

The Analysis

Michael Jensen
MBA 614
December 8, 2011

Executive Summary

Performing financial analyses require time. If they are desired to be run on multiple companies it can be repetitive. Many financial measurements as well as ratios are already available on financial websites. More complex analyses that are not accessible for free require time to compute for each company. This VBA project automates the process of performing certain analyses.

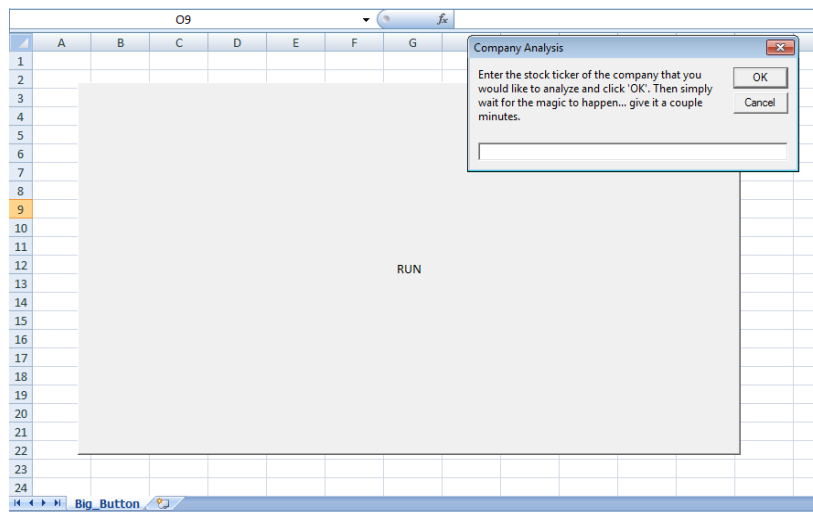
The program pulls financial data from the web, compute analyses, and formats the excel worksheets so the user can follow the assumptions and calculations made. The program is designed with one company in mind and then computes all of the same analyses for the three most comparable firms. The data is pulled from Yahoo! Finance, while the comparable firms are located on Google's finance website. The program takes several minutes to run depending on the speed of the user's internet connection.

The user simply inputs a stock ticker of a company that they want to look at and then waits while the program runs. Over 20 web queries are made to pull in the necessary information while the user waits. Screen updating has been left on so that the user doesn't think the program stopped working and exits Excel. Key financial figures are pulled into an inputs tab for the company and then the excess web query tabs are deleted. Four analysis tabs are created as the final product; valuation, profitability, solvency, and accruals. The computations on these tabs allow the user to view the formulas and trace the inputs.

Documentation

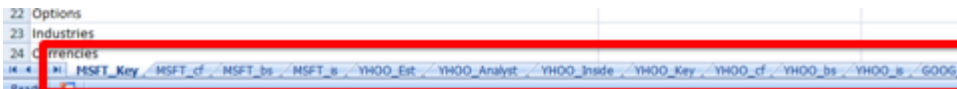
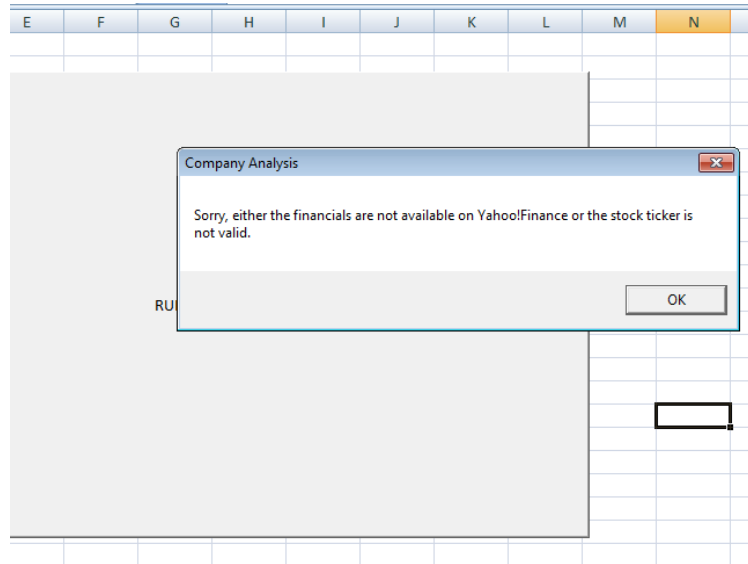
The beauty of the program is that it only requires the user to click, type in a stock ticker, and then click again. The program then takes off from there and does the rest without requiring anything from the user.

