Problem Set 1

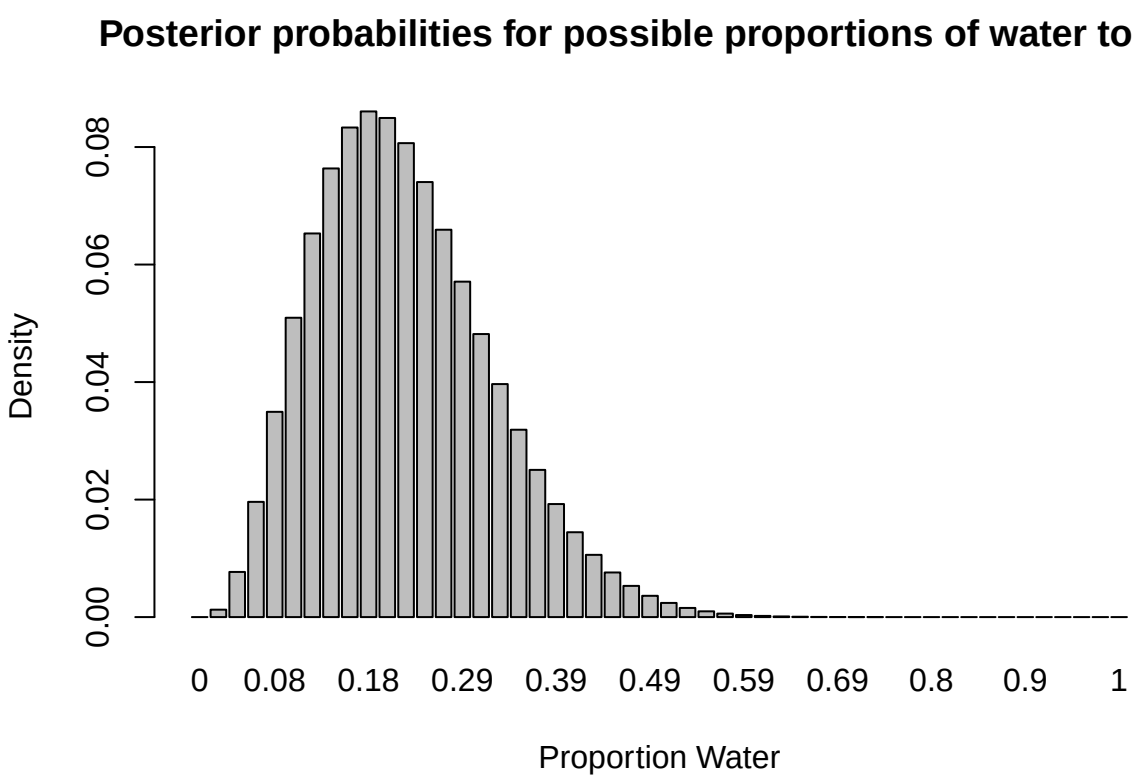
Nate Gonzales-Hess

2025-04-07

Problem 1: Globe Tossing Posterior Distribution

Suppose the globe tossing experiment turned out to be 3 water and 11 land. Construct the posterior distribution.

```
# Function to compute posterior for a given sample and range of possible proportions.
compute_posterior = function(sample, possible){

  W = sum(sample == "W")
  L = sum(sample == "L")

  prior = rep(1, length(possible)) # 2. Calculate likelihood and prior
  likelihood = sapply(possible, function(x) dbinom(x = W, size = W+L, prob = x))
  post = (prior * likelihood) / sum(prior * likelihood)  # 3. Calculate posterior probability

  return(post)
}

# Sample with 3 water and 13 land:
sample = c("W","W","W","L","L","L","L","L","L","L","L","L","L","L","L","L","L")
# 1. Create a grid of possible proportion values
possible = seq(0,1,length.out=50)
# Compute posteriors
posterior = compute_posterior(sample, possible)

# 4. Create a plot of the posterior distribution
barplot(posterior, names.arg = round(possible,2),
        main = "Posterior probabilities for possible proportions of water to land",
        xlab = "Proportion Water",
        ylab = "Density")
```



