APPENDIX A

Chapter 3- Development of a New Model using Dynamic Medium

```
clear all
clc
filename=input('Enter the Product Name: ','s');
Horizon=input('Enter the Forecast Horizon: ');
Y=xlsread(filename);
t=1:length(Y);
[mse1,e1,alpha1]=ses1(Y);
[mse2,e2,alpha2,beta2]=holt1(Y);
[mse3,e3,coeff1]=linearregression(Y);
[mse4,coeff2]=nonlinearregression(Y);
[mse5,e5,p,d,q]=ARIMA(Y);
err=[mse1,mse2,mse3,mse4,mse5];
er=min(err);
disp('SES MSE :');
disp(mse1);
disp('Holt MSE :');
disp(mse2);
disp('Linear Regression MSE :');
disp(mse3);
disp('Polynomial Regression MSE :');
disp(mse4);
disp('Auto Regressive Integrated Moving Average(ARIMA) MSE :');
disp(mse5);
l=length(Y);
```

```
t1=1:(l+Horizon);
t2=(l+1):(l+Horizon);
if er==mse1
            disp('SES Model has the lowest error');
           disp('The forecast is :');
            F1=ses(alpha1,Y,Horizon);
            for i=(l+1):(l+Horizon)
            disp(F1(i));
           end
            %for i=1:Horizon
           %f(i)=F1(l+i);
            %end
            %f=f';
           %disp('The Prediction Intervals for the forecast(s) :');
            %Predint=predictioninterval(mse1,f,e1);
            %disp(Predint);
            plot(t,Y,'g+',t1',F1,'b-'), grid on;
            legend('Sales Data','Forecast','Location','NorthWest');
           xlabel('Time period (Weeks)');
           ylabel('Sales');
elseif er==mse2
            disp('Holt Model has the lowest error');
           disp('The forecast is :');
            smt=[alpha2,beta2];
            F2=holt(smt,Y,Horizon);
           for i=(l+1):(l+Horizon)
            disp(F2(i));
```

```
        end
        %for i=1:Horizon
        %f(i)=F2(l+i);
       %end
        %f=f';
       %disp('The Prediction Intervals for the forecast(s) :');
%Predint=predictioninterval(mse2,f,e2);
%disp(Predint);
plot(t,Y,'g+',t1',F2,'b-'), grid on;
legend('Sales Data','Forecast','Location','NorthWest');
xlabel('Time period (Weeks)');
ylabel('Sales');
elseif er==mse3
disp('Linear Regression Model has the lowest error');
disp('The forecast is :');
F3=polyval(coeff1,t1');
for i=(l+1):(l+Horizon)
disp(F3(i));
end
%for i=1:Horizon
%f(i)=F3(l+i);
%end
%f=f';
%disp('The Prediction Intervals for the forecast(s) :');
%Predint=predictioninterval(mse3,f,e3);
%disp(Predint);
plot(t,Y,'g+',t1',F3,'b-'), grid on;
```

```
legend('Sales Data','Forecast','Location','NorthWest');
xlabel('Time period (Weeks)');
ylabel('Sales');
elseif er==mse4
disp('Polynomial Regression Model has the lowest  error');
disp('The forecast is :');
F4=polyval(coeff2,t1');
for i=(l+1):(l+Horizon)
disp(F4(i));
end
%for j=1:l
%e4(j)=Y(j)-F4(j);
%end
%for i=1:Horizon
%f(i)=F4(l+i);
%end
%f=f';
%disp('The Prediction Intervals for the forecast(s) :');
%Predint=predictioninterval(mse4,f,e4);
%disp(Predint);
plot(t,Y,'g+',t1',F4,'b-'), grid on;
legend('Sales Data','Forecast','Location','NorthWest');
xlabel('Time period (Weeks)');
ylabel('Sales');
else
fprintf('ARIMA(%d,%d,%d) model has the lowest error,',p,d,q);
disp(' the forecast is :');
```

```
F5=ARIMAPREDICTION(Y,p,d,q,Horizon);
for i=(l+1):(l+Horizon)
disp(F5(i));
end
%for i=1:Horizon
%f(i)=F5(l+i);
%end
%f=f';
%disp('The Prediction Intervals for the forecast(s) :');
%Predint=predictioninterval(mse5,f,e5);
%disp(Predint);
plot(t,Y,'g+',t1',F5,'b-'), grid on;
legend('Sales Data','Forecast','Location','NorthWest');
xlabel('Time period (Weeks)');
ylabel('Sales');
end

function [e6,e3,alpha]=ses1(Y)
l=length(Y);
e1=nan(101,1);
e2=nan(101,1);
e3=nan(l-1,1);
e4=nan(l-1,1);
for i=1:1001
 g=i-1;
 h=g/1000;
 F=ses(h,Y);
```

```
 e=Y-F;
 e1(i)=0;
 for j=1:l
 e1(i)=e1(i)+(e(j)*e(j));
 end
 e2(i)=e1(i)/l;
end
m=min(e2);
for k=1:1001
 if e2(k)==m
 alpha=(k-1)/1000;
 end
end
