Comparable Valuation Application

End User Modelling

Taylor Carloss, Matt Hunter,
Dustin Maleganeas, Kelvin Mu,
Dominic Leung, Peter Zhu

INTRODUCTION

Comparable Valuation Overview

Valuation is one of the core duties of a financial analyst, as the analysis answers the simple yet vital question of "What is this company worth?" Comparable Company Analysis (**Comparable Analysis**) is one of the three main methodologies used to value a company. This technique compares the value of a company to the values assessed by the market for similar companies. Compare to the **Discounted Cash Flow** (DCF) Method, Comparable Analysis is elegantly simplistic because it does not require analysts to make wild assumptions such as those regarding the company's future growth rate.

To perform a Comparable Analysis, the first step is to identify companies similar to the one being analyzed and obtain their market values. Since the absolute prices cannot be compared directly, they must be converted into standardized values (commonly known as **valuation multiples**), which will be used to value the target company. As the last step, valuation derived from the analysis is compared to the current market price of the target company to judge whether it is undervalued or overvalued.

There are many types of valuation methods involving several sets of metrics. For equities, the most common valuation metric is the **Price/Earnings ratio** (P/E). Other popular valuation metrics include: **Price/Book Value** (P/B), **Price/Sales** (P/S), **Enterprise Value/EBIDTA** (EV/EBITDA) and **Enterprise Value/Revenue** (EV/REV).

Valuation Challenges & How this Application Can Help

While performing Comparable Analysis, analysts often run into two challenges. The first challenge is to identify companies that are comparable to the one being analyzed, and the second challenge is to perform this time-consuming valuation efficiently.

To identify comparable companies, analysts need to filter companies through a list of criteria such as market capitalization or historical growth rate. This process involves going through a large list of potentially comparable companies, and apply screening tests to each and every one of them. After obtaining a list of companies that met all the selection criteria, analysts must dive into each company's financial statements to locate ratios and other information needed to complete the valuation. The entire process is extremely tedious and could cause excruciating pain to the analysts.

Due to the repetitiveness of the calculations and the simplicity of the mathematics inherent in this valuation method, VBA is an ideal solution to automate the process. This application helps the user identify comparables by filtering through companies using criteria established by the user. It retrieves information needed for valuation for all the comparable companies, and automatically calculates the value of the target company using its comparables' multiples under six common valuation methods.

FUNCTIONALITY OF THE APPLICATION

The Application has the following functionality:

1. It first asks the user for the ticker symbol of the company to be analyzed (**target company**). It will use this information to retrieve a list of ticker symbols for all publicly listed companies within the same industry as the target company. These companies would be considered as **default comparables**.

2. Next, using the list of default comparables, the application will retrieve their respective key statistics from **Yahoo! Finance** key statistics webpage. These information are needed for (1) applying the screener tests and (2) calculating relative values for the target company.

3. Users can choose to narrow down the list of default comparables by filtering them through eight screeners:

   1) Market Cap   4) Net Income   7) Quarterly Earnings
   2) Revenue   5) Profit Margin   Growth
   3) EBITDA   6) Operating Margin   8) Total Debt/Equity

4. For each screener, user can either enter a percentage value (%), or leave it blank. If user enters a value into a particular screener, companies with value outside the prescribed percentage range in relation to the target company will be considered non-comparable. If user leaves the screener blank, the application will not apply that particular screener in the selection process. Only companies that pass all selected screeners would be considered **valid comparables**.

Figure 1

Using the information in Figure 1, the application will retrieve a list of ticker symbols for companies that meet the following criteria:

1) Be in the same industry as the Toronto-Dominion Bank (TD)
2) Has revenue less than 175% but higher than 25% of TD's revenue
3) Has profit margin less than 150% but higher than 50% of TD's profit margin

5. The application then displays a selection of key statistics for the target company along with each of its valid comparables, as shown in Figure 2:

