Generative simulations for HAI as a function of distance

Zane

2023-05-30

# Basic simulation

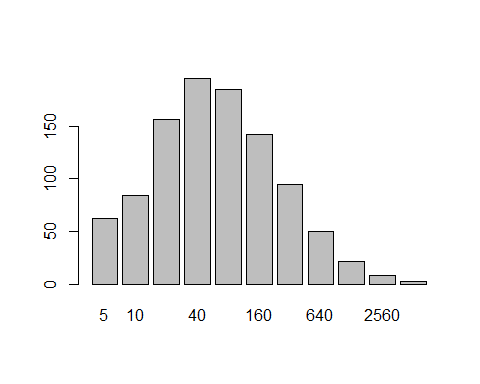
First we will attempt to simulate raw titer data, without including antigenic distance in the simulation. The outcome we want this simulation to produce is a semi-realistic distribution of titer values with support on the observed measurement space. We adopt the following model, where is the th individual’s measured titer.

We adopt the transformation

because the physical limit of detection (LoD) of the HAI assay is

and values below this threshold are, by convention, recorded as . So we know those values are below the LoD, but not what the values are. The distributional parameters , the mean, and , the variance, are unknown to us practically and must be estimated from the data.

one\_titer\_sim <- function(N = 1e4, seed = 370, mean = 3, sd = 1) {  
 set.seed(seed)  
 sim <-  
 tibble::tibble(  
 # Assume log(titer) is drawn from a normal distribution  
 raw\_log\_titer = rnorm(N, mean, sd),  
 # If we observe a titer with log(titer) < 1 (LOD), mark it as 0  
 trunc\_log\_titer = ifelse(raw\_log\_titer >= 1, raw\_log\_titer, 0),  
 # The assay is dilution based, so we only observe the floor of each  
 # value.  
 rounded\_titer = floor(trunc\_log\_titer),  
 # Now final observed titer is equal to this transformation.  
 sim\_titer = 5 \* 2 ^ rounded\_titer  
 ) |>  
 dplyr::arrange(raw\_log\_titer)  
}  
  
out <- one\_titer\_sim(1000, mean = 4, sd = 2)  
#plot(out$raw\_log\_titer, out$trunc\_log\_titer)  
#points(out$raw\_log\_titer, out$rounded\_titer, col = "blue")  
barplot(table(out$sim\_titer))



In the above plot, you can see the result of 1000 simulations, where and . (Note that the simulation is parametrized in terms of the standard deviation, .) Notably, while the mean was specified as (an observed titer of ), the mode of the distribution is at instead.

We can also notice that the observed mean is biased, estimated at on the scale of in the above model ( on the observed measurement scale). A [stackexchange post](https://math.stackexchange.com/questions/3662314/statistics-of-a-gaussian-random-variable-with-the-floor-function-transformation) suggested that as long as is “not small” (suggested as [here](https://mathematica.stackexchange.com/questions/278040/finding-the-mean-and-variance-of-a-distribution/278164#278164)) suggests that is a good approximation to the mean of the transformed random variable (on the log scale), which is close to what we observed.

I guess that this is probably somehow related to a Poisson distribution, but I don’t know or care enough to think more deeply about that.

# Including antigenic distance

We can introduce the antigenic distance into this simple model by taking

for some function . For the sake of simplicity, we assume that is **linearly dependent on distance**. Our model then becomes

