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)
```

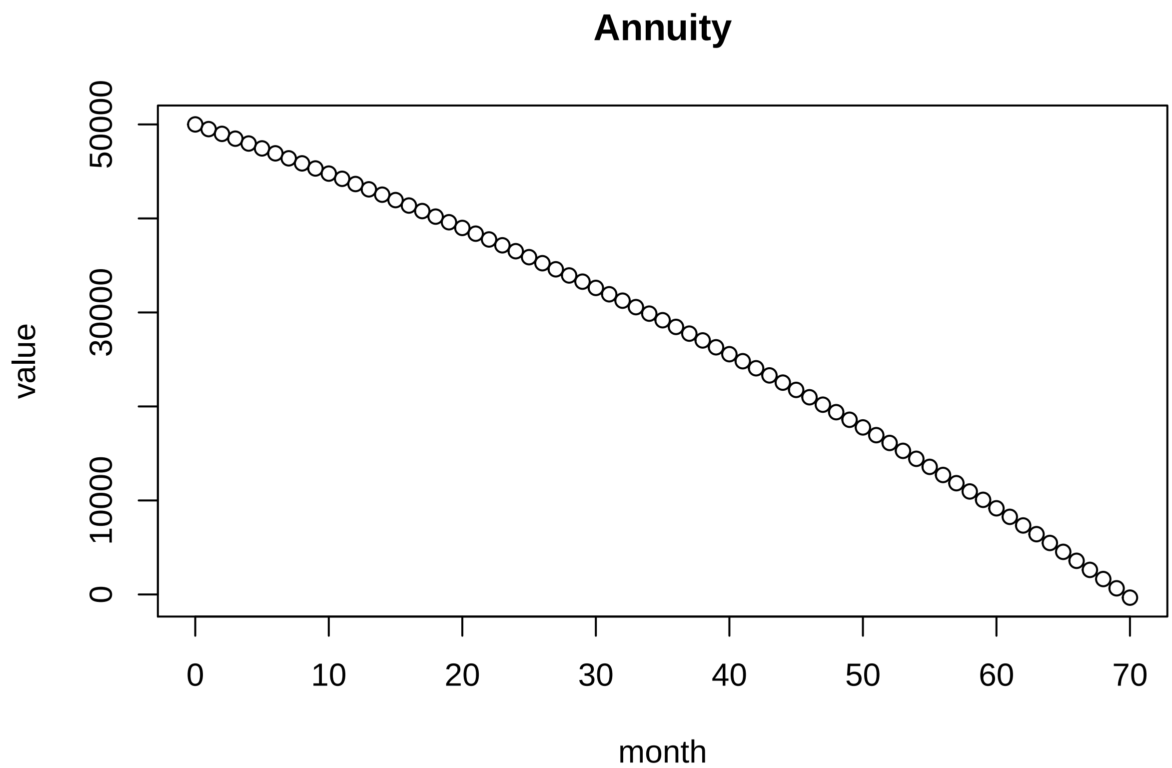
```
##   month    value
## 1     0 50000.00
## 2     1 49500.00
## 3     2 48995.00
## 4     3 48484.95
## 5     4 47969.80
## 6     5 47449.50
```

```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.

