

2015 – 2016

Group MEMBER:

Yiting LIAN

XXX XXX

XXX XXX

PROFESSOR:

Ms. Aurélie SANNAJUST

Final Report

VBA Initiation

Table of content

[INTRODUCTION 2](#_Toc439256286)

[I. Description of the model 3](#_Toc439256287)

[1.1 Definition 3](#_Toc439256288)

[1.2 Objectives / Interests / Limits 3](#_Toc439256289)

[II. Description of formulas in English 4](#_Toc439256290)

[Input 4](#_Toc439256291)

[Output 4](#_Toc439256292)

[III. Description of mathematical formulas 4](#_Toc439256293)

[IV. Program in VBA and Comments of Code 7](#_Toc439256294)

[4.1 User interface 7](#_Toc439256295)

[4.2 VBA Code 8](#_Toc439256296)

[a. Inside <MFModelForm> 8](#_Toc439256297)

[b. Inside < MFModelCalculation > 12](#_Toc439256298)

[V. Error Handling 14](#_Toc439256299)

[VI. Conclusion 15](#_Toc439256300)

[Bibliography 16](#_Toc439256301)

# INTRODUCTION

As a part of the <VBA Initiation> course at EMLyon Business School, this final project is accomplished by three team members (Yiting LIAN, XXX and XXXX). The goal of this project is to create a pricing tool in VBA programming language by implementing the Molodovsky and Fuller model.

This report will begin with the introduction of Molodovsky and Fuller model, which includes its functional description and formulas.

After that, this report will focus on the technical part. It will show our programming logic by explaining the codes in this final project. And In this part, we will also talk about the error handling issue which is obviously necessary for a better user experience.

We will end this report by a conclusion with difficulties met during this project.

# Description of the model

## Definition

The Molodovsky and Fuller model describe a form of the dividend discount model which divides a company into three phases. A dividend discount model is a calculation of the value of a publicly-traded company's common stock based on the present value of its future dividends. The three-phase DDM assumes that a company's dividend policy is determined by one of three phases: a growth phase, a transition phase, and a maturity phase. The company's goals differ in each phase and its dividend policy changes accordingly. The three-phase DDM attempts to account for this in its calculation.

## Objectives / Interests / Limits

The objective of this model is to propose a more realistic model for the practitioners when compared to the constant growth dividend discount model. Obviously, in reality the dividends do not grow at a constant rate forever.

In the case, the interests of this model are very easy to understand. It aligns the financial theory to the real practice. It also gives us the possibility to vary the discount rate R with a required return by giving the level of risk of the studied company.

On the other hand, this evaluation method does not take account of the profitability of the enterprise or the context (especially for the inflation). That is why this model is very hard to be used by financial market specialists in the brokerage firms. What’s more, it is based on the observations which are too old-timey to be adapted by the current economic environment.

# Description of formulas in English

As described before, the Molodovsky and Fuller model compute the price of today by a three-phase model. That leads us to define the following variables for our VBA project.

## Input

**D(0)** : Last paid dividend (in money units , $, €, £ etc)

**Gb**: Growth rate at the beginning phase, i.e. Phase 1 in the model (in percentages)

**Nb:** Time for the beginning phase (Phase 1) (in years)

**Nd:** Time for the second phase (Phase 2) (in years)

**Gc:** Growth rate at the last phase (Phase 3), which is also the final growth rate (in percentages)

**R:** Discount rate (in percentages)

In our project, we assume that our users could estimate the discount rate R by using the Capital Asset Pricing Model. In this case, we need to take three more variables as the input variables for our program.

**Rf:** Risk free rate (in percentages)

**Rm:** Return of the market portfolio (in percentages)

**Beta:** Measure of the volatility or systematic risk of a security or a portfolio in comparison to the market as a whole.

## Output

**M:** the dividend multiplier

**P:** the calculated price (in money units, $, €, £ etc)

**R:** Discount rate (in percentages). Obviously, it’s an output variable if the user tried to calculate this variables via Capital Asset Pricing Model.

# Description of mathematical formulas

In order to give a more precise description of the mathematical formulas, we define at first four temporary variables used in the formulas.

