# Forecasting Demand

*Natalia Clivio*

*November, 2014*

This code is for predict the CV subscribers data and write in a cvs file the forecasting and the model´s performance

## Load and clean Data

The file loaded is the CV subscribers from 2005 up to 2014, for each service. The service offering by the company are: Internet residential and business, Video on demand and VoIP.

```
library(caret)
library(forecast)
library(grofit)
library(growthmodels)

subs<-read.csv("C:/Users/NataliaA/Documents/DataR/Subscribers_CV.csv",header=TRUE,
               sep=";",na.strings="NA",dec=",")
data_subs<-na.exclude(subs[1:5])
```

*Falta actualizar los datos*

```
data_subs
```

```
##    Year Internet_Res Internet_Bus     VoD  VoIP
## 1  2005       400000            0     100     0
## 2  2006       500000            0     200     0
## 3  2007       676237         1000     300     0
## 4  2008       900095         2000     400     0
## 5  2009       972130         3000    5000     0
## 6  2010      1130214         4000  100000  1000
## 7  2011      1356488         5000  800000  2000
## 8  2012      1519591         6000 1000000  4000
## 9  2013      1701668        34020 1242097 17776
## 10 2014      1800000        35000 1300000 20000
```

## Data Preparation

```
inTrain<-createDataPartition(y=data_subs$Year, p=0.75, list=FALSE)
training<-data_subs[inTrain,]
testing<-data_subs[-inTrain,]
```

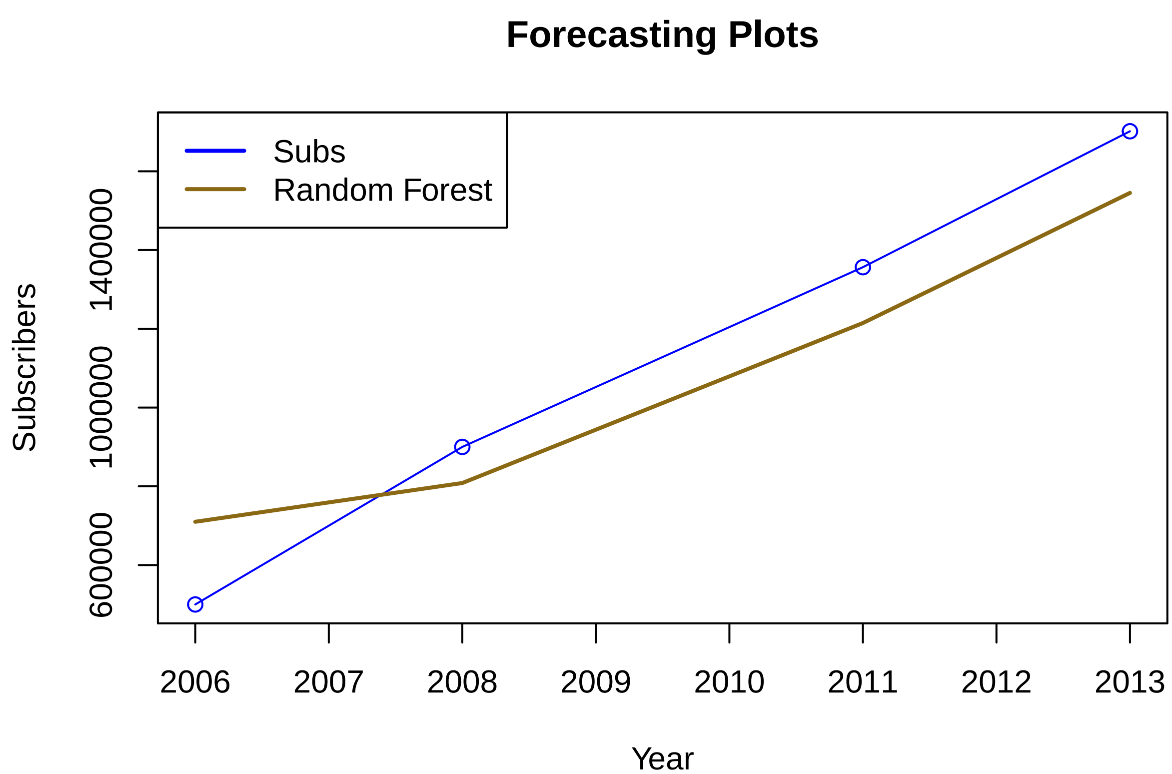
## Demand Models for Internet Residential Subscribers

```r
trainres<-data.frame(Year=training$Year,Subs=training$Internet_Res)
testres<-data.frame(Year=testing$Year,Subs=testing$Internet_Res)
```

### 1.Predicting with Random Forest

```r
modrf<-train(Subs~.,data=trainres,method="rf",prox=TRUE)   #Model
predrf <- predict(modrf, testres)                          #Prediction

fore1<-data.frame(testres,Random_Forest=predrf)
fore1
```

```
##   Year    Subs Random_Forest
## 1 2006  500000        709942
## 2 2008  900095        808335
## 3 2011 1356488       1214710
## 4 2013 1701668       1545074
```



