# Statistics 143 — Spring 2024 — Assignment 4

Due Monday April 8, 2024

Homework is to be uploaded on Gradescope by 10:00pm on Monday evening.

Please make sure on your assignment you indicate clearly other students with whom you collaborated, as well as any assistance you received from generative AI tools.

## Written assignment

1. Suppose $n$ athletes compete in a race, and that the outcome is a rank ordering from best to worst. Let $X_i$, for $i = 1, \ldots, n$ be the latent performance of competitor $i$, and assume that the cumulative distribution function of $X_i$ is given by

$$F_{X_i}(x) = \Pr(X_i \leq x) = \exp(-\exp(-(x-\theta_i))),$$

that is, a Gumbel distribution with location parameter $\theta_i$. Assume the $X_i$ are independently distributed.

(a) For $i = 1, \ldots, n$, let $Y_i = \exp(-X_i)$. Show that the distribution of $Y_i$ is exponential, that is,

$$F_{Y_i}(y) = \Pr(Y_i \leq y) = 1 - \exp(-\lambda_i y),$$

where $\lambda_i = \exp(\theta_i)$.

Under the latent variable model, the winner of the race is the athlete with the highest realized value of $X_i$. Argue that the winner, therefore, is the athlete with the smallest value of $Y_i$.

Solution:

$$\begin{aligned} F_{Y_i}(y) &= \Pr(Y_i \leq y) = \Pr(\exp(-X_i) \leq y) \\ &= \Pr(-X_i \leq \log y) \\ &= \Pr(X_i \geq -\log y) \\ &= 1 - \Pr(X_i \leq -\log y) \\ &= 1 - \exp(-\exp(-(-\log y - \theta_i))) \\ &= 1 - \exp(-\exp(\log y + \theta_i)) \\ &= 1 - \exp(-y\exp(\theta_i)) \\ &= 1 - \exp(-\lambda_i y) \end{aligned}$$

where $\lambda_i = \exp(\theta_i)$.

Since $Y_i$ is a monotonically decreasing function of $X_i$, the athlete with the highest value of $X_i$ has the smallest value of $Y_i$.

(b) Suppose you want to determine the probability athlete 1 wins the race. Show that this can be answered by solving the integral

$$\Pr(\text{Athlete 1 wins}) = \lambda_1\lambda_2\cdots\lambda_n \int_0^\infty \int_{y_1}^\infty \cdots \int_{y_1}^\infty \exp(-(\lambda_1 y_1 + \lambda_2 y_2 + \cdots + \lambda_n y_n))\, dy_n dy_{n-1} \cdots dy_2 dy_1.$$

*For this part of the problem, please do not actually solve the integral. That will be addressed in the subsequent parts of this problem.*

Solution:

$$f_{Y_i}(y) = \frac{\partial F_{Y_i}(y)}{\partial y} = \lambda_i \exp(-\lambda_i y)$$

$$\begin{aligned}
\Pr(\text{Athlete 1 wins}) &= \Pr(Y_2 \geq Y_1, Y_3 \geq Y_1, \ldots, Y_n \geq Y_1) \\
&= \int_0^\infty \Pr(Y_2 \geq Y_1, Y_3 \geq Y_1, \ldots, Y_n \geq Y_1 | Y_1 = y_1) f_{Y_1}(y_1) dy_1 \\
&= \int_0^\infty \left( \prod_{i=2}^n \Pr(Y_i \geq Y_1 | Y_1 = y_1) \right) f_{Y_1}(y_1) dy_1 \\
&= \int_0^\infty \left( \prod_{i=2}^n \Pr(Y_i \geq y_1) \right) f_{Y_1}(y_1) dy_1 \\
&= \int_0^\infty \left( \prod_{i=2}^n \int_{y_1}^\infty \lambda_i \exp(-\lambda_i y_i) dy_i \right) \lambda_1 \exp(-\lambda_1 y_1) dy_1 \\
&= \int_0^\infty \left( \int_{y_1}^\infty \cdots \int_{y_1}^\infty \lambda_2 \cdots \lambda_n \exp(-(\lambda_2 y_2 + \lambda_n y_n)) dy_n \cdots dy_2 \right) \lambda_1 \exp(-\lambda_1 y_1) dy_1 \\
&= \lambda_1 \lambda_2 \cdots \lambda_n \int_0^\infty \int_{y_1}^\infty \cdots \int_{y_1}^\infty \exp(-(\lambda_1 y_1 + \lambda_2 y_2 + \cdots + \lambda_n y_n)) dy_n \cdots dy_2 dy_1
\end{aligned}$$

(c) Show that, by evaluating the $n-1$ inner integrals in part (b), the integral expression reduces to

$$\Pr(\text{Athlete 1 wins}) = \lambda_1 \int_0^\infty \left( \prod_{i=2}^n \exp(-\lambda_i y_1) \right) \exp(-\lambda_1 y_1) dy_1.$$

Solution:

$$\int_{y_1}^\infty \lambda_i \exp(-\lambda_i y_i) dy_i = [-\exp(-\lambda_i y_i)]_{y_1}^\infty = 0 - (-\exp(-\lambda_i y_1)) = \exp(-\lambda_i y_1)$$

$$\begin{aligned}
\Pr(\text{Athlete 1 wins}) &= \int_0^\infty \left( \prod_{i=2}^n \int_{y_1}^\infty \lambda_i \exp(-\lambda_i y_i) dy_i \right) \lambda_1 \exp(-\lambda_1 y_1) dy_1 \\
&= \lambda_1 \int_0^\infty \left( \prod_{i=2}^n \exp(-\lambda_i y_1) \right) \exp(-\lambda_1 y_1) dy_1
\end{aligned}$$

(d) Now show that the integral in part (c) simplifies to

$$\Pr(\text{Athlete 1 wins}) = \frac{\lambda_1}{\lambda_1 + \lambda_2 + \cdots + \lambda_n}.$$

What is this probability in terms of the original $\theta_i$?