The project starts with the user clicking on the giant "RUN" button. This opens up an input message box. The user then inputs the stock ticker of the company that they would like to run the analysis on. Here the user can click cancel or esc if they would like to not run the program. The clicking of OK initializes the remainder of the macro program.



If the user typed in an invalid stock ticker then this message will appear and the program will automatically end.

With a successful first web query to Yahoo! Finance, then the program continues to generate new tabs for each new web query. Since I don't like the comparable companies that are supplied by Yahoo, the macro accesses Google's finance page for the original stock ticker. It locates the three closest competitors and then runs these three through the same process as the original company.



After all of the web queries have been completed the program then goes through a massive finding procedure to locate the desired financial inputs. The input numbers are being selected from the Yahoo! Web queries and made easily available on the inputs tab. After this procedure is done and all of the necessary inputs gathered, the web query tabs are deleted.

Each of the financial numbers on the inputs tab are uniquely named for ease in referencing them on other worksheets. The average is computed for balance sheet items. Also, a timestamp is provided in cell "A2" for reference of when the data was pulled from the internet.

The next step in the program is to create a new tab for each analysis and compute the analysis. The cell name references come in handy here because the user can see the unique name in the formula and understand the computation. The user can also go to this cell by selecting this unique name in the names box to the left of the formula bar.

	CY	PY	AVG
1 GOOG			
2 12_over_8_over_2011_8:46:41_PM			
3	2010	2009	
4 Earnings_Before_Interest_And_Taxes	\$10,796,000,000	\$8,381,000,000	
5 Interest_Expense	\$0	\$0	
6 Income_Tax_Expense	\$2,291,000,000	\$1,861,000,000	
7 Income_Before_Tax	\$10,796,000,000	\$8,381,000,000	
8 Net_Income_From_Continuing_Ops	\$8,505,000,000	\$6,520,000,000	
9 Selling_General_and_Administrative	\$4,761,000,000	\$3,652,000,000	
10 Total_Revenue	\$29,321,000,000	\$23,651,000,000	
11 Cost_of_Revenue	\$10,417,000,000	\$8,844,000,000	
12 Gross_Profit	\$18,904,000,000	\$14,807,000,000	
13 Other_Items	\$0	\$0	
14 Short_Term_Investments	\$21,345,000,000	\$14,287,000,000	\$17,816,000,000
15 Current_Long_Term_Debt	\$3,465,000,000	\$0	\$1,732,500,000
16 Accounts_Payable	\$6,137,000,000	\$2,462,000,000	\$4,299,500,000
17 Net_Receivables	\$5,261,000,000	\$3,845,000,000	\$4,553,000,000
18 Total_Current_Assets	\$41,562,000,000	\$29,167,000,000	\$35,364,500,000
19 Property_Plant_and_Equipment	\$7,759,000,000	\$4,845,000,000	\$6,302,000,000
20 Total_Assets	\$57,851,000,000	\$40,497,000,000	\$49,174,000,000
21 Total_Current_Liabilities	\$9,996,000,000	\$2,747,000,000	\$6,371,500,000
22 Long_Term_Debt	\$0	\$0	\$0
23 Total_Liabilities	\$11,610,000,000	\$4,493,000,000	\$8,051,500,000
24 Retained_Earnings	\$27,868,000,000	\$20,082,000,000	\$23,975,000,000

```

Range("A15").Value = "Net Operating Profit Margin"
Range("A17").Value = "Net Operating Asset Turnover"
'compute the decomposition of ROE for each of the companies
Range("B4").Select
For X = 0 To 3
ActiveCell.Value = Y(X)
'ROE tm
ActiveCell.Offset(1, 0).Formula = "=" & Y(X) & "_Return_on_Equity_CY"
'ROE
ActiveCell.Offset(3, 0).Formula = "=" & Y(X) & "_Net_Income_CY/" & Y(X) & "_
Total_Stockholder_Equity_AVG"
'NOEAT
ActiveCell.Offset(5, 0).Formula = "(" & Y(X) & "_Earnings_Before_Interest_And_Taxes_CY-" & _
Y(X) & "_Income_Tax_Expense_CY+(" & Y(X) & "_Other_Items_CY*" & Y(X) & "_Tax_Rate_CY)" & _
'NOA
ActiveCell.Offset(7, 0).Formula = "(" & Y(X) & "_Total_Assets_CY-" & _
Y(X) & "_Short_Term_Investments_CY-" & Y(X) & "_Long_Term_Investments_CY)-(" & _
Y(X) & "_Total_Liabilities_CY-" & Y(X) & "_Current_Long_Term_Debt_CY-" & _
Y(X) & "_Long_Term_Debt_CY-" & Y(X) & "_Negative_Goodwill_CY-" & Y(X) & "_Minority_Interest_CY)"
'RNA
ActiveCell.Offset(9, 0).Formula = "=" & ActiveCell.Offset(5, 0).Address & "/((" & _
ActiveCell.Offset(7, 0).Address & "+(" & Y(X) & "_Total_Assets_PY-" & _
Y(X) & "_Short_Term_Investments_PY-" & Y(X) & "_Long_Term_Investments_PY)-(" & _
Y(X) & "_Total_Liabilities_PY-" & Y(X) & "_Current_Long_Term_Debt_PY-" & _
Y(X) & "_Long_Term_Debt_PY-" & Y(X) & "_Negative_Goodwill_PY-" & Y(X) & "_Minority_Interest_PY)))/2)"
'NOPM
ActiveCell.Offset(11, 0).Formula = "=" & ActiveCell.Offset(5, 0).Address & "/" & Y(X) & "_Total_Revenue_CY"
'NOAT
ActiveCell.Offset(13, 0).Formula = "(" & Y(X) & "_Total_Revenue_CY/(((" & _
ActiveCell.Offset(7, 0).Address & "+(" & Y(X) & "_Total_Assets_PY-" & _
Y(X) & "_Short_Term_Investments_PY-" & Y(X) & "_Long_Term_Investments_PY)-(" & _
Y(X) & "_Total_Liabilities_PY-" & Y(X) & "_Current_Long_Term_Debt_PY-" & _
Y(X) & "_Long_Term_Debt_PY-" & Y(X) & "_Negative_Goodwill_PY-" & Y(X) & "_Minority_Interest_PY)))/2))"