**M1**: the dividend multiplier for Phase1

**M2**: the dividend multiplier for Phase2

**M3**: the dividend multiplier for Phase3

**D(t)**: Calculated dividend rate at Time t

**G(t):** Calculated Growth Rate for Phase2 at time t

So, the final the dividend multiplier M should be:

(1)

And the final price would be:

(2)

Thus, we will get the following formulas to compute the output variables by using the input and calculated variables.

**For Phase 1:**

(3)

**For phase 2:**

(4)

With:

(5)

(6)

So:

(7)

Finally:

(8)

\*Note: At the beginning of the sum function, which means when t = 1, we have:

**For Phase 3:**

(9)

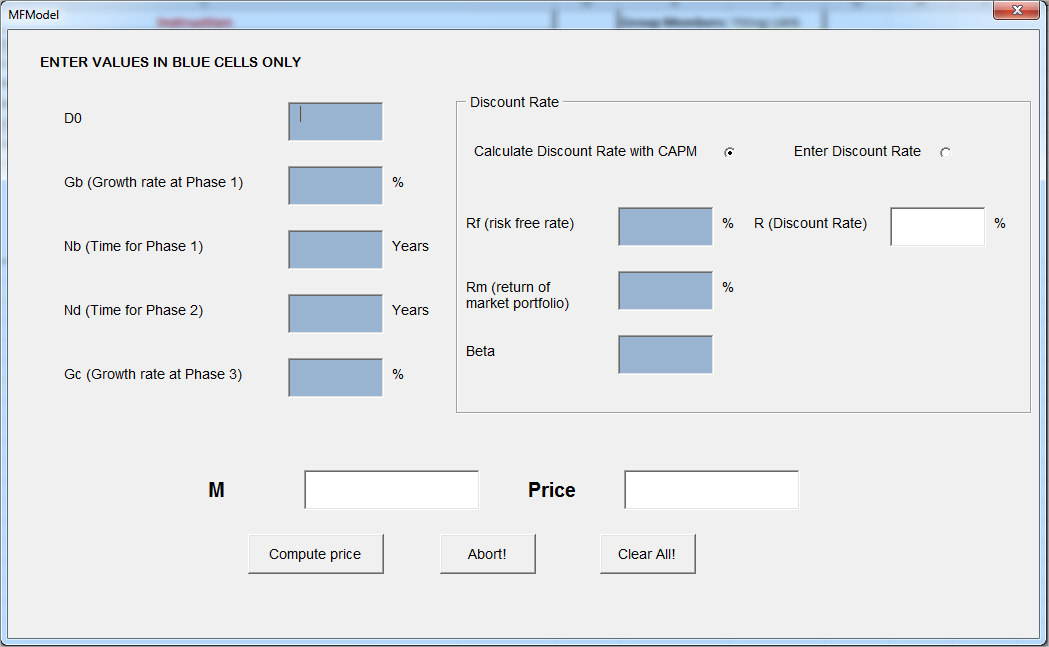
**And for calculating the discount rate R:**

(10)