Solution:

$$\begin{aligned}
\Pr(\text{Athlete 1 wins}) &= \lambda_1 \int_0^\infty \left(\prod_{i=2}^n \exp(-\lambda_i y_1)\right) \exp(-\lambda_1 y_1) dy_1 \\
&= \lambda_1 \int_0^\infty \exp(-(\sum_{i=1}^n \lambda_i) y_1) dy_1 \\
&= -\frac{\lambda_1}{\sum_{i=1}^n \lambda_i} \left[\exp(-(\sum_{i=1}^n \lambda_i) y_1)\right]_0^\infty \\
&= -\frac{\lambda_1}{\sum_{i=1}^n \lambda_i}(0-1) \\
&= \frac{\lambda_1}{\lambda_1 + \lambda_2 + \cdots + \lambda_n}
\end{aligned}$$

Substitute $\lambda_i = \exp(\theta_i)$ into the expression, we get

$$\Pr(\text{Athlete 1 wins}) = \frac{\exp(\theta_1)}{\exp(\theta_1) + \exp(\theta_2) + \cdots + \exp(\theta_n)}$$

2. For this problem, you will examine the results of competitive fishing tournaments in the Bassmaster Elite series. Competitive fishing essentially involves participants (anglers) who attempt to maximize the total weight of the fish they catch, typically limited to five fish each day of the competition.

   The file `fishing_2122.csv` in the Data Sets folder on the course Canvas site contains information on 20 competitive anglers based on the first four Elite Bassmaster tournaments of 2020, and all Elite Bassmaster tournaments from 2021 and 2022, totaling 22 tournaments. The following variables are included in this data set.

   `tournament.id`: Numerical ID of the tournament

   `rank`: Rank in the tournament (beyond the 20 anglers in the data set)

   `name`: Angler's name

   `weights`: Total weight of fish caught in the tournament

   `player.id`: Numerical ID of the angler

   The file `fishing_2122_ranking.csv` contains comma-separated rank orderings for these 22 tournaments, with the entries corresponding to the player IDs in the previous file.

   (a) For these data, fit a least-squares model of the total weight caught by the anglers, with separate strength parameters per angler and separate additive parameters for each tournament. That is, fit a model that assumes $y_{ij}$ the total weight of fish caught by angler $j$ ($j = 1, \ldots, 20$) during tournament $i$ ($i = 1, \ldots, 22$) has a mean

$$\mathrm{E}(y_{ij}) = \beta_i + \theta_j$$

where $\theta_j$ is the strength parameter for angler $j$, and $\beta_i$ is the effect of playing in tournament $i$.

i. Why does it make sense to have separate parameters $\beta_i$ for each tournament?

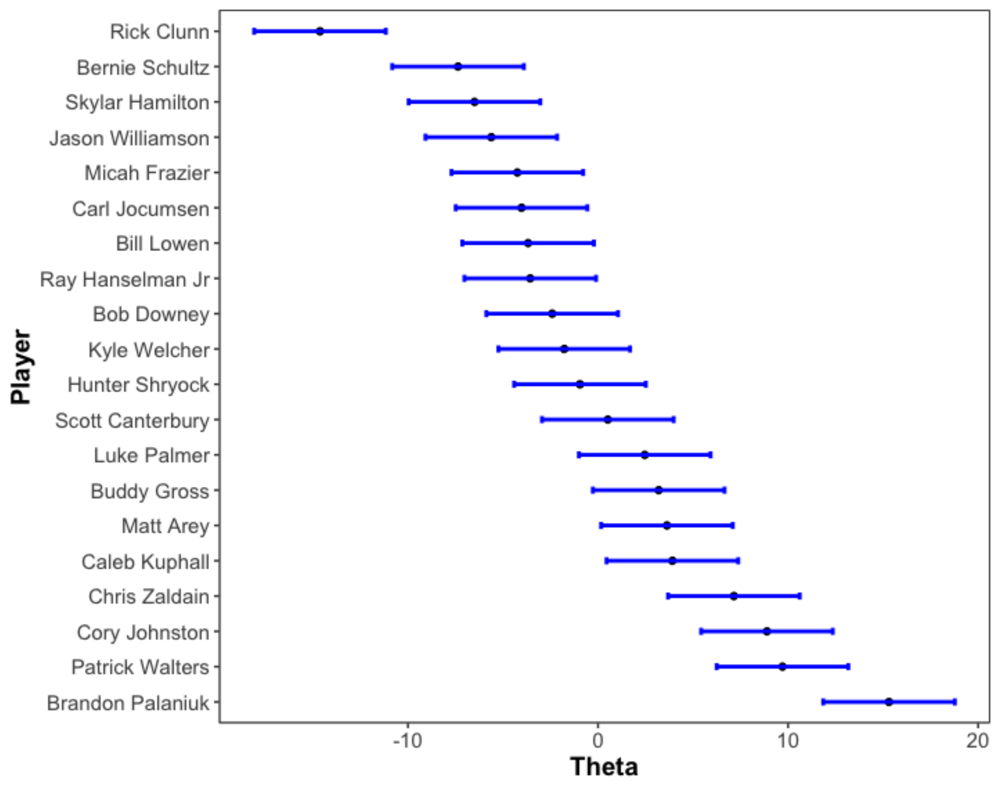
Solution:
Because the conditions of each tournament may be different from each other due to environmental conditions such as temperature, humidity, wind, etc.

ii. Based on the fit of the model, display the rank order the anglers from best to worst according to their estimated strength parameters, and display the estimates and their 1-SE bars in a graph (as in the lecture notes slide). Who are the top three anglers according to the estimated strength parameters?

