Simulation Example: Simulated Count Outcome

In this RMD file, we reproduce the results for analyzing a simulated count outcome over the Californian counties.

1 Packages and Data Setup

```
library(rstan)
library(parallel)
library(data.table)
library(sf)
library(spdep)
library(maps)
library(maptools)
library(magrittr)
library(stringr)
library(ggplot2)
library(fields)
rm(list = ls())
set.seed(113001)
```

Load in helper functions:

```
source(file.path(getwd(), "src", "R", "simulation", "simulation_helper.R"))
source(file.path(getwd(), "src", "R", "eps_loss_FDR.R"))
source(file.path(getwd(), "src", "R", "vij_computation.R"))
```

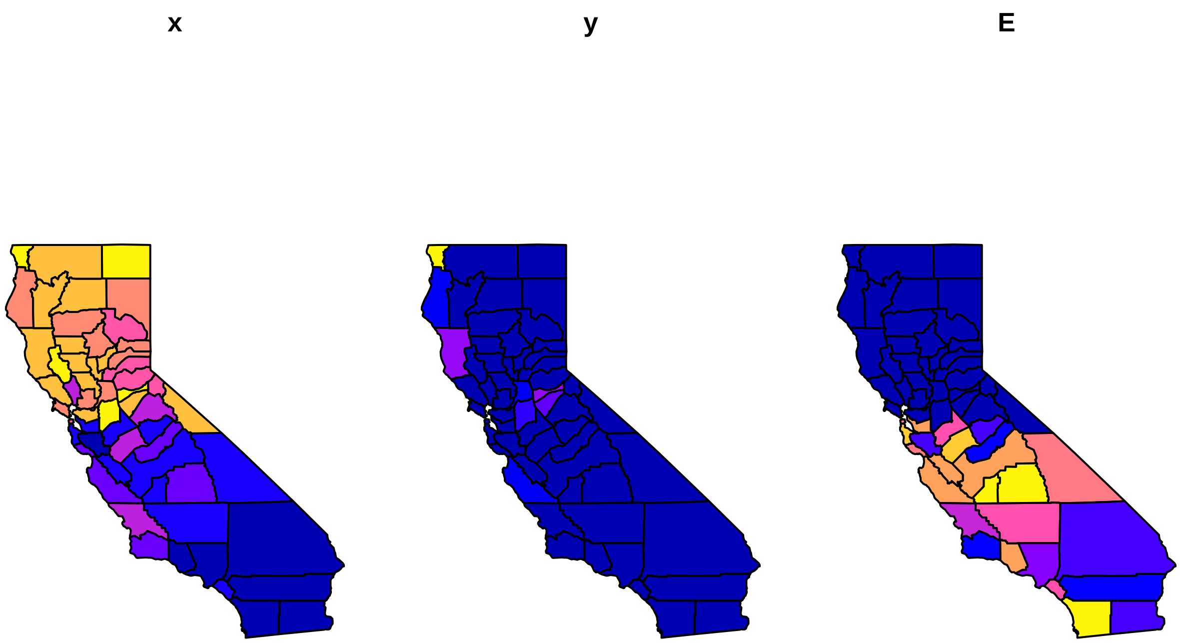
2 Data Generation

Data generation using Matern covariance kernel on county centroids:

```
# Import US counties
county_poly <- maps::map("county", "california", fill = TRUE, plot = FALSE)
county_state <- strsplit(county_poly$names, ",") %>%
  sapply(function(x) str_to_title(x[[1]]))
county_names <- strsplit(county_poly$names, ",") %>%
  sapply(function(x) str_to_title(x[[2]]))
sf_use_s2(TRUE)
county_sp <- maptools::map2SpatialPolygons(county_poly, IDs = county_poly$names)
county_nbs <- poly2nb(county_sp)
no_neighbors <- vapply(county_nbs, function(x) identical(x, 0L), logical(1))
# restrict to connected county map
county_sp <- county_sp[!no_neighbors,]
```

```r
county_state <- county_state[!no_neighbors]
county_names <- county_names[!no_neighbors]
county_nbs <- poly2nb(county_sp)
county_sf <- st_as_sf(county_sp)
rownames(county_sf) <- NULL
st_crs(county_sf) <- st_crs(st_as_sf(county_poly))

# data generation spatial variance: Matern covariance
county_cent <- st_centroid(st_as_sf(county_sp))
st_crs(county_cent) <- st_crs(county_sf)
dist_matrix <- matrix(st_distance(county_cent), nrow = nrow(county_sf),
                      ncol = nrow(county_sf)) / 1000
#dist_matrix <- st_distance(county_cent[1,], county_cent[2,])
Sigma <- Matern(dist_matrix, range = 0.5 * 100, phi = 1, smoothness = 0.5, nu = 0.5)
Sigma_chol <- chol(Sigma)
Q <- chol2inv(Sigma_chol)
N <- nrow(Q)

adj_df <- data.frame(
  i = rep(seq_len(N), times = vapply(county_nbs, length, numeric(1))),
  j = unlist(county_nbs)
)
adj_df <- adj_df[adj_df$i < adj_df$j, ]
rownames(adj_df) <- NULL

beta <- c(-5, 0.5)
cent_coords <- st_coordinates(county_cent)
mean_lat <- mean(cent_coords[,2])
x <- numeric(N)
high_risk <- cent_coords[,2] > mean_lat
x[high_risk] <- rnorm(sum(high_risk), mean = 2, sd = 1)
x[!high_risk] <- rnorm(sum(!high_risk), mean = -2, sd = 1)
county_sf$x <- x
X <- cbind(1, x)
E <- ceiling(runif(N, 10000, 5e5))
E[high_risk] <- ceiling(runif(sum(high_risk), 10000, 50000))
E[!high_risk] <- ceiling(runif(sum(!high_risk), 50000, 5e5))
sigma2 <- 2
rho <- 0.93
#phi <- solve(Q_scaled_cholR, rnorm(N))
phi <- t(Sigma_chol) %*% rnorm(N)
eps <- rnorm(N)
total_err <- sqrt(sigma2) * (sqrt(rho) * phi + sqrt(1 - rho) * eps)
Y <- rpois(N, exp(log(E) + X %*% beta + total_err))
county_sf$y <- Y
county_sf$E <- E
# analysis parameters
W <- nb2mat(county_nbs, style="B")
D <- diag(rowSums(W))
alpha <- 0.99
Q_analysis <- D - alpha * W
Sigma_analysis <- chol2inv(chol(Q_analysis))
scaling_factor <- exp(mean(log(diag(Sigma_analysis))))
```

```r
Sigma_analysis <- Sigma_analysis / scaling_factor
Sigma_analysis_chol <- chol(Sigma_analysis)

a0_sigma <- 0.1
b0_sigma <- 0.1
data <- list(
  N = N,
  Sigma_chol = t(Sigma_analysis_chol),
  mu_phi = rep(0, N),
  Y = Y,
  E = E,
  p = ncol(X),
  X = X,
  a0_sigma = a0_sigma,
  b0_sigma = b0_sigma
)
plot(county_sf)
```