```

Allowing the user to view the formulas and not just have them hard coded in provides for a better experience with this program. The user can have more trust in the analyses since they can follow the logic. Below is an example of the code to enable these types of formulas in Excel.

```

'compute the Days Receivables Index
ActiveCell.Formula = "=" & ActiveCell.Offset(-12, 0).Address & "/" &
ActiveCell.Offset(-12, 1).Address & "/" & ActiveCell.Offset(-6, 1).
ActiveCell.Offset(1, 0).Select
'Gross Margin Index
ActiveCell.Formula = "=" & (((" & ActiveCell.Offset(-7, 1).Address & "-" &
ActiveCell.Offset(-7, 1).Address & ")/(" & ActiveCell.Offset(-7, 0)
ActiveCell.Offset(-6, 0).Address & ")/" & ActiveCell.Offset(-7, 0).
'Asset Quality Index
ActiveCell.Formula = "=" & (((" & ActiveCell.Offset(-13, 0).Address & "+" &
ActiveCell.Offset(-11, 0).Address & ")/(((" & ActiveCell.Offset(-
ActiveCell.Offset(-12, 1).Address & ")/" & ActiveCell.Offset(-11, 1)
ActiveCell.Offset(1, 0).Select
ActiveCell.Formula = "=" & (" & ActiveCell.Offset(-9, 0).Address & "/" & Ac
ActiveCell.Offset(1, 0).Select
'Depreciation Index
ActiveCell.Formula = "=" & (((" & ActiveCell.Offset(-5, 1).Address & "/" & (" &
ActiveCell.Offset(-14, 1).Address & ")/(" & ActiveCell.Offset(-5, 0)
ActiveCell.Offset(-5, 0).Address & "+" & ActiveCell.Offset(-14, 0).
ActiveCell.Offset(1, 0).Select
'SGA Index
ActiveCell.Formula = "=" & (((" & ActiveCell.Offset(-9, 0).Address & "/" & A
ActiveCell.Offset(-9, 1).Address & "/" & ActiveCell.Offset(-11, 1).
ActiveCell.Offset(1, 0).Select
'Leverage Index
ActiveCell.Formula = "=" & (((" & ActiveCell.Offset(-13, 0).Address & "+" &
ActiveCell.Offset(-15, 0).Address & ")/(((" & ActiveCell.Offset(-13,
ActiveCell.Offset(-14, 1).Address & ")/" & ActiveCell.Offset(-15, 1)
ActiveCell.Offset(1, 0).Select
'Total Accruals/TA Index
ActiveCell.Formula = "=" & (" & ActiveCell.Offset(-10, 0).Address & "-" & A
ActiveCell.Offset(-16, 0).Address
'Calculate the Benish M Score and Probability of Manipulation

