# ISDA609 Week 1 Homework

*Daniel Dittenhafer*

*August 27, 2015*

**Page 8, Problem #10**

**Annuity**

- Interest Rate: 1%
- Withdrawl: \$1000/month
- Current Value: \$50,000

The dynamical system can be modeled using the following equation:

$$a_{n+1} = a_n + 0.01a_n - 1000$$
$$a_0 = 50000$$

The `annuityModel` function, below, defines the basic dynamical system:

```
annuityModel <- function(a_n, i, w)
{
  a_next <- a_n + (a_n * i) - w

  return (a_next)
}
```

If we run the model through some iterations, what happens?

```
# Setup variables related to the annuity
a <- 50000
rate <- 0.01
withdrawl <- 1000
# Store results in a data frame.
result <- data.frame(month=c(0), value=c(a))
# Loop through time
for(n in 1:100)
{
  a <- annuityModel(a, rate, withdrawl)

  result <- rbind(result, c(n, a))

  if(a < 0)
  {
    # End when a_n is less than zero
    break
  }
}

# Update data frame names to be user friendly.
```

```r
colnames(result) <- c("month", "value")

# show some raw data
head(result)
```

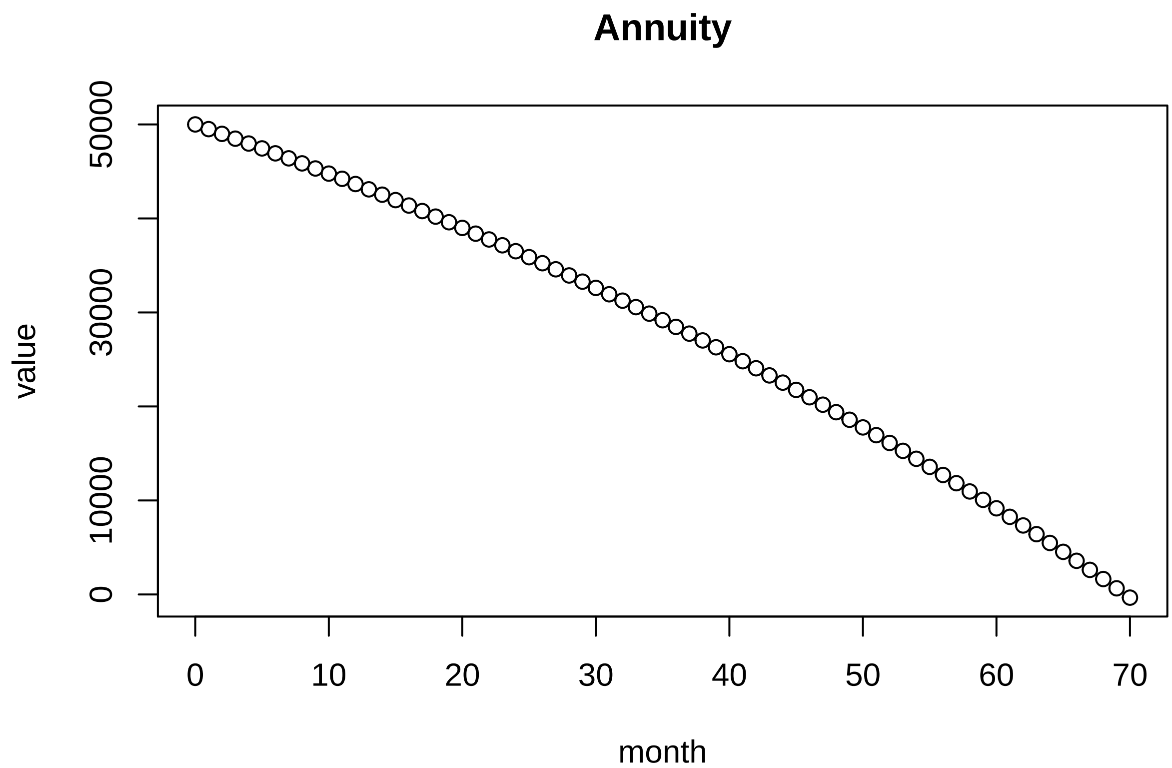
```
##   month    value
## 1     0 50000.00
## 2     1 49500.00
## 3     2 48995.00
## 4     3 48484.95
## 5     4 47969.80
## 6     5 47449.50
```

```r
tail(result)
```

```
##    month     value
## 66    65 4531.6756
## 67    66 3576.9923
## 68    67 2612.7623
## 69    68 1638.8899
## 70    69  655.2788
## 71    70 -338.1684
```