Solution:

```
> library(ggplot2)
> df.score = read.csv("fishing_2122.csv")
> df.score$player.id = df.score$player.id + 1
>
> Games = unique(df.score$tournament.id)
> Players.df = unique(df.score[,c("name","player.id")])
> Players.df = Players.df[order(Players.df$player.id),]
> Players = Players.df$player.id  # unique player IDs
> Players.names = Players.df$name  # names attached to players
> K=length(Players)
>
> # Construct design matrix
> X.theta = outer(df.score$player.id, Players, "==")+0
> X.beta = outer(df.score$tournament.id, Games, "==")+0
> W = rbind(diag(K-1), rep(-1, K-1))
> X.theta.star = X.theta %*% W
> df.score$X.theta.star = X.theta.star
>
> # Fit Gaussian model
> ability.normal = lm(weights~X.theta.star+X.beta+0, data = df.score)
> Players.ability = data.frame("Normal.ability.est"=
           W %*% ability.normal$coefficients[1:(length(Players)-1)])
> V.normal = W %*% vcov(ability.normal)[1:(length(Players)-1),
           1:(length(Players)-1)] %*% t(W)
> Players.ability["Normal.ability.stderr"] = sqrt(diag(V.normal))
> row.names(Players.ability) = Players.names
> dimnames(V.normal) = list(Players.names, Players.names)
>
> Players.ability[order(Players.ability$Normal.ability.est,
                 decreasing=T),]
                 Normal.ability.est Normal.ability.stderr
Brandon Palaniuk         15.3110795               3.46297
Patrick Walters           9.7059659               3.46297
Cory Johnston             8.8877841               3.46297
Chris Zaldain             7.1491477               3.46297
```

```
Caleb Kuphall                3.9133523              3.46297
Matt Arey                    3.6264205              3.46297
Buddy Gross                  3.1917614              3.46297
Luke Palmer                  2.4588068              3.46297
Scott Canterbury             0.5156250              3.46297
Hunter Shryock              -0.9502841              3.46297
Kyle Welcher                -1.7769886              3.46297
Bob Downey                  -2.4105114              3.46297
Ray Hanselman Jr            -3.5667614              3.46297
Bill Lowen                  -3.6775568              3.46297
Carl Jocumsen               -4.0241477              3.46297
Micah Frazier               -4.2457386              3.46297
Jason Williamson            -5.6150568              3.46297
Skylar Hamilton             -6.4985795              3.46297
Bernie Schultz              -7.3650568              3.46297
Rick Clunn                 -14.6292614              3.46297
>
> # Plot the estimates and their 1-SE bars
> oo = order(Players.ability$Normal.ability.est,decreasing=T)
> Players.ability.ordered = round(Players.ability[oo,1:2],3)
> Players.ability.ordered$Upper =
+   Players.ability.ordered[,1]+Players.ability.ordered[,2]
> Players.ability.ordered$Lower =
+   Players.ability.ordered[,1]-Players.ability.ordered[,2]
> Players.ability.ordered$Player =
+   factor(row.names(Players.ability.ordered),
+          levels = row.names(Players.ability.ordered))
> ggplot(Players.ability.ordered,
+        aes(x = Normal.ability.est, y = Player)) +
+   geom_point() +
+   geom_errorbarh(aes(xmin = Lower, xmax = Upper),
+                  height = 0.2, color = "blue", size = 1) +
+   theme_bw() +
+   labs(x = "Theta", y = "Player")  +
+   theme(text = element_text(size = 14),
+         axis.title = element_text(size = 14, face = "bold"),
+         plot.title = element_text(hjust = 0.5),
+         panel.grid.major = element_blank(),
+         panel.grid.minor = element_blank(),
+         panel.background = element_rect(fill = "white",
+                                  colour = "black"),
+         legend.position = "bottom")
```



The top three anglers are Brandon Palaniuk, Patrick Walters and Cory Johnston.

iii. Construct a 95% confidence interval for the difference in strengths between Matt Arey and Bob Downey. What do you make of the confidence interval?

Solution:

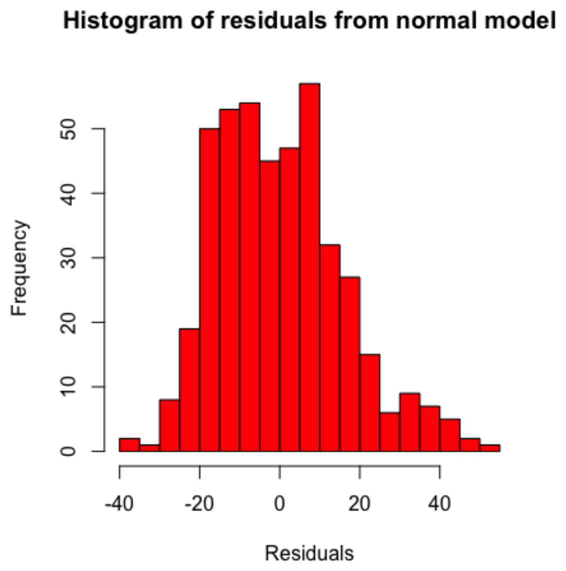
```
> diff.Matt.Bob = Players.ability["Matt Arey", "Normal.ability.est"] -
+   Players.ability["Bob Downey", "Normal.ability.est"]
> stderr.Matt.Bob = sqrt(
+   Players.ability["Matt Arey", "Normal.ability.stderr"]^2 +
+     Players.ability["Bob Downey", "Normal.ability.stderr"]^2 -
+     2*V.normal["Matt Arey", "Bob Downey"])
> CI.Matt.Bob = c(diff.Matt.Bob-1.96*stderr.Matt.Bob,
+               diff.Matt.Bob+1.96*stderr.Matt.Bob)
> CI.Matt.Bob
[1] -3.811295 15.885158
```

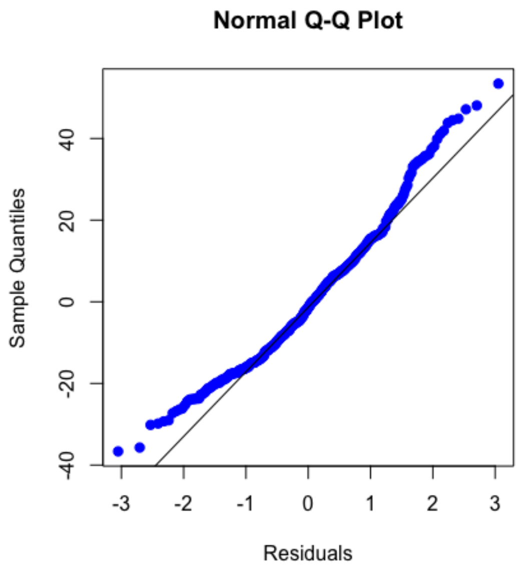
The 95% confidence interval covers 0. Therefore, there is no strong evidence at the 95% confidence level that Matt is a stronger or weaker player than Bob.

iv. Create a plot of the residuals from the model, and a normal probability plot of the residuals. Do you believe that there are any concerns about the normality of the residuals from the model fit?

Solution:

```
> par(mfrow=c(1,2))
> hist(residuals(ability.normal),
+      main = "Histogram of residuals from normal model",
+       xlab = "Residuals", col="red",
+       breaks = 20)
> qqnorm(residuals(ability.normal), pch=19,
+          xlab = "Residuals", col="blue")
> qqline(residuals(ability.normal))
```





The distribution of residuals is slightly right-skewed, which is evidenced from both the histogram and the normal quantile plot. The model assumes that the distribution should be approximately normal, so we might have reason to be skeptical that the model assumptions are met. A follow-up might be to transform the observed weights by, for example, taking the log of the weights, and then refitting the model.