Figure 2

Financial Information of Comparable Companies
as of Apr 11, 2009

Company	Revenue (M)	EBITDA (M)	Net Earning (M, ttm)	Net Earning (M, forward	...		EV / REV (ttm)	Diluted EPS	ted Shares Outstanding
TD	11,320,000.00	-	2,850,000.00	3,581,111.11	...	-	4.26	3.55	802,816.90
BNS	9,460,000.00	-	2,470,000.00	2,886,975.24	...	-	2.09	2.48	995,967.74
PNC	5,610,000.00	-	861,000.00	1,910,614.53	...		8.27	2.46	350,000.00
RY	17,000,000.00	-	3,470,000.00	5,254,335.26	...	-	4.27	2.58	1,344,961.24

Abbreviation Guide:

M = Millions

mrq = Most Recent Quarter

ttm = Trailing Twelve Months

lfy = Last Fiscal Year

fye = Fiscal Year Ending

6. These key statistics will be automatically used to calculate the relative values for the target company using its comparables' multiples under six valuation methods, as shown in Figure 3. The user needs to manually enter the target company's "Minority Interest" and "Preferred Shares" into the Input Cells on the top right of the sheet to complete the "EV/EBITDA" and "EV/Revenue" valuation. These two figures must be manually entered because they are not readily available from the Yahoo! Finance key statistics webpage.

Figure 3

Comparable Valuation
as of Apr 12, 2009

Input:

	Minority Interest (M)	-
	Preferred Shares (M)	-

Company	P/E (ttm)	P/E (forward)	P/S (ttm)	P/B (mrq)	EV / EBITDA (ttm)	EV / REV (ttm)
BNS	10.68 x	9.29 x	2.68 x	1.62 x	0.00 x	(2.09 x)
	$37.91	$41.44	$37.79	$61.36	$0.00	$70.79
PNC	15.64 x	8.95 x	2.53 x	0.81 x	0.00 x	8.27 x
	$55.52	$39.92	$35.67	$30.68	$0.00	$216.87
RY	12.54 x	8.65 x	2.52 x	1.73 x	0.00 x	(4.27 x)
	$44.52	$38.58	$35.53	$65.52	$0.00	$40.05
Average	12.95 x	8.96 x	2.58 x	1.39 x	0.00 x	0.64 x
	$45.98	$39.98	$36.33	$52.52	$0.00	$109.24
Median	12.54 x	8.95 x	2.53 x	1.62 x	0.00 x	(2.09 x)
	$44.52	$39.92	$35.67	$61.36	$0.00	$70.79

7. This application only works with open Internet access. If user tries to run the application without Internet, he/she must press the "CLEAN CACHE" button before proceeding.

THE VBA CODE IN THE MODULE

The **COMPARABLE_MODEL_MAIN** sub is attached to the "COMPARABLE MODEL" button in the Control Panel sheet. The purpose of this sub is to simply define the ranges for both the ticker symbol and the screener criteria, and then calls on the **RNG_COMPARABLE_TRANSACTIONS_FUNC** function to create the output worksheet that contains all the pre-formatted, pre-formulated calculations.

Before the **RNG_COMPARABLE_TRANSACTIONS_FUNC** function can formulate any calculations in VBA, it must first find out a list of comparables to the target company and their selected key statistics. This is where the **COMPARABLE_DATA_TABLE_FUNC** function comes in to play.

COMPARABLE_DATA_TABLE_FUNC Code

As it was explained earlier, the first step to look up comparables is to find out what industry the target company belongs to. This goal is accomplished by the following code:

```
URL_STR = "http://finviz.com/export.ashx?"
SRC_URL_STR = URL_STR & "v=152&ft=1&t=" & TICKER_STR & "&ta=1&p=d&r=1&c=4"
```

By utilizing the **SAVE_WEB_DATA_PAGE_FUNC** function developed by Professor Nicolas Fermin Cota, the application retrieves from **finviz.com** a string that contains the industry name of the target company preceded by the "Industry" heading. To extract only the industry name, a few steps are required. First all the irrelevant characters from the string must be removed by replacing them with blanks. Then the "Industry" heading needs to be removed from the string so only the industry name will be preserved. Lastly, the application will end itself if an industry name could not be found, in which case the string would contain only the heading and nothing else.

```
DATA_STR = SAVE_WEB_DATA_PAGE_FUNC(SRC_URL_STR)
If DATA_STR = "--" Then: GoTo ERROR_LABEL
DATA_STR = Replace(DATA_STR, Chr(34), "")
For i = 8 To 14: DATA_STR = Replace(DATA_STR, Chr(i), ""): Next i 'Remove HTML Syntax
If DATA_STR = "Industry" Then: GoTo ERROR_LABEL 'Wrong ticker symbol
DATA_STR = Trim(Replace(DATA_STR, "Industry", ""))
```

