Quantitative Methods in Political Science - Homework 10

Team member 1 (with percentage) Team member 2 (with percentage)
Team member 3 (with percentage)

Due: December 1, 2022

Contents

Part 1: Poisson MLE 2
Part 2: Civil Wars - Revisited 6
R code 10
References 15

- *Please try to answer the questions with **short** but very **precise** statements. However, do not hide behind seemingly fancy jargon.*
- *If you do not have any special reason, please do not load additional packages to solve this homework assignment. If you nevertheless do so, please indicate why you think this is necessary and add the package to the `p_needed` vector in the setup chunk.*
- *In addition to the `Rmd`, please make sure that in the repo, there is a PDF knitted from the latest version of your code. The automated check for reproducibility on Github will run only when you include the word "final" into the commit message.*
- *This time, we will ask you to start paying special attention to editing and formatting (this will be useful for your Data essay), so you start to use chunk options. You are welcome to have a look at lab code for examples of using chunk options or to consult this page.*

Part 1: Poisson MLE

In 1982, S. Sidney Ulmer, writing in the American Journal of Political Science (AJPS), demonstrated that yearly Supreme Court appointments follow a Poisson distribution. Between 1790 and 1980, Ulmer (1982) tabulated the following:

Number of appointments k	Number of years with k appointments
0	114
1	58
2	17
3	2
4 or more	0

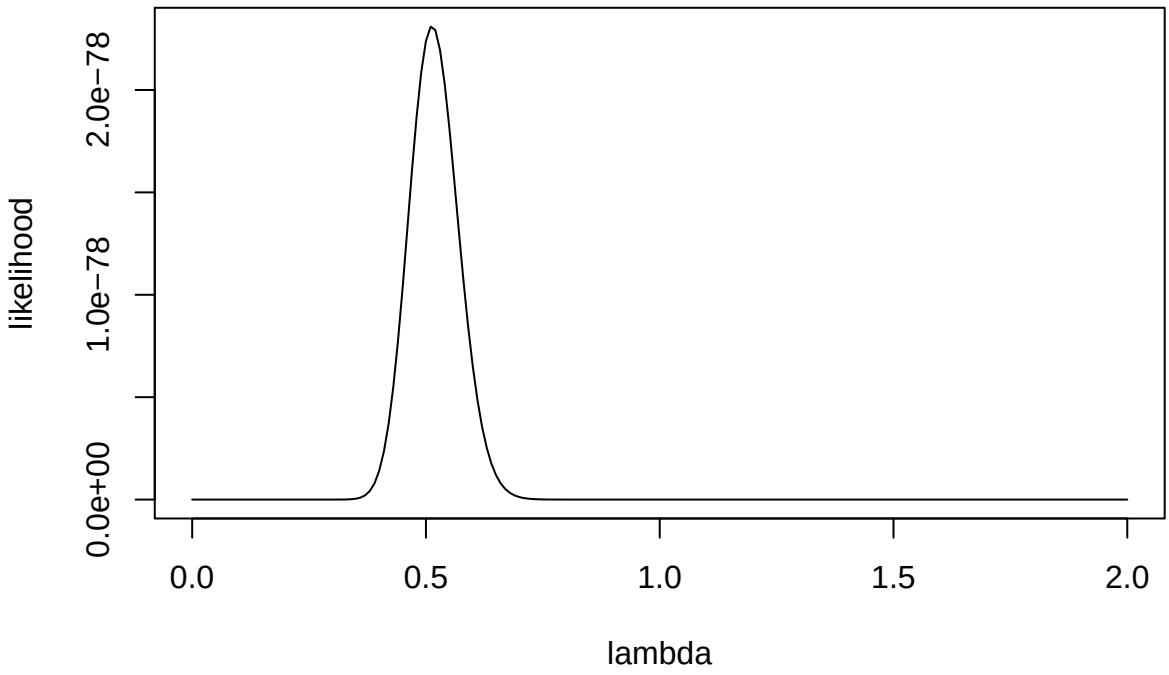
To begin, enter these data into `R`, such that N = 191 (N = length of the vector). Then:

1. *Write both, a likelihood function and a log-likelihood function for the Poisson distribution in* `R`*,*

