Distance Sampling is Nifty!

Heather Gaya

March 18, 2022

What is Distance Sampling?

- We can never count every animal at a site, unless they are very very rare
- We know objects that are far away are harder to see than objects that are close by
- Distance sampling uses this idea to help us estimate abundance and density

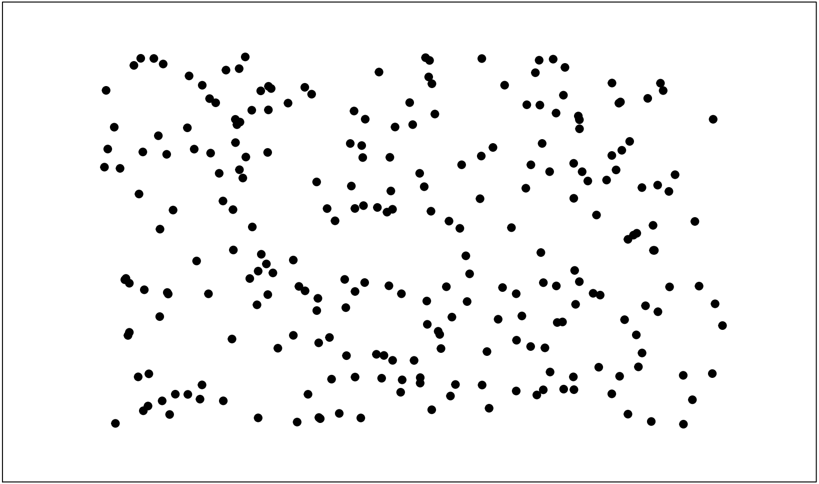
Distance Sampling Assumptions

Distance sampling makes some key assumptions!

- Animals are distributed independent of lines (or points)
- On the line, detection is certain (100% detection at distance 0)
- Distances are recorded correctly
- Animals don't move before detection (and no double counting of individuals)

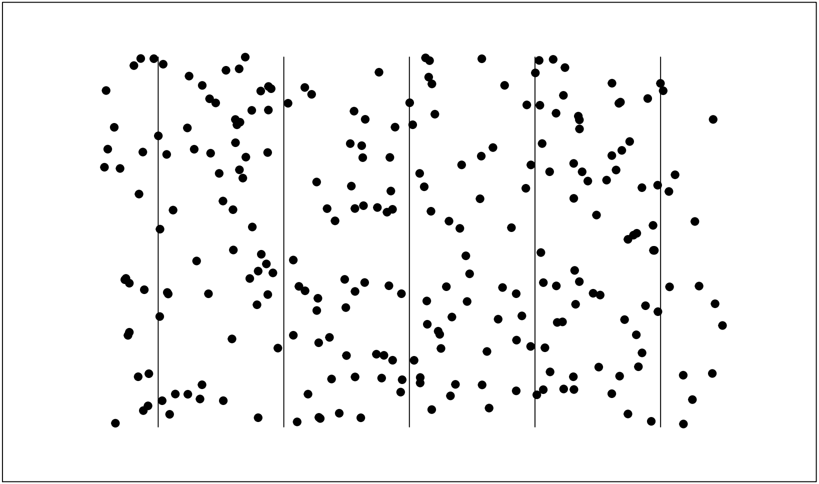
An Example

Let's say we want to know abundance on a plot. Each dot is an animal.

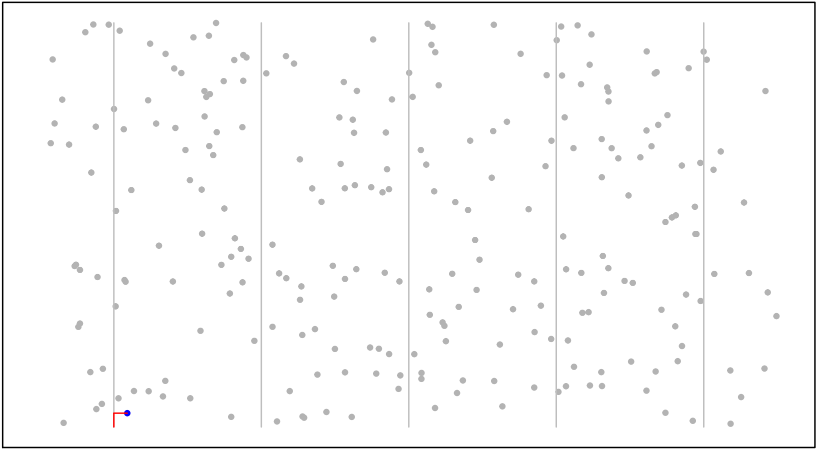


An Example

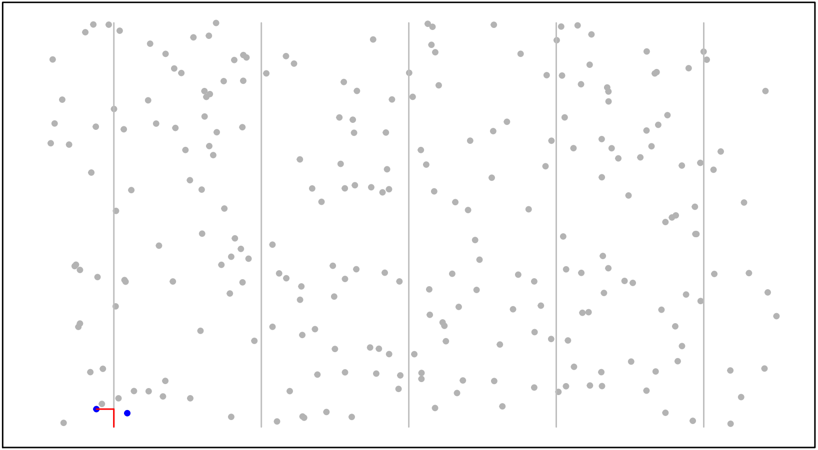
We want to have even coverage of the plot, so we put down some transects and have our observers walk (or fly, or boat, or drive, or whatever) on them