3 Analysis

We fit the BYM2 model using the `rstan` package.

```r
fit1 <- stan(
  file = file.path(getwd(), "src", "stan", "bym2_poisson.stan"),
  pars = c("beta", "phi", "sigma2", "rho", "alpha"),
  data = data,
  chains = 4,
  warmup = 40000,
  iter = 60000,
  cores = 4
)

print(fit1)
```

```
## Inference for Stan model: anon_model.
## 4 chains, each with iter=60000; warmup=40000; thin=1;
## post-warmup draws per chain=20000, total post-warmup draws=80000.
##
##                mean se_mean   sd      2.5%       25%       50%       75%
## beta[1]       -5.08    0.00 0.69     -6.50     -5.47     -5.07     -4.69
## beta[2]        0.76    0.00 0.12      0.53      0.68      0.76      0.84
## phi[1]        -0.11    0.00 0.87     -1.83     -0.70     -0.12      0.47
## phi[2]         0.15    0.00 0.84     -1.51     -0.42      0.15      0.71
## phi[3]         0.32    0.00 0.86     -1.37     -0.26      0.32      0.90
## phi[4]        -0.49    0.01 0.85     -2.14     -1.07     -0.49      0.08
## phi[5]         0.24    0.00 0.85     -1.42     -0.34      0.25      0.81
## phi[6]        -0.54    0.01 0.87     -2.24     -1.13     -0.54      0.05
## phi[7]        -0.34    0.00 0.88     -2.05     -0.94     -0.34      0.24
## phi[8]         0.28    0.01 1.08     -1.91     -0.43      0.30      1.01
## phi[9]         0.28    0.01 0.88     -1.47     -0.31      0.28      0.87
## phi[10]        0.00    0.00 0.81     -1.59     -0.55      0.00      0.55
## phi[11]       -0.58    0.01 0.88     -2.29     -1.18     -0.59      0.01
## phi[12]       -0.08    0.01 0.94     -1.95     -0.71     -0.06      0.56
## phi[13]        1.44    0.01 1.16     -0.87      0.68      1.45      2.21
## phi[14]        0.44    0.01 0.87     -1.29     -0.15      0.44      1.03
## phi[15]        0.60    0.01 0.84     -1.07      0.03      0.61      1.17
## phi[16]        0.11    0.01 0.88     -1.61     -0.49      0.11      0.71
## phi[17]       -0.54    0.01 0.86     -2.20     -1.12     -0.54      0.04
## phi[18]       -0.31    0.01 0.91     -2.10     -0.93     -0.31      0.30
## phi[19]        1.12    0.01 0.97     -0.84      0.47      1.13      1.78
## phi[20]       -0.42    0.01 0.87     -2.11     -1.01     -0.42      0.17
## phi[21]        0.09    0.01 1.20     -2.31     -0.69      0.10      0.88
## phi[22]       -0.20    0.00 0.89     -1.94     -0.79     -0.20      0.40
## phi[23]       -0.19    0.00 0.86     -1.88     -0.76     -0.18      0.39
## phi[24]       -0.35    0.00 0.82     -1.95     -0.90     -0.36      0.21
## phi[25]       -0.47    0.01 0.97     -2.38     -1.13     -0.48      0.17
## phi[26]        0.18    0.00 0.84     -1.46     -0.38      0.18      0.75
## phi[27]        0.38    0.00 0.85     -1.29     -0.19      0.38      0.96
## phi[28]       -0.18    0.01 0.89     -1.94     -0.78     -0.18      0.42
## phi[29]        0.00    0.01 0.95     -1.87     -0.63      0.01      0.64
## phi[30]        1.11    0.01 1.00     -0.86      0.45      1.11      1.78
## phi[31]       -0.09    0.00 0.85     -1.77     -0.66     -0.09      0.48
## phi[32]       -0.34    0.00 0.85     -2.00     -0.90     -0.34      0.23
## phi[33]        1.33    0.01 1.03     -0.72      0.65      1.33      2.02
## phi[34]        0.09    0.00 0.80     -1.48     -0.45      0.09      0.63
```

```
## phi[35]         0.21     0.00 0.86     -1.49     -0.38      0.21       0.79
## phi[36]         0.80     0.01 0.93     -1.05      0.17      0.80       1.42
## phi[37]         1.14     0.01 1.09     -0.99      0.41      1.14       1.87
## phi[38]        -0.03     0.01 1.20     -2.43     -0.81     -0.03       0.75
## phi[39]        -0.05     0.00 0.80     -1.61     -0.59     -0.05       0.49
## phi[40]         0.43     0.01 0.91     -1.36     -0.18      0.43       1.05
## phi[41]        -0.29     0.01 0.98     -2.20     -0.95     -0.30       0.35
## phi[42]         0.67     0.01 0.97     -1.26      0.02      0.68       1.33
## phi[43]        -0.25     0.00 0.81     -1.83     -0.80     -0.25       0.30
## phi[44]         0.08     0.01 0.90     -1.67     -0.52      0.08       0.69
## phi[45]        -0.48     0.01 0.88     -2.21     -1.07     -0.48       0.10
## phi[46]        -0.11     0.01 0.91     -1.92     -0.73     -0.11       0.50
## phi[47]        -0.37     0.01 0.91     -2.17     -0.99     -0.37       0.24
## phi[48]        -0.32     0.01 0.88     -2.04     -0.91     -0.32       0.28
## phi[49]        -0.31     0.01 0.93     -2.12     -0.94     -0.31       0.31
## phi[50]        -0.26     0.00 0.81     -1.84     -0.81     -0.26       0.29
## phi[51]        -0.42     0.00 0.84     -2.06     -0.99     -0.43       0.14
## phi[52]        -0.53     0.01 0.86     -2.20     -1.12     -0.54       0.04
## phi[53]        -0.51     0.01 0.90     -2.26     -1.11     -0.51       0.09
## phi[54]         0.29     0.01 0.90     -1.48     -0.32      0.29       0.90
## phi[55]         0.04     0.00 0.81     -1.55     -0.51      0.04       0.58
## phi[56]         0.61     0.01 1.00     -1.34     -0.06      0.61       1.28
## phi[57]        -0.27     0.00 0.84     -1.91     -0.83     -0.27       0.30
## phi[58]        -0.25     0.00 0.85     -1.91     -0.82     -0.25       0.32
## sigma2          2.70     0.01 0.77      1.65      2.17      2.55       3.05
## rho             0.34     0.00 0.19      0.03      0.18      0.32       0.47
## alpha[1]       13.33     0.01 0.77     11.87     12.85     13.28      13.78
## alpha[2]        9.55     0.00 0.70      8.10      9.15      9.55       9.95
## alpha[3]       11.67     0.01 0.79      9.99     11.20     11.71      12.16
## alpha[4]        9.80     0.00 0.71      8.32      9.39      9.81      10.21
## alpha[5]       12.26     0.01 0.73     10.73     11.84     12.29      12.69
## alpha[6]        8.27     0.01 0.77      6.64      7.82      8.30       8.74
## alpha[7]        9.04     0.01 0.74      7.48      8.61      9.07       9.49
## alpha[8]       12.48     0.01 0.80     10.78     12.00     12.52      12.98
## alpha[9]       12.03     0.00 0.69     10.61     11.64     12.03      12.42
## alpha[10]      11.56     0.01 0.79     10.07     11.07     11.52      12.02
## alpha[11]       7.89     0.01 0.76      6.29      7.44      7.92       8.35
## alpha[12]      11.41     0.01 0.72      9.91     11.00     11.43      11.83
## alpha[13]      13.99     0.01 0.79     12.50     13.49     13.94      14.45
## alpha[14]      13.60     0.01 0.76     12.16     13.13     13.56      14.03
## alpha[15]      13.93     0.01 0.74     12.51     13.48     13.89      14.35
## alpha[16]      11.51     0.01 0.76     10.06     11.04     11.47      11.95
## alpha[17]       7.98     0.01 0.80      6.30      7.51      8.02       8.48
## alpha[18]      10.34     0.00 0.70      8.89      9.95     10.35      10.74
## alpha[19]      15.10     0.01 0.82     13.57     14.58     15.05      15.58
## alpha[20]       8.33     0.01 0.76      6.84      7.87      8.32       8.78
## alpha[21]      11.02     0.01 0.72      9.51     10.61     11.04      11.44
## alpha[22]      11.48     0.01 0.78     10.01     11.00     11.45      11.94
## alpha[23]      12.19     0.01 0.74     10.63     11.77     12.22      12.64
## alpha[24]      10.36     0.00 0.70      8.97      9.95     10.34      10.75
## alpha[25]       9.79     0.01 0.77      8.15      9.33      9.82      10.26
## alpha[26]      10.71     0.01 0.74      9.17     10.29     10.74      11.15
## alpha[27]      14.56     0.01 0.72     13.16     14.13     14.53      14.97
## alpha[28]      11.27     0.00 0.69      9.88     10.87     11.25      11.66
```

