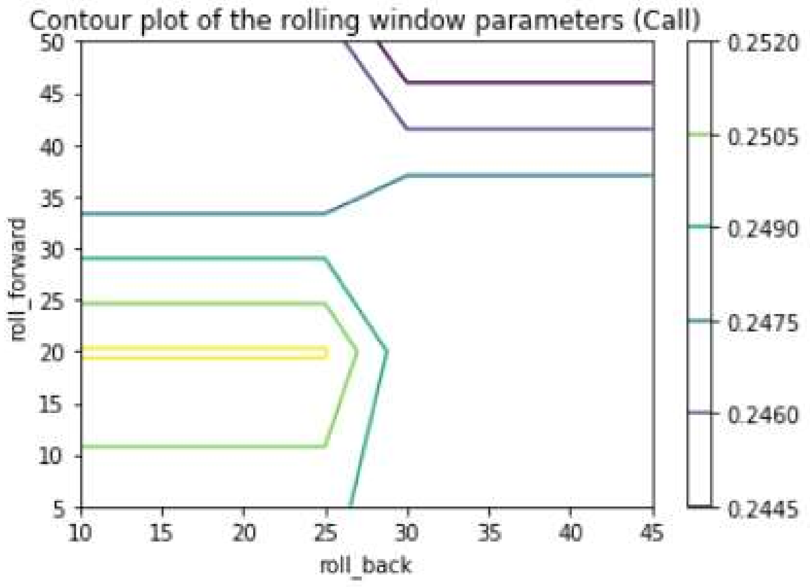


Figure 4 Contour plot of the rolling window parameters (backward and forward) versus the GAIN (calls)

Based on the contour, in our final attempt to improve performance we used the modified approach with a back window of 20 months and a forward window of 20 days. The GAIN on a call, within all buckets, is 33% and for a put is 30%. When analysing the improvement per bucket, we see a higher benefit for buckets between 0.5 and 0.8 for calls and a widespread benefit for puts. The next charts are plots comparing the GAINs (y-axis) with the delta buckets (X-axis) using the modified delta MV, a back rolling window of 20 months and a forward of 20 days.

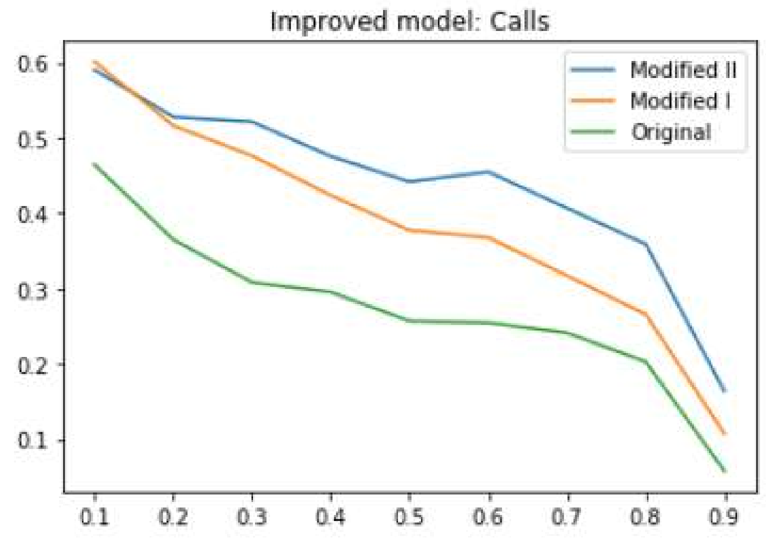


Figure 5 Gain (y-axis) versus delta bucket (x-axis) - Calls

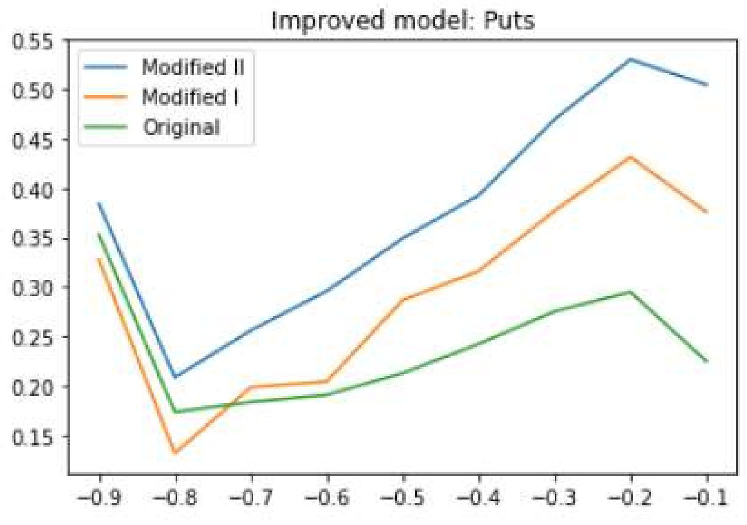


Figure 6 Gain (y-axis) versus delta bucket (x-axis) - Puts

Rotman School of Management
UNIVERSITY OF TORONTO

Rotman

Final Page

Grade: ____________