Midterm Math 438

Fall 2023

13-Oct-2023

Contents

IMPORTANT: In problem 2 please set a random seed so that your results are reproducible.

SUPER IMPORTANT: Your exam is due by 5:00 p.m. Friday the 13th as an Rmd file uploaded on Canvas.

Problem 1 (25 pts.)

Ol' Doc Ridenhour is droning on again about the wonders of math in an 8:30 a.m. Calc 3 class. You, and most of your classmates, barely got any sleep last night; you look over your shoulder and see your friend has already succumbed to the sandman, and others nearby are close as well. From a biology class you took, you remember the average length of a nap is 15 minutes; once awake again, people are unlikely to fall asleep for many hours. You also recall that sleep (like yawns) spreads through a crowd as a diffusion process and depends **proportionally** on the square root-concentration of people that are asleep.

1. Write down a system of **difference** equations that describes the dynamics of students sleeping in Calc 3.

   One *possible* answer would be something like:

   $$\Delta A = -\rho A\sqrt{\frac{Z}{70}}$$
   $$\Delta Z = \rho A\sqrt{\frac{Z}{70}} - \frac{Z}{15}$$

   where ρ is the constant of proportionality, A is the number of awake students who have not previously fallen asleep, and Z is the number of students that are asleep.

2. Based on your equations, under what condition does the number of sleeping students increase during the period?

This depends on the system written down in (1). For the above example, we solve $\Delta Z > 0$ for ρ which yields $\rho = \frac{\sqrt{70Z}}{15A}$, which at $t = 0$ gives $\rho > \frac{\sqrt{70}}{1035} \approx 0.0081$.

3. Using one minute time steps, and assuming that one student is asleep at the very begginning of class (Billy), iterate the system of equations for a 50 minute class of 70 students. Assume the proportionality constant to be 0.02.

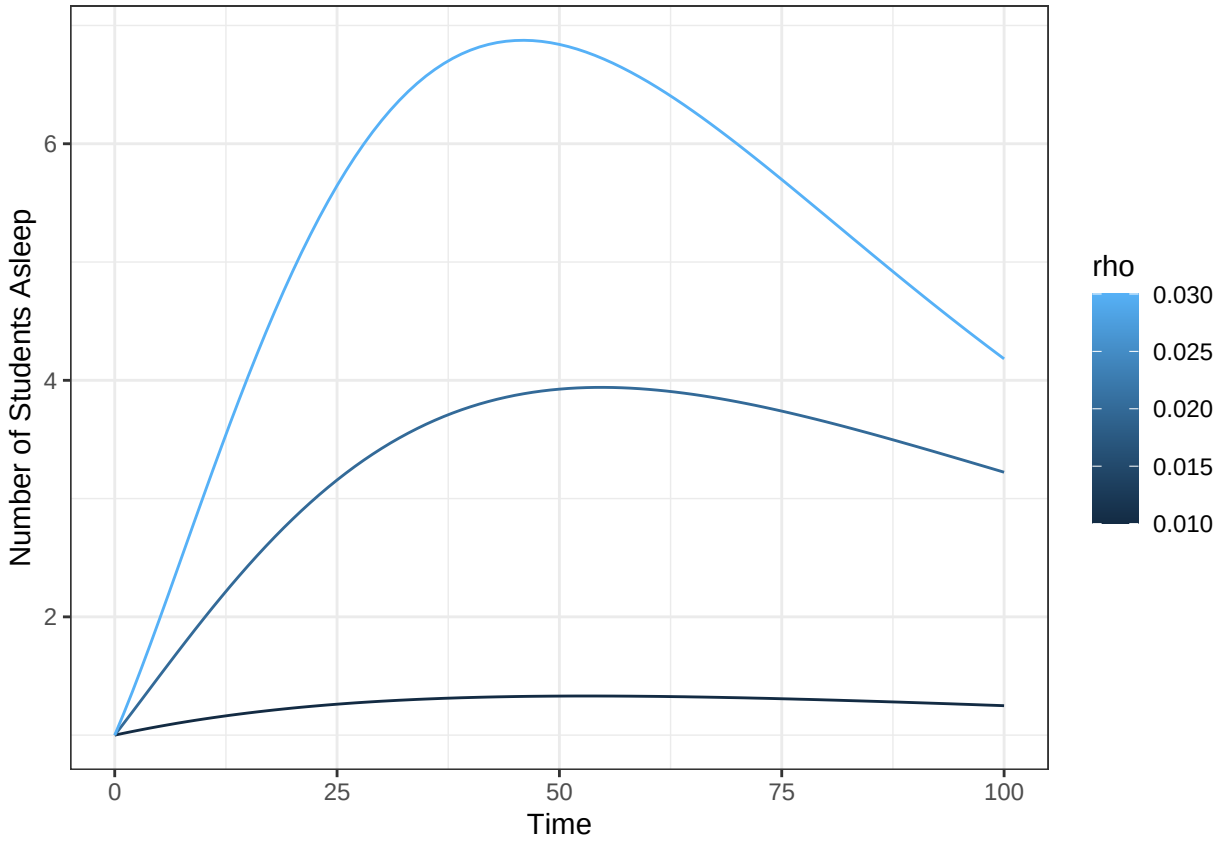
- Plot the resulting data for both awake and asleep students.
- Vary the constant of proportionality some (pick 2 more values); iterate and plot the results.

```
rho <- 0.02
A <- 69
Z <- 1

data <- data.frame(Time=0, A = A, Z = Z)

for(i in 1:100){
  DeltaA <- -rho*A*(Z/70)^0.5
  DeltaZ <- rho*A*(Z/70)^0.5 - Z/15
  A <- A + DeltaA
  Z <- Z + DeltaZ
  data <- rbind(data, data.frame(Time = i, "A" = A, "Z" = Z))
}

data$rho <- 0.02

rho <- 0.03 #my second choice
A <- 69
Z <- 1

data2 <- data.frame(Time=0, A = A, Z = Z)

for(i in 1:100){
  DeltaA <- -rho*A*(Z/70)^0.5
  DeltaZ <- rho*A*(Z/70)^0.5 - Z/15
  A <- A + DeltaA
  Z <- Z + DeltaZ
  data2 <- rbind(data2, data.frame(Time = i, "A" = A, "Z" = Z))
}

data2$rho <- 0.03

rho <- 0.01 #my third choice
A <- 69
Z <- 1

data3 <- data.frame(Time=0, A = A, Z = Z)

for(i in 1:100){
  DeltaA <- -rho*A*(Z/70)^0.5
  DeltaZ <- rho*A*(Z/70)^0.5 - Z/15
  A <- A + DeltaA
  Z <- Z + DeltaZ
  data3 <- rbind(data3, data.frame(Time = i, "A" = A, "Z" = Z))
}
```