```
## alpha[29]     10.35    0.00 0.71      8.88      9.95     10.37     10.76
## alpha[30]     13.74    0.01 0.76     12.30     13.28     13.70     14.17
## alpha[31]     10.45    0.00 0.69      9.03     10.06     10.45     10.84
## alpha[32]     10.16    0.00 0.69      8.75      9.77     10.16     10.55
## alpha[33]     13.52    0.01 0.81     12.00     13.00     13.47     14.00
## alpha[34]     11.98    0.00 0.70     10.53     11.58     11.99     12.38
## alpha[35]     14.61    0.01 0.78     13.14     14.12     14.57     15.06
## alpha[36]     11.61    0.01 0.87     10.00     11.05     11.56     12.13
## alpha[37]     13.59    0.01 0.80     12.09     13.08     13.54     14.05
## alpha[38]     13.01    0.00 0.69     11.61     12.62     13.01     13.40
## alpha[39]     10.64    0.01 0.81      8.93     10.16     10.68     11.14
## alpha[40]     12.77    0.00 0.70     11.39     12.37     12.75     13.16
## alpha[41]     11.60    0.01 0.78     10.13     11.11     11.56     12.06
## alpha[42]     12.40    0.01 0.72     11.00     11.97     12.37     12.81
## alpha[43]      9.76    0.01 0.84      8.17      9.22      9.73     10.26
## alpha[44]     13.23    0.01 0.74     11.82     12.79     13.19     13.65
## alpha[45]      8.90    0.01 0.76      7.31      8.46      8.93      9.36
## alpha[46]     10.75    0.00 0.70      9.31     10.36     10.76     11.15
## alpha[47]      9.36    0.01 0.73      7.82      8.94      9.38      9.79
## alpha[48]      8.79    0.00 0.71      7.33      8.39      8.80      9.20
## alpha[49]      8.93    0.01 0.76      7.34      8.49      8.96      9.39
## alpha[50]     10.45    0.01 0.79      8.94      9.95     10.41     10.91
## alpha[51]      8.94    0.01 0.77      7.32      8.49      8.98      9.41
## alpha[52]      9.02    0.01 0.72      7.51      8.61      9.05      9.45
## alpha[53]      8.67    0.01 0.73      7.13      8.25      8.69      9.11
## alpha[54]     13.40    0.01 0.73     11.99     12.96     13.37     13.81
## alpha[55]     11.85    0.00 0.69     10.45     11.45     11.84     12.23
## alpha[56]     12.69    0.01 0.85     11.11     12.15     12.65     13.20
## alpha[57]     10.46    0.01 0.74      8.92     10.04     10.49     10.90
## alpha[58]     10.51    0.01 0.73      8.97     10.09     10.54     10.95
## lp__      677222.58    0.07 9.36 677203.42 677216.46 677222.87 677229.07
##               97.5% n_eff Rhat
## beta[1]       -3.69 20467    1
## beta[2]        1.01 10476    1
## phi[1]         1.59 33881    1
## phi[2]         1.82 31947    1
## phi[3]         1.98 31409    1
## phi[4]         1.18 28048    1
## phi[5]         1.90 31024    1
## phi[6]         1.19 29008    1
## phi[7]         1.38 30989    1
## phi[8]         2.35 25298    1
## phi[9]         1.99 29988    1
## phi[10]        1.59 29734    1
## phi[11]        1.17 28547    1
## phi[12]        1.74 29531    1
## phi[13]        3.71 23424    1
## phi[14]        2.13 29643    1
## phi[15]        2.23 26099    1
## phi[16]        1.85 28226    1
## phi[17]        1.17 28504    1
## phi[18]        1.47 30182    1
## phi[19]        3.00 23278    1
## phi[20]        1.29 26322    1
```

```
## phi[21]        2.43 32022    1
## phi[22]        1.53 33245    1
## phi[23]        1.49 30318    1
## phi[24]        1.27 27579    1
## phi[25]        1.47 29786    1
## phi[26]        1.82 32920    1
## phi[27]        2.04 29079    1
## phi[28]        1.56 31004    1
## phi[29]        1.84 29141    1
## phi[30]        3.06 25834    1
## phi[31]        1.58 31711    1
## phi[32]        1.32 29805    1
## phi[33]        3.34 22905    1
## phi[34]        1.64 29741    1
## phi[35]        1.89 31553    1
## phi[36]        2.61 26706    1
## phi[37]        3.30 24295    1
## phi[38]        2.31 31963    1
## phi[39]        1.51 33850    1
## phi[40]        2.22 30164    1
## phi[41]        1.65 32797    1
## phi[42]        2.56 28298    1
## phi[43]        1.35 32281    1
## phi[44]        1.84 32061    1
## phi[45]        1.24 28771    1
## phi[46]        1.67 29713    1
## phi[47]        1.42 29089    1
## phi[48]        1.42 30877    1
## phi[49]        1.52 31314    1
## phi[50]        1.31 32002    1
## phi[51]        1.23 30969    1
## phi[52]        1.18 28125    1
## phi[53]        1.26 28394    1
## phi[54]        2.05 30409    1
## phi[55]        1.62 31180    1
## phi[56]        2.56 30797    1
## phi[57]        1.37 31187    1
## phi[58]        1.42 30059    1
## sigma2         4.63  4924    1
## rho            0.75  4227    1
## alpha[1]      14.98 16435    1
## alpha[2]      10.96 20580    1
## alpha[3]      13.18 17751    1
## alpha[4]      11.20 20160    1
## alpha[5]      13.68 19435    1
## alpha[6]       9.75 18457    1
## alpha[7]      10.48 19124    1
## alpha[8]      14.00 17570    1
## alpha[9]      13.44 20512    1
## alpha[10]     13.25 16463    1
## alpha[11]      9.35 18759    1
## alpha[12]     12.82 19821    1
## alpha[13]     15.69 15907    1
## alpha[14]     15.22 17058    1
```

```
## alpha[15]     15.51 17652    1
## alpha[16]     13.13 17164    1
## alpha[17]      9.50 17716    1
## alpha[18]     11.74 20421    1
## alpha[19]     16.86 15277    1
## alpha[20]      9.90 22425    1
## alpha[21]     12.44 19610    1
## alpha[22]     13.16 16731    1
## alpha[23]     13.63 18977    1
## alpha[24]     11.83 20324    1
## alpha[25]     11.27 18214    1
## alpha[26]     12.14 19232    1
## alpha[27]     16.10 18797    1
## alpha[28]     12.71 20370    1
## alpha[29]     11.76 20081    1
## alpha[30]     15.36 17035    1
## alpha[31]     11.86 20604    1
## alpha[32]     11.58 20640    1
## alpha[33]     15.26 15498    1
## alpha[34]     13.38 20301    1
## alpha[35]     16.28 16171    1
## alpha[36]     13.47 15109    1
## alpha[37]     15.30 15825    1
## alpha[38]     14.42 20530    1
## alpha[39]     12.17 17447    1
## alpha[40]     14.25 19878    1
## alpha[41]     13.27 16722    1
## alpha[42]     13.93 19201    1
## alpha[43]     11.52 17574    1
## alpha[44]     14.81 17863    1
## alpha[45]     10.36 18734    1
## alpha[46]     12.15 20392    1
## alpha[47]     10.79 19380    1
## alpha[48]     10.21 20612    1
## alpha[49]     10.39 18736    1
## alpha[50]     12.13 17422    1
## alpha[51]     10.42 18349    1
## alpha[52]     10.45 19761    1
## alpha[53]     10.10 19467    1
## alpha[54]     14.96 18237    1
## alpha[55]     13.28 20453    1
## alpha[56]     14.51 14985    1
## alpha[57]     11.89 19202    1
## alpha[58]     11.94 19313    1
## lp__      677239.97 17927    1
##
## Samples were drawn using NUTS(diag_e) at Thu Mar  6 12:16:45 2025.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```r
samps <- as.matrix(fit1)
#HDInterval::hdi(samps[,"rho"])
phi_samps <- samps[, paste0("phi[", seq_len(N), "]")]
```

```r
sigma2_samps <- samps[, "sigma2"]
summary(sigma2_samps)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.075   2.173   2.546   2.697   3.053   9.325
```

```r
rho_samps <- samps[, "rho"]
summary(rho_samps)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000186 0.1833442 0.3204583 0.3374784 0.4733237 0.9912097
```