F1=ses(alpha,Y);
e5=0;
for n=21:l
 e3(n)=Y(n)-F1(n);
 e4(n)=e3(n)*e3(n);
 e5=e5+e4(n);
end
e6=e5/(l-20);
end
function f=ses(a,y,h)
% Single exponential smoothing
%
% Syntax: f=ses(a,y,h)
% Inputs: % a = smoothing parameter alpha or [a level], where level is the % initial level
```

```
% y = time series
% h = horizon (default = 0)
% Outputs:
% f = forecasts
if nargin<3
 h=0;
end
if length(a)>1
init=a(2);
 a=a(1);
else
 init=y(1);
end
l=length(y);
f=nan(l+h,1);
f(1)=init;
for i=2:l
 f(i)=a*y(i-1)+(1-a)*f(i-1);
end

if h>0 % create forecasts
 f(l+1:l+h)=a*y(l)+(1-a)*f(l);
end
function [e6,e3,alpha,beta]=holt1(Y)
l=length(Y);
e1=nan(101,1);
e2=nan(101,1);
```

```
e3=nan(l-1,1);
e4=nan(l-1,1);
for i=1:1001
 g=i-1;
 h(1)=g/1000;
 for j=1:1001
s=j-1;
 h(2)=s/1000;
 F=holt(h,Y);
 e=Y-F;
 e1(i,j)=0;
 for c=1:l
 e1(i,j)=e1(i,j)+(e(c)*e(c));
 end
 e2(i,j)=e1(i,j)/l;
 end
end
m1=min(e2);
m=min(m1);
for k=1:1001
 for j=1:1001
 if e2(k,j)==m
 a(1)=(k-1)/1000;
 a(2)=(j-1)/1000;
 end
 end
end
```

```
alpha=a(1);
beta=a(2);
F1=holt(a,Y);
e5=0;
for n=21:l
 e3(n)=Y(n)-F1(n);
 e4(n)=e3(n)*e3(n);
 e5=e5+e4(n);
end
e6=e5/(l-20);
end

function f=holt(a,y,h)
% Holt's linear trend exponential smoothing
%
% Syntax: f=holt(a,y,h)
% Inputs:
% a = vector of smoothing parameters a(1)=a, a(2)=b,
a(3)=initial
% level (optional), a(4) = initial trend (optional)
% y = time series
% h = horizon (default = 0)
% Outputs:
% f = forecasts
%
% Argument check
if length(a)<2
```

```
 error('Not enough smoothing parameters specified')
end
if nargin<3
 h=0;
end
% Initialisation
l=length(y);
f=nan(l+h,1);
level=nan(l,1);
trend=nan(l,1);
if length(a)>2
 level(1)=a(3);
 trend(1)=a(4);
else
 trend(1)=y(2)-y(1);
 level(1)=y(1);
end
b=a(2);
a=a(1);
f(1)=level(1)+trend(1);
for i=2:l
 level(i)=a*y(i)+(1-a)*(level(i-1)+trend(i-1));
 trend(i)=b*(level(i)-level(i-1))+(1-b)*trend(i-1);
 f(i)=level(i-1)+1*trend(i-1); % calculate forecast of the previous period
end
if h>0
 for i=l+1:l+h
```

```
 f(i)=level(l)+(i-l)*trend(l);
 end
end
function [e3,e1,p]=linearregression(Y)
e1=nan(101,1);
l=length(Y);
t=1:l;
p=polyfit(t',Y,1);
f=polyval(p,t');
e2=0;

for i=21:l
 e1(i)=Y(i)-f(i);
 e2=e2+(e1(i)*e1(i));
end
e3=e2/(l-20);
end

function [m,p]=nonlinearregression(Y)
e1=nan(101,1);
e3=nan(101,1);
l=length(Y);
t=1:l;
for i=2:3
 p1=polyfit(t',Y,i);
 f1=polyval(p1,t');
 e2=0;
```

```
 for j=21:l
 e1(j)=Y(j)-f1(j);
 e2=e2+(e1(j)*e1(j));
 end
 e3(i-1)=e2/(l-20);
end
m=min(e3);
for k=1:2
 if e3(k)==m
 n=k+1;
 p=polyfit(t',Y,n);
 end
end
end

function [e4,e1,pbest,I,qbest] = ARIMA(Input)
MaxI = 2; %Maximum integrand period%
%Checking for I(n) process%
VarianceMatrix = zeros(MaxI,1);
for I = 1: MaxI
 VarianceMatrix(I,1) = var(diff(Input,I));
end
IMatrix = find(ismember(VarianceMatrix, min(VarianceMatrix))==1);
if var(Input) < min(VarianceMatrix)
 I = 0;
else
 I = IMatrix;
```

```
end
ALPHA=.01; % Pr[type 1 error] for threshold line to test acf for N(0,1/sqrt(n))
pvec=[0,1,2]; % orders of AR
%pvec=[4,5,6];
d=I; % number of differences
qvec=[0,1,2]; % orders of MA
%qvec=[2,3];
RESIDPLOTS=1; % > 0 to plot residual acfs
%descriptor='FMCG Sales'; % strings to be used for plotting
%xvalues='time period (Weeks)';
%dindex=[1:120]; % select which of the input data to use
sales=Input;
time=1:length(Input);
dataname='sales'; %%%%%%%%%%%% dataname statement
%%%%%%%%%%%%%%%%%%%%
descriptor='SALES';
xvalues='time period (Weeks)';
dindex=1:length(sales);
eval(['data=',dataname,';']); % puts the input into data
ddata=data(dindex); % difference it d times
for j=1:d
 ddata=diff(ddata);
end
% ddata has been 1st differenced d times
nd=length(ddata);
%%%%%%%%%% now plot ddiff and acf %figure;
%subplot(211);
```

```
%plot(1:nd,ddata,'-');