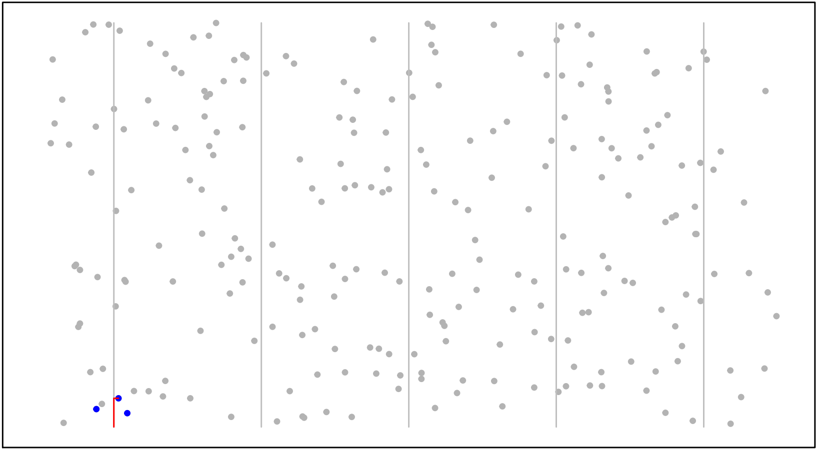
An Example



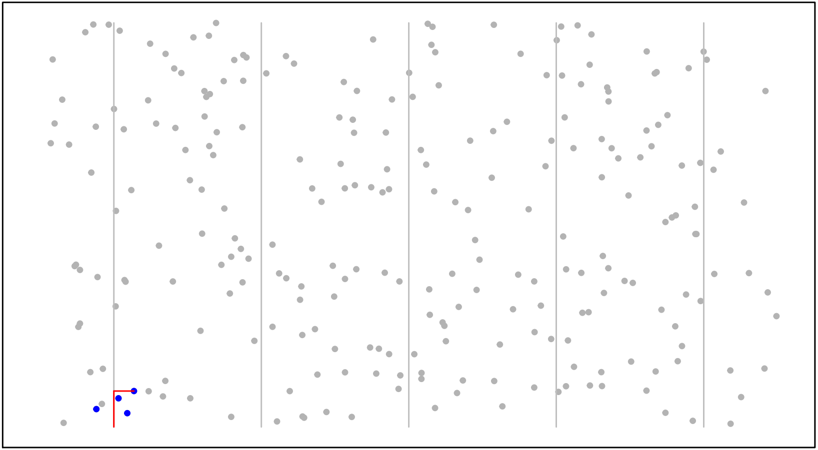
An Example



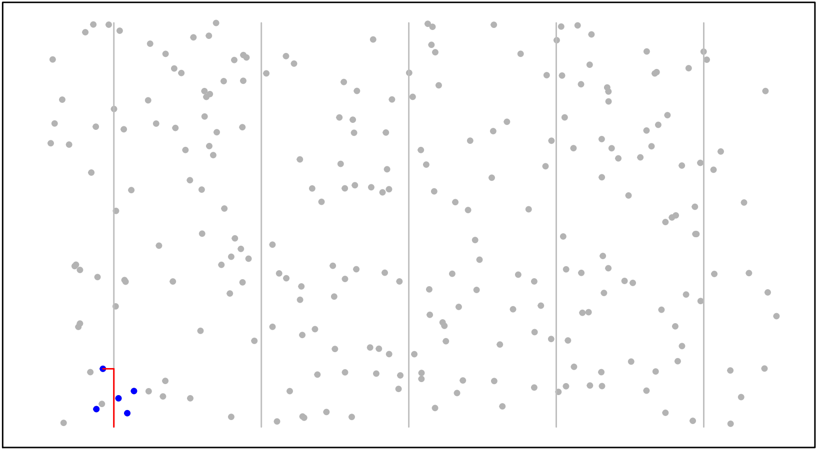
An Example



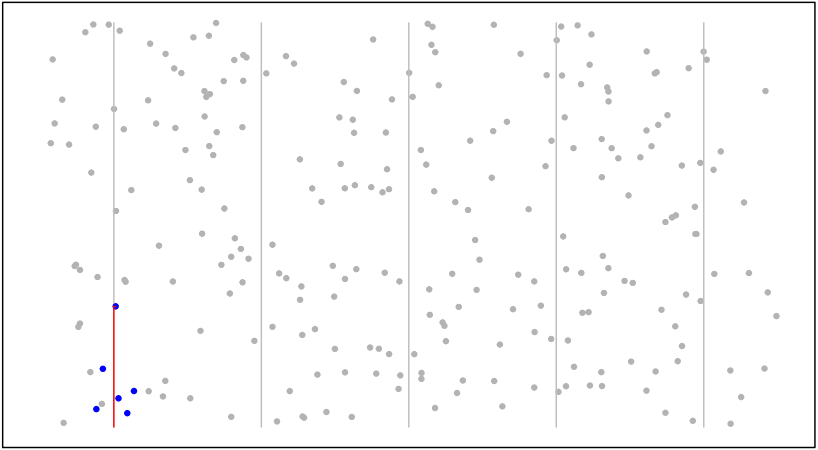
An Example



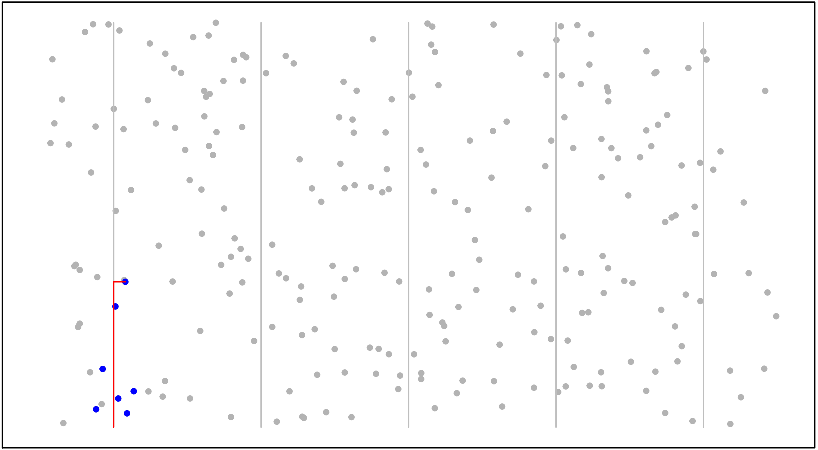
An Example



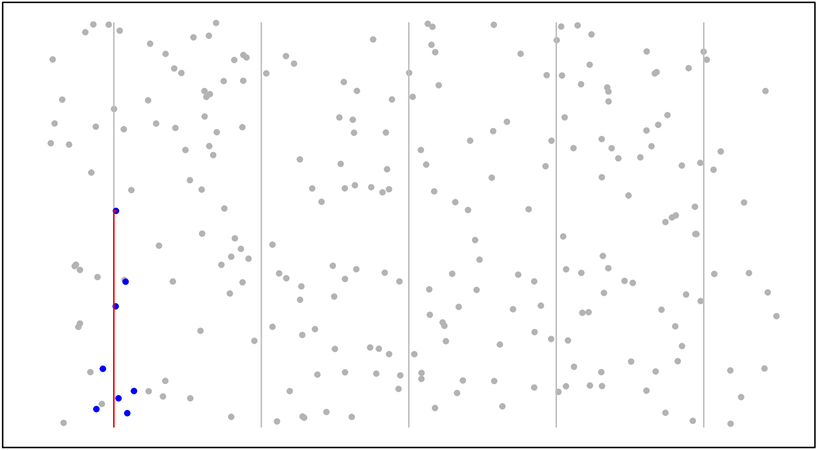
An Example



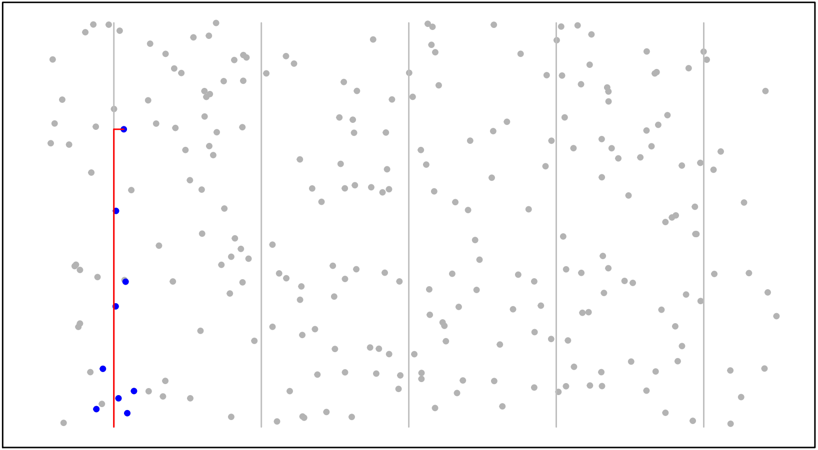
An Example



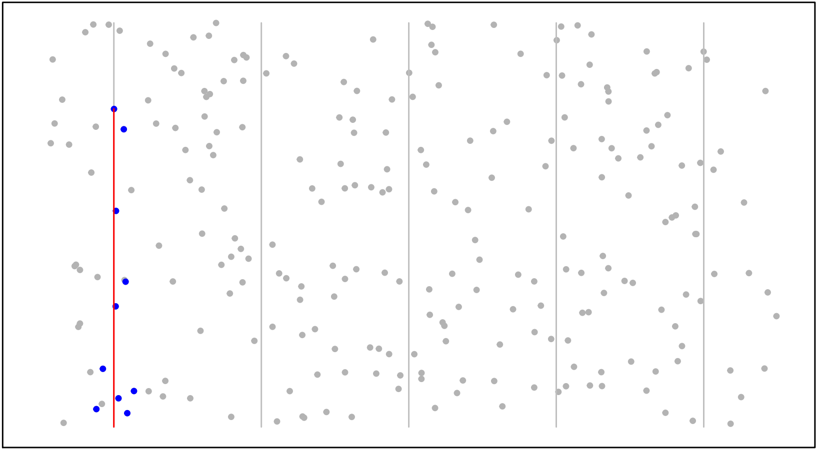
An Example



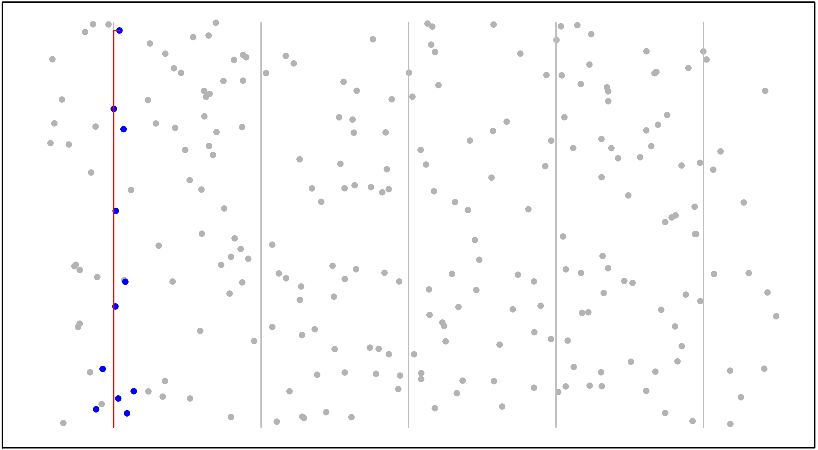
An Example



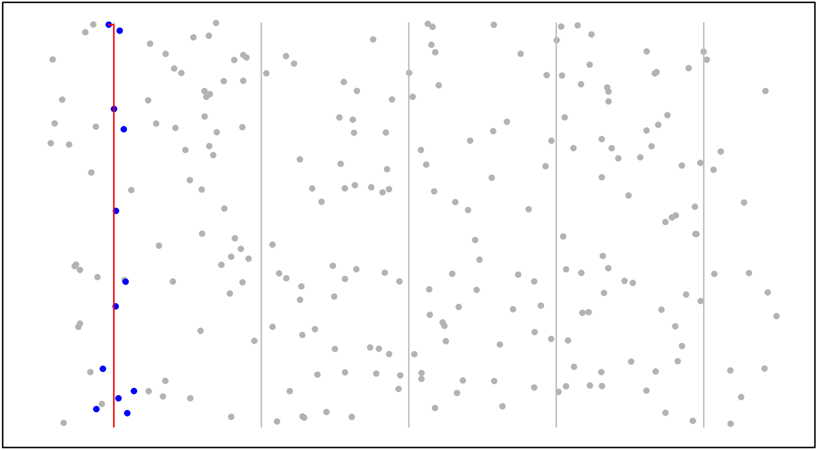
An Example



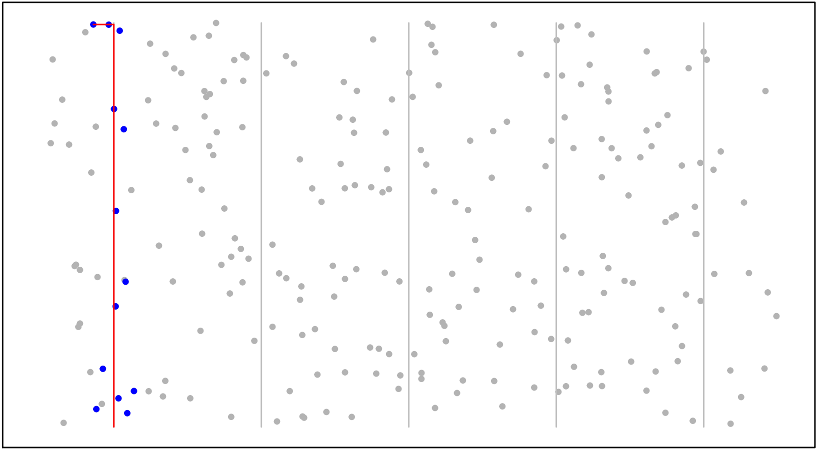
An Example



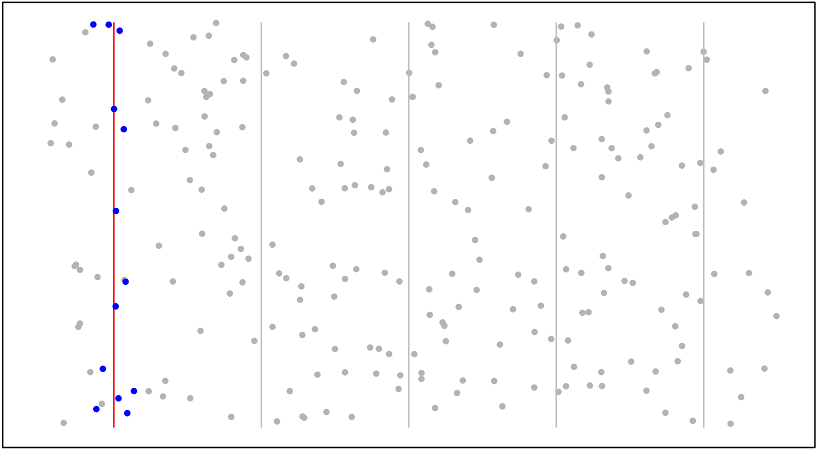
An Example



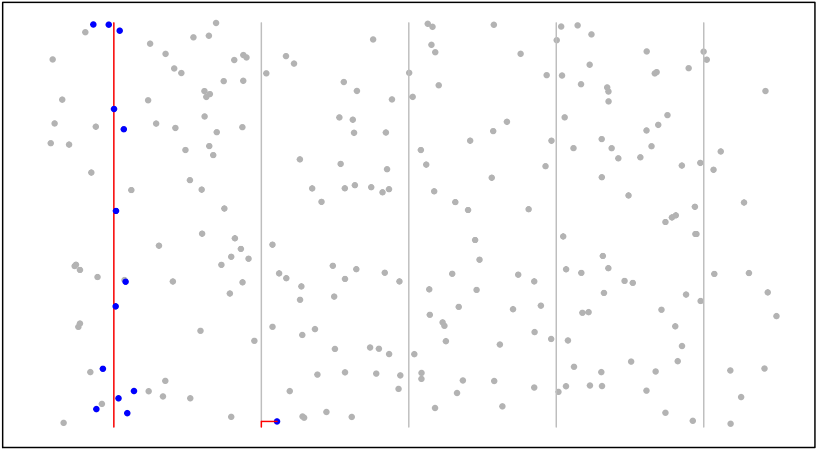
An Example



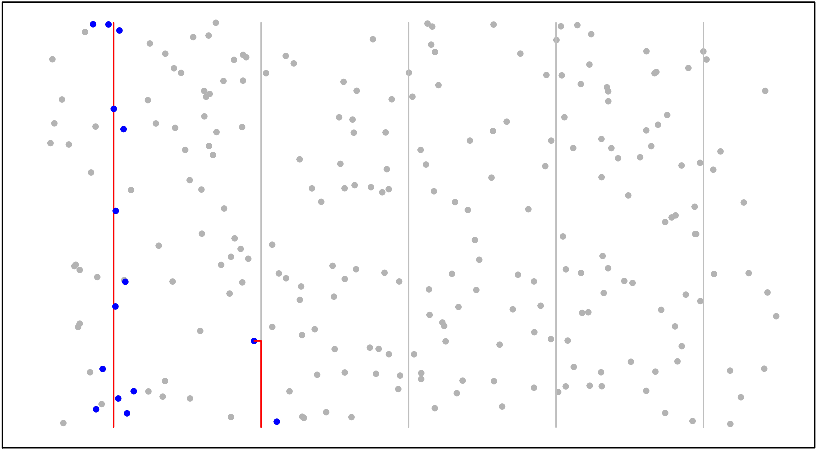
An Example



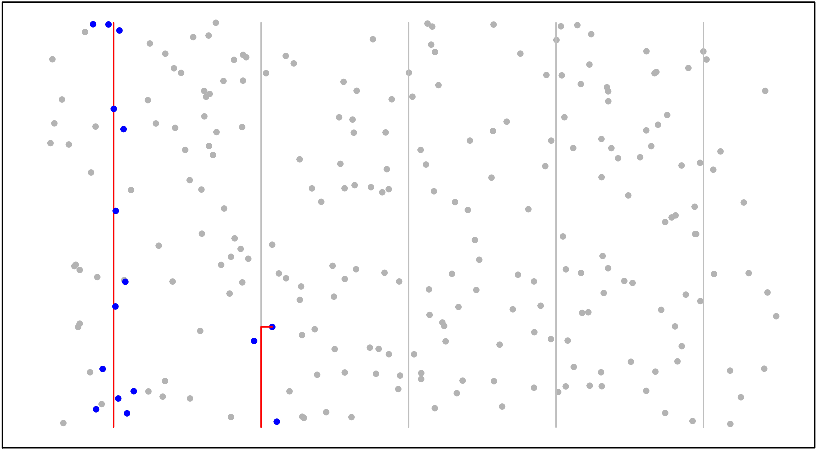
An Example



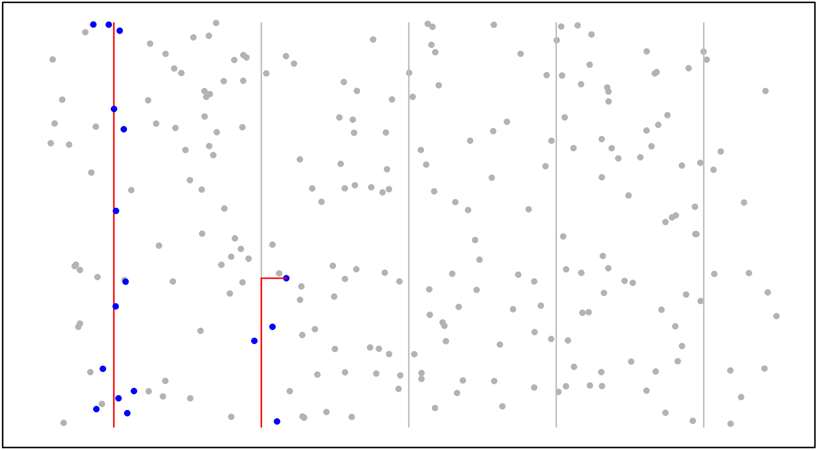
An Example



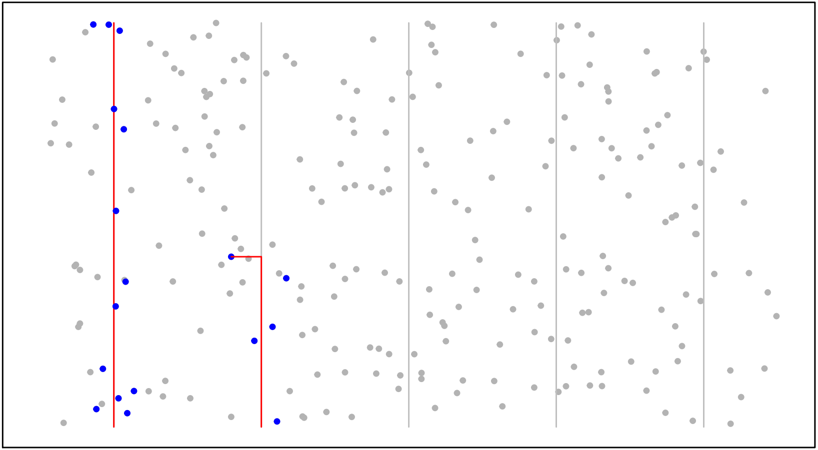