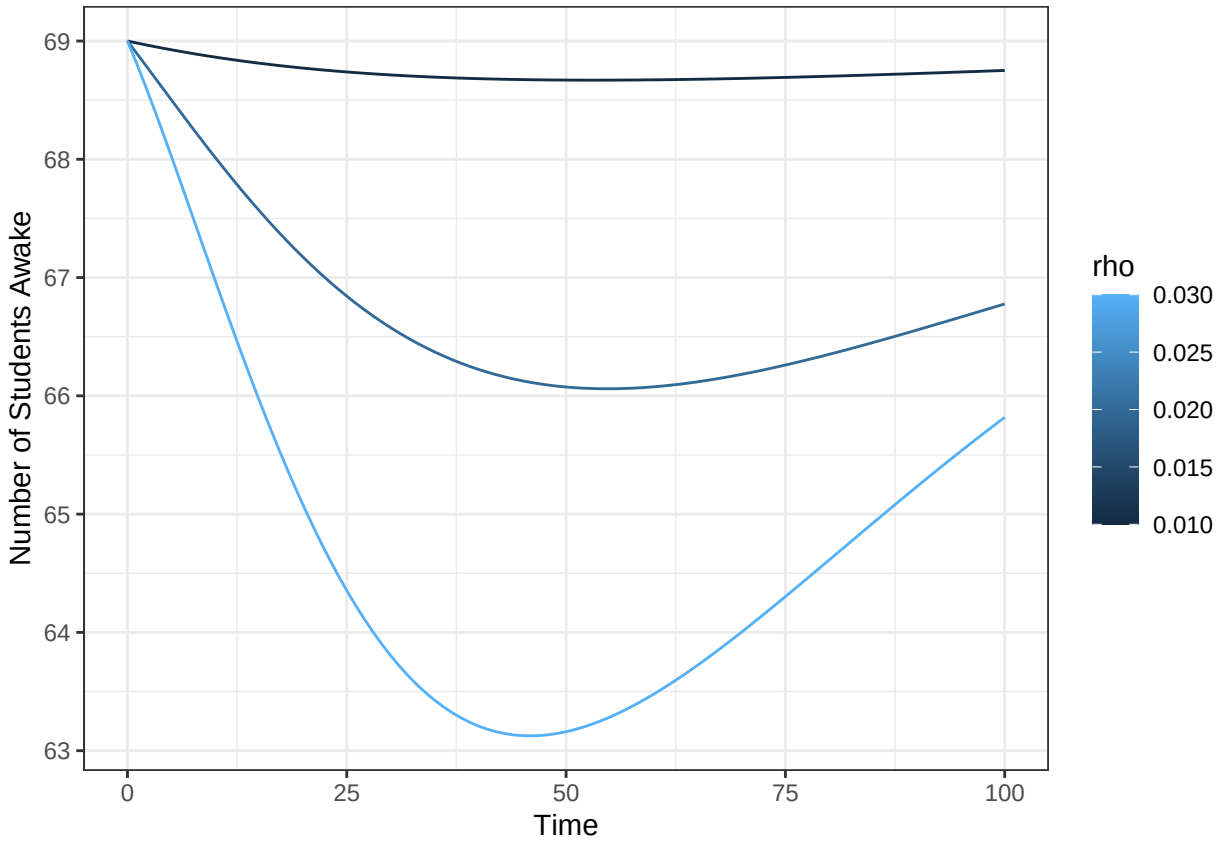
```r
data3$rho <- 0.01

Data <- rbind(data, rbind(data2,data3))

library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.1.2
```

```r
g <- ggplot(Data,aes(x=Time, y = Z, col = rho, group = rho))

g + geom_line() + theme_bw() + ylab("Number of Students Asleep")
```




```r
g <- ggplot(Data,aes(x=Time, y = 70-Z, col = rho, group = rho))

g + geom_line() + theme_bw() + ylab("Number of Students Awake")
```



4. According to your model (with the originally assumed proportionality constant), what is the probability that you fell asleep during the lecture?

   We can easily find the proportion of students who had fallen asleep during the class by calculating $(70 - A_{50})/70$, which for the original $\rho = 0.02$ and the model above yields $P_{sleeper} = 0.1932$.

5. How many students would you expect to still be asleep at the end of the lecture?

   This is simply checking what Z_{50} is for the given model; here it is $Z_{50} = 3.9254$.

6. When does the maximum number of students that are asleep occur? (*Hint:* Make sure you check times longer than 50 minutes if your number of sleeping students is still increasing at 50 minutes.)

   Now we look in our data frame for the time at which $\max(Z)$ occurs. For the example model, $t_{\max} = 55$ which you can visually confirm by looking at the plot.

7. You decide that it is necessary to fit a model to actual data. While diligently staying awake, you observe when each new individual falls asleep and the total number asleep at that time. For your model, fit the proportionality constant by using optimization based on the *median absolute deviation*, the Brent algorithm, and the following data:

Minutes	Asleep
5	2
7	2
10	3
12	4
14	4
16	5
18	5
20	6

Minutes	Asleep
21	6
23	6
25	6
27	7
28	7
30	7
32	7
33	7
35	8
37	8
39	8
41	8
42	8
44	8
46	8
48	8

To accomplish, recall that we need to write down a function that applies whatever criteria we wish to modeled data ($\hat{y}$) and the actual data (y). We also need an objective function that will take parameters and calculate $\hat{y}$ and return the value of the criteria. Once we have those two functions we can then run `optim()`.

```
medianAD <- function(y, yhat) median(abs(y - yhat)) #calculate the median absolute deviation per instru

objectiveFunction <- function(x, obs, dist = "medianAD") {

  ##model the values
  rho <- x
  A <- 69
  Z <- 1

  sim <- data.frame(Time=0, A = A, Z = Z)

  for(i in 1:100){
    DeltaA <- -rho*A*(Z/70)^0.5
    DeltaZ <- rho*A*(Z/70)^0.5 - Z/15
    A <- A + DeltaA
    Z <- Z + DeltaZ
    sim <- rbind(sim, data.frame(Time = i, "A" = A, "Z" = Z))
  }

  eval(call(dist, obs$Asleep, sim$Z[sim$Time %in% obs$Minutes]))
}

fit1 <- optim(0.02, objectiveFunction, obs = sleepers, method = "Brent", lower = 0, upper = 0.1)
fit1$par #best value of rho
```

```
## [1] 0.034243
```

```
fit1$convergence #check convergence
```

```
## [1] 0
```

8. Repeat the optimization two more times but using first the Chebyshev criteria and second a log-likelihood criteria based on a Poisson distribution.