The annuity will run out of money after 70 months. When the annuity is depleted, the value of $a_n$ would be -338.1684198 if the full withdrawal were allowed. Otherwise $a_n$ will be 0.

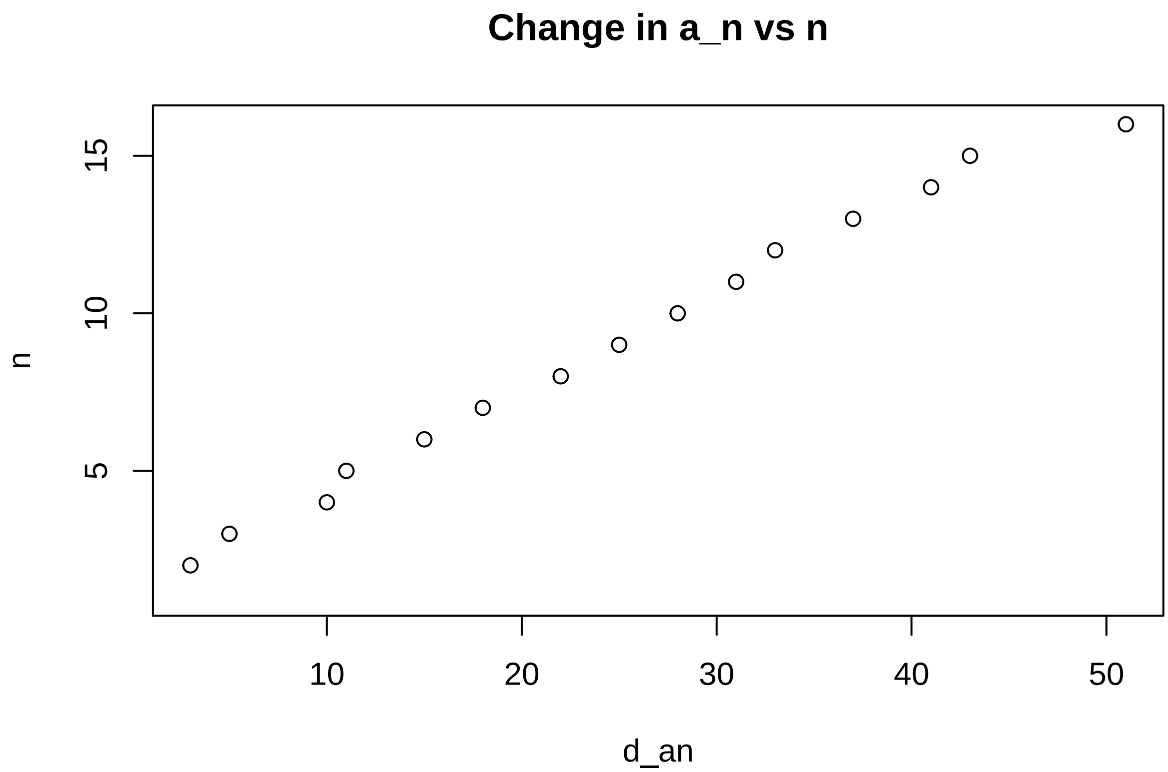
The visualization below shows the graphical representation of the dynamical system.




