Business Data Analysis Story

Business analytics (BA) is the process of collating, sorting, processing, and studying business data, and using statistical models and iterative methodologies to transform data into business insights. It involves skills, technologies, practices for continuous iterative exploration and investigation of past business performance to gain insight and to drive business planning. In particular, business analytics focuses on developing new insights and understanding of business performance based on data and statistical methods. The related area of business intelligence (BI) focuses on using a consistent set of metrics to both measure past performance and guide business planning, which is also based on data and statistical methods. It involves querying, reporting, online analytical processing (OLAP), and "alerts."

Business analytics makes extensive use of statistical analysis, including explanatory and predictive modeling, and fact-based management to drive decision making. It is therefore closely related to management science. Analytics may be used as input for human decisions or may utilized for automated decision making (ADM).

In other words, querying, reporting, OLAP, and alert tools can answer questions such as what happened, how many, how often, where the problem is, and what actions are needed. Business analytics can answer questions like why is this happening, what if these trends continue, what will happen next (that is, predict), what is the best that can happen (that is, optimize).

Analytics have been used in business since the management exercises were put into place by Frederick Winslow Taylor in the late 19th century. Henry Ford measured the time of each component in his newly established assembly line. But analytics began to command more attention in the late 1960s when computers were used in decision support systems. Since then, analytics have changed and formed with the development of enterprise resource planning (ERP) systems, data warehouses, and a large number of other software tools and processes.

In later years the business analytics have exploded with the introduction of computers. This change has brought analytics to a whole new level and has made the possibilities endless. As far as analytics has come in history, and what the current field of analytics is today many people would never think that analytics started in the early 1900s with Mr. Ford himself.

Pathways to Business Intelligence

The early tools used for queries on data and generating reports were all sold as “do-it-yourself” solutions. In the mid-1970s, several vendors began offering tools that allowed a non-programmer to delve in the world of data access and analysis. Nearly every vendor’s product set included internal, proprietary data formats. One of the compelling reasons for this was to give an end user the ability to create his own data and to place data into a form that was optimized for the tool.

Another reason was that the era of relational databases, such as DB2, had not yet established itself for common usage and implementation of end-user data, so vendors were forced to offer their own data solutions.

There are obvious difficulties with such data sources:

- They were closed and proprietary; they worked only with that vendor's tool.
- Extractions of sets of source data were normally required.
- These extractions were then out of sync with the customer's original source data.
- Most could not contain the volume of data needed.
- IT assistance was always required to pull information from the original source.
- Significant investment into these technologies could isolate and trap key data used within a tool that might later fall behind the technology curve.

There were significant numbers of customers quietly dabbling in these tools. Well, maybe *quiet* is not a good adjective. Many were *noisily* trying to make these technical miracles work for them. The majority of the systems and data being accessed were mainframe-based because that was where the majority of the data resided. The tools themselves tended to provide very powerful capabilities if you could learn to use them.

Many of these tools were command-line driven, and the interfaces provided were seldom something to write home about. However, they did offer some hope to the non-technical user, and many departmental specialists emerged who were capable of navigating the technical issues and difficulties in using these powerful but primitive tools.

One positive aspect for those learning and using these tools was the need to understand how data is stored and accessed. Departmental specialists also would learn how to handle the processing of data and the steps required performing calculations. For example, if data were not sorted in the proper order, specialists would often obtain strange results when calculating subtotals or producing totals by breaks. Sometimes, they would find that sorting took extremely lengthy processing because the source data had not been stored in physical record sequence. They simply had to learn some of the issues that the IT staff dealt with every day.

Needless to say, anyone in the early 1980s who expressed the opinion that PCs were merely toys and not to be taken seriously feels a little foolish today. At first, PCs seemed like quaint little versions of more powerful systems with some simple functions but little analysis or processing power. Then came the announcement of Lotus 1-2-3. The spreadsheet revolutionized the ability of individuals to perform their own analysis and computing.

In the late 1980s came the near lemming-like run to embrace client/server systems. The basic tenets behind this revolution were:
- Mainframes were expensive and passé.
- Data should reside on smaller, less expensive boxes.
- The logic and calculations took place on the server database and the end-user tools.
- Distributed processing would be the norm.