# Program in VBA and Comments of Code

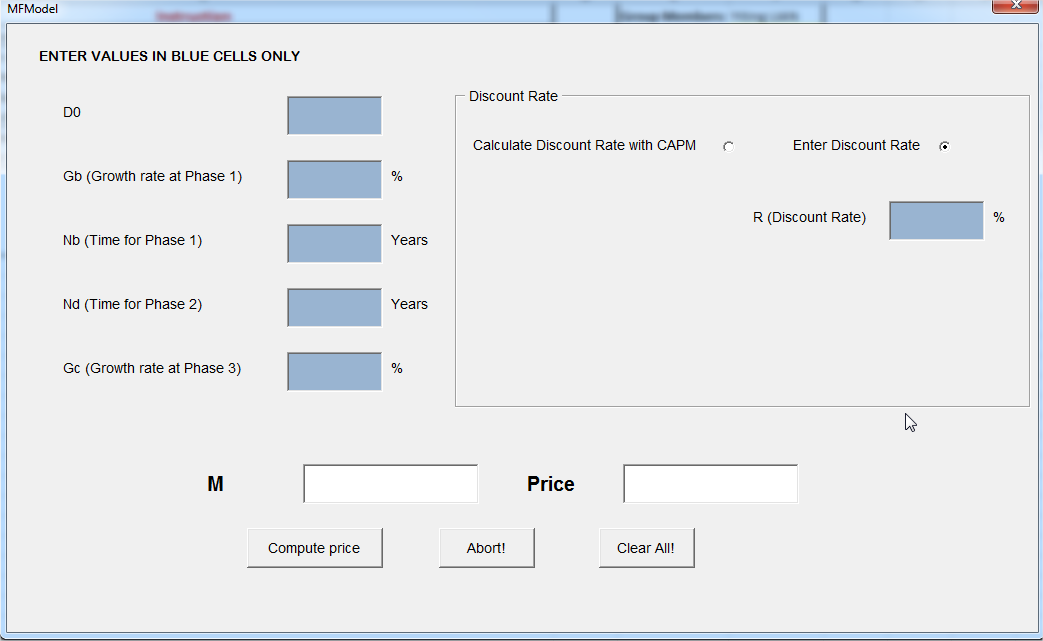
## User interface

We have created a UserForm to establish the communication between users and the program, as shown below:



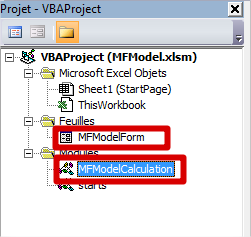
We use the blues cells to indicate the input variables (the variables need to be entered by the users) and the white cells to print out the output variables.

As said before, by clicking the option box <Enter Discount Rate> in “Discount Rate” frame, our users could also enter directly a Discount rate R instead of calculating it by our program. In this case, this interface will turn into:



## VBA Code

We have divided our codes into two major parts.



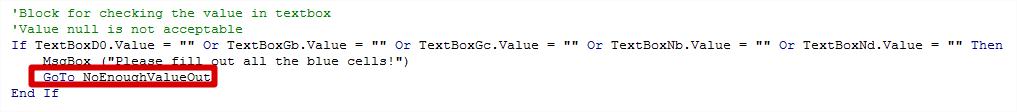
All the codes concern the actions and the reactions of the userform are integrated in the <***MFModelForm***> and all the codes about the mathematical calculations are included in the module named <***MFModelCalculation***>.

### **Inside <MFModelForm>**

#### When a user clicks the <Compute price> button:



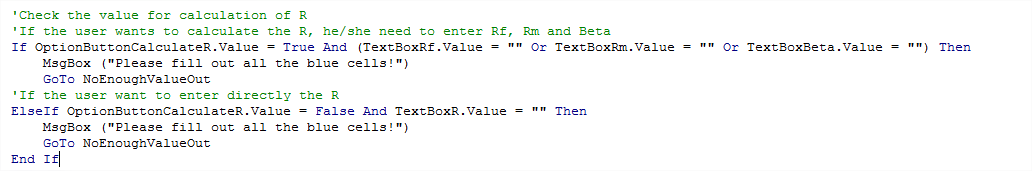
Our program will check at first if the user has entered all the necessary input variables. If not, it will pop-up a message box to inform the user and quit the program with a GOTO action.



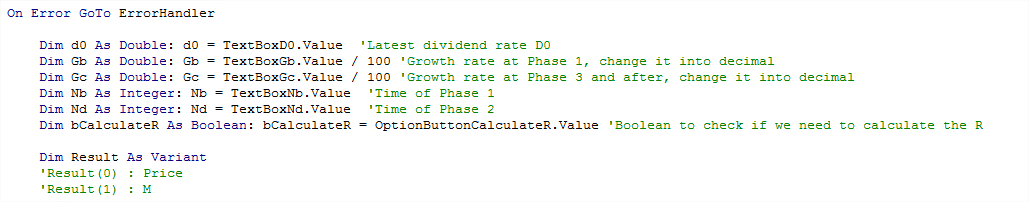
The NoEnoughValueOut is written at the end of this Sub, before the Error Handler:



For calculating the discount rate R, we check the option activated by the user. If the user wants to calculate R, he/she needs to enter Rf, Rm and Beta. Otherwise, he/she needs to enter directly R in the textbox.