**Page 17, Problem #9**

```
# Setup the vectors of data
n <- 1:16
speed <- n * 5
a_n <- c(3,6,11,21,32,47,65,87,112,140,171,204,241,282,325,376)
# Compute the delta between a_n and a_n+1
d_an <- c()
d_an[0] <- NA
for(i in 1:length(a_n))
{
  d_an[i] <- a_n[i] - a_n[i - 1]
}
# Convert to data.frame
d9 <- data.frame(n, speed, a_n, d_an)
```

**(a) Calculate and plot the change $\Delta a_n$ versus $n$. Does the graph reasonably approximate a linear relationship?** The visualization below plots the change in $a_n$ vs $n$. As you can see, the graph does reasonably approximate a linear relationship.




**(b) Based on your conclusions in part (a), find a difference equation model for the stopping distance data. Test your model by plotting the errors in the predicted values against $n$. Discuss the appropriateness of the model.** First find slope of the estimated difference line:

```
delta_d_an <- max(d9$d_an, na.rm=TRUE)
delta_n <- max(d9$n)
r <-  delta_d_an / delta_n
r
```

```
## [1] 3.1875
```

Next define a function `stoppingDistanceModel` to wrap the proposed model, run the model over some period of time:

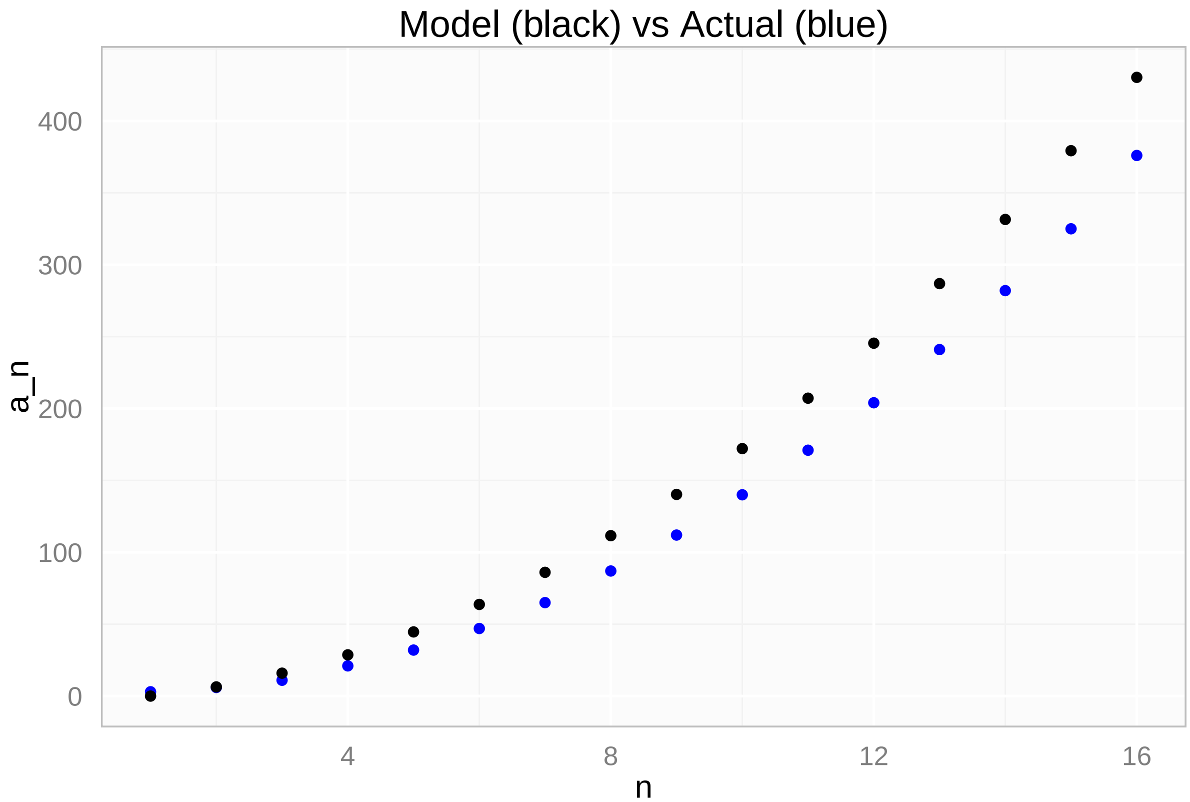
```
# Define the difference equation model
stoppingDistanceModel <- function(n, a, r)
{
    an <- r * n + a
    return(an)
}

m <- c()
m[1] <- 0
for(i in 2:length(a_n))
```