We use the collected samples to compute difference probabilities of the form $\tau_k(\epsilon) = \Pr\left(\frac{|c_k^T\phi|}{\sqrt{\mathrm{Var}(c_k^T\phi \mid y)}} > \epsilon \;\middle|\; y\right)$.

Rejection path graph:

```r
V_est <- cov(phi_samps)
n_s <- nrow(phi_samps)
k <- nrow(adj_df)
phi_diffs <- vapply(seq_len(k), function(pair_indx) {
  i <- adj_df[pair_indx,]$i
  j <- adj_df[pair_indx,]$j
  var <- V_est[i, i] + V_est[j, j] - 2 * V_est[i, j]
  (phi_samps[,i] - phi_samps[,j]) / sqrt(var)
}, numeric(n_s))

phi_truediff <- vapply(seq_len(k), function(pair_indx) {
  i <- adj_df[pair_indx,]$i
  j <- adj_df[pair_indx,]$j
  #var <- V_est[i, i] + V_est[j, j] - 2 * V_est[i, j]
  var <- Sigma[i, i] + Sigma[j, j] - 2 * Sigma[i, j]
  (phi[i] - phi[j]) / sqrt(var)
}, numeric(1))



loss_function <- function(v, epsilon) -ConditionalEntropy(v)
system.time({
  eps_optim <- optim(1, function(e) {
    e_vij <- ComputeSimVij(phi_diffs, epsilon = e)
    loss_function(e_vij, epsilon = e)
  }, method = "Brent", lower = 0.0001, upper = 2.0)
})
```

```
##    user  system elapsed
##   1.488   0.686   2.329
```

```r
optim_e <- eps_optim$par
optim_e_vij <- ComputeSimVij(phi_diffs, epsilon = optim_e)
optim_e_vij_order <- order(optim_e_vij, decreasing = F)
```

```r
true_diff <- abs(phi_truediff) > optim_e
#true_diff <- (abs(true_phi_diffs) > optim_e)
mean(true_diff)
```