(b) Now fit a Plackett-Luce model to these data based on the rankings, using the `PLMIX` library, and fitting the model using the Gibbs sampler (a version of Monte Carlo Markov chain simulation). Set the procedure to have a burn-in period of 1000 iterations, and run for a total of 5000 iterations.

i. Rank order the Plackett-Luce estimated parameters. According to the model fit, who are the top 3 anglers among the 20?

Solution:

```
> library(PLMIX)
> set.seed(2024) # set seed for reproducibility
> df.order = read.csv("fishing_2122_ranking.csv")
> colnames(df.order) <- c('order')
>
> # Construct ordering matrix to fit PL model
> constr.ordering.matrix <- function(df, k=K){
+   l = as.integer(c(strsplit(df['order'], ",")[[1]]))+1
+   return (c(l, rep(0, k-length(l))))
+ }
> order = t(apply(df.order, MARGIN=1, FUN=constr.ordering.matrix))
> order = as.matrix(order)
> ability.PL.MCMC = gibbsPLMIX(as.top_ordering(order,
+                   format_input = "ordering", aggr=FALSE),
+                   K=K, G=1, n_burn=1000, n_iter=5000)
> Players.ability["PlackettLuce.est"] =
+   colMeans(log(ability.PL.MCMC$P))
> Players.ability["PlackettLuce.stderr"] =
+   apply(log(ability.PL.MCMC$P), 2, sd)
> Players.ability[order(Players.ability$PlackettLuce.est,decreasing=T),]
                 Normal.ability.est Normal.ability.stderr
Brandon Palaniuk         15.3110795               3.46297
Patrick Walters           9.7059659               3.46297
Cory Johnston             8.8877841               3.46297
Matt Arey                 3.6264205               3.46297
Buddy Gross               3.1917614               3.46297
Chris Zaldain             7.1491477               3.46297
Luke Palmer               2.4588068               3.46297
Hunter Shryock           -0.9502841               3.46297
Caleb Kuphall             3.9133523               3.46297
Scott Canterbury          0.5156250               3.46297
Carl Jocumsen            -4.0241477               3.46297
Jason Williamson         -5.6150568               3.46297
Bob Downey               -2.4105114               3.46297
Ray Hanselman Jr         -3.5667614               3.46297
Bill Lowen               -3.6775568               3.46297
Kyle Welcher             -1.7769886               3.46297
Skylar Hamilton          -6.4985795               3.46297
Micah Frazier            -4.2457386               3.46297
Bernie Schultz           -7.3650568               3.46297
Rick Clunn              -14.6292614               3.46297
                 PlackettLuce.est PlackettLuce.stderr
Brandon Palaniuk        -2.317960           0.2186399
Patrick Walters         -2.518296           0.2174240
Cory Johnston           -2.638674           0.2161035
Matt Arey               -2.649926           0.2151540
Buddy Gross             -2.696984           0.2114441
Chris Zaldain           -2.886580           0.2158963
Luke Palmer             -2.913517           0.2164791
Hunter Shryock          -2.915382           0.2112400
Caleb Kuphall           -2.954358           0.2209328
Scott Canterbury        -3.036502           0.2303308
Carl Jocumsen           -3.208535           0.2259877
Jason Williamson        -3.273843           0.2224248
Bob Downey              -3.293477           0.2302964
Ray Hanselman Jr        -3.303453           0.2209428
Bill Lowen              -3.321997           0.2297058
Kyle Welcher            -3.408692           0.2388802
Skylar Hamilton         -3.438003           0.2246179
```

```
Micah Frazier            -3.449862           0.2390192
Bernie Schultz           -3.869927           0.2442770
Rick Clunn               -4.068835           0.2476073
```

The top three anglers are Brandon Palaniuk, Patrick Walters and Cory Johnston, same as the results of the normal model.

ii. The `$P` component of the fitted model contains a matrix with 20 columns (corresponding to the individual anglers) and with 4000 rows (corresponding to saved MCMC iterations). Each element in `P` in column $j$ is a simulated value of $\exp(\theta_j)$. Determine the posterior mean probability that in a tournament involving Matt Arey, Luke Palmer, and Bob Downey, that the outcome would be Matt Arey coming in first, Luke Palmer coming in second, and Bob Downey coming in third. *Hint: Evaluate the Plackett-Luce probability per MCMC itermation, and then average these probabilities across all 4000 iterations.*

Solution:

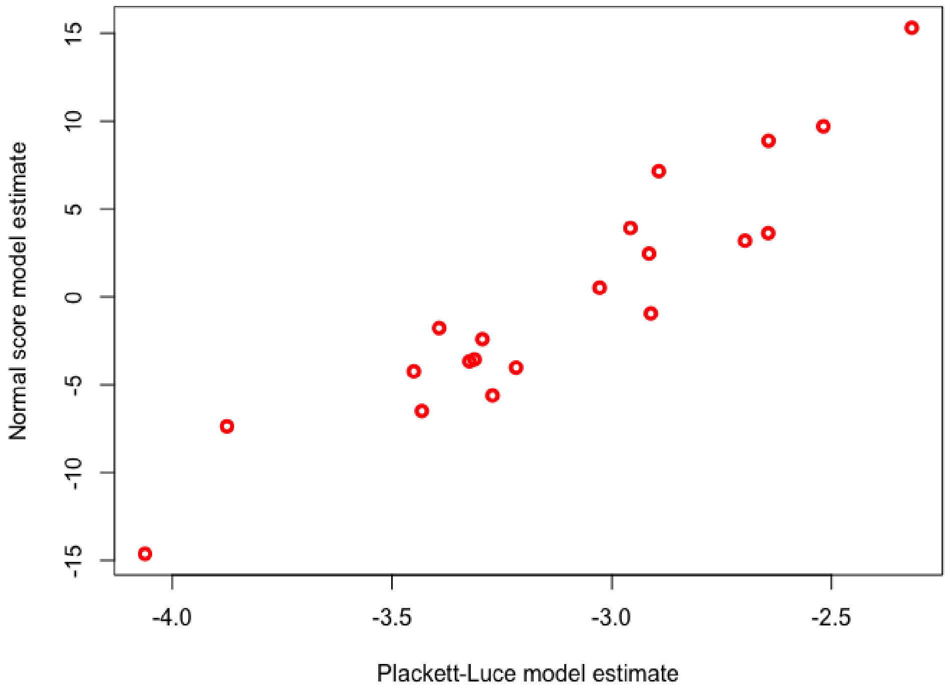
```
> matt = Players.df$player.id[Players.df$name == 'Matt Arey']
> luke = Players.df$player.id[Players.df$name == 'Luke Palmer']
> bob = Players.df$player.id[Players.df$name == 'Bob Downey']
>
> calc.prob = function(x) {
+   matt.first = x[matt]/(x[matt]+x[luke]+x[bob])
+   luke.second = x[luke]/(x[luke]+x[bob])
+   return (matt.first*luke.second)
+ }
>
> prob = mean(apply(ability.PL.MCMC$P,1,calc.prob))
> prob
[1] 0.2566229
```

The posterior mean probability of Matt Arey coming in first, Luke Palmer coming in second, and Bob Downey coming in third, is 0.26.

iii. Plot the estimated strengths from the normal model against the estimated strengths from the Plackett-Luce model. Interpret the correspondence between the two sets of model estimates.