%title(['Result of ',int2str(d),' 1st differences of data: ',descriptor]);
%subplot(212); % now the acf
rhod=xcov(ddata,'coeff'); % this returns the acf normalized by the variance
% it is even, so ignore the negative lags and take only half of the positive ones
rhod=rhod(nd:nd+floor(nd/2)); % floor needed in case nd/2 not
integer nrho=length(rhod);
%stem(rhod,'.'); % a stem plot with dot marker
%hold on;
%plot(zeros(1,nrho));
thr=norminv(1-ALPHA/2,0,1/sqrt(nd));
%plot(1:nrho,thr*ones(1,nrho),'--r',1:nrho,-thr*ones(1,nrho),'--r');
%title(['autocorrelation, n=',int2str(nd),' alpha = ',num2str(ALPHA)]);

 % now estimate ARMA model; ARMAX is AX=BU + Ce so identify phi with A and theta with C
np=length(pvec);
nq=length(qvec);
aicsave=-99*ones(np,nq);
fpesave=-99*ones(np,nq);
minaic=1e+6;
for pp=1:np
 p=pvec(pp);
 for qq=1:nq
 q=qvec(qq);
 if p+q ~=0
 orders=[p q];
 m=armax(ddata,orders); % m is a structure with the model stuff in it
```

```
resids=pe(m,ddata); % pe returns the prediction errors
nres=length(resids);
rhores=xcov(resids,'coeff'); % this returns the acf normalized by the variance
% it is even, so ignore the negative lags and take only half of the positive ones
rhores=rhores(nres:nres+floor(nres/2)); % floor needed in case nd/2 not integer
nrho=length(rhores); % now rhores(1) is zero lag
% next compute the Ljung - Box statistic and P-values
deltak=floor(nrho/10)+1;
kvec=(p+q+1):deltak:(p+q+1+4*deltak);
for kk=1:5
Q=0;
for j=2:kvec(kk)+1
Q=Q+(rhores(j).^2)/(nd-j);
end

Q=nd*(nd-1)*Q;
ljpv(kk)=1-chi2cdf(Q,kvec(kk)-p-q); % df=kvec(kk)-p-q
end

aicsave(pp,qq)=aic(m);
fpesave(pp,qq)=fpe(m);
if aicsave(pp,qq) < minaic
minaic=aicsave(pp,qq); % save the min
pbest=p;
qbest=q;
mbest=m;
end
```

```
 %fprintf('(p,d,q)=(%d,%d,%d) aic=%d
fpe=%d',p,d,q,aicsave(pp,qq),fpesave(pp,qq));
 QQ=[kvec;ljpv];
 %fprintf('Ljung-Box P-values : ');
 %fprintf(' K=%d P-v=%6.3e \n',QQ(:));
 if RESIDPLOTS
 %%%%%%%%% resids and acf  %figure;
 %subplot(211);
 %plot(1:nd,resids,'-');
 %title(['Residuals from ARIMA(',sprintf('%d,%d,%d)',p,d,q),' for data: ',descriptor]);
 %subplot(212); % now the acf
 %stem(rhores,'.'); % a stem plot with dot marker
 %hold on;
 %plot(zeros(1,nrho));
 thr=norminv(1-ALPHA/2,0,1/sqrt(nres));

 %plot(1:nrho,thr*ones(1,nrho),'--r',1:nrho,-
thr*ones(1,nrho),'--r');
 %title(['autocorrelation of residuals,
n=',int2str(nres),' alpha = ',num2str(ALPHA)]);
 end % RESIDPLOTS
 end
 end
end
%disp(['SUMMARY OF ARIMA MODELS FOR DATA : ',descriptor]);
%fprintf('AIC min at (p,d,q)=(%d,%d,%d)
aic=%d',pbest,d,qbest,aic(mbest));
```

```
[A,B,C,D,F]=polydata(mbest);
%disp([' phi coefficients : ',num2str(A)]);
%disp(['theta coefficients : ',num2str(C)]);
Fct = ARIMAPREDICTION(Input,pbest,I,qbest);
l=length(Input);
e3=0;
for n=21:l
 e1(n)=Input(n)-Fct(n);
 e2(n)=e1(n)*e1(n);
 e3=e3+e2(n);
end
e4=e3/(l-20);
end

function [pred] = ARIMAPREDICTION(y,p,d,q,predictions)
if nargin<5
 predictions=0;
end
yy = y(1);
yy1=y;
for m=1:d
 y = diff(y);
end
l=length(y);
[Parameter,~,Errors]=armaxfilter(y,1,p,q);
f=y-Errors;
%tsresidualplot(y,Errors);
```

```
if predictions==0 && d==0
 pred=f;
elseif predictions==0 && d>0
 pred(1)=yy;
 for k=1:l
 pred(k+1)=yy1(k)+f(k);
 end
 pred=pred';
 elseif predictions>0 && d>0
 R=l-predictions;
 [f1] = arma_forecaster(y,Parameter,1,p,q,R,predictions);
 j=1;
 for i=(l-predictions)+1:l
 f2(j) = f1(i);
 j=j+1;
 end
 pred(1)=yy;
 for k=1:l
 pred(k+1)=yy1(k)+f(k);
 end
 pred(l+2)=yy1(l)+f2(1);
 for h=l+3:l+predictions+1
 pred(h)=pred(l+1)+f2(h-(l+1));
 end
 pred=pred';
 else
 R=l-predictions;
```

```
 [f1] = arma_forecaster(y,Parameter,1,p,q,R,predictions);
 j=1;
 for i=(l-predictions)+1:l
 f2(j) = f1(i);
 j=j+1;
 end
 f2=f2';
 pred=cat(1,f,f2);
end
end
```