The function continues by converting the industry name into an industry tag that could be recognized by finviz.com. To do this it calls upon the **FINVIZ_INDUSTRY_TAG_FUNC** function, which contains a library of all the industries along with their associated industry tag. One limitation of this method is that this library is not dynamic and will need to be updated over time as new industries are added, or if the industry tags are ever changed.

```
DATA_STR = FINVIZ_INDUSTRY_TAG_FUNC(DATA_STR) 'Getting Industry Tag
```

With the industry tag converted, the function proceeds to retrieve a list of ticker symbols for all companies in that industry. Besides cleaning up the string by removing certain characters, it is necessary to separate all the ticker symbols by rows to form an array. Since the list of ticker symbols retrieved from finviz.com contains the heading "Ticker" in its first row, the heading needs to be replaced with the ticker symbol of the target company.

```
SRC_URL_STR = URL_STR & "v=152&f=" & DATA_STR & "&ft=1&ta=1&p=d&r=1&c=1" 'Getting Competitors
DATA_STR = Trim(SAVE_WEB_DATA_PAGE_FUNC(SRC_URL_STR))
DATA_STR = Replace(DATA_STR, Chr(13), "")
DATA_STR = Replace(DATA_STR, Chr(34), "")
DATA_STR = Split(DATA_STR, Chr(10)) 'turn string into rows
DATA_STR(LBound(DATA_STR)) = TICKER_STR 'replace first row heading with reference symbol
```

The *DATA_STR* array now represents a list of ticker symbols that begins with the target company and includes every company in that particular industry. Since the original list already contains the target company, there is now a duplicate that needs to be removed, using the **VECTOR_REMOVE_ DUPLICATS_FUNC** function which preserves the first unique examples of any data. Finally, the application now has a list of company ticker symbols that will be used to retrieve key statistics from Yahoo! Finance, using another function developed by Professor Nicolas Fermin Cota – the **YAHOO_KEY_STATISTICS_FUNC** function.

```
ReDim TEMP_MATRIX(1 To UBound(DATA_STR), 1 To 1)
For i = 1 To UBound(DATA_STR)
    TEMP_MATRIX(i, 1) = DATA_STR(i - 1)
Next i
TEMP_MATRIX = VECTOR_REMOVE_DUPLICATES_FUNC(TEMP_MATRIX)
TEMP_MATRIX = YAHOO_KEY_STATISTICS_FUNC(TEMP_MATRIX) 'Getting fundamental data
```

When called upon, this function returns a *TEMP_MATRIX* array which contains the ticker symbol and 58 accounts of key statistics from Yahoo Finance for the reference company as well as every other company in the industry (See Exhibit 1). This function can also be called independently by clicking the "COMPARABLE DATA" button on the Control Panel sheet.

COMPARABLE_SCREENER_FUNC Code

The **COMPRABLE_SCREENER_FUNC** function takes the two-dimensional array from the **COMPRABLE _DATA_TABLE_FUNC** function and applies the screening criteria as specified by the user. It reduces the list of default comparable companies to only valid comparables which passed the screening tests.

The section below prepares the variables and arrays for the screening process. *DATA_RNG* is the array passed from the **COMPRABLE_DATA_TABLE_FUNC** function and *SCREENER_RNG* is the list of screening criteria. The *INDEX_ARR* stores the locations (column number) of where each criteria stat is stored in the larger *DATA_MATRIX* array.

```
DATA_MATRIX = DATA_RNG
NROWS = UBound(DATA_MATRIX, 1)
SCREENER_VECTOR = SCREENER_RNG
If UBound(SCREENER_VECTOR, 1) = 1 Then
    SCREENER_VECTOR = MATRIX_TRANSPOSE_FUNC(SCREENER_VECTOR)
End If
INDEX_ARR = Array(2, 17, 21, 22, 13, 14, 24, 28)
NCOLUMNS = UBound(INDEX_ARR)
```