To accomplish this we simply define the two new criteria and pass them to the objective function/`optim()`. Remember however that for likelihood functions, we *maximize* the criterion value.

```
chebyshev <- function(y, yhat) max(abs(y - yhat))
llpois <- function(y, yhat) sum(dpois(y, yhat, log = T))

fit2 <- optim(0.02, objectiveFunction, obs = sleepers, dist = "chebyshev", method = "Brent", lower = 0,
fit3 <- optim(0.02, objectiveFunction, obs = sleepers, dist = "llpois", method = "Brent", lower = 0, up

c("rho"=fit2$par,"conv"=fit2$convergence)
```

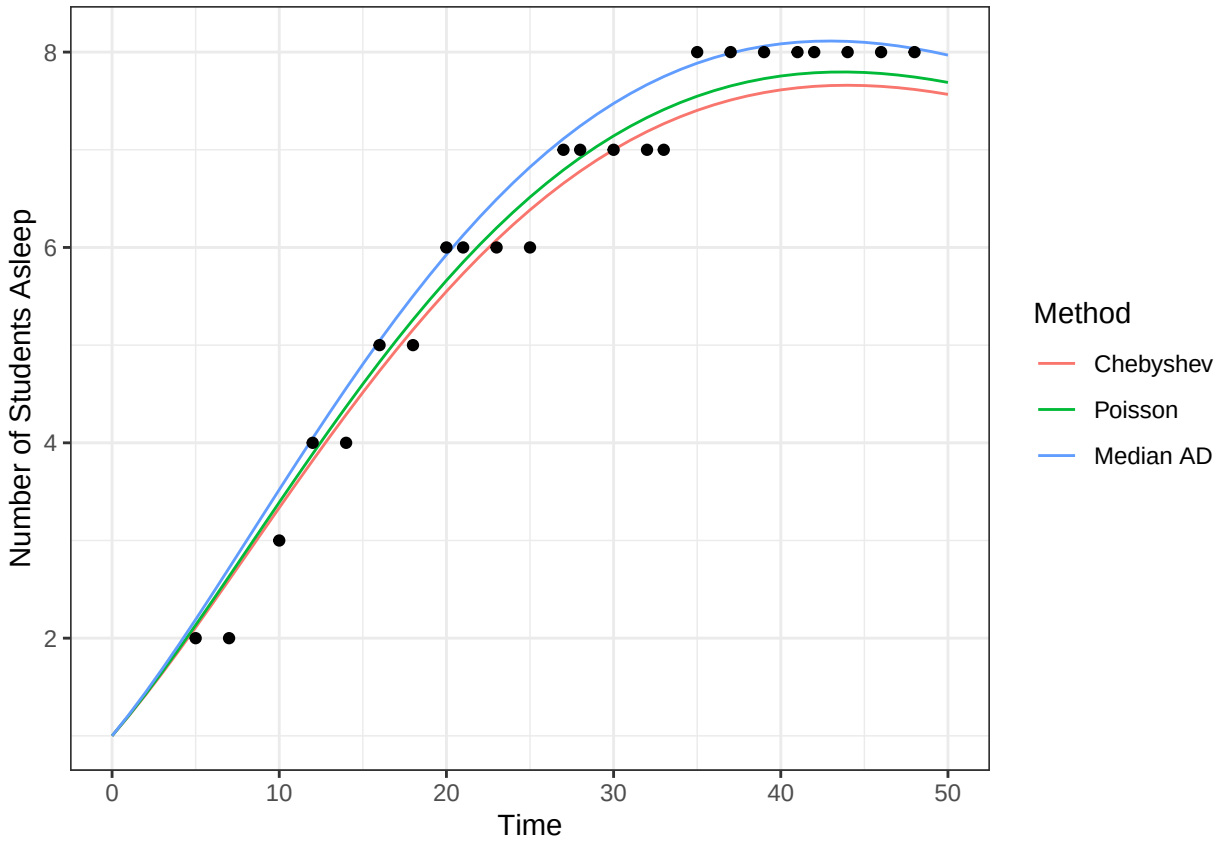
```
##        rho       conv
## 0.03268364 0.00000000
```

```
c("rho"=fit3$par,"conv"=fit3$convergence)
```

```
##      rho     conv
## 0.033151 0.000000
```

9. Plot the fitted models in 7 & 8 versus the observed data.

```
FitData <- data.frame(Time = 0, A = 0, Z = 0, rho = 0)
FitData <- FitData[0,]

for(rho in c(fit1$par,fit2$par,fit3$par)){
  A <- 69
  Z <- 1

  fitdata <- data.frame(Time=0, A = A, Z = Z)

  for(i in 1:50){
    DeltaA <- -rho*A*(Z/70)^0.5
    DeltaZ <- rho*A*(Z/70)^0.5 - Z/15
    A <- A + DeltaA
    Z <- Z + DeltaZ
    fitdata <- rbind(fitdata, data.frame(Time = i, "A" = A, "Z" = Z))
  }

  fitdata$rho <- rho

  FitData <- rbind(FitData,fitdata)
}

g <- ggplot(FitData,aes(x=Time, y = Z, col = as.factor(rho), group = rho))

g + geom_line() + theme_bw() + ylab("Number of Students Asleep") + geom_point(data=sleepers, aes(x = Mi
```



Problem 2 (25 pts.)