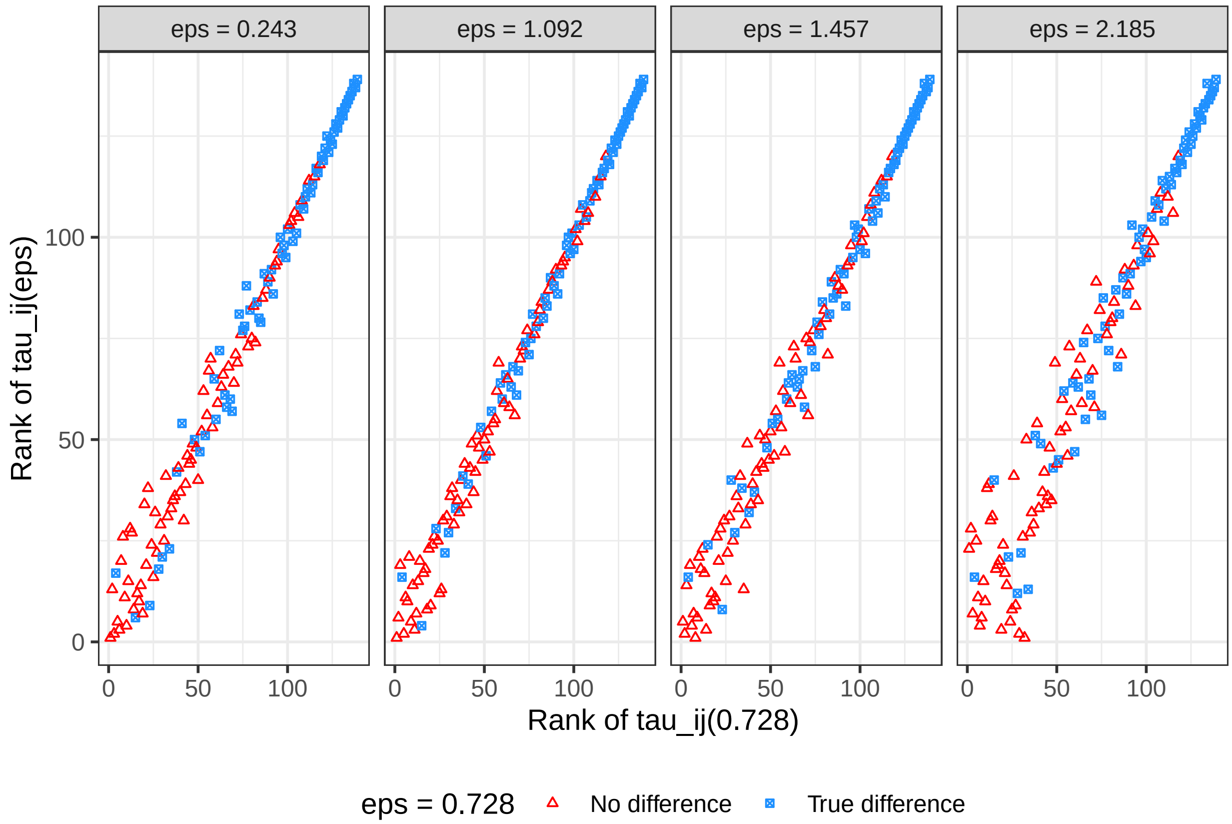
```
## [1] 0.4748201
```

```r
optim_e_vij <- ComputeSimVij(phi_diffs, epsilon = optim_e)
optim_e_vij_order <- order(optim_e_vij, decreasing = F)

# indx <- abs(phi_truediff)  > median(abs(phi_truediff))
indx <- optim_e_vij >= sort(optim_e_vij, decreasing = TRUE)[40]
detected_borders <- adj_df[indx,]
county_sf2 <- county_sf
county_sf2$x <- NULL
node1_all <- county_sf2[detected_borders$i,]
node2_all <- county_sf2[detected_borders$j,]
sf_use_s2(FALSE)
intersections <- lapply(seq_len(sum(indx)), function(i) {
  #print(i)
  node1 <- node1_all[i,]
  node2 <- node2_all[i,]
  suppressMessages(st_intersection(st_buffer(node1, 0.001),
                                   st_buffer(node2, 0.001)))

}) %>%
  do.call(rbind, .)
rates <- Y / E
rates_boundaries_df <- data.frame(node1_rate = rates[adj_df[indx,]$i],
                                  node2_rate = rates[adj_df[indx,]$j])
mean(apply(rates_boundaries_df, 1, function(x) all(x < 0.05)))
```

```
## [1] 0.625
```

```r
rate_map <- ggplot() +
  geom_sf(data = county_sf, aes(fill = y / E), color = "black") +
  geom_sf(data = intersections, color = "red", linewidth = 1) +
  scale_fill_viridis_c(name = "Simulated Rate") +
  coord_sf(crs = st_crs(5070)) +
  theme_bw() +
  theme(legend.position = "bottom", legend.title=element_text(size=10))
lograte_map <- ggplot() +
  geom_sf(data = county_sf, aes(fill = log(y / E)), color = "black") +
  geom_sf(data = intersections, color = "red", linewidth = 1) +
  scale_fill_viridis_c(name = "Simulated Log(Rate)") +
  coord_sf(crs = st_crs(5070)) +
  theme_bw() +
  theme(legend.position = "bottom", legend.title=element_text(size=10))


## rejection order graph
e2 <- round(optim_e / 3, digits = 3)
e3 <- round(optim_e * 1.5, digits = 3)
```

```r
e4 <- round(optim_e * 2, digits = 3)
e5 <- round(optim_e * 3, digits = 3)

e2_vij <- ComputeSimVij(phi_diffs, epsilon = e2)
e3_vij <- ComputeSimVij(phi_diffs, epsilon = e3)
e4_vij <- ComputeSimVij(phi_diffs, epsilon = e4)
e5_vij <- ComputeSimVij(phi_diffs, epsilon = e5)

optim_e_vij_order <- order(optim_e_vij, decreasing = F)
e2_vij_order <- order(e2_vij[optim_e_vij_order], decreasing = F)
e3_vij_order <- order(e3_vij[optim_e_vij_order], decreasing = F)
e4_vij_order <- order(e4_vij[optim_e_vij_order], decreasing = F)
e5_vij_order <- order(e5_vij[optim_e_vij_order], decreasing = F)
rejection_path <- data.table(
  optim_e_vij = seq_along(optim_e_vij),
  e2_vij_order = e2_vij_order,
  e3_vij_order = e3_vij_order,
  e4_vij_order = e4_vij_order,
  e5_vij_order = e5_vij_order,
  true_diff = true_diff[optim_e_vij_order]
)

rejection_path <- melt(rejection_path,
                       id.vars = c("optim_e_vij", "true_diff"),
                       variable.name = "order_type",
                       value.name = "order")
rejection_path[, order_type := fcase(
  order_type == "e2_vij_order", paste0("eps = ", e2),
  order_type == "e3_vij_order", paste0("eps = ", e3),
  order_type == "e4_vij_order", paste0("eps = ", e4),
  order_type == "e5_vij_order", paste0("eps = ", e5)
)]
sim_vij_order_graph <- ggplot() +
  geom_point(data = rejection_path,
             aes(x = optim_e_vij, y = order, color = true_diff,
                 shape = true_diff),
             alpha = 1, size = 1) +
  #geom_vline(xintercept = nrow(ij_list) - sum(optim_e_vij == 1)) +
  facet_grid(~order_type) +
  labs(x = paste0("Rank of tau_ij(", round(optim_e, digits = 3), ")"),
       y = "Rank of tau_ij(eps)") +
  theme_bw() +
  scale_color_manual(name = paste0("eps = ", round(optim_e, digits = 3)),
                     labels = c("No difference", "True difference"),
                     values = c("FALSE" = "red", "TRUE" = "dodgerblue")) +
  scale_shape_manual(name = paste0("eps = ", round(optim_e, digits = 3)),
                     labels = c("No difference", "True difference"),
                     values = c("FALSE" = 2, "TRUE" = 7)) +
  theme(legend.position = "bottom")
sim_vij_order_graph
```