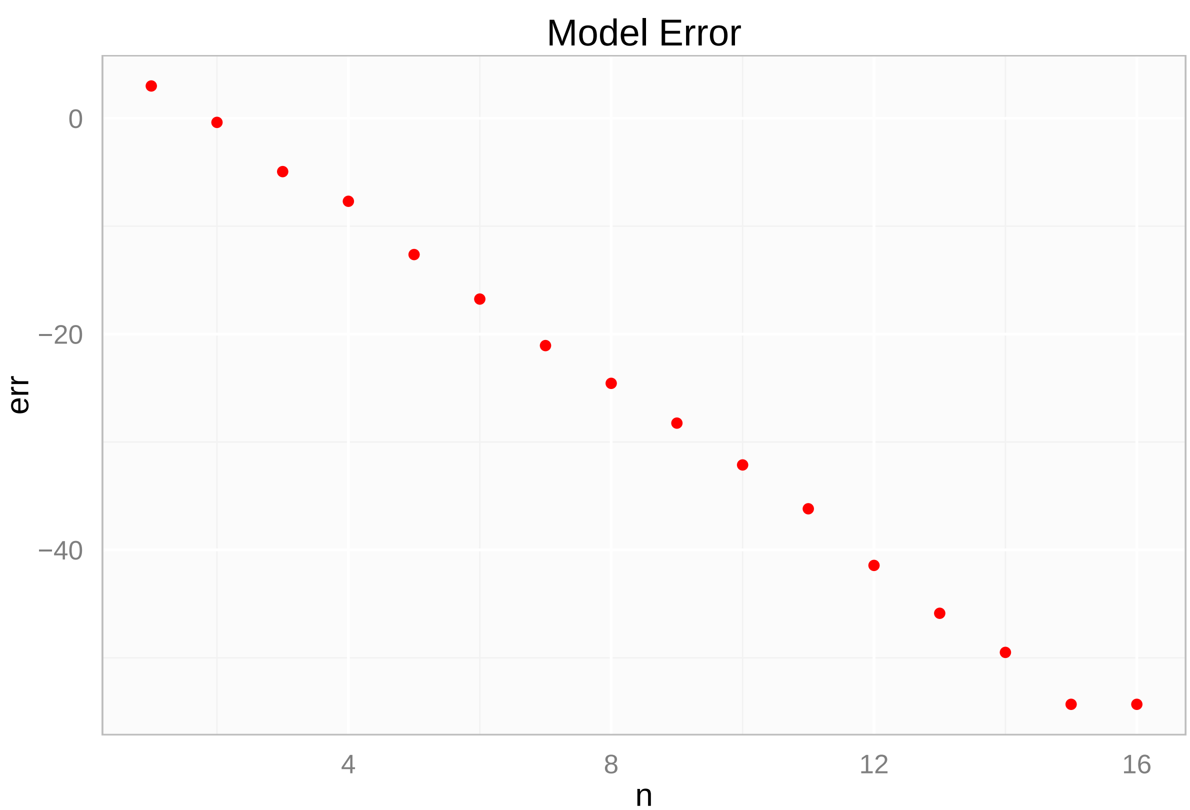
```r
{
  m[i] <- stoppingDistanceModel(i, m[i-1], r)
}

d9a <- cbind(d9, m)
d9a
```

```
##     n speed a_n d_an        m
## 1   1     5   3   NA   0.0000
## 2   2    10   6    3   6.3750
## 3   3    15  11    5  15.9375
## 4   4    20  21   10  28.6875
## 5   5    25  32   11  44.6250
## 6   6    30  47   15  63.7500
## 7   7    35  65   18  86.0625
## 8   8    40  87   22 111.5625
## 9   9    45 112   25 140.2500
## 10 10    50 140   28 172.1250
## 11 11    55 171   31 207.1875
## 12 12    60 204   33 245.4375
## 13 13    65 241   37 286.8750
## 14 14    70 282   41 331.5000
## 15 15    75 325   43 379.3125
## 16 16    80 376   51 430.3125
```



