Bass Model

Yeva Manukyan

2024-02-26

Medivis SurgicalAR and the look alike Head Up Display

In this work, we will be looking at the innovation of Medivis SurgicalAR. This device overlays 3D images onto the patient's body to plan complex procedures and surgeries accurately. The look-alike innovation chosen from the past is the Head Up Display (HUD). Important information is projected directly into the user's eye's view using this technology, commonly onto a transparent screen or surface like a windscreen. Although HUDs were initially developed for military aircraft, they have since been used in several sectors, including the aviation, surgical, and automobile industries.

By providing crucial information to surgeons during procedures without requiring them to take their eyes off the patient, Medivis SurgicalAR's goal is to assist doctors. Medivis Surgical AR aims to improve patient care and make procedures faster, just as HUDs helped make driving and flying safer and more efficient. Examining past HUD implementations might provide valuable information that will help improve Medivis SurgicalAR's usefulness for patients and surgeons.

```
suppressWarnings({data <- read.xlsx("head_up_display_data.xlsx",
                                    rows = 5:17,
                                    colNames = TRUE,
                                    sheet = 2)

data[, c("Others", "Wearables", "Aviation", "Automotive")] <- sapply(data[,
                     c("Others", "Wearables", "Aviation", "Automotive")], as.numeric)

data[is.na(data)] <- 0

data$All <- rowSums(data[, c("Others", "Wearables", "Aviation", "Automotive")],
                    na.rm = TRUE)

colnames(data)[1] <- "Year"
print(data)

})
```

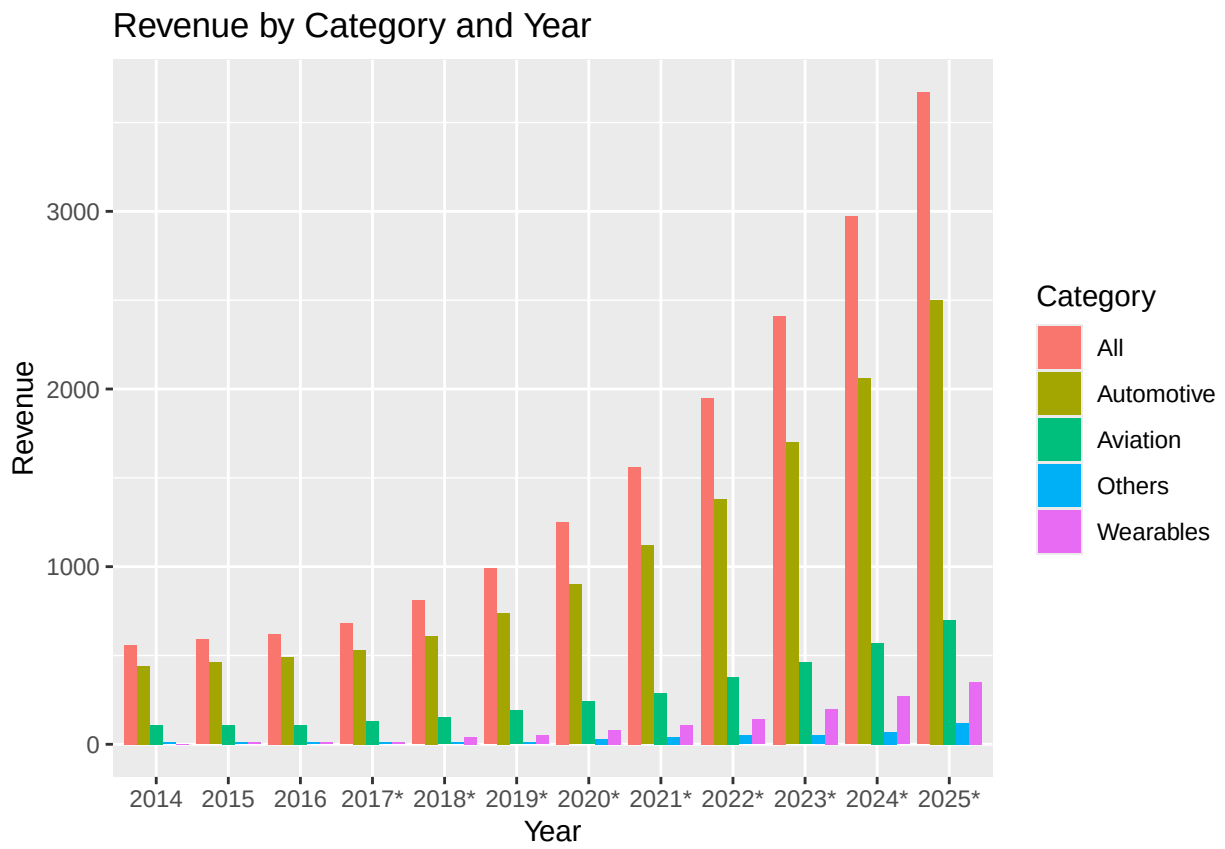
```
##     Year Others Wearables Aviation Automotive  All
## 1   2014     10         0      110        440  560
## 2   2015     10        10      110        460  590
## 3   2016     10        10      110        490  620
## 4  2017*     10        10      130        530  680
## 5  2018*     10        40      150        610  810
## 6  2019*     10        50      190        740  990
## 7  2020*     30        80      240        900 1250
```

```
## 8  2021*     40       110      290       1120 1560
## 9  2022*     50       140      380       1380 1950
## 10 2023*     50       200      460       1700 2410
## 11 2024*     70       270      570       2060 2970
## 12 2025*    120       350      700       2500 3670
```