The screening process is done by examining each company from the *DATA_MATRIX* list and determining if they pass all selected screening criteria. It is assumed at the onset that all companies have passes the tests, unless the test result indicates otherwise. The screening begins from the 3rd row of the *DATA_MATRIX* array as the 1st and 2nd row contain headings and the target company respectively. This corresponds with the first company on the *DATA_MATRIX* list and each screening criteria is tested before moving on to the next comparable.

While testing the validity of the default comparables, there are a few cases in which we would skip over certain screener tests. One situation would be when the user intentionally left the screener blank. Another situation would be when the target company has a zero or non-value for the criteria to be tested, which would disable any comparisons. Lastly, if a default comparable has a zero or non-value for a selected screener criteria, it would fail the test and be excluded from the list of valid comparables.

```
For i = 3 To NROWS
    MATCH_FLAG = True
    For j = 1 To NCOLUMNS
        If SCREENER_VECTOR(j, 1) = "" Then: GoTo 1984
        k = INDEX_ARR(j)
        If DATA_MATRIX(2, k) = "" Or DATA_MATRIX(2, k) = 0 Then: GoTo 1984
        If DATA_MATRIX(i, k) = "" Or DATA_MATRIX(i, k) = 0 Then
            MATCH_FLAG = False
            Exit For
```

The math behind the screener is simple: it compares the percentage inputted by user with the absolute value of the percentage difference between the comparable's value and the target company's value. If the difference is greater than the screening threshold, the company would fail the test, and the function would skip ahead to test the next company on the list.

```
    ElseIf Abs(DATA_MATRIX(i, k) / DATA_MATRIX(2, k) - 1) > SCREENER_VECTOR(j, 1) Then
        MATCH_FLAG = False
        Exit For
    End If
```

If a company passes all the screening criteria, its position (row number) in the *DATA_MATRIX* array is stored in the *SYMBOLS_ARR* array. At this point, *SYMBOLS_ARR* represents a list of all valid comparables, and it does not include the target company.

```
    If MATCH_FLAG = True Then
        l = l + 1
        ReDim Preserve SYMBOLS_ARR(1 To l)
        SYMBOLS_ARR(l) = i
    End If
```

Furthermore, this function selectively reduces the total number of key statistics retrieved from Yahoo! Finance to only what is needed by the user for their analysis (see Exhibit 2). After it identifies the location (column number) of the needed information, the transfer of data from *DATA_MATRIX* to *TEMP_ MATRIX* is completed in two steps. The first step transfers all the relevant key statistics headings as well as the statistics of the target company. It then turns to the dynamic *SYMBOL_ARR* and transfers all the relevant key statistics for those valid comparables.

```
INDEX_ARR = Array(1, 2, 4, 5, 7, 8, 9, 10, 11, 12, 17, 21, 22, 23, 25, 27)
NROWS = l
NCOLUMNS = UBound(INDEX_ARR)
ReDim TEMP_MATRIX(1 To NROWS + 2, 1 To NCOLUMNS)
For jj = 1 To NCOLUMNS
    j = INDEX_ARR(jj)
    TEMP_MATRIX(1, jj) = DATA_MATRIX(1, j)
    TEMP_MATRIX(2, jj) = DATA_MATRIX(2, j)
    For ii = 1 To NROWS
        i = SYMBOLS_ARR(ii)
        TEMP_MATRIX(ii + 2, jj) = DATA_MATRIX(i, j)
    Next ii
Next jj
```

RNG_COMPARABLE_TRANSACTIONS_FUNC Code

Now with all needed information available from *TEMP_MATRIX*, the **RNG_COMPARABLE_TRANSACTIONS_FUNC** function can now create a pre-formatted, pre-formulated output worksheet which calculates the valuations of the target company.