Figure 1: Battle

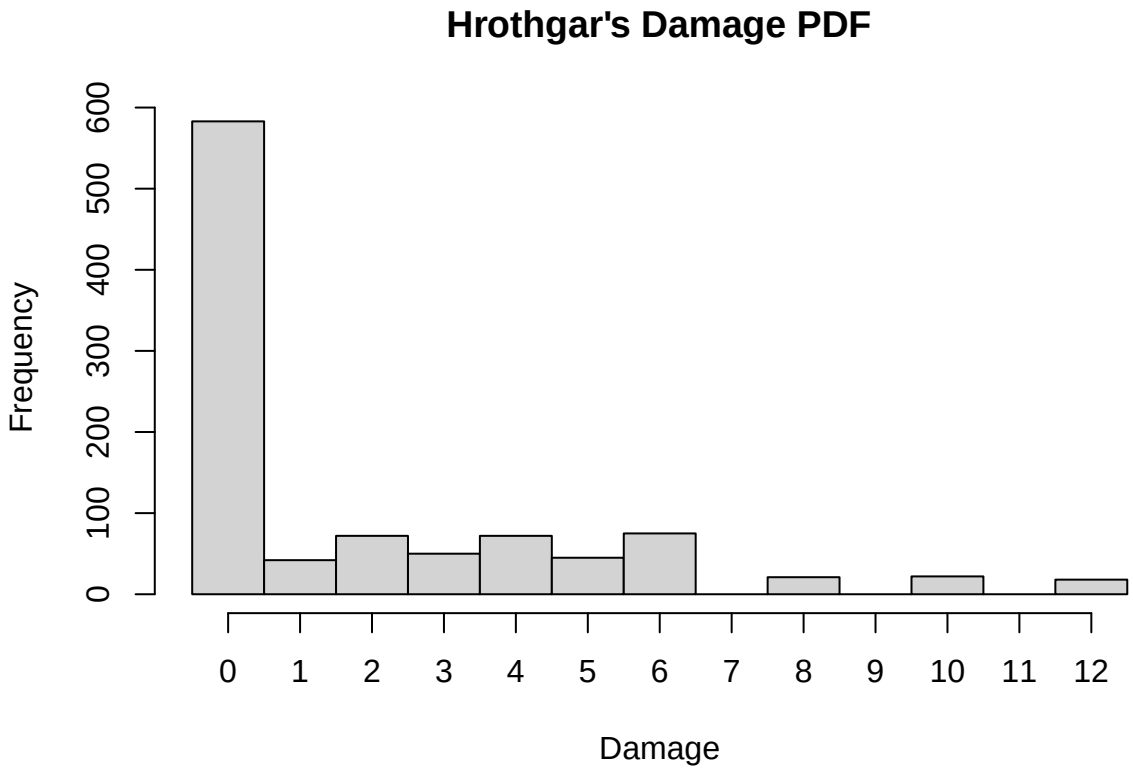
Hrothgar, a level 6 half-dwarf warrior, has wandered into the lair of the evil drow elf Grimweard, a level 8 mage. A battle to the death ensues! Fortunately, **our hero got an A in MATH 438** and can figure out what his chances of surviving the harrowing encounter are. Hrothgar is attacking Grimweard with a rather sad club that does 1d6 damage, and, given that he is rather inexperienced, he must roll >11 on 1d20 to actually hit Grimweard; should he roll >17, he gets lucky and gets double the damage. He also has 25 hit points. In comparison, his foe is casting a magic missile spell that does 2d4 damage and requires >6 on 1d10 to hit our hero. Grimweard is weaker and only has 15 hit points. Assume that who attacks first is chosen at random. Based on 1000 simulated battles between Hrothgar and Grimweard, answer the following:

1. Using an empirical cumulative distribution function (eCDF), create a random number generator that takes a single argument (n), that returns n random amounts of damage done by Hrothgar. Do this for

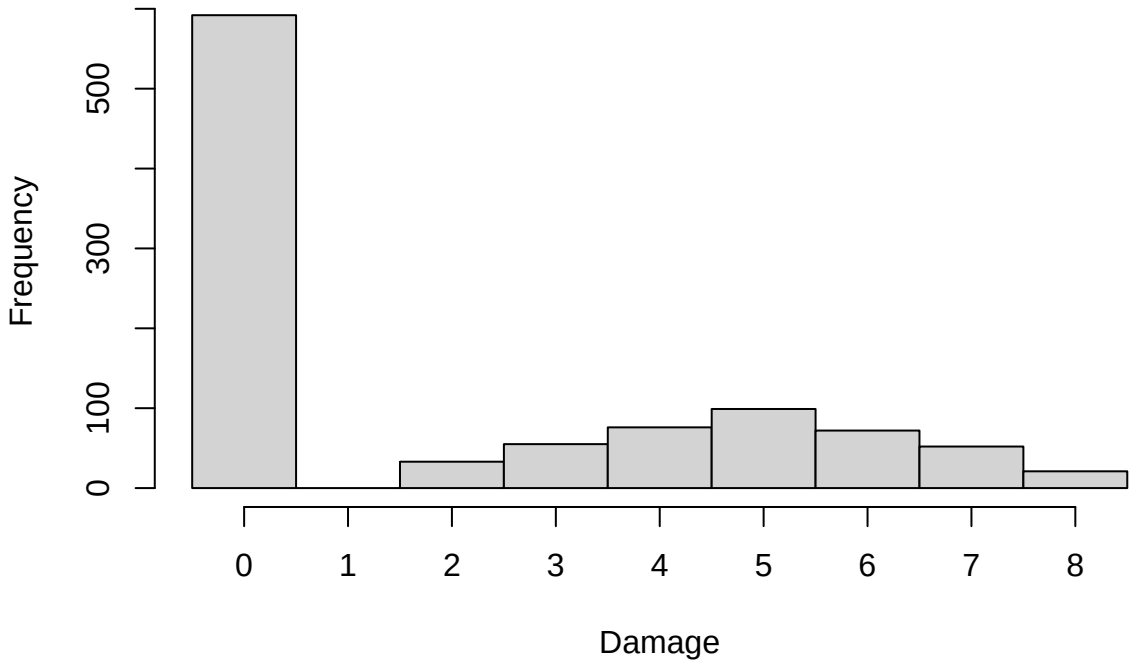
Grimweard as well. Make 2 histograms using these RNGs by setting $n = 1000$.