Time series data

This is the HUD time series data starting from 2014 - 2023 The data originally had columns "Others," 'Wearables," "Aviation," and "Automotive," which described the revenue generated from sales of each type. I created a new column, "All," to calculate the overall revenue for each year (in millions of dollars).

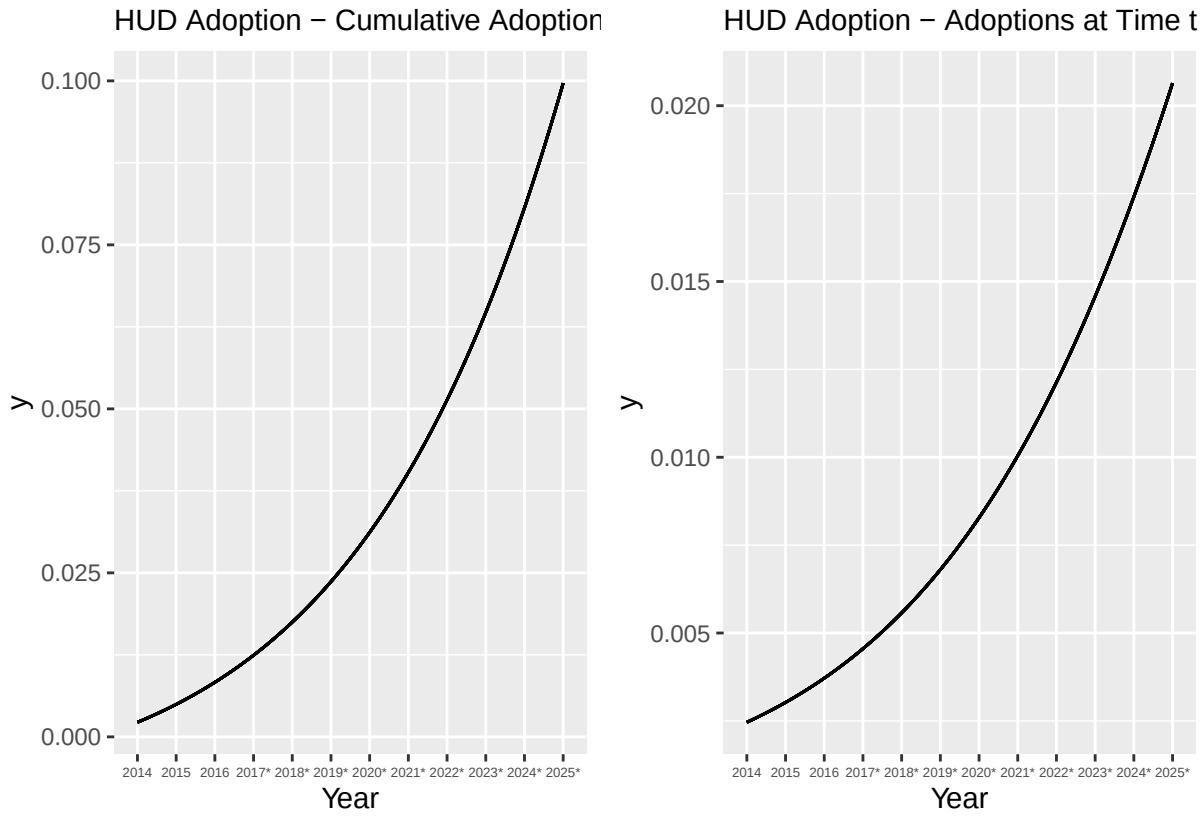
```
data_long <- tidyr::pivot_longer(data, cols = -Year,
                                 names_to = "Category",
                                 values_to = "Revenue")

ggplot(data_long, aes(x = as.factor(Year),
                      y = Revenue,
                      fill = Category)) +
  geom_bar(stat = "identity",
           position = "dodge") +
  labs(title = "Revenue by Category and Year",
       x = "Year",
       y = "Revenue",
       fill = "Category")
```



Here, we can see the bar plot for the data for the whole revenue and the revenue by category. We can see that the overall revenue goes up.