```

The program currently creates four tabs: accruals, solvency, profitability, and valuation. At the end of the program a message box appears notifying the user that the program is done. The key ratios and figures are highlighted to more easily catch the attention of the user in the mix of all the other numbers on the screens. Below are screenshots of each of the four tabs:

The Benesh model computes the probability of manipulation in accounting accruals. If the probability of manipulation is above 10% then there may be cause for concern.

Benesh Model	GOOG		YHOO		MSFT		AAPL	
M Score	(2.88)		(2.45)		(2.54)		(2.40)	
Manipulation Prob	0%		1%		1%		1%	
	GOOG		YHOO		MSFT		AAPL	
	CY	PY	CY	PY	CY	PY	CY	PY
Accounts Receivable	5,261,000,000	3,845,000,000	1,028,900,000	1,000,000,000	15,198,000,000	13,731,000,000	11,560,000,000	
Current Assets	41,562,000,000	29,167,000,000	4,345,548,000	4,550,000,000	55,676,000,000	44,988,000,000	41,678,000,000	
PP&E (net)	7,759,000,000	4,845,000,000	1,653,422,000	1,420,000,000	7,630,000,000	7,777,000,000	4,768,000,000	
Total Assets	57,851,000,000	40,497,000,000	14,928,104,000	14,930,000,000	86,113,000,000	116,371,000,000	75,183,000,000	
Current Liabilities	9,996,000,000	2,747,000,000	1,625,872,000	1,710,000,000	26,147,000,000	27,970,000,000	20,722,000,000	
Long-term Debt	-	-	142,799,000	800,000,000	4,939,000,000	-	-	
Sales	29,321,000,000	23,651,000,000	6,324,651,000	6,460,000,000	62,484,000,000	108,249,000,000	65,225,000,000	
Cost of Goods Sold	10,417,000,000	8,844,000,000	2,627,545,000	2,870,000,000	12,395,000,000	64,431,000,000	39,541,000,000	
Selling & Admin. Exp	4,761,000,000	3,652,000,000	1,752,823,000	1,825,702,000	17,277,000,000	7,599,000,000	5,517,000,000	
Income from Cont. Operations	8,505,000,000	6,520,000,000	1,231,663,000	597,992,000	18,760,000,000	25,922,000,000	14,013,000,000	
Cash Flow from Operations	11,081,000,000	9,316,000,000	1,240,190,000	1,310,346,000	24,073,000,000	37,529,000,000	18,595,000,000	
Depreciation	1,396,000,000	1,524,000,000	682,509,000	738,855,000	2,673,000,000	1,814,000,000	1,027,000,000	
Days Receivables Index	1.10		1.05		1.03		0.72	
Gross Margin Index	0.97		0.95		1.03		0.97	
Asset Quality Index	0.92		1.00		0.89		1.43	
Sales Growth Index	1.24		0.98		1.12		1.66	
Depreciation Index	1.57		1.17		1.02		0.94	
SG&A Index	1.05		0.98		0.94		0.83	
Leverage Index	2.55		0.98		1.04		0.87	

The Altman Z Score is a model that predicts bankruptcy in the short term. The other ratios on the right hand side are just common solvency ratios and can be compared across firms.