Many organizations had an eclectic mixture of mainframes, distributed systems, fixed- function terminals, several databases, and personal computers. Processing was fragmented across multiple systems, and there was data duplication everywhere. Getting the data into the new server in a form that was useful and timely could be a nightmare. Along with these client/server solutions came numerous analysis tools. The overwhelming majority of these tools were based on using SQL (Structured Query Language) as a base for asking data-related questions. Because the majority of

the data required for analysis was in non-relational format, the tedious and costly job of extracting it from the host and transferring it to the new servers for loading was a full-time job.
Several relational database vendors emerged in this era, and one very good aspect emerged. All implementations of SQL were not the same. The need to establish an open standard among all the vendor offerings of RDBMSs became paramount in the industry. As a result, the customers received some very important BI-related benefits to this cooperation among the vendors:

- The analytics tools supported multiple vendor DBMS offerings with a common language.
- The RDBMS vendors pushed each other to excel in enhancements within the various vendor offerings and to make these enhancements part of the open standards.
- Skills in relational technologies (SQL skills and others) were reasonably transportable from one system to the next.
- Some common ground emerged by which to evaluate databases and tools. Query scenarios were established for benchmarks that permit an intelligent comparison of providers' wares.

In the late 1980s and early 1990s, one intriguing but mercifully short-lived fad was to implement information warehousing (IW). Instead of transforming the existing data into new, useful information, the idea was to leave it where it was and access it from anywhere with any tool. Elaborate technologies emerged as many sought to define complex data relationships in order to access it by software and hardware "plumbing and wiring." Users could get to the data *in situ* and perform analysis. There were many negative aspects related to such an approach, including the following:

- Any anomalies or errors in the data were brought back as-is, and the users had to deal with them.
- Many BI applications require data from multiple, disparate sources that need to be matched and joined; thus, the complexity and sheer volumes of data were extreme.
- Validating and qualifying the results for accuracy was problematic. Most implementers were so relieved to get data back that they didn't care if the output was accurate and had no way to validate it anyway.
- Lack of performance was a huge problem.

The one very positive aspect of the IW approach was that everyone realized there was a very strong requirement for *metadata*. Because there were so many different and disparate sources and definitions, there had to be a way to define and understand not only the original data but also any new definitions and terms being applied.

The nightmarish aspect other than data-related issues was that many customers were convinced that they could snap any tool onto the IW infrastructure and pummel the data into submission. We were faced with a situation in which different users with different tools from different locations could all access the same data repository that may be replete with errors and anomalies.

If you contrast this approach with the data warehouse approach, we see that there are some common elements that carried over into today's approaches to BI, including:

- Definition of all source data and associated metadata

- A central repository for users to access data
- Concern that the end users must work from a common set of "math" for analysis.
- The current form of the data may not be amenable to BI analysis; thus, access in place may not be a very wise approach.

So, what is the optimal form of data for BI? I suggest that the data warehouse or data mart approach with a star schema topology is the best format for BI. Because we are going to take existing data and redefine it, why don't we also add the changes and embellishments (the math) as well?

The data warehouse or data mart is far more than just a reorganization of data. It also is much more than a "cleaner" version of existing data. It is an opportunity for you to deliver creative and new information that is oriented toward the analyses used in the business. If new values and calculations are used within the enterprise, there will never be a better time to add them.

The entire gamut of data-related functions (extract, cleanse, etc.) has become a set of standard and expected processes that are associated with data warehousing. Most individuals working with such projects can cite the steps by rote. This is goodness for the customer, because the many vendors wishing to offer data-related solutions understand that they must provide these functions or interoperate with the most popular providers of ETL (Extract, Transform and Load) tools.

All tools are not alike, and even similar tools will have their quirks. Query and reporting tools in particular can all begin to blur as you evaluate them. One of the dilemmas with tools is that you are constantly trying to match the features and functions with the source data to try to exhibit meaningful information to key end users. What is the proper output for a vice president of marketing that would most accurately reflect the data he needs to see? Should you present the analysis results in a pie chart? Would it be best to produce a bar chart? What is best to portray any results? Are you collecting and calculating any information that is going to change the business?

Can't someone deliver a suite of analytics that are predefined and germane to users holding specific positions within the corporation? Are there solutions out there that provide both "canned" and changeable options?

You might hear the terms KPI (key performance indicator) and dashboards applied to BI solutions. Most executives want to be in on the BI action as well. However, the rarified air level of data that they deal with is seldom produced in most BI environments. The primary reasons for this are the math they require and the coalescing of data from the multitude of sources needed.

Today's executive may receive some reports or charts from a tool that allows them to play some "What if?" scenarios. Their existing tool set may allow them to learn how to perform some changes in scenarios where the goal is to turn the red figures into yellow and preferably green.

We've come quite a way in the quarter century and more of BI-like activities. However, in many ways, we haven't traveled far at all. What appears to be lacking is the embracing of BI as a key part of all corporate strategies. What have we learned so far?

- Early user-friendly languages emerged to offer a bridge between end users and

the hostile IT environment establishing the concept of end-user computing.