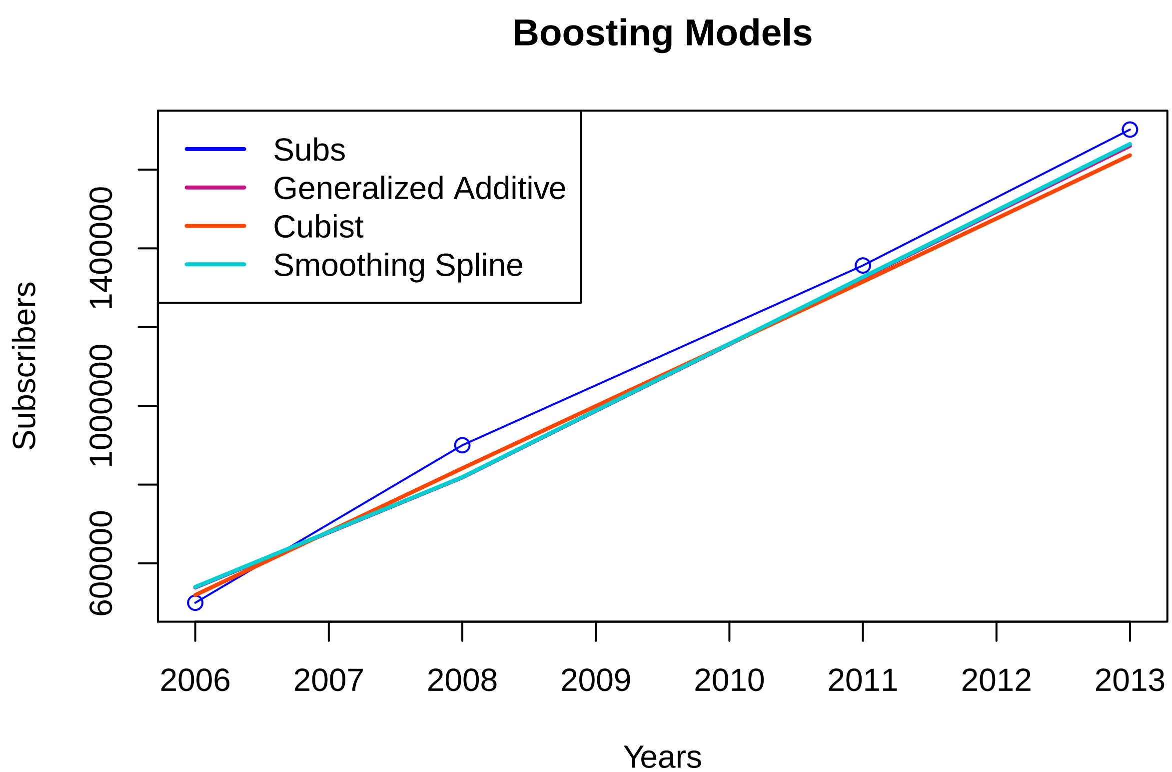

### 2. Predicting with Boosting

```r
#Boosted Generalized Additive Model
modbsgam <- train(Subs ~ ., method = "gamboost", data = trainres)
predbsgam <- predict(modbsgam, testres)

#Cubist
modbscub <- train(Subs ~ ., method = "cubist", data = trainres)
predbscub <- predict(modbscub, testres)

#Boosted Smoothing Spline
modbsSm <- train(Subs ~ ., method = "bstSm", data = trainres)
predbsSm <- predict(modbsSm, testres)
```

The predictions with boosting models are:

```r
fore2<-data.frame(fore1,Pred_Bs_gam=predbsgam,Pred_Bs_cub=predbscub,
                  Pred_Bs_Sm=predbsSm)
fore2
```

```
##   Year    Subs Random_Forest Pred_Bs_gam Pred_Bs_cub Pred_Bs_Sm
## 1 2006  500000        709942      538744      519578     540155
## 2 2008  900095        808335      818277      841603     819068
## 3 2011 1356488       1214710     1325454     1315237    1326945
## 4 2013 1701668       1545074     1660302     1636334    1665014
```



## 3. Predicting with Growth curves

Simple random sampling of time series is probably not the best way to resample times series data. Hyndman and Athanasopoulos (2013)) discuss rolling forecasting origin techniques that move the training and test sets in time .

```
tssubs<-ts(data_subs$Internet_Res,start=2005,end=2014)
tsset<-tssubs/1000000

tstrain<-window(tssubs/1000000,start=2005,end=2011)
tstest<-window(tssubs/1000000,start=2011,end=2014)

year<-c(2005:2014)
years<-c(2005:2011)
years2<-c(2011:2014)
```