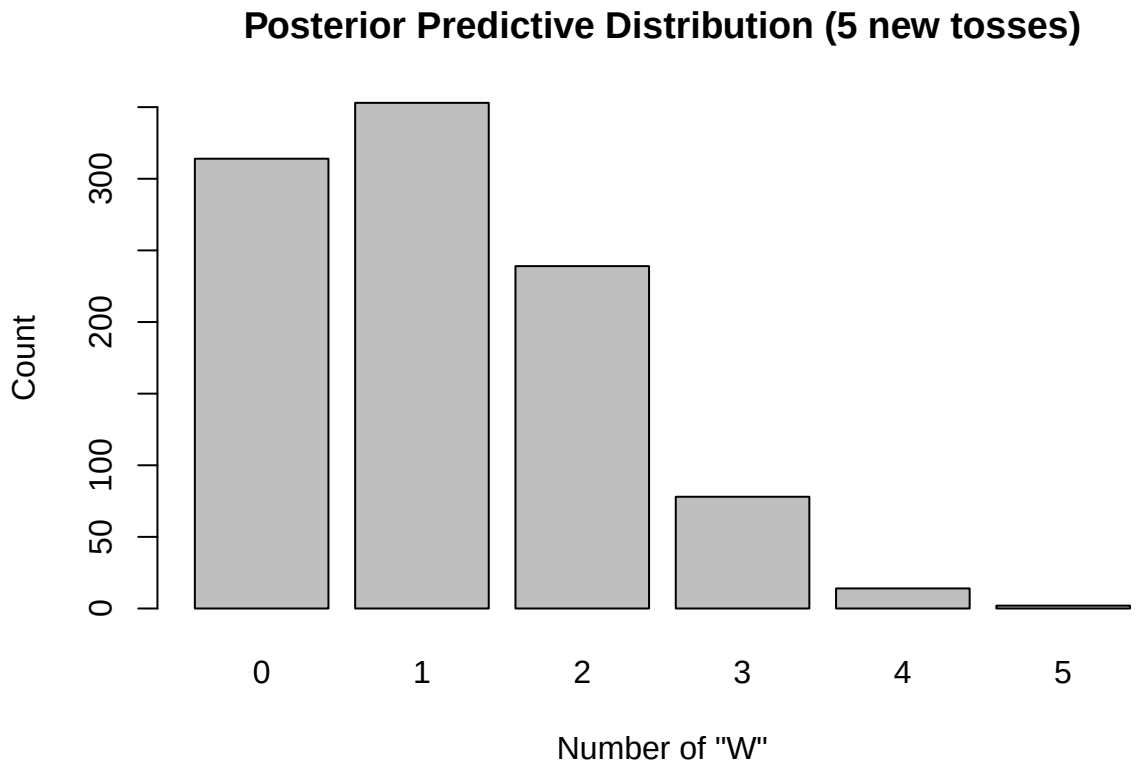
Starting from a flat prior, the posterior probability peaks around .20 (proportion of "W" to total) which makes sense, given that we observed 3 "W"s in 16 tosses and 3/16 = .1875. *This being a distribution, we aren't concerned with a particular value along the x-axis, but with the entire distribution. Still the distribution reflects the values in the sample pretty closely.

Problem 2: Posterior Predictive Distribution

Using the posterior distribution from Problem 1, compute the posterior predictive distribution for the next 5 tosses of the globe.

```
n_tosses = 5  # Number of steps in our simulation
sim_iters = 1000  # Number of times to run the simulated globe toss
predicted_counts = rep(0, n_tosses+1)  # Empty vector to store predicted "W" counts (0-5 possible "W" o

# 1. Sample from your posterior distribution
post_samples <- sample(possible, size=sim_iters, prob=posterior, replace=TRUE )

# 2. Use these samples to simulate new tosses
sim_globe = function( p=0.7 , N=5 ){
  sample(
    x = c("W", "L"),  # possible values
    size = N,         # how many draws
    prob = c(p, 1-p), # probability of each possibility
    replace = TRUE    # the same value can be drawn multiple times
  )
```

```
}

# Run simulation
for (i in 1:sim_iters){
  p = post_samples[i]
  tosses = sim_globe(p, n_tosses)
  water_count = sum(tosses == "W")
  predicted_counts[water_count+1] = predicted_counts[water_count+1] + 1
}

# 3. Create a visualization of the predictions
barplot(predicted_counts, names.arg = seq(0,5),
        main = "Posterior Predictive Distribution (5 new tosses)",
        xlab = "Number of \"W\"",
        ylab = "Count")
```



Using the posterior distribution to fuel 5 simulated tosses yielded a posterior predictive distribution that agreed well with the posterior probabilities. In 1000 simulated 5-toss runs, we saw 347 runs that yielded zero "W"s, 362 runs that yielded one "W", 198 runs that yielded two "W"s, 69 runs that yielded three "W"s, 22 runs that yielded four "W"s and 2 runs that gave five "W"s. –These counts aren't important on their own, but they give an approximate probability for each of the outcomes, and these probabilities fit well with the posterior probability, generated from a sample containing 3 "W"s and 13 "L"s. –In terms of generating data, this approach would yield samples pretty similar to the 3 "W" 13 "L" sample we started from.

Session Information

```
# This will help with debugging and reproducibility
sessionInfo()
```

```
## R version 4.4.3 (2025-02-28 ucrt)
## Platform: x86_64-w64-mingw32/x64
## Running under: Windows 11 x64 (build 26100)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: America/Los_Angeles
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] lubridate_1.9.4 forcats_1.0.0   stringr_1.5.1   dplyr_1.1.4
##  [5] purrr_1.0.4     readr_2.1.5     tidyr_1.3.1     tibble_3.2.1
##  [9] ggplot2_3.5.1   tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] gtable_0.3.6      compiler_4.4.3    tidyselect_1.2.1  scales_1.3.0
##  [5] yaml_2.3.10       fastmap_1.2.0     R6_2.6.1          generics_0.1.3
##  [9] knitr_1.50        munsell_0.5.1     pillar_1.10.1     tzdb_0.5.0
## [13] rlang_1.1.5       stringi_1.8.7     xfun_0.51         timechange_0.3.0
## [17] cli_3.6.4         withr_3.0.2       magrittr_2.0.3    digest_0.6.37
## [21] grid_4.4.3        rstudioapi_0.17.1 hms_1.1.3         lifecycle_1.0.4
## [25] vctrs_0.6.5       evaluate_1.0.3    glue_1.8.0        colorspace_2.1-1
## [29] rmarkdown_2.29    tools_4.4.3       pkgconfig_2.0.3   htmltools_0.5.8.1
```