DATA 609: Homework 1

Modeling Change

Aaron Grzasko

August 31, 2017

```
library(ggplot2)
library(tidyr)
library(dplyr)
library(knitr)
library(grid)
library(gridExtra)
```

Page 8, Question 10

Your grandparents have an annuity. The value of the annuity increases each month by an automatic deposit of 1% interest on the previous month's balance. Your grandparents withdraw $1,000 at the beginning of each month for living expenses. Currently, they have $50,000 in the annuity. Model the annuity with a dynamical system. Will the annuity run out of money? When? *Hint*: What value will a_n have when the annuity is depleted?

We can model this problem using the a dynamical system with the following form:

$$a_{n+1} = r \times a_n \ + b$$

In this problem, $r = 1.01$ and $b = -1,000$

$$a_{n+1} = 1.01 \times a_n \ - 1,000$$

Also given: $a_0 = \$50,000$

Numerical Solution

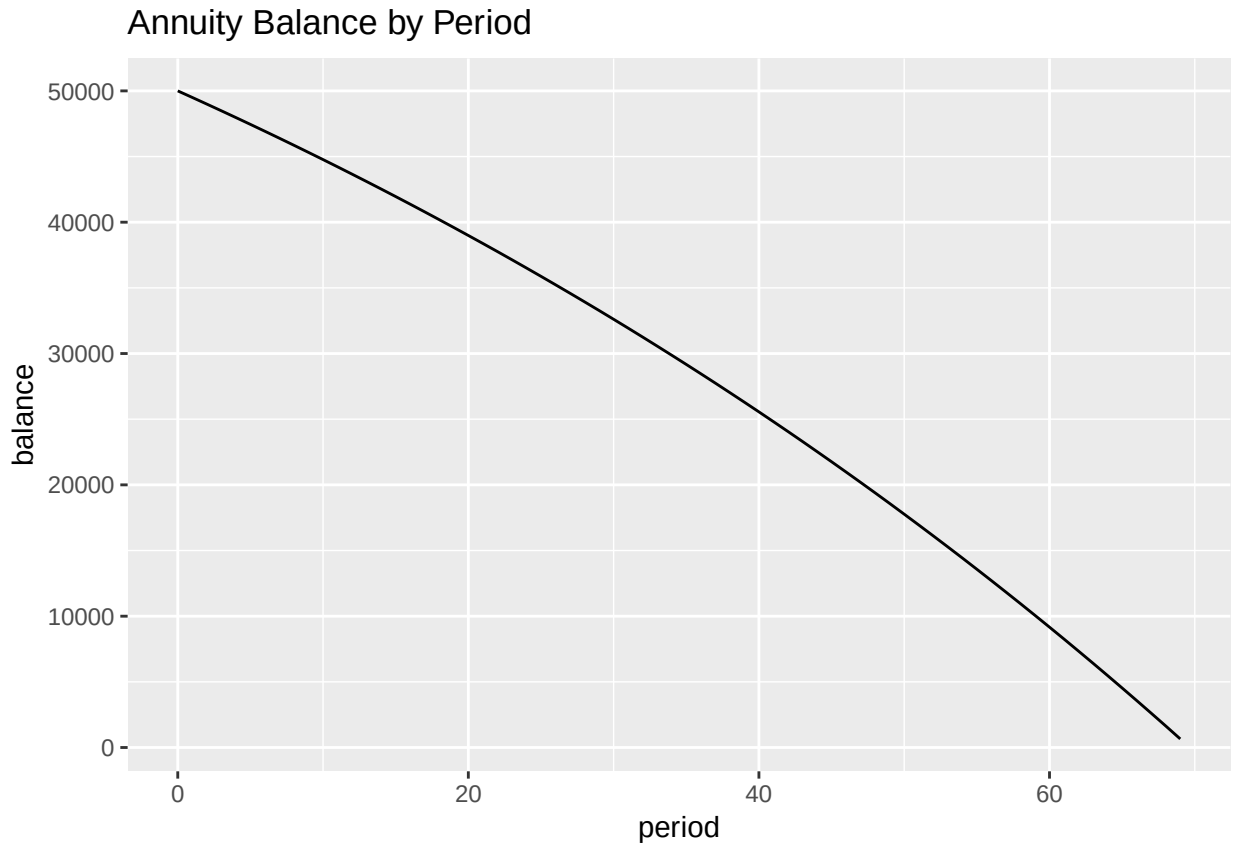
```
# initial balance and period, n
bal_calc = 50000
n <- 0

# initialize data frame to track balance by period
mydf <- data.frame(period = integer(), balance = double())


# loop through periods in dynamical model while balance >= 0
while (bal_calc >= 0) {
    mydf[n + 1, ] <- c(n, bal_calc)
    bal_calc = 1.01 * bal_calc - 1000
    n <- n + 1

    if (n > 10000)
        break  # break out of loop if balance still positive at period 10,000
}
```