#### Linear Model

```
fitlm<-tslm(tstrain~trend)
fitlm
```

```
##
## Call:
## lm(formula = formula, data = "tstrain", na.action = na.exclude)
##
## Coefficients:
## (Intercept)        trend
##       0.216        0.158
```

```
predlm<-forecast(fitlm, h=5)

summary(predlm)
```

```
##
## Forecast method: Linear regression model
##
## Model Information:
##
## Call:
## lm(formula = formula, data = "tstrain", na.action = na.exclude)
##
## Coefficients:
## (Intercept)        trend
##       0.216        0.158
##
##
## Error measures:
##                    ME    RMSE     MAE     MPE  MAPE   MASE    ACF1
## Training set 2.379e-17 0.03398 0.03227 0.01266 4.249 0.2024 -0.3588
##
## Forecasts:
```

```
##      Point Forecast Lo 80 Hi 80 Lo 95 Hi 95
## 2012          1.480 1.402 1.558 1.345 1.615
## 2013          1.638 1.554 1.723 1.491 1.786
## 2014          1.796 1.704 1.889 1.635 1.957
## 2015          1.954 1.853 2.055 1.779 2.130
## 2016          2.112 2.003 2.222 1.921 2.304
```

### Parabolic Model

```r
time<-1:10
fitpar=lm(tsset ~ time + I(time^2))

predpar<-predict(fitpar)
```

### Exponential Model

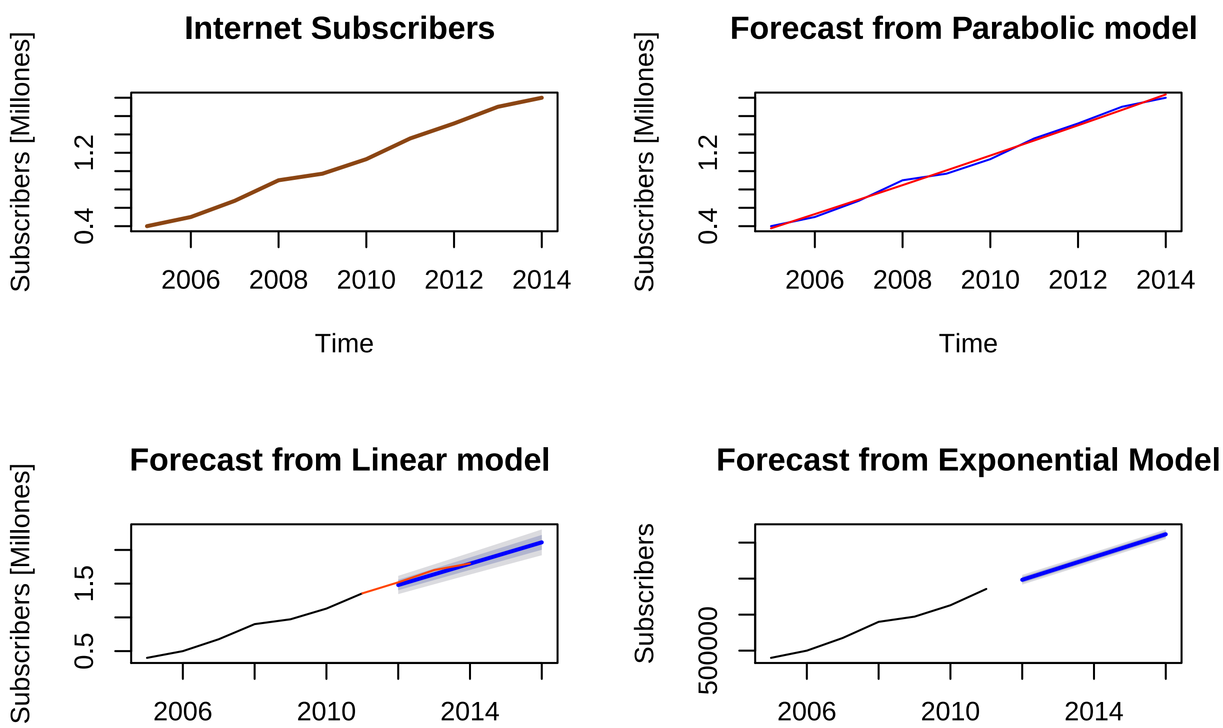
In this case turn the unit Million since this model had errors with the initial data set.

```r
#Exponential smoothing state space model

fitexp <- ets(tstrain*1000000)
predexp<-forecast(fitexp,h=5)

predexp
```

```
##      Point Forecast   Lo 80   Hi 80   Lo 95   Hi 95
## 2012        1483394 1439645 1527143 1416486 1550302
## 2013        1641455 1597706 1685204 1574547 1708363
## 2014        1799516 1755767 1843265 1732608 1866424
## 2015        1957577 1913829 2001326 1890669 2024485
## 2016        2115639 2071890 2159387 2048730 2182547
```