Model (black) vs Actual (blue)
400
300
200
100
0
a_n
4
8
12
16
n

Model Error
0
−20
−40
err
4
8
12
16
n

This is a very crude linear based model. The actual change in not quite linear, which contributes to the error. As larger values of $n$ are applied, the model error increases steadily. The model works as a rough estimator, but there is definitely room for improvement.

**Page 34, Problem #13**

The rumor model from the text is shown below:

$$r_{n+1} = r_n + kr + n(1000 - n)$$

We recreate this model in R code below, in the `rumorModel` function:

```
rumorModel <- function(k, rn, n)
{
  rn1 <- rn + (k * rn * (1000 - n))
  return (rn1)
}
```

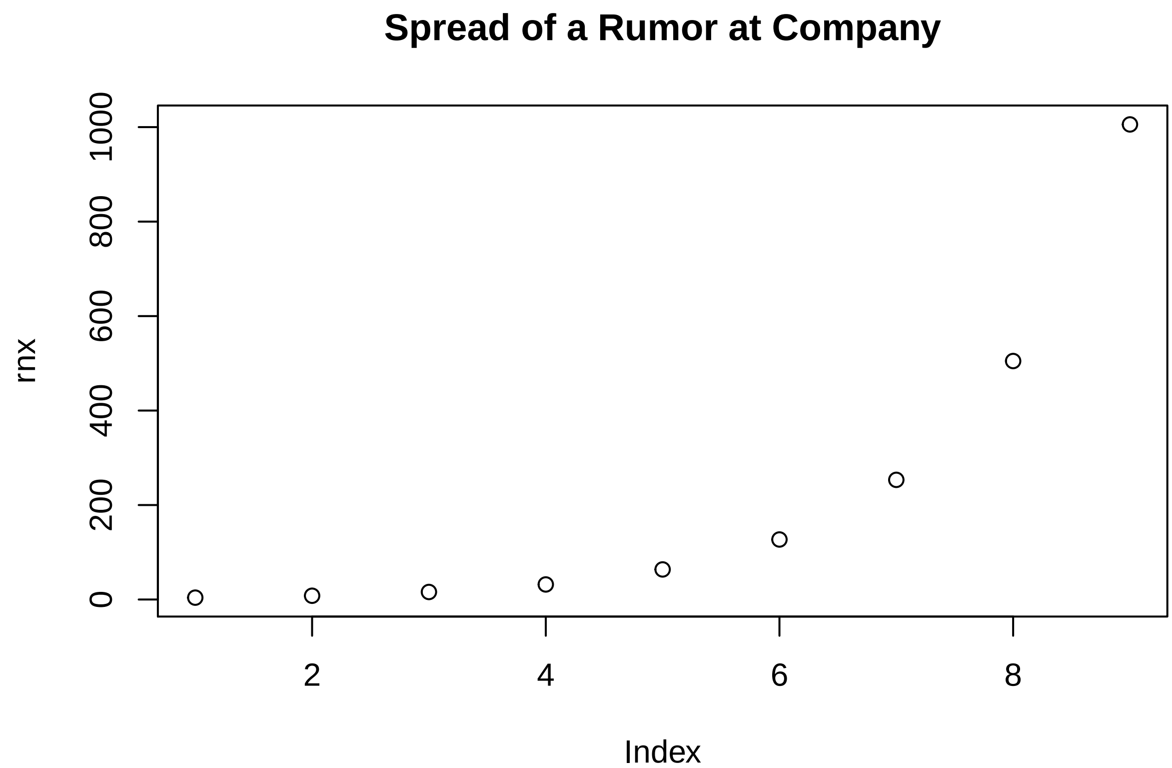
Next we define the starting assumptions of $k = 0.001$, and $r_0 = 4$, followed by a loop to iterate over the days, $n$, until all 1000 people have heard the rumor.

```
rnx <- c()
rnx[1] <- 4
k <- 0.001
for(n in 1:100)
{
  rnx[n + 1] <- rumorModel(k, rnx[n], n)
  if(rnx[n+1] > 1000)
  {
    break
  }
}
```