Let's look at a selection of output from our model:

```
# print initial pd, then every 10th pd, until last pd with positive bal,
# print final pd
kable(mydf[c(seq(1, nrow(mydf), 10), nrow(mydf)), ], row.names = FALSE, digits = 2)
```

period	balance
0	50000.00
10	44768.89
20	38990.50
30	32607.55
40	25556.81
50	17768.41
60	9165.17
69	655.28

```
# plot balance by period
ggplot(data = mydf, aes(x = period, y = balance)) + geom_line() + ggtitle("Annuity Balance by Period")
```



The annuity still has a positive value,$655.28, after 69 months. However, there are not enough funds to withdraw 1,000 dollars in month 70. Plugging into our formula, $a_{70} = 1.01 \times a_{69} - 1000$:

```
# calculated balance after 70 months, assuming full 1,000 withdrawal
a70 <- 1.01 * mydf$balance[which(mydf$period == 69)] - 1000
a70
```

```
[1] -338.1684
```

In other words, the annuity is fully depleted after 70 months, and the maximum withdrawal–assuming no borrowing–at this time is $661.83.

Analytical Solution
The solution to a dynamical system of the form $a_{n+1} = r \times a_n + b$, where $r \neq 1$ is:

$$a_k = r^k \times c + \frac{b}{1-r}$$

```
# solve for constant c
a_0 <- 50000
r <- 1.01
b <- -1000

c <- a_0 - b/(1 - r)
c
```

```
[1] -50000
```

Plugging in the constants to equation above, we have:

$$a_k = 1.01^k \times -50,000 + 100,000$$

Now we can explicitly solve for when the balance is equal to zero:

```
# solve for k where a_k = zero
k <- log(-1e+05/-50000)/log(1.01)
k
```

```
[1] 69.66072
```

Page 17, Question 9

The data in the accompanying table show the speed n (in increments of 5 mph) of an automobile and the associated distance an in feet required to stop it once the brakes are applied. For instance, $n = 6$ (representing $6 \times 5 = 30$ mph) requires a stopping distance of $a_6 = 47$ ft.

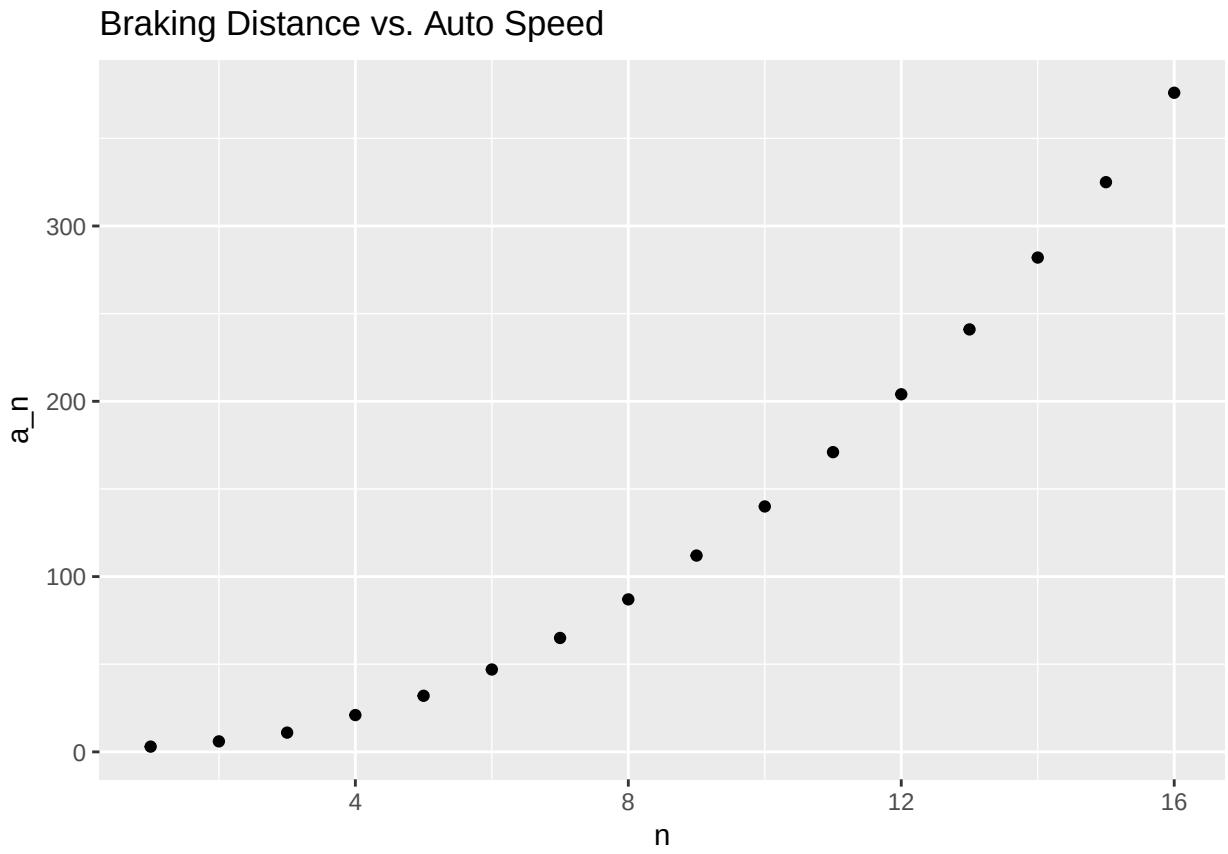
```
# reconstruct data in text; save to df
n <- (seq(1, 16))
a_n <- c(3, 6, 11, 21, 32, 47, 65, 87, 112, 140, 171, 204, 241, 282, 325, 376)
mydf <- data.frame(n, a_n)
```