An Example



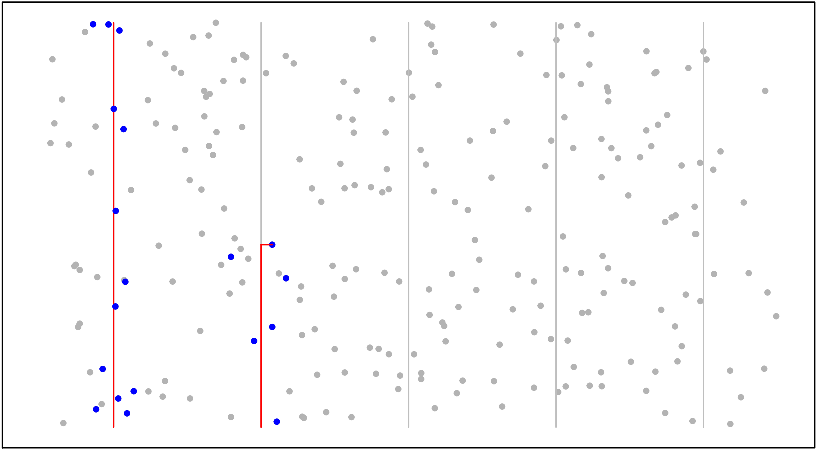
An Example



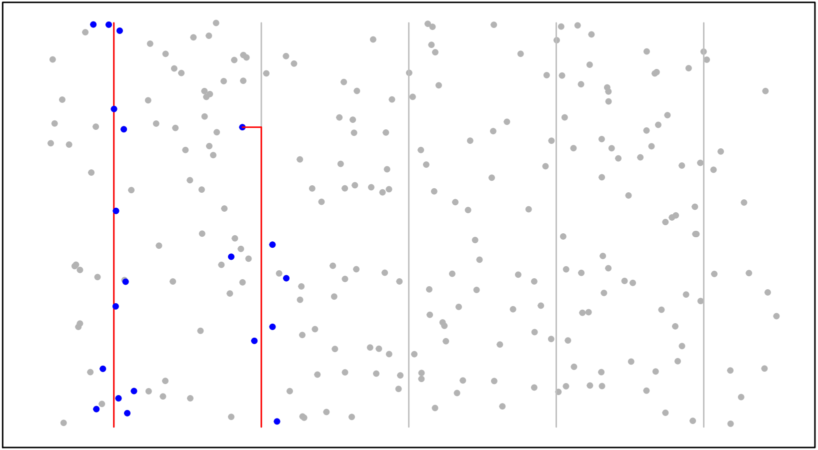
An Example



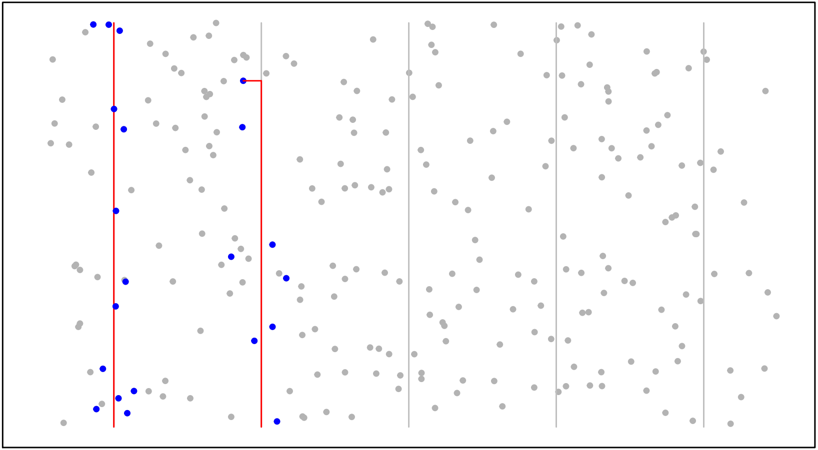
An Example



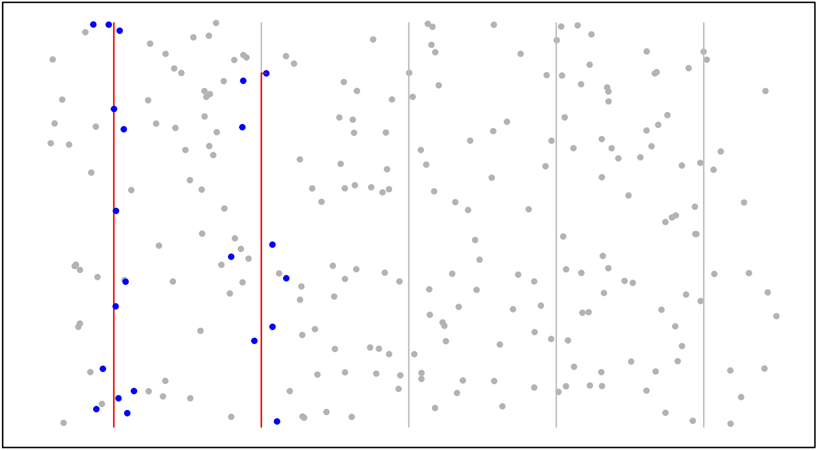
An Example



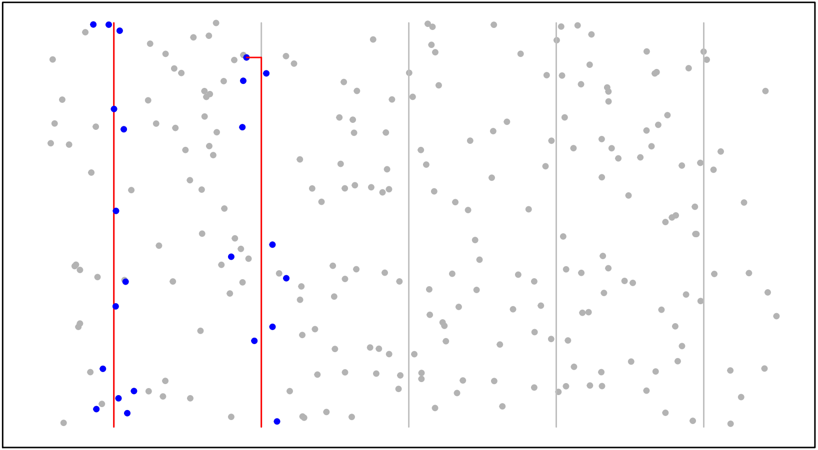
An Example



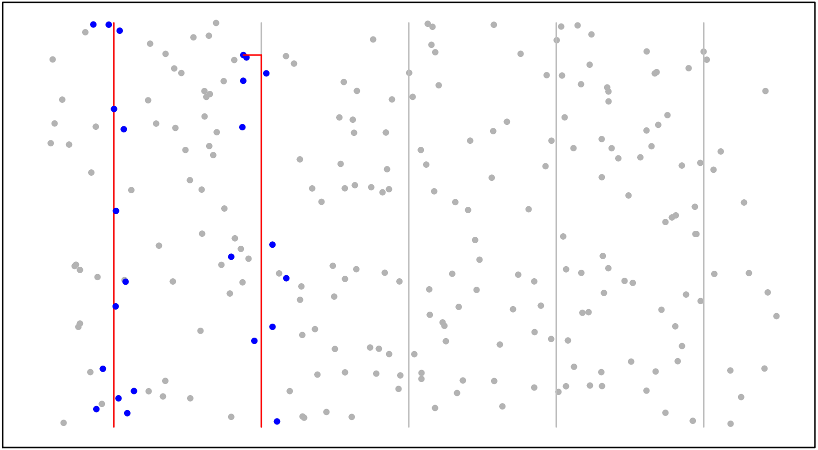
An Example



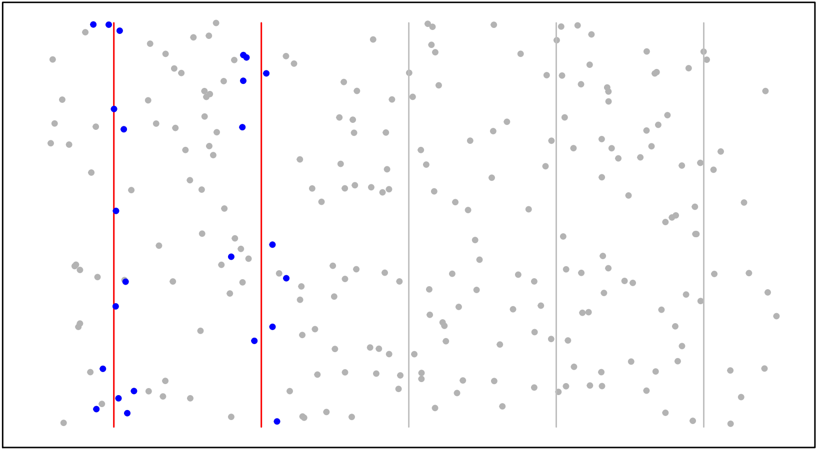
An Example



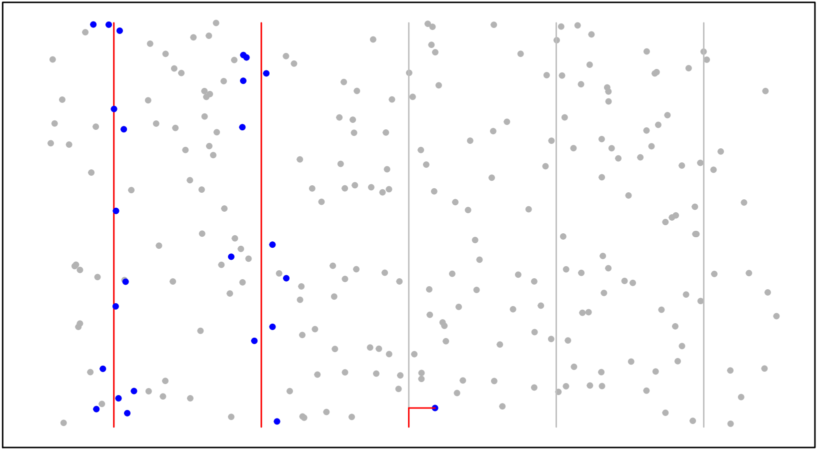
An Example



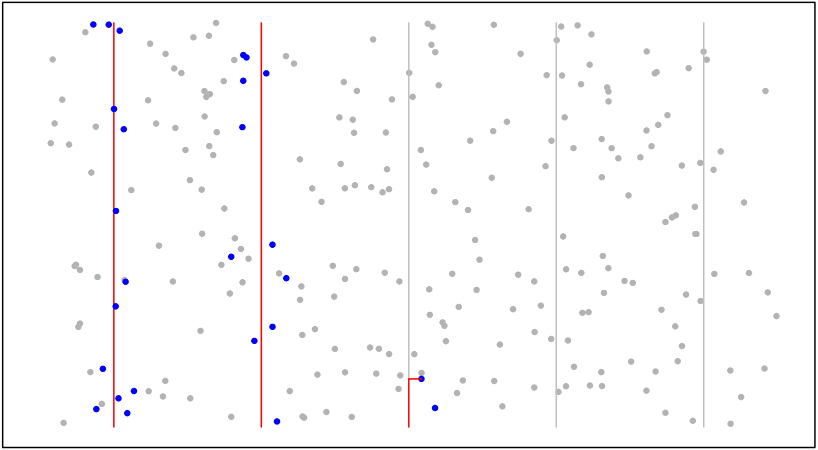
An Example



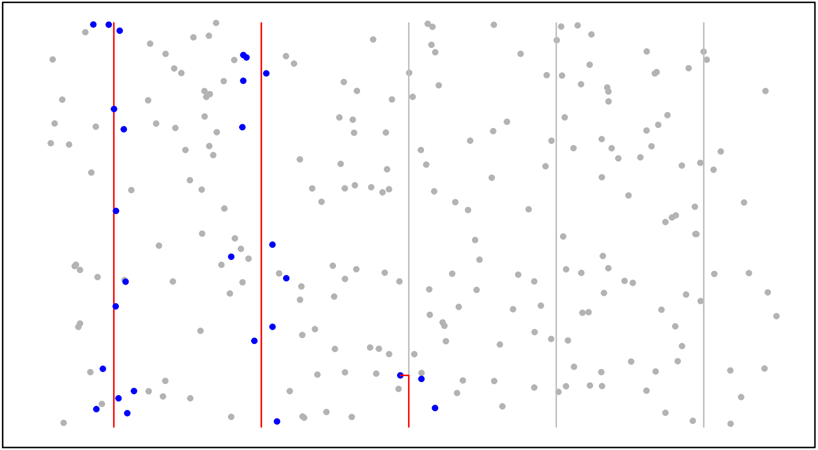
An Example



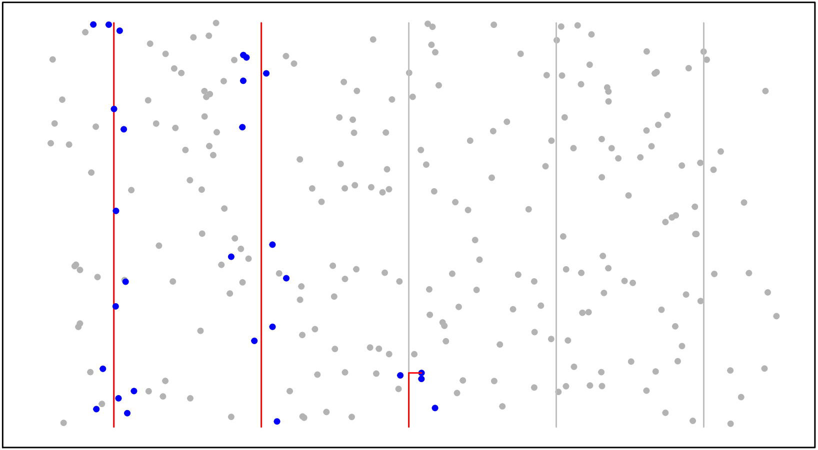
An Example



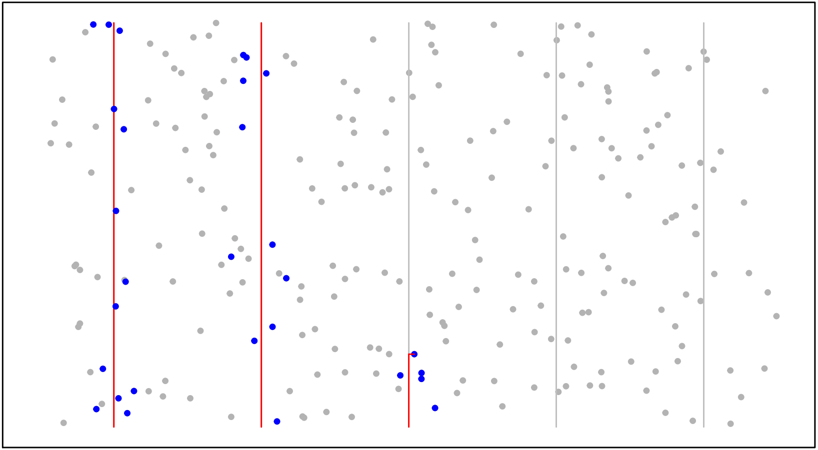
An Example



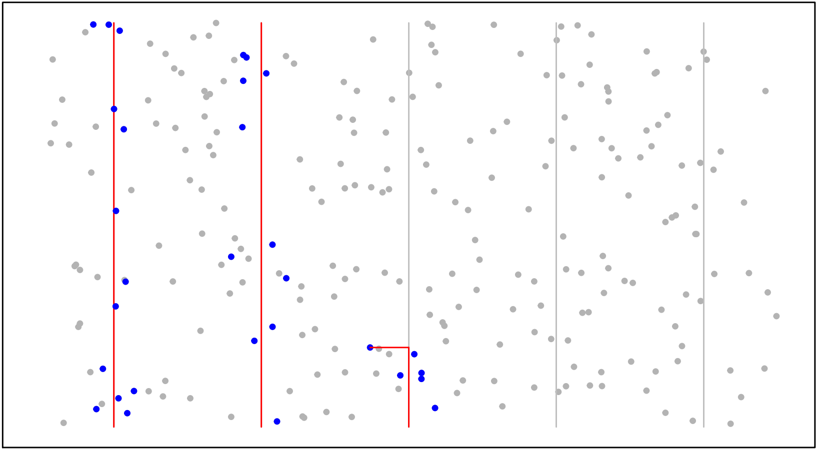
An Example



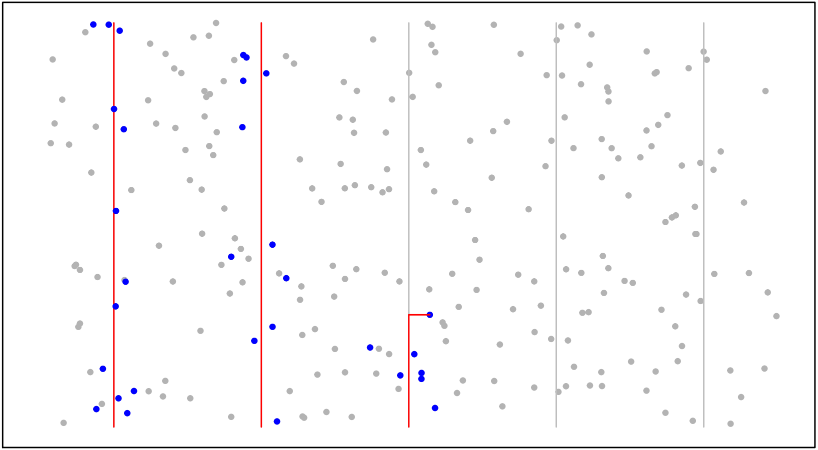
An Example



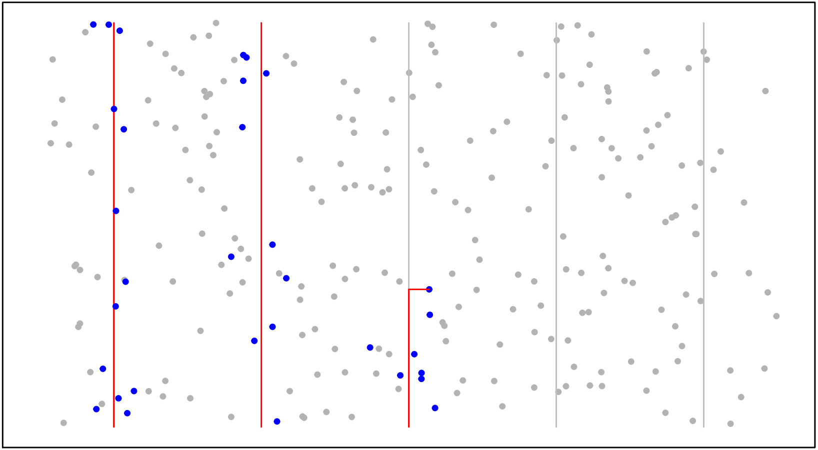
An Example



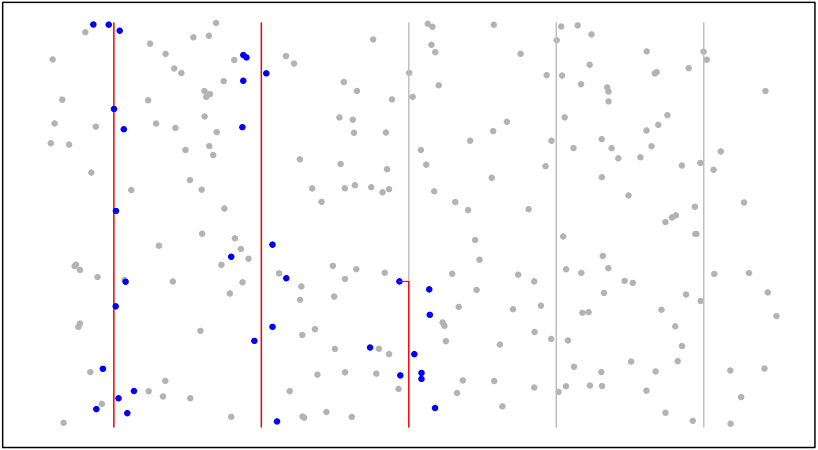
An Example



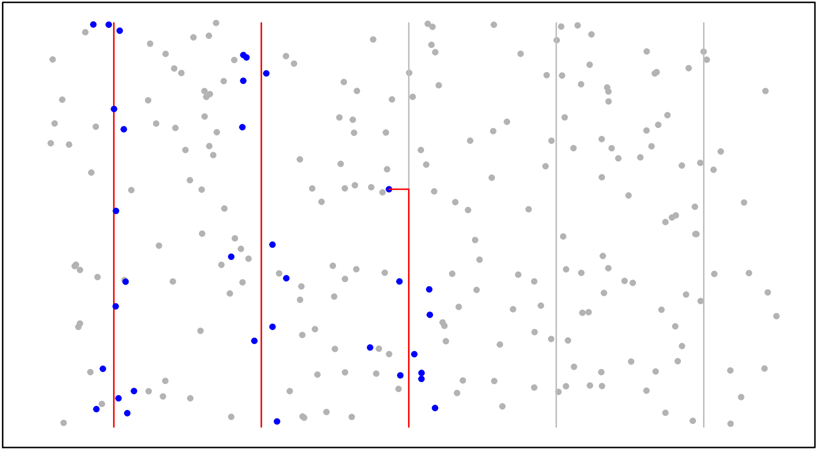