Plots

```r
par(mfrow = c(2, 2))

plot(tsset,lwd=2,main="Internet Subscribers ",ylab="Subscribers [Millones]",col="chocolate4")

plot(tsset,lwd=1,main="Forecast from Parabolic model ",ylab="Subscribers [Millones]",col="blue")
lines(year,predpar, col="red",lwd=1)

plot(predlm,main="Forecast from Linear model ",ylab="Subscribers [Millones]")
lines(tstest,lwd=1,col="orangered")

plot(predexp,main="Forecast from Exponential Model",ylab="Subscribers")
lines(tstest,lwd=1,col="green2")
```




The predictions with growth curves are:

```
set1<-data.frame(predlm)
x<-set1$Point.Forecast
set3<-data.frame(predexp)
y<-set3$Point.Forecast
z<-predpar[8:10]

set2<-data.frame(Year=c(2012:2016),Linear=x,Exponencial=y/1000000)
set4<-subset(set2,Year<=2014)
fore3<-data.frame(set4,Parabolic=z)
fore3
```

```
##   Year Linear Exponencial Parabolic
## 1 2012  1.480       1.483     1.499
## 2 2013  1.638       1.641     1.667
## 3 2014  1.796       1.800     1.836
```

## 4.Predicting with Logistic Model

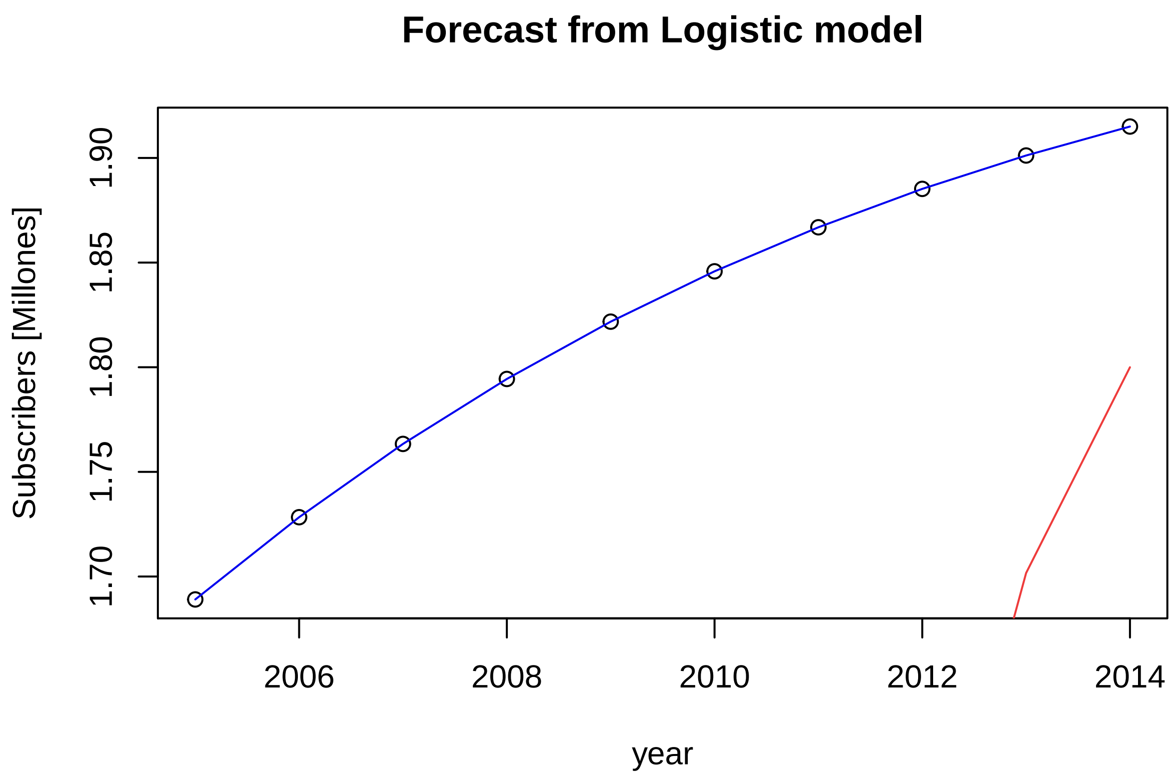
### Logistic Model

Using the **growthmodels** package, with the *logistic* function to get the logistic curve

**Usage** logistic(t, alpha, beta, k)

**Arguments** t time x size alpha upper asymptote beta growth range k growth rate

```r
parmlm<-as.list(fitlm$coeff)

alpha<-2 #upper asymptote (M)
beta<- parmlm$"(Intercept)"  #0.2156 growth range (a)
k<- parmlm$trend             #0.1581 growth rate (b)

fitlog <- logistic(1:10, alpha, beta, k)

plot(year,fitlog,main="Forecast from Logistic model",ylab="Subscribers [Millones]")
lines(year,fitlog,col="blue2")
lines(tstrain,lwd=1,col="brown2")
lines(tstest,lwd=1,col="brown2")
```




```r
fore4<-data.frame(fore3,Logistic=fitlog[8:10])
fore4
```

```
##   Year Linear Exponencial Parabolic Logistic
## 1 2012  1.480       1.483     1.499    1.885
## 2 2013  1.638       1.641     1.667    1.901
## 3 2014  1.796       1.800     1.836    1.915
```