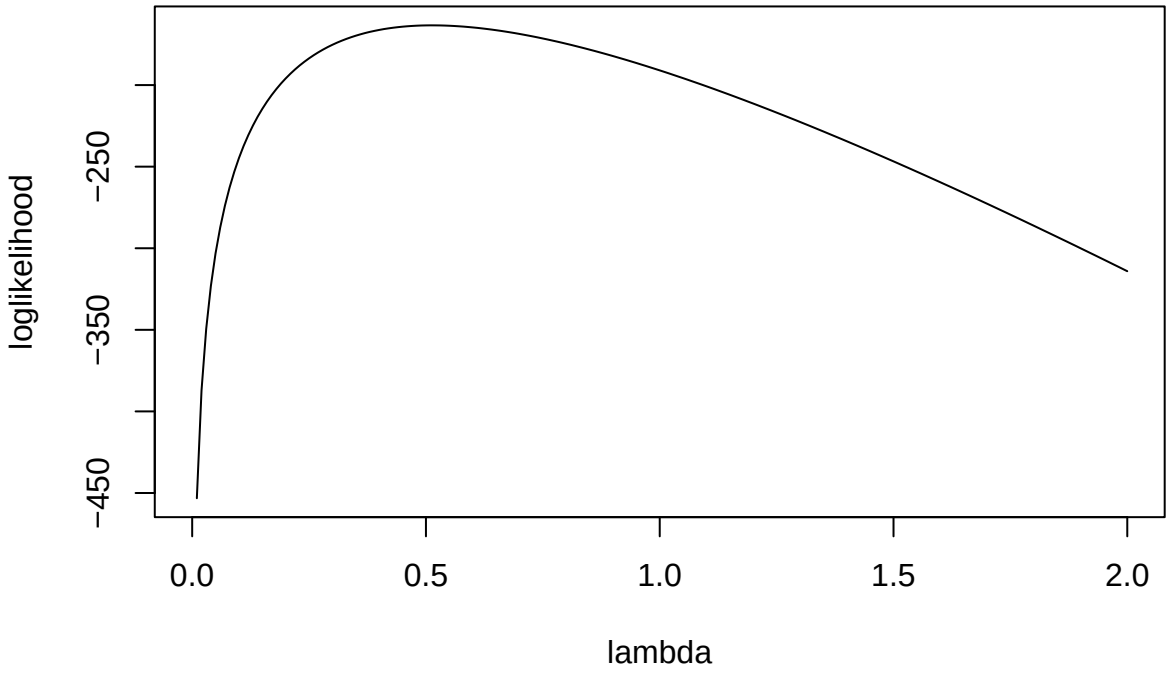
$$f_{Poisson}(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$$

and plot the two functions for values of the parameter λ between 0 and 2 (Hint: You might want to use sapply or a for loop to achieve this). k represents the number of occurrences of an event. In this case, the observed number of Supreme Court appointments in a year. Briefly describe the plot.

```
#create data vector
k <- rep(0, 191)
k[115:172] <- 1
k[173:189] <- 2
k[190:191] <- 3
#likelihood function
lik <- function(k, lambda){
  lk <- prod(exp(-lambda)*lambda^k/factorial(k))
  return(lk)
}
#log-likelihood function
lglik<- function(k, lambda){
  lglk <- sum(k*log(lambda)-lambda)
  return(lglk)
}
lambda_x <- seq(0,2,0.01)
lk_y <- sapply(lambda_x, lik, k=k )
lglk_y <- sapply(lambda_x, lglik, k=k )
plot(lambda_x, lk_y,type = 'l', col = "black", ylab="likelihood", xlab= "lambda")
```