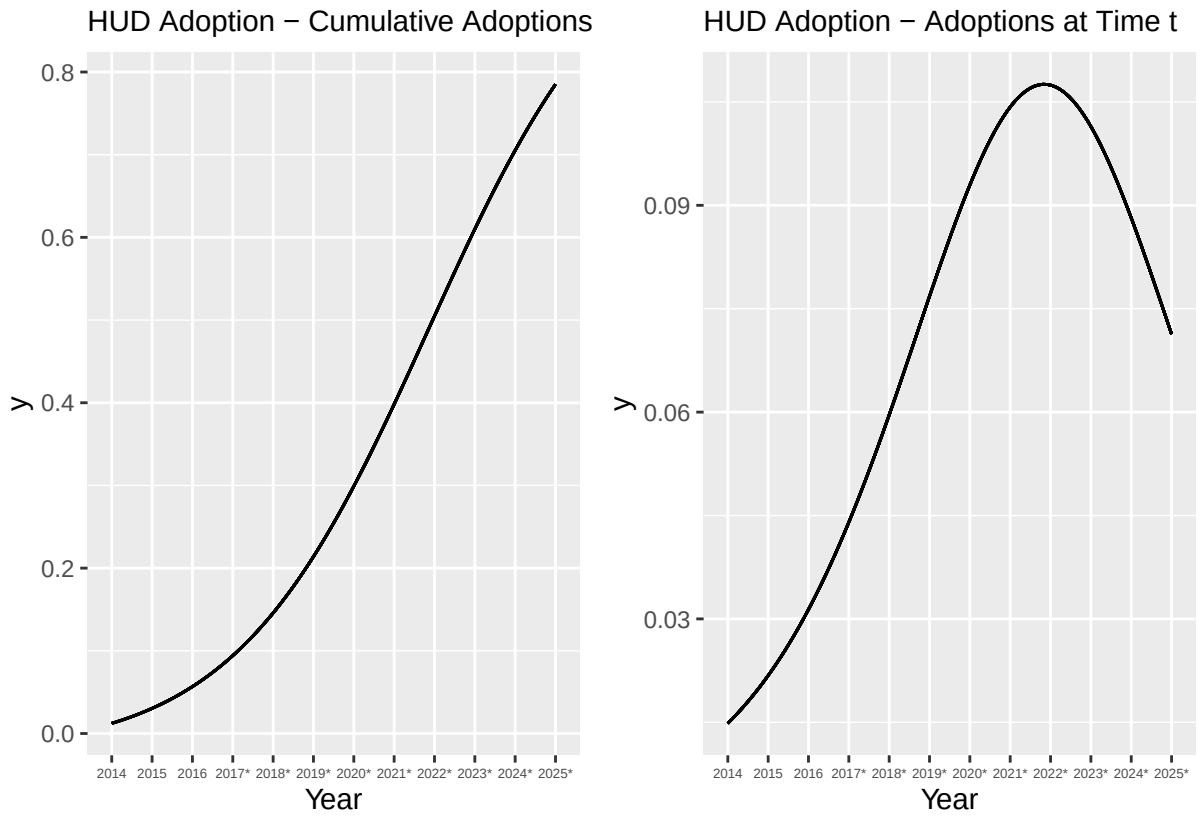
```
bass.f <- function(t,p,q){
((p+q)^2/p)*exp(-(p+q)*t)/
(1+(q/p)*exp(-(p+q)*t))^2
}
bass.F <- function(t,p,q){
(1-exp(-(p+q)*t))/
(1+(q/p)*exp(-(p+q)*t))
}

cum_ad <-ggplot(data = data,
                aes(x = Year)) +
  stat_function(fun = bass.F,
                args = list(p=0.002, q=0.21)) +
  labs(title = 'HUD Adoption - Cumulative Adoptions') +
  theme(
    axis.text.x = element_text(size = 5),
    plot.title = element_text(size = 11)

  )

time_ad<-ggplot(data = data,
                aes(x = Year)) +
  stat_function(fun = bass.f,
                args = c(p=0.002, q=0.21)) +
```

```
  labs(title = 'HUD Adoption - Adoptions at Time t') +
  theme(
    axis.text.x = element_text(size = 5),
    plot.title = element_text(size = 11)
  )

suppressWarnings({
  plot <- cum_ad + time_ad
  print(plot)
})
```




```
cum_ad_2 <-ggplot(data = data,
                  aes(x = Year)) +
  stat_function(fun = bass.F,
                args = list(p=0.01, q=0.41)) +
  labs(title = 'HUD Adoption - Cumulative Adoptions') +
  theme(
    axis.text.x = element_text(size = 5),
    plot.title = element_text(size = 11)
  )

time_ad_2<-ggplot(data = data,
                  aes(x = Year)) +
  stat_function(fun = bass.f,
                args = c(p=0.01, q=0.41)) +
```

```
  labs(title = 'HUD Adoption - Adoptions at Time t')  +
  theme(
    axis.text.x = element_text(size = 5),
    plot.title = element_text(size = 11)
  )

suppressWarnings({
  plot_2 <- cum_ad_2 + time_ad_2
  print(plot_2)
})
```



