FISH621 - Homework 4

Marcel Gietzmann-Sanders

```r
library(Distance)
```

```
## Loading required package: mrds
```

```
## This is mrds 2.3.0
## Built: R 4.1.2; ; 2024-04-12 19:50:10 UTC; unix
##
## **Change to default variance estimator for point transects. The default encounter rate varianc
e estimator for point transects is now 'P2' (changed from 'P3'). See 'Uncertainty' section of ?dh
t for more information.**
##
## MCDS.exe not detected, single observer analyses will only be run using optimiser in mrds R lib
rary. See ?MCDS for details.
```

```
##
## Attaching package: 'Distance'
```

```
## The following object is masked from 'package:mrds':
##
##     create.bins
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
```

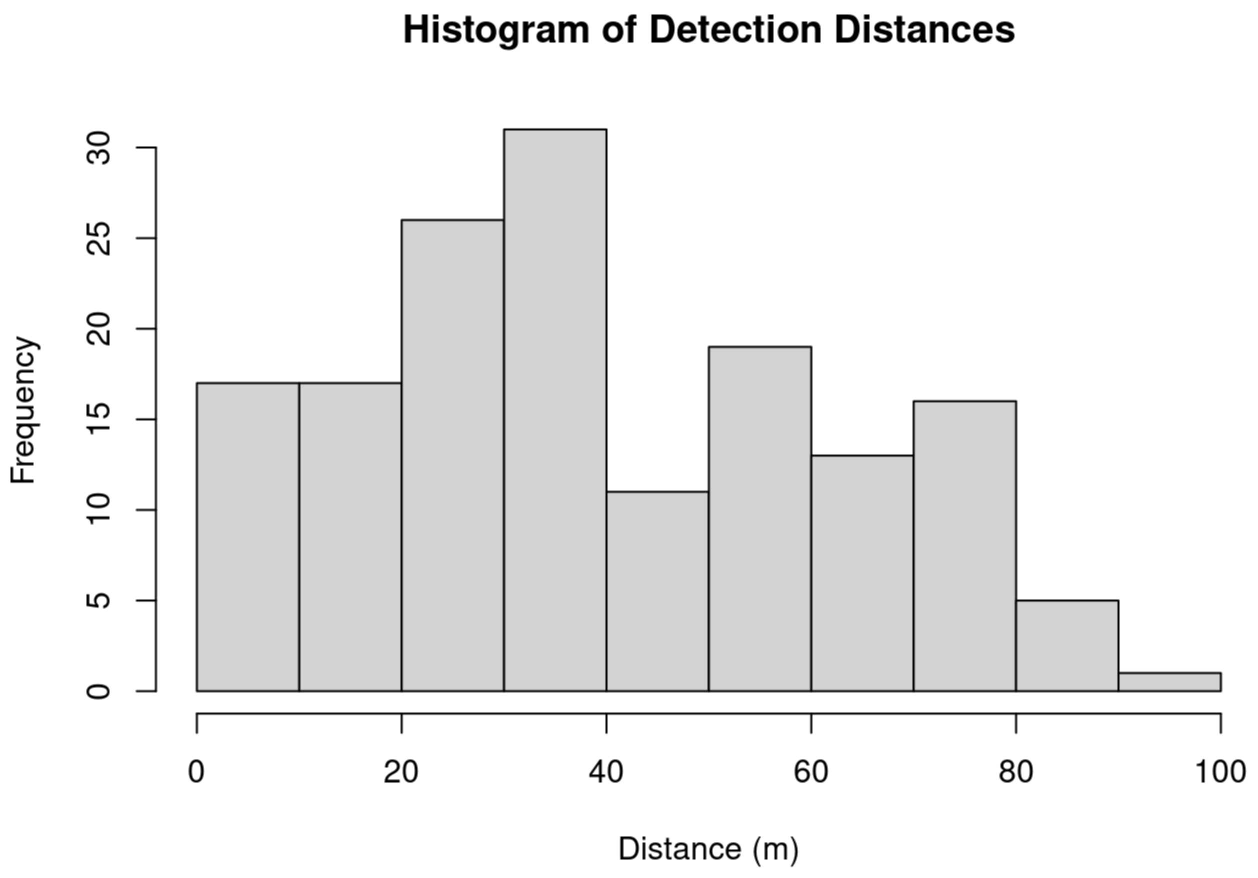
Problem 1

```r
data = read.csv("wren.csv")
head(data)
```

```
##   X Region.Label  Area Sample.Label Effort object distance Study.Area
## 1 1     Montrave 0.332            1  0.416      5       15 Montrave 4
## 2 2     Montrave 0.332            1  0.416      6       80 Montrave 4
## 3 3     Montrave 0.332            1  0.416      7       35 Montrave 4
## 4 4     Montrave 0.332            1  0.416      8       55 Montrave 4
## 5 5     Montrave 0.332            1  0.416     12       12 Montrave 4
## 6 6     Montrave 0.332            1  0.416     13       75 Montrave 4
```

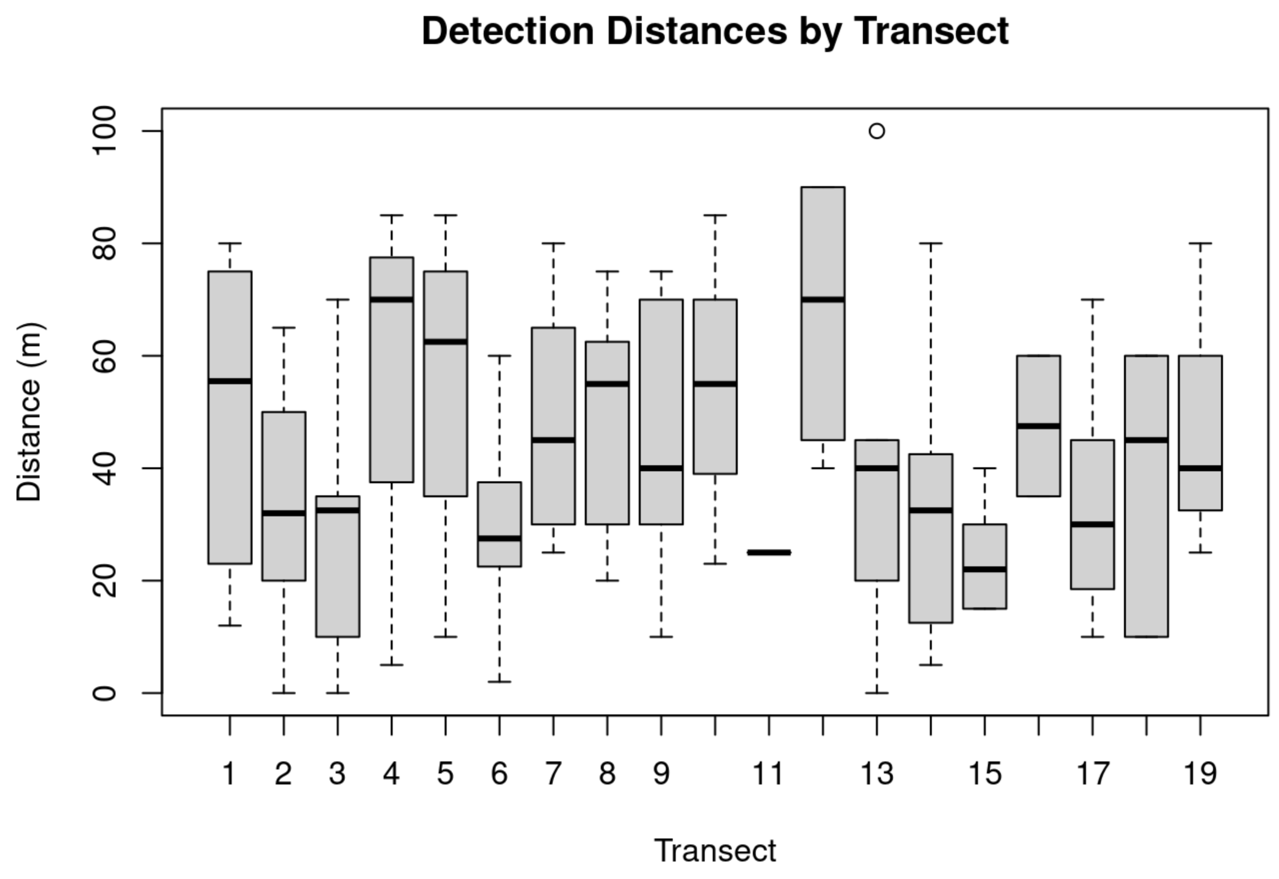
1. Plot a histogram of detection distances.

```
hist(
    data$distance,
    main="Histogram of Detection Distances",
    xlab="Distance (m)",
    ylab="Frequency",
)
```