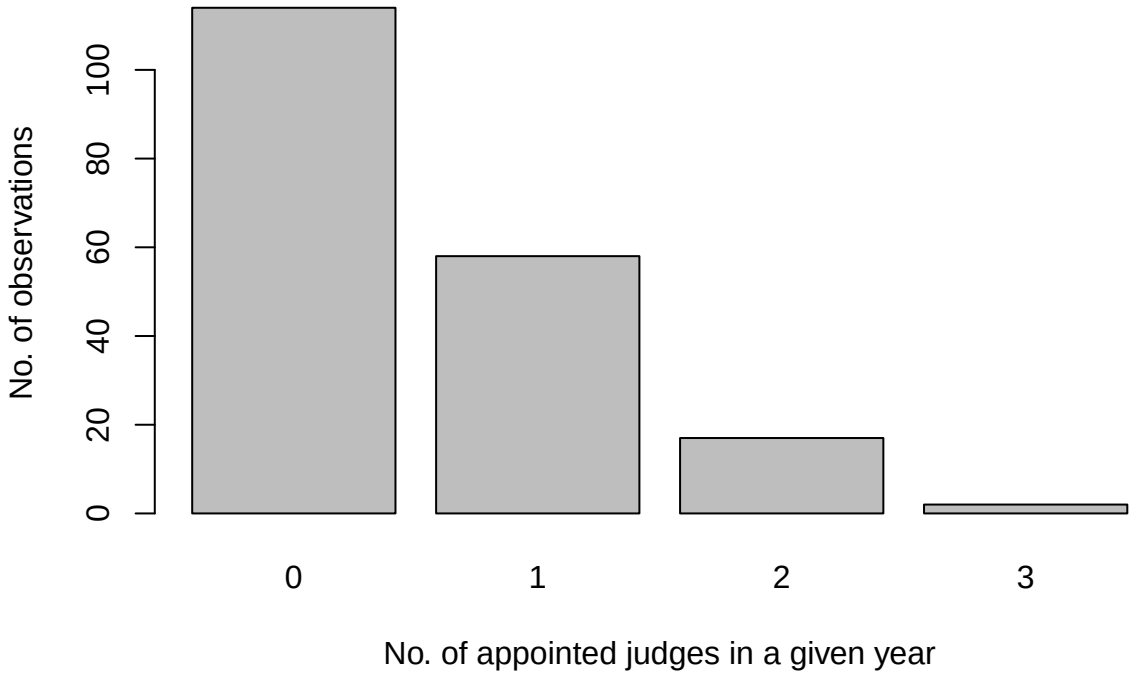



```
plot(lambda_x, lglk_y,type = 'l', col = "black", ylab="loglikelihood", xlab= "lambda")
```



2. Create a descriptive plot of the dependent variable ***number of yearly Supreme Court appointments****. Briefly describe the variable, what is displayed in the plot and how the observed data is distributed.*

```
k_table = as.matrix(table(k))

barplot(k_table[,1], names = rownames(k_table), ylab = "No. of observations", xlab = "No. of appointed
```



Answer:

3. Use the appropriate `R` *function (for a one-dimensional optimization problem) to find the maximum likelihood estimate of* λ*. Given this parameter estimate, check to make sure that the data really are Poisson distributed. For this, plot expected values from a Poisson distribution with the estimated* λ *and a histogram with the actual observations. How close are our observed and expected values? If your function worked properly, show that* $\hat{\lambda} \approx \bar{k}$*, where* $\bar{k} =$ *the mean of appointments per year.*

```
# Starting Values
stval <- 0.1

# Optimize
res <-
  optimx(
    stval,
    lglik,
    k = k,
    control = list(maximize = T)
  )
```

```
## Maximizing -- use negfn and neggr
```

```
## Warning in optim(par = par, fn = ufn, gr = ugr, method = meth, control = mcontrol, : one-dimensional
## use "Brent" or optimize() directly
```

```
## Warning in log(lambda): NaNs produced
```

```
## Warning in log(lambda): NaNs produced
```

```
## Warning in log(lambda): NaNs produced
```

```
## Warning in log(lambda): NaNs produced
```

```
## Warning in log(lambda): NaNs produced

## Warning in log(lambda): NaNs produced

## Warning in log(lambda): NaNs produced
```