Predictions of diffusion

```
diff_m = diffusion(data$All)
p=round(diff_m$w,4)[1]
q=round(diff_m$w,4)[2]
m=round(diff_m$w,4)[3]
print(diff_m)
```

```
## bass model
##
## Parameters:
```

```
##                               Estimate p-value
## p - Coefficient of innovation  17.7275      NA
## q - Coefficient of imitation    0.7607      NA
## m - Market potential           39.4983      NA
##
## sigma: 1802.7415
```

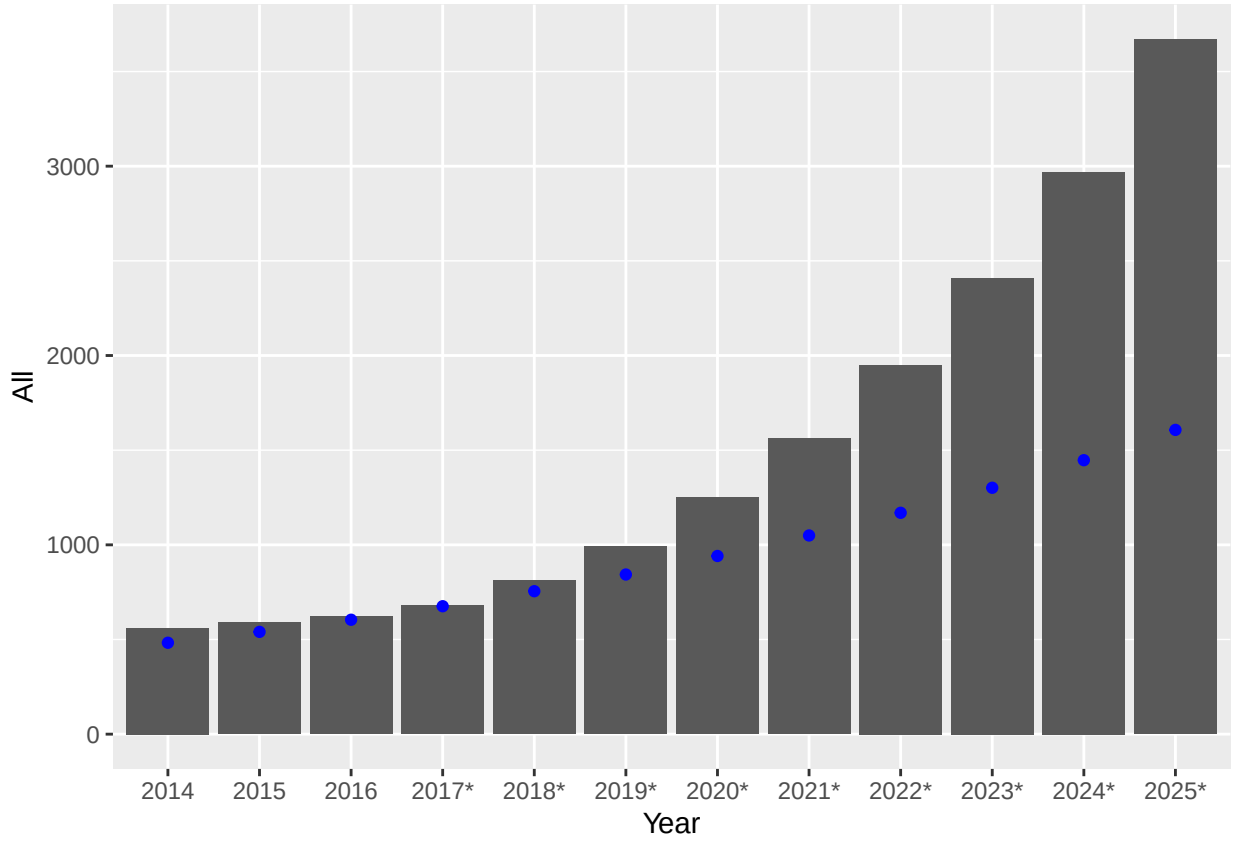
```
suppressWarnings({sales = data$All
t = 1:length(sales)
bass_m = nls(sales ~ m*(((p+q)**2/p)*exp(-(p+q)*t))/
               (1+(q/p)*exp(-(p+q)*t))**2,
             start=c(list(m=sum(sales),p=0.02,q=0.4)),
             control=nls.control(maxiter = 150,
                                 minFactor = 1/1024,
                                 printEval = TRUE,
                                 warnOnly = TRUE))
print(bass_m)
})
```

```
##   It.   1, fac=           1, eval (no.,total): ( 1,  1): new dev = 2.51279e+07
##   It.   1, fac=         0.5, eval (no.,total): ( 2,  2): new dev = 1.0578e+07
##   It.   2, fac=           1, eval (no.,total): ( 1,  3): new dev = 1.27047e+08
##   It.   2, fac=         0.5, eval (no.,total): ( 2,  4): new dev = 4.83667e+07
##   It.   2, fac=        0.25, eval (no.,total): ( 3,  5): new dev = 1.46458e+07
##   It.   2, fac=       0.125, eval (no.,total): ( 4,  6): new dev = 9.87617e+06
##   It.   3, fac=        0.25, eval (no.,total): ( 1,  7): new dev = 3.27546e+07
##   It.   3, fac=       0.125, eval (no.,total): ( 2,  8): new dev = 1.21019e+07
##   It.   3, fac=      0.0625, eval (no.,total): ( 3,  9): new dev = 9.66751e+06
##   It.   4, fac=       0.125, eval (no.,total): ( 1, 10): new dev = 1.6095e+07
##   It.   4, fac=      0.0625, eval (no.,total): ( 2, 11): new dev = 1.02566e+07
##   It.   4, fac=     0.03125, eval (no.,total): ( 3, 12): new dev = 9.48721e+06
##   It.   5, fac=      0.0625, eval (no.,total): ( 1, 13): new dev = 1.07703e+07
##   It.   5, fac=     0.03125, eval (no.,total): ( 2, 14): new dev = 9.46701e+06
##   It.   6, fac=      0.0625, eval (no.,total): ( 1, 15): new dev = 1.19524e+07
##   It.   6, fac=     0.03125, eval (no.,total): ( 2, 16): new dev = 9.70511e+06
##   It.   6, fac=    0.015625, eval (no.,total): ( 3, 17): new dev = 9.37331e+06
##   It.   7, fac=     0.03125, eval (no.,total): ( 1, 18): new dev = 9.79575e+06
##   It.   7, fac=    0.015625, eval (no.,total): ( 2, 19): new dev = 9.32486e+06
##   It.   8, fac=     0.03125, eval (no.,total): ( 1, 20): new dev = 1.00014e+07
##   It.   8, fac=    0.015625, eval (no.,total): ( 2, 21): new dev = 9.33686e+06
##   It.   8, fac=   0.0078125, eval (no.,total): ( 3, 22): new dev = 9.25437e+06
##   It.   9, fac=    0.015625, eval (no.,total): ( 1, 23): new dev = 9.3036e+06
##   It.   9, fac=   0.0078125, eval (no.,total): ( 2, 24): new dev = 9.19357e+06
##   It.  10, fac=    0.015625, eval (no.,total): ( 1, 25): new dev = 9.28687e+06
##   It.  10, fac=   0.0078125, eval (no.,total): ( 2, 26): new dev = 9.14405e+06
##   It.  11, fac=    0.015625, eval (no.,total): ( 1, 27): new dev = 9.2904e+06
##   It.  11, fac=   0.0078125, eval (no.,total): ( 2, 28): new dev = 9.10788e+06
##   It.  12, fac=    0.015625, eval (no.,total): ( 1, 29): new dev = 9.31931e+06
##   It.  12, fac=   0.0078125, eval (no.,total): ( 2, 30): new dev = 9.08785e+06
##   It.  13, fac=    0.015625, eval (no.,total): ( 1, 31): new dev = 9.38087e+06
##   It.  13, fac=   0.0078125, eval (no.,total): ( 2, 32): new dev = 9.08776e+06
##   It.  14, fac=    0.015625, eval (no.,total): ( 1, 33): new dev = 9.48574e+06
##   It.  14, fac=   0.0078125, eval (no.,total): ( 2, 34): new dev = 9.11302e+06
##   It.  14, fac=  0.00390625, eval (no.,total): ( 3, 35): new dev = 9.05843e+06
```

```
##   It.  15, fac=   0.0078125, eval (no.,total): ( 1, 36): new dev = 9.09865e+06
##   It.  15, fac=  0.00390625, eval (no.,total): ( 2, 37): new dev = 9.03292e+06
##   It.  16, fac=   0.0078125, eval (no.,total): ( 1, 38): new dev = 9.09019e+06
##   It.  16, fac=  0.00390625, eval (no.,total): ( 2, 39): new dev = 9.01173e+06
##   It.  17, fac=   0.0078125, eval (no.,total): ( 1, 40): new dev = 9.08861e+06
##   It.  17, fac=  0.00390625, eval (no.,total): ( 2, 41): new dev = 8.99547e+06
##   It.  18, fac=   0.0078125, eval (no.,total): ( 1, 42): new dev = 9.09512e+06
##   It.  18, fac=  0.00390625, eval (no.,total): ( 2, 43): new dev = 8.98489e+06
##   It.  19, fac=   0.0078125, eval (no.,total): ( 1, 44): new dev = 9.11123e+06
##   It.  19, fac=  0.00390625, eval (no.,total): ( 2, 45): new dev = 8.98093e+06
##   It.  20, fac=   0.0078125, eval (no.,total): ( 1, 46): new dev = 9.13893e+06
##   It.  20, fac=  0.00390625, eval (no.,total): ( 2, 47): new dev = 8.98478e+06
##   It.  20, fac=  0.00195312, eval (no.,total): ( 3, 48): new dev = 8.96435e+06
##   It.  21, fac=  0.00390625, eval (no.,total): ( 1, 49): new dev = 8.97258e+06
##   It.  21, fac=  0.00195312, eval (no.,total): ( 2, 50): new dev = 8.94889e+06
##   It.  22, fac=  0.00390625, eval (no.,total): ( 1, 51): new dev = 8.9619e+06
##   It.  22, fac=  0.00195312, eval (no.,total): ( 2, 52): new dev = 8.93465e+06
##   It.  23, fac=  0.00390625, eval (no.,total): ( 1, 53): new dev = 8.9529e+06
##   It.  23, fac=  0.00195312, eval (no.,total): ( 2, 54): new dev = 8.92175e+06
##   It.  24, fac=  0.00390625, eval (no.,total): ( 1, 55): new dev = 8.94575e+06
##   It.  24, fac=  0.00195312, eval (no.,total): ( 2, 56): new dev = 8.9103e+06
##   It.  25, fac=  0.00390625, eval (no.,total): ( 1, 57): new dev = 8.94065e+06
##   It.  25, fac=  0.00195312, eval (no.,total): ( 2, 58): new dev = 8.90045e+06
##   It.  26, fac=  0.00390625, eval (no.,total): ( 1, 59): new dev = 8.93783e+06
##   It.  26, fac=  0.00195312, eval (no.,total): ( 2, 60): new dev = 8.89237e+06
##   It.  27, fac=  0.00390625, eval (no.,total): ( 1, 61): new dev = 8.93756e+06
##   It.  27, fac=  0.00195312, eval (no.,total): ( 2, 62): new dev = 8.88624e+06
##   It.  28, fac=  0.00390625, eval (no.,total): ( 1, 63): new dev = 8.94018e+06
##   It.  28, fac=  0.00195312, eval (no.,total): ( 2, 64): new dev = 8.8823e+06
##   It.  29, fac=  0.00390625, eval (no.,total): ( 1, 65): new dev = 8.94607e+06
##   It.  29, fac=  0.00195312, eval (no.,total): ( 2, 66): new dev = 8.88081e+06
##   It.  30, fac=  0.00390625, eval (no.,total): ( 1, 67): new dev = 8.95568e+06
##   It.  30, fac=  0.00195312, eval (no.,total): ( 2, 68): new dev = 8.88207e+06
##   It.  30, fac= 0.000976562, eval (no.,total): ( 3, 69): new dev = 8.87246e+06
##   It.  31, fac=  0.00195312, eval (no.,total): ( 1, 70): new dev = 8.87523e+06
##   It.  31, fac= 0.000976562, eval (no.,total): ( 2, 71): new dev = 8.8645e+06
##   It.  32, fac=  0.00195312, eval (no.,total): ( 1, 72): new dev = 8.86886e+06
##   It.  32, fac= 0.000976562, eval (no.,total): ( 2, 73): new dev = 8.85694e+06
##   It.  33, fac=  0.00195312, eval (no.,total): ( 1, 74): new dev = 8.86301e+06
##   It.  33, fac= 0.000976562, eval (no.,total): ( 2, 75): new dev = 8.84982e+06
##   It.  34, fac=  0.00195312, eval (no.,total): ( 1, 76): new dev = 8.8577e+06
##   It.  34, fac= 0.000976562, eval (no.,total): ( 2, 77): new dev = 8.84316e+06
##   It.  35, fac=  0.00195312, eval (no.,total): ( 1, 78): new dev = 8.85298e+06
##   It.  35, fac= 0.000976562, eval (no.,total): ( 2, 79): new dev = 8.83698e+06
##   It.  36, fac=  0.00195312, eval (no.,total): ( 1, 80): new dev = 8.84889e+06
##   It.  36, fac= 0.000976562, eval (no.,total): ( 2, 81): new dev = 8.83133e+06
##   It.  37, fac=  0.00195312, eval (no.,total): ( 1, 82): new dev = 8.84547e+06
##   It.  37, fac= 0.000976562, eval (no.,total): ( 2, 83): new dev = 8.82625e+06
##   It.  38, fac=  0.00195312, eval (no.,total): ( 1, 84): new dev = 8.84279e+06
##   It.  38, fac= 0.000976562, eval (no.,total): ( 2, 85): new dev = 8.82177e+06
##   It.  39, fac=  0.00195312, eval (no.,total): ( 1, 86): new dev = 8.8409e+06
##   It.  39, fac= 0.000976562, eval (no.,total): ( 2, 87): new dev = 8.81794e+06
##   It.  40, fac=  0.00195312, eval (no.,total): ( 1, 88): new dev = 8.83986e+06
##   It.  40, fac= 0.000976562, eval (no.,total): ( 2, 89): new dev = 8.81481e+06
```

```
##   It.  41, fac=  0.00195312, eval (no.,total): ( 1, 90): new dev = 8.83976e+06
##   It.  41, fac= 0.000976562, eval (no.,total): ( 2, 91): new dev = 8.81243e+06
##   It.  42, fac=  0.00195312, eval (no.,total): ( 1, 92): new dev = 8.84068e+06
##   It.  42, fac= 0.000976562, eval (no.,total): ( 2, 93): new dev = 8.81088e+06
##   It.  43, fac=  0.00195312, eval (no.,total): ( 1, 94): new dev = 8.8427e+06
##   It.  43, fac= 0.000976562, eval (no.,total): ( 2, 95): new dev = 8.81022e+06
##   It.  44, fac=  0.00195312, eval (no.,total): ( 1, 96): new dev = 8.84595e+06
##   It.  44, fac= 0.000976562, eval (no.,total): ( 2, 97): new dev = 8.81054e+06
## Nonlinear regression model
##   model: sales ~ m * (((p + q)^2/p) * exp(-(p + q) * t))/(1 + (q/p) *     exp(-(p + q) * t))^2
##    data: parent.frame()
##         m         p         q
## 2.780e+05 1.551e-03 1.147e-01
##  residual sum-of-squares: 8810536
##
## Number of iterations till stop: 43
## Achieved convergence tolerance: 19.91
## Reason stopped: step factor 0.000488281 reduced below 'minFactor' of 0.000976562
```