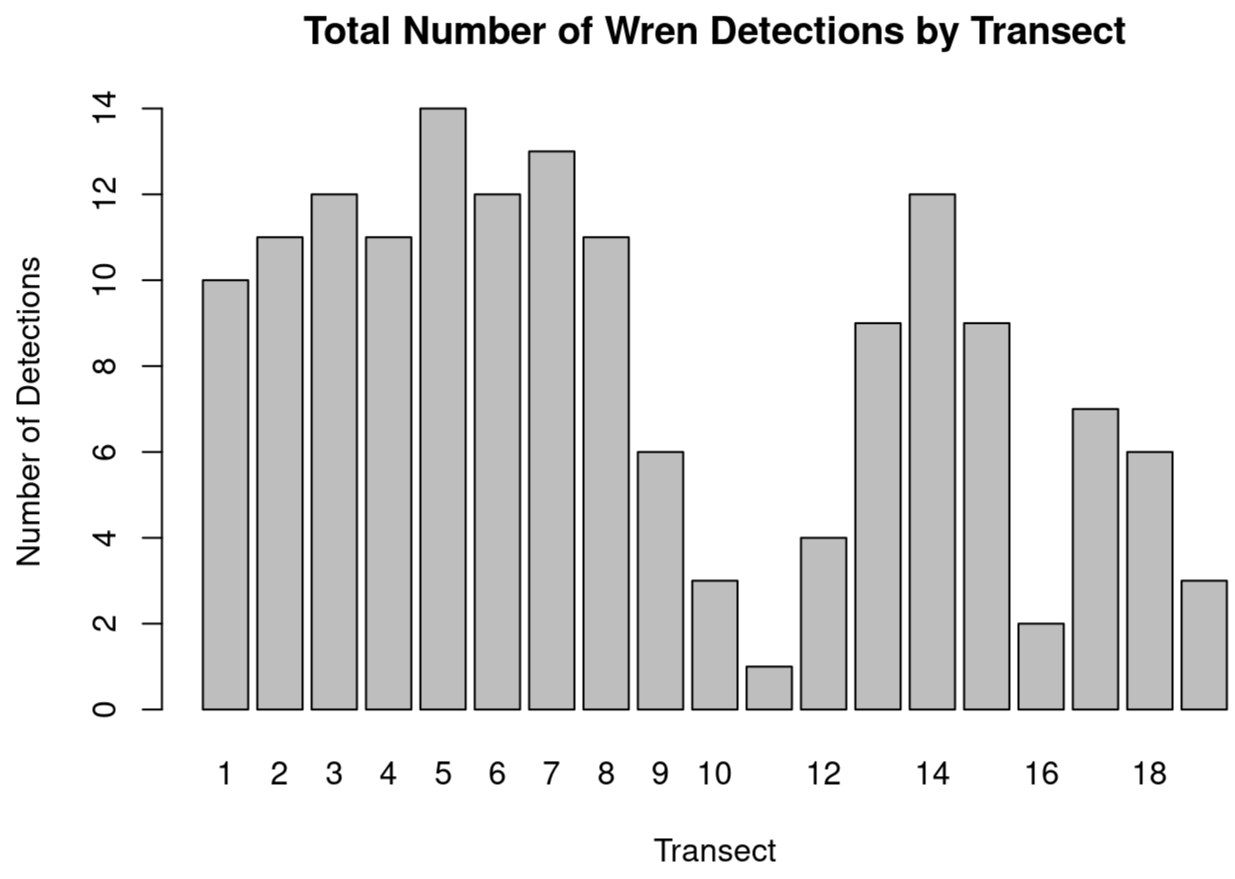
2. Use boxplots to visualize detection distances by transect.

```
boxplot(
    distance ~ Sample.Label,
    data=data,
    main="Detection Distances by Transect",
    xlab="Transect",
    ylab="Distance (m)"
)
```



3. Create a bar or column plot showing the total number of wren detections by transect number.

```
barplot(
    table(data$Sample.Label),
    main="Total Number of Wren Detections by Transect",
    xlab="Transect",
    ylab="Number of Detections"
)
```



4. Fit three alternative models to estimate abundance and density from these distance sampling data.

a. Half-normal

```
wren.hn = ds(data, key="hn", adjustment=NULL, convert_units=0.001)
```

```
## Fitting half-normal key function
```

```
## AIC= 1418.188
```

b. Hazard-rate

```
wren.hr = ds(data, key="hr", adjustment=NULL, convert_units=0.001)
```

```
## Fitting hazard-rate key function
```

```
## AIC= 1412.133
```

c. Uniform with a cosine adjustment

```
wren.cos = ds(data, key="unif", adjustment="cos", convert_units=0.001)
```

```
## Starting AIC adjustment term selection.
## Fitting uniform key function
```

```
## AIC= 1436.813
```

```
## Fitting uniform key function with cosine(1) adjustments
```

```
## AIC= 1421.634
```

```
## Fitting uniform key function with cosine(1,2) adjustments
```

```
## AIC= 1417.802
```

```
## Fitting uniform key function with cosine(1,2,3) adjustments
```

```
## Warning in check.mono(result, n.pts = control$mono.points): Detection function
## is not strictly monotonic!
```

```
## AIC= 1416.433
```

```
## Fitting uniform key function with cosine(1,2,3,4) adjustments
```

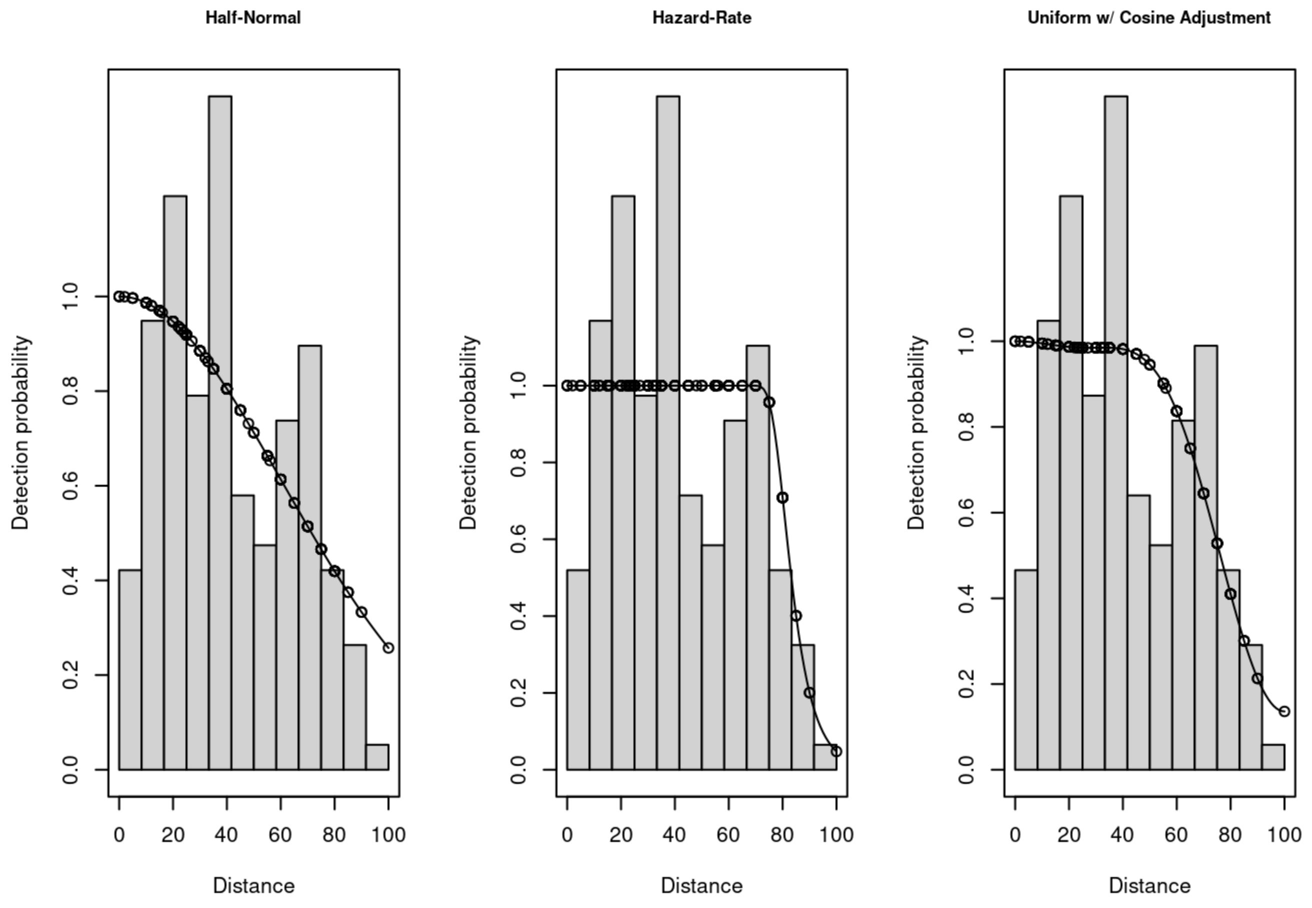
```
## AIC= 1416.966
```

```
##
## Uniform key function with cosine(1,2,3) adjustments selected.
```

```
## Warning in mrds::check.mono(model, n.pts = 20): Detection function is not
## strictly monotonic!
```