- Centralized centers of competency were created to provide a means for end users to become productive quickly. The need to set corporate standards for analysis tools was one of the most significant benefits from these centers.
- With the era of client/server systems came the understanding that keeping data *in situ* may not be conducive to analysis; thus, reengineering of data into BI- friendly forms and formats was ideal. The most commonly accepted form of database was a relational store that supported SQL. The need to establish and adhere to standards for all vendors' SQL became a mantra.
- The Information Warehouse proved that accessing data in place is not always desirable, but capturing the metadata about existing information makes perfect sense. Before we transform current information, we need to know all we can about its current contents and form.
- Data Warehousing projects brought all the pertinent steps together for taking existing information sources and creating new, analysis-based data. It also proved that the tasks related to data transformation could be incredibly long and costly. The argument as to whether a warehouse or a mart is more appropriate continues. The most significant aspect of warehousing or "marting" is the realization that the back ends will probably remain and processes to transform and create new data stores must be automated. These are not one-time events.
- We are entering an era where packaged BI solutions are desired. One driving force behind these is the need to deliver sophisticated metrics and analyses to top management.

Functions for Calculating Descriptive Statistics

Use the following MATLAB® functions to calculate the descriptive statistics for your data.
Statistics Function Summary

Function	Description
max	Maximum value
mean	Average or mean value
median	Median value
min	Smallest value
mode	Most frequent value
std	Standard deviation
var	Variance, which measures the spread or dispersion of the values

The following examples apply MATLAB functions to calculate descriptive statistics:

Example 1 — Calculating Maximum, Mean, and Standard Deviation
Example 2 — Subtracting the Mean

Example 1 — Calculating Maximum, Mean, and Standard Deviation
This example shows how to use MATLAB functions to calculate the maximum, mean, and standard deviation values for a 3-by-3 matrix called t1. MATLAB computes these statistics independently for each column in the matrix.

In MATLAB

```
%Generate data in workspace
>> m=magic(3)
m =
    8    1    6
    3    5    7
    4    9    2
% Save data for future use
>> dlmwrite('t1.txt',m,'delimiter','\t')
>> clear
% Load the sample data
>> load('t1.txt')
>> t1
t1 =
    8    1    6
    3    5    7
    4    9    2

% Find the maximum value in each column
mx = max(t1);
% Calculate the mean of each column
mu = mean(t1);
% Calculate the standard deviation of each column
sigma = std(t1);
% The results are
mx =      8     9     7
mu =     5     5     5
sigma = 2.6458    4.0000    2.6458
%
% Get the row numbers where the maximum data values occur in each data column, specify a
%second output parameter indx to return the row index. For example:
[mx,indx] = max(count)
mx =  8     9     7
Index = 1     3     2
```

Example 2 — Subtracting the Mean

Subtract the mean from each column of the matrix by using the following syntax:

In MATLAB

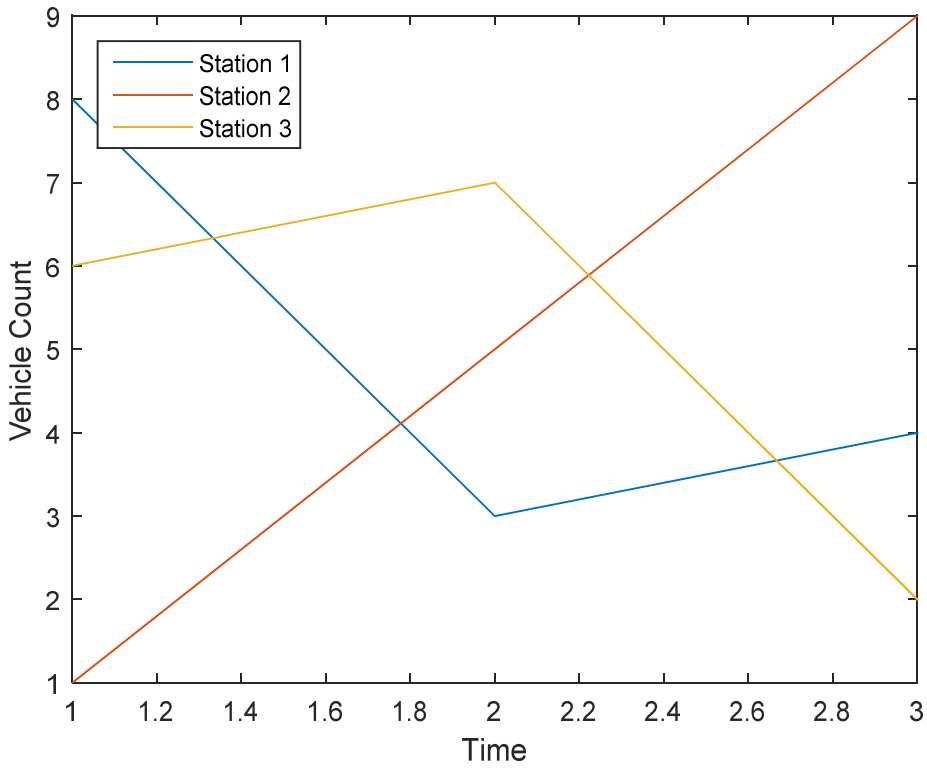
```
% Get the size of the t1 matrix
>> [n,p]=size(t1)
n = 3
p = 3
% Compute the mean of each column
mu = mean(count)
% Create a matrix of mean values by
% replicating the mu vector for n rows
>> MeanMAT=repmat(mu,n,1)
```

```
MeanMAT =
    5    5    5
    5    5    5
    5    5    5
% Subtract the column mean from each element
% in that column
>> x=t1-MeanMAT
x =
    3   -4    1
   -2    0    2
   -1    4   -3
%Note: Subtracting the mean from the data is also called detrending.
```