The rumor spread throughout the company after 9 days.

```
# Show number of people who've heard the rumor each day.
rnx
```

```
## [1]    4.00000    7.99600   15.97601   31.90409   63.68056  127.04272
## [7]  253.32318  504.87309 1005.70720
```




**Page 55, Problem #6**

First we start with the model from the text:

$$P_{n+1} = P_n - 0.1(Q_n - 500)$$

$$Q_{n+1} = Q_n + 0.2(P_n - 100)$$

To find the equilibrum values, where $X_{n+1} = X_n$, we set both $X_{n+1}$ and $X_n$ equal to $X$:

$$P = P - 0.1(Q - 500)$$

$$Q = Q + 0.2(P - 100)$$

And then solve for $Q$ and $P$, respectively:

$$0 = P - 0.1(Q - 500) - P$$

$$0 = 0.1(Q - 500)$$

$$0 = 0.1Q - 50$$

$$50 = 0.1Q$$

$$Q = 500$$

$$0 = Q + 0.2(P - 100) - Q$$

$$0 = 0.2(P - 100)$$

$$0 = 0.2P - 20$$

$$20 = 0.2P$$

$$P = 100$$

These equilibrium values, $Q = 500$ and $P = 100$, make sense as we can see they will zero out the second term in each equation and produce $X_{n+1} = X_n$ result.

**a. Does the model make sense intuitively? What is the significance of the constants 100 and 500? Explain the significance of the signs on the constants -0.1 and 0.2.** Referring back to the exercise's write-up, "increasing quantity of the product supplied tends to drive the price down". We see this in the price equation where a Q > 500 will begin reducing the price. Likewise, "a high price for the product in the market attracts more suppliers". Again, we can see this in the quantity equation where P > 100 will increase the quantity supplied. The constants 100 and 500 represent threshold values where the price and quantity respectively change their effect on the others outcome. The signs of the constants -0.1 and 0.2 are key to representing the economic behaviour of more suppliers puts downward pressure on prices, and higher prices encourages more suppliers.

**b. Test the initial conditions in the following table and predict the long-term behaviour** The table from the text is reproduced below:

| Case X | Price | Quantity |
|---|---|---|
| Case A | 100 | 500 |
| Case B | 200 | 500 |
| Case C | 100 | 600 |
| Case D | 100 | 400 |

The following `R` code contains the price and quantity equations defined in the functions `priceModel` and `quantityModel`. Additionally, a helper function `execModelLoop` is defined to help with the repeated execution of the models with the various initial conditions.

```r
priceModel <- function(pn, qn)
{
  pnx <- pn - (0.1 * (qn - 500))
  return (pnx)
}

quantityModel <- function(pn, qn)
{
  qnx <- qn + (0.2 * (pn - 100))
  return (qnx)
}

execModelLoop <- function(p0, q0, maxN, caseId)
{
  pnc <- c()
  pnc[1] <- p0

  qnc <- c()
  qnc[1] <- q0
  for(n in 1:maxN)
  {
    pnc[n + 1] <- priceModel(pnc[n], qnc[n])
    qnc[n + 1] <- quantityModel(pnc[n], qnc[n])
  }

  dfPQ <- data.frame(case=rep_len(caseId, maxN+1), n=1:(maxN+1), pnc, qnc)
  return (dfPQ)
}
```

Case A, from the table, using the equilibrium values. Let's see how that behaves.

```r
maxIterations <- 100
# First run the equilibrium values and show result.
caseA <- execModelLoop(100, 500, maxIterations, "Case A")
head(caseA)
```

```
##     case n pnc qnc
## 1 Case A 1 100 500
## 2 Case A 2 100 500
## 3 Case A 3 100 500
## 4 Case A 4 100 500
## 5 Case A 5 100 500
## 6 Case A 6 100 500
```