An Example



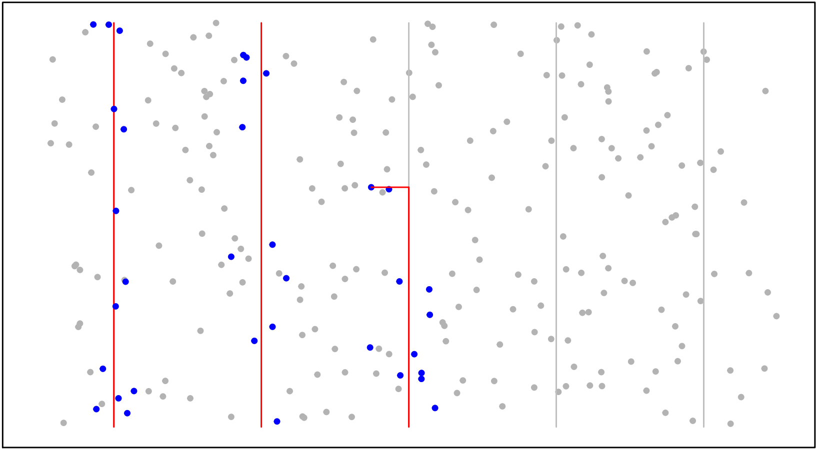
An Example



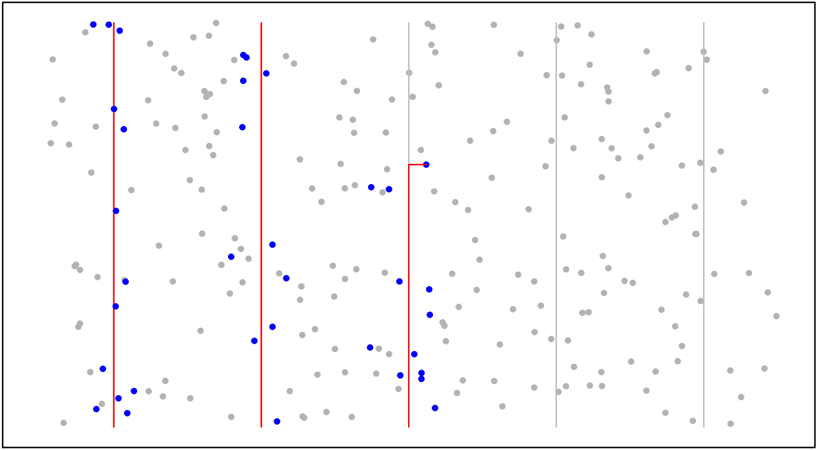
An Example



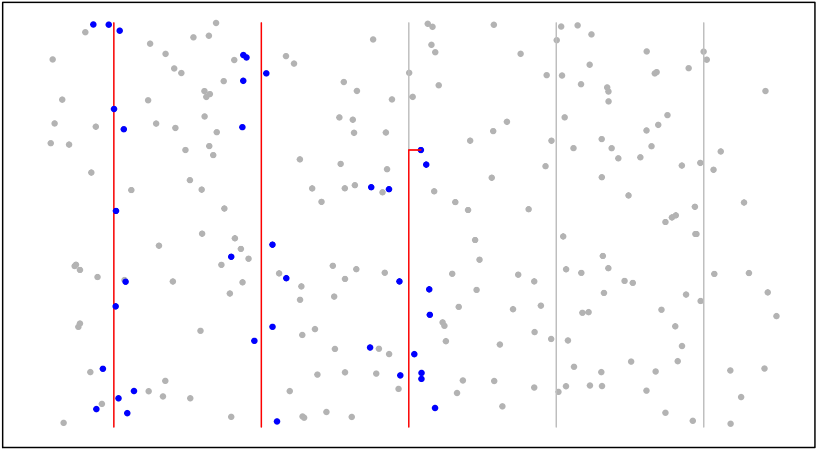
An Example



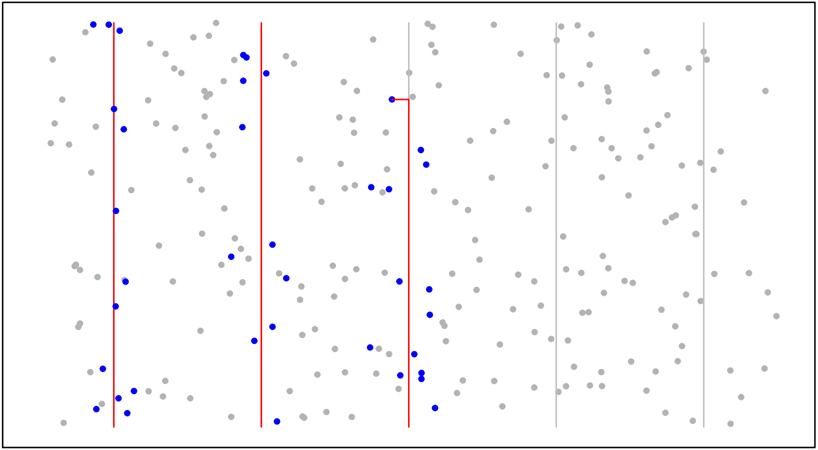
An Example



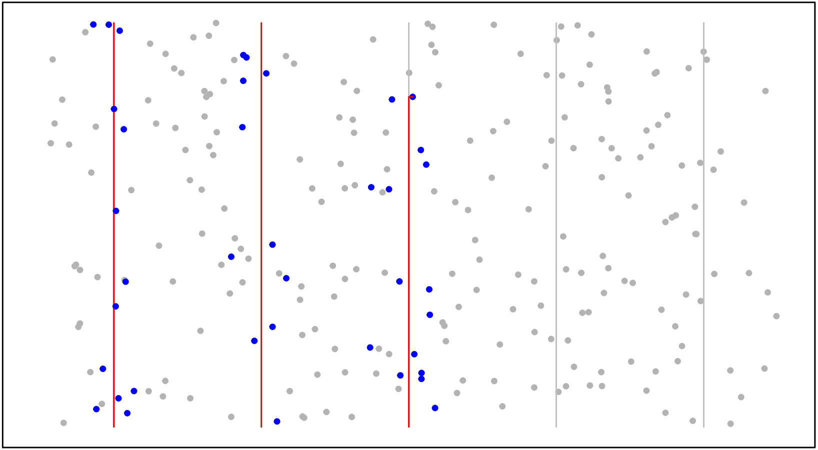
An Example



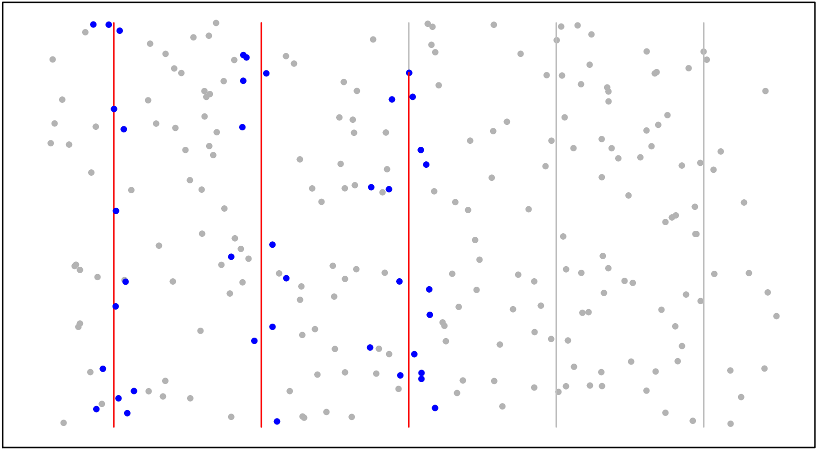
An Example



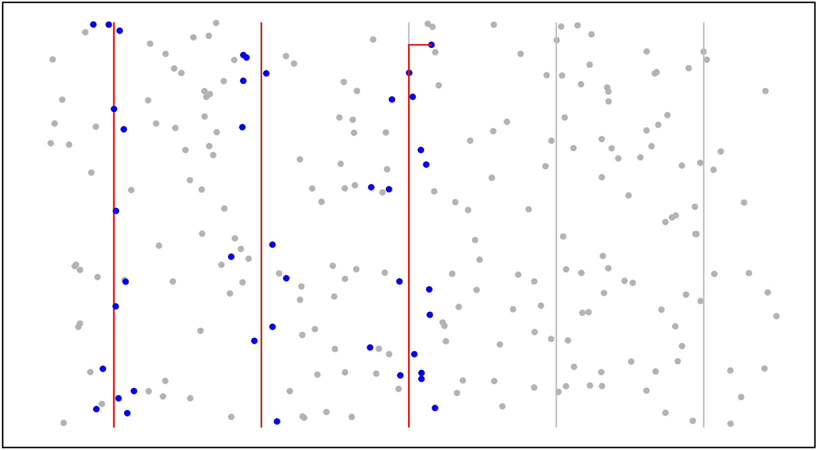
An Example



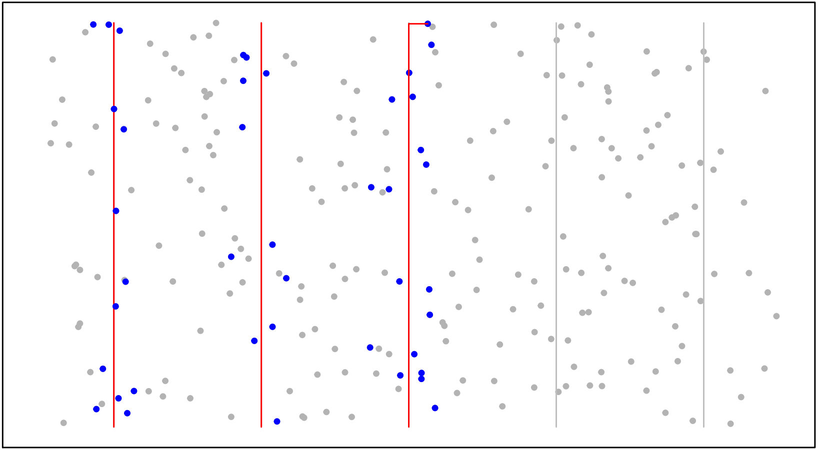
An Example



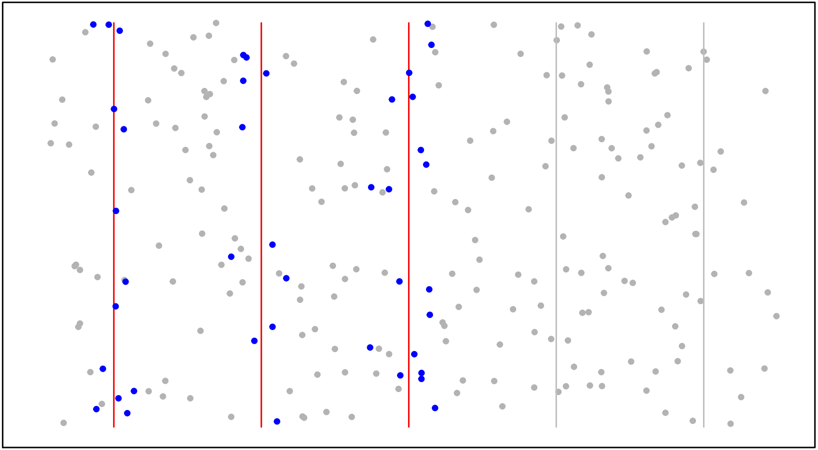
An Example



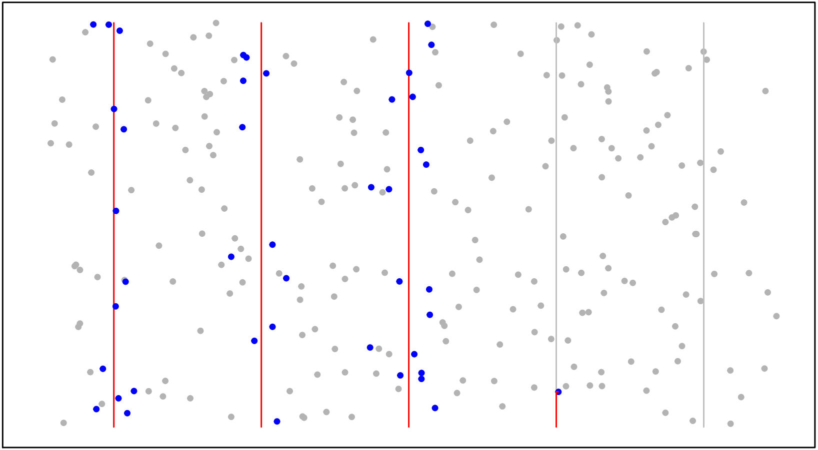
An Example



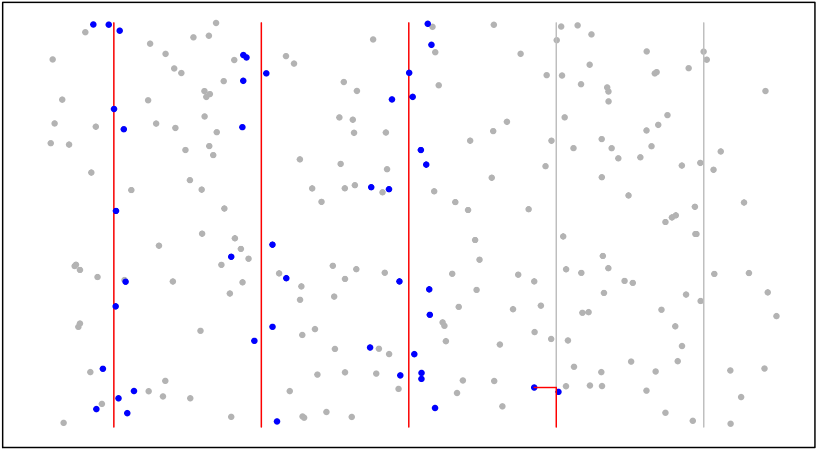
An Example



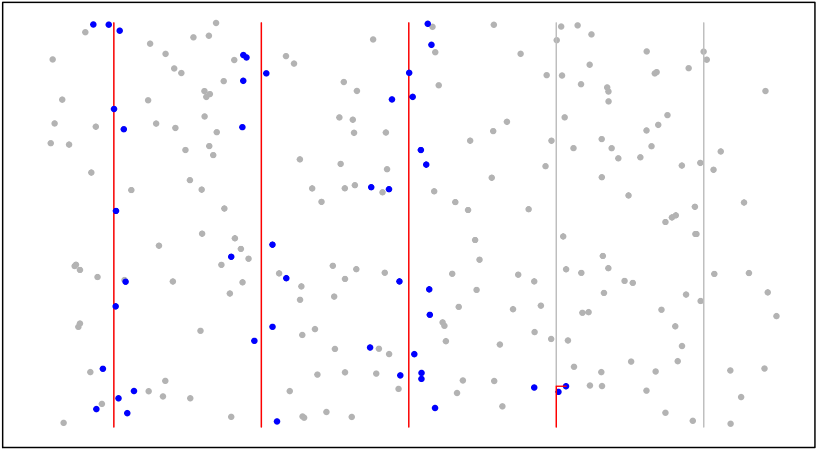
An Example



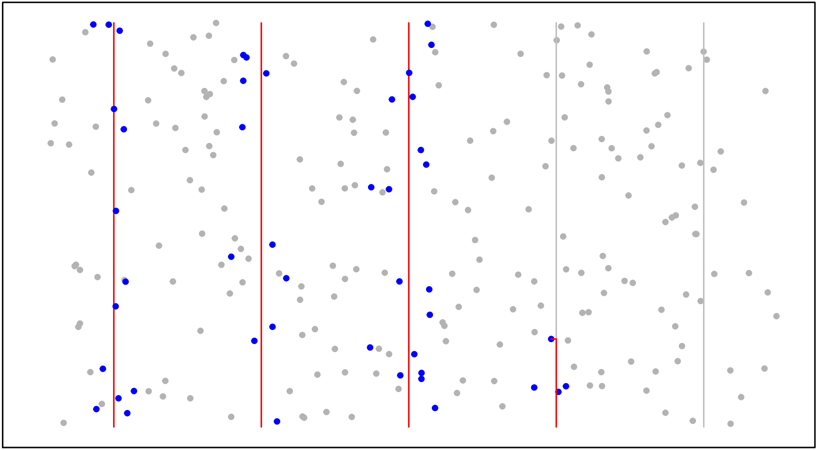
An Example



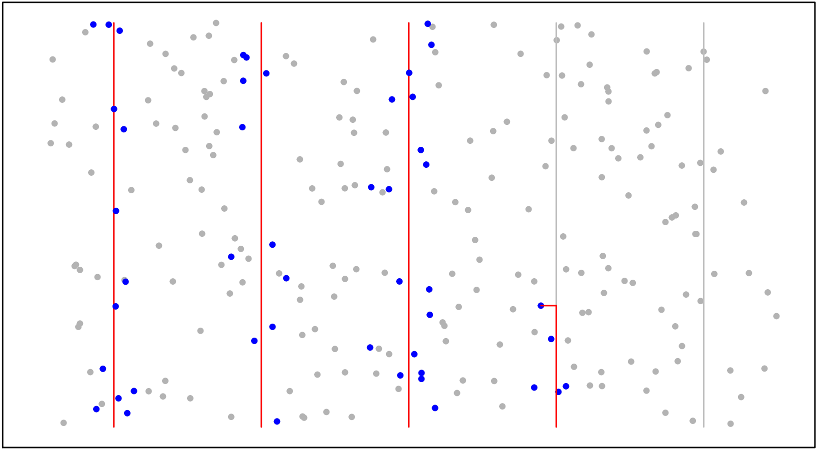
An Example



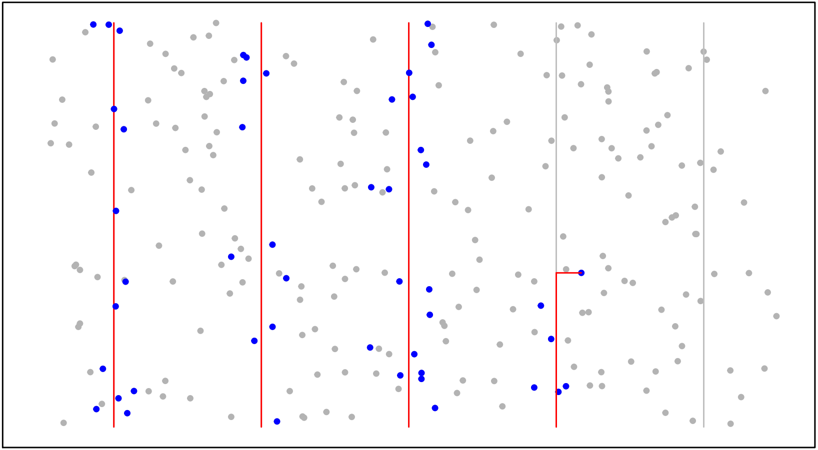
An Example



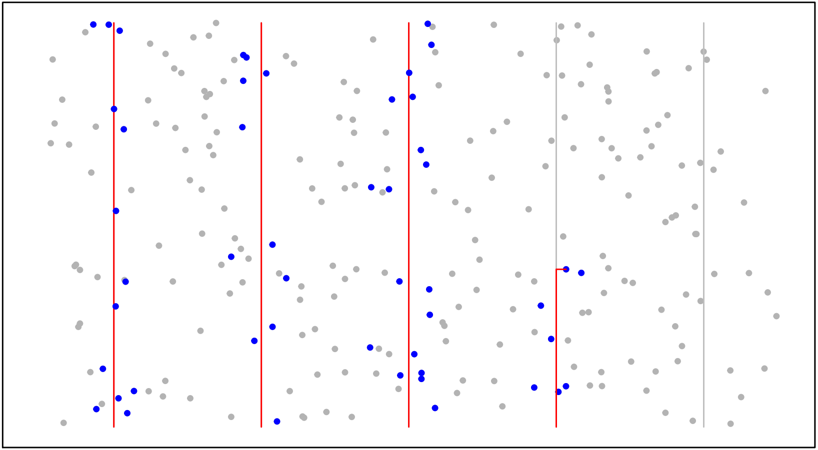
An Example