Example 3: Using MATLAB Data Statistics

The Data Statistics dialog box helps you calculate and plot descriptive statistics with the data. This example shows how to use MATLAB Data Statistics to calculate and plot statistics for a 3-by-3 matrix, called t1. The data represents how many vehicles passed by traffic counting stations on three streets.

```
% Define the x-values
>> x=1:n
x =  1     2     3
% Plot the data and annotate the graph
>> plot(x,t1)
>> xlabel('Time')
>> ylabel('Vehicle Count')
>> legend('Station 1','Station 2','Station 3','Location','Northwest')
```



Example 4: Business Intelligence and Decision Making
Gemini theatre sells ticket a INR 30 per ticket. Theatre capacity is 2000 seats. The current attendance averages 1000. Through a survey, Gemini found that for every x of INR .25 reduction in ticket price, theatre would have 20 more movie goers. Find the ticket price that will maximize revenues.

Solution:
The current price of ticket
p=50
For every decrease x of INR .25
p=30-.25x= 30-x/4
Increased attendance
A(x)=1000+20x
Theatre revenue is the attendance multiplied by attendance
R(x)=(1000+20x)*(30-.25x)
$R(x)=30000-250x+600x-5x^2$ (1)
$R(x)=30000+350x-5x^2$
In order to determine the stationary points on R(x), that points of minimum and max values.
RX=diff(Rx) and make it equal to zero
RX=350-10x=0
x=350/10=35
Substituting this value in (1)
R(35)=50000+350*75-5*75^2=50000+12250-5625=56625

After reducing the price by .25*35=8.75 from INR 50 to INR 41.25
A(35)=1000+20*35=1700

In MATLAB

```
>> p(x)=30-x/4;
>> A(x)=1000+20*x;
>> R(x)=30000+350*x-5*x^2;
>> DR=diff(R(x))
DR = 350 - 10*x
% Therefore x=35
% Reduced ticket price
>> p35=p('35')
p35 = 85/4
% Therefore, reduced ticket price is 21.25 from INR 30 reduced by 35/4=8.75
% Increased attendance
>> A35=A('35')
A35 = 1700
%Increased revenue
>> R35=R('35')
R35 = 36125
%From the current revenue of 30000 for 1000 tickets at INR 30
```

Example 5: From Business Data to Business Intelligence

tal2.txt file of Airline Passenger Counts in Thousands; Columns are years, rows are month