APPENDIX B

Chapter 4- Development of a New Model for Slow Moving Products

Comparison of EPDM with other models

```
alpha=zeros(0,1000);
SES=zeros(0,1000);
crd=zeros(0,1000);
crt=zeros(0,1000);
crt1=zeros(0,1000);
cr1=zeros(0,1000);
cr2=zeros(0,1000);
p=zeros(0,1000);
cr=zeros(0,1000);
SBA=zeros(0,1000);
pr=zeros(0,1000);
TSB=zeros(0,1000);
EPDM=zeros(0,1000);
m=input('enter the total number of smoothing constants you are going to use\n');
for i=1:m
   alpha(i)=input('enter the values of smoothing constant\n');
end
D=load('C:\Users\dell\Desktop\demand.txt','-ascii');
n=size(D,1);
fprintf('the forecast for next period  F %d \n',n+1);
for i=1:m
   fprintf('using smoothing constant %f is \n',alpha(i))
```

```
    SES(i,2)=D(1);
    for t=3:n+1
        SES(i,t)=(SES(i,t-1)+(alpha(i)*(D(t-1)-SES(i,t-1))));
    end
    fprintf('By SES %f\n',SES(i,t));
x=0;
y=0;
z=0;
for t=1:n
    if D(t) == 0
       p(i,t)=0;
       y=y+1;
    else
       p(i,t)=1;
       x=x+1;
       crd(i,x)=D(t);
       crt1(i,x)=t;
       z=z+1;
    end
end
cr1(i,2)=crd(i,1);
for t=3:x+1
    cr1(i,t)=(cr1(i,t-1)+(alpha(i)*(crd(i,t-1)-cr1(i,t-1))));
end
for t=2:x
```

```
    crt(i,t)= (crt1(i,t)-crt1(i,t-1));
end
cr2(i,2)=crt1(i,1);
for t=3:x+1
    cr2(i,t)=(cr2(i,t-1)+(alpha(i)*(crt(i,t-1)-cr2(i,t-1))));
end
cr(i)=(cr1(i,t)/cr2(i,t));
fprintf('By CRO %f\n',cr(i));
SBA(i)=0.975*cr(i);
fprintf('By SBA %f\n',SBA(i));
pr(i,2)=p(i,1);
    for t=3:n+1
        pr(i,t)=(pr(i,t-1)+(alpha(i)*(p(i,t-1)-pr(i,t-1))));
    end
    TSB(i,t)=pr(i,t)*cr1(i,(x+1));
    fprintf('By TSB %f\n',TSB(i,t));
    EPDM(i)=(z/(z+y))*cr1(i,(x+1));
    fprintf('By EPDM %f\n',EPDM(i));
end
```

To calculate Demand Forecast through DSBM

```
D=zeros(0,1000);
p=zeros(0,1000);
crd=zeros(0,1000);
crt=zeros(0,1000);
crt1=zeros(0,1000);
x=0;
y=0;
alpha=0.1;
n=input('enter the number of periods for which actual data is present\n');
for t=1:n
   fprintf('enter the actual sales of period %d\n',t);
   D(t)=input('');
end
for t=1:n
   if D(t) == 0
      p(t)=0;
   else
      p(t)=1;
      x=x+1;
      crd(x)=D(t);
      crt1(x)=t;
   end
end
cr1(2)=crd(1);
```

```
for t=3:x+1
   cr1(t)=(cr1(t-1)+(alpha*(crd(t-1)-cr1(t-1))));
end
for t=2:x
   crt(t)= (crt1(t)-crt1(t-1));
end
cr2(2)=crt1(1);
for t=3:x+1
   cr2(t)=(cr2(t-1)+(alpha*(crt(t-1)-cr2(t-1))));
end
cr=(cr1(x+1)/cr2(x+1));
fprintf('%f',cr);
```