Solution:

```
> plot(Players.ability$PlackettLuce.est,
+      Players.ability$Normal.ability.est,
+      col="red",
+      xlab="Plackett-Luce model estimate",
+      ylab="Normal score model estimate")
> cor(Players.ability$Normal.ability.est,
+     Players.ability$PlackettLuce.est)
[1] 0.9369692
```

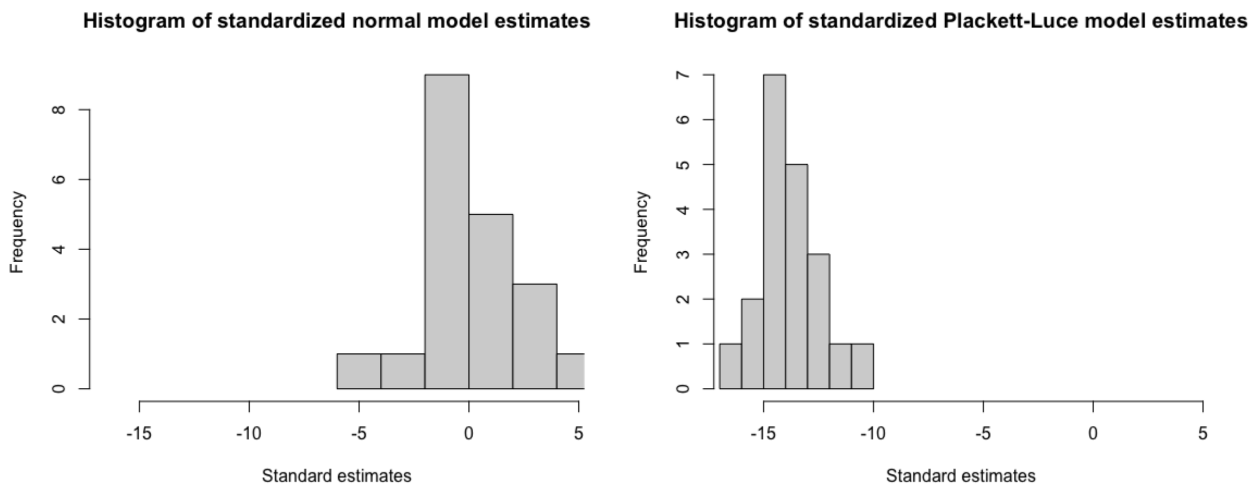


The plot of the estimated strengths from the normal model against the Plackett-Luce model appear to follow a straight line, with a correlation of 0.937.

iv. For the normal model and the Plackett-Luce models, compute standardized estimates of the strengths by dividing the estimated strengths by the corresponding standard errors. Create a histogram of the standardized estimates for the Plackett-Luce model and a histogram of the standardized estimates for the normal model. How do the spread of standardized strengths for the normal model compare to that of the Placett-Luce model? What do you make of the comparison?

Solution:

```
> Players.ability["Normal.ability.est.standardized"] =
+   Players.ability["Normal.ability.est"] /
+   Players.ability["Normal.ability.stderr"]
> Players.ability["PlackettLuce.est.standardized"] =
+   Players.ability["PlackettLuce.est"] /
+   Players.ability["PlackettLuce.stderr"]
> par(mfrow=c(1,2))
> hist(Players.ability$Normal.ability.est.standardized,
+      main = "Histogram of standardized normal model estimates",
+      xlab = "Standard estimates",
+      xlim=range(c(Players.ability$Normal.ability.est.standardized,
+                   Players.ability$PlackettLuce.est.standardized)))
> hist(Players.ability$PlackettLuce.est.standardized,
+      main = "Histogram of standardized Plackett-Luce model estimates",
+      xlab = "Standard estimates",
+      xlim=range(c(Players.ability$Normal.ability.est.standardized,
+                   Players.ability$PlackettLuce.est.standardized)))
> max(Players.ability$Normal.ability.est.standardized) -
+   min(Players.ability$Normal.ability.est.standardized)
[1] 8.645856
> max(Players.ability$PlackettLuce.est.standardized) -
+   min(Players.ability$PlackettLuce.est.standardized)
[1] 5.830893
```



The spread of standardized strengths for the normal model and the Placett-Luce model are 8.65 and 5.83, respectively. The normal model has a larger spread, as it retains more information of the competition, and thereby better differentiate between the strengths of individual players.

3. Suppose that you decide to model team strengths for game outcomes using a stochastic approach for the evolution of abilities over time. For team $i$, assume at period $t = 1$

$$\theta_{i1}|\sigma_1^2 \sim \mathrm{N}(0,\ \sigma_1^2),$$

where $\sigma_1^2$, the variance of team strengths during period 1, is known. Also assume for time period $t = 2, 3, \ldots$,

$$\theta_{it}|\theta_{i,t-1}, \rho, \tau^2 \sim \mathrm{N}(\rho\theta_{i,t-1},\ \tau^2)$$

where $\tau^2$ is a known innovation variance, and $\rho$ is a known autoregressive parameter. The expressions above show the explicit conditioning on the particular parameters. Usually the parameters $\sigma_1^2$, $\tau^2$ and $\rho$ are assumed unknown, but for this problem they will be treated as known.