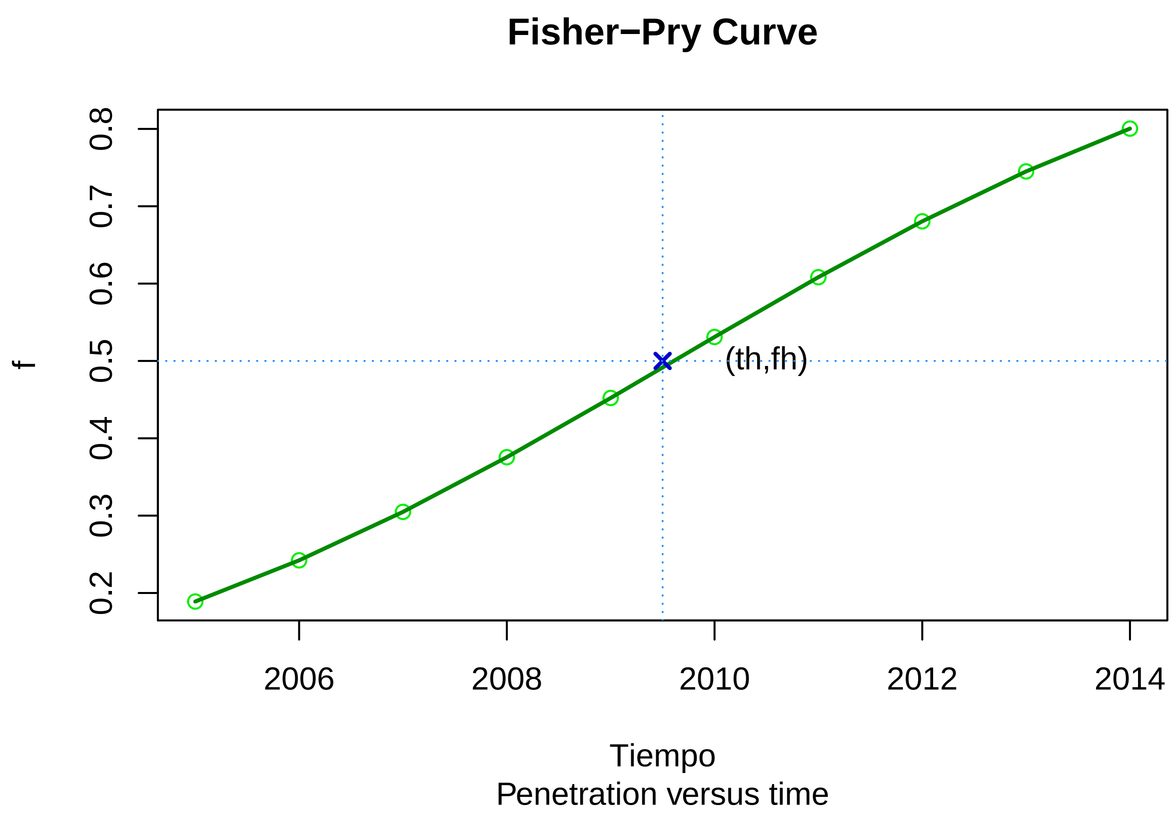
### 5.Modelos Fisher Pry

Model applied When substitution is driven by superior technology. The new product or service presents some technological advantage over the old one.

```
time<-(year-mean(year))*2

parmlm<-as.list(fitlm$coeff)

a<- parmlm$"(Intercept)"
b<- parmlm$trend

fitpry<-1/(1+exp(-b*(time-a)))
```

The shape curve **S** represents the adoption of the service, this is the market penetration.

```
plot(year,fitpry,lwd=1,col="green2", main="Fisher-Pry Curve",sub="Penetration versus time", xlab="Tiempo
lines(year,fitpry,lwd=2,col="green4")
points(mean(year),0.5,lwd=2,col="blue3",pch=4)
text(mean(year)+1, 0.5, "(th,fh)")
points(2014,1,lwd=2,col="blue3",pch=4)
text(2014, 0.95, "(to)")
abline(h=0.5,v=mean(year),lty=3,col="dodgerblue")
```