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
a_n	3	6	11	21	32	47	65	87	112	140	171	204	241	282	325	376

Part A) Calculate and plot the change Δa_n versus n. Does the graph reasonably approximate a linear relationship?

First, let's examine the relationship between braking distance and speed:

```
# graph of a_n vs n
ggplot(mydf, aes(x = n, y = a_n)) + geom_point() + ggtitle("Braking Distance vs. Auto Speed")
```



The relationship between these variables appears to be quadratic.

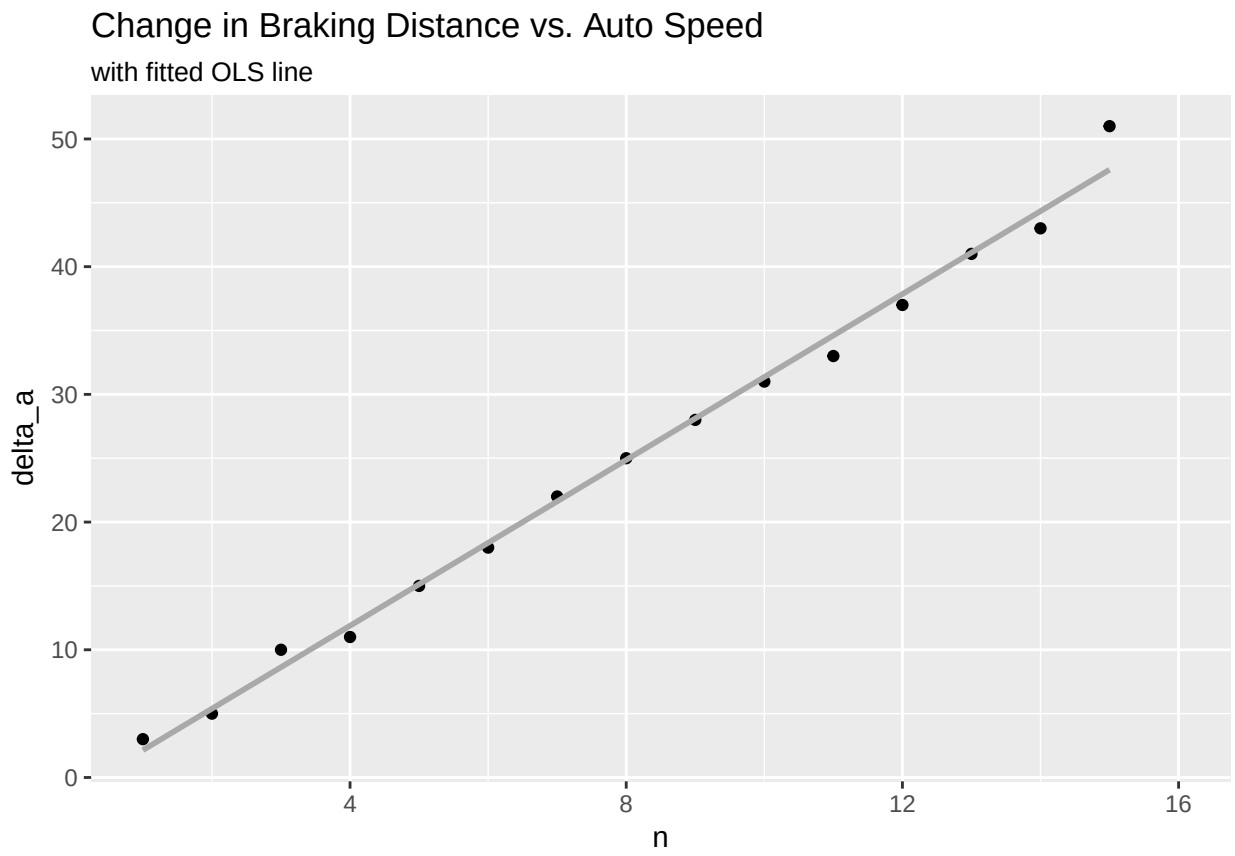
Now, let's examine the relationship between the change in braking distance and speed:

```
# add new field to capture change in braking distance
mydf <- mydf %>% mutate(delta_a = lead(a_n) - a_n)
```