To do this, I will create functions that just randomly generate attacks and their damage. Using those, we can simulate a PDF for both Hrothgar and Grimweard. The simulated PDFs are then converted to empirical CDFs that we can directly sample from.

```
set.seed(132412)

hrothgar.attack <- function(){
  hit.die <- sample(1:20,1)
  dmg <- sample(1:6,1)
  if(hit.die > 17) return(dmg*2)
  if(hit.die > 11) return(dmg)
  0
}

grimweard.attack <- function(){
  hit.die <- sample(1:10,1)
  dmg <- sum(sample(1:4,2, replace = T))
  if(hit.die > 6) return(dmg)
  0
}

CDF.hrothgar <- ecdf(replicate(10000,hrothgar.attack()))
rdmg.hrothgar <- function(n=1) unname(quantile(CDF.hrothgar, runif(n)))

CDF.grimweard <- ecdf(replicate(10000,grimweard.attack()))
rdmg.grimweard <- function(n=1) unname(quantile(CDF.grimweard, runif(n)))

h1 <- hist(rdmg.hrothgar(1000),main = "Hrothgar's Damage PDF", xlab = "Damage", breaks = 0:13,right=F, 
axis(1, at = h1$mids, labels = 0:12)
```




```
h2 <- hist(rdmg.grimweard(1000),main = "Grimweard's Damage PDF", xlab = "Damage", breaks = 0:9, right=F
axis(1, at = h2$mids, labels = 0:8)
```



2. What percentage of the time does Hrothgar defeat Grimweard?

   Now we actually need to simulate battles between our two contestants. To do this, you have to write

a small simulation program and determine who kills whom first following the rules outlined in the problem. Here is how I accomplished this:

```
battle <- function(){
  hrothgar <- 25
  grimweard <- 15

  #generate random damage and then look at the cumulative sum
  cumdmg.hrothgar <- cumsum(rdmg.hrothgar(100))
  cumdmg.grimweard <- cumsum(rdmg.grimweard(100))

  #figure out how many turns it took for the cumulative sum to exceed the respective HP values
  nToWin.hrothgar <- sum(cumdmg.hrothgar < grimweard) + 1
  nToWin.grimweard <- sum(cumdmg.grimweard < hrothgar) + 1

  #check the number of turns and determine who won; if turn count is equal, pick someone randomly
  if(nToWin.grimweard == nToWin.hrothgar){ return(data.frame("Winner" = sample(c("Hrothgar", "Grimweard
  if(nToWin.grimweard < nToWin.hrothgar) return(data.frame("Winner" = "Grimweard", Attacks = nToWin.grim
  return(data.frame("Winner" = "Hrothgar", Attacks = nToWin.hrothgar))
}

#FIGHT!!!!
results <- battle()
for(i in 1:999) results <- rbind(results, battle())

table(results$Winner)/1000
```

```
##
## Grimweard  Hrothgar
##     0.211     0.789
```

Our hero Hrothgar is dominating in this fight! He is winning an astounding 21.1% of the simulations.

3. How many turns, on average (mean and median), does it take Hrothgar to defeat Grimweard? What is the 95% CI?

   To answer this, we simply get the appropriate summary statistics from the results.

```
mean(results$Attacks[results$Winner == "Hrothgar"])
```

```
## [1] 7.114068
```

```
quantile(results$Attacks[results$Winner == "Hrothgar"], c(0.025,0.5,0.975))
```

```
##  2.5%   50% 97.5%
##     2     7    14
```

It looks like it takes Hrothgar ~ 7 attacks to dispatch Grimweard and we're very confident that he will get the job done in somewhere between 2 and 14 attacks!

4. How many turns, on average (mean and median), does it take Grimweard to defeat Hrothgar? What is the 95% CI?

Now we repeat this for Grimweard:

```
mean(results$Attacks[results$Winner == "Grimweard"])
```

```
## [1] 9.350711
```

```
    quantile(results$Attacks[results$Winner == "Grimweard"], c(0.025,0.5,0.975))
```

```
##  2.5%   50% 97.5%
##     5     9    15
```

Looking at Grimweard's statistics, it is clear why he is losing to Hrothgar: it takes him an average of around 9 attacks to kill Hrothgar, and his lower 95% bound is 5 attacks!

5. Assume that the battle does not need to be to the death, but that Hrothgar can decide to flee the battle below some threshold. Once Hrothgar decides to flee, Grimweard is allowed to attack him one last time. Search for an optimal threshold where fleeing improves Hrothgar's chance of survival, but does not affect his chance of winning much. Create a table that has 3 pieces of information: the threshold at which Hrothgar flees, the percentage of the battles he wins, and the percentage of times that fleeing prevented dying (i.e., Hrothgar fled and Grimweard did not manage to kill him with his last attack).

To answer this question, we need to add a condition where Hrothgar flees to the `battle()` function. We

```
#thresh is the threshold at which Hrothgar runs
battle2 <- function(thresh = 0){
  hrothgar <- 25
  grimweard <- 15

  cumdmg.hrothgar <- cumsum(rdmg.hrothgar(100))
  cumdmg.grimweard <- cumsum(rdmg.grimweard(100))

  #Note that Grimweard's winning condition is now based upon subtracting the threshold from his HP
  #if thresh = 0, we should get the same results as our first simulations
  nToWin.hrothgar <- sum(cumdmg.hrothgar < grimweard) + 1
  nToWin.grimweard <- sum(cumdmg.grimweard < hrothgar - thresh) + 1

  #similar logic as previous function, but we track fleeing and death now
  if(nToWin.hrothgar < nToWin.grimweard) return(data.frame("Winner" = c("Hrothgar"), Flee  = F, Killed=
  if(sample(c("Hrothgar","Grimweard"),1) == "Hrothgar" & nToWin.grimweard == nToWin.hrothgar) return(da
  return(data.frame("Winner" = "Grimweard", Flee = cumdmg.grimweard[nToWin.grimweard] >= hrothgar - thr

}

#now loop over all possible thresholds to look for optimal strategy
strategy <- data.frame(threshold = 0:24, winning.percent = NA, saved.death = NA)
for(f in strategy$threshold){
  results2 <- battle2(f)
  for(i in 1:999) results2 <- rbind(results2, battle2(f))
  strategy$winning.percent[f+1] <- sum(results2$Winner == "Hrothgar")/1000
  summaryTab <- xtabs(rep(1, 1000) ~ Flee  + Killed + Winner, data = results2)
  strategy$saved.death[f+1] <- sum(summaryTab["TRUE","FALSE","Grimweard"])/sum(summaryTab["TRUE",,"Grimw
}

knitr::kable(strategy)
```

threshold	winning.percent	saved.death
0	0.804	0.0000000
1	0.788	0.1320755
2	0.766	0.3247863
3	0.723	0.3682310
4	0.696	0.4375000

threshold	winning.percent	saved.death
5	0.668	0.5903614
6	0.625	0.6960000
7	0.612	0.7680412
8	0.556	0.8581081
9	0.552	0.9129464
10	0.479	0.9366603
11	0.470	0.9566038
12	0.428	0.9807692
13	0.389	0.9950900
14	0.325	0.9985185
15	0.292	1.0000000
16	0.282	1.0000000
17	0.243	1.0000000
18	0.208	1.0000000
19	0.182	1.0000000
20	0.126	1.0000000
21	0.096	1.0000000
22	0.073	1.0000000
23	0.047	1.0000000
24	0.062	1.0000000

6. What would you recommend Hrothgar do?

   This is totally up to our hero's danger tolerance and the risk-reward of the battle! Assuming a reasonably strong payoff for vanquishing Grimweard, I would suggest fleeing at around 4 HP. At this threshold, Hrothgar still wins $> 70\%$ of battles, but he dies at half (!!!) the rate of staying in the battle longer.