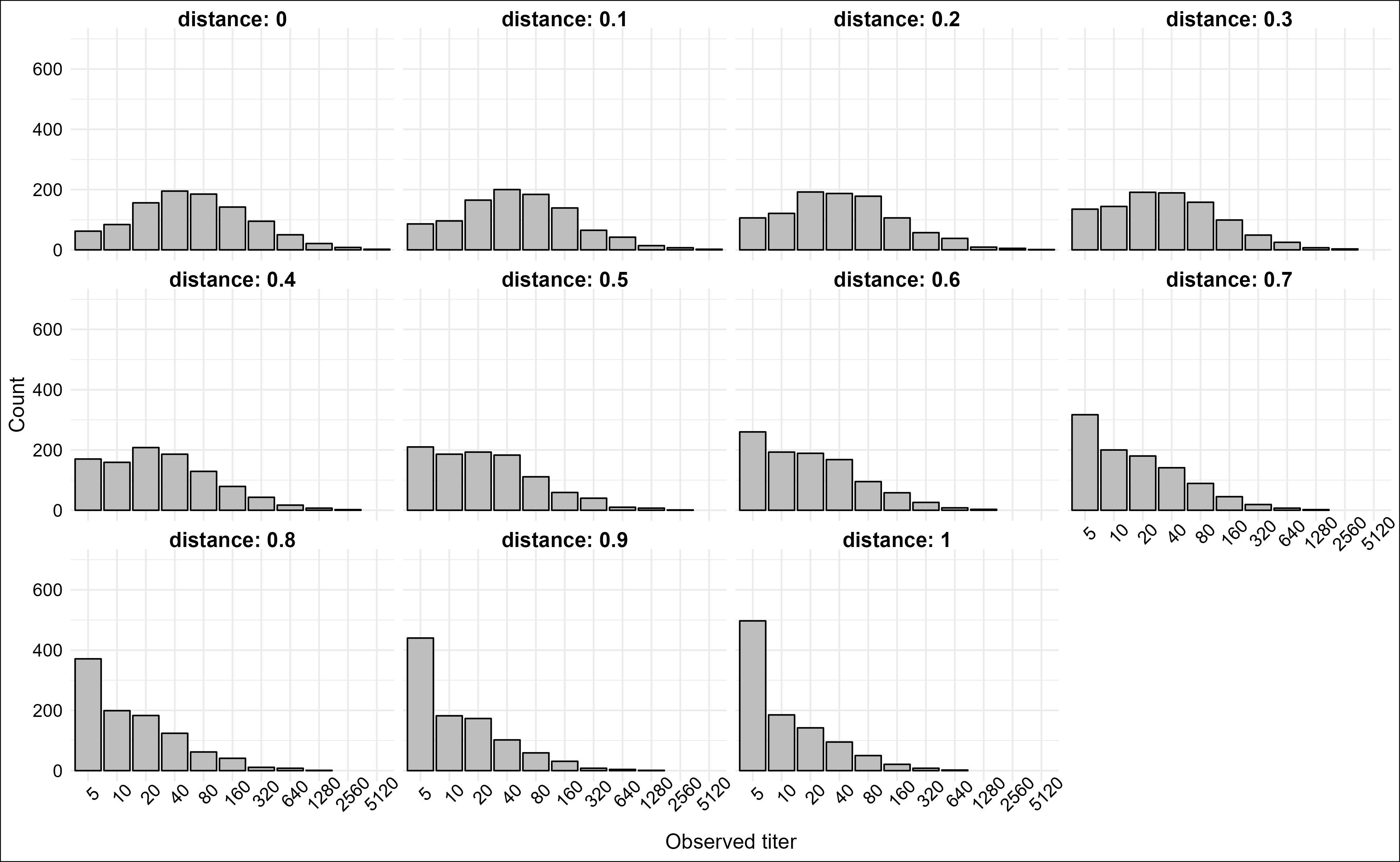
where . Under this model, represents the expected titer value when , which corresponds to the response to the homologous vaccine strain in real life. The slope, , should be negative, and thus represents the rate at which expected titers should decrease as strains become more distant. Furthermore, we can note that

Thus, since we can interpret as the difference between the expected titer for a completely novel strain and the expected titer for the homologous strain. Imposing the condition that when is thus equivalent to specifying that . This would be a standard regression problem with linear constraints with an analytic maximum likelihood solution if the true values were observed.

Anyways, as an example, consider the situation when and . For now, we leave the standard deviation as , as in the previous simulation.

set.seed(100)  
ex\_dist\_sim <-  
 tibble::tibble(  
 d = seq(0, 1, 0.1),  
 mu = 4 - 3 \* d  
 ) |>  
 dplyr::mutate(  
 sim = purrr::map(mu, \(x) one\_titer\_sim(1000, mean = x, sd = 2))  
 )  
  
library(ggplot2)  
plt <- ex\_dist\_sim |>  
 tidyr::unnest(sim) |>  
 dplyr::mutate(  
 sim\_titer = factor(sim\_titer),  
 mu = factor(mu) |> forcats::fct\_inorder(),  
 distance = factor(d) |> forcats::fct\_inorder()  
 ) |>  
 ggplot() +  
 aes(x = sim\_titer) +  
 geom\_bar(col = "black", fill = "gray") +  
 facet\_wrap(~distance, labeller = "label\_both") +  
 labs(  
 x = "Observed titer",  
 y = "Count"  
 ) +  
 zlib::theme\_ms() +  
 theme(  
 axis.text.x = element\_text(angle = 45),  
 plot.background = element\_rect(fill = "white")  
 ) +  
 coord\_cartesian(  
 ylim = c(0, 700)  
 )  
  
fn <- here::here("Misc", "Lab-Notes", "06-Generative-Sims", "p01.png")  
ggsave(  
 filename = fn,  
 plot = plt,  
 width = 13,  
 height = 8  
)