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
a_n	3	6	11	21	32	47	65	87	112	140	171	204	241	282	325	376
delta_a	3	5	10	11	15	18	22	25	28	31	33	37	41	43	51	NA

In the chart below, we fit an OLS line to the observed data. This linear model appears to provide a fairly accurate approximation, particularly for moderate driving speeds.

```
# plot of delta a_n vs n, along with fitted ols line
g <- ggplot(mydf, aes(x = n, y = delta_a))
g <- g + geom_point() + ggtitle("Change in Braking Distance vs. Auto Speed")
g <- g + labs(subtitle = "with fitted OLS line")
g + geom_smooth(method = "lm", color = "darkgray", se = FALSE)
```



Part 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.

Below are the coefficient and intercept from the OLS model:

```
mylm <- lm(delta_a ~ n, data = mydf)

mylm$coefficients
```

```
(Intercept)           n
  -1.104762    3.246429
```

Our difference equation is therefore:

$$\Delta a_n = 3.246n - 1.105$$

Or, equivalently:

$$a_{n+1} = a_n + 3.246n - 1.105$$

Now we will use this difference equation model to predict braking distances for speed increments 2 through 16.
Note: we used the sample observation at increment 1 as the initial value condition in our model.

```
# function to predict period (n+1) braking distance, a(n+1) inputs: speed
# increment, n; braking distance, a_n; Beta coeff from OLS; alpha intercept
# from OLS
```