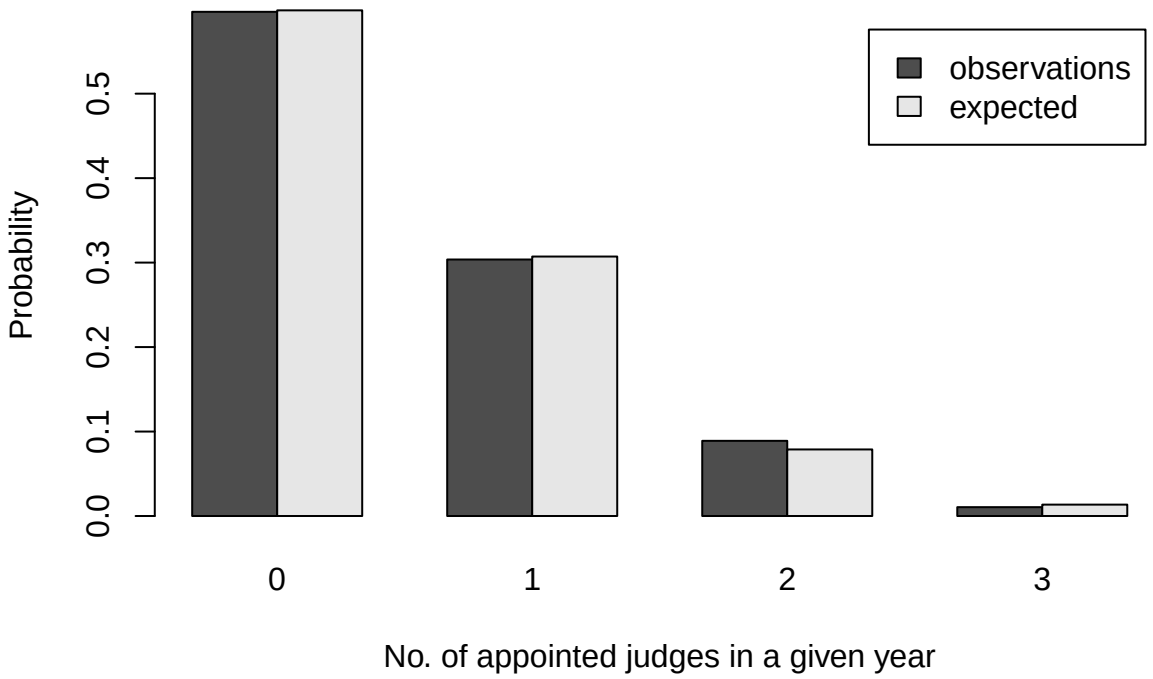
```
## Maximizing -- use negfn and neggr
lambda_hat <- res$p1[1]
lambda_hat
```

```
## [1] 0.513125
```

```
## [1] 0.513125
lambda_bar <- mean(k)
lambda_bar
```

```
## [1] 0.513089
```

```
## [1] 0.513089
prob_obs <- c(114/191,58/191,17/191,2/191)
prob_hat <- unique(exp(-lambda_hat)*lambda_hat^k/factorial(k))

plot_prob <- cbind(prob_obs,prob_hat)

colnames(plot_prob) <- c("observations", "expected")

barplot(t(plot_prob), names= c(0,1,2,3),beside =T, ylab = "Probability", xlab = "No. of appointed judges
```



Answer:

4. Explain the concept over- and underdispersion in the context of yearly number of Supreme Court appointments (no coding required).

Answer:

Part 2: Civil Wars - Revisited

In their paper, Eck and Hultman (2007) present new data on one-sided violence in intrastate armed conflicts. They present the results of six different models, assessing the determinants of one-sided violence. Four of the models are negative binomial models.

The dataset `eck_rep.dta` contains the data used in Eck and Hultman (2007). Please exclude the outlier Rwanda 1994 - as in the lab - for the analysis. The following table gives an overview of the variables you will need for the models:

Variables in Table IV	Name in dataset
One-sided killings	`os_best`
Previous War	`war15yrs`
Civil War	`intensity_dyad`
Autocracy	`auto`
Democracy	`demo`
Trade	`trade_gdp_alt`
Government	`govt`
One-sided violence$_{t-1}$	`prior_os`

1. Load the data and create a your own version of the lagged variable ***`prior_os`*** *that measures the number of one-sided killings,* ***`os_best`****, in the previous year $t-1$ (Hint: The resulting variable takes the value of* ***`os_best`*** *one year prior to the existing observation). Compare the lagged variable you created to the existing one,* ***`prior_os`****. Are there any differences and where do they originate from?* [1]

```
#first we load the data
df <- read.dta("raw-data/eck_rep.dta", convert.factors = FALSE)
df$os_best[df$os_best == 500000] <- NA

#let's proceed with the "lagging"
library("dplyr")
df_mod <- df %>%
  group_by(id_act) %>%
  dplyr::mutate(os_lag = lag(os_best, n = 1, default = NA))
#now we have a new dataframe with the newly created variable os_lag

#let's compare the two
summary(df$prior_os)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    0.00    0.00    0.00   48.13    0.00 4070.00      96
```

```
summary(df_mod$os_lag)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    0.00    0.00    0.00   57.22    0.00 4070.00     280
```