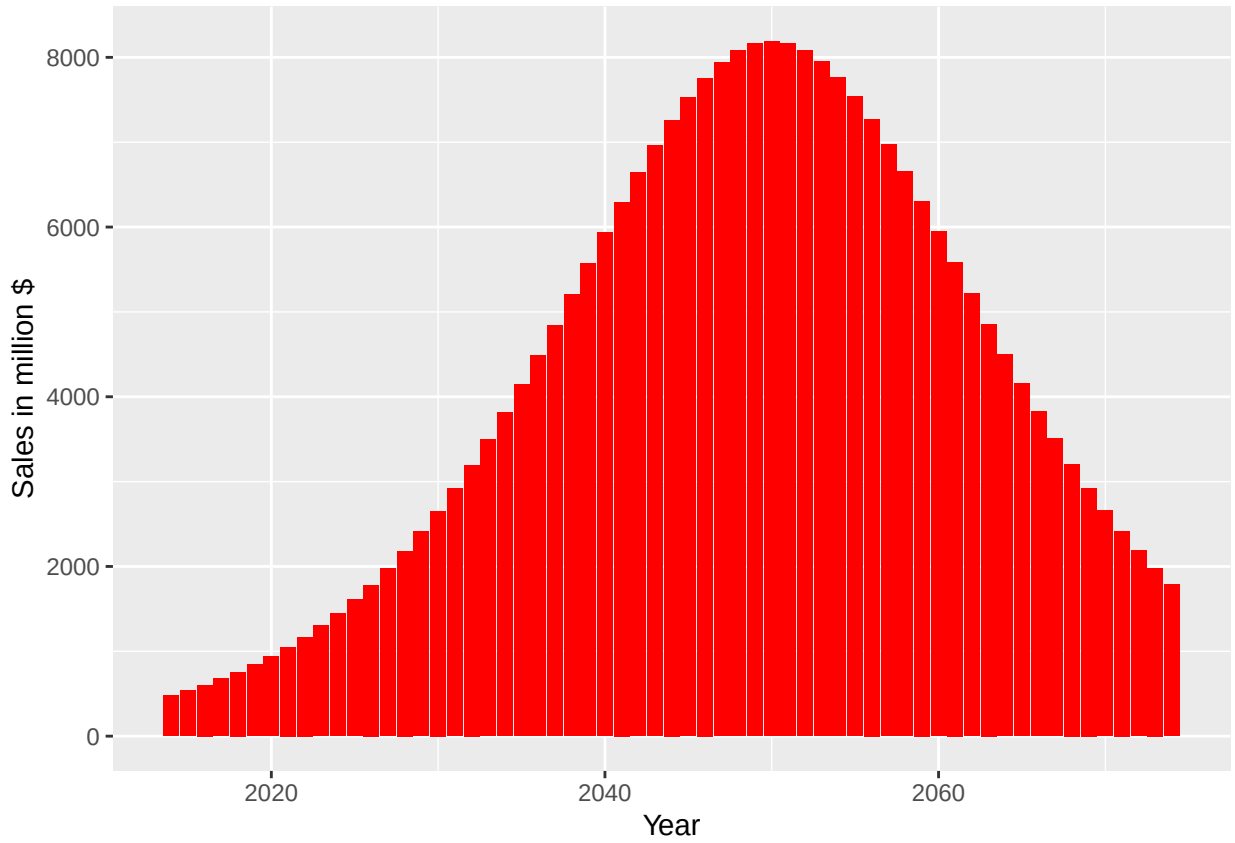
```
p <- 1.551e-03
q<-1.147e-01
m<-2.780e+05

suppressWarnings({data$Pred_sales = bass.f(1:12,
                                           p = p,
                                           q = q)*m
ggplot(data = data,
       aes(x = Year,
           y = All)) +
  geom_bar(stat = 'identity') +
  geom_point(mapping = aes(x=Year,
                           y=Pred_sales),
             color = 'blue')
})
```