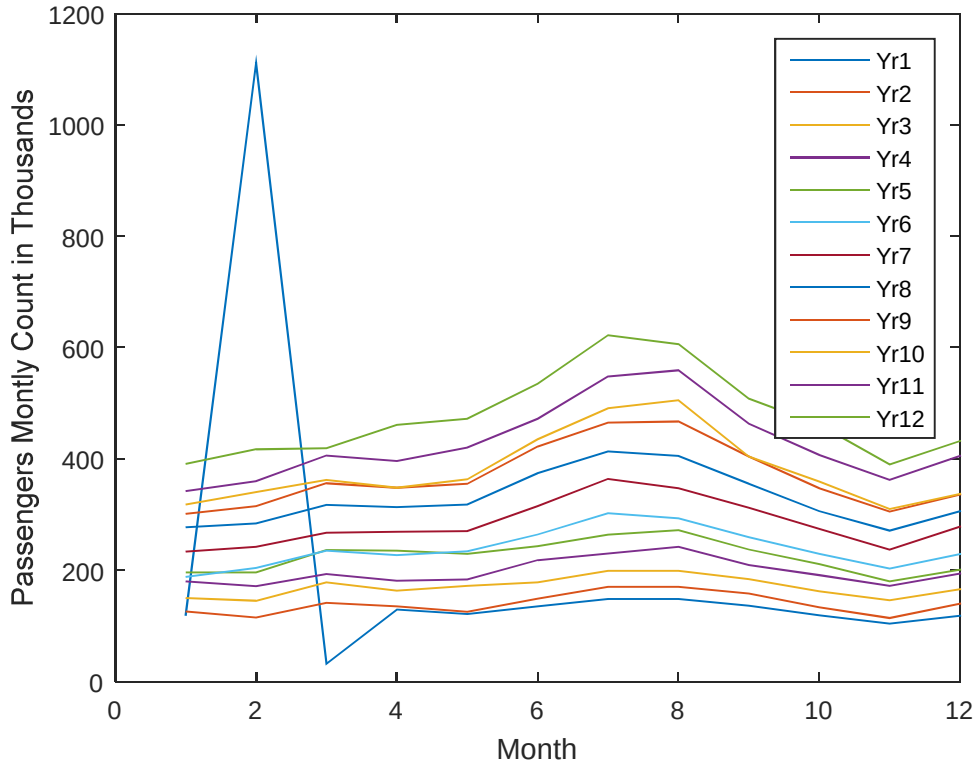
```
118,126,150,180,196,188,233,277,301,318,342,391
1112,115,145,171,196,204,242,284,315,340,360,417
32,141,178,193,236,235,267,317,356,362,406,419
129,135,163,181,235,227,269,313,348,348,396,461
121,125,172,183,229,234,270,318,355,363,420,472
135,149,178,218,243,264,315,374,422,435,472,535
148,170,199,230,264,302,364,413,465,491,548,622
148,170,199,242,272,293,347,405,467,505,559,606
136,158,184,209,237,259,312,355,404,404,463,508
119,133,162,191,211,229,274,306,347,359,407,461
104,114,146,172,180,203,237,271,305,310,362,390
118,140,166,194,201,229,278,306,336,337,405,432
```

In this data it is hard to figure out the state of business affairs regarding the passenger counts. You have heard something like "A picture is worth a thousand numbers". Well let us plot the data as shown below in MATLAB to get an initial view the intelligence underlying the business data.

Shown in MATLAB are also other forms of data analysis for business intelligence.

In Matlab:

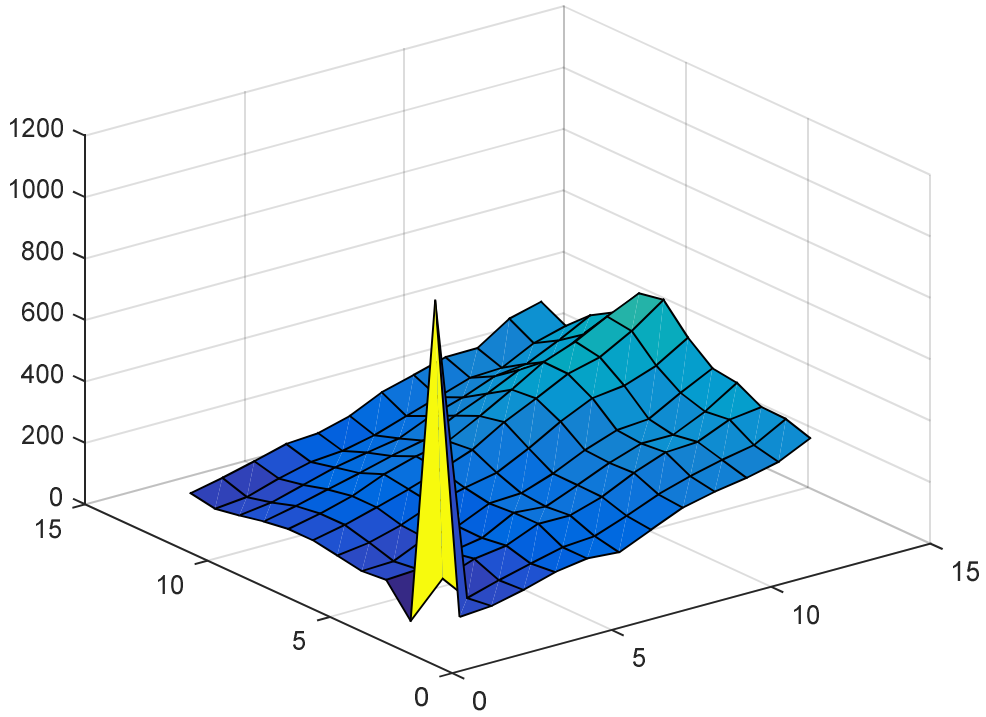
```
>> load tal.txt;
>> [n,p]=size(tal2)
n =12
p =12
>> t=1:12
t = 1    2    3    4    5    6    7    8    9    10    11    12
>> plot(t,tal2)
>> xlabel('year')
>> ylabel('Passengers in thousands')
>> ylabel('Passengers Montly Count in thousands')
>> xlabel('Month')
>>
legend('Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec')
>>% Select 'Edit' on plot shown, and use 'Copy Figure' and paste it your document
```



How about a 3-D picture? Just for fun.