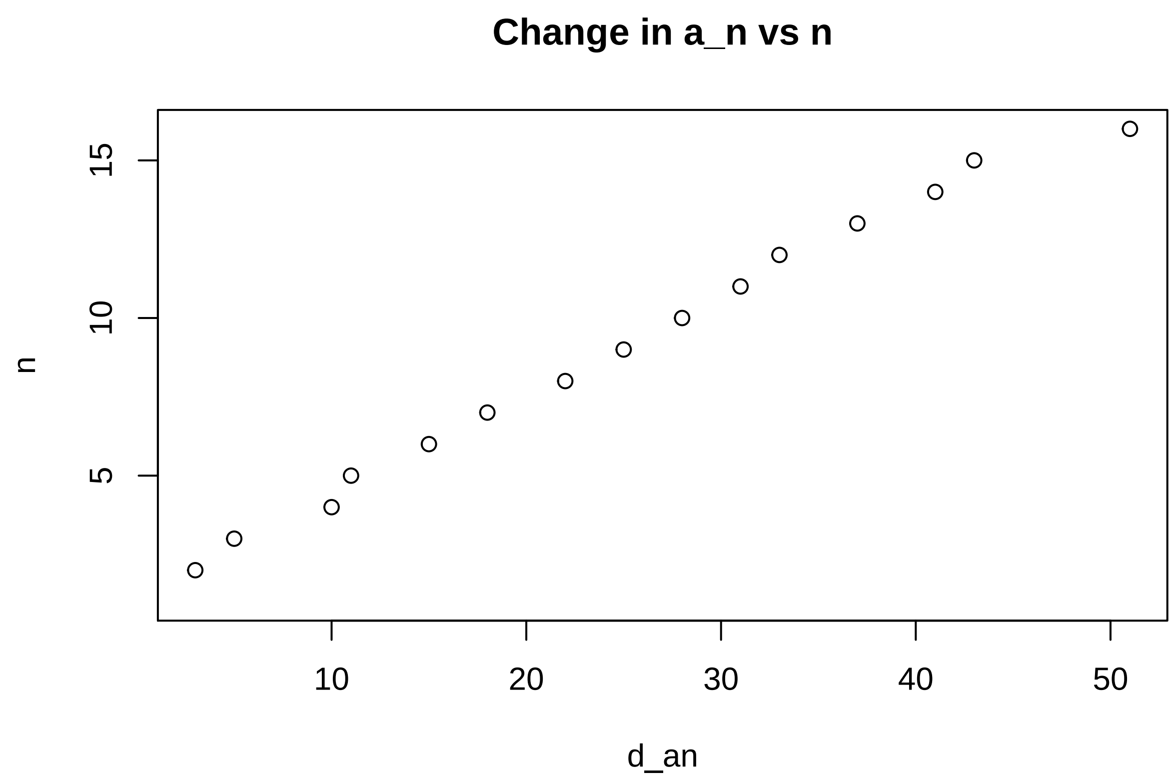
The visualization below shows the graphical representation of the dynamical system.



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 Δa_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.




```
# 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
```

(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.

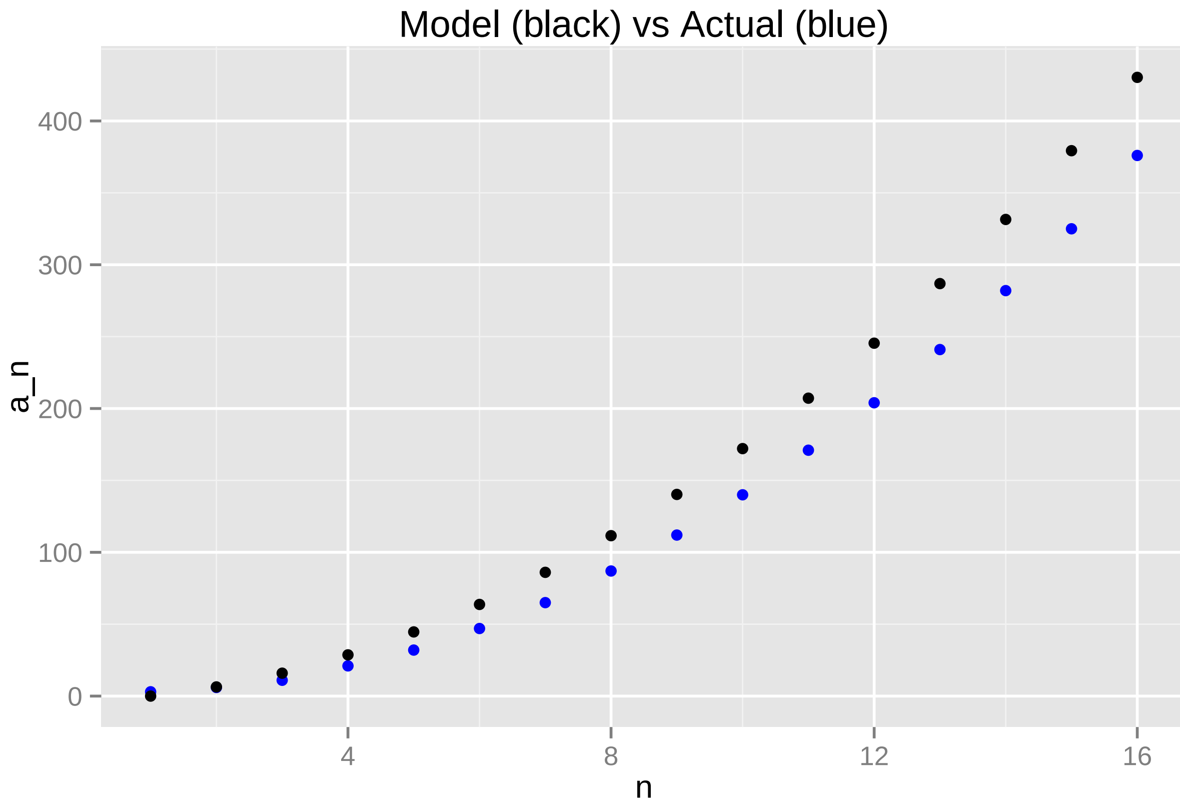
```
## [1] 3.1875
```

```
# 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
```

```r
library(ggplot2)
d9viz <- ggplot(data=d9a, aes(x=n)) +
  geom_point(color="blue", aes(y=a_n)) +
  geom_point(aes(y=m)) +
  labs(title="Model (black) vs Actual (blue)")
d9viz
```




```
# Plot the errors in predicted values against n
d9a$err <- d9a$a_n - d9a$m
d9ErrViz <- ggplot(data=d9a, aes(x=n)) +
  geom_point(color="red", aes(y=err)) +
  labs(title="Model Error")
d9ErrViz
```