(a) What is the distribution of $\theta_{i2}$ unconditional on $\theta_{i1}$, but conditional on $\sigma_1^2$, $\rho$ and $\tau^2$? *Hint: One way to approach this is to explicitly calculate*

$$\int_{-\infty}^{\infty} \varphi(\theta_{i2}|\rho\theta_{i1}, \tau^2)\varphi(\theta_{i1}|0, \sigma_1^2)d\theta_{i1}$$

*where $\varphi$ is a normal density function. Use the properties at the start of problem 1 in homework 3. You may also use the fact*

$$\varphi(\theta|\mu, \sigma^2) = c\varphi(c\theta|c\mu, c^2\sigma^2)$$

*for all real $c$ (this result can be assumed without proof).*

Solution:

By property $\varphi(\theta|\mu, \sigma^2) = c\varphi(c\theta|c\mu, c^2\sigma^2)$, we have

$$\varphi(\theta_{i1}|0, \sigma_1^2) = \rho\varphi(\rho\theta_{i1}|0, \rho^2\sigma_1^2),$$

combined with the result that $d(\rho\theta_{i1}) = \rho d\theta_{i1}$, we get

$$\begin{aligned}\int_{-\infty}^{\infty} \varphi(\theta_{i2}|\rho\theta_{i1}, \tau^2)\varphi(\theta_{i1}|0, \sigma_1^2)d\theta_{i1} &= \int_{-\infty}^{\infty} \varphi(\theta_{i2}|\rho\theta_{i1}, \tau^2)\rho\varphi(\rho\theta_{i1}|0, \rho^2\sigma_1^2)d\theta_{i1} \\ &= \int_{-\infty}^{\infty} \varphi(\theta_{i2}|\rho\theta_{i1}, \tau^2)\varphi(\rho\theta_{i1}|0, \rho^2\sigma_1^2)(\rho d\theta_{i1}) \\ &= \int_{-\infty}^{\infty} \varphi(\theta_{i2}|\rho\theta_{i1}, \tau^2)\varphi(\rho\theta_{i1}|0, \rho^2\sigma_1^2)d(\rho\theta_{i1})\end{aligned}$$

By property $\varphi(\theta|\mu_0, \sigma^2 + \sigma_0^2) = \int_{-\infty}^{\infty} \varphi(\theta|\mu, \sigma^2)\varphi(\mu|\mu_0, \sigma_0^2)d\mu$ from problem 1 in homework 3, we get

$$\int_{-\infty}^{\infty} \varphi(\theta_{i2}|\rho\theta_{i1}, \tau^2)\varphi(\rho\theta_{i1}|0, \rho^2\sigma_1^2)d(\rho\theta_{i1}) = \varphi(\theta_{i2}|0, \tau^2 + \rho^2\sigma_1^2)$$

Therefore, $\theta_{i2} \sim \mathrm{N}(0, \tau^2 + \rho^2\sigma_1^2)$ unconditional on $\theta_{i1}$.

(b) Based on part (a), what values of $\rho$ would ensure that the variance of $\theta_{i2}$ unconditional on $\theta_{i1}$ is greater or equal to the variance of $\theta_{i1}$?

Solution:

$$\begin{aligned}\tau^2 + \rho^2\sigma_1^2 &\geq \sigma_1^2 \\ (1-\rho^2)\sigma_1^2 &\leq \tau^2 \\ 1 - \rho^2 &\leq \frac{\tau^2}{\sigma_1^2} \\ \rho^2 &\geq 1 - \frac{\tau^2}{\sigma_1^2}\end{aligned}$$

If $\tau^2 \geq \sigma_1^2$, $\rho$ can take any value.
If $\tau^2 < \sigma_1^2$, $\rho \geq \sqrt{1 - \frac{\tau^2}{\sigma_1^2}}$ or $\rho \leq -\sqrt{1 - \frac{\tau^2}{\sigma_1^2}}$.

(c) For $t > 1$, obtain an expression for the variance of $\theta_{it}$ unconditional on $\theta_{i1}, \theta_{i2}, \ldots, \theta_{i,t-1}$, but conditional on the other model parameters ($\rho$, $\tau^2$, $\sigma_1^2$). *Hint: Use the probability identity $Var(Y) = E(Var(Y|X)) + Var(E(Y|X))$ recursively.*

Solution:

$$\begin{aligned}
\text{Var}(\theta_{it}) &= \text{E}(\text{Var}(\theta_{it}|\theta_{i,t-1})) + \text{Var}(\text{E}(\theta_{it}|\theta_{i,t-1})) \\
&= \text{E}(\tau^2) + \text{Var}(\rho\theta_{i,t-1}) \\
&= \tau^2 + \rho^2\text{Var}(\theta_{i,t-1}) \\
&= \tau^2 + \rho^2(\tau^2 + \rho^2\text{Var}(\theta_{i,t-2})) \\
&= (1+\rho^2)\tau^2 + \rho^4\text{Var}(\theta_{i,t-2}) \\
&= \cdots \qquad \text{(expanding the variance recursively)} \\
&= (1+\rho^2+\cdots+\rho^{2(t-2)})\tau^2 + \rho^{2(t-1)}\text{Var}(\theta_{i1})) \\
&= (1+\rho^2+\cdots+\rho^{2(t-2)})\tau^2 + \rho^{2(t-1)}\sigma_1^2
\end{aligned}$$

4. In lecture, we examined the fit of an exponentially time-weighted Bradley-Terry model for NBA basketball game outcomes from 2003 to 2018 with a decay parameter estimated by train/test validation. For this problem, you will analyze NFL game outcome data using the same train/test split strategy, but instead use a normal model for score differences. The data file `nfl-2002-2019.csv` is in the Data sets folder. The variables in the file are

   `season`: NFL season (2002 through 2019)

