Stat 344 Final

Trey Tipton

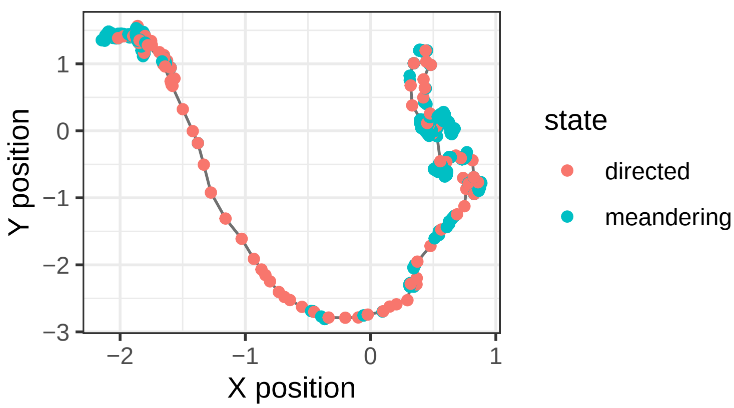
April 27, 2022

0. Academic Honesty Statement

I, Trey Tipton, affirm on my honor that this submission contains my own work and I did not get help from, or give help to, another person.

1. Circular Oxen

```
ox <- read.csv('https://sldr.netlify.app/data/ox.csv') |>
  mutate(state = factor(state)) |>
  na.omit()

gf_path(ys ~ xs, data = ox, color = 'grey44') |>
  gf_point(color = ~state) |>
  gf_labs(x = 'X position', y = 'Y position')
```



A. von Mises MLE

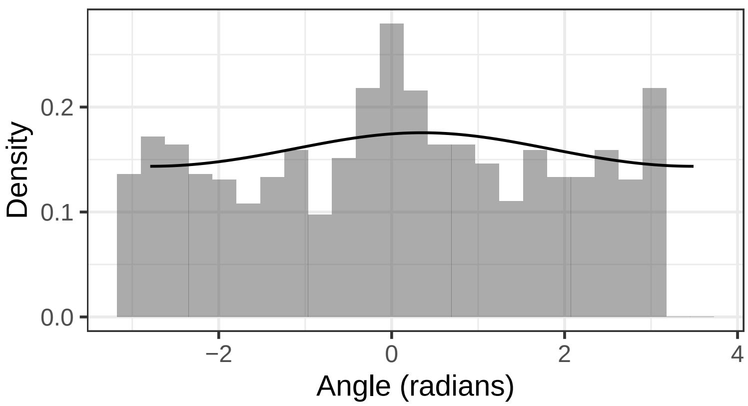
```
ll_vm <- function(theta, x){
  mu <- theta[1]
  kappa <- theta[2]
  if (kappa < 0) return(NA)
  dvonmises(x, mu = mu, kappa = kappa, log = TRUE)
}

maxLik(ll_vm, start = c(mu = 1, kappa = .1), x = ox$angle)
```

```
## Maximum Likelihood estimation
## Newton-Raphson maximisation, 4 iterations
## Return code 8: successive function values within relative tolerance limit (reltol)
## Log-Likelihood: -2589.687 (2 free parameter(s))
## Estimate(s): 0.349443 0.1005169
```

Using maximum-likelihood estimation to fit a von Mises distribution to the angle data, the fitted parameter estimates are as follows:

$\mu = 0.349443, \kappa = 0.1005169$.

```
gf_dhistogram(~angle, data = ox) %>%
  gf_labs(x = 'Angle (radians)', y = 'Density') %>%
  gf_dist('vonmises', mu = 0.349443, kappa = 0.1005169)
```