Problem 3 (5 pts.)

Examine the following plots of divided differences. Based on the plots, do you think there is an obvious polynomial order with which to fit the underlying data? Explain your answer!!

It looks like a 4th degree polynomial would work best for fitting these data. The fourth divided differences are essentially constant, which is what we would expect for a 4th degree polynomial. We also see patterns that look cubic, quadratic, and linear for the first, second, and third divided differences, respectively; all of this supports the conclusion that we should use a 4th order polynomial to fit the data.

Problem 4 (5 pts.)

Given the points (0,2), (1,5), (2,1), write out the **unsimplified** Lagrange polynomial (i.e., do not try to simplify the sum or the fractions). After you have written down the polynomial, evaluate the polynomial **by hand** at $x = 3$

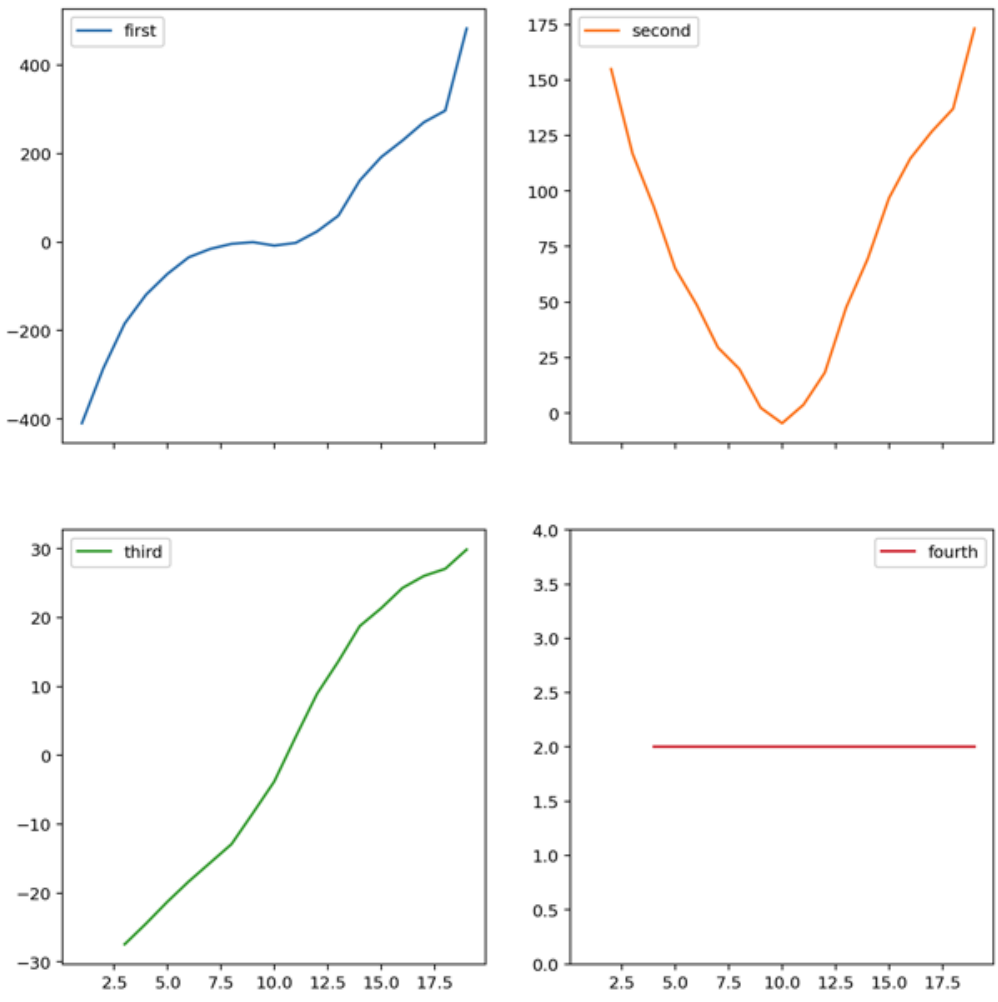


Figure 2: Divided difference

$$L(x) = y_0 \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + y_1 \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} + y_2 \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$
$$L(x) = 2\frac{(x-1)(x-2)}{(-1)(-2)} + 5\frac{(x)(x-2)}{(1)(1-2)} + 1\frac{(x)(x-1)}{(2)(2-1)}$$
$$L(x) = (x-1)(x-2) - 5(x)(x-2) + \frac{(x)(x-1)}{2}$$
$$L(x) = x^2 - 3x + 2 - 5x^2 + 10x + \frac{1}{2}x^2 - \frac{1}{2}x$$
$$L(x) = -\frac{7}{2}x^2 + \frac{13}{2}x + 2$$
$$L(3) = -\frac{63}{2} + \frac{39}{2} + \frac{4}{2} = -\frac{20}{2} = -10$$

Problem 5 (5 pts.)

Derive the inverse CDF that could be used to generate random numbers distributed according to $p(x) = 3x^2+5$ on the interval $x \in [1, 4]$. Use R to generate 1000 random numbers from this CDF and plot the resulting histogram (use 50 bins).

First, we need to normalize the area under the curve to be one. To do this, we find the integral of $p(x)$ over the interval $[1, 4]$:

$$\int_1^4 3x^2 + 5\,dx = x^3 + 5x\Big|_{x=1}^{4} = 64 + 20 - (1+5) = 78$$

Therefore the normalized curve is simply $p^*(x) = \frac{1}{78}p(x) = \frac{1}{78}(3x^2+5)$. Next, we need to find the CDF. The CDF is given by:

$$\Phi(x) = \frac{1}{78}\int_1^x 3z^2 + 5\,dz = \frac{1}{78}\left(z^3 + 5z\Big|_{z=1}^{x}\right) = \frac{1}{78}(x^3 + 5x - (1+5)) = \frac{1}{78}(x^3 + 5x - 6)$$

.