   `home`: Home team

   `visitor`: Away team

   `outcome`: Home score minus Away score

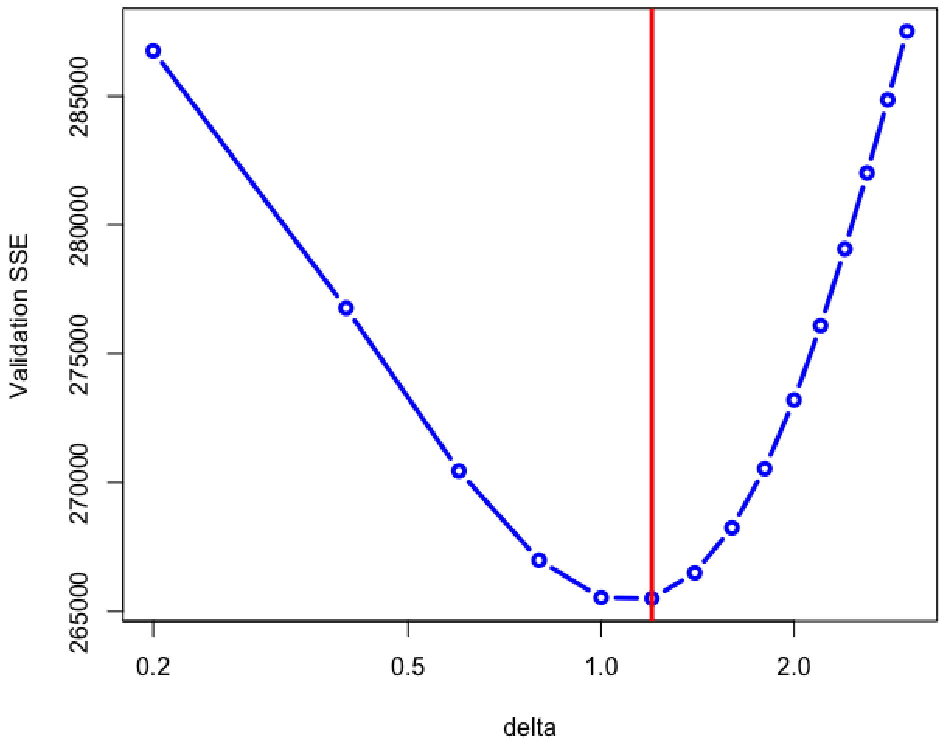
   (a) Adjust the existing R code on dynamic modeling to make the following changes:
   - Instead of fitting a Bradley-Terry model for each season, fit a normal model for the score difference.
   - Instead of computing a Bernoulli log-likelihood on validation data, compute the sum of squared deviations between the observed and predicted score differences.
   - Experiment with different ranges of the decay parameter (the ones used for Bradley-Terry models and basketball may not be appropriate for normal score difference models and NFL football).
   - You can still start computing the validation measure at year 2009 (so this is not a change!).

   Graph the set of values of the decay parameter, $\delta$, and the predictive validity measure (sum of squared prediction errors). What decay parameter did you choose as the best?

   Solution:

```
> nfl = read.csv("nfl-2002-2019.csv")
>
> Teams = sort(unique(nfl$home))
> n_teams = length(Teams)
> valid.start.seas = 2009  # start of validation data
> max.seas = max(nfl$season)
>
```

```
> # Construct design matrix
> X = outer(nfl$home, Teams, "==") - outer(nfl$visitor, Teams, "==")
> W = rbind(diag(n_teams-1), rep(-1, n_teams-1))
> X.star = X %*% W
> nfl$X.star = X.star
>
> train_val_split = function(df,valid.start.season, max.season){
+   idx = list()
+   i = 1
+   for (cur.season in valid.start.season:max.season){
+     range = as.vector((nrow(df[df$season<cur.season,])+1):
+                       nrow(df[df$season<=cur.season,]))
+     idx[[i]] = as.list(sample(x=range,
+                       size=nrow(df[df$season==cur.season,])%/%2))
+     i = i+1
+   }
+   val.samples.idx = data.frame("sample.idx"=I(idx))
+   row.names(val.samples.idx) = valid.start.season:max.season
+   return (val.samples.idx)
+ }
>
> val.samples.idx=train_val_split(nfl, valid.start.seas, max.seas)
>
> fit_and_validate =
+   function(df, delta, val.samples.idx, valid.start.season, max.season){
+   total.sse = 0
+   for (cur.season in valid.start.season:max.season){
+     df.cp = df[df$season <= cur.season,]
+
+     # Train-val split
+     idx = unlist(val.samples.idx[as.character(cur.season), 1])
+     val = df.cp[idx,]
+     df.cp = df.cp[-idx, ]
+
+     # Assign decaying weights
+     df.cp["weights"] = exp(delta * (df.cp$season - cur.season))
+
+     # Fit a normal model
+     ability.lm = lm(outcome~X.star+0, weights=weights, data=df.cp)
+
+     # Compute SSE
+     p = predict(ability.lm, newdata=val)
+     total.sse = total.sse + sum((p-val$outcome)^2)
+   }
+   return (total.sse)
+ }
>
> sse.prediction = NULL
> deltas = seq(from = 0.2, to = 3, by = 0.2)
> for (delta in deltas){
+   sse.prediction = c(sse.prediction,
+         fit_and_validate(nfl, delta,
+             val.samples.idx,valid.start.seas, max.seas))
+ }
> best.delta = deltas[which(sse.prediction==min(sse.prediction))]
> best.delta
[1] 1.2
> plot(deltas, sse.prediction, log = 'x',type='b',col="blue",lwd=3,
+       xlab="delta",ylab="Validation SSE")
> abline(v = best.delta,lwd=3,col="red")
```



The best decay parameter according to validation SSE is 1.2.

(b) With the optimized decay parameter, rerun the normal score difference model on all NFL seasons, and report the estimated team strengths and standard errors of the strengths in 2019. Who are the top four teams? Did any of the top four win the Super Bowl at the start of 2020 (i.e., the Chiefs)?

Solution:

```
> ability.lm = lm(outcome~X.star+0,
+                    weights=exp(best.delta * (nfl$season - max.seas)),
+                    data=nfl)
```

```
> Teams.ability =
+   data.frame(est = W %*% ability.lm$coefficients,
+              stderr = sqrt(diag(W %*% vcov(ability.lm) %*% t(W))))
>
> row.names(Teams.ability) = Teams
>
> oo = order(Teams.ability$est,decreasing=T)
> Teams.ability.order = Teams.ability[oo,]
> round(Teams.ability.order, 4)
                         est stderr
Baltimore Ravens     11.0943 0.7459
New England Patriots  8.9727 0.7464
Kansas City Chiefs    8.5870 0.7459
New Orleans Saints    7.2569 0.7461
Minnesota Vikings     5.0801 0.7460
San Francisco 49ers   4.8278 0.7461
Los Angeles Rams      4.6372 0.7462
Dallas Cowboys        3.7912 0.7461
Seattle Seahawks      2.9886 0.7462
Green Bay Packers     2.0147 0.7460
Tennessee Titans      1.9765 0.7456
Pittsburgh Steelers   1.5860 0.7460
Los Angeles Chargers  1.3308 0.7456
Philadelphia Eagles   1.2957 0.7462
Chicago Bears         1.2815 0.7460
Houston Texans        0.7747 0.7457
Atlanta Falcons       0.2384 0.7461
Buffalo Bills        -0.5562 0.7459
Indianapolis Colts   -0.6973 0.7456
Tampa Bay Buccaneers -0.8362 0.7461
Denver Broncos       -1.6342 0.7456
Detroit Lions        -3.1445 0.7461
Cleveland Browns     -3.4949 0.7460
Carolina Panthers    -4.2373 0.7461
Jacksonville Jaguars -4.4802 0.7457
Arizona Cardinals    -4.9368 0.7462
New York Giants      -6.2676 0.7462
Cincinnati Bengals   -6.4258 0.7459
New York Jets        -6.4536 0.7460
Oakland Raiders      -6.5533 0.7456
Washington Redskins  -8.2263 0.7462
Miami Dolphins       -9.7898 0.7459
```