APPENDIX C

Chapter 5- Demand Forecasting Model for Seasonal Products

```
clear all
clc
echo off
filename1=input('enter the product name:   ', 's');
filename2=input('enter the y data:   ', 's');
y=xlsread(filename2);
x=xlsread(filename1);
q=length(x);
h=input('enter the forecast horizon:   ');
z=1:1:q;
[mse1, f1] = wma(x);
m(1)=mse1;
disp('the mse error due to weighted average method is:  ');
disp(mse1);
[mse2, f2] = linreg(x);
m(2)=mse2;
disp('the mse error due to linear regression is:   ');
disp(mse2);
[mse3, f3] = reg2(x);
m(3)=mse3;
disp('the mse error due to Polynomial regression of order 2 is:   ');
disp(mse3);
[mse4, f4] = reg3(x);
m(4)=mse4;
disp('the mse error due to Polynomial regression of order 3 is:   ');
disp(mse4);
disp('for Single Exponential Smoothing');
[mse5, f5] = ses(x);
m(5)=mse5;
 disp('the mse error due to single exponential smoothing is:   ');
disp(mse5);
[mse6, f6] = holt(x);
m(6)=mse6;
disp('the mse error due to holts method is:   ');
disp(mse6);
disp('for winters method');
[mse7, f7] = winter(x);
m(7)=mse7;
disp('the mse error due to winter method is:  ');
disp(mse7);
n=min(m);
disp('the minimum MSE error is:   ');
disp(n);
```

```
if n==mse1
   disp('when using wma function');
   disp('                ');
   disp('the forecast for the period'); disp(h); disp('using wma is:   ');
   for i=1:1:h
      a(i)=f1(i);
      disp(f1(i))
   end
   plot(z,f1,'-');
   title('Plot between Forecast Values and Time Period');
   xlabel('Time Period');
   ylabel('Forecast Values');
elseif n==mse2
   disp('when using linear regression function');
   disp('                ');
   disp('the forecast for the period'); disp(h); disp('using linear regression is:   ');
   for i=1:1:h
      a(i)=f2(i);
      disp(f2(i))
   end
   plot(z,f2,'-');
   title('Plot between Forecast Values and Time Period');
   xlabel('Time Period');
   ylabel('Forecast Values');
elseif n==mse3
   disp('when using non-lin reg of order 2 function');
   disp('                ');
   disp('the forecast for the period'); disp(h); disp('using non-lin reg of order 2 is:   ');
  for i=1:1:h
     a(i)=f3(i);
     disp(f3(i))
  end
   plot(z,f3,'-');
   title('Plot between Forecast Values and Time Period');
   xlabel('Time Period');
   ylabel('Forecast Values');
elseif n==mse4
   disp('when using non-lin reg of order 3 function');
   disp('                ');
   disp('the forecast for the period'); disp(h); disp('using non-lin reg of order 3 is:   ');
   for i=1:1:h
     a(i)=f4(i);
    disp(f4(i))
   end
 plot(z,f4,'-');
   title('Plot between Forecast Values and Time Period');
```

```
   xlabel('Time Period');
   ylabel('Forecast Values');
elseif n==mse5
   disp('when using single exponential smoothing function');
   disp('                ');
   disp('the forecast for the period'); disp(h); disp('using ses is:   ');
  for i=1:1:h
       a(i)=f5(i);
      disp(f5(i))
  end
  plot(z,f5,'-');
   title('Plot between Forecast Values and Time Period');
   xlabel('Time Period');
   ylabel('Forecast Values');
elseif n==mse6
   disp('when using holts method');
   disp('                ');
   disp('the forecast for the period'); disp(h); disp('using holts is:   ');
  for i=1:1:h
     a(i)=f6(i);
    disp(f6(i))
  end
plot(z,f6,'-');
   title('Plot between Forecast Values and Time Period');
   xlabel('Time Period');
   ylabel('Forecast Values');
else
   disp('when using winters function');
   disp('                ');
   disp('the forecast for the period'); disp(h); disp('using winters method is:   ');
   for i=1:1:h
      a(i)=f7(i);
      disp(f7(i));
   end
   plot(z,f7,'-');
   title('Plot between Forecast Values and Time Period');
   xlabel('Time Period');
   ylabel('Forecast Values');
end
```

FUNCTION wma:

```
% forecast using weighted moving average
% weight value taken is 0.2, 0.3 and 0.5
function [mse1, f1] = wma(x)
l=length(x);
```

```
for i=4:1:l
   a(i)=((0.2)*(x(i-3)))+(0.3*(x(i-2)))+(0.5*(x(i-1)));
 %  disp(a(i));
 f1(i-3)=a(i);
   g(i)=((a(i)-x(i))^2);
end
e=sum(g)/(l-3);
mse1=e;
```