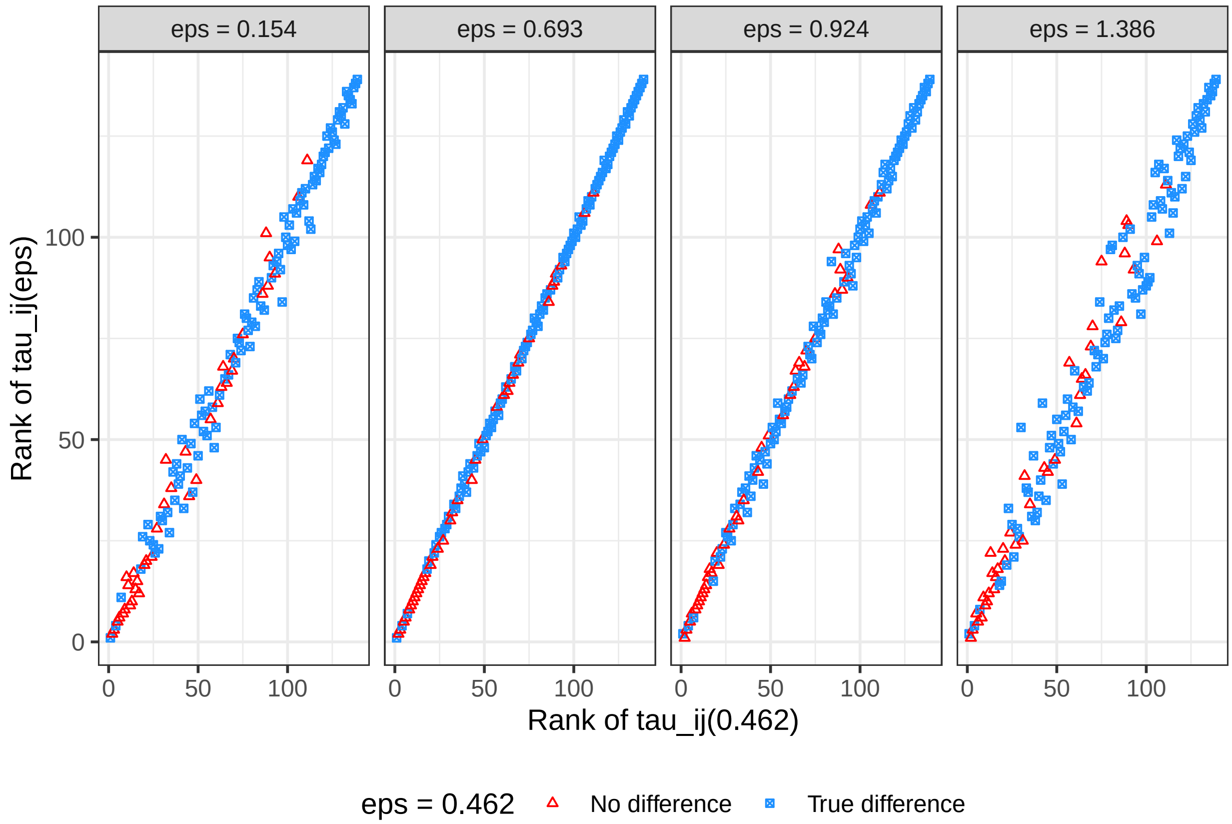


We also compute unstandardized difference probabilities of the form $\tau_{ij} = \mathbb{P}(|\phi_i - \phi_j| > \epsilon \,|\, y)$ to compare the classification performance:

```r
# compute unstandardized difference probabilities
phi_diffs2 <- vapply(seq_len(k), function(pair_indx) {
  i <- adj_df[pair_indx,]$i
  j <- adj_df[pair_indx,]$j
  (phi_samps[,i] - phi_samps[,j])
}, numeric(n_s))

phi_truediff2 <- vapply(seq_len(k), function(pair_indx) {
  i <- adj_df[pair_indx,]$i
  j <- adj_df[pair_indx,]$j
  (phi[i] - phi[j])
}, numeric(1))
system.time({
  eps_optim <- optim(1, function(e) {
    e_vij <- ComputeSimVij(phi_diffs2, epsilon = e)
    loss_function(e_vij, epsilon = e)
  }, method = "Brent", lower = 0.0001, upper = 4.0)
})
```

```
##    user  system elapsed
##   1.330   0.975   2.806
```

```r
e <- eps_optim$par
optim_e_vij2 <- ComputeSimVij(phi_diffs2, epsilon = optim_e)
e2 <- round(e / 3, digits = 3)
e3 <- round(e * 1.5, digits = 3)
e4 <- round(e * 2, digits = 3)
e5 <- round(e * 3, digits = 3)
true_diff2 <- abs(phi_truediff2) > e

e2_vij2 <- ComputeSimVij(phi_diffs2, epsilon = e2)
e3_vij2 <- ComputeSimVij(phi_diffs2, epsilon = e3)
e4_vij2 <- ComputeSimVij(phi_diffs2, epsilon = e4)
e5_vij2 <- ComputeSimVij(phi_diffs2, epsilon = e5)
optim_e_vij_order <- order(optim_e_vij2, decreasing = F)
e2_vij2_order <- order(e2_vij2[optim_e_vij_order], decreasing = F)
e3_vij2_order <- order(e3_vij2[optim_e_vij_order], decreasing = F)
e4_vij2_order <- order(e4_vij2[optim_e_vij_order], decreasing = F)
e5_vij2_order <- order(e5_vij2[optim_e_vij_order], decreasing = F)
rejection_path <- data.table(
  optim_e_vij = seq_along(optim_e_vij),
  e2_vij_order = e2_vij2_order,
  e3_vij_order = e3_vij2_order,
  e4_vij_order = e4_vij2_order,
  e5_vij_order = e5_vij2_order,
  true_diff = true_diff2[optim_e_vij_order]
)
rejection_path <- melt(rejection_path,
                       id.vars = c("optim_e_vij", "true_diff"),
                       variable.name = "order_type",
                       value.name = "order")
rejection_path[, order_type := fcase(
  order_type == "e2_vij_order", paste0("eps = ", e2),
  order_type == "e3_vij_order", paste0("eps = ", e3),
  order_type == "e4_vij_order", paste0("eps = ", e4),
  order_type == "e5_vij_order", paste0("eps = ", e5)
)]
sim_vij2_order_graph <- ggplot() +
  geom_point(data = rejection_path,
             aes(x = optim_e_vij, y = order, color = true_diff,
                 shape = true_diff),
             alpha = 1, size = 1) +
  #geom_vline(xintercept = nrow(ij_list) - sum(optim_e_vij == 1)) +
  facet_grid(~order_type) +
  labs(x = paste0("Rank of tau_ij(", round(e, digits = 3), ")"),
       y = "Rank of tau_ij(eps)") +
  theme_bw() +
  scale_color_manual(name = paste0("eps = ", round(e, digits = 3)),
                     labels = c("No difference", "True difference"),
                     values = c("FALSE" = "red", "TRUE" = "dodgerblue")) +
  scale_shape_manual(name = paste0("eps = ", round(e, digits = 3)),
                     labels = c("No difference", "True difference"),
                     values = c("FALSE" = 2, "TRUE" = 7)) +
  theme(legend.position = "bottom")
sim_vij2_order_graph
```