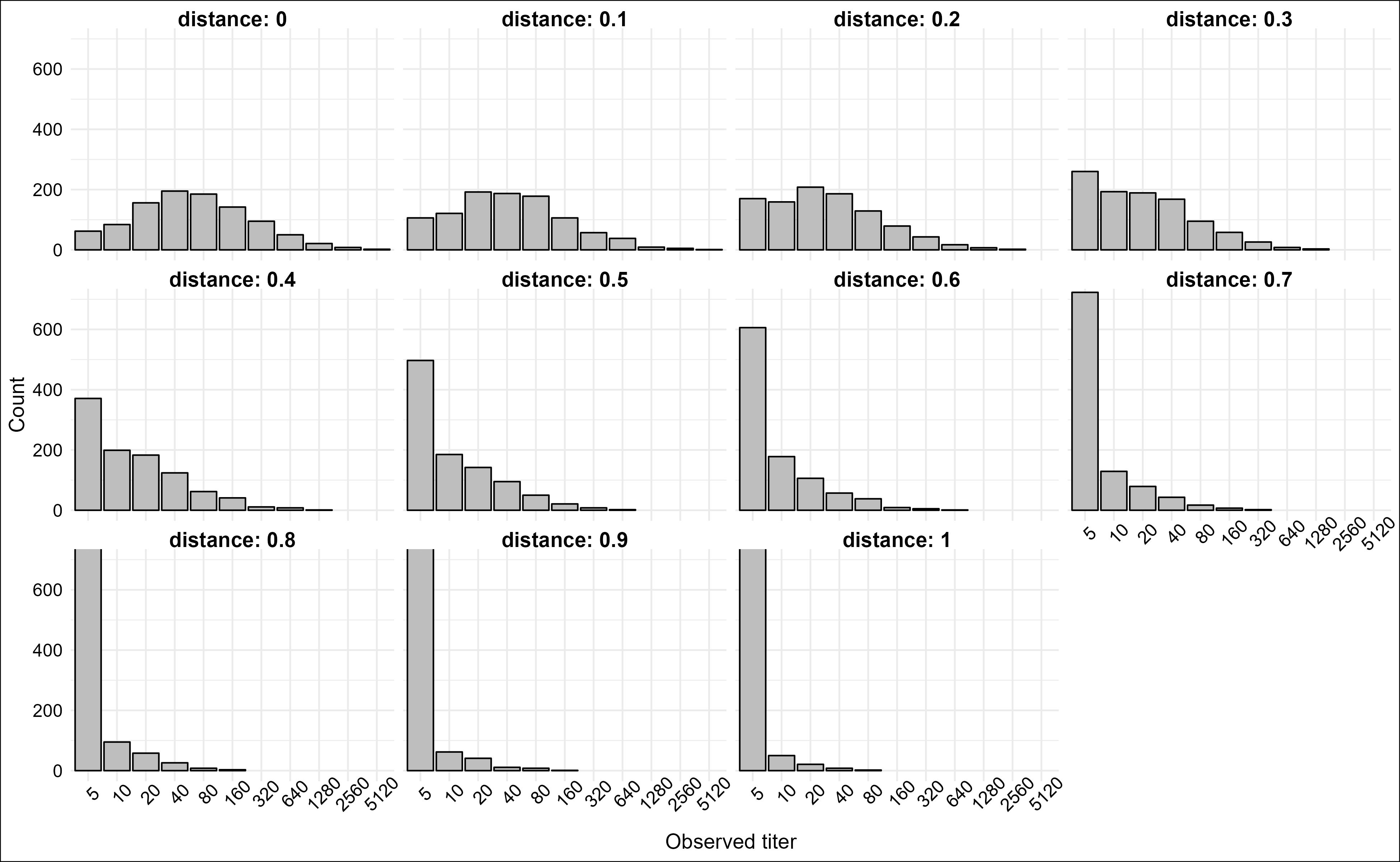
knitr::include\_graphics(fn)



We can also do a more extreme example where we assume which will cause the titers to decay towards the limit of detection rapidly.

set.seed(100)  
ex\_dist\_sim <-  
 tibble::tibble(  
 d = seq(0, 1, 0.1),  
 mu = 4 - 6 \* d  
 ) |>  
 dplyr::mutate(  
 sim = purrr::map(mu, \(x) one\_titer\_sim(1000, mean = x, sd = 2))  
 )  
  
plt <- ex\_dist\_sim |>  
 tidyr::unnest(sim) |>  
 dplyr::mutate(  
 sim\_titer = factor(sim\_titer),  
 mu = factor(mu) |> forcats::fct\_inorder(),  
 distance = factor(d) |> forcats::fct\_inorder()  
 ) |>  
 ggplot() +  
 aes(x = sim\_titer) +  
 geom\_bar(col = "black", fill = "gray") +  
 facet\_wrap(~distance, labeller = "label\_both") +  
 labs(  
 x = "Observed titer",  
 y = "Count"  
 ) +  
 zlib::theme\_ms() +  
 theme(  
 axis.text.x = element\_text(angle = 45),  
 plot.background = element\_rect(fill = "white")  
 ) +  
 coord\_cartesian(  
 ylim = c(0, 700)  
 )  
  
fn <- here::here("Misc", "Lab-Notes", "06-Generative-Sims", "p02.png")  
ggsave(  
 filename = fn,  
 plot = plt,  
 width = 13,  
 height = 8  
)

knitr::include\_graphics(fn)



# TODO

* Fit our bayesian model (doesn’t need to be partial pooling right now) to see if we recover parameters.
* Introduce noise to the model and see when parameters are recovered and how noise changes the estimates.
* Calculate the suite of metrics from the model fit under varying levels of noise and parameters and see how much the metrics vary under different settings.
* Need to discuss with Andreas what exactly to do next.

# Individual variations in the mean

Allow