Case B:

```r
# Case B
caseB <- execModelLoop(200, 500, maxIterations, "Case B")
head(caseB)
```

```
##     case n    pnc     qnc
## 1 Case B 1 200.00 500.000
```

```
## 2 Case B 2 200.00 520.000
## 3 Case B 3 198.00 540.000
## 4 Case B 4 194.00 559.600
## 5 Case B 5 188.04 578.400
## 6 Case B 6 180.20 596.008
```

Case C:

```r
# Case C
caseC <- execModelLoop(100, 600, maxIterations, "Case C")
head(caseC)
```

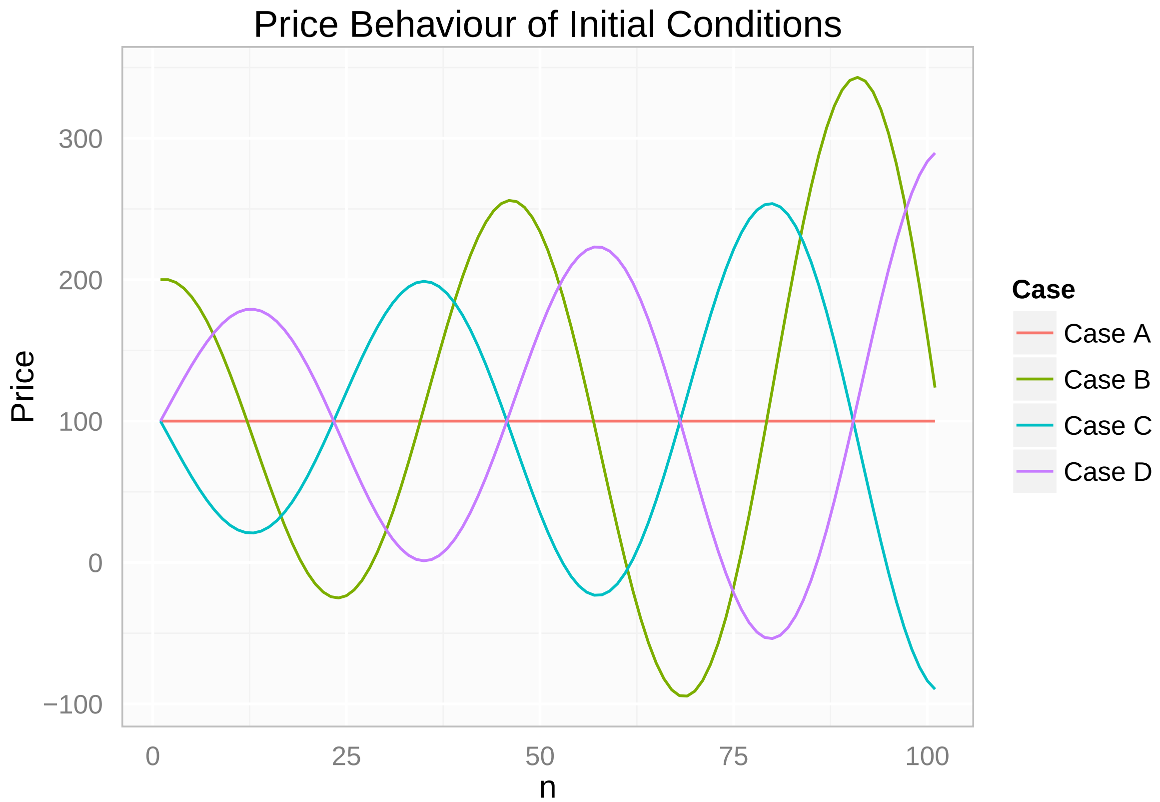
```
##     case n     pnc    qnc
## 1 Case C 1 100.000 600.00
## 2 Case C 2  90.000 600.00
## 3 Case C 3  80.000 598.00
## 4 Case C 4  70.200 594.00
## 5 Case C 5  60.800 588.04
## 6 Case C 6  51.996 580.20
```