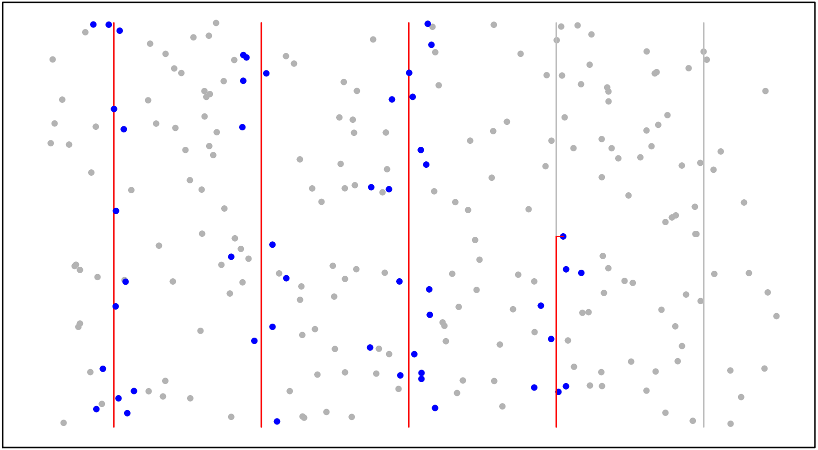
An Example



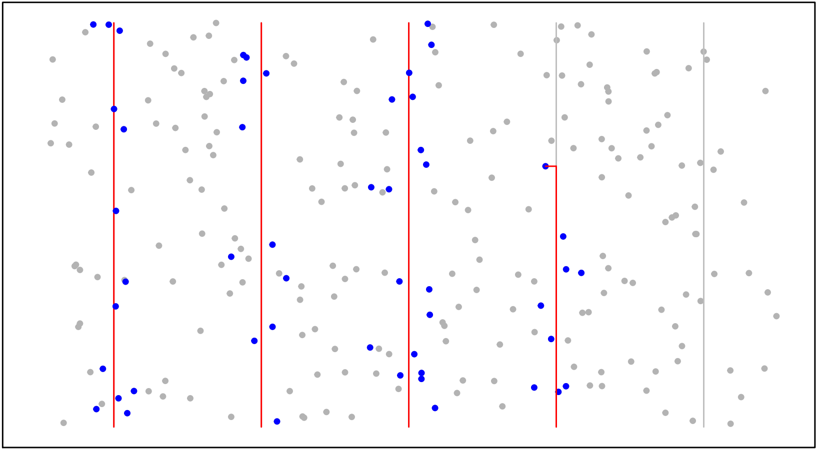
An Example



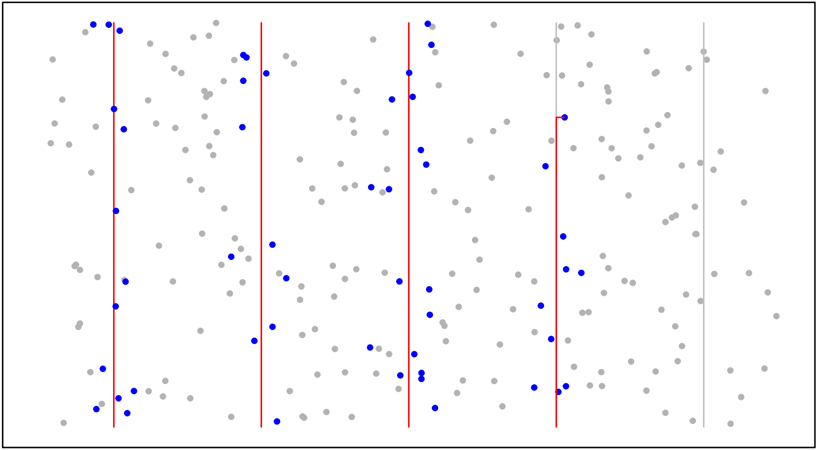
An Example



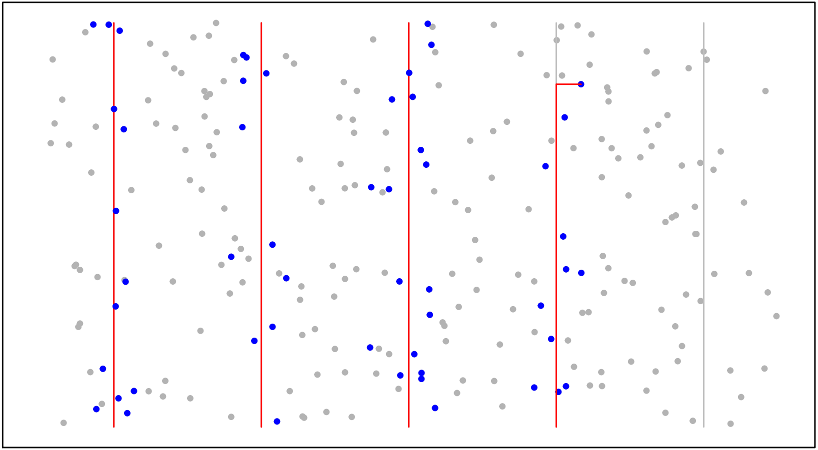
An Example



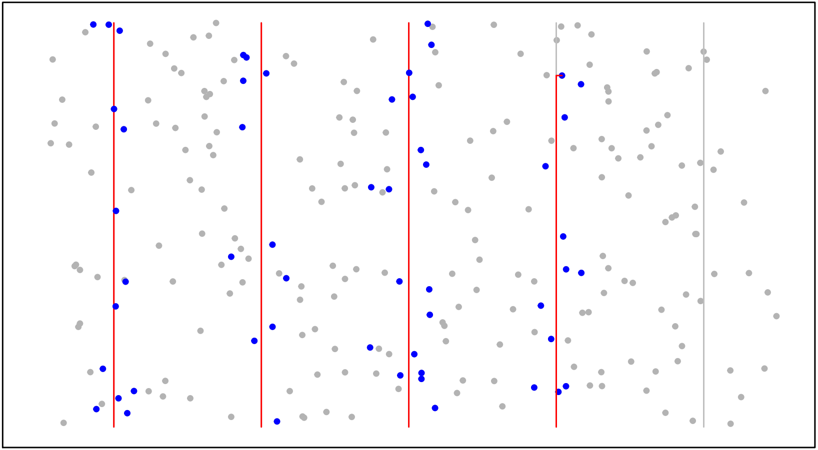
An Example



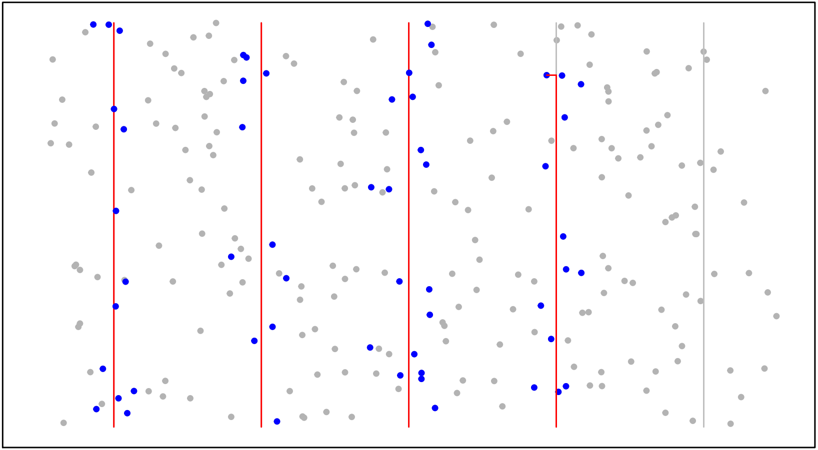
An Example



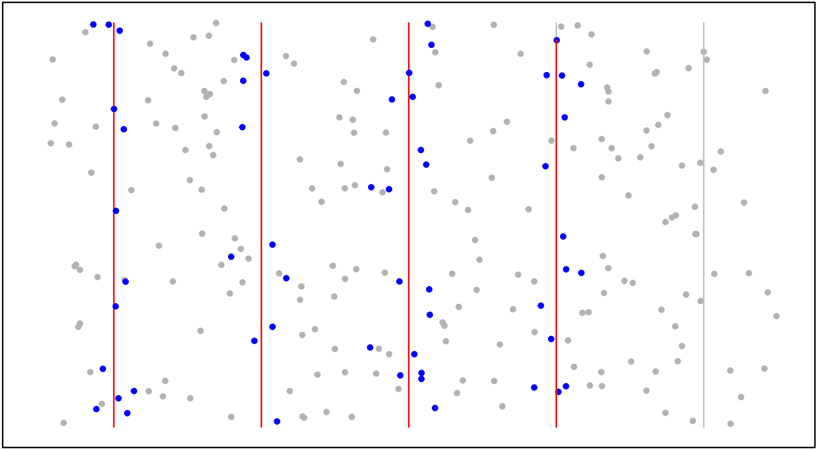
An Example



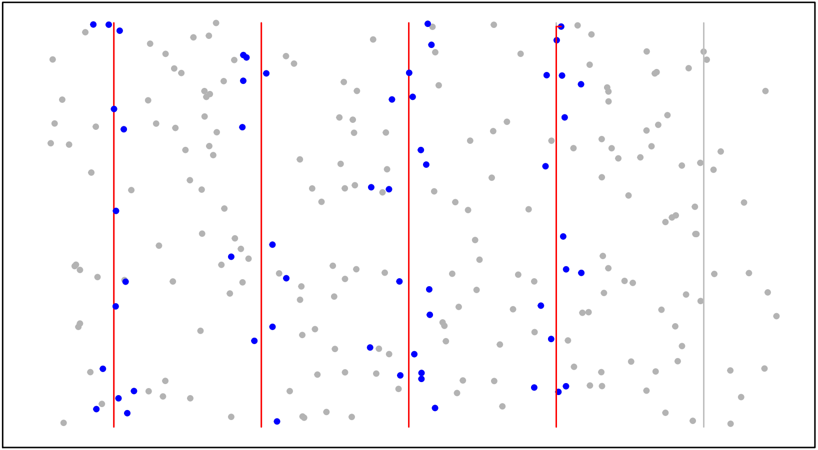
An Example



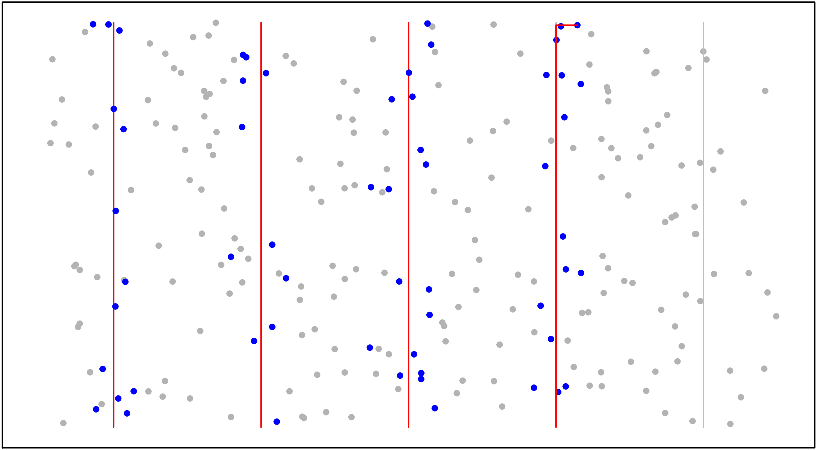
An Example



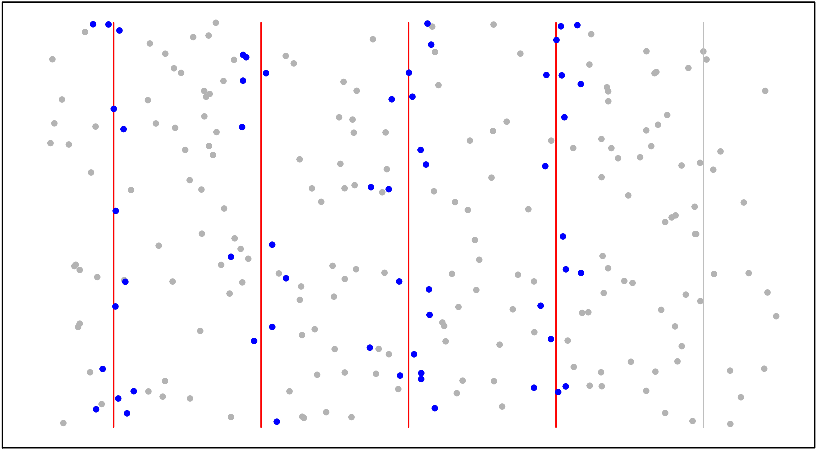
An Example



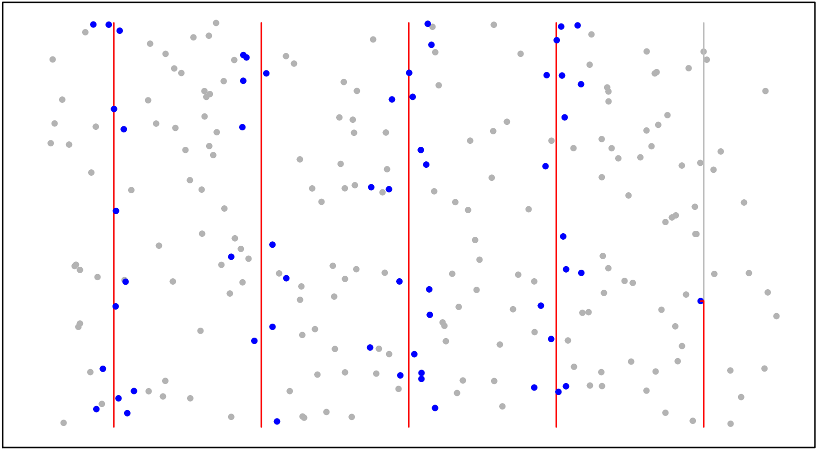
An Example



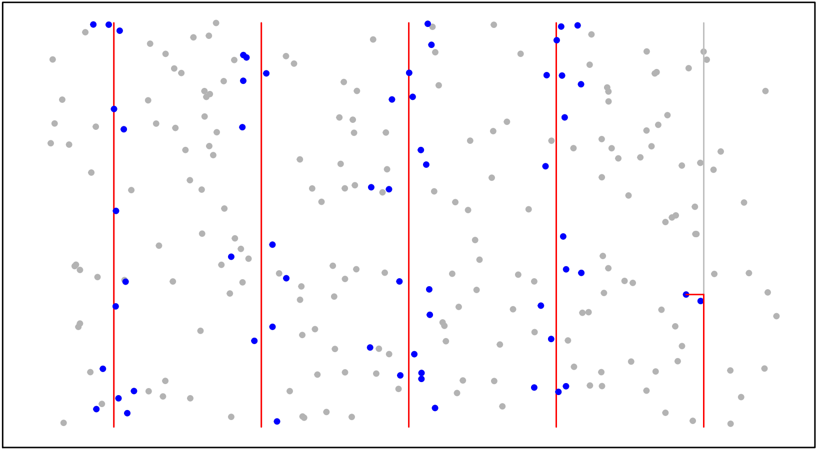
An Example



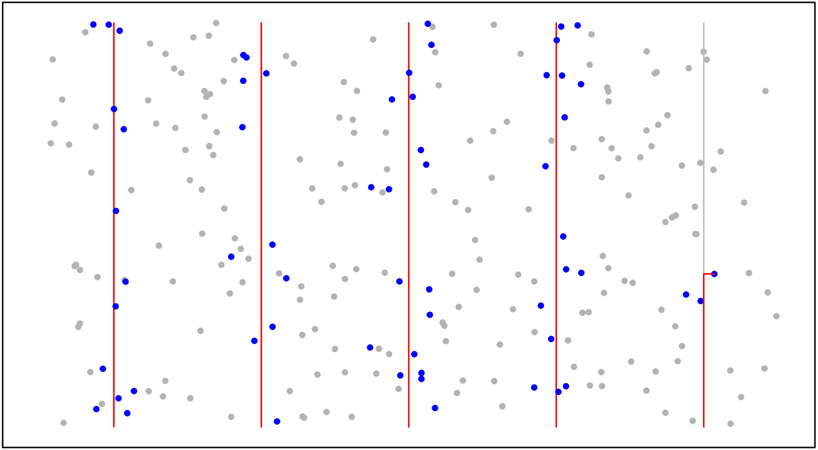
An Example



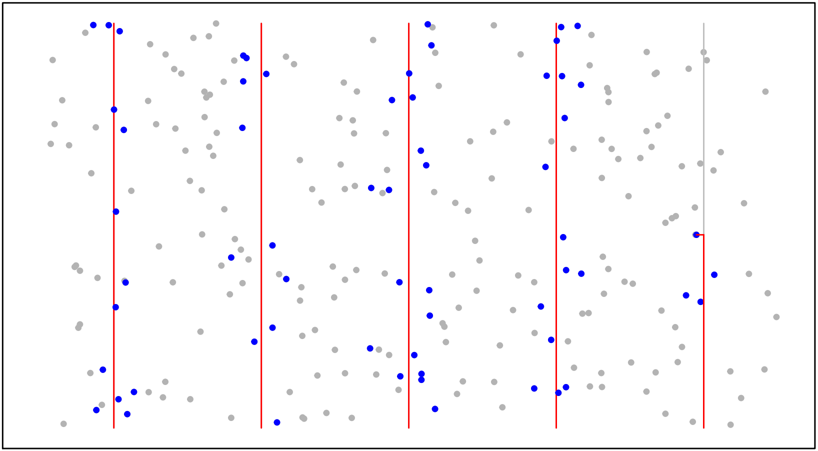
An Example



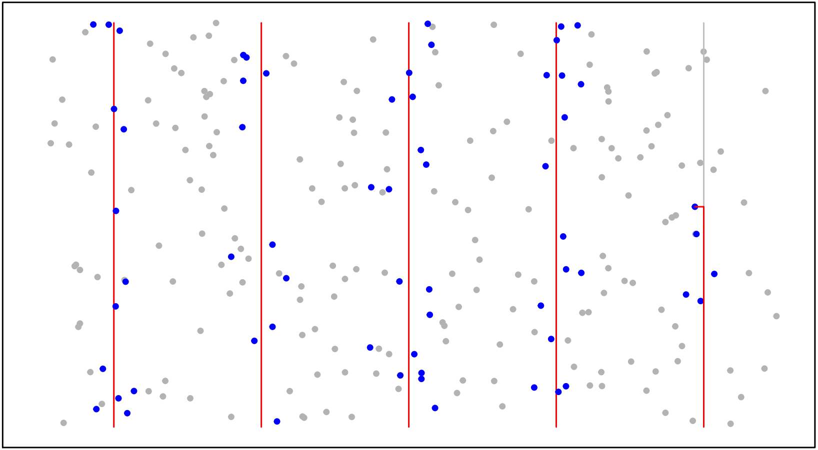
An Example



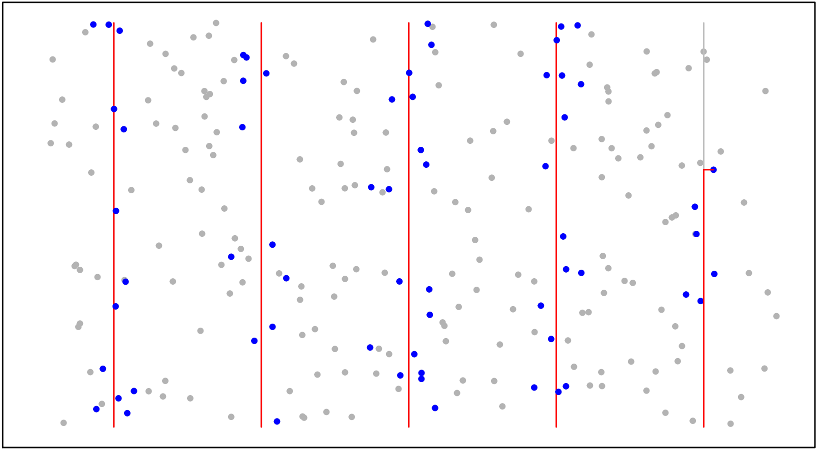
An Example



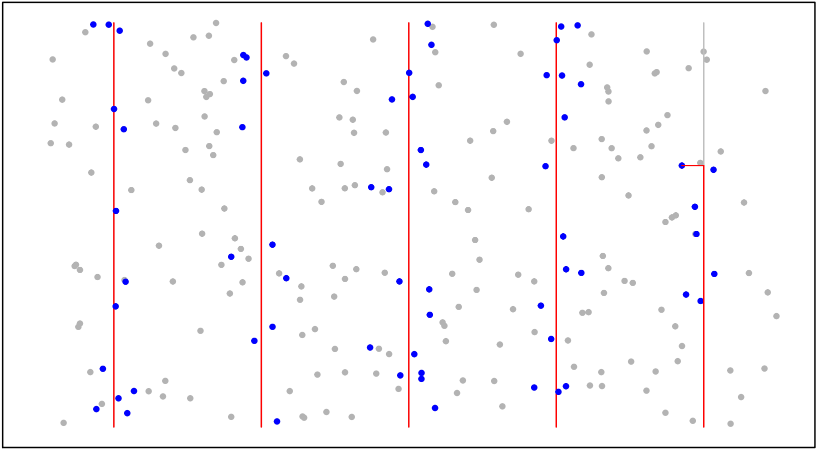