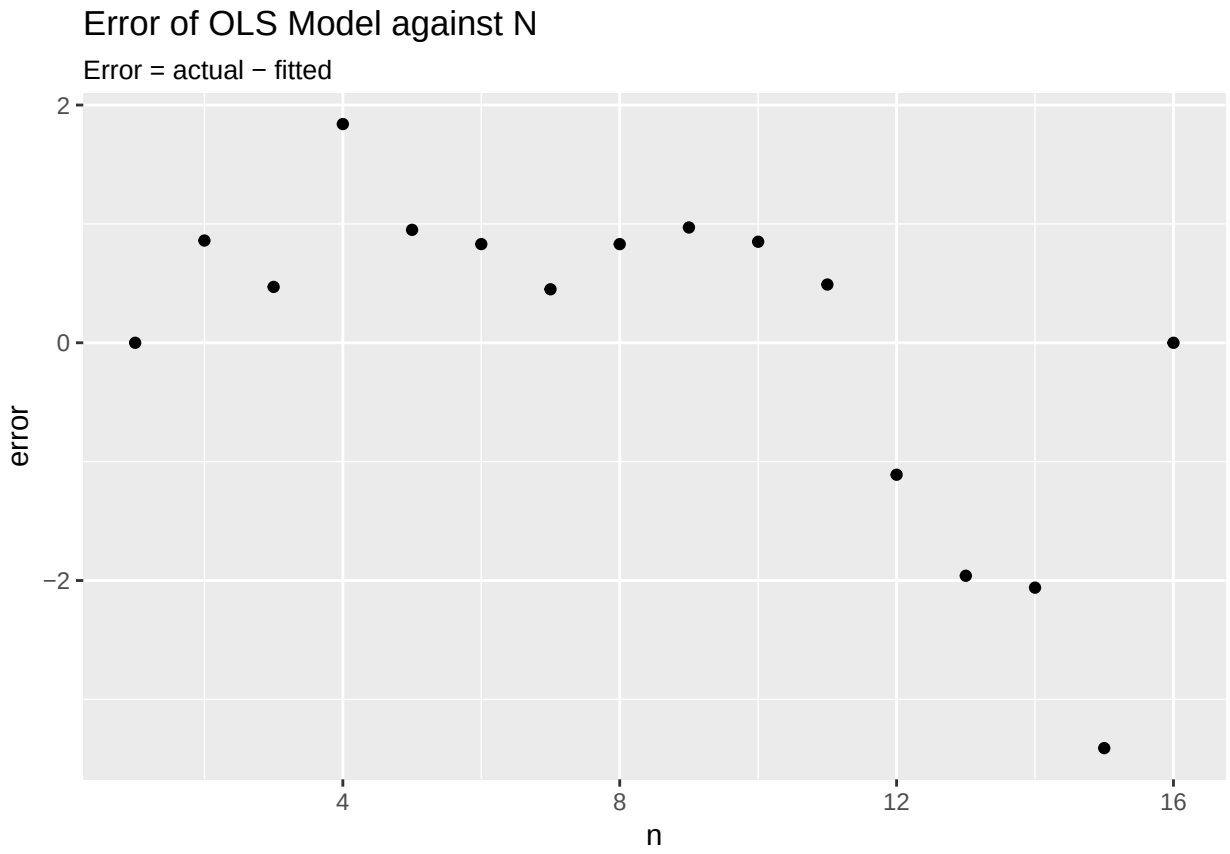
```
pred_brake_dist <- function(n, a_n, slope, intercept) {
    a_n + slope * n + intercept
}

# set initial speed and braking distance for model predictions using sample
# observation at n=1 as initial value condition
n <- 1
a_n <- 3

# initialize prediction vector for observation at n = 1
pred_vector <- 3

# make additional predctions for n=2:16
for (i in seq(2, 16)) {
    pred_vector[i] <- pred_brake_dist(n, a_n, mylm$coefficients[2], mylm$coefficients[1])
    n <- n + 1
    a_n <- pred_vector[i]
}

# append predictions to master data frame
mydf["pred_a_n"] <- round(pred_vector, 2)

# calculate error: observed - predicted
mydf$error <- mydf$a_n - mydf$pred_a_n
```

Let's look at our model predictions and errors:

n	a_n	pred_a_n	error
1	3	3.00	0.00
2	6	5.14	0.86
3	11	10.53	0.47
4	21	19.16	1.84
5	32	31.05	0.95
6	47	46.17	0.83
7	65	64.55	0.45
8	87	86.17	0.83
9	112	111.03	0.97
10	140	139.15	0.85
11	171	170.51	0.49
12	204	205.11	-1.11
13	241	242.96	-1.96
14	282	284.06	-2.06
15	325	328.41	-3.41
16	376	376.00	0.00

Below is a plot of errors vs. n:

```
e <- ggplot(mydf, aes(x = n, y = error)) + geom_point()
e <- e + ggtitle("Error of OLS Model against N")
e + labs(subtitle = "Error = actual - fitted")
```



In the plot above we note that the error terms do not appear to be entirely random. Instead, they appear to exhibit serial correlation. We must be careful when interpreting the OLS estimates of the standard errors, as these estimates are likely understated. We should consider performing formal tests for serial correlation (e.g. Durbin Watson), and then take corrective action if necessary. The model may also be suffering from heteroskedasticity, although its hard to verify visually given the small sample size.

Page 34, Question 13

Consider the spreading of a rumor through a company of 1,000 employees, all working in the same building. We assume that the spreading of a rumor is similar to the spreading of a contagious disease in that the number of people hearing the rumor each day is proportional to the product of the number who have heard the rumor previously and the number who have not heard the rumor. This is given by $r_{n+1} = r_n + kr_n(1000 - r_n)$ where k is a parameter that depends on how fast the rumor spreads and n is the number of days. Assume $k = 0.001$ and further assume that four people initially have heard the rumor. How soon will all 1,000 employees have heard the rumor?

Given:

$r_0 = 4$
$k = 0.001$
$r_{n+1} = r_n + kr_n(1000 - r_n)$

For the purpose of this problem, we assume that fractional persons in the model output are relevant. We interpret a fractional person as someone who has heard only a portion of the full rumor.