Case D:

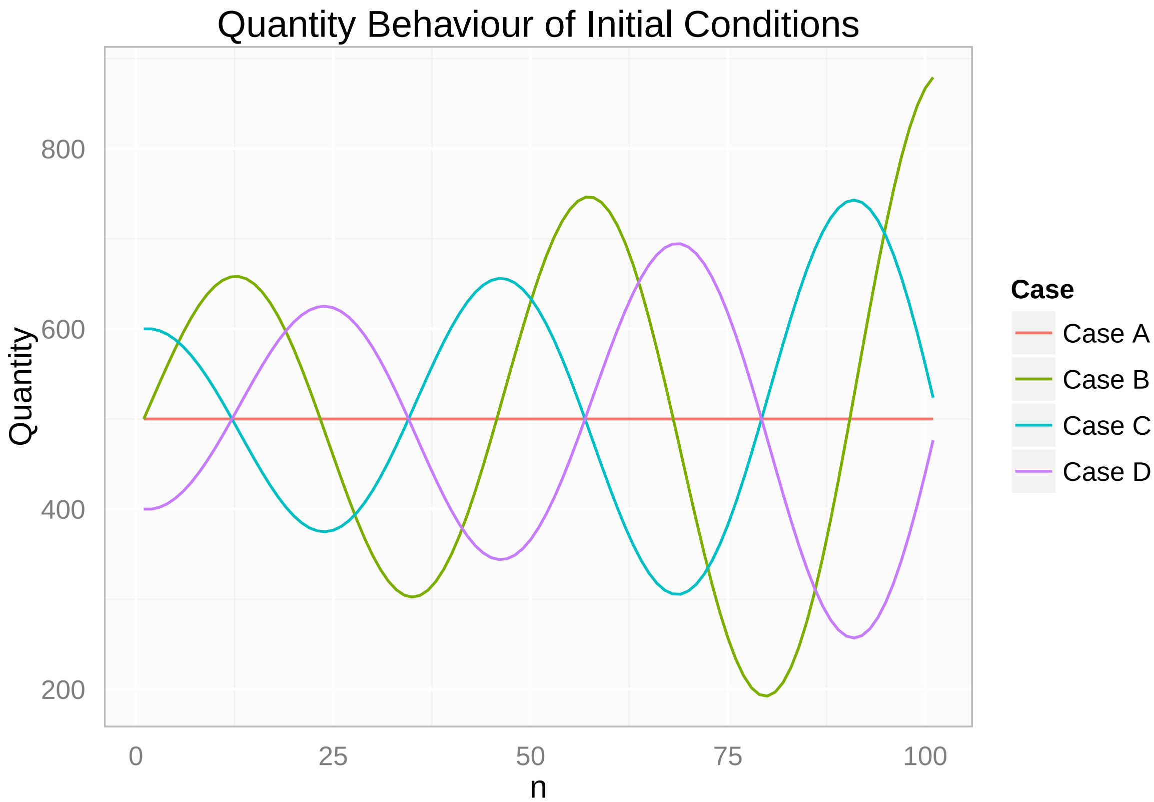
```r
# Case D
caseD <- execModelLoop(100, 400, maxIterations, "Case D")
head(caseD)
```

```
##     case n     pnc    qnc
## 1 Case D 1 100.000 400.00
## 2 Case D 2 110.000 400.00
## 3 Case D 3 120.000 402.00
## 4 Case D 4 129.800 406.00
## 5 Case D 5 139.200 411.96
## 6 Case D 6 148.004 419.80
```

How do they look graphically?



Price Behaviour of Initial Conditions
Price
n
300
200
100
0
−100
0
25
50
75
100
Case
Case A
Case B
Case C
Case D




Using the visualizations shown above as a guide, the non-equilibrium initial conditions tested will result in larger and larger oscillations across the equilibrium values, but in an unstable manner which will not converge on any equilibrium.