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, #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 iterator over the days, n, until all 1000 people have heard the rumor.

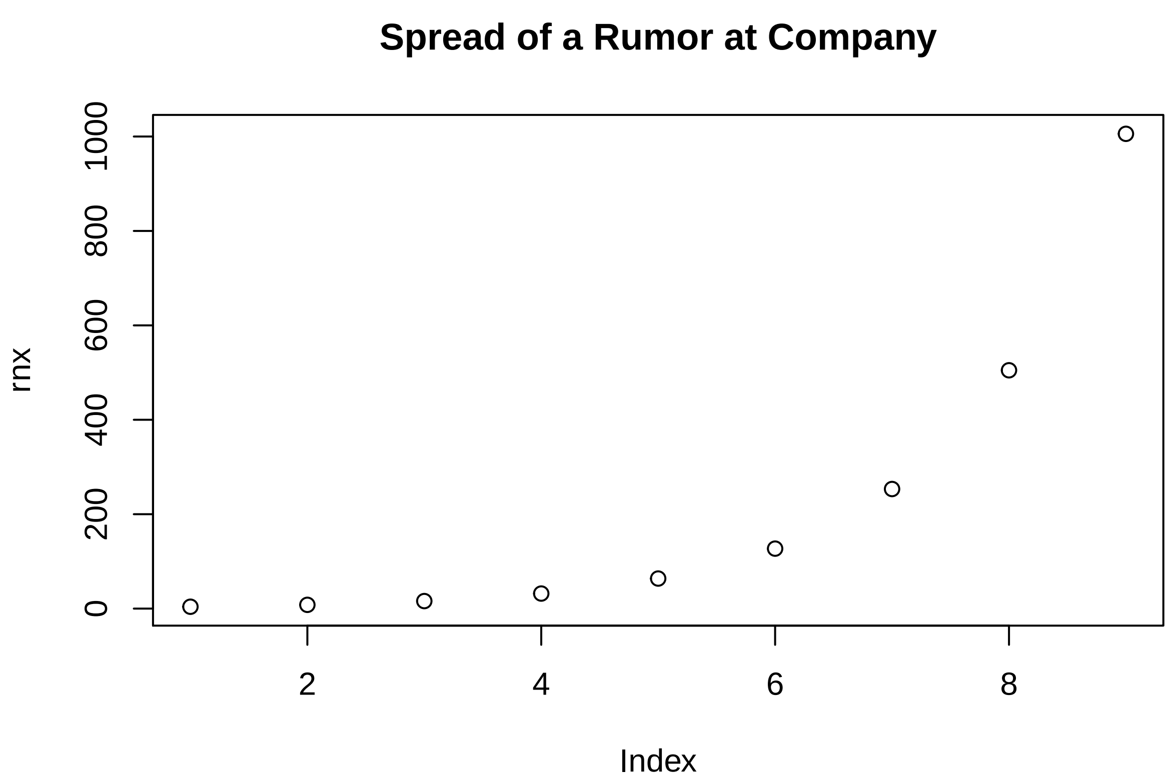
```
rnx <- c()
rnx[1] <- 4
k <- 0.001
for(n in 1:100)
```

```
{
  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, #6

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

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

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

```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)
{
  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(pnc, qnc)
  return (dfPQ)
}

execModelLoop(100, 500, 10)
```

```
##    pnc qnc
## 1  100 500
## 2  100 500
## 3  100 500
## 4  100 500
## 5  100 500
## 6  100 500
## 7  100 500
## 8  100 500
## 9  100 500
## 10 100 500
## 11 100 500
```