Altman Z Score	Coefficients		1.2		1.4		3.3		0.6		1	
	Numerator	CA-CL	RE	EBIT	share*price	Sales						
	Denominator	TA	TA	TA	TL	TA						
LF		154,942,500	(184,854,500)	6,218,000	374,572,700	432,564,000	LF	25.59	EBIT/Interest Exp			
		299,737,500	299,737,500	299,737,500	100,608,500	299,737,500		2.65	Current Ratio			
	3.50 Z_Score	0.62	(0.86)	0.07	2.23	1.44		-	LT Debt/Equity			
	0.01 Prob_2_yrs											
HAS		1,365,696,000	2,849,433,000	589,832,000	4,697,451,600	4,002,161,000	HAS	7.18	EBIT/Interest Exp			
		3,995,059,000	3,995,059,000	3,995,059,000	2,389,963,000	3,995,059,000		2.34	Current Ratio			
	4.08 Z_Score	0.41	1.00	0.49	1.18	1.00		0.79	LT Debt/Equity			
	0.00 Prob_2_yrs											
JAKK		369,720,500	96,359,500	56,474,000	488,924,800	747,268,000	JAKK	8.39	EBIT/Interest Exp			
		633,749,500	633,749,500	633,749,500	241,491,000	633,749,500		3.07	Current Ratio			
	3.60 Z_Score	0.70	0.21	0.29	1.21	1.18		0.22	LT Debt/Equity			
	0.00 Prob_2_yrs											
MAT		1,685,036,500	2,530,075,500	911,664,000	9,570,249,000	5,856,195,000	MAT	14.06	EBIT/Interest Exp			
		5,099,144,000	5,099,144,000	5,099,144,000	2,519,357,500	5,099,144,000		2.13	Current Ratio			
	5.11 Z_Score	0.40	0.69	0.59	2.28	1.15		0.32	LT Debt/Equity			
	0.00 Prob_2_yrs											

This profitability tab shows a decomposition of return on equity. The key behind this analysis is to identify the return from operating assets and look at operating profit and asset turnover to determine what makes up their returns to investors.

	GE	MMM	SI	DHR
Return on Equity (ttm)	12%	26%	23%	13%
Return on Equity	10%	29%	21%	14%
Net Operating Profit After Tax	29,068,649,916	4,364,000,000	11,485,490,484	1,913,775,000
Net Operating Assets	181,487,000,000	19,682,000,000	59,911,000,000	16,086,151,000
Return on Net Operating Assets	14%	24%	19%	12%
Net Operating Profit Margin	19%	16%	12%	14%
Net Operating Asset Turnover	0.75	1.44	1.64	0.86

This valuation tab uses a market multiple theory to value the price of a stock assuming that the competitors' stock prices and ratios reflect true economics. The user should beware of the mean share price predicted in the red section as it takes an average of all of the estimates. Consideration needs to be given to the standard deviation of this average. It may be better for the user to use their own intuition and select the most comparable firm and use that single estimate for the valuation of the company.

	P/E	Forward P/E	Price/Book
GE	1.31	10.46	1.42
MMM	5.88	12.81	3.4
SI	9.33	8.34	2.09
DHR	3	14.04	1.97
Averages:	6.69	12.93	28.01

	Net Income	Shares Outstanding	Equity Bookvalue
GE	11,644,000,000.00	10,560,000,000.00	118,936,000,000.00

	Stock Value Per Share
MMM	6.48
SI	10.29
DHR	3.31
Averages:	6.69

	GE
Actual Share Price:	16.31
Mean Share Price:	15.88
Standard Deviation:	10.76
Median Share Price:	14.12

I am continually working on the program to create more tabs for the user to view and interpret. This program's purpose is to automate the process of performing these analyses.

Learning and Conceptual Difficulties

Find Method: A large portion of my project required using the find method to locate a number on a worksheet and bring it into a summary area. I ran into some difficulties when I wanted to do the opposite of a find, and do a if not found statement. The problem with this is that when you don't find something an error occurs. I used an error resume next, but this still causes problems because the find is located in a loop. I learned that it is much easier to find something else that is unique. For example, when the user inputs a stock ticker that is not valid on Yahoo! Finance, I have a find to locate if specific text is found that is only on a webpage where a stock ticker is not valid. Another aspect of the find method that I learned is that you can start your search in different locations and the default is to start after the active cell. I also learned that you can make it search the cells in excel in different directions. This helped me in a couple areas where I knew that if I started my search in cell "A1" that it would work.