In MATLAB

```
>> [X,Y]=meshgrid(1:1:12);
>>surf(X,Y,tal2)
```



Data Analysis Operations

In MATLAB

```
>> %sum of passenger count by year
>> sumy=sum(tal2)
sumy =2420 1676 2042 2364 2700 2867 3408 3939 4421 4572 5140
5714
>> mu=mean(tal2);
>> % standard deviation from the mean
>> sigma=std(tal2);
```

>>%When an *outlier* is considered to be more than three standard deviations away from the mean, you can use the >>following %syntax to determine the number of outliers in each column of the count matrix.

```
>> % Create a matrix of mean values by replicating the mu vector for n rows
>> MeanMat=repmat(mu,n,1);
```

>>% Create a matrix of standard deviation values by replicating the sigma vector for n rows

```
>> SigmaMat=repmat(sigma,n,1);
```

>>% Create a matrix of zeros and ones, where ones indicate the location of outliers

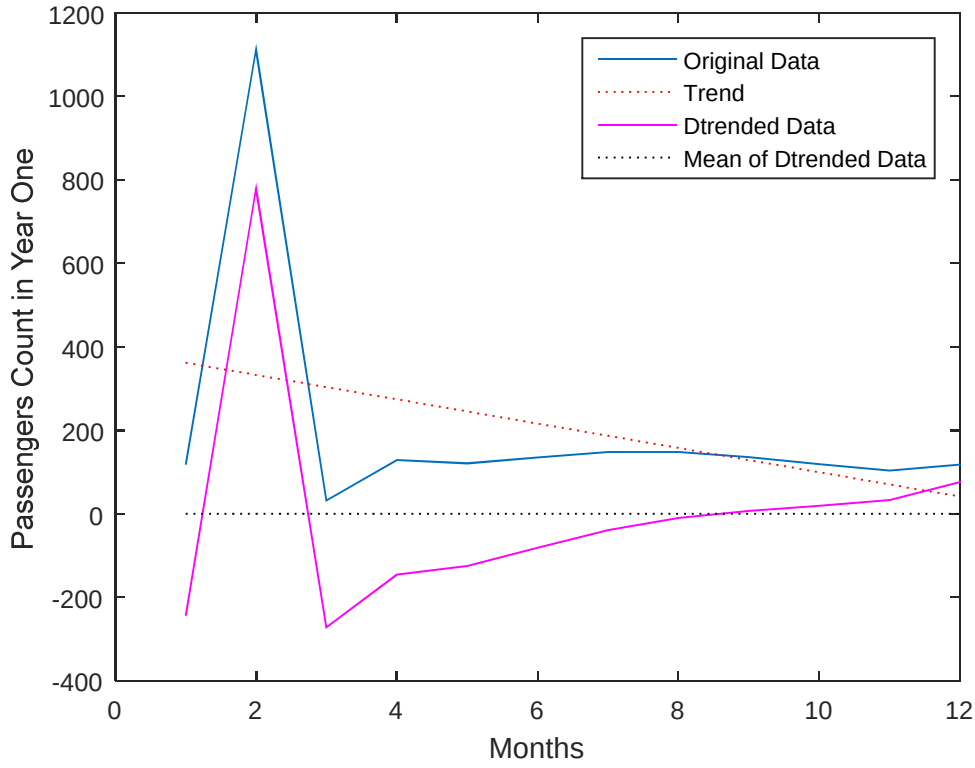
```
>> outliers=abs(tal2-MeanMat) >3*SigmaMat;
>> mout=sum(outliers)
```