5. Plot a comparison of the fitted detection functions to the distance data.

```
par(mfrow = c(1, 3))
plot(wren.hn, main="Half-Normal")
plot(wren.hr, main="Hazard-Rate")
plot(wren.cos, main="Uniform w/ Cosine Adjustment")
```



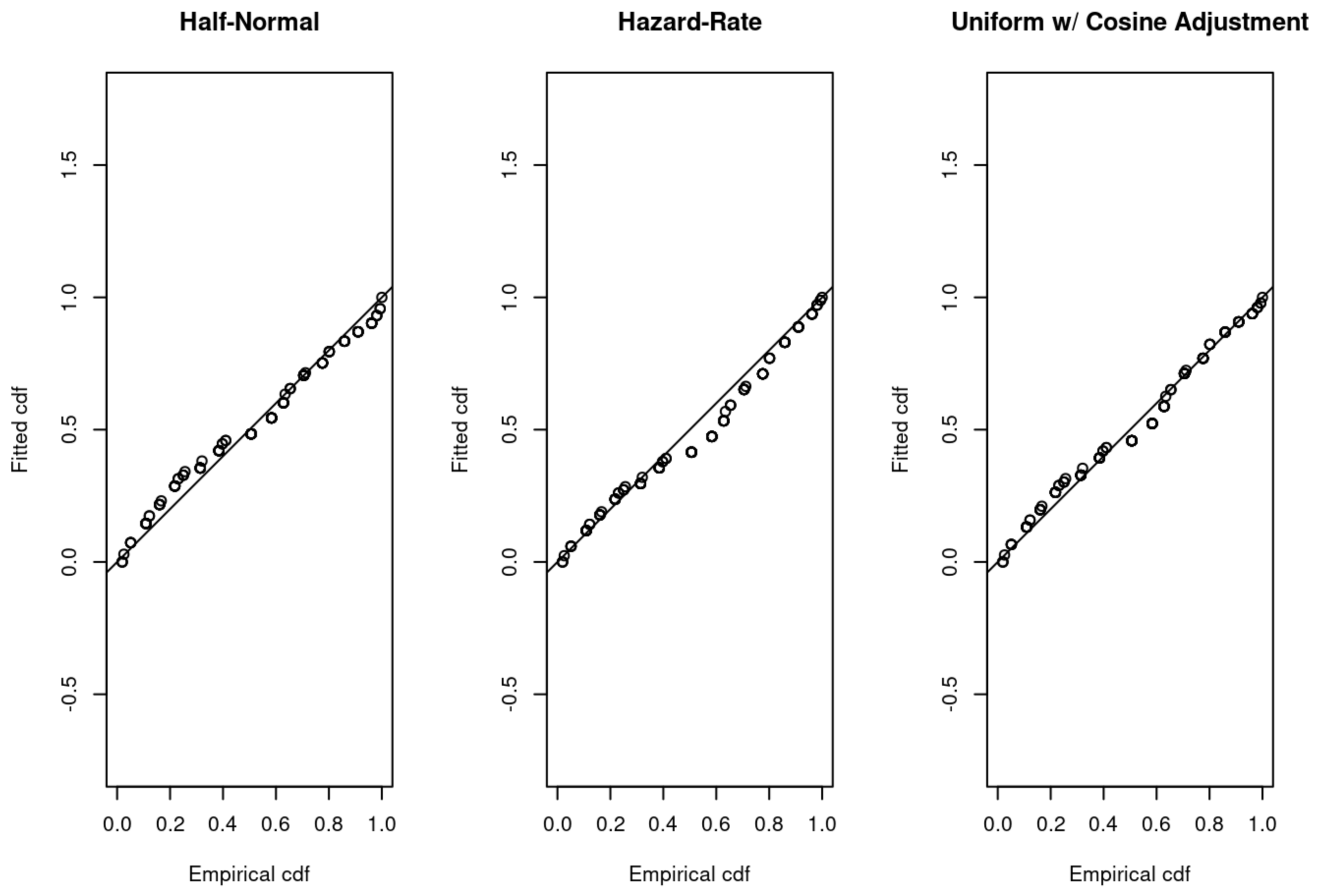
```
par(mfrow = c(1, 3))
gof_ds(wren.hn, main="Half-Normal")
```

```
##
## Goodness of fit results for ddf object
##
## Distance sampling Cramer-von Mises test (unweighted)
## Test statistic = 0.389692 p-value = 0.0769337
```

```
gof_ds(wren.hr, main="Hazard-Rate")
```

```
##
## Goodness of fit results for ddf object
##
## Distance sampling Cramer-von Mises test (unweighted)
## Test statistic = 0.249898 p-value = 0.1885
```

```
gof_ds(wren.cos, main="Uniform w/ Cosine Adjustment")
```



```
##
## Goodness of fit results for ddf object
##
## Distance sampling Cramer-von Mises test (unweighted)
## Test statistic = 0.241302 p-value = 0.199913
```

6. Create a summary table comparing density estimates, CV’s and lower/upper confidence intervals, across the three alternative models.