Then, the program will read all the inputs into the defined variables:



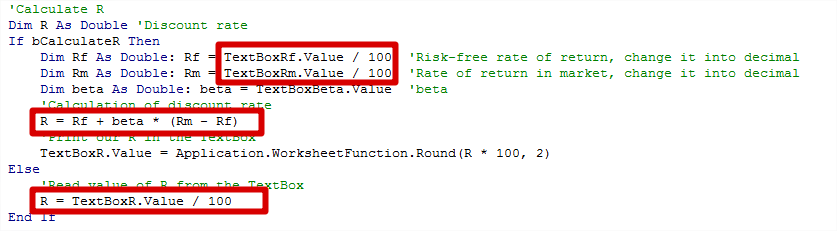
\*The error handing action will be explained in the next chapter.

We define the variable ***Result*** as Variant which will be used to stock the output variables. ***Result(0)*** is for calculated price and ***Result(1)*** is for M.

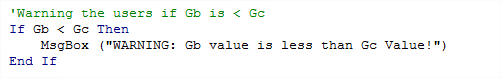
After that, we will calculate the discount rate R. Again, we will check the option activated by user.

If he/she wants to calculate R by CAPM model, we will apply the Formula (10) here. Otherwise, the program will read directly R into the defined variable.

Since all the rates will be entered in percentages and be calculated in decimals, we will also need to do a transformation when reading the values.

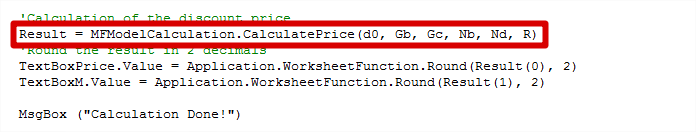


The model implicit that Gc will always be lower than Gb. So, if it’s not this case, we will warn the user by a message box but we will not quit the program.



As said before, the calculation will be done in the function named ***CalculatePrice*** inside the module <***MFModelCalculation***>. This function will be explained later.

Finally, the program will round the results into 2 decimals and print it in the textboxs.



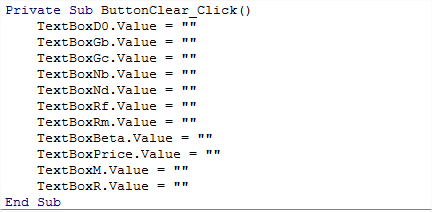
#### When a user clicks the <Abort!> button:

The program will hide the userform.



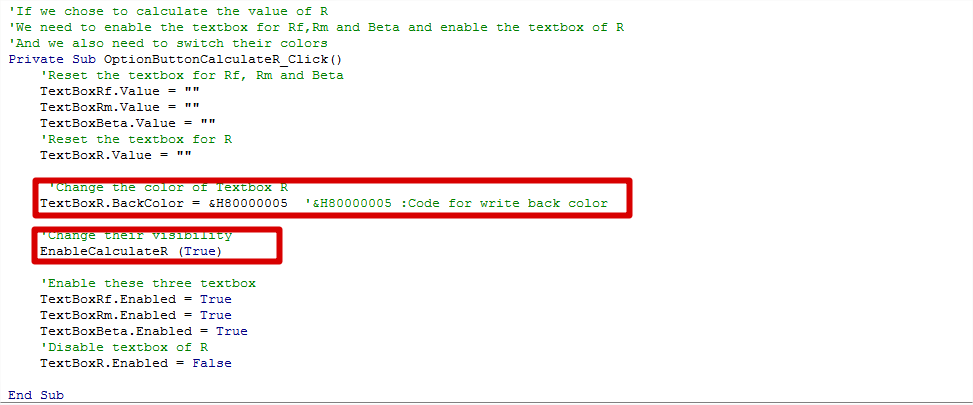
#### When a user clicks the <Clear All!> button:

It will clear all the textbox by setting their values to “”.



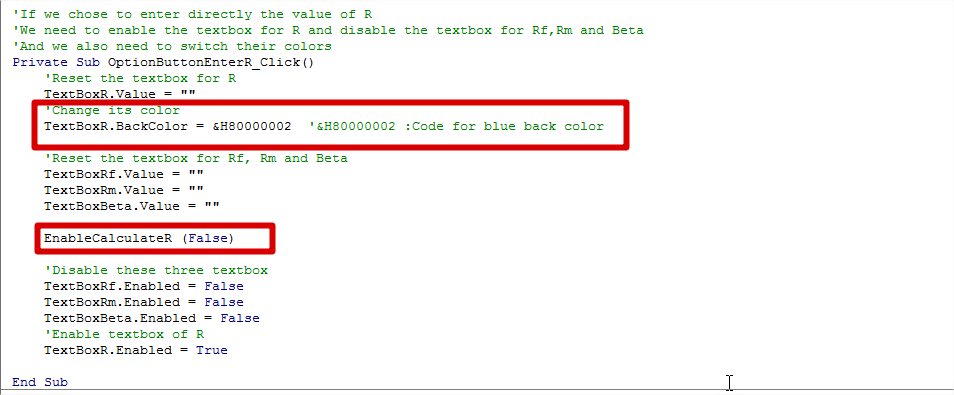
#### When a user clicks the < Calculate Discount Rate wit CAPM> Option:

It means that the user wants to calculate the discount rate R. In this case, we will reset the textboxs in “Discount Rate” frame and change the color of textbox R into white as it turns into an output variable. We will enable the textbox for Rf, Rm and Beta and make them visible by an external function named ***EnableCalculateR().***



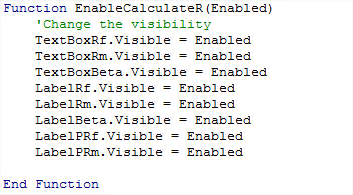
#### When a user clicks the < Enter Discount Rate> Option:

It means that the user wants to enter directly the discount rate R. In this case, we will reset the textboxs in “Discount Rate” frame and change the color of textbox R into blue as it turns into an input variable. We will disable the textbox for Rf, Rm and Beta and make them invisible by an external function named ***EnableCalculateR().***



#### Function EnableCalculateR(Enabled)

We use this function to change the visibility of the textboxs and labels for Rf, Rm and Beta. It takes a Boolean ***Enabled*** as an input variable.



### **Inside <** **MFModelCalculation >**

#### Function CalculatePrice



This function is used for calculating the output variables by using the input variables.

It takes six variables as input which correspond the input variables as defined in Chapter II.

**d0** : Last paid dividend (in money units , $, €, £ etc)

**Gb**: Growth rate at the beginning phase, i.e. Phase 1 in the model (in percentages)

**Nb:** Time for the beginning phase (Phase 1) (in years)

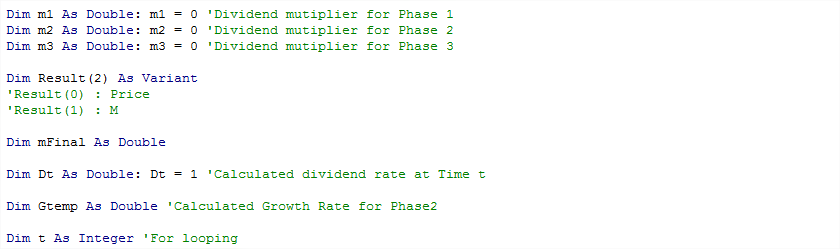
**Nd:** Time for the second phase (Phase 2) (in years)

**Gc:** Growth rate at the last phase (Phase 3), which is also the final growth rate (in percentages)

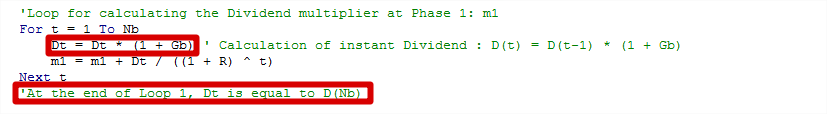
**R:** Discount rate (in percentages)

And it returns a Variant which contains the calculated results. Once again, ***Result(0)*** is for calculated price and ***Result(1)*** is for M.

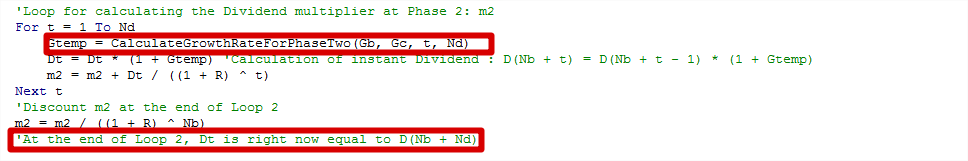
At first, we define the temporary variables as described in Chapter III.



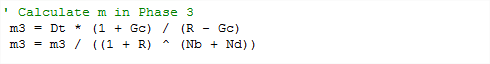
After that, we use a first For loop to simulate Phase 1 in model and apply the Formula (3) for calculating M1. As said in comments, at the end of Loop 1, the variable Dt will be equals to D(Nb).