An Example



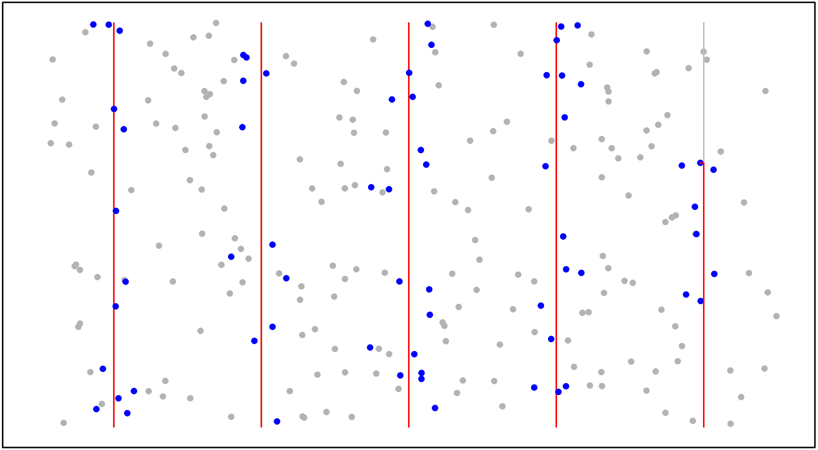
An Example



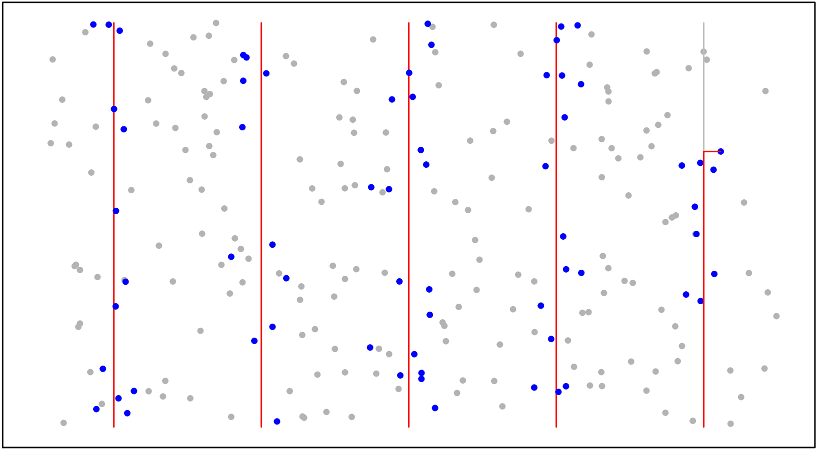
An Example



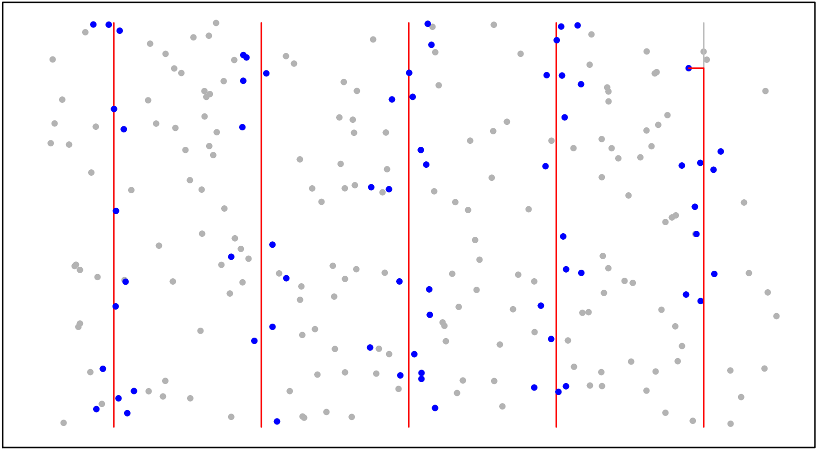
An Example



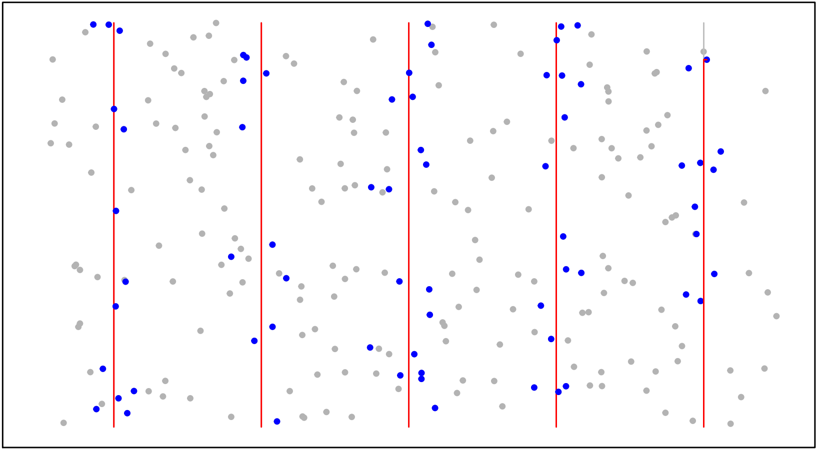
An Example



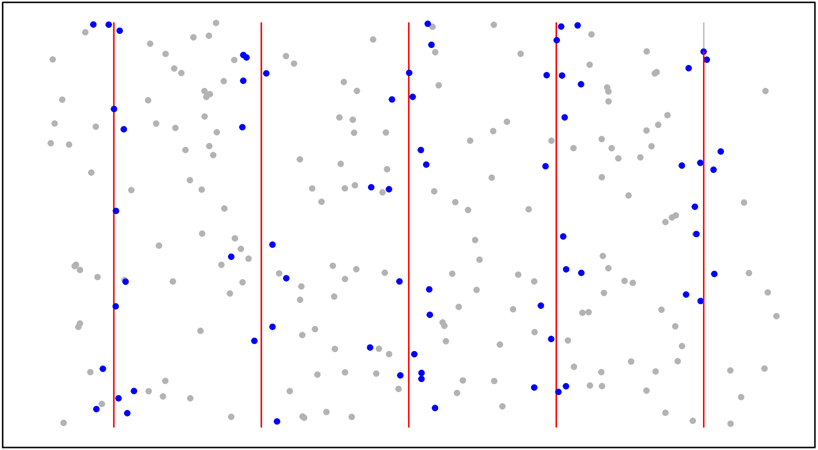
An Example



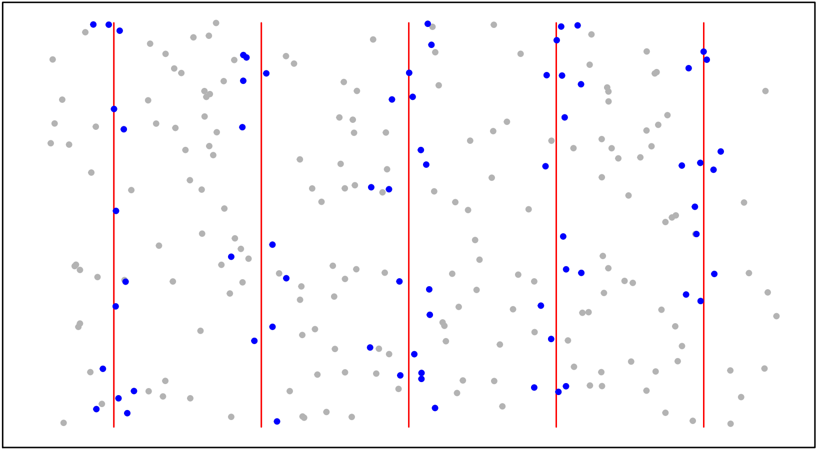
An Example



An Example

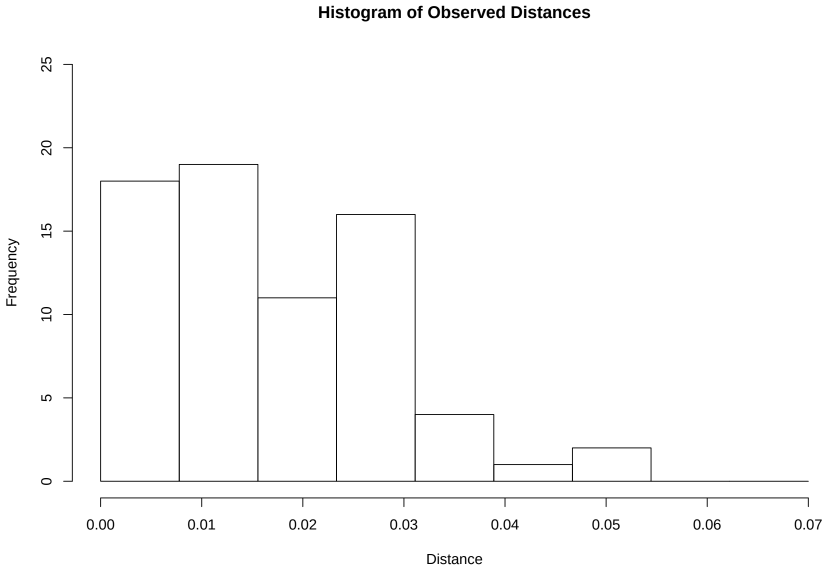


An Example



An Example

You'll notice this sample isn't random - we clearly found more animals near the transects than far away.



The Math

If we can quantify this bias, we can estimate our "effective" search area, which gives us density.

$$D = \frac{n}{a}$$

$$N = DA$$

where n = number of animals in your searched area,

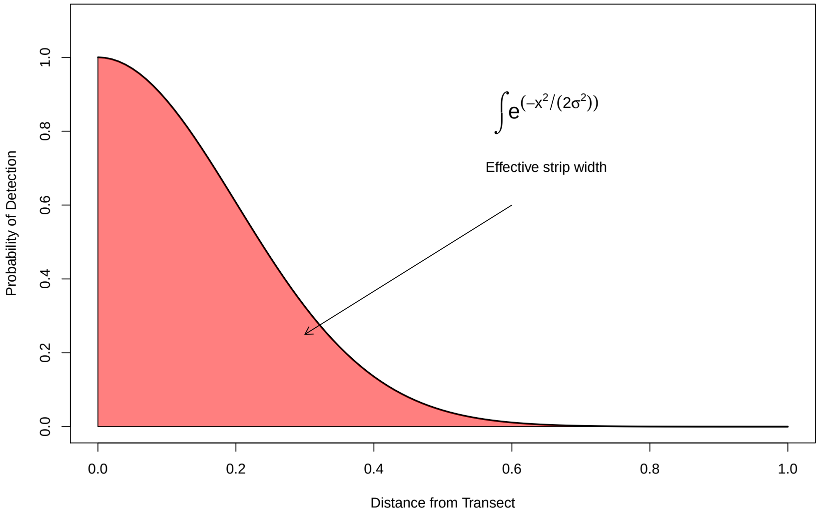
N = true abundance (derived),

A = total area of interest,

and a = effective area searched

The Math (Cont')

How do we estimate this? Integrals!