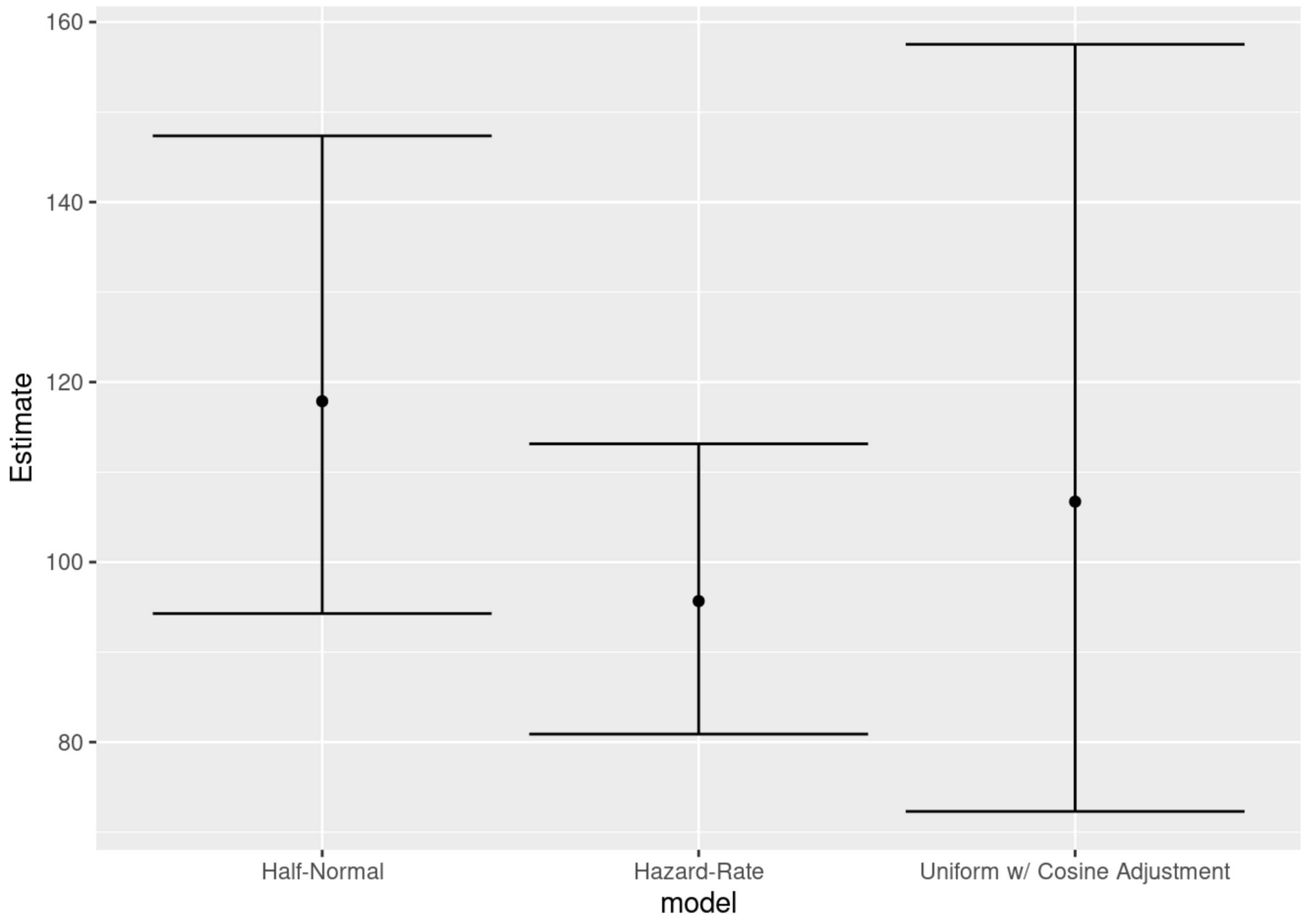
```
summary = rbind(
    wren.hn$dht$individuals$D[,c("Estimate", "se", "cv", "lcl", "ucl")],
    wren.hr$dht$individuals$D[,c("Estimate", "se", "cv", "lcl", "ucl")],
    wren.cos$dht$individuals$D[,c("Estimate", "se", "cv", "lcl", "ucl")]
)
summary$model = c("Half-Normal", "Hazard-Rate", "Uniform w/ Cosine Adjustment")
summary
```

```
##   Estimate        se        cv      lcl      ucl                        model
## 1 117.8700 13.250021 0.1124121 94.28403 147.3563                  Half-Normal
## 2  95.6668  7.782016 0.0813450 80.89338 113.1383                  Hazard-Rate
## 3 106.7171 21.255248 0.1991738 72.29793 157.5223 Uniform w/ Cosine Adjustment
```

7. Create a figure showing the density estimates from the three models

and associated uncertainty.

```
ggplot(summary, aes(model, Estimate)) + geom_point() +
geom_errorbar(aes(ymin = lcl, ymax = ucl))
```



8. Create a summary table comparing abundance estimates for the entire estate, CV's and lower/upper confidence intervals, across the three alternative models.

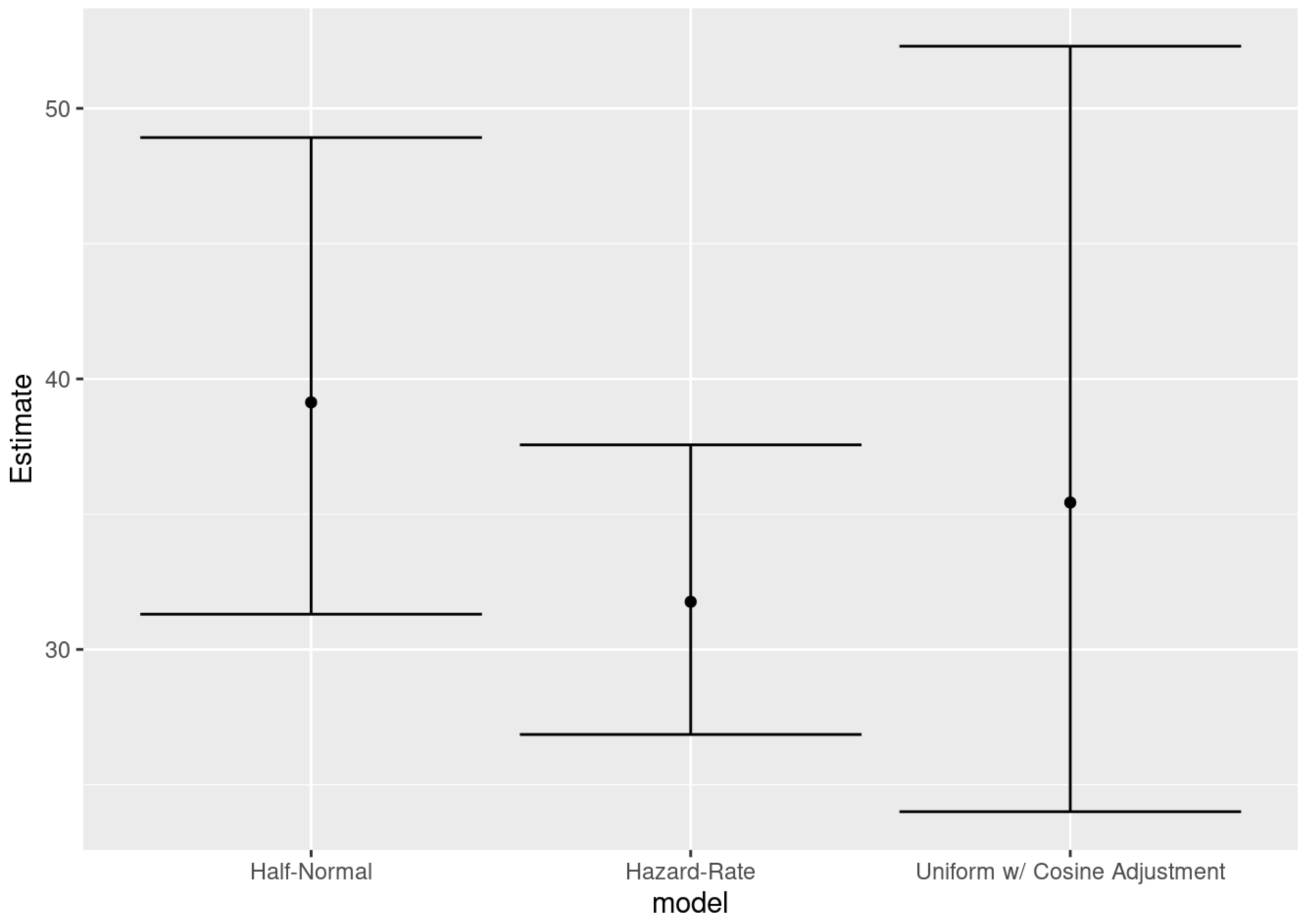
```
summary = rbind(
    wren.hn$dht$individuals$N[,c("Estimate", "se", "cv", "lcl", "ucl")],
    wren.hr$dht$individuals$N[,c("Estimate", "se", "cv", "lcl", "ucl")],
    wren.cos$dht$individuals$N[,c("Estimate", "se", "cv", "lcl", "ucl")]
)
summary$model = c("Half-Normal", "Hazard-Rate", "Uniform w/ Cosine Adjustment")
summary
```

```
##   Estimate       se        cv      lcl      ucl                        model
## 1 39.13286 4.399007 0.1124121 31.30230 48.92230                  Half-Normal
## 2 31.76138 2.583629 0.0813450 26.85660 37.56191                  Hazard-Rate
## 3 35.43007 7.056742 0.1991738 24.00291 52.29740 Uniform w/ Cosine Adjustment
```

9. Create a figure showing the abundance estimates from the three

models and associated uncertainty.

```
ggplot(summary, aes(model, Estimate)) + geom_point() +
geom_errorbar(aes(ymin = lcl, ymax = ucl))
```



10. Please identify which model you believe most reliable and provide your justification for this choice.

I’d go with the hazard rate model for a couple of reasons.