[1] Create Lagged Variable by Group

```r
#we can also create a column for the differences between the two variables
df_mod$diff <- df_mod$os_lag - df_mod$prior_os

#we can notice some differences
```

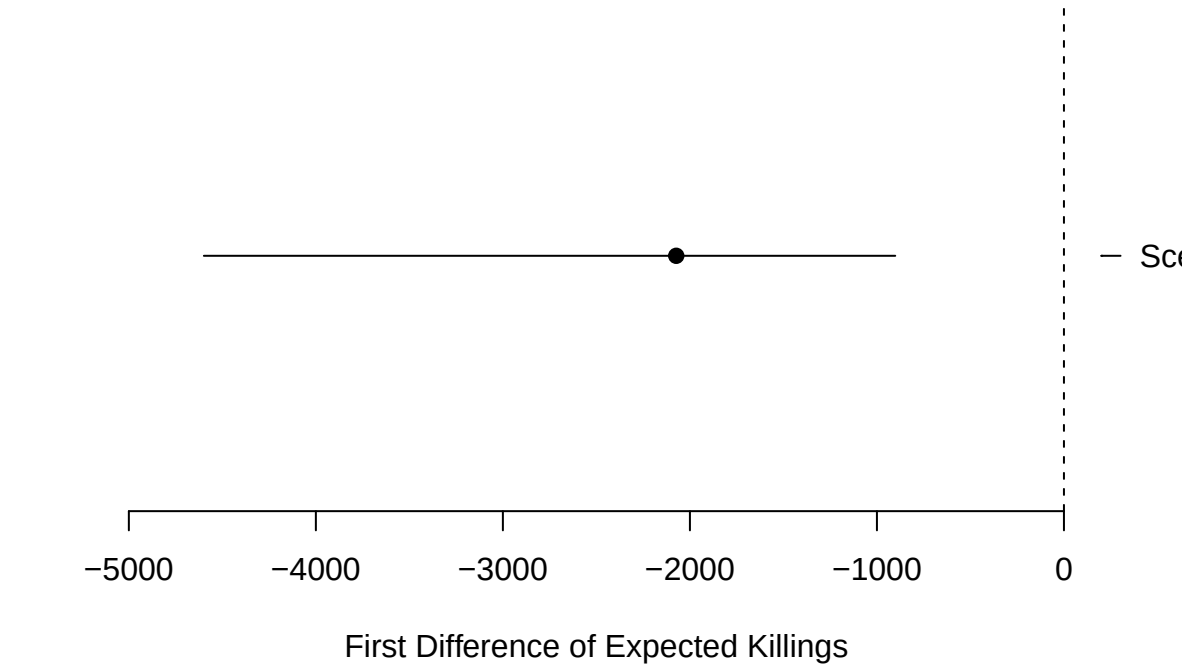
Answer: From my understanding, the new lagged variable os_lag has a flaw: it does not always effectively account for the t-1 year for its values. This is because in cases where two consecutive observations are not referring to consecutive years, e.g. 1996 followed by 1998 or 2001, the lagged still reports the previous data, which in our example would actually be t-2 or t-5; on the contrary, the original lagged prior_os, to minimize this, is "coded manually" (p242, note 23). *Also, os killings may differ from os violence; actually, "os killings" never appears in the paper, thus I would not understand why they are named differently in the previous table.*

2. Replicate models 3 and 4 presented in Table IV on page 243 (Hint: You can use the glm.nb function for this). Present your results in a nice-looking table. Your (coefficient) estimates should be identical to the ones reported in the paper, but standard errors will be different. Hide the code you use to generate the table, but report the table itself.

