Collection of Tick Simulation

Piper Zimmerman

December 12, 2023

```r
tab.lambda
tab.lambda.mean
tab.inv.lambda
tab.mean.inv.lambda
tab.inv.mean.lambda
```

```r
df.a <- data.frame(
  x = factor(c("Lambda","Lambda Mean","Inverse Lambda","Mean Inverse Lambda","Lambda Inverse Mean")),
  Mean = c(exp(mean(unlist(lapply(param_list1[[1]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(param_list2[[1]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(param_list3[[1]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(param_list4[[1]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(param_list5[[1]], function(matrix) matrix[, 1]))))),
  upper = c(exp(hdi(unlist(lapply(param_list1[[1]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(param_list2[[1]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(param_list3[[1]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(param_list4[[1]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(param_list5[[1]], function(matrix) matrix[, 1])))[2])),
  lower = c(exp(hdi(unlist(lapply(param_list1[[1]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(param_list2[[1]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(param_list3[[1]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(param_list4[[1]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(param_list5[[1]], function(matrix) matrix[, 1])))[1]))
)

a <- ggplot(df.a, aes(x, Mean))+
  geom_point(col="blue") + geom_linerange(aes(ymin = lower, ymax = upper))+labs(title="A",x="Method")+
  geom_hline(yintercept = true.a, linetype = "dashed", color = "darkorchid")+
  theme_classic()
a
```

```r
df.b <- data.frame(
  x = factor(c("Lambda","Lambda Mean","Inverse Lambda","Mean Inverse Lambda","Lambda Inverse Mean")),
  Mean = c(exp(mean(unlist(lapply(param_list1[[2]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(param_list2[[2]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(param_list3[[2]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(param_list4[[2]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(param_list5[[2]], function(matrix) matrix[, 1]))))),
  upper = c(exp(hdi(unlist(lapply(param_list1[[2]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(param_list2[[2]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(param_list3[[2]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(param_list4[[2]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(param_list5[[2]], function(matrix) matrix[, 1])))[2])),
  lower = c(exp(hdi(unlist(lapply(param_list1[[2]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(param_list2[[2]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(param_list3[[2]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(param_list4[[2]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(param_list5[[2]], function(matrix) matrix[, 1])))[1]))
)

b <- ggplot(df.b, aes(x, Mean))+
  geom_point(col="blue")+ geom_linerange(aes(ymin = lower, ymax = upper))+labs(title="B",x="Method")+
  geom_hline(yintercept = true.b, linetype = "dashed", color = "darkorchid")+
  theme_classic()
b
```

```r
df.c <- data.frame(
  x = factor(c("Lambda","Lambda Mean","Inverse Lambda","Mean Inverse Lambda","Lambda Inverse Mean")),
  Mean = c((mean(unlist(lapply(param_list1[[3]], function(matrix) matrix[, 1])))),
           (mean(unlist(lapply(param_list2[[3]], function(matrix) matrix[, 1])))),
           (mean(unlist(lapply(param_list3[[3]], function(matrix) matrix[, 1])))),
           (mean(unlist(lapply(param_list4[[3]], function(matrix) matrix[, 1])))),
           (mean(unlist(lapply(param_list5[[3]], function(matrix) matrix[, 1]))))),
  upper = c((hdi(unlist(lapply(param_list1[[3]], function(matrix) matrix[, 1])))[2]),
            (hdi(unlist(lapply(param_list2[[3]], function(matrix) matrix[, 1])))[2]),
            (hdi(unlist(lapply(param_list3[[3]], function(matrix) matrix[, 1])))[2]),
            (hdi(unlist(lapply(param_list4[[3]], function(matrix) matrix[, 1])))[2]),
            (hdi(unlist(lapply(param_list5[[3]], function(matrix) matrix[, 1])))[2])),
  lower = c((hdi(unlist(lapply(param_list1[[3]], function(matrix) matrix[, 1])))[1]),
            (hdi(unlist(lapply(param_list2[[3]], function(matrix) matrix[, 1])))[1]),
            (hdi(unlist(lapply(param_list3[[3]], function(matrix) matrix[, 1])))[1]),
            (hdi(unlist(lapply(param_list4[[3]], function(matrix) matrix[, 1])))[1]),
            (hdi(unlist(lapply(param_list5[[3]], function(matrix) matrix[, 1])))[1]))
)

c <- ggplot(df.c, aes(x, Mean))+
  geom_point(col="blue")+ geom_linerange(aes(ymin = lower, ymax = upper))+labs(title="C",x="Method")+
  geom_hline(yintercept = true.c, linetype = "dashed", color = "darkorchid")+
  theme_classic()
c
```

```r
N_tab.lambda
N_tab.lambda.mean
N_tab.inv.lambda