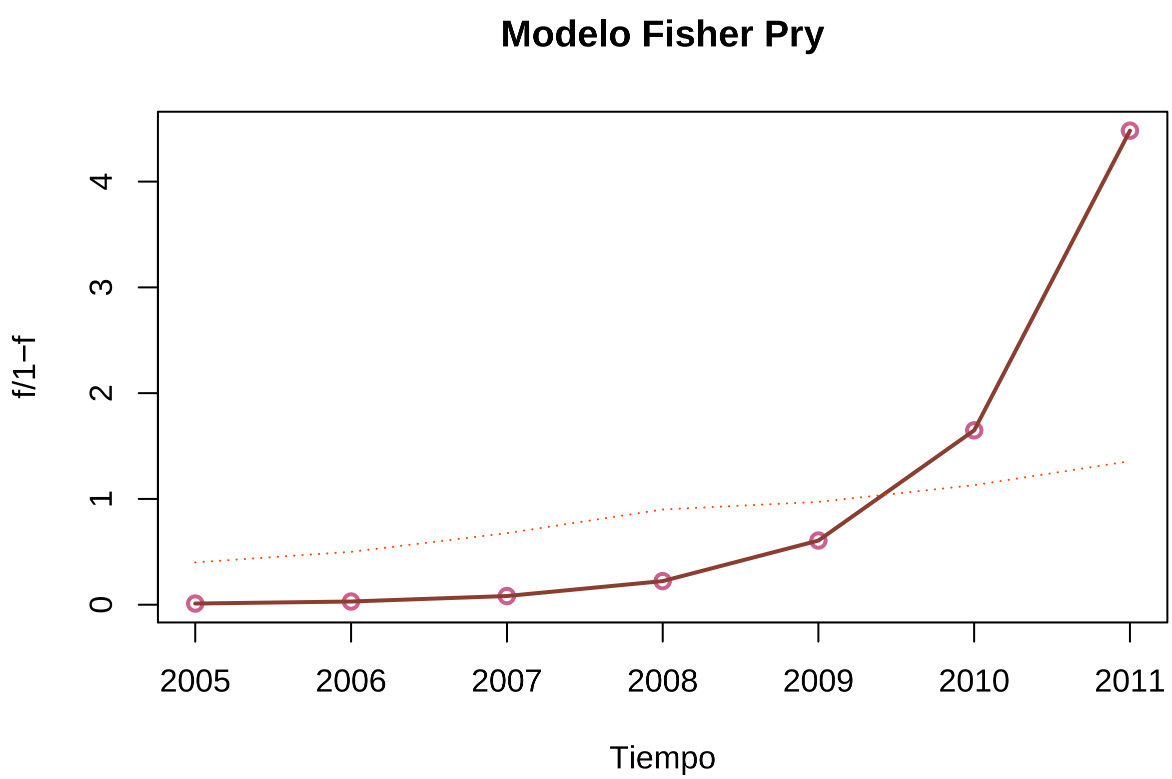

*Simple substitution Model* Its "take over time" defined as the time required to go from f=0.1 to f=0.9. This is inversely proportional to alpha.

**f/(1-f) = exp.2alpha(t-t0)**

where: f= old technology fraction replaced by the new. alpha= 1/2 annual percentage growth in the early years. to= time when f is 1/2

This expression allows one to plot the substitution data in the form of f/(1-f),the resulting points as ilustrated follow:

```
alpha<-2*0.5
to<-mean(year)
tdelta<-to-years

modpry<-exp(alpha*(years-to))

plot(years,modpry,lwd=2,col="hotpink3", main="Modelo Fisher Pry", xlab="Tiempo",ylab="f/1-f")
lines(years,modpry,lwd=2,col="coral4")
lines(tstrain,lty=3,lwd=1,col="orangered")
```




## 6.Modelos Bass Model

This model was developed by Frank Bass in 1969 and it consists of a simple differential equation that describes the process of how new products get adopted in a population. The basic premise of the model is that adopters can be classified as innovators or as imitators and the speed and timing of adoption depends on their degree of innovativeness and the degree of imitation among adopters.

**m** Total number of potential buyers of the new product **p** The coefficient of innovation **q** The coefficient of imitation

In order to test the model, linear regression estimates the parameters of the model

```r
time<-(years-mean(years))*2
tsset<-tssubs/1000000

regpol= lm(tstrain ~ time + I(time^2))

parm<-as.list(regpol$coeff)

a<-parm$"(Intercept)"
b<-parm$time*(1)
c<-parm$"I(time^2)"

Par<-data.frame(a=a,b=b,c=c)
Par
```

```
##        a       b         c
## 1 0.8366 0.07903 0.0007052
```

```r
#Bass Model Coefficients

P<-c
Q<-b+P
M<-a/P

#M<-(-b-sqrt((b^2)-(4*a*c)))/(2*c)
#P<-a/M
#Q<--c*M


Coeff<-data.frame(P=P,Q=Q,M=M)
Coeff
```

```
##           P       Q    M
## 1 0.0007052 0.07974 1186
```

The coefficient p is called the coefficient of innovation, external influence or advertising effect. The coefficient q is called the coefficient of imitation, internal influence or word-of-mouth effect.