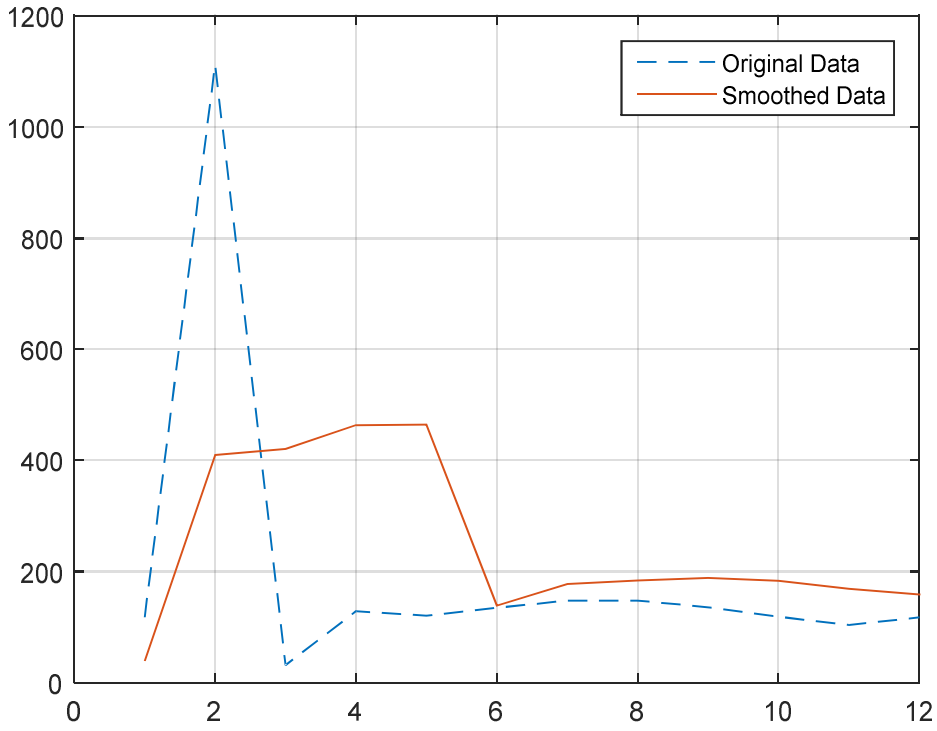
```
>>% Calculate the number of outliers in each column
>>nout = sum(outliers)
>>%MATLAB returns the following number of outliers in each column:
mout = 1     0     0     0     0     0     0     0     0     0     0     0
>> yr1=tal2(1:n)
yr1 =
  Columns 1 through 4
       118       1112         32        129
  Columns 5 through 8
       121        135        148        148
  Columns 9 through 12
       136        119        104        118
>> plot(t,yr1)
>> xlabel('Months')
>> ylabel('Passengers Count in Year One')
>> hold on
>> dtrendyr=detrend(yr1);
>> trend=yr1-dtrendyr;
>> mean(dtrendyr)
ans =   -6.3949e-14
>> plot(t,trend,':r')
>> plot(t,dtrendyr,'m')
>> plot(t,zeros(size(t)),':k')
>> legend('Original Data','Trend','Dtrended Data','Mean of Dtrended Data')
```

Data Smoothing Using Filters

In MATLAB

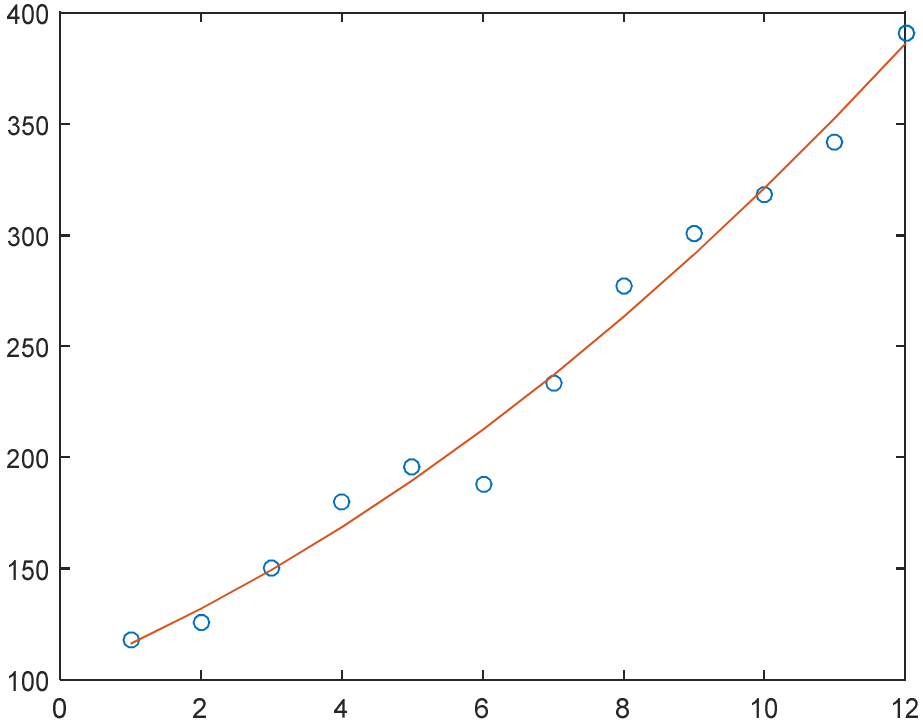
```
>> a=1;
% For a three months moving average
>> b=[1/3 1/3 1/3 1/3];
%moving average
>> ma=filter(b,a,yr1);
>> t=1:length(yr1);
>> plot(t,12,'--',t.ma.'-'),grid on
>> plot(t,yr1,'--',t,ma,'-'),grid on
>> legend('Original Data','Smoothed Data',2)
```


Original Data
Trend
Dtrended Data
Mean of Dtrended Data
Passengers Count in Year One
Months
1200
1000
800
600
400
200
0
-200
-400
0
2
4
6
8
10
12



Curve fitting to a vector of values
An example of a polynomial for curve fitting to a vector of data.