B. von Mises - uniform

```
maxLik(ll_vm, start = c(mu = -1, kappa = 0), fixed = 2, x = ox$angle)
```

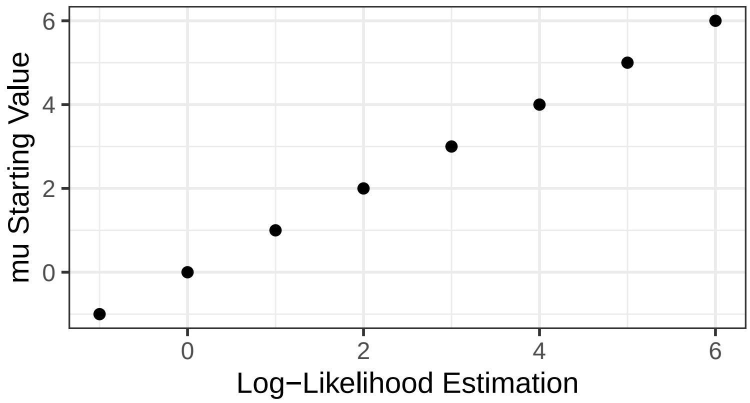
```
## Maximum Likelihood estimation
## Newton-Raphson maximisation, 1 iterations
## Return code 1: gradient close to zero (gradtol)
## Log-Likelihood: -2593.245 (1 free parameter(s))
## Estimate(s): -1 0
```

```
ll_unif <- function(theta, x){
  mu <- theta[1]
  dvonmises(x, mu = mu, kappa = 0, log = TRUE)
}

maxLik(ll_unif, start = c(mu = 10), x = ox$angle)
```

```
## Maximum Likelihood estimation
## Newton-Raphson maximisation, 1 iterations
## Return code 1: gradient close to zero (gradtol)
## Log-Likelihood: -2593.245 (1 free parameter(s))
## Estimate(s): 10
```

```
mu_starting_val <- c(-1, 0, 1, 2, 3, 4, 5, 6)
ll_estimation <- c(-1, 0, 1, 2, 3, 4, 5, 6)
```

```r
gf_point(mu_starting_val ~ ll_estimation)%>%
  gf_labs(x = "Log-Likelihood Estimation", y = "mu Starting Value")
```



It appears that when κ is zero, the log-likelihood estimation for μ is just the starting value for μ. This is likely because when κ is zero, the distribution is uniform (perfectly circular), meaning that the mean (μ) is fixed based on what starting value you use, because the maxLik function cannot optimize for μ when the data is circular.

C. von Mises test

Hypotheses:

H_0: $\Omega = \Omega_0$, κ is equal to zero and the distribution of the oxen movement angles is therefore uniform.

H_a: $\Omega = \Omega_1$, κ is greater than zero and the oxen movement angles fit better to some sort of von Mises distribution.

where $\theta = <\mu, \kappa>$,

$\Omega_0 = \{\theta \mid \kappa = 0\}$, and

$\Omega_1 = \{\theta \mid \kappa > 0\}\}$

Test Stat:

Using the maximum likelihood estimations from the previous question, we got two values, one from the von Mises MLE and one from the uniform MLE (von Mises but with κ fixed at zero):

von Mises Log-Likelihood: -2589.687

Uniform Log-Likelihood: -2593.245

We use those for our W test statistic:

$W = 2 * (l(\Omega_0) - l(\Omega_1))$

```r
w <- 2*(-2589.687  - -2593.245 )
1 - pchisq(w, 1)
```

```
## [1] 0.007639898
```

With a low p-value of 0.00764, we reject the null hypothesis that a uniform distribution generates the data and that κ is zero. Therefore, it is likely that κ is greater than zero and a von Mises distribution fits the data better, since we do not have enough evidence to say that $\kappa = 0$.

D. Goodness!

Hypotheses:

$H_0 : \Omega_0 = \{\theta | \kappa > 0\}$, where $\theta = <\mu, \kappa>$, A von Mises distribution generates the angle data for the ox data set.

H_a : A von Mises distribution does not generate the angle data for the ox data set.