We compute a rank stability score for each type of difference probability as the Spearman correlation between the top 40 difference probabilities when increasing the optimal ϵ_{CE} value (obtained via minimizing conditional entropy) by a factor of 3 versus the top 40 difference probabilities when decreasing the optimal ϵCE value by a factor of 3.

```
# examine top 40 rankings
indx1 <- optim_e_vij >= sort(optim_e_vij)[100]
sum(indx1)
```

```
## [1] 40
```

```
rank_stability_score <- cor(rank(e2_vij[indx1]), rank(e5_vij[indx1]))
indx2 <- optim_e_vij2 >= sort(optim_e_vij2)[100]
rank_stability_score2 <- cor(rank(e2_vij2[indx2]), rank(e5_vij2[indx2]))
Wt2 <- DescTools::KendallW(cbind(rank(e2_vij2[indx2]), rank(e5_vij2[indx2])),
                    correct = TRUE, test = TRUE)
data.table(
  "Difference Type" = c("Standardized Difference", "Unstandardized Difference"),
  "Rank Stability Score" = c(rank_stability_score, rank_stability_score2)
)
```

```
##                Difference Type Rank Stability Score
##                         <char>                <num>
## 1:   Standardized Difference            0.9726079
## 2: Unstandardized Difference            0.8410882
```

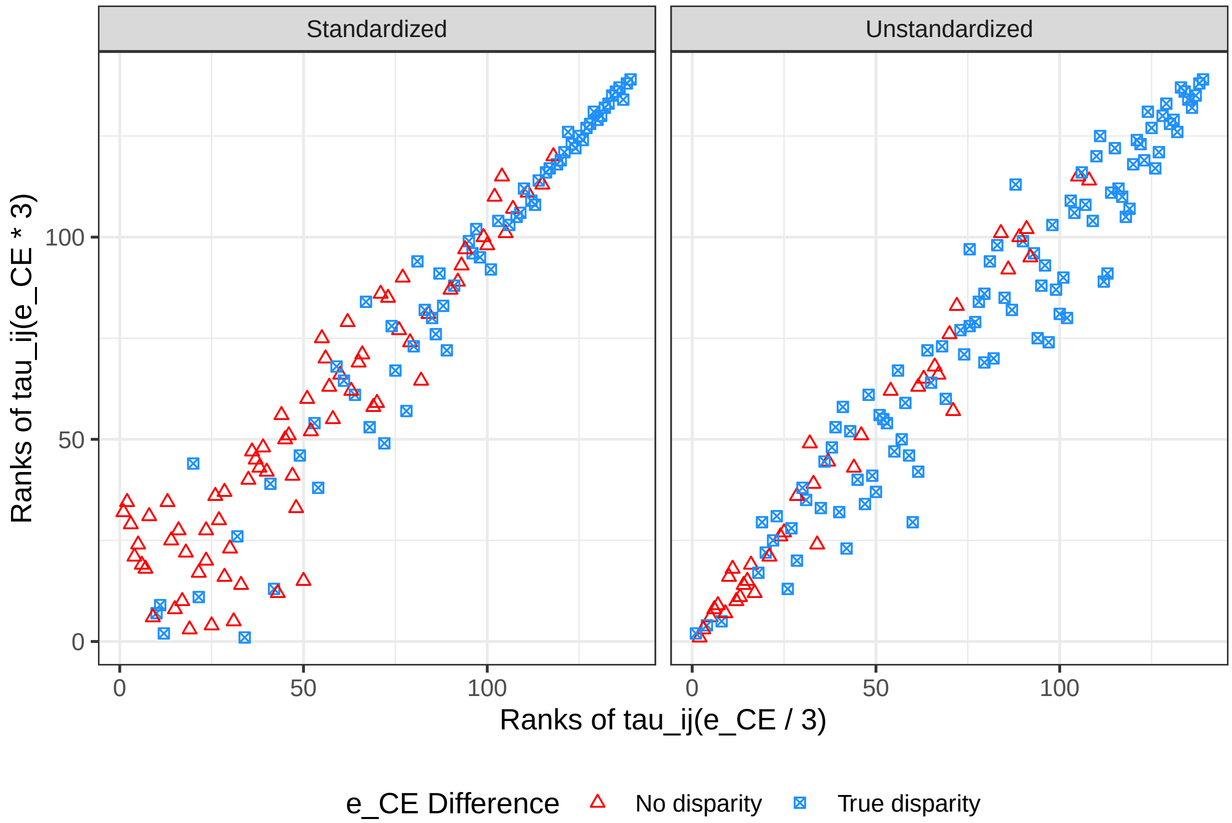
AUC of the ROC curve from each classification method:

```r
roc_list <- list(
  "Standardized Difference" = pROC::roc(as.vector(true_diff), as.vector(optim_e_vij)),
  "Unstandardized Difference" = pROC::roc(as.vector(true_diff2), as.vector(optim_e_vij2))
)
auc_values <- vapply(roc_list, function(x) x$auc, numeric(1))
auc_values
```

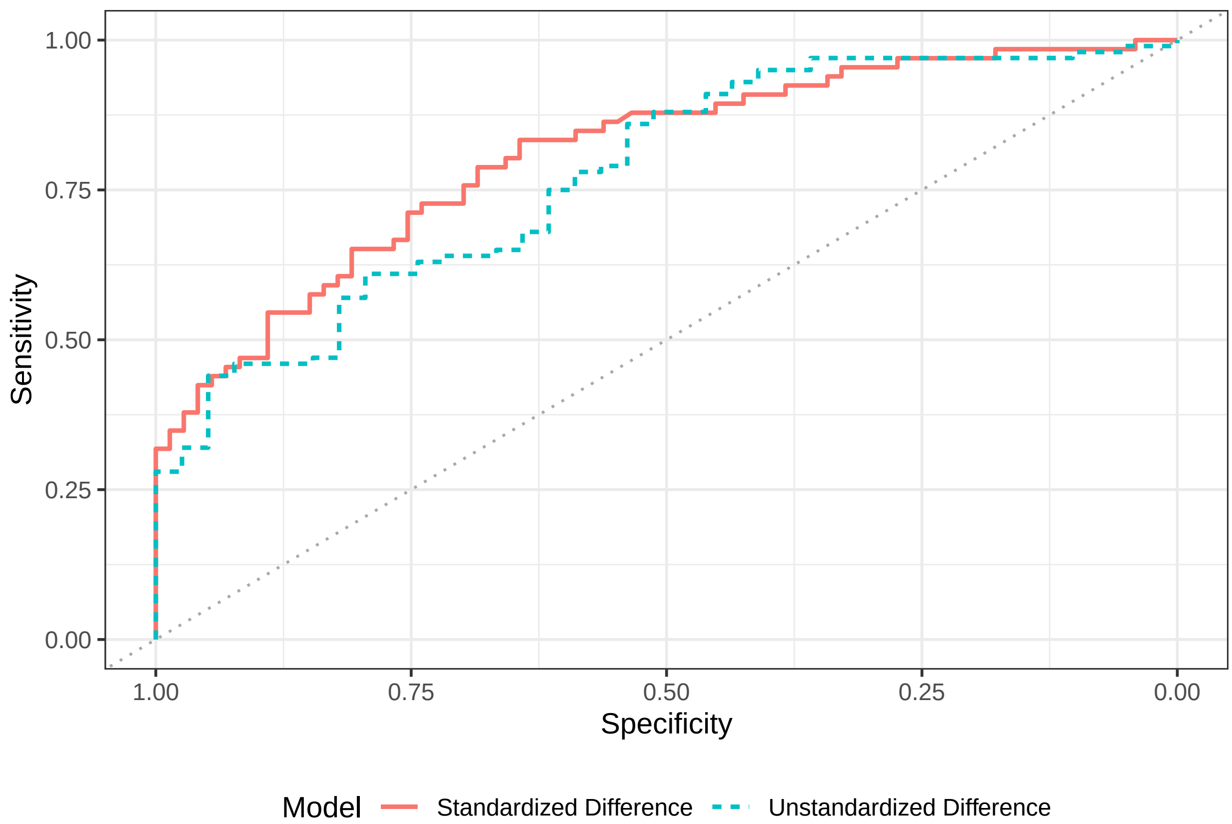
```
##   Standardized Difference Unstandardized Difference
##                 0.8093607                 0.7753846
```