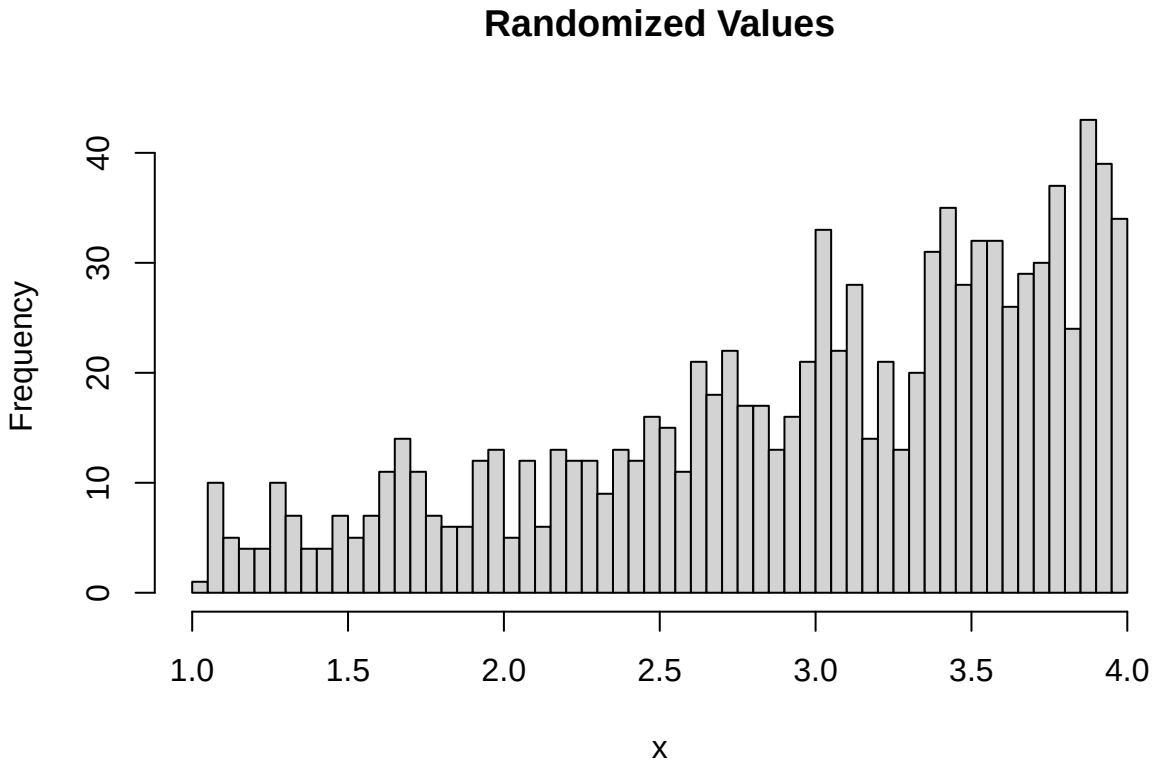
Finally, we wish to solve for the previous equation for x given $\Phi(x) \in [0, 1]$.

$$q = \frac{1}{78}(x^3 + 5x - 6)$$
$$\frac{1}{78}(x^3 + 5x - 6) - q = 0$$
$$x^3 + 5x - 6 - 78q = 0$$
$$\therefore x_k = -2\sqrt{\frac{5}{3}}\sinh\left[\frac{1}{3}\sinh^{-1}\left(\frac{-3(6+78q)}{10}\sqrt{\frac{3}{5}}\right)\right]$$

We get the final solution because we know the solution of a (depressed) cubic with one real root and a positive coefficient on the linear term. We can then generate random numbers like such:

```
#R doesn't have arcsinh
arcsinh <- function(x) {
    y <- log(x + sqrt(x ^ 2 + 1))
    return(y)
}

InvCDF <- function(q) -2*sqrt(5/3)*sinh(1/3*arcsinh(-3/10*(6+78*q)*sqrt(3/5)))

randQs <- runif(1000)
```

```r
xvals <- InvCDF(randQs)

hist(xvals,50, main = "Randomized Values", xlab = "x")
```



We could also tackle this with a numerical solution using `uniroot()` to find the inverse CDF:

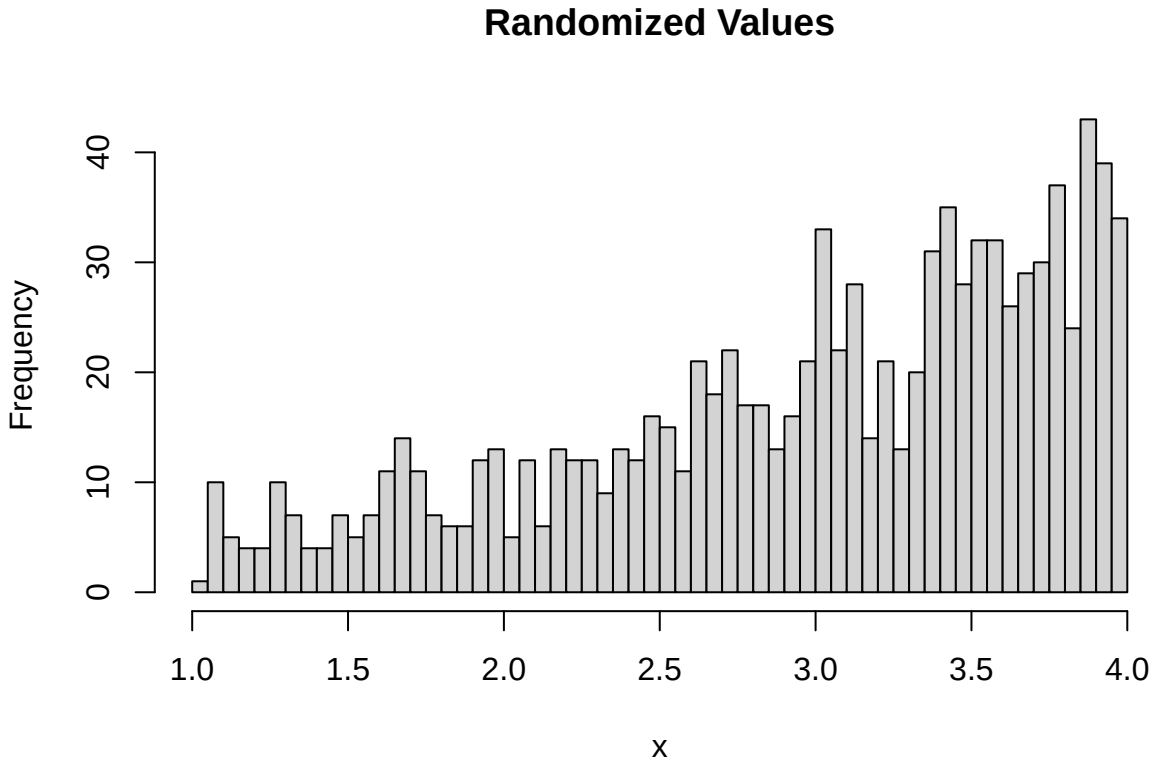
```r
numericalInverse <- function(q) uniroot(function(x) 1/78*(x^3 + 5*x - 6)- q, lower = 1, upper = 4)$root

#compare solutions
InvCDF(0.5)
```

```
## [1] 3.091378
```

```r
numericalInverse(0.5)
```

```
## [1] 3.091377
```

```r
hist(sapply(randQs,numericalInverse),50, main = "Randomized Values", xlab = "x")
```



Problem 6 (5 pts.)