N_tab.mean.inv.lambda
N_tab.inv.mean.lambda
```

```r
N_df.a <- data.frame(
  x = factor(c("Lambda","Lambda Mean","Inverse Lambda","Mean Inverse Lambda","Lambda Inverse Mean")),
  Mean = c(exp(mean(unlist(lapply(N_param_list1[[1]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(N_param_list2[[1]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(N_param_list3[[1]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(N_param_list4[[1]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(N_param_list5[[1]], function(matrix) matrix[, 1]))))),
  upper = c(exp(hdi(unlist(lapply(N_param_list1[[1]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(N_param_list2[[1]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(N_param_list3[[1]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(N_param_list4[[1]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(N_param_list5[[1]], function(matrix) matrix[, 1])))[2])),
  lower = c(exp(hdi(unlist(lapply(N_param_list1[[1]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(N_param_list2[[1]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(N_param_list3[[1]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(N_param_list4[[1]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(N_param_list5[[1]], function(matrix) matrix[, 1])))[1]))
)

N_a <- ggplot(N_df.a, aes(x, Mean))+
  geom_point(col="blue") + geom_linerange(aes(ymin = lower, ymax = upper))+labs(title="A",x="Method")+
  geom_hline(yintercept = true.a, linetype = "dashed", color = "darkorchid")+
  theme_classic()
N_a
```

```
N_df.b <- data.frame(
  x = factor(c("Lambda","Lambda Mean","Inverse Lambda","Mean Inverse Lambda","Lambda Inverse Mean")),
  Mean = c(exp(mean(unlist(lapply(N_param_list1[[2]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(N_param_list2[[2]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(N_param_list3[[2]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(N_param_list4[[2]], function(matrix) matrix[, 1])))),
           exp(mean(unlist(lapply(N_param_list5[[2]], function(matrix) matrix[, 1]))))),
  upper = c(exp(hdi(unlist(lapply(N_param_list1[[2]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(N_param_list2[[2]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(N_param_list3[[2]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(N_param_list4[[2]], function(matrix) matrix[, 1])))[2]),
            exp(hdi(unlist(lapply(N_param_list5[[2]], function(matrix) matrix[, 1])))[2])),
  lower = c(exp(hdi(unlist(lapply(N_param_list1[[2]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(N_param_list2[[2]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(N_param_list3[[2]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(N_param_list4[[2]], function(matrix) matrix[, 1])))[1]),
            exp(hdi(unlist(lapply(N_param_list5[[2]], function(matrix) matrix[, 1])))[1]))
)

N_b <- ggplot(N_df.b, aes(x, Mean))+
  geom_point(col="blue")+ geom_linerange(aes(ymin = lower, ymax = upper))+labs(title="B",x="Method")+
  geom_hline(yintercept = true.b, linetype = "dashed", color = "darkorchid")+
  theme_classic()
N_b
```

```
N_df.c <- data.frame(
  x = factor(c("Lambda","Lambda Mean","Inverse Lambda","Mean Inverse Lambda","Lambda Inverse Mean")),
  Mean = c((mean(unlist(lapply(N_param_list1[[3]], function(matrix) matrix[, 1])))),
           (mean(unlist(lapply(N_param_list2[[3]], function(matrix) matrix[, 1])))),
           (mean(unlist(lapply(N_param_list3[[3]], function(matrix) matrix[, 1])))),
           (mean(unlist(lapply(N_param_list4[[3]], function(matrix) matrix[, 1])))),
           (mean(unlist(lapply(N_param_list5[[3]], function(matrix) matrix[, 1]))))),
  upper = c((hdi(unlist(lapply(N_param_list1[[3]], function(matrix) matrix[, 1])))[2]),
            (hdi(unlist(lapply(N_param_list2[[3]], function(matrix) matrix[, 1])))[2]),
            (hdi(unlist(lapply(N_param_list3[[3]], function(matrix) matrix[, 1])))[2]),
            (hdi(unlist(lapply(N_param_list4[[3]], function(matrix) matrix[, 1])))[2]),
            (hdi(unlist(lapply(N_param_list5[[3]], function(matrix) matrix[, 1])))[2])),
  lower = c((hdi(unlist(lapply(N_param_list1[[3]], function(matrix) matrix[, 1])))[1]),
            (hdi(unlist(lapply(N_param_list2[[3]], function(matrix) matrix[, 1])))[1]),
            (hdi(unlist(lapply(N_param_list3[[3]], function(matrix) matrix[, 1])))[1]),
            (hdi(unlist(lapply(N_param_list4[[3]], function(matrix) matrix[, 1])))[1]),
            (hdi(unlist(lapply(N_param_list5[[3]], function(matrix) matrix[, 1])))[1]))
)

N_c <- ggplot(N_df.c, aes(x, Mean))+
  geom_point(col="blue")+ geom_linerange(aes(ymin = lower, ymax = upper))+labs(title="C",x="Method")+
  geom_hline(yintercept = true.c, linetype = "dashed", color = "darkorchid")+
  theme_classic()
N_c
```