There are three important functionalities embedded within this function that are worth mentioning. The first one being the mechanism used to determine the location (row and column coordinates) for different ranges. This is important because in a dynamic array the location of ranges would change depending on the value of the variable – in this case, the number of valid comparables for the target company. The second key functionality is the function's ability to calculate valuations by referring to specific ranges within the worksheet. In other words, it is able to insert formulas into the calculating ranges and makes it extremely easy for user to review the inputs used for the calculations. The last important functionality calculates the average and the median of the valuations derived from using different comparables' valuation multiples. The calculation is challenging because not only does the function need to locate specific ranges within a dynamic array, it also needs to exclude valuations with no values from the computation.

The section below shows an example of the mechanism locates ranges within a dynamic array. This part of the code tries to identify the location of the heading "Financial Information of Comparable Companies". Its location is defined as $(9 + h * (NSIZE + 2), 2)$ as shown in Exhibit 3, where NSIZE represents the number of valid comparables (NSIZE = 9 in this example), and h represents the number of rows between two comparables (h = 3 in this example).

```
Set FORMAT_RNG = DST_WSHEET.Cells
GoSub RNG_FORMAT_CELLS
Set DST_RNG = FORMAT_RNG.Cells(1, 1)
'-------------------------------------------------------------------------------------------
h = 3
n = 9 + h * (NSIZE + 2)
Set DST_RNG = DST_RNG.Offset(n - 1, 1)
```

When the function calculates the average and the median for all different valuations, it first looks for the last comparable on the list because it is the easiest to determine now that the definitive number of valid comparable is known at this point. The function records the address of that last comparable, and works backwards to include the valuations of every other valid comparable. This would work all wonderful except when certain valuation multiples have values of zero because they were unavailable from the Yahoo! Finance key statistics website. When this happens, those valuations with zero values should be excluded to avoid skewing the calculation output. The function loops through all valid comparables starting from the bottom up, and it will register the cell address of the first non-zero valuation. The counter *g* is used to keep track of the position of the first non-zero valuation, so the function knows where to continue once the starting position has been identified.

```
        For j = 2 To 7
           g = 0
           ADDRESS_STR = "0"
           For m = 1 To NSIZE
              If .Cells(3 + h * (NSIZE) - h * m, j).Value <> 0 Then
                 ADDRESS_STR = .Cells(3 + h * (NSIZE) - h * m, j).Address
                 g = g + 2
                 GoTo 1
              End If
              g = g + 1
           Next m
1:
           For m = g To NSIZE
              If .Cells(3 + h * (NSIZE) - h * m, j).Value <> 0 Then
                 ADDRESS_STR = ADDRESS_STR & "," & .Cells(3 + h * (NSIZE) - h * m, j).Address
              End If
           Next m
           .Cells(h * (NSIZE + i) + 3, j).Formula = "=AVERAGE(" & ADDRESS_STR & ")"
```

SUMMARY

This application has been developed to save financial analysts time while performing fundamental relative valuations. This project has been ambitious, and a lot of code has been written to achieve something of this complexity so it is not always straightforward. For the future, there are definitely opportunities to improve the application either by simplifying the code or to remove some of its current limitations. For one, the author of this program hopes to find a way to automatically insert the value for "Minority Interests" and "Preferred Shares" instead of having to insert the value manually.

Exhibit 1

Symbol	Market Cap	Enterprise Value	Trailing P/E	Forward P/E	...	Dividend Date	Ex-Dividend Date	Last Split Factor	Last Split Date
TD	32230000	-48270000	10.69	9	...	29/04/2009	01/04/2009	21	04/08/1999
BAC	61130000	415660000	17.24	7.46	...	26/03/2009	04/03/2009	21	30/08/2004
BK	34070000	-52590000	24.67	10.48	...	02/02/2009	21/01/2009	21	14/08/1998
BNS	26820000	-19730000	10.68	9.29	...	27/04/2009	03/04/2009	21	29/04/2004
C	16650000	61440000	0	0	...	26/02/2009	29/01/2009	43	28/08/2000
CM	1950000	-9790000	0	7.76	...	27/04/2009	25/03/2009	0	0
JPM	123070000	-51640000	23.94	12.89	...	29/04/2009	02/04/2009	32	12/06/2000
KEY	4080000	26570000	0	23.57	...	12/03/2009	27/02/2009	21	09/03/1998
OFG	134320	4270000	6.14	2.82	...	14/04/2009	30/03/2009	1110	29/12/2004
OLCB	3100	22540	0	0	...	0	0	0	0
PNC	17100000	46430000	15.64	8.95	...	23/04/2009	08/04/2009	21	16/11/1992
RY	45450000	-72560000	12.54	8.65	...	21/05/2009	22/01/2009	21	07/04/2006
STI	5040000	24800000	6.63	14.88	...	15/03/2009	26/02/2009	21	22/05/1996
TCB	1890000	6210000	14.66	12.97	...	26/02/2009	28/01/2009	21	07/09/2004
UBOH	30980	102300	7.03	0	...	12/03/2009	25/02/2009	0	0
WFC	83100000	330700000	28.05	10.06	...	28/02/2009	04/02/2009	21	14/08/2006