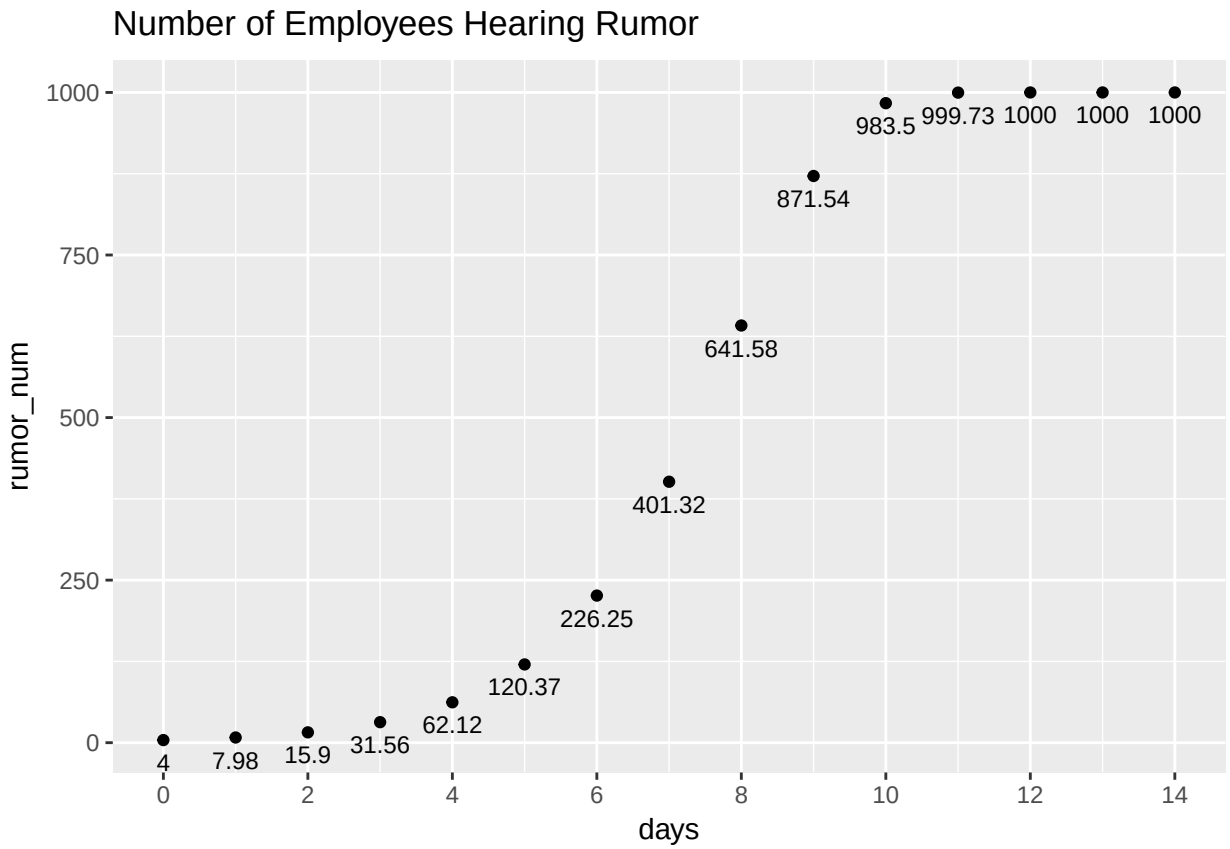
Let's solve this problem numerically, iterating through our difference equation until the model output reaches $r_{n+1} = 1,000$:

```
# function to calculate number infected inputs: number infected prior pd,
# r_n; constant k; n number of days
calc_rumor_num <- function(r_n, k, n) {
    r_n + k * r_n * (1000 - r_n)
}

# starting values for r and n
r <- 4
n <- 0
k <- 0.001

# initialize data frame to track balance by period, populate with starting
# values
mydf <- data.frame(days = n, rumor_num = r)

# loop through periods in dynamical model until entire workforce (i.e. 1000
# EEs) hear rumor
while (r < 1000) {
    r <- calc_rumor_num(r, k, n)
    mydf[n + 2, ] <- c(n + 1, r)
    n <- n + 1

    if (n > 10000)
        break  # break out of loop if rumor not fully circulated after 10,000 days
}
```

Below is the output from our model:

days	rumor_num
0	4.000
1	7.984
2	15.904
3	31.556
4	62.115
5	120.372
6	226.255
7	401.319
8	641.581
9	871.536
10	983.497
11	999.728
12	1000.000
13	1000.000
14	1000.000

```
h <- ggplot(mydf, aes(days, rumor_num)) + geom_point()
h <- h + ggtitle("Number of Employees Hearing Rumor")
h <- h + geom_text(aes(label = round(rumor_num, 2)), vjust = 1.8, size = 3)
h <- h + scale_x_continuous(breaks = seq(0, 14, 2))
h
```



The model reaches 1,000 rumors circulated at 14 days. In the table and plot above, this is difficult to see, as the displayed output rounds to 1,000 after only 12 days.