Option Compare Text: I learned that option compare text is very useful when using multiple finds and in strings throughout a program. I have option compare text at the top of my code and it solves many little issues that kept popping up.

Professor Allen's IE Agent: I am so thankful for the internet explorer agent that Professor Allen provided to the class. I found out that a web query on Google's web pages doesn't pull in everything that you see in a web browser. I had to use the IE agent to navigate on the webpage to find the stock tickers of the 3 most comparable firms to the first stock ticker inputted. Without the IE agent I'm sure that I wouldn't have been able to do this.

Making Robust Loops: I had to learn how to use loops to run through a process multiple times and then slightly change the process but continue to run it. I had to make everything in this project perform itself at least 4 times, once for each company. From working on this project I know feel fully confident in using not only do loops but also for next loops. There is one instance in my program where I am running a loop within a loop, within a loop.

Using Arrays: Arrays can be very useful when using large numbers of variables. It took me a while playing around with declaring string arrays to make sure that it is doing it correctly. I had an idea to create a large three dimension array for my financial variables. The three dimensions being number, year, and company. I couldn't get it to work and found it easier to just create multiple arrays.

Importance of Commenting: After working on my project for over two weeks I found myself looking at lines of code that I had written but didn't know what their purpose or function was. As the hours of coding began to build up I realized that my code was getting very large in length. I went through the entire program and tried to put in a one line comment above each section of code to explain the purpose of that section of code. I now am in a better habit of commenting while writing code to not only explain to someone else but also to remind myself what the code is doing.

Naming Cells: I struggled with referencing cells from one sheet to another in a manner that is robust and easy to use in loops. I met with Professor Allen and he recommended a simple solution. Uniquely naming the cells that held the financial data would allow me to call them easily in formulas on different worksheets. This way I could show the formula in the cell for the user but not have to hardcode the formula. My program runs a loop through all of the financial figures to name them based on the company, year, and title of the figure.

User Visibility: I came to a point in my program where I had to decide how I would allow the user to be able to see and understand the calculations being made. I didn't want the user to question where the numbers were coming from on the analyses tabs. This part of the code took me a while but I created it so that the user can see the formulas while maintaining a robust code that will run through a loop.

Bad User Input: I ran into the problem of the user putting in a stock ticker that isn't valid on Yahoo! Finance. The problem was that the entire code would still run for a couple minutes and then the end result would be garbage. I wanted a way to end the code if the first web query didn't return financial data from Yahoo! Finance. I ended up doing an if statement around the first web query that will display a message box telling the user that the stock ticker isn't valid. This then ends the program.

Variability in Stock Tickers: When testing the program in the early stages of creating the web queries and finding the numbers to fill the input tab I used generally know companies. The more I tried different, small, or unique stock tickers the more errors I encountered. For example many times a comparable company doesn't have all of the desired information on Yahoo! Finance. I had to create error traps around each different web query so that the program will continue to function and run properly after encountering a bad comparable firm. Another example is one letter stock tickers. I found that they sometimes do funny things with the finds and searches I perform because they found multiple cases of things that I thought were unique on a worksheet. Foreign companies and tickers with numbers in their name also create problems. More fixes are required to fully enable a user to select any publicly traded company to analyze.

Long Program: From the original idea to what it is now the program has grown a lot. The time that it currently takes is a couple minutes depending on your internet connection. I ran into the problem of what I should show the user while the program is running. I originally had the screen updating turned off so but this because a problem because sometimes excel would say that it is not responding and then the user may force quit the program while it is running. The simply solution is to just leave on the screen updating so that the user sees step by step what is happening even if it may cause a seizure. Another idea that I have is to implement a progress bar. There are two ways that I've thought about doing this. One, to show a user form that updates while the program is running or configure excel's status bar on the bottom bar to show the status of the program. I couldn't easily figure this out as a last minute item and will have to work on it more to implement it.