In MATLAB
% Month 1 of Time Series of passenger counts for airline data
>> m1=tal2(1,:);
% Plot the given data
>> plot(t,m1,'o')
% Polynomial of order 2 to be used is $y\ a_2t\ +a_1t+a_0$
% Find polynomial coefficient values by curve fitting
>> p=polyfit(t,m1,2)
p = 0.8826 13.0574 102.3182
% Calculate the curve using the coefficients
>> y=polyval(p,t);
% Draw it
>> plot(t,m1,'o',t,y)



Economic Applications
Example 6a: Business Production Economic Decisions
Consider a company called ABC Corporation producing a single product such as a television sets, car batteries or bars of soap.

Let
x be the units of the product sold in a unit of time such as a year.
pr(x) is the price per unit, it may charge less if it sells more.
c(x) is the total cost of producing x units
r(x) is the total revenue, r(x)=x*p(x)
Therefore, the total profit pr(x)=r(x)-c(x)=r(x)-x*pr(x)
m(x) is the marginal cost, the cost per unit if the production is increased slightly
m(x)=c'(x) [Note: the c'(x) stands for differential of c with respect to x]
Suppose that $c(x)=8300+3.25x+40x^{1/3}$
Find the cost per unit and the marginal cost, and evaluate them for x=1000
Average cost $av(x)=c(x)/x= (8300+3.25x+40^{1/3})/x$
Therefore,
$m(x)=c'(x)=3.25+(40/3)x^{-2/3}$

In MATLAB
>> syms x
>> c(x)=8300+3.25*x+40^(1/3);
>> m(x)=diff(c(x))
m(x) =

```
13/4
>> display(double(m(x)))
ans =
    3.2500
>> display(double(c('1000')/1000))
ans =
   11.5534
```

Example 6b:
Given the profit and cost functions as follows:
pr(x)=5.0-0.002x
c(x)=3.0+1.1x

Determine expressions for the marginal revenue, marginal cost, and marginal profit, and determine the production level that will produce the maximum total profit.

Revenue $r(x)=xpr(x)=5x-.002x^2$
profit $pro(x)=r(x)-c(x)=-3+3.9x-.002x^2$
Therefore
marginal revenue, mr(x)=r'(x)=5-.004x
marginal cost, mc(x)=c'(x)=1.1
marginal profit, mpro(x)=pro'(x)=r'(x)-c'(x)=3.9-004x
For maximizing profit, we set pro'(x)=0 i.e. 3.9-.004x=0, therefore, x=975

In MATLAB

```
> pr(x)=5.0-.002*x;
>> c(x)=3.0+1.1*x;
>> r(x)=x*pr(x);
>> pro(x)=r(x)-c(x);
 >> mr(x)=diff(r(x));
 >> mc(x)=diff(c(x));
 >> display(double(mc(x)))
ans =
    1.1000
>> mpro(x)=diff(pro(x))
mpro(x) =
39/10 - x/250
>> x=(39/10)*250
x =
   975
```

Example 7
The operating cost for a certain truck is estimated to be $ .3+v/200 per milebnwhen drivenat a speed of v miles per hour. The driver is paid $14 per hour. What speed will minimize the cost

of a delivery to a city k miles (any distance such as 100) away? Assume that the law restricts the speed 40<=v<=60.

The cost,
C=driver cost+operating cost
=14*100/v + 100(.3 + v/200)
diffC(v)=14kv^-2 + k/200, making it equal to zero
14k/v^2=k/200
v^2=2800
v=53 mph

In MATLAB
>> syms v
>> c(v)=1400/v+100*(.3+v/200);
>> diffc(v)=diff(c(v))
diffc(v) =
1/2 - 1400/v^2
>> OptSpeed=2800^.5
OptSpeed =
52.9150

References:

https://www.microstrategy.com/us/resources/introductory-guides/business-analytics-everything-you-need-to-know
http://www.mathworks.com/help/matlab/data_analysis/descriptive-statistics.html?refresh=true
https://en.wikipedia.org/wiki/Business_analytics
http://ptgmedia.pearsoncmg.com/images/0131413031/samplechapter/0131413031_ch03.pdf
Fernandez, Oscar, Everyday Calculus, Discovering the Hidden Math All Around Us, Princeton University Press, 2014.
Purcell, E.J. and Varberg, D., Calculus with Analytic Geometry, 5e, Prentice-Hall, 1987.