Page 55, 6

An economist is interested in the variation of the price of a single product. It is observed that a high price for the product in the market attracts more suppliers. However, increasing the quantity of the product supplied tends to drive the price down. Over time, there is an interaction between price and supply. The economist has proposed the following model, where P_n represents the price of the product at year n, and Q_n represents the quantity. Find the equilibrium values for this system.

$$P_{n+1} = P_n - 0.1(Q_n - 500)$$
$$Q_{n+1} = Q_n + 0.2(P_n - 100)$$

To find the equilibrium solution to this equation, we set $P = P_{n+1} = P_n$ and $Q = Q_{n+1} = Q_n$, then rearrange the 2 x 2 linear system as follows:

$$\begin{bmatrix} 0 & -0.1 \\ 0.2 & 0 \end{bmatrix} \begin{bmatrix} P \\ Q \end{bmatrix} = \begin{bmatrix} -50 \\ 20 \end{bmatrix}$$

```
# transition matrix
A <- matrix(c(0, -0.1, 0.2, 0), 2, 2, byrow = TRUE)

# inverse matrix
```

```
A_inv <- solve(A)

# b in Ax = b
b <- c(-50, 20)

# solve for P and Q
A_inv %*% b
```

```
     [,1]
[1,]  100
[2,]  500
```

Part A) Does the model make sense intuitively? What is the significance of the constants 100 and 500? Explain the significance of the signs of the constants -0.1 and 0.2.

The model has an intuitive interpretation.

Price and quantity interact as follows: a high (low) quantity has a negative (positive) impact on price; a high (low) price has a positive (negative) influence on quantity.

The constant 100 in the second equation means that whenever the price is above $100, the quantity supplied will increase in the subsequent period. When the price is below $100, the quantity will decrease in the subsequent period.

Similarly, the constant 500 in the first equation means that when the quantity supplied is greater than 500, the price will increase in the next period. On the hand, when the quantity is less than 500, the price will decrease in the subsequent period.

The multiplicative constant -0.1 in the first equation refers to the sensitivity of price to changes in quantity. Our model assumes that price decreases by $0.10 for every additional unit supplied.

The 0.2 constant in the second equation is used to describe the sensitivity of quantity to changes in price. In this model, the quantity supplied increases by 0.2 units for ever $1 increase to price.

Part b) Test the initial conditions in the following table and predict the long-term behavior.

```
case <- c("A", "B", "C", "D")
price <- c(100, 200, 100, 100)
quantity <- c(500, 500, 600, 400)
mydf <- data.frame(case = case, price = price, quantity = quantity)
kable(mydf)
```

case	price	quantity
A	100	500
B	200	500
C	100	600
D	100	400

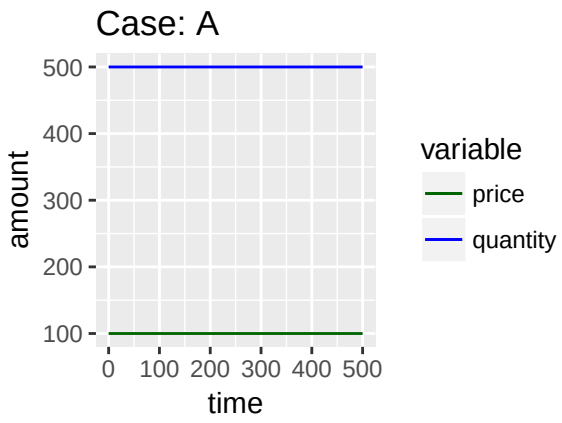
Based on our explanation in part A, we expect the following each case to evolve as follows:

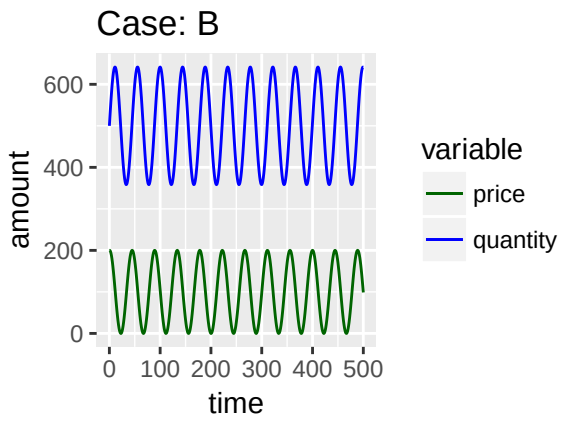
- Case A: No change, as both price and quantity are at their equilibrium values of $100 and 500, respectively.
- Case B: Price is above its equilibrium value; so quantity will increase. Price will then grow more slowly and eventually decrease in response to the quantity increase. Quantity will grow more slowly as price decreases, and eventually decrease. When quantity falls below 500, price will begin to rebound. This results in a never-ending tug-of-war between price and quantity. In summary, both price and quantity exhibit oscillating behavior.

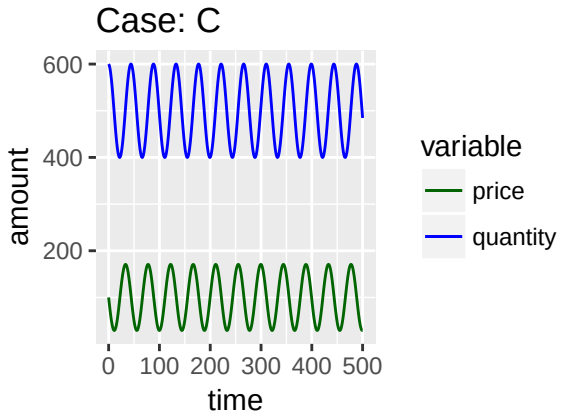
- Case C: Quantity is above its equilibrium value; so price will react negatively. Quantity grows more slowly and then falls as as price dips below equilibrium value. But price eventually rebounds as quantity falls below 500. Like Case B, this scenario involves oscillatory behavior among both variables.

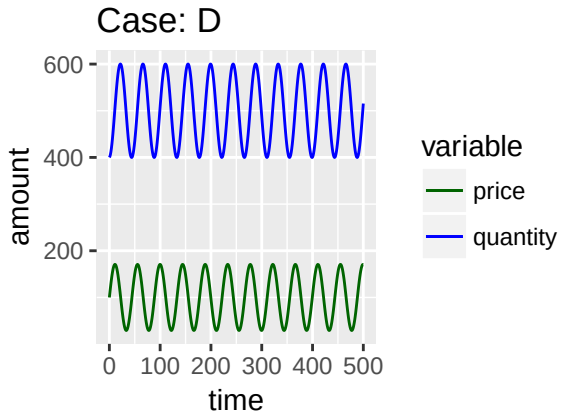
- Case D: Quantity is below below equilibrium value; so price increases. The price increase spawns additional increases in quantity until quantity exceeds equilibrium. At this point, prices decrease, which results to a slowing increase, and eventually, a decrease in quantity. Once again, both variables exhibit oscillatory behavior.

```
# price model
price_calc <- function(p, q) p - 0.1 * (q - 500)

# quantity model
quantity_calc <- function(p, q) q + 0.2 * (p - 100)


# loop through all cases
for (i in seq(1, 4)) {
    p <- price[i]
    q <- quantity[i]
    model_df <- data.frame(time = 0, price = p, quantity = q)


    # loop through multiple time periods
    for (pd in seq(1, 500)) {
        p <- price_calc(p, q)
        q <- quantity_calc(p, q)
        model_df[pd + 1, ] <- c(pd, p, q)
    }

    # prepare plot
    g <- ggplot(data = model_df)
    g <- g + geom_line(aes(x = time, y = quantity, color = "quantity"))
    g <- g + geom_line(aes(x = time, y = price, color = "price"))
    g <- g + ylab("amount")
    g <- g + scale_colour_manual("variable", values = c(quantity = "blue", price = "darkgreen"))
    g <- g + ggtitle(paste("Case:", case[i]))

    assign(paste0("Case", i), g)

}

# print plots in grid
blank <- rectGrob(gp = gpar(col = "white"))
grid.arrange(Case1, blank, Case2, Case3, blank, Case4, ncol = 3, nrow = 2, widths = c(5,
    0.5, 5))
```


Case: A
500
400
300
200
100
amount
0
100
200
300
400
500
time
variable
price
quantity


Case: B
600
400
200
0
amount
0
100
200
300
400
500
time
variable
price
quantity


Case: C
600
400
200
amount
0
100
200
300
400
500
time
variable
price
quantity


Case: D
600
400
200
amount
0
100
200
300
400
500
time
variable
price
quantity