1. It has the best AIC score of the three models.
2. Much like the duck nest data it seems like we have a pretty steady detection probability for a while before the drop off and the hazard rate model does a good job of capturing that.
3. The deviation in the QQ plot gives me some hesitation but its in the same all the models have trouble.

Problem 2

```
data = read.csv("ducks-area-effort.csv")
head(data)
```

```
##   Region.Label  Area Sample.Label Effort distance
## 1      Default 40.47            1 128.75     0.06
## 2      Default 40.47            1 128.75     0.07
## 3      Default 40.47            1 128.75     0.04
## 4      Default 40.47            1 128.75     0.01
## 5      Default 40.47            1 128.75     0.37
## 6      Default 40.47            1 128.75     0.36
```

1. Estimate density and abundance for duck nests using distance methods and a half-normal detection (sighting) function.

```
duck.hn = ds(data, key="hn", adjustment=NULL, convert_units=0.001)
```

```
## Fitting half-normal key function
```

```
## AIC= 928.134
```

```
duck.hn$dht$individuals$N
```

```
##   Label Estimate       se         cv      lcl      ucl       df
## 1 Total 2011.232 118.8492 0.05909274 1788.908 2261.188 99.55677
```

2. Calculate the area sampled by each strip transect (a), in km2, if you were to treat each line transect as a strip transect and use nest sightings out to 1.25 meters on each side of the transect.

```
grouped = data[data$distance <= 1.25,] %>% group_by(Sample.Label) %>% summarise(Effort = mean(Eff
ort), S = n())
grouped$Area = 2 * 1.25 / 1000 * grouped$Effort
head(grouped)
```

```
## # A tibble: 6 x 4
##   Sample.Label Effort     S  Area
##          <int>  <dbl> <int> <dbl>
## 1            1   129.    18 0.322
## 2            2   129.    15 0.322
## 3            3   129.    24 0.322
## 4            4   129.    21 0.322
## 5            5   129.    14 0.322
## 6            6   129.    17 0.322
```

3. Estimate density and abundance, if you treat each line transect as a strip transect and use nest sightings out to 1.25 meters on each side of the transect.

I'm assuming the desire here is for overall density as if we have one long transect.

```
density = sum(grouped$S) / sum(grouped$Area)
density
```

```
## [1] 47.06796
```

```
regions = data %>% group_by(Region.Label) %>% summarise(Area = mean(Area))
regions
```

```
## # A tibble: 1 x 2
##   Region.Label  Area
##   <chr>        <dbl>
## 1 Default       40.5
```

```
(abundance = density * sum(regions$Area))
```

```
## [1] 1904.84
```

4. If n = 20 is the total number of transects sampled, given the area sampled by each transect a, calculate the effective number of transects that could be sampled N.

```
(N_transects = sum(regions$Area) / mean(grouped$Area))
```

```
## [1] 125.732
```

5. Assuming the strip transects are randomly placed, please use simple random sampling theory to calculate an estimate of the variance and the coefficient of variation (CV) for the total abundance estimate.

```
n = 20
S_bar.mean = mean(grouped$S)
S_bar.var = (N_transects-n)/N_transects * var(grouped$S)/n
N_bar = N_transects * S_bar.mean
N_bar.var = N_transects^2 * S_bar.var
N_bar.se = sqrt(N_bar.var)
N_bar.cv = N_bar.se / N_bar
D_bar = N_bar / sum(regions$Area)
c(N_bar, N_bar.var, N_bar.se, N_bar.cv, D_bar)
```

```
## [1] 1.904840e+03 1.317321e+04 1.147746e+02 6.025419e-02 4.706796e+01
```

```
duck.strip1.N = round(N_bar, 2)
duck.strip1.N.var = round(N_bar.var, 2)
duck.strip1.N.se = round(N_bar.se, 2)
duck.strip1.N.cv = round(N_bar.cv, 3)
duck.strip1.D = round(D_bar, 2)
```

6. Next, please repeat this process, if we only considered nest sightings out to 1.0 meters on each side of the transects.

```
grouped = data[data$distance <= 1,] %>% group_by(Sample.Label) %>% summarise(Effort = mean(Effor
t), S = n())
grouped$Area = 2 * 1 / 1000 * grouped$Effort
(N_transects = sum(regions$Area) / mean(grouped$Area))
```

```
## [1] 157.165
```

```
S_bar.mean = mean(grouped$S)
S_bar.var = (N_transects-n)/N_transects * var(grouped$S)/n
N_bar = N_transects * S_bar.mean
N_bar.var = N_transects^2 * S_bar.var
N_bar.se = sqrt(N_bar.var)
N_bar.cv = N_bar.se / N_bar
D_bar = N_bar / sum(regions$Area)
c(N_bar, N_bar.var, N_bar.se, N_bar.cv, D_bar)
```

```
## [1] 1.933130e+03 2.361119e+04 1.536593e+02 7.948732e-02 4.776699e+01
```

```
duck.strip2.N = round(N_bar, 2)
duck.strip2.N.var = round(N_bar.var, 2)
duck.strip2.N.se = round(N_bar.se, 2)
duck.strip2.N.cv = round(N_bar.cv, 3)
duck.strip2.D = round(D_bar, 2)
```

7. Please create a table (and report this in your word document) summarizing your estimates of: (a) density $D$, (b) total abundance $N$, standard error or standard deviation in $N$, and the CV of $N$, based on distance methods and the two strip transect approximations.

```
table = rbind(
    c("distance", round(duck.hn$dht$individuals$D$Estimate, 2), round(duck.hn$dht$individuals$N$E
stimate, 2), round(duck.hn$dht$individuals$N$se, 2), round(duck.hn$dht$individuals$N$cv, 3)),
    c("1.25m", duck.strip1.D, duck.strip1.N, duck.strip1.N.se, duck.strip1.N.cv),
    c("1.0m", duck.strip2.D, duck.strip2.N, duck.strip2.N.se, duck.strip2.N.cv)
)
colnames(table) = c("Method", "Density", "Abundance", "SE", "CV")
table
```

```
##      Method     Density Abundance SE       CV
## [1,] "distance" "49.7"  "2011.23" "118.85" "0.059"
## [2,] "1.25m"    "47.07" "1904.84" "114.77" "0.06"
## [3,] "1.0m"     "47.77" "1933.13" "153.66" "0.079"
```

8. Please describe in the word document you submit, why treating our line transect data as strip transects and applying estimators under simple random sampling may or may not be appropriate.

As is clear from comparing the 1m strip to the 1.25m strip, by removing the observations beyond our strip width we effectively "lose" effort and therefore increase the variance in our estimate somewhat needlessly. So by extending the allowable distance we end up with more stable estimates due to an increase in "effort". However if we don't account for the decrease in observation probability as the distance increases we'll end up underestimating our total abundance as we'll have effectively sampled far fewer individuals that we would have expected.

Distance sampling allows us to increase our effort without invalidating a basic assumption required for strip transects and simple random sampling.