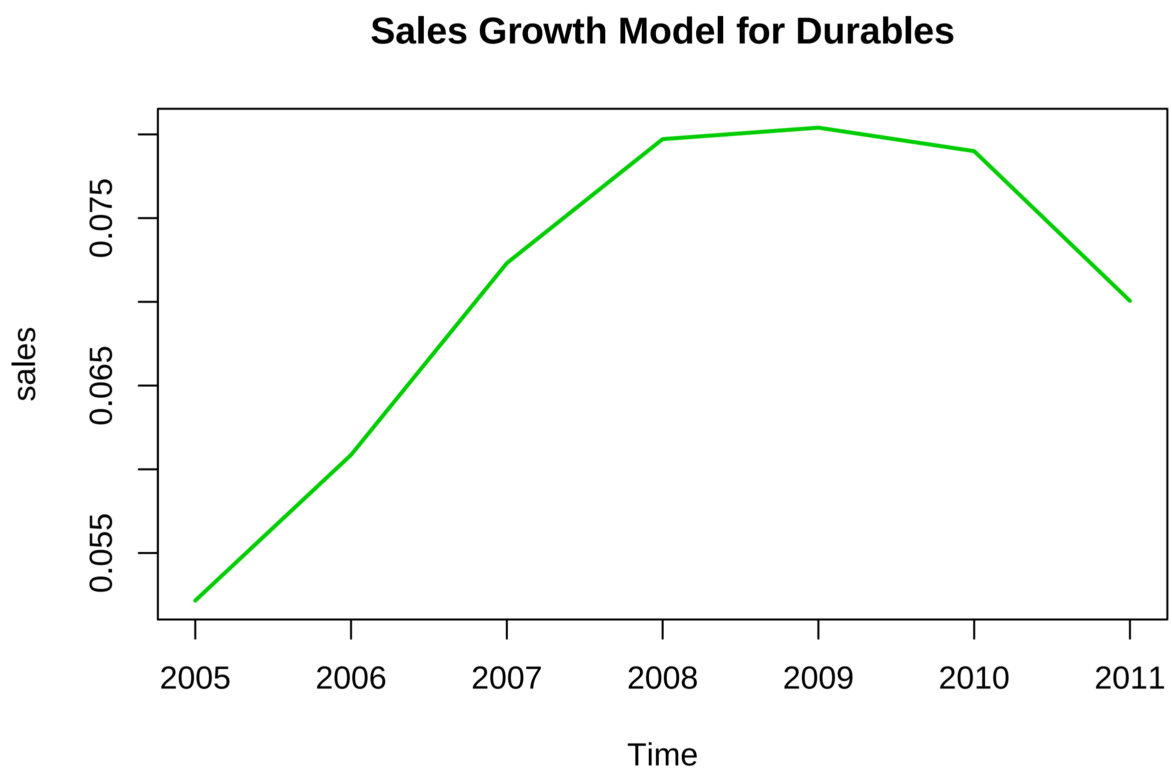
The Sales Growth Model for Durables are determinated by:

**S(t) = Innovation effect + Imitation effect**

where: S(t) Sales at time t Innovation effect = p * Remaining Potencial Imitation effect = q * Adopters *Remaining Potential Remaining Potencial = Total Potential - Q Adopters

```r
#Remaining Potencial
Rem<-2-tstrain

#Innovation effect
pe<-P*Rem

#Imitation effect
qe<-Q*tstrain*Rem

#Sales at time t
```

```r
sales<-pe+qe

plot(sales, main="Sales Growth Model for Durables",lwd=2,lty=1,col="green3")
lines(tstrain,col="orangered")
```




The Bass Model proposes that the likelihood that someone in the population will purchase a new product at a particular time t given that she has not already purchased the product until then, is summarized by the following simplification mathematical.