3. Calculate and present a first difference for government initiated one-sided killings between a ***Democracy without a civil war and without prior one-sided violence*** *and an* ***Autocracy with a civil war and 400 prior one-sided killings****. Use model 4 to calculate first differences. In your answer, describe the scenarios you compare, what the plot shows (axes, data points, etc.) and your interpretation of the results.*

```r
# setting up and conducting simulation
gamma_hat <- coef(m4)
V_hat <- vcov(m4)
S <- mvrnorm(1000, gamma_hat, V_hat)


# preparing scenarios
scenario1 <- cbind(
  1, #intercept
  1, #intensity_dyad
  0, #autocracy
  1, #demo
  1, #govt
  0 #prior_os
)
colnames(scenario1) <- names(gamma_hat)

scenario2 <- cbind(
  1, #intercept
  2, #intensity_dyad
  1, #autocracy
  0, #demo
  1, #govt
  400 #prior_os
)
colnames(scenario2) <- names(gamma_hat)


# calculating linear predictor
Xbeta1 <- S %*% t(scenario1)
Xbeta2 <- S %*% t(scenario2)
```

```r
# generating expected values for lambda
lambda1 <- exp(Xbeta1)
lambda2 <- exp(Xbeta2)

# setting up theta
theta <- m4$theta
exp_scenario1 <- sapply(lambda1, function(x) {
  mean(rnbinom(1000, size = theta, mu = x))
})
exp_scenario2 <- sapply(lambda2, function(x) {
  mean(rnbinom(1000, size = theta, mu = x))
})
exp_values <- cbind(exp_scenario1, exp_scenario2)
df <- data.frame(exp_values)

# generating first differences
df$fd <- exp_scenario1 - exp_scenario2

# calculating median and confidence intervals
median_fd <- median(df$fd)
ci_fd <- quantile(df$fd, probs = c(0.025, 0.975))

# plotting results

plot(
  y = 1,
  x = median_fd,
  xlab = 'First Difference of Expected Killings',
  ylab = '',
  yaxt = "n",
  xlim = c(-5000, 0),
  pch = 19,
  main = 'First Differences of Expected Killings between Specified Scenarios',
bty = "n"
)
axis(4,
     at = 1,
     labels = c("Scenario 2"),
     las = 2)
segments(
  x0 = ci_fd[1],
  y0 = 1,
  x1 = ci_fd[2],
  y1 = 1,

)
segments(
  x0 = 0,
  y0 = 0,
  x1 = 0,
  y1 = 2,
  lty = "dashed"
)
```



Answer:

The graph compares the expected number of killings between a baseline scenario of an autocracy with a civil war and 400 prior one-sided killings and a democracy without a civil war and no prior one-sided violence. The model expects a mean of roughly 2000 killings less for the second scenario, with a 95%-confidence interval ranging from about 900 and 4900 killings.