Function linreg:

```
%forecast using 1st order of linear regression
function [mse2, f2] = linreg(x)
l=length(x);
% finding out which order of equation...
... would fit
p=polyfit(y,x,1);
for i=1:1:l
a(i)=(p(1)*(i))+p(2);
f2(i)=a(i);
g(i)=((a(i)-x(i))^2);
end
e=(sum(g))/l;
mse2=e;
```

FUNCTION reg2:

```
% forecast using 2nd order of Polynomial regression
function [mse3, f3] = reg2(x)
l=length(x);
% finding out which order of equation...
... would fit
p=polyfit(y,x,2);
for i=1:1:l
a(i)=(p(1)*(i^2))+(p(2)*i)+p(3);
f3(i)=a(i);
g(i)=((a(i)-x(i))^2);
end
e=(sum(g))/l;
mse3=e;
```

FUNCTION reg3:

```
% Polynomial regression of order 3
function [mse4, f4] = reg3(x)
l=length(x);
% finding out which order of equation..... would fit
p=polyfit(y,x,3);
```

```
for i=1:1:l
   b(i)=(p(1)*((i)^3))+(p(2)*((i)^2))+(p(3)*(i))+p(4);
   f4(i)=b(i);
   g(i)=((b(i)-x(i))^2);

end
e=sum(g)/l;
mse4=e;

FUNCTION ses:
% forecast using single exponential smoothing
function [mse5, f5] = ses(x)
y=xlsread('yvalue.xls');
l=length(x);
a=input('enter the value of alpha:   ');
q(1)=sum(x)/l;
f5(1)=q(1);
for j=2:1:l+1
   q(j)=(a*(x(j-1)))+((1-a)*(q(j-1)));
      f(j)=q(j);

      f5(j)=f(j);
      g(j-1)=((f5(j-1)-x(j-1))^2)/l;
end
 m=sum(g);
mse5=m;

FUNCTION holts:
% forecast using holt's method
function [mse6, f6] = holt(x)
w=length(x);
l(1)=input('enter the initial level:   ');
t(1)=input('enter the initial trend:   ');
a=input('enter the alpha value:   ');
b=input('enter the beta value:   ');
f6(1)=t(1)+l(1);
g(1) = ((f6(1)-x(1))^2)/w;
for j=2:1:w
      l(j)=(a*(x(j-1)))+((1-a)*(l(j-1)+t(j-1)));
      t(j)=(b*(l(j)-l(j-1)))+((1-b)*(t(j-1)));
      f6(j)=l(j)+t(j);

     g(j)=((f6(j)-x(j))^2)/w;
   end
   q=sum(g);
   e=q+g(1);
```

```
    mse6=e;
```

FUNCTION winter:

```
%forecast using winter's method
function [mse7, f7] = winter(x)
l=length(x);
r(1)=input('enter initial level:   ');
t(1)=input('enter initial trend:   ');
p=input('enter the periodicity:   ');
for i=1:1:p
   disp('enter the ');
   disp(i);
   s(i)=input('seasonality factor:   ');
end
f7(1)=(r(1)+t(1))*s(1);
a=input('enter the value of alpha:   ');
b=0.2;
y=0.1;
for j=1:1:l
   r(j+1)=(a*(x(j)/s(j)))+((1-a)*(r(j)+t(j)));
      t(j+1)=(b*(r(j+1)-r(j)))+((1-b)*(t(j)));
      s(j+p)=(y*(x(j)/r(j+1)))+((1-y)*s(j));
      f7(j+1)=(r(j+1)+t(j+1))*s(j+1);
     g(j)=((f7(j)-x(j))^2)/l;
end
u=sum(g);
mse7=u;
```

FUNCTION sesalpha (to find the optimum value of alpha for single exponential smoothing):

```
%finding the value of alpha for ses
filename1=input('enter the filename:   ', 's');
filename2=input('enter the y data filename:   ', 's');
x=xlsread(filename1);
y=xlsread(filename2);
w=length(x);
l(1)=sum(x)/w;
f(1)=l(1);
for i=1:1:9
   a=i/10;
   disp('the value of a is:   ');
   disp(a);
   for j=2:1:w+1
      l(j)=(a*(x(j-1)))+((1-a)*(l(j-1)));
      f(j)=l(j);
```

```
        %disp(f(j));
        g(j-1)=((f(j-1)-x(j-1))^2)/w;
    end
    disp('the error involved is:   ');
    m(i)=sum(g);
    disp(m(i));
end
n=min(m);
disp('the minimum error is:   ');
disp(n);
if n==m(1)
    disp('when value of alpha is 0.1');
elseif n==m(2)
    disp('when value of alpha is 0.2');
    elseif n==m(3)
    disp('when value of alpha is 0.3');
    elseif n==m(4)
    disp('when value of alpha is 0.4');
    elseif n==m(5)
    disp('when value of alpha is 0.5');
    elseif n==m(6)
    disp('when value of alpha is 0.6');
    elseif n==m(7)
    disp('when value of alpha is 0.7');
    elseif n==m(8)
    disp('when value of alpha is 0.8');
else
    disp('when value of alpha is 0.9');
end
```