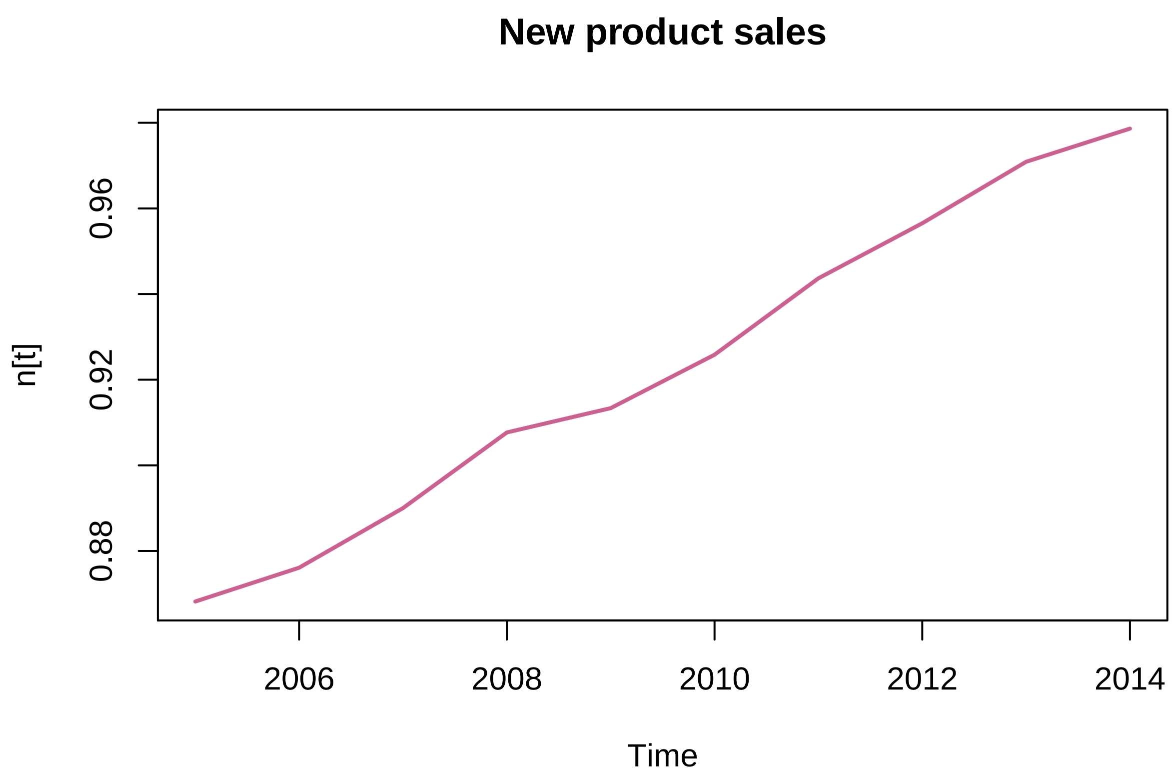
**n(t) = pm + (q - p) [N(t)] -q/m * [N(t)]2**

where: n(t) = basic diffusion equation for predicting new product sales N(t) = Adopters at time t

```r
ter1<-P*M
ter2<-(Q-P)*tsset
ter3<-(Q/M)*tsset^2

fitbas<-ter1+ter2-ter3
fitbas
```

```
## Time Series:
## Start = 2005
## End = 2014
## Frequency = 1
##  [1] 0.8682 0.8761 0.8900 0.9077 0.9134 0.9258 0.9437 0.9565 0.9709 0.9786
```

```r
plot(fitbas,ylab="n[t]",col="hotpink3",lwd=2,main="New product sales")
```




### Comparación de Modelos

```r
#Modelo Lineal
acc_a<-accuracy(predlm)
#Modelo Parabólico
acc_b<-accuracy(predpar,tsset)
#Modelo Exponencial
acc_c<-accuracy(predexp)
#Modelo Logístico
acc_d<-accuracy(fitlog,tsset)
#Modelo Gompertz
#acc_e<-accuracy(predgom)
#Modelo Fisher-Pry
acc_f<-accuracy(fitpry,tsset) #Duda
#Modelo Bass
acc_g<-accuracy(fitbas,tsset)        #Duda

#Extract ME,RMSE, MAE, MPE, MAPE
acca<-acc_a[1,1:5]
accb<-acc_b[1,1:5]
accc<-acc_c[1,1:5]
```

```r
accd<-acc_d[1,1:5]
accf<-acc_f[1,1:5]
accg<-acc_g[1,1:5]

acc_all<-rbind(acca,accb,accc,accd,accf,accg)
Modelos<-c("Linear","Parabolic", "Exponential"
          ,"Logistic","Fisher-Pry","Bass")

perform<-data.frame(Modelos,acc_all)

perform
```

```
##          Modelos         ME      RMSE       MAE        MPE    MAPE
## acca      Linear  2.379e-17 3.398e-02 3.227e-02    0.01266   4.249
## accb   Parabolic  1.110e-16 3.281e-02 3.093e-02   -0.04004   3.414
## accc Exponential -3.269e+03 3.414e+04 3.274e+04   -0.44237   4.298
## accd    Logistic -7.255e-01 8.266e-01 7.255e-01 -105.85594 105.856
## accf  Fisher-Pry  6.028e-01 6.582e-01 6.028e-01   54.61448  54.614
## accg        Bass  1.726e-01 4.634e-01 3.857e-01   -4.45337  40.491
```

Extract parameters model

```
##       Modelos     a     b     c     p    q        m
## 1      Linear 0.216 0.158    NA    NA   NA       NA
## 2   Parabolic 0.223 0.153 0.001    NA   NA       NA
## 3 Exponential    NA    NA    NA    NA   NA       NA
## 4    Logistic    NA    NA    NA    NA   NA       NA
## 5  Fisher-Pry 0.216 0.158    NA    NA   NA       NA
## 6        Bass 0.223 0.153 0.001 0.001 0.08 1186.267
```

Write performance of the models in a **csv** file

```r
write.csv(perform,"Performance_models.csv")
```