Exhibit 2
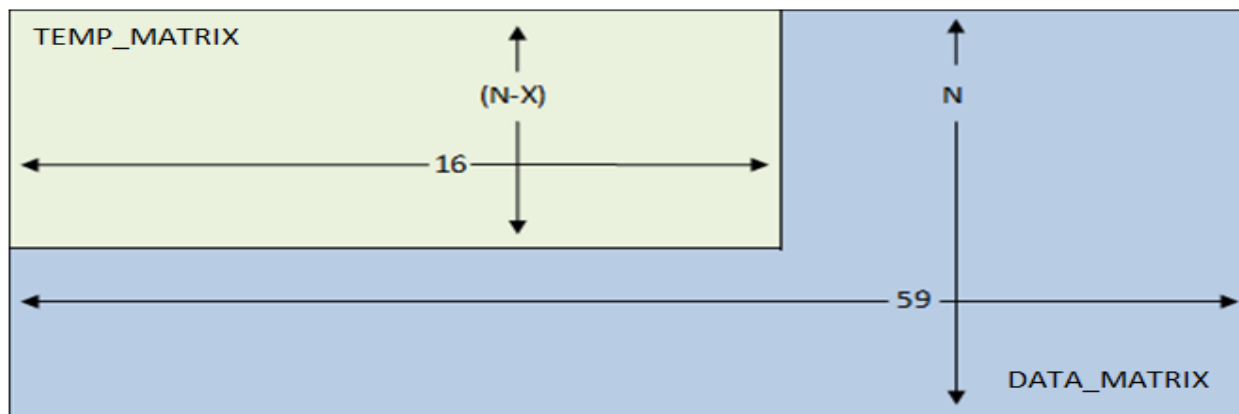
Note: The function reduces the size from [N, 59] to [(N-X), 17] where X is the number of companies that does not pass the screening process.

Exhibit 3

	A	B	C	D	E	F	G	H
1	1							
2	2	**Comparable Valuation**				Input:	Minority Interest (M)	-
3	3	as of Apr 11, 2009					Preferred Shares (M	-
4	4							
5	5	Company	P/E (ttm)	P/E (forward)	P/S (ttm)	P/B (mrq)	EV / EBITDA (ttm)	EV / REV (ttm)
6	6							
7	7	AE	-	-	0.01	0.76	5.82	0.01
8			-	-	0.23	12.44 -	10.02 -	32.28
9								
10	7+1*h	ANW	18.13	5.66	0.26	2.49	13.36	0.33
11			-	71.98	11.07	6.07	40.74	19.09 -	24.81
12								
31	7+(N-1)*h	RVEP	-	-	0.09	0.17 -	27.51	2.14
32			-	-	2.10	2.78 -	138.82	17.47
33								
34	7+(N)*h	Average	11.10	9.05	0.19	1.16	2.62	0.43
35			-	42.96	18.04	4.33	18.96 -	22.38 -	22.58
36								
37	7+(N+1)*h	Median	9.40	6.81	0.09	1.10	4.33	0.21
38			-	37.32	13.32	2.10	18.00 -	15.77 -	27.61
39								
40	7+(N+2)*h							
41								
42	9+(N+2)*h	**Financial Information of Comparable Companies**						
43		as of Apr 11, 2009						