FUNCTION holta (to find the optimum value of alpha for holt's function):

```
%alpha finder for holts
clear all
clc
echo off
filename1=input('Enter the Product Name: ','s');
x=xlsread(filename1);
w=length(x);
filename2=input('Enter the y data: ','s');
y=xlsread(filename2);
z(1)=1000000000000000000000000000;
p=polyfit(y,x,1);
l=length(x);
for i=1:1:9
    t(i,1)=p(1);
```

```
l(i,1)=p(2);
end
g(1)=t(1,1)+l(1,1);
for i=2:1:10
   a=(i-1)/10;
   disp('the value of a is');
   disp(a);
   for k=2:1:10
      b=(k-1)/10;
    %  disp('the value of b is');
    %disp(b);
      for j=2:1:w;
      l(i,j)=(a*(x(j-1))) + ((1-a)*(l(i-1,j-1)+t(i-1,j-1)));
      t(i,j)=(b*(l(i,j)-l(i-1,j-1)))+((1-b)*t(i-1,j-1));
      f(i,j)=l(i,j)+t(i,j);
       l(i-1,j)=l(i,j);
       t(i-1,j)=t(i,j);
       g(j)=f(i,j);
       %disp(g(j));
       v(j)=((g(j-1)-x(j))^2)/w;
       c(j)=sum(v);
      end
      z(k)=c(j);
      %disp(z(k));
   end
   disp('the min value of mse when alpha is constant');
   disp('and beta is varied from 0.1 to 0.9 is');
         o(i-1)=min(z);
      disp(o(i-1));

end
disp('the minimum value of mse is');
r=min(o);
disp(r);
```

FUNCTION holtb (to find the optimum value of beta for holt's function):

```
%beta finder for holts
clear all
clc
echo off
filename1=input('Enter the Product Name: ','s');
x=xlsread(filename);
w=length(x);
filename2=input('Enter the y data: ','s');
y=xlsread(filename2);
```

```
p=polyfit(y,x,1);
disp(p);
t(1)=p(1);
l(1)=p(2);
f(1)=t(1)+l(1);
g(1)=((f(1)-x(1))^2)/12;
disp(g(1));
a=input('enter the value of alpha    ');
for i=1:1:9
   b=i/10;
   disp('the value of beta is');
   disp(b);
   for j=2:1:w
      l(j)=(a*(x(j-1)))+((1-a)*(l(j-1)+t(j-1)));
      t(j)=(b*(l(j)-l(j-1)))+((1-b)*(t(j-1)));
      f(j)=l(j)+t(j);
     % disp(f(j));
     g(j)=((f(j)-x(j))^2)/w;
   end
   e(i)=sum(g);
   disp(e(i));
end
m=min(e);
disp('The minimum value of MSE is');
disp(m);
disp('The initial level is:   ');
disp(l(1));
disp('the initial trend is:   ');
disp(t(1));
for i=1:1:6
   disp(f(i));
end
```

FUNCTION winterinit

% to find the initial value of trend and level; periodicity and associated seasonal factors

```
function [t,l,s,p]=winterinit(x)
filename2=input('enter the y file:   ','s');
y=xlsread(filename2);
w=length(x);
%disp(w);
plot(y,x);
title('time series')
xlabel('time period')
ylabel('forecast value')
```

```
p=input('enter the periodicity:   ');
if mod(p,2)==0
    a=1+(p/2);
    b=w-(p/2);
    %disp(a);
    %disp(b);
    for i=a:1:b
        c=i+1-(p/2);
        q=i-1+(p/2);
        for j=c:1:q
            m(j)=x(j);
        end
        e=sum(m);
        d(i-a+1)=((x(i-(p/2)))+(x(i+(p/2)))+(2*e))/(2*p);
     %   disp(d(i-a+1));
       m=0;
    end
else
a=1+(p/2)-0.5;
b=w-(p/2)-0.5;
%disp(a);
%disp(b);
for i=a:1:b
   c=i-(p/2)+0.5;
   q=i+(p/2)-0.5;
   %disp(c);
   %disp(d);
   for j=c:1:q
      m(j)=x(j);
   end
   e=sum(m);
   d(i-a+1)=e/p;
   % disp(d(i-a+1));
   m=0;
end
end
% disp(d);
z=a:1:b;
n=polyfit(z,d,1);
% disp(n);
t=n(1);
l=n(2);
% disp('the deseasonalized demand is');
for i=1:1:w
  r(i)=l+(i*t);
  s(i)=x(i)/r(i);
```

```
end
disp('the initial level is:   ');
disp(l);
disp('the initial trend is:   ');
disp(t);
disp('the first ');
disp(p);
disp('seasonal factors is:   ');
for i=1:1:p
   disp(s(i));
end
```