Problem 3

```
data = read.csv("Scallop CPUE.csv")
head(data)
```

```
##       Region Season Catch_Meat Catch_Round Dredge_Hours
## 1 Bering Sea   2000     205520     2376601         3355
## 2 Bering Sea   2001     140871     1700500         3072
## 3 Bering Sea   2002      92240      951938         2038
## 4 Bering Sea   2003      42590      537552         1020
## 5 Bering Sea   2004      10050      128128          275
## 6 Bering Sea   2005      23220      231700          602
```

1. Calculate catch per unit effort (CPUE) for both meat and round weights.

```
data$CPUE_Meat = data$Catch_Meat / data$Dredge_Hours
data$CPUE_Round = data$Catch_Round / data$Dredge_Hours
head(data)
```

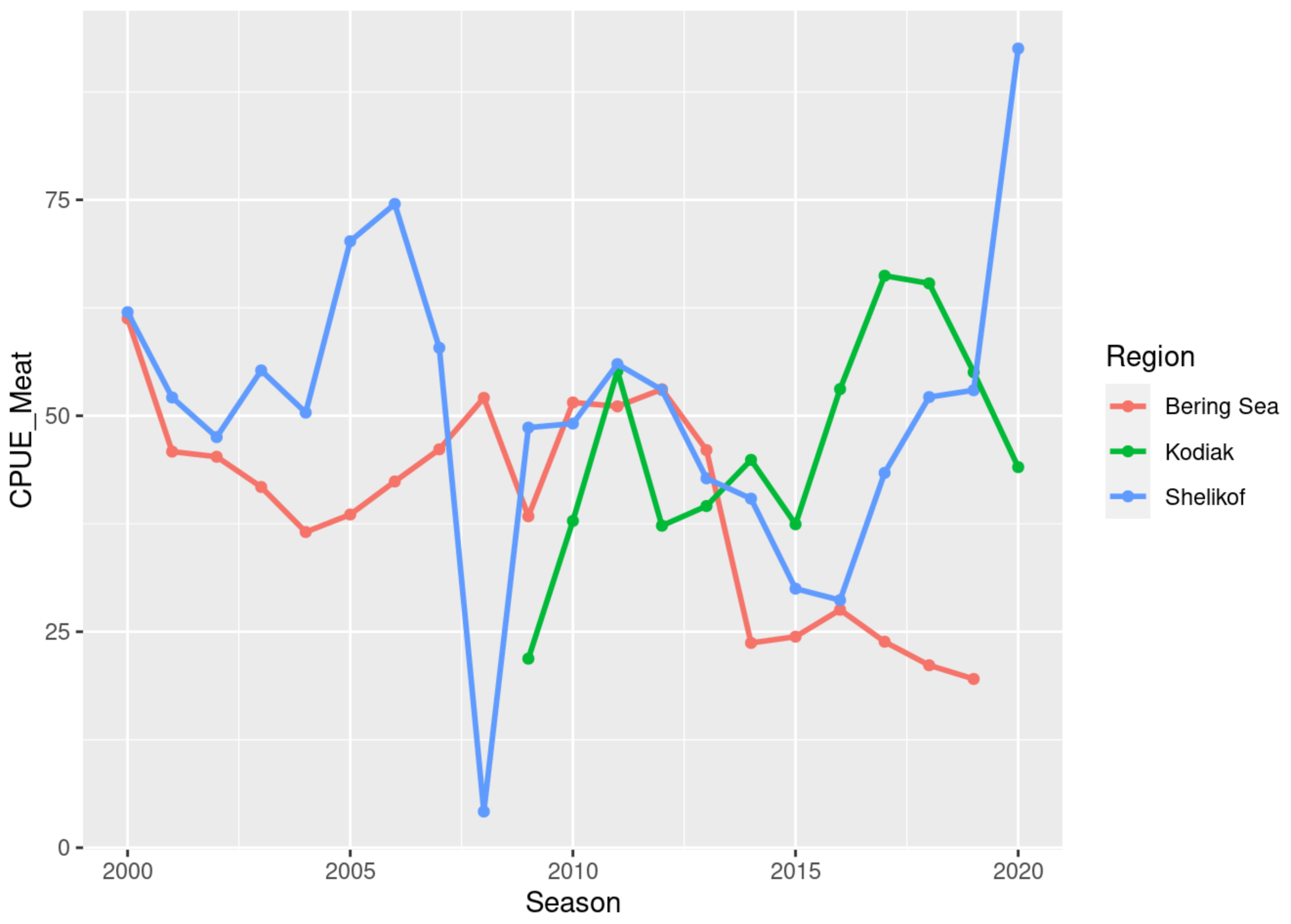
```
##       Region Season Catch_Meat Catch_Round Dredge_Hours CPUE_Meat CPUE_Round
## 1 Bering Sea   2000     205520     2376601         3355  61.25782   708.3759
## 2 Bering Sea   2001     140871     1700500         3072  45.85645   553.5482
## 3 Bering Sea   2002      92240      951938         2038  45.26006   467.0942
## 4 Bering Sea   2003      42590      537552         1020  41.75490   527.0118
## 5 Bering Sea   2004      10050      128128          275  36.54545   465.9200
## 6 Bering Sea   2005      23220      231700          602  38.57143   384.8837
```

2. Plot the timeseries (i.e. across seasons) of CPUE by region for both

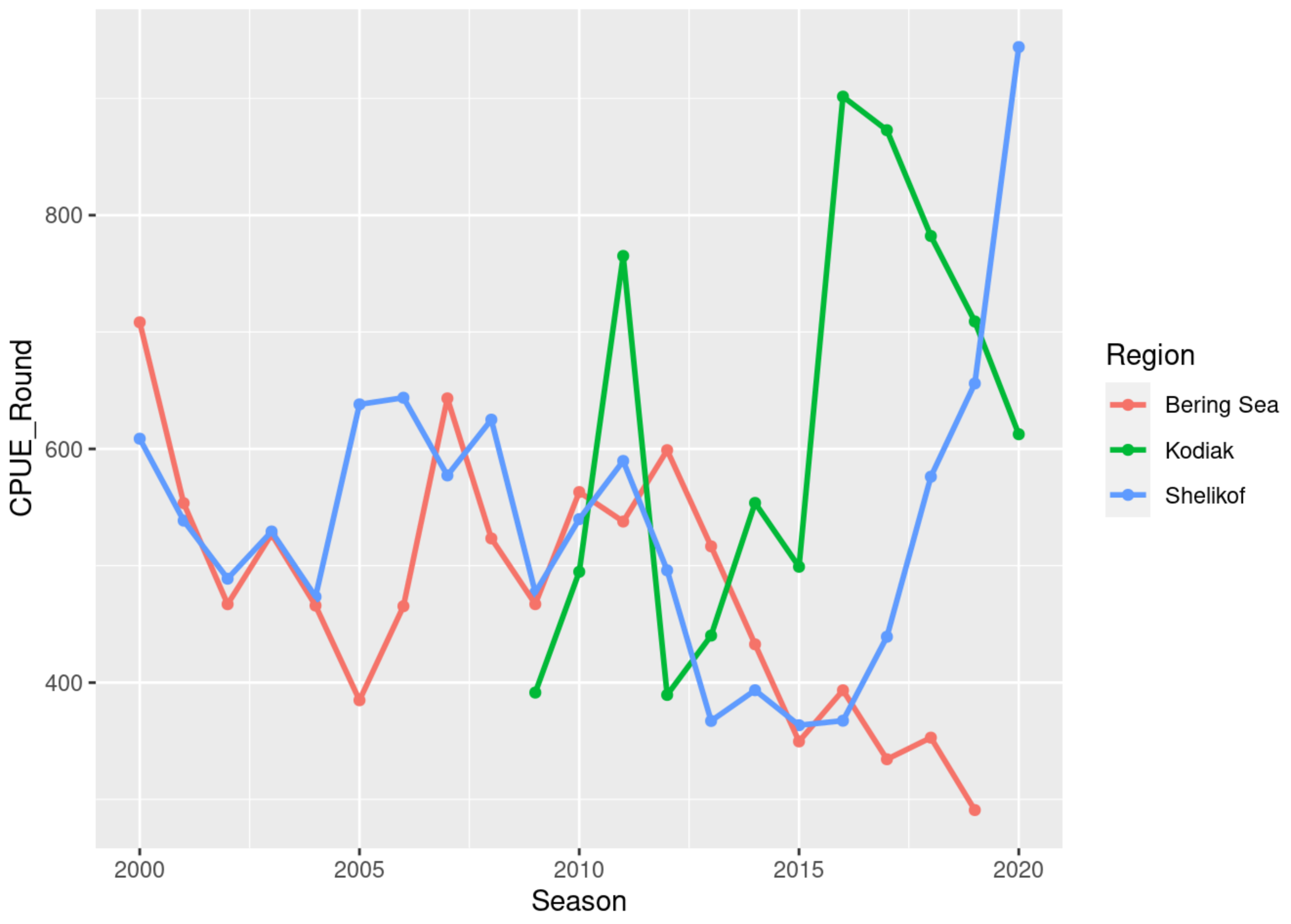
meat and round weights.

```
ggplot(data, aes(x=Season, y=CPUE_Meat, color=Region)) +
    geom_point() + geom_line(size = 1)
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