Alternative Math

Alternatively, we can reformat the problem so that WE define the area and ask the model to tell us how many animals are in the area we choose. Our equation is now:

$$D = \frac{N}{L(2W)}$$

where N is the true abundance ($n +$ some unknown quantity of undetected animals)

L is the length of transects walked

and W is the width of interest (generally, $1/2$ spacing between transects)

Distance Sampling - Gopher Tortoises!

For my MS, we wanted to improve distance sampling techniques for Gopher Tortoise surveys throughout the Southeast.

Tortoise burrows make great sedentary objects for distance sampling!

Even Clint Went Outside

Burrows can range in size quite dramatically, so we also wanted to include the problem of size variation in our model.

Gopher Tortoise Model Goals

We needed our model to do (4) things:

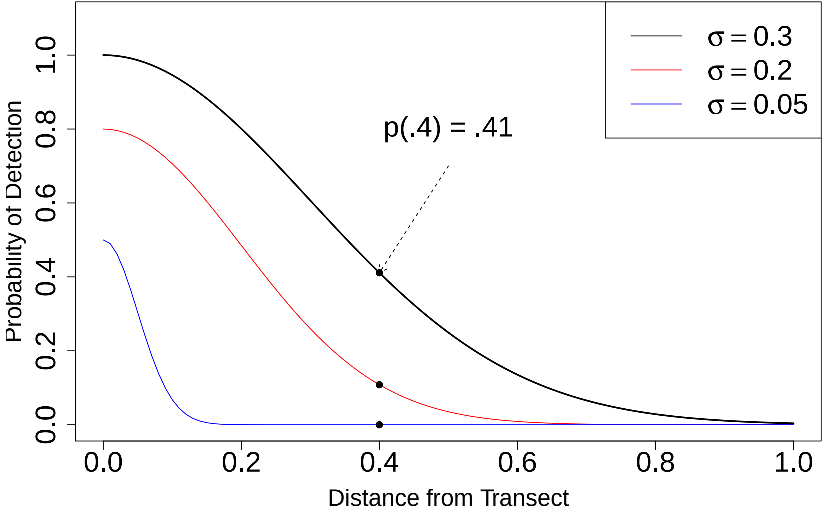
- Produce an accurate estimate of gopher tortoise abundance
- Allow for burrow size to influence detection probability
- Account for imperfect detection on the transect line
- Incorporate size-dependent burrow occupancy

Formulating the Model

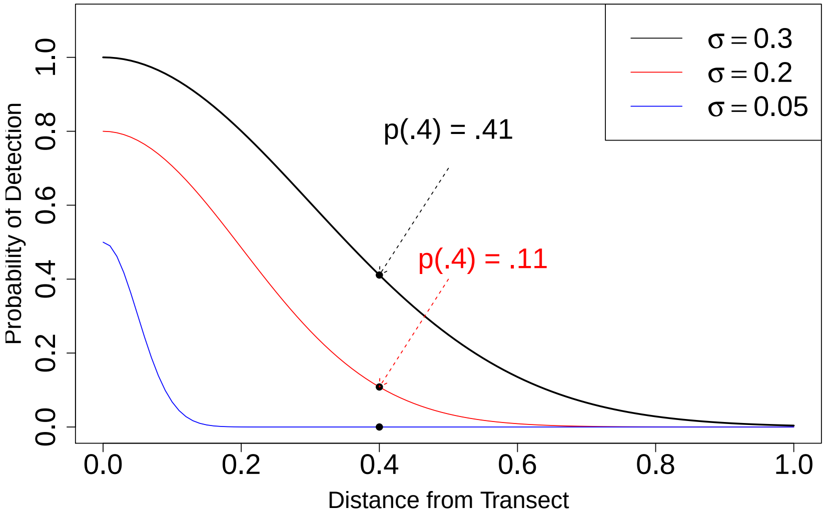
Assuming some M » N:

$$
\begin{aligned}
x_i &\sim Uniform(0, B) && \text{Distance to transect} \\
w_i &\sim Bern(\psi) && \text{Is this tortoise real?} \\
\psi &\sim Uniform(0, 1) && \text{Only } \frac{N}{M} \text{ are real} \\
N &= \sum_{i=1}^{M} w_i && \text{Count up how many burrows are real} \\
p_i &= \xi_i e^{\frac{-(x_i)^2}{2\sigma_i{}^2}} && \text{Detection probability} \\
ln(\sigma_i) &= \beta_0 + \beta_1 z_i && \text{Detection depends on size of burrow} \\
y_i &= Bern(p_i w_i) && \text{Was this burrow detected?} \\
o_i &\sim Bern(\theta_i) && \text{Was this burrow occupied?} \\
logit(\theta_i) &= \alpha_0 + \alpha_1 w_i && \text{smaller burrows more likely to be occupied} \\
N_{occ} &= \sum_{i=1}^{M} w_i o_i && \text{Count up tortoises}
\end{aligned}
$$

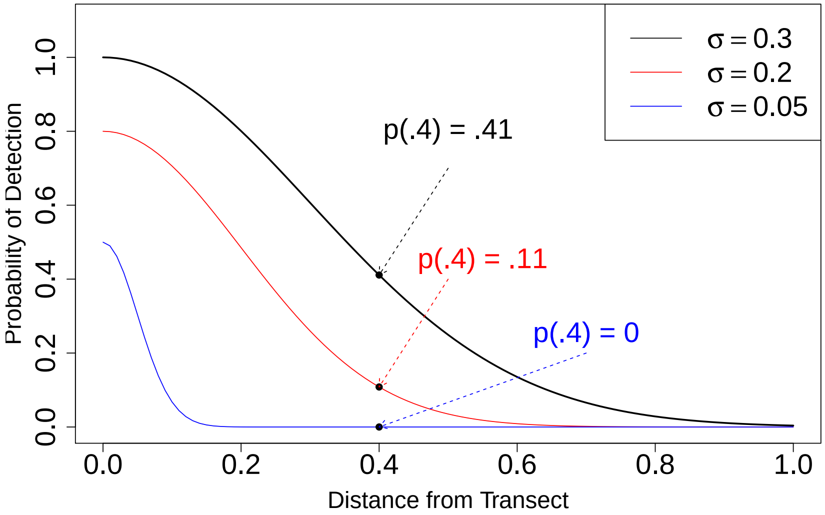
Variation in Detection



Variation in Detection



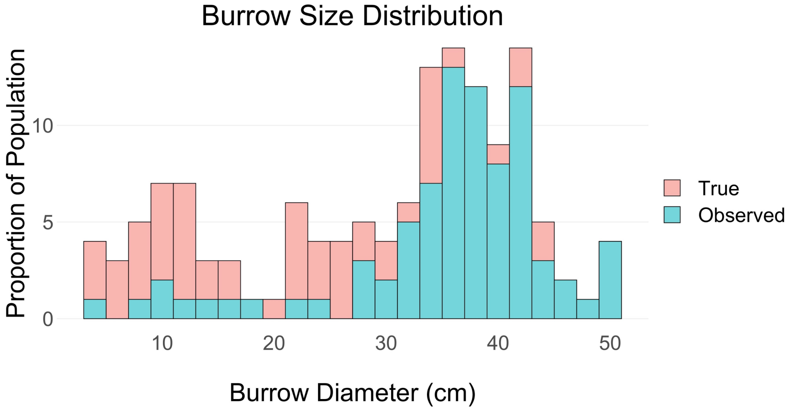
Variation in Detection



Formulating the Model

Two small problems:

- How do we calculate detection on the line (ξ_i)?
- What sizes (z_i) were the burrows we didn't see?



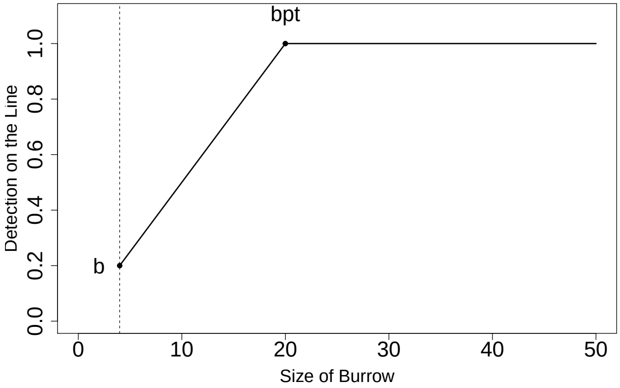
Imperfect Detection on the Line

Assuming a minimum burrow size of 4 cm wide and imperfect detection for all burrows $< b_{pt}$:

$$\xi_i = \begin{cases} \frac{1-b}{b_{pt}-4}(z_i - 4) + b & 4 \leq z_i < b_{pt} \\ 1 & z_i \geq b_{pt} \end{cases}$$

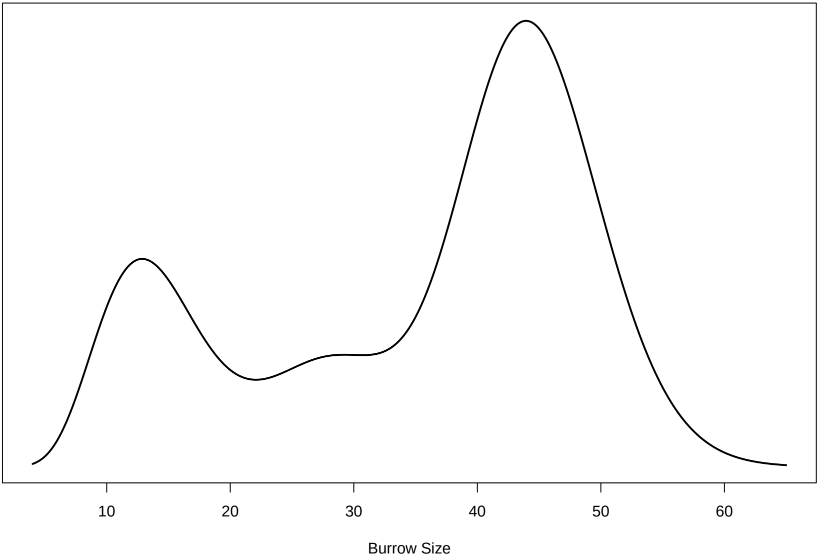
$$b \sim Uniform(0.2, 1)$$

$$b_{pt} \sim Uniform(15, 20)$$



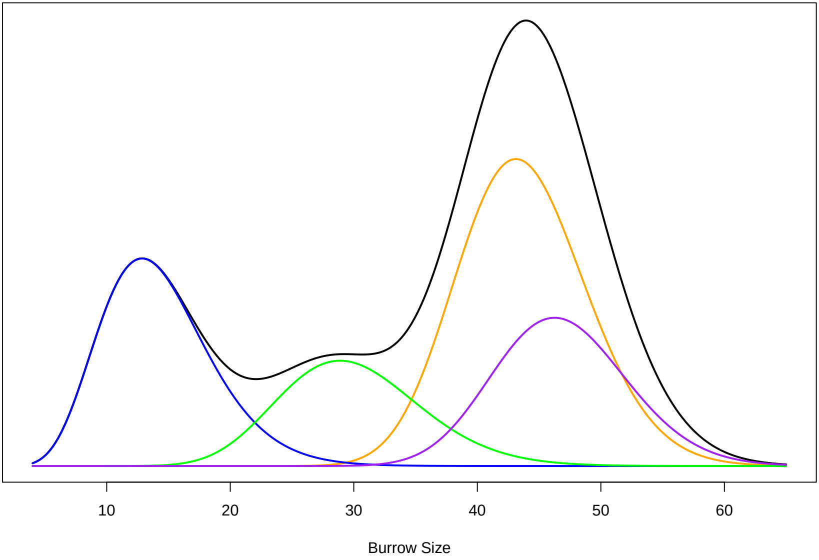
Dealing with Size

This part is extra tricky. Here's what a tortoise population size distribution might look like:



Dealing with Size

We chose to model this as a mixture of an unknown number of gamma distributions.

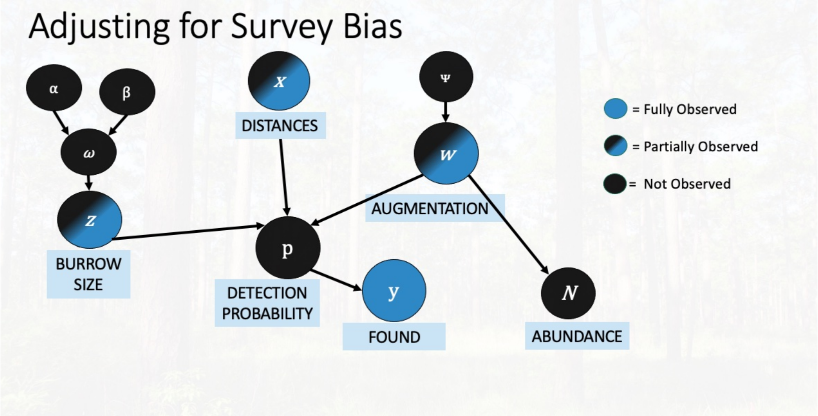


Dealing with Size

Each tortoise's size is drawn from a given gamma (k) with probability p_k.

$P(z_i \in k) \sim Dirichlet(\omega_1, \ldots, \omega_K)$	Choose a gamma
$\omega_k \sim Gamma(.001, .001)$	Uniformative prior for ω
$z_i \sim Gamma(\alpha_k, \beta_k)$	Exact size depends on which gamma

Model Summary



Let's Put it in JAGS!

*Note: full code w/ priors available on Github

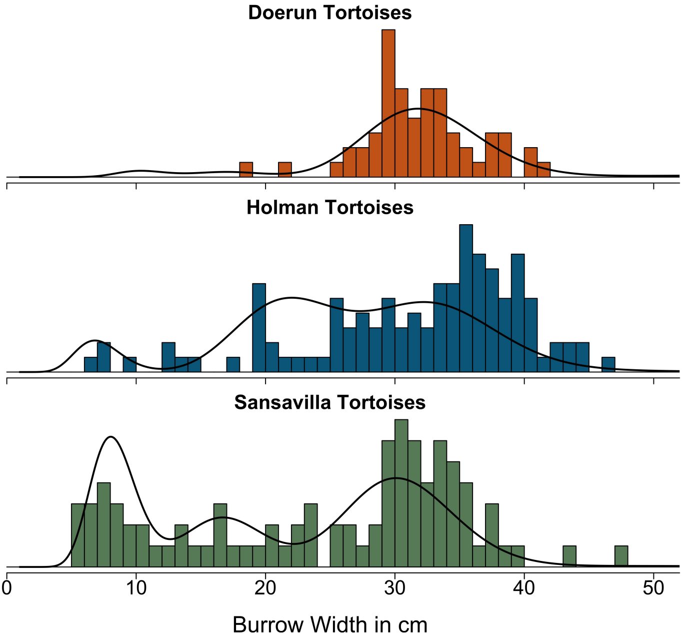
```
modelstring.Foo = "
model
{
    for (i in 1:(nind +nz)) {
        w[i] ~ dbern(psi)
        x[i] ~ dunif(0,Bx)
        z[i] ~ dgamma(a[i],c.beta[i])T(4,50)
        a[i] <- shape[clust[i]]
        c.beta[i] <- betaClust[clust[i]]
    clust[i] ~ dcat( pClust[1:Nclust] )
        sigma[i] <- exp(sigma.int+sigma.beta*z[i])
        logp[i] <- -((x[i]*x[i])/(2*sigma[i]*sigma[i]))
        p[i] <- exp(logp[i])*xi[i]
        xi[i] <- ifelse(z[i] < b.point, m*z[i]+intercept, 1)
        mu[i] <- w[i]*p[i]
        y[i] ~ dbern(mu[i])
    o[i]~ dbin(o2[i], 1)
    logit(o2[i]) <- o.int + z.beta*z[i]
    wo[i] <- o[i]*w[i]
    }
    intercept ~ dunif(.1,.8)
    b.point <- 20
    p.online <- m*4+intercept
    m <- (1-intercept)/b.point

    for (clustIdx in 1: Nclust) {
        shape[clustIdx] ~ dunif(1,100)
    betaClust[clustIdx] ~ dunif(.2,2)
        }
  pClust[1:Nclust] ~ ddirch(psizes)
    psi~ dunif(0,1)         #exists or not
    Nt <- sum(wo)
    N <- sum(w)
}
"
```

Side Note

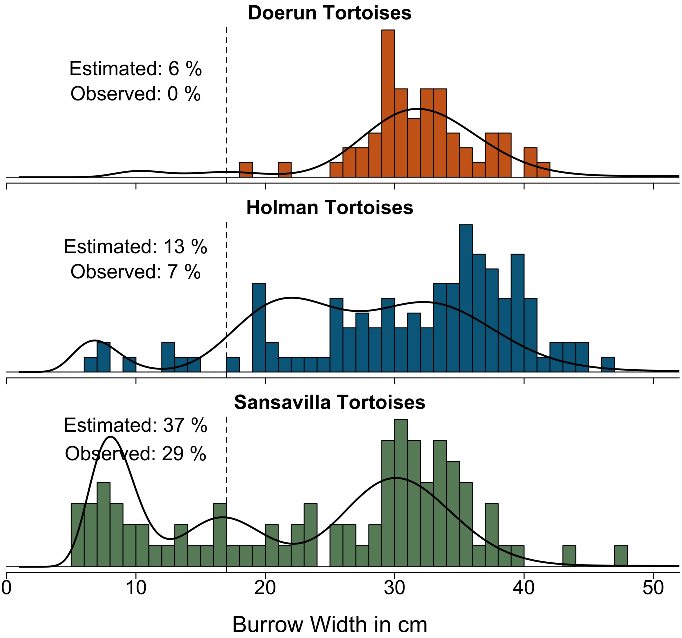
If we had ignored size variation and imperfect detection on the transect, our code would have been so much nicer. . .

```
modelstring.Foo = "
model
{
    for (i in 1:(nind +nz)) {
        w[i] ~ dbern(psi)                   # augmentation
        x[i] ~ dunif(0,Bx)              # distances
        logp[i] <- -((x[i]*x[i])/(2*sigma*sigma))
        p[i] <- exp(logp[i])
        mu[i] <- w[i]*p[i]
        y[i] ~ dbern(mu[i])
    }

    sigma ~ dunif(0, 20)
    psi~ dunif(0,1)         #exists or not

    N <- sum(w)
    D <- N/(2*L*Bx)         #burrow density
}
"
```

Results


Doerun Tortoises
Holman Tortoises
Sansavilla Tortoises
0
10
20
30
40
50
Burrow Width in cm

Results



Doerun Tortoises
Estimated: 6 %
Observed: 0 %
Holman Tortoises
Estimated: 13 %
Observed: 7 %
Sansavilla Tortoises
Estimated: 37 %
Observed: 29 %
0
10
20
30
40
50
Burrow Width in cm

Making It More Complicated

This model works great for including individual covariates like size. But what about environmental covariates that affect detection?