Test Statistic:

$G = -2 * log(\lambda) = 2 * \sum o_i * log(\frac{o_i}{e_i})$

```
min(ox$angle)
```

```
## [1] -3.140357
```

```
max(ox$angle)
```

```
## [1] 3.141593
```

```
angle_breaks <- c(-pi+.000001, -(3*pi)/4, -pi/2, -pi/4, 0, pi/4, pi/2, (3*pi)/4, pi)
ox <- ox %>%
  mutate(binned_angles = cut(angle, breaks = angle_breaks))

tally(~binned_angles, data = ox)
```

```
## binned_angles
##  (-3.14,-2.36]  (-2.36,-1.57] (-1.57,-0.785]     (-0.785,0]      (0,0.785]
##            169            158            138            222            223
##   (0.785,1.57]    (1.57,2.36]    (2.36,3.14]
##            145            175            181
```

```
count_dat <- data.frame(tally(~binned_angles, data = ox))
count_dat
```

```
##    binned_angles Freq
## 1  (-3.14,-2.36]  169
## 2  (-2.36,-1.57]  158
## 3 (-1.57,-0.785]  138
## 4     (-0.785,0]  222
## 5      (0,0.785]  223
## 6   (0.785,1.57]  145
## 7    (1.57,2.36]  175
## 8    (2.36,3.14]  181
```

```
count_dat <- count_dat %>%
  mutate(probs = diff(pvonmises(angle_breaks, mu = 0.349443, kappa = 0.1005169,
                                from = pi+.000000001)), e = sum(count_dat$Freq) * probs)
count_dat$e
```

```
## [1] 159.5294 164.6831 176.7215 189.1231 194.0184 187.9976 175.2307 163.6960
```

```
G_angle <- 2 * sum( count_dat$Freq * log( count_dat$Freq / count_dat$e))
G_angle
```

```
## [1] 31.99943
```

```
1 - pchisq(G_angle, df = nrow(count_dat) - 1 - 2)
```

```
## [1] 5.942794e-06
```

```r
pearson_angle <- sum(((count_dat$Freq - count_dat$e)^2) / count_dat$e)

1 - pchisq(pearson_angle, df = 2)
```

```
## [1] 1.831628e-07
```

Both of the p-values we computed from the Likelihood Ratio Test Statistic and the Pearson chi-square test stat are very low at 5.942794e-06 and 1.831628e-07, therefore we reject the null hypothesis that a von Mises distribution is a good fit to the data. This means that although it is a better fit than a uniform distribution, we do not have enough evidence to say that the angle data are formed by a von Mises distribution.

2. More Oxen

A. Regression - Fit

```r
ox_lm <- glm(state ~ altitude + time, data = ox, family = binomial(link = 'logit'))
msummary(ox_lm)
```

```
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.36518    0.17224   7.926 2.27e-15 ***
## altitude     0.00825    0.00117   7.048 1.81e-12 ***
## timenight    0.82901    0.26471   3.132  0.00174 **
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 664.15  on 1410  degrees of freedom
## Residual deviance: 597.05  on 1408  degrees of freedom
## AIC: 603.05
##
## Number of Fisher Scoring iterations: 6
```

Model Equation:

$$logit(p_{meandering}) = log\left(\frac{p_{meandering}}{(1 - p_{meandering})}\right) = 1.36518 + 0.00825x_{altitude} + 0.82901x_{time}$$

$$y_{state} \sim \textsf{Binom}(1, p_{meandering})$$

where $p_{meandering}$ is the probability that an ox is meandering, and $x_{time} = 2$ if it is nighttime and 1 if daytime.

B. Ox at night

```r
p <- 1/(1 + exp(-(1.36518 + 0.82901*2)))
p
```

```
## [1] 0.9536113
```

The probability that an ox would be meandering when at sea level ($x_{altitude} = 0$) and at midnight ($x_{timenight} = 2$) is 0.9536113.

C. Ox at night again

```
odds_night <- 0.9536113/(1 - 0.9536113)
odds_night
```

```
## [1] 20.55697
```

```
p_day <- 1/(1 + exp(-(1.36518 + 0.82901)))
odds_day <- p_day/(1-p_day)
p_day
```

```
## [1] 0.8997266
```

```
odds_day
```

```
## [1] 8.97273
```

```
st.err <- sqrt(diag(vcov(ox_lm)))
st.err
```

```
## (Intercept)    altitude   timenight
## 0.172243372 0.001170412 0.264705334
```

```
odds <- -(odds_night - odds_day)

odds
```

```
## [1] -11.58424
```

```
odds + c(-1, 1)*st.err[3]*qnorm(0.975)
```

```
## [1] -12.10306 -11.06543
```

The odds of meandering for the sea-level ox decreases by about 11.58424 as time advances from midnight to daytime. From this we can create a confidence interval: we are 95% confidence that the change in odds of meandering at sea-level from midnight to daytime is contained in the interval (-12.10306, -11.06543).