FUNCTION winter1:

```
% to find the optimized value of alpha and beta

filename=input('enter the product name:   ');
x=xlsread(filename);
[t,l,s,p] = winterinit(x);  %finding the value of alpha for winters
w=length(x);
r(1)=l;
e(1)=t;
for i=1:1:p
q(i)=s(i);
end
f(1)=(r(1)+e(1))*q(1);
y=0.1;   % keeping the value of gamma as constant
b=0.2;      % keeping the value of beta as constant
for  i=1:1:9
    a=i/10;
    disp('the value of a is:  ');
    disp(a);
    for j=1:1:w
       r(j+1)=(a*(x(j)/q(j)))+((1-a)*(r(j)+e(j)));
       e(j+1)=(b*(r(j+1)-r(j)))+((1-b)*(e(j)));
       q(j+p)=(y*(x(j)/r(j+1)))+((1-y)*q(j));
       f(j+1)=(r(j+1)+e(j+1))*q(j+1);
      g(j)=((f(j)-x(j))^2)/w;
    end
  m(i)=sum(g);
  disp('the error involved is:   ');
  disp(m(i));
end
disp('the least mse error is:   ');
n=min(m);
disp(n);
```

APPENDIX D

Chapter 6- Development of a New Model for New Products

To calculate sales with and without repeat Purchase

```
e=2.71828;
a=input('any chance of multiple or repeated purchases enter 1.)Y or 2.)N  \n');
t= input('enter the period of forecast');
V = zeros(0,100);
h = zeros(0,100);
S = zeros(0,100);
R = zeros(0,100);
P = input('enter the values of innovation constant\n');
Q = input('enter the values of immitation constant\n');
R= input('enter the value of promotion constant');
N = input('enter the value of total market potential assuming one time purchase\n');
p=P*R;
q=Q*R;
h(t)= power(e,((p+q)*(-(t-1))));
K =((N*(p+q)*(p+q))/(p));
r= q/p;
S(t)= ((K*h(t))/(power((1+(r*h(t))),2)));
if a == 2
fprintf('S(%d)=%f\n',t-1,S(t));
elseif a == 1
TU = input('enter the value of Trail units\n');
RR = input('enter the value of repeat rate\n');
```

```
RU = input('enter the value of Repeat units\n');
m = input('enter the number of people doing repeat purchases\n');
V(t)=(S(t) * TU) + (m* RR * RU);
fprintf('V(%d)=%f\n',t-1,V(t));
else
fprintf('invalid variables');
end
```

To calculate Sales using Regression Equations

```
t= input('enter the period which you want to foreast\n');
if t == 3
fprintf('enter the actual sales of period %d \n', t-3);
j = input(' ');
fprintf('enter the actual sales of period %d \n', t-2);
k = input(' ');
fprintf('enter the actual sales of period %d \n', t-1);
l = input(' ');
a=0;
aa=0;
b=j+a;
      bb = b*b;
c  = j+k+a;
cc = c*c;
elseif t>=4
fprintf('enter the cumulative sales up to period %d \n',t-4);
a= input(' ');
```

```
fprintf('enter the actualsales of period %d \n', t-3);

j = input(' ');

fprintf('enter the actual sales of period %d \n', t-2);

k = input(' ');

fprintf('enter the actual sales of period %d \n', t-1);

l = input(' ');

aa = a*a;

b  = j+a;

bb = b*b;

c  = j+k+a;

cc = c*c;

else

fprintf( 'invalid to execute regression analysis as three equations are needed for three unknown variables ');

break;

end

      fprintf('THE REGRESSION EQUATIONS ARE :\n');

fprintf('x + %f y + %f z - %f = 0 \n',a,aa,j);

fprintf('x + %f y + %f z - %f = 0 \n',b,bb,k);

fprintf('x + %f y + %f z - %f = 0 \n',c,cc,l);

D = (1*b*cc+a*1*bb+aa*1*c)-(1*bb*c+a*1*cc+aa*b*1);

x = ((a*cc*-k+aa*b*-l+(-j*bb*c))-(a*bb*-l+aa*c*-k+(-j*b*cc)))/D;

y = ((1*bb*-l+aa*1*-k+(-j*1*cc))-(1*cc*-k+aa*1*-l+(-j*bb*1)))/D;

z = ((1*c*-k+a*1*-l+(-j*b*1))-(1*b*-l+a*1*-k+(-j*1*c)))/D;

q = (y+sqrt((y*y)-(4*x*z)))/2;
```

```
p = q-y;

M = x/p;

fprintf('new innovation paremeter is%f \n',p);

fprintf('new immitation paremeter is%f \n',q);

fprintf('remaining sales are %f \n',M);

e=2.71828;

    N= input('enter the value of total market potential assuming one time purchase\n');

h=power(e,((p+q)*(-t)));

K=((N*(p+q)*(p+q))/(p));

r= q/p;

S= ((K*h)/(power((1+(r*h)),2)));

w = input('any chance of multiple or repeated purchases enter 1.Y or 2.N  \n');

if w == 1

TU = input('enter the value of Trail units\n');

RR = input('enter the value of repeat rate\n');

RU = input('enter the value of Repeat units\n');

m = input('enter the number of people doing repeat purchases\n');

V=(S * TU) + (m* RR * RU);

    fprintf('the expected sales for period %d are %f\n',t,V);

elseif w == 2

    fprintf('the expected sales for period %d are %f\n',t,S);

else

fprintf('invalid input');

end
```