R code

```
# The first line sets an option for the final document that can be produced from
# the .Rmd file. Don't worry about it.
knitr::opts_chunk$set(echo = TRUE)

# The next bit (lines 22-43) is quite powerful and useful.
# First you define which packages you need for your analysis and assign it to
# the p_needed object.
p_needed <-
  c("viridis", "stargazer", "MASS", "optimx", "foreign", "dplyr")
#added foreign to read the .dta dataset file, dplyr to create the lagged variable

# Now you check which packages are already installed on your computer.
# The function installed.packages() returns a vector with all the installed
# packages.
packages <- rownames(installed.packages())
# Then you check which of the packages you need are not installed on your
# computer yet. Essentially you compare the vector p_needed with the vector
# packages. The result of this comparison is assigned to p_to_install.
p_to_install <- p_needed[!(p_needed %in% packages)]
# If at least one element is in p_to_install you then install those missing
# packages.
if (length(p_to_install) > 0) {
  install.packages(p_to_install)
}
# Now that all packages are installed on the computer, you can load them for
# this project. Additionally the expression returns whether the packages were
# successfully loaded.
sapply(p_needed, require, character.only = TRUE)

# This is an option for stargazer tables
# It automatically adapts the output to html or latex,
# depending on whether we want a html or pdf file
stargazer_opt <- ifelse(knitr::is_latex_output(), "latex", "html")

#create data vector
k <- rep(0, 191)
k[115:172] <- 1
k[173:189] <- 2
k[190:191] <- 3
#likelihood function
lik <- function(k, lambda){
  lk <- prod(exp(-lambda)*lambda^k/factorial(k))
  return(lk)
}
#log-likelihood function
lglik<- function(k, lambda){
  lglk <- sum(k*log(lambda)-lambda)
  return(lglk)
}
lambda_x <- seq(0,2,0.01)
lk_y <- sapply(lambda_x, lik, k=k )
```

```r
lglk_y <- sapply(lambda_x, lglik, k=k )
plot(lambda_x, lk_y,type = 'l', col = "black", ylab="likelihood", xlab= "lambda")
plot(lambda_x, lglk_y,type = 'l', col = "black", ylab="loglikelihood", xlab= "lambda")

k_table = as.matrix(table(k))

barplot(k_table[,1], names = rownames(k_table), ylab = "No. of observations", xlab = "No. of appointed j


# Starting Values
stval <- 0.1

# Optimize
res <-
  optimx(
    stval,
    lglik,
    k = k,
    control = list(maximize = T)
  )

## Maximizing -- use negfn and neggr
lambda_hat <- res$p1[1]
lambda_hat
## [1] 0.513125
lambda_bar <- mean(k)
lambda_bar
## [1] 0.513089
prob_obs <- c(114/191,58/191,17/191,2/191)
prob_hat <- unique(exp(-lambda_hat)*lambda_hat^k/factorial(k))

plot_prob <- cbind(prob_obs,prob_hat)

colnames(plot_prob) <- c("observations", "expected")

barplot(t(plot_prob), names= c(0,1,2,3),beside =T, ylab = "Probability", xlab = "No. of appointed judges
#first we load the data
df <- read.dta("raw-data/eck_rep.dta", convert.factors = FALSE)
df$os_best[df$os_best == 500000] <- NA

#let's proceed with the "lagging"
library("dplyr")
df_mod <- df %>%
  group_by(id_act) %>%
  dplyr::mutate(os_lag = lag(os_best, n = 1, default = NA))
#now we have a new dataframe with the newly created variable os_lag

#let's compare the two
summary(df$prior_os)
summary(df_mod$os_lag)

#we can also create a column for the differences between the two variables
df_mod$diff <- df_mod$os_lag - df_mod$prior_os
```

```r
#we can notice some differences


# replicating model 3
m3 <-
  glm.nb(
    os_best ~ war15yrs + auto + trade_gdp_alt + prior_os,
    data = df,
    control = glm.control(maxit = 100)
  )


# replicating model 4
m4 <-
  glm.nb(
    os_best ~ intensity_dyad + auto + demo + govt + prior_os,
    data = df,
    control = glm.control(maxit = 100)
  )


stargazer(m3, m4)


# setting up and conducting simulation
gamma_hat <- coef(m4)
V_hat <- vcov(m4)
S <- mvrnorm(1000, gamma_hat, V_hat)


# preparing scenarios
scenario1 <- cbind(
  1, #intercept
  1, #intensity_dyad
  0, #autocracy
  1, #demo
  1, #govt
  0 #prior_os
)
colnames(scenario1) <- names(gamma_hat)

scenario2 <- cbind(
  1, #intercept
  2, #intensity_dyad
  1, #autocracy
  0, #demo
  1, #govt
  400 #prior_os
)
colnames(scenario2) <- names(gamma_hat)


# calculating linear predictor
```

```r
Xbeta1 <- S %*% t(scenario1)
Xbeta2 <- S %*% t(scenario2)

# generating expected values for lambda
lambda1 <- exp(Xbeta1)
lambda2 <- exp(Xbeta2)

# setting up theta
theta <- m4$theta
exp_scenario1 <- sapply(lambda1, function(x) {
  mean(rnbinom(1000, size = theta, mu = x))
})
exp_scenario2 <- sapply(lambda2, function(x) {
  mean(rnbinom(1000, size = theta, mu = x))
})
exp_values <- cbind(exp_scenario1, exp_scenario2)
df <- data.frame(exp_values)

# generating first differences
df$fd <- exp_scenario1 - exp_scenario2

# calculating median and confidence intervals
median_fd <- median(df$fd)
ci_fd <- quantile(df$fd, probs = c(0.025, 0.975))

# plotting results

plot(
  y = 1,
  x = median_fd,
  xlab = 'First Difference of Expected Killings',
  ylab = '',
  yaxt = "n",
  xlim = c(-5000, 0),
  pch = 19,
  main = 'First Differences of Expected Killings between Specified Scenarios',
bty = "n"
)
axis(4,
     at = 1,
     labels = c("Scenario 2"),
     las = 2)
segments(
  x0 = ci_fd[1],
  y0 = 1,
  x1 = ci_fd[2],
  y1 = 1,

)
segments(
  x0 = 0,
  y0 = 0,
  x1 = 0,
```

```
  y1 = 2,
  lty = "dashed"
)
```

References

Eck, Kristine, and Lisa Hultman. 2007. "One-Sided Violence Against Civilians in War: Insights from New Fatality Data." *Journal of Peace Research* 44 (2): 233–46.

Ulmer, Sidney S. 1982. "Supreme Court Appointments as a Poisson Distribution." *American Journal of Political Science* 26 (1): 90–112.