The plot visualizes the difference between the estimated and actual revenue generated by sales. This shows that the prediction is not very good.

Prediction for the next 50 years

```
innovation_predicted <- bass.f(1:61,
                               p = p,
                               q = q) * m
years <- seq(from = 2014,
             to = 2025 + 49,
             by = 1)

innovation_data <- data.frame(Year = years,
                              Sales = innovation_predicted)

ggplot(data = innovation_data,
       aes(x = Year,
           y = Sales)) +
  geom_bar(stat='identity',
           fill = 'red') + ylab("Sales in million $")
```



We can see that the sales go up throughout the time. However, at some point they start declining due to market glut.

Estimate of the number of adopters by period

```
ad_by_period <- numeric(length(1:61))
cumulative_ad <- 0
for (i in 1:61) {
  new_adopters <- (p + (q * cumulative_ad / m)) * (m - cumulative_ad)
  ad_by_period[i] <- new_adopters
  cumulative_ad <- cumulative_ad + new_adopters
}

adoption_df<- data.frame(Year = seq(from = 2014,
                                    to = 2014 + length(1:61) - 1),
                         New_Adopters = ad_by_period,
                         Market_Percentage = ad_by_period / m * 100)

print(adoption_df)
```

```
##    Year New_Adopters Market_Percentage
## 1  2014     431.1780         0.1551000
## 2  2015     479.8887         0.1726218
```

```
## 3  2016     533.9218         0.1920582
## 4  2017     593.8155         0.2136027
## 5  2018     660.1516         0.2374646
## 6  2019     733.5567         0.2638693
## 7  2020     814.7021         0.2930583
## 8  2021     904.3034         0.3252890
## 9  2022    1003.1177         0.3608337
## 10 2023    1111.9401         0.3999785
## 11 2024    1231.5977         0.4430208
## 12 2025    1362.9410         0.4902666
## 13 2026    1506.8323         0.5420260
## 14 2027    1664.1307         0.5986082
## 15 2028    1835.6723         0.6603138
## 16 2029    2022.2460         0.7274266
## 17 2030    2224.5638         0.8002028
## 18 2031    2443.2248         0.8788578
## 19 2032    2678.6735         0.9635516
## 20 2033    2931.1512         1.0543709
## 21 2034    3200.6419         1.1513100
## 22 2035    3486.8122         1.2542490
## 23 2036    3788.9483         1.3629310
## 24 2037    4105.8908         1.4769391
## 25 2038    4435.9710         1.5956730
## 26 2039    4776.9534         1.7183286
## 27 2040    5125.9883         1.8438807
## 28 2041    5479.5819         1.9710726
## 29 2042    5833.5893         2.0984134
## 30 2043    6183.2378         2.2241863
## 31 2044    6523.1867         2.3464700
## 32 2045    6847.6275         2.4631754
## 33 2046    7150.4289         2.5720967
## 34 2047    7425.3230         2.6709795
## 35 2048    7666.1309         2.7576010
## 36 2049    7867.0145         2.8298613
## 37 2050    8022.7439         2.8858791
## 38 2051    8128.9591         2.9240860
## 39 2052    8182.4090         2.9433126
## 40 2053    8181.1434         2.9428574
## 41 2054    8124.6436         2.9225337
## 42 2055    8013.8746         2.8826887
## 43 2056    7851.2547         2.8241923
## 44 2057    7640.5420         2.7483964
## 45 2058    7386.6480         2.6570676
## 46 2059    7095.3931         2.5522997
## 47 2060    6773.2263         2.4364123
## 48 2061    6426.9308         2.3118456
## 49 2062    6063.3377         2.1810567
## 50 2063    5689.0677         2.0464272
## 51 2064    5310.3144         1.9101850
## 52 2065    4932.6773         1.7743444
## 53 2066    4561.0492         1.6406652
## 54 2067    4199.5539         1.5106309
## 55 2068    3851.5302         1.3854425
## 56 2069    3519.5538         1.2660265
```

```
## 57 2070    3205.4877         1.1530531
## 58 2071    2910.5530         1.0469615
## 59 2072    2635.4105         0.9479894
## 60 2073    2380.2476         0.8562042
## 61 2074    2144.8641         0.7715339
```

Reference

Laricchia, F. (2022, February 14). US head-up display market by application 2014-2025. Statista. https://www.statista.com/statistics/781813/head-up-display-market-size-in-the-us-by-application/