Then, we use another For loop to simulate Phase 2 and apply the Formula (8) in order to calculate M2. Here, we create an external function named ***CalculateGrowthRateForPhaseTwo()*** to apply the Formula (6). At the end of Loop 2, Dt is right now equal to D(Nb + Nd).



At the end, we perform two lines of codes to apply the Formula (9) for calculating M3.

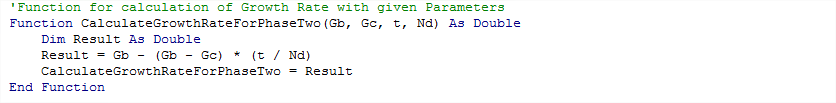


We calculate the final M and calculated price by applying Formula (1) and (2). And we return the result via a Variant as said before.



#### Function CalculateGrowthRateForPhaseTwo

This function calculates the instant Growth Rate at Phase2 with a given t by applying the Formula (6).

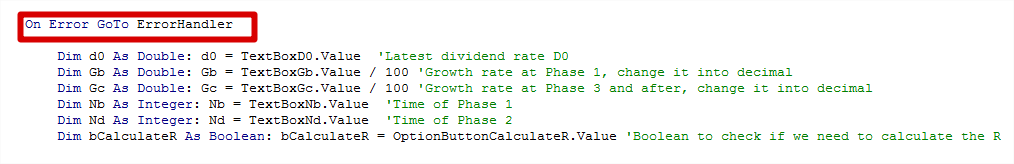


# Error Handling

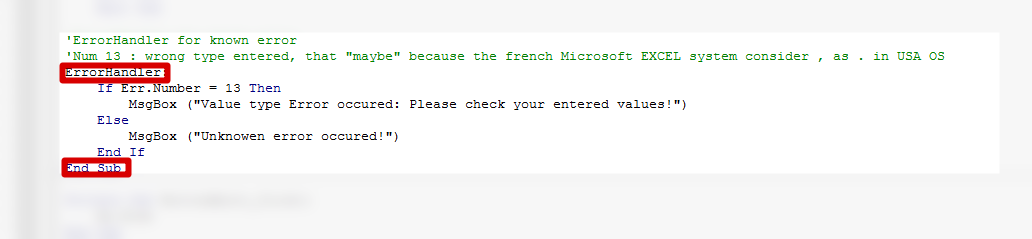
We have included an error handling mechanism when the program reads the input variables and calculates the output variables.

It begins before the reading of the input variables by an ***On Error GoTo*** action. As shown below, every error occurred after this line of code will lead the program go to the pre-defined area which is usually called Error Handler.

We have placed the Error Handler at the end of the sub BouttonComputePrice\_Click() because in this case, after the executing of the Error Handler, the program will ends automatically as it runs into its last line of code.



… …



During the development of this project, we have discovered a known error with an error number 13. This error will be occurred when the program can’t read or calculate the data with the given type of variables. For instance, the action of saving a String value into an integer variable will occurs this type of errors.

So here, we have given more details of this type of error with a message box to inform the user.

# Conclusion

Thanks to the documentation given by Ms. Aurélie SANNAJUST, the financial model in this project is not very hard to understand for us.

On the other hand, we have spent the majority of our time in applying mathematical formulas in VBA codes. It was a little bit difficult at the beginning but once we have understood the principles of Loops in VBA, especially the For loop. Everything turns to be clear for us.

We were also blocked by the Error handling issue at the beginning but hopefully, there is a lot of useful information on the internet. At the same time, while we were searching the information on the internet, we discovered that there is still a lot of new stuffs in VBA, which encourage us to continue in this area.

As a group of business school students, we have noticed that VBA is right now a very important and useful tool for business life. That’s why we have taken this course to improve our skills in this area and to prepare for our careers. Indeed, this project has given us a great opportunity to practice what we have learned during the course <VBA initiation>.

# Bibliography

http://financial-dictionary.thefreedictionary.com/3-Phase+Dividend+Discount+Model

http://www.memoireonline.com/03/10/3231/m\_Evaluation-dune-societe6.html

<Programming the three-phase dividend discount model> by Russell J. Fuller