Overview:

The purpose of this is to simulate lifetime and mortality data. Instead of fitting lifetime directly, it is often just modeled as the inverse of mortality. However, this only works if lifetime is truly exponentially distributed. λ is taken to be exponentially distributed with a rate parameter μ, where μ is a function of temperature, which is assumed to be quadratic. The issue with this method is that Dr. Johnson expects that lifetime is more of a Weibull distribution.

The process was to simulate data from an exponential distribution at five temperatures throughout a plausible range (10, 17, 21, 25, 32). The parameter, λ, is a function of temperature, of the form $f(T) = aT^2 + bT + c$, was taken from Figure B.1 (d) of *Understanding uncertainty in temperature effects on vector-borne disease: A Bayesian approach*, and the parameters a, b, and c were found. The result was $f(T) = 0.001094 * T^2 - 0.0488 * T + 0.609$. 100 data points where then simulated from five different exponential distributions. The parameter of these distributions was the function of temperature $(f(T))$ at each of the five chosen temperatures. A data frame (500x2) was made with the simulated value as the response, and the temperature as the dependent variable.

The idea was to use this dataset of 500 points, use JAGS to fit a model, and regain the posterior distributions of the 3 parameters. Theoretically, the posterior distribution should find the true parameter just about every time. This was done in a loop so that 100 models were made. The first thing done was to create a list of lists. There was an outer list with 4 or 5 elements (one for each parameter: a, b, c, deviance, tau), and then an inner list with 100 elements for each of the 100 models. Each element was a 5x4 matrix where each row represented a chain, and the columns were the mean, the median, the lower hdi bound, and the upper hdi bound, respectively, of the posterior samples for that specific parameter. Within the loop itself, the model was specified, using a likelihood of

$$mu[i] < -(exp(la) * temp[i]^2 - exp(b.l) * temp[i] + c) * ((exp(la) * temp[i]^2 - exp(b.l) * temp[i] + c) > 0) +$$

$$((exp(la) * temp[i]^2 - exp(b.l) * temp[i] + c) <= 0) * 0.00001$$

$$trait[i] \sim dexp(mu[i]).$$

Since the value has to be non-zero, there is an indicator function so that if it is less than or equal to zero, it is replaced by 0.00001. Priors were set for a and b as normal with mean 0 and standard deviation $\frac{1}{10}$. c has to be positive, so the prior was exponential with parameter 0.5. Then, the initial values are set. The initial values for a, b, c, and tau were set as 0.001, 0.05, 0.6, and 0.2. Tau was only used in the inverse models. 60,000 iterations were used with a burn in of 30,000. The thinning rate was 8, with 5 chains. The mean, median, and hdi bounds were saved for each parameter for each chain in the list mentioned above.

Histograms were made for each by unlisting the first column (mean) of each element of the list. The coverage was found by checking if the true value was in the hdi or not. The average coverage of the five chains was taken for each model, and then those averages were averaged. This was included in a table, along with the true value, mean, lower and upper hdi values of the mean, and the RMSE.

Five different methods were evaluated: using the 100 individual observations from each temperature (lambda), dividing observations from each temperature into 10 sets of 10 and then taking the average and using that (resulting in a 50x2 dataframe) (mean lambda), inverting all the original observations and using those (inverse lambda), taking the average like mentioned before and then inverting the values (mean inverse lambda), and inverting all the values and then taking the average like mentioned before (inverse mean lambda). 100 models were run for each of these methods with a wider temperature interval (10, 17, 21, 25, 32), as well as a narrow interval (13, 17, 21, 25, 29).

Problems:

We had a very hard time getting parameters a and b to converge. Since they are close to zero, Dr. Johnson suggested putting them in log space, which seemed to work, so both a and b are in log space. We also changed their priors from an exponential distribution to a normal distribution.

Results:

Wide: The coverage was the best for the mean lambda method. The RMSE was the worst for the inverse lambda and inverse mean lambda. The posterior estimates for c were also far off. The coverage doesn't look horrible, but the width of the hdi is massive compared to that of the other methods.

Narrow: The coverage was the best for the mean lambda method. The RMSE was the worst for the inverse lambda and inverse mean lambda. The posterior estimates for c were also far off. The coverage doesn't look horrible, but the width of the hdi is massive compared to that of the other methods. The coverage for the lambda method was basically 0, with all parameters being underestimated.