3.Yet more oxen

A. Fit a model

```
ox <- ox %>%
  mutate(time2 = ifelse(time=="day",1,0))


ll_mod <- function(theta, x){
  beta0 <- theta[1]
  beta1 <- theta[2]
  beta2 <- theta[3]
  beta3 <- theta[4]
  beta4 <- theta[5]
  sigma <- theta[6]
  resids <- ox$step - (beta0 + beta1*ox$time2 + beta2*ox$temp + beta3*ox$altitude + beta4*ox$xs)
  if (sigma < 0) return(NA)
  dnorm(resids, mean = 0, sd = sigma, log = TRUE)
}
```

```
maxLik(logLik = ll_mod, start = c(beta0 = 1, beta1 = 1, beta2 = -1, beta3 = 0, beta4 = -1, sigma = 1), 
```

```
## Maximum Likelihood estimation
## Newton-Raphson maximisation, 46 iterations
## Return code 8: successive function values within relative tolerance limit (reltol)
## Log-Likelihood: -78591044 (6 free parameter(s))
## Estimate(s): 144.2272 64.61116 -6.31048 -0.7636456 -41.22386 0.9229477
```

$$y_{step} = 144.2272 + 64.61116x_{time} - 6.31048x_{temp} - 0.7636456x_{altitude} + 41.22386x_{xs} + \epsilon$$

$$\epsilon \sim N(0, 0.9229477)$$

where y_{step} is ox's step length and x_{time} is zero if it is night, and 1 if it is day.

B. Select

```
ox_model <- lm(step ~ time2 + temp + altitude + xs, data = ox)
msummary(ox_model)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 144.1810    24.3082   5.931 3.78e-09 ***
## time2        54.4010    16.8966   3.220 0.001313 **
## temp         -6.8724     1.2666  -5.426 6.78e-08 ***
## altitude     -0.3951     0.1231  -3.208 0.001366 **
## xs          -44.6570    13.1955  -3.384 0.000733 ***
##
## Residual standard error: 300.6 on 1406 degrees of freedom
## Multiple R-squared:  0.06172,    Adjusted R-squared:  0.05905
## F-statistic: 23.12 on 4 and 1406 DF,  p-value: < 2.2e-16
```

Backwards Step-wise Selection: we will do an Anova test on the full model to see which predictor produces the largest large p-value, and continue doing so until all p-values are small.

```
car::Anova(ox_model)
```

```
## Anova Table (Type II tests)
##
## Response: step
##               Sum Sq   Df F value    Pr(>F)
## time2         936788    1  10.366 0.0013129 **
## temp         2660658    1  29.442 6.778e-08 ***
## altitude      930095    1  10.292 0.0013662 **
## xs           1035040    1  11.453 0.0007333 ***
## Residuals  127061446 1406
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the full model, there is no large p-values, so each predictor plays a significant role in the model. The best model includes time, temp, altitude, and xs as predictors for step length.

C. Predict

```
new_data <- data.frame(altitude = 0, time2 = 0, xs = .5, temp = -10)
conf_int <- predict(ox_model, newdata = new_data,
        interval = 'prediction',
        level = 0.95)
conf_int
```

```
##        fit       lwr     upr
## 1 190.5765 -402.2231 783.376
```

A 95% prediction interval interval for the distance (meters) moved in the next hour for an ox at sea level, at midnight, when it was -10 degrees, and at an xs position of 0.5 is (-402.2231 , 783.376). Since we know that the ox can only move a positive distance, we can predict with 95% confidence that the ox will move between 0 and 783.376 meters in the next hour.