Examine the following diagnostic plots produced after a fitting a linear model to some data. Discuss any issues you observe for EACH of the 4 panels. Would you be happy with model based on these diagnostics? How might you fix/improve the model?

The plots suggest there are some definite issues with the model fit. In the "residuals vs fitted" plot (top left), we can see that our errors are not iid, and that, in particular, when the fitted value is small, the residual is large. The "scale-location" plot (bottom left) is essentially telling us the same thing: small fitted values have abnormally large residuals. The "normal Q-Q" plot shows that we have some "heavy tails" in the distribution of our residuals. Again, this plot shows that we have a preponderance of large, positive residuals. Finally, the "residuals vs. leverage" plot in the bottom right shows that there are (at least) 2 outliers we might consider trimming from the data: points 56 and 14.

Based on these plots, I would not be happy with model fit. The fitted values in the smaller range of the y-variate have poor fits. Depending on what the goal of the model was, you could perhaps, e.g., 1) try to fit the lower half and upper half of the data independently, 2) attempt a transform of the data, or 3) try using a model that accommodate non-linearity better.

Problem 7 (5 pts.)

What are the two main categories of models that we have discussed in class? Discuss the strengths and weaknesses of both types of models. Give an example of each type of model.

The two categories of models are mechanistic and statistical models. Many examples could be given for both types of models: Essentially in a mechanistic model this is a hypothesized mechanism, so gravity acting on a falling object would be a classical mechanistic model. For statistical models, we are simply seeking seeking a method by which we can predict outcomes to the best of our ability; machine learning models are very good examples of this type of models. Both types of models can fall prey to pitfalls (poor fits,

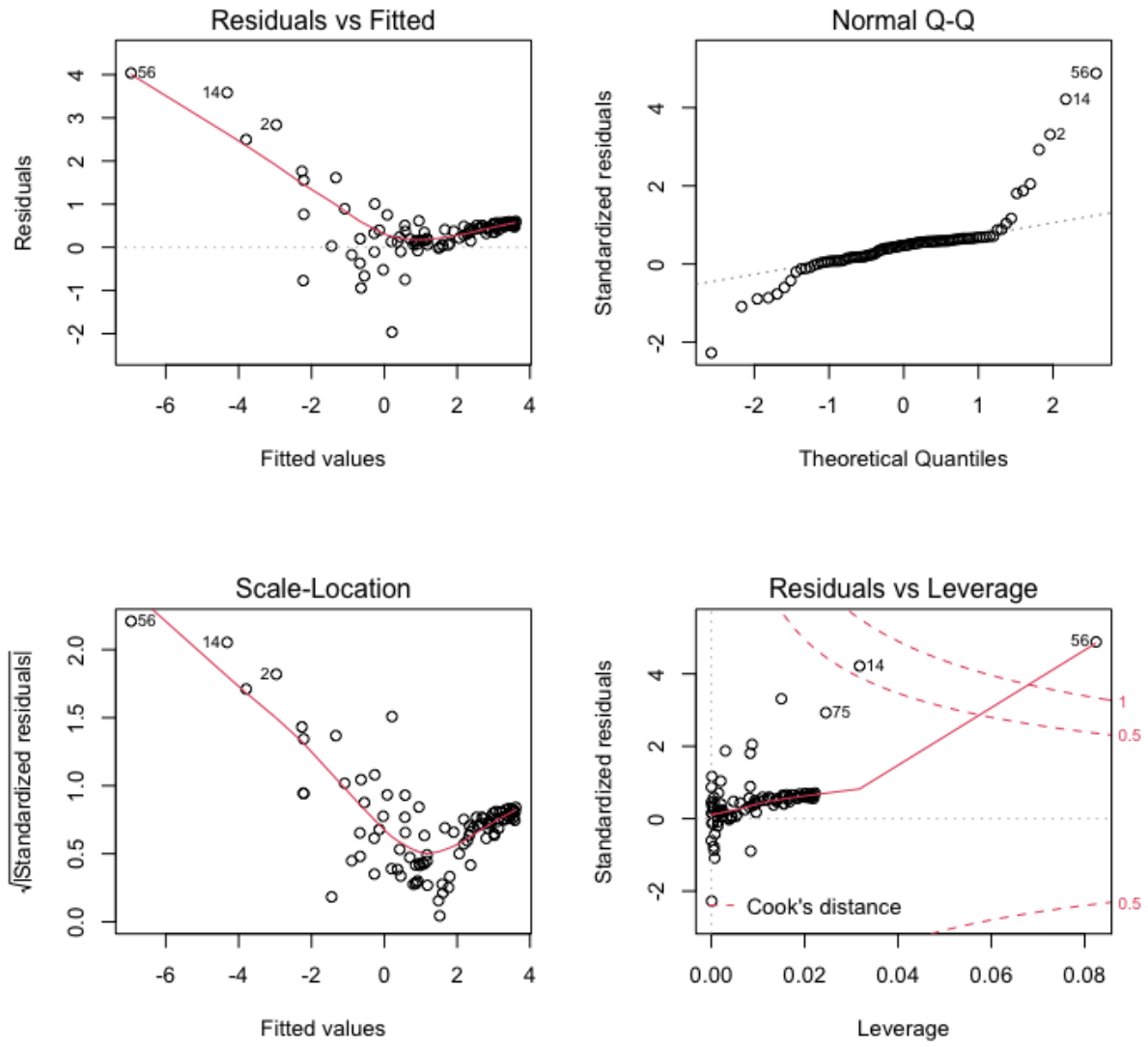


Figure 3: Diagnostics

over-parameterization, extrapolation, misspecification, etc); you as a modeler must choose which works best for your application!