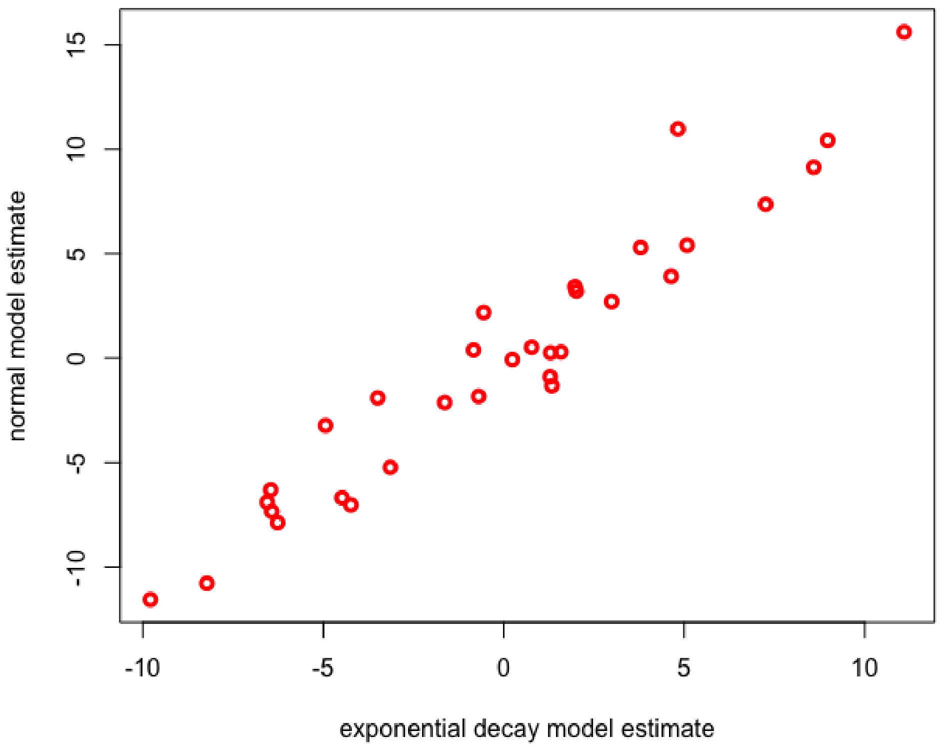
The top four teams are Baltimore Ravens, New England Patriots, Kansas City Chiefs and New Orleans Saints, which indeed include the Chiefs!

(c) Run the normal score difference model using only the 2019 regular season data. Compare the results of this model to the exponential decay model results. What do you make of the comparison?

Solution:

```
> ability.lm.2019 = lm(outcome~X.star+0, subset=(season==2019), data=nfl)
> Teams.ability$est.2019 = W %*% ability.lm.2019$coefficients
> Teams.ability$stderr.2019 =
+   sqrt(diag(W %*% vcov(ability.lm.2019) %*% t(W)))
>
> Teams.ability.order = Teams.ability[oo,]
> round(Teams.ability.order, 4)
                         est stderr est.2019 stderr.2019
Baltimore Ravens     11.0943 0.7459  15.6161      3.1672
New England Patriots  8.9727 0.7464  10.4276      3.1672
Kansas City Chiefs    8.5870 0.7459   9.1410      3.1672
New Orleans Saints    7.2569 0.7461   7.3599      3.1672
Minnesota Vikings     5.0801 0.7460   5.4042      3.1672
San Francisco 49ers   4.8278 0.7461  10.9696      3.1672
Los Angeles Rams      4.6372 0.7462   3.9130      3.1672
Dallas Cowboys        3.7912 0.7461   5.2927      3.1672
Seattle Seahawks      2.9886 0.7462   2.7054      3.1672
Green Bay Packers     2.0147 0.7460   3.1990      3.1672
Tennessee Titans      1.9765 0.7456   3.4205      3.1672
Pittsburgh Steelers   1.5860 0.7460   0.2932      3.1672
Los Angeles Chargers  1.3308 0.7456  -1.3208      3.1672
Philadelphia Eagles   1.2957 0.7462   0.2573      3.1672
Chicago Bears         1.2815 0.7460  -0.8937      3.1672
Houston Texans        0.7747 0.7457   0.5184      3.1672
Atlanta Falcons       0.2384 0.7461  -0.0700      3.1672
Buffalo Bills        -0.5562 0.7459   2.1852      3.1672
Indianapolis Colts   -0.6973 0.7456  -1.8323      3.1672
Tampa Bay Buccaneers -0.8362 0.7461   0.3870      3.1672
Denver Broncos       -1.6342 0.7456  -2.1236      3.1672
Detroit Lions        -3.1445 0.7461  -5.2333      3.1672
Cleveland Browns     -3.4949 0.7460  -1.9040      3.1672
Carolina Panthers    -4.2373 0.7461  -7.0280      3.1672
Jacksonville Jaguars -4.4802 0.7457  -6.6868      3.1672
Arizona Cardinals    -4.9368 0.7462  -3.2266      3.1672
New York Giants      -6.2676 0.7462  -7.8802      3.1672
Cincinnati Bengals   -6.4258 0.7459  -7.3418      3.1672
New York Jets        -6.4536 0.7460  -6.3082      3.1672
Oakland Raiders      -6.5533 0.7456  -6.8976      3.1672
Washington Redskins  -8.2263 0.7462 -10.7813      3.1672
Miami Dolphins       -9.7898 0.7459 -11.5620      3.1672
>
> plot(Teams.ability$est, Teams.ability$est.2019,
+      col="red",lwd=3,
+      xlab="exponential decay model estimate",
+      ylab="normal model estimate")
```

```
> cor(Teams.ability$est, Teams.ability$est.2019)
          [,1]
[1,] 0.9628854
```



The strength estimates from the two models are generally consistent, with a correlation of 0.96. However, the estimates from the normal model using only the 2019 data exhibit much larger variance due to its significantly smaller dataset size. A exponential decay model is preferable as it uses data more efficiently to reduce the uncertainty of estimates.