```
ggplot(data, aes(x=Season, y=CPUE_Round, color=Region)) +
    geom_point() + geom_line(size = 1)
```



3. Use a GLM to calculate a model-based index of abundance for scallops based on meat weight, not controlling for region.

```
data$fSeason = factor(data$Season)
data$fRegion = factor(data$Region)
indices = data[data$Region == "Shelikof", c("fSeason", "fRegion", "Season")] %>% distinct(fSeaso
n, Season, fRegion)
```

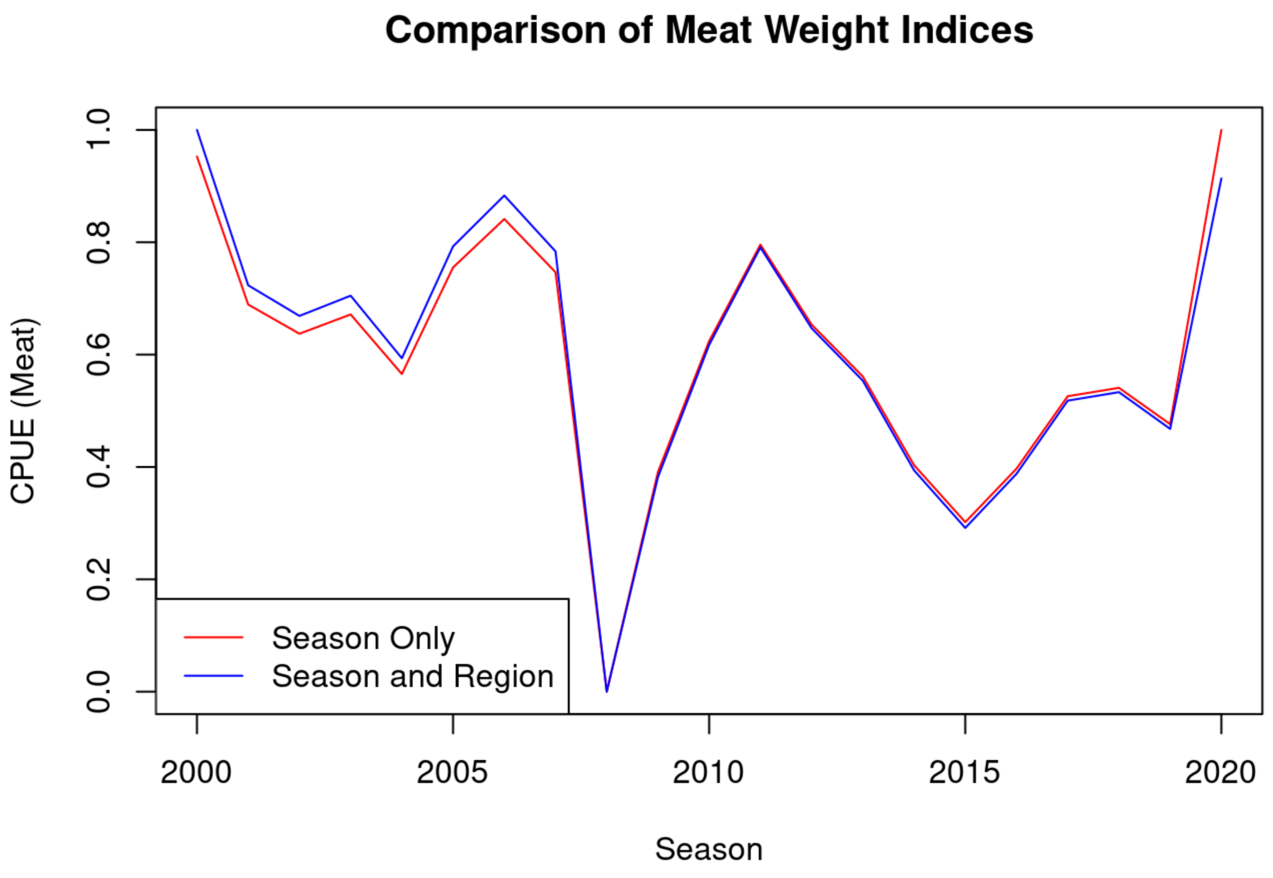
```
model.s = glm(log(CPUE_Meat + 1) ~ fSeason, data=data)
indices$meat_season = exp(predict(model.s, newdata=indices, type="response")) - 1
# normalize this index
indices$meat_season = (indices$meat_season - min(indices$meat_season)) / (max(indices$meat_seaso
n) - min(indices$meat_season))
```

4. Use a GLM to calculate a model-based index of abundance for scallops based on meat weight, this time controlling for region. Use the Shelikof region as your reference region when generating the index.

```
model.sr = glm(log(CPUE_Meat + 1) ~ fSeason + fRegion, data=data)
indices$meat_season_region = exp(predict(model.sr, newdata=indices, type="response")) - 1
indices$meat_season_region = (indices$meat_season_region - min(indices$meat_season_region)) / (ma
x(indices$meat_season_region) - min(indices$meat_season_region))
```

5. Plot a comparison of the meat weight indices from models with and without a region effect.