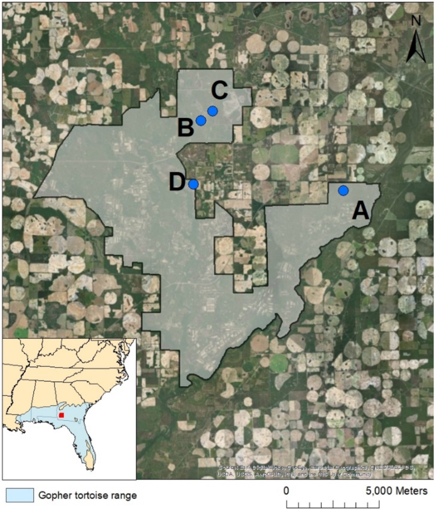
Making It More Complicated

Luckily, all we need to do is adjust our formula for σ and choose a prior for vegetation (v_i).

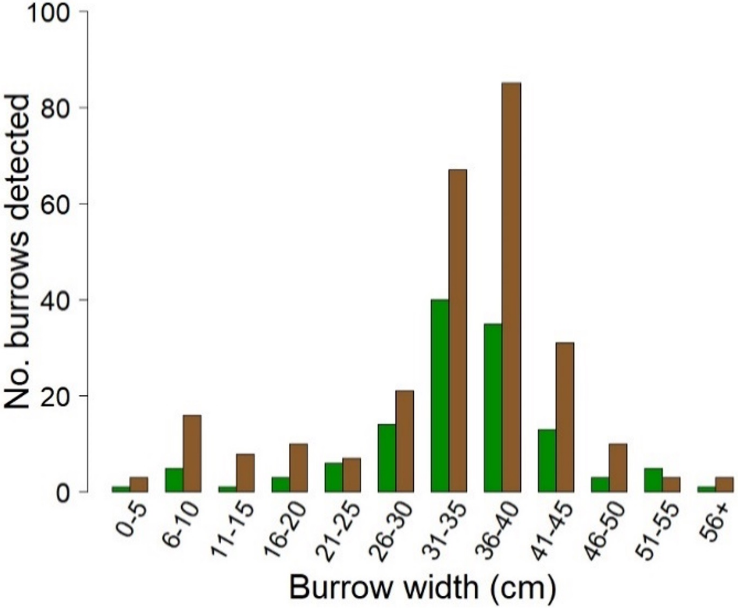
$p_i = \xi_i e^{\frac{-(x_i)^2}{2\sigma_i{}^2}}$	Detection probability
$ln(\sigma_i) = \beta_0 + \beta_1 z_i + \beta_2 v_i$	Detection depends on size + veg
$y_i = Bern(p_i w_i)$	Was this burrow detected?
$v_i \sim Beta(d, e)$	Prior for vegetation

Testing it Out

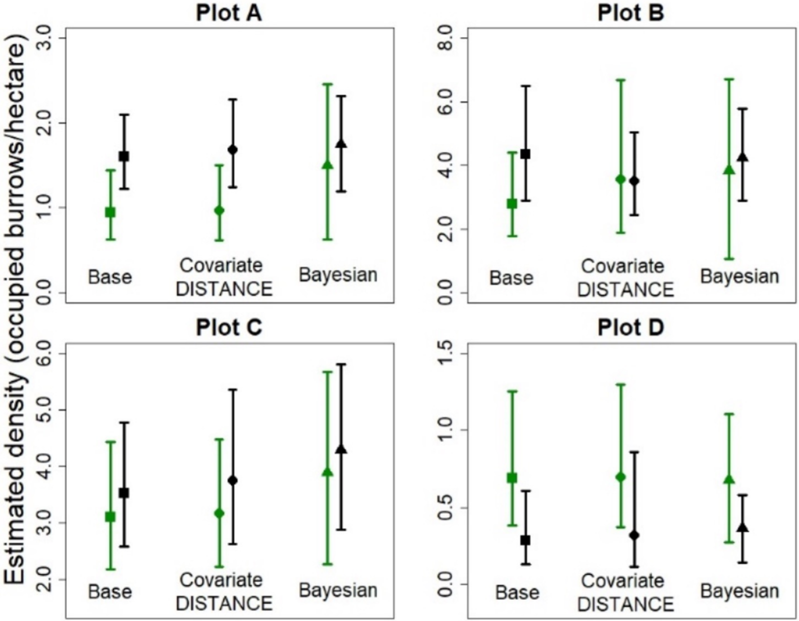
In addition to simulation, we tested our new model on field data collected before and after prescribed burns.



Testing it Out



Testing it Out



Did we meet our goals?

We were able to produce a model that accounted for all (5) objectives:

- Produce an accurate estimate of gopher tortoise abundance
- Allow for burrow size to influence detection probability
- Account for imperfect detection on the transect line
- Incorporate size-dependent burrow occupancy
- Account for spatial heterogeneity in a detection covariate

Pet Interlude

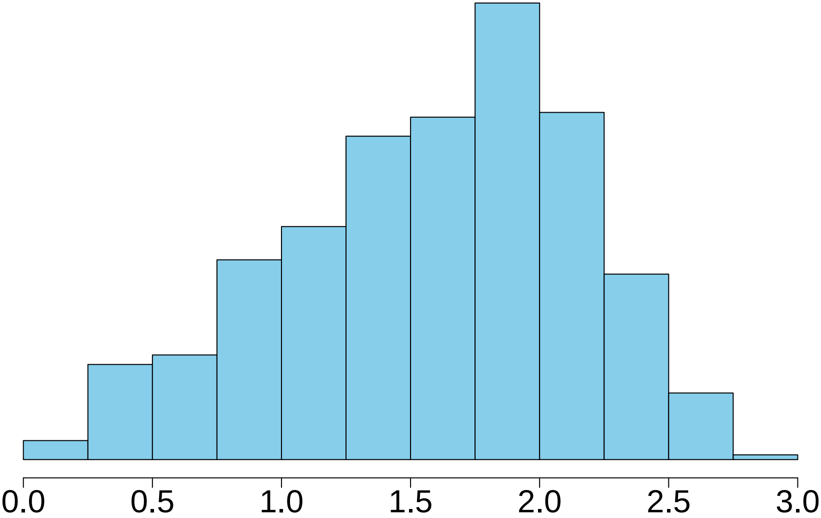



Current Bird Adventures

In my PhD, we also use distance sampling - but for birds! Instead of transects we use point counts.

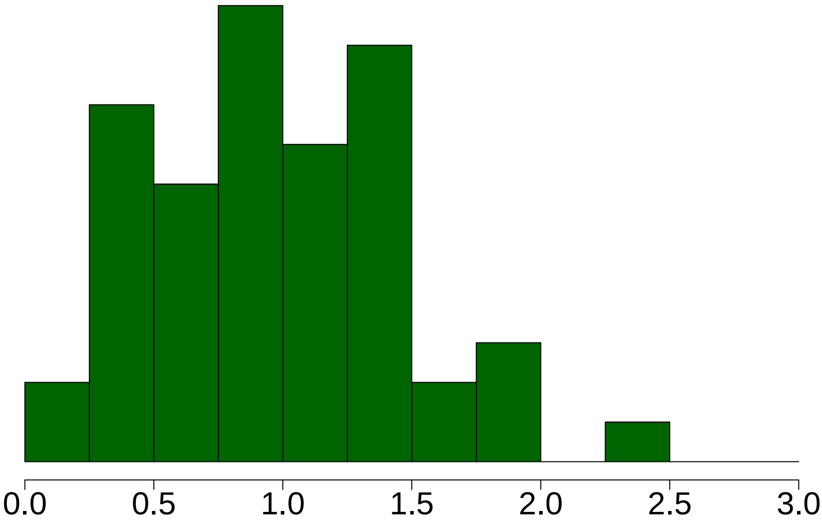
Point Counts

Assuming that birds are uniformly placed, the distance from the observer might look something like this:



Point Counts

But the observed distance might look like this:



Point Count Math

For point count data, we need to account for the increasing probability of being at a given distance as you move away from the observer. For a half-normal detection curve:

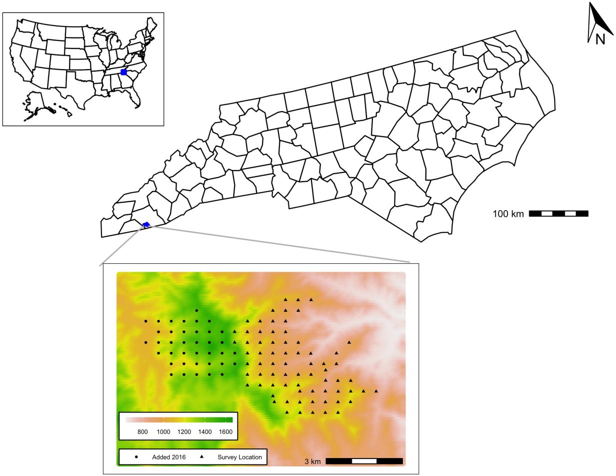
$$p = \int 2\pi e^{\frac{-x^2}{2\sigma^2}} dx$$

When you convert this into JAGS code, you split the integral into distance bins of equal width (10m, 20m, etc). Then you can use this fancy integration-avoidance trick:

```
pbar <- ((pnorm(b[j+1], 0, tau) - pnorm(b[j], 0, tau)) /
                 dnorm(0, 0, tau) / (b[j+1]-b[j]))*psi[j]
```

Setup

In the field, we perform point counts across our study site in North Carolina. The goal is to estimate the yearly abundance of all bird species present in the basin.



Protocol

Instead of returning to these point counts multiple times, we combine distance sampling with time-to-first-detection methods to help us estimate detection.

Observers listen/look for birds for 10 minutes in 4 2.5-minute increments and record the estimated distance of each bird in each time period.

This gives us 4 "visits" to work with - repeatability without all the effort!

During analysis, we run a dynamic N-mixture model (think Dail and Madsen 2011) to quantify the apparent survival, apparent emigration, and abundance of species at each site.

Fun Math

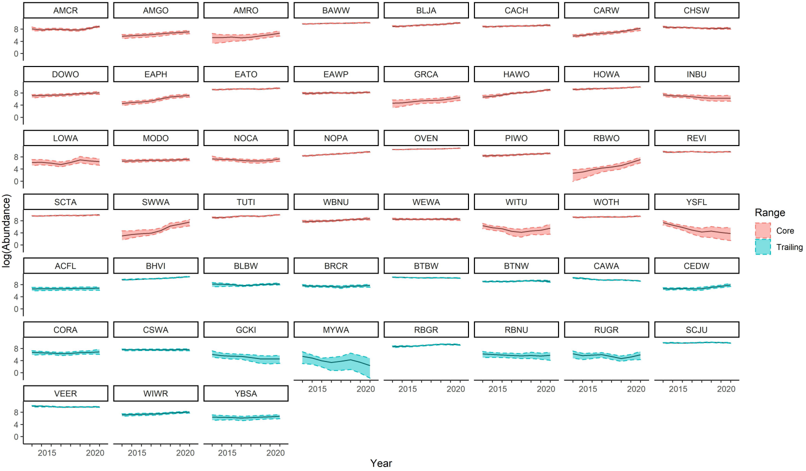
We group all detections into 10-m distance bins, b and calculate all bin midpoints, r_b.The probability of a detection in distance bin b with area A_b for species k at site i in year t is

$$p^d_{iktb} = \frac{2\pi}{A_b} \int_{r_b}^{r_{b+1}} exp(\frac{-x_b^2}{2\sigma_{it}^2})dx$$

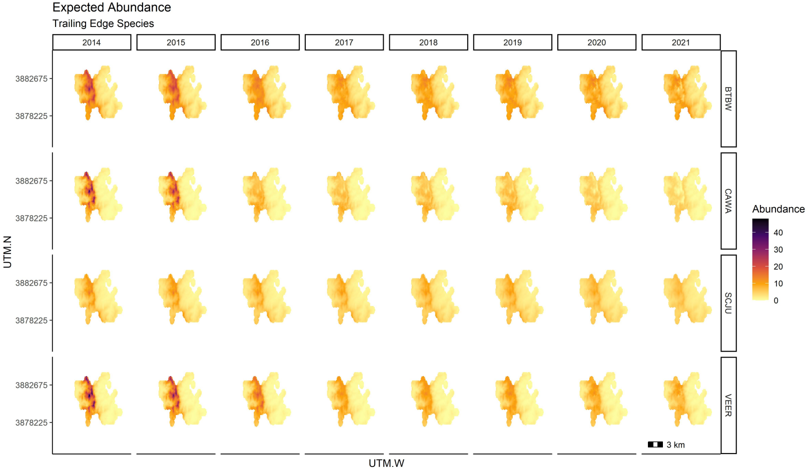
Yearly counts are then drawn from a binomial distribution given the latent site-abundance N_{ikt}, the availability probability (not described here), and the detection probability (distance + time-removal sampling).

$$Dist_{ikt} \sim Multi(\pi^d_{it1:10}, n_{ikt})$$
$$n_{ikt} \sim Binomial(N_{ikt}, \sum p^a_{it1:4} \sum p^d_{ikt1:10})$$

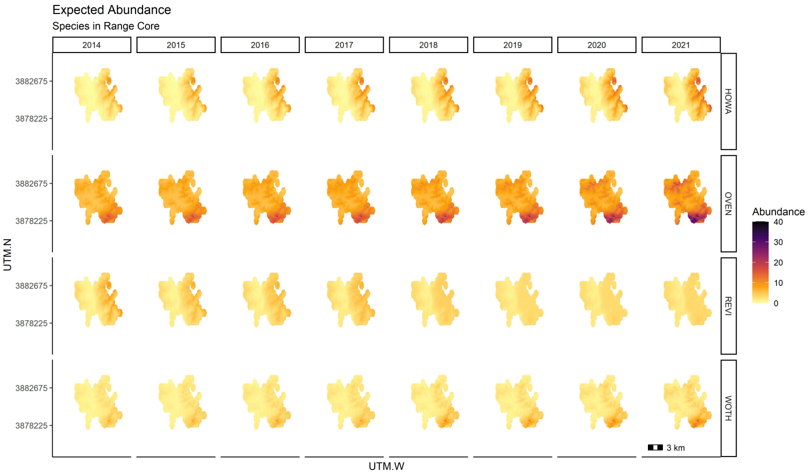
Results so far



Results so far



Results so far



Results so far

Questions?

I probably ran out of time, but if not... questions?

Feel free to email me at heather.gaya(@)uga.edu

Most of my code can be found at https://github.com/heathergaya or https://sites.google.com/view/heather-gaya/

Also, while on the topic of self promotion, check out my LLC: https://www.hhecoanalytics.com/