```
plot.default(
    indices$Season,
    indices$meat_season,
    type="l",
    col="red",
    xlab="Season",
    ylab="CPUE (Meat)",
    main="Comparison of Meat Weight Indices"
)
lines(
    indices$Season,
    indices$meat_season_region,
    col="blue"
)
legend("bottomleft", legend=c("Season Only", "Season and Region"), col=c("red", "blue"), lty=1:1)
```



6. Use a GLM to calculate a model-based index of abundance for scallops based on round weight, not controlling for region.

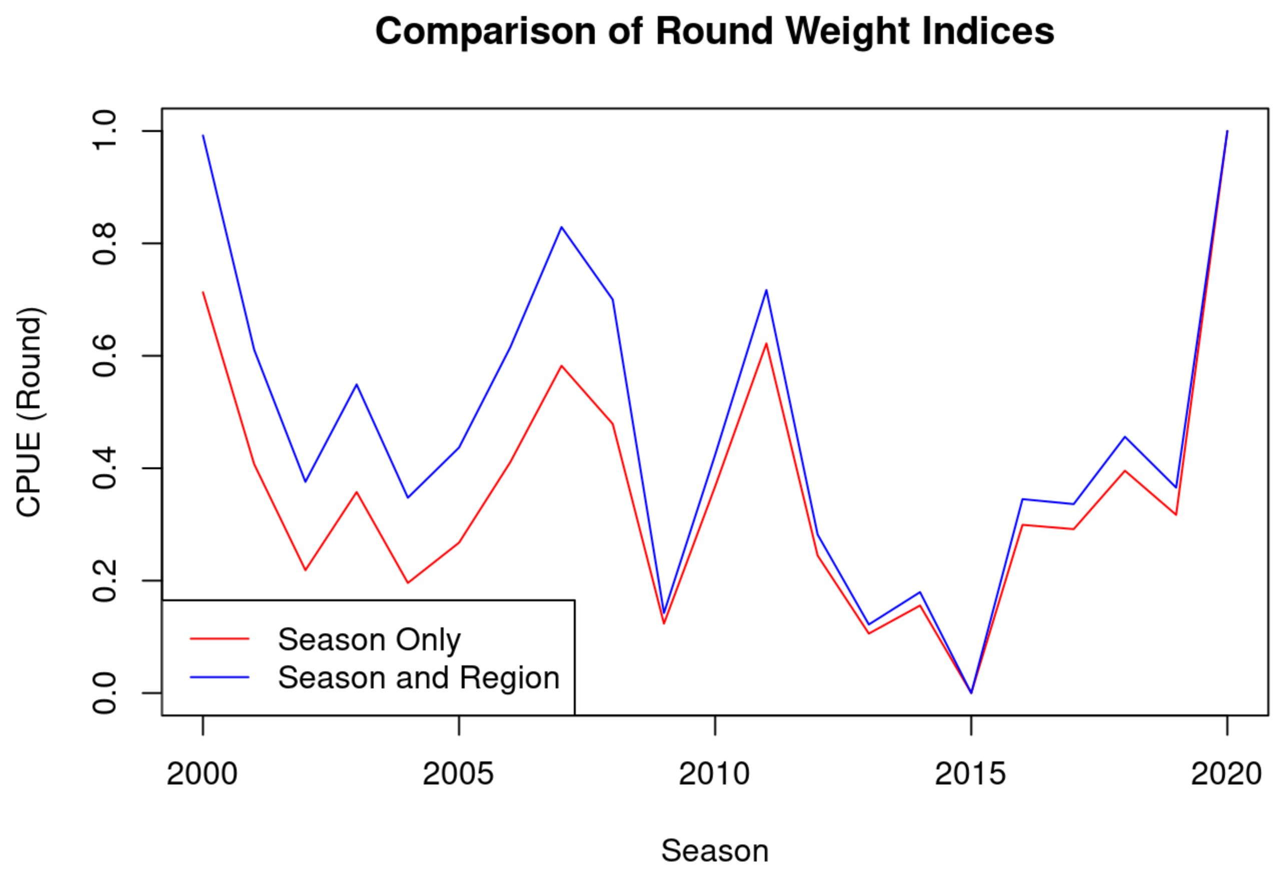
```
model.s_r = glm(log(CPUE_Round + 1) ~ fSeason, data=data)
indices$round_season = exp(predict(model.s_r, newdata=indices, type="response")) - 1
indices$round_season = (indices$round_season - min(indices$round_season)) / (max(indices$round_se
ason) - min(indices$round_season))
```

7. Use a GLM to calculate a model-based index of abundance for scallops based on round weight, this time controlling for region. Use the Shelikof region as your reference region when generating the index.

```
model.sr_r = glm(log(CPUE_Round + 1) ~ fSeason + fRegion, data=data)
indices$round_season_region = exp(predict(model.sr_r, newdata=indices, type="response")) - 1
indices$round_season_region = (indices$round_season_region - min(indices$round_season_region)) /
(max(indices$round_season_region) - min(indices$round_season_region))
```

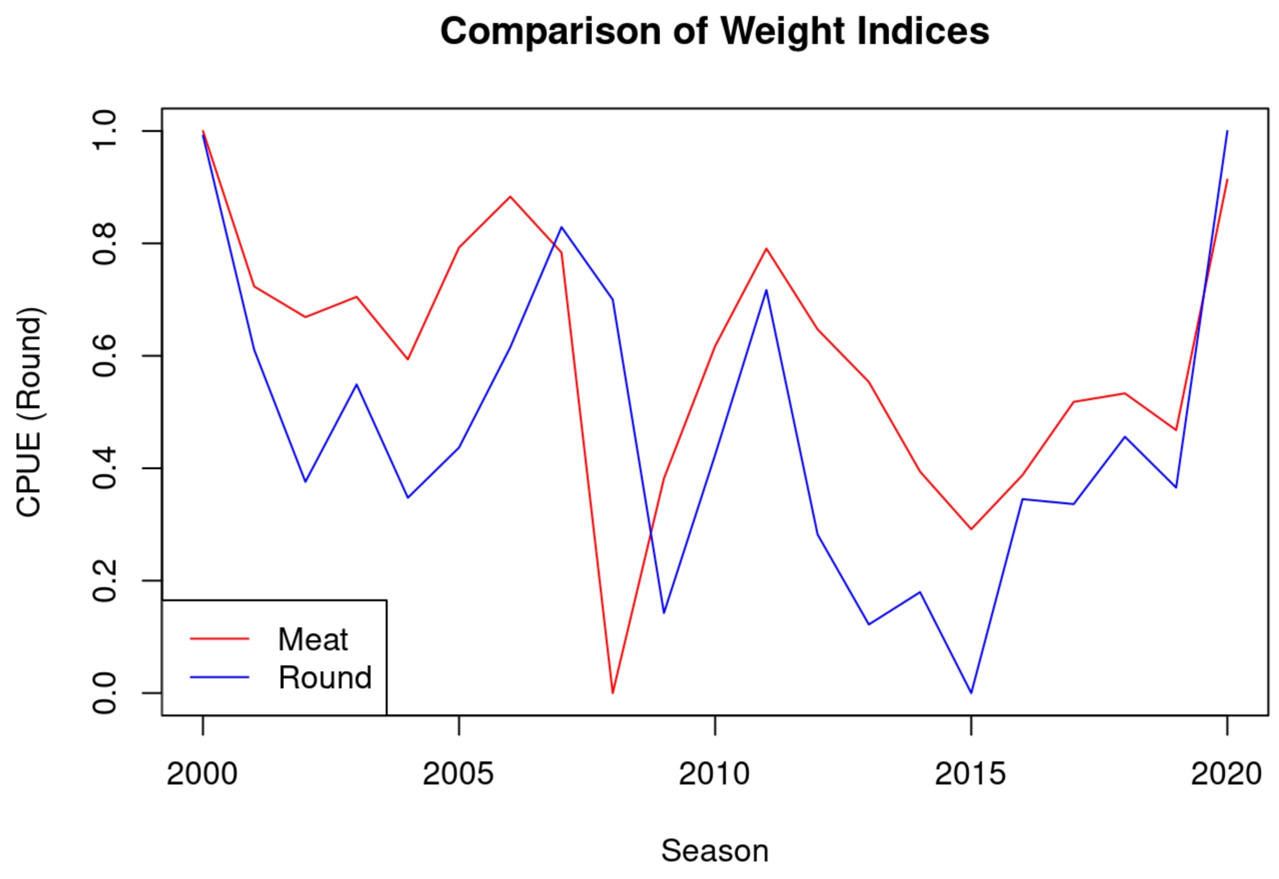
8. Plot a comparison of the round weight indices from models with and without a region effect.

```
plot.default(
    indices$Season,
    indices$round_season,
    type="l",
    col="red",
    xlab="Season",
    ylab="CPUE (Round)",
    main="Comparison of Round Weight Indices"
)
lines(
    indices$Season,
    indices$round_season_region,
    col="blue"
)
legend("bottomleft", legend=c("Season Only", "Season and Region"), col=c("red", "blue"), lty=1:1)
```



9. Please plot a comparison of fishery-dependent indices of abundance for scallops based on round and meat weights, using models that include the region effect. Summarize the differences you observe.

```
plot.default(
    indices$Season,
    indices$meat_season_region,
    type="l",
    col="red",
    xlab="Season",
    ylab="CPUE (Round)",
    main="Comparison of Weight Indices"
)
lines(
    indices$Season,
    indices$round_season_region,
    col="blue"
)
legend("bottomleft", legend=c("Meat", "Round"), col=c("red", "blue"), lty=1:1)
```



- It’s interesting to note that there seems to be a kind of delay betwen the meat and round with the meat usually responding first. I wonder if this indicates some kind of selectivity on the fisher’s part.
- One thing apparent from the plots above is that controlling for region for meat doesn’t make a huge difference whereas there definitely is a difference in round weight. Very curious as to why this is.
- Also interesting to note that generally the meat index remains at a higher level overall except for that one exceptional dip somewhere around 2008.

Time Allocation

Probably ~4 hours. That first problem really threw me for a loop :)