Rate map, log rate map, ROC curve of ϵ-difference method (standardized) and rejection path graph:

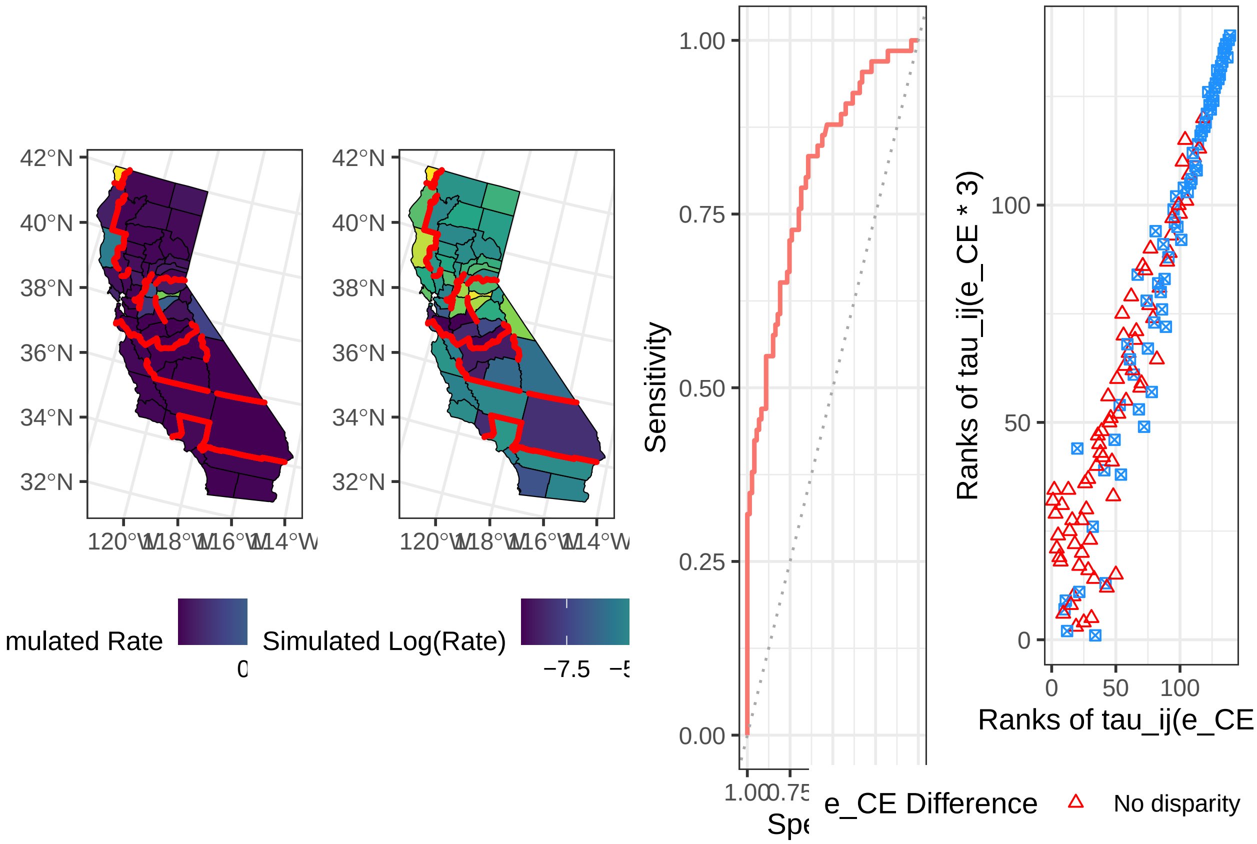
```r
tau_df <- data.table(
  diff_prob = rep(c("Standardized", "Unstandardized"), each = nrow(adj_df)),
  e1 = rep(c(round(optim_e / 3, digits = 3),
             round(e / 3, digits = 3)), each = nrow(adj_df)),
  e2 = rep(c(round(optim_e * 3, digits = 3),
              round(e * 3, digits = 3))),
  tau1_rank = c(rank(e2_vij), rank(e2_vij2)),
  tau2_rank = c(rank(e5_vij), rank(e5_vij2)),
  true_diff = c(true_diff, true_diff2)
)

stability_plot <- ggplot(data = tau_df) +
  geom_point(aes(x = tau1_rank, y = tau2_rank, color = true_diff, shape = true_diff)) +
  facet_grid(~diff_prob) +
  scale_color_manual(name = "e_CE Difference",
                     labels = c("No disparity", "True disparity"),
                     values = c("FALSE" = "red", "TRUE" = "dodgerblue")) +
  scale_shape_manual(name = "e_CE Difference",
                     labels = c("No disparity", "True disparity"),
                     values = c("FALSE" = 2, "TRUE" = 7)) +
  labs(x = "Ranks of tau_ij(e_CE / 3)",
       y = "Ranks of tau_ij(e_CE * 3)") +
  theme_bw() +
  theme(legend.position = "bottom")
stability_plot
```




```r
stability_plot2 <- ggplot(data = tau_df[diff_prob == "Standardized",]) +
  geom_point(aes(x = tau1_rank, y = tau2_rank, color = true_diff, shape = true_diff)) +
  scale_color_manual(name = "e_CE Difference",
                     labels = c("No disparity", "True disparity"),
                     values = c("FALSE" = "red", "TRUE" = "dodgerblue")) +
  scale_shape_manual(name = "e_CE Difference",
                     labels = c("No disparity", "True disparity"),
                     values = c("FALSE" = 2, "TRUE" = 7)) +
  labs(x = "Ranks of tau_ij(e_CE / 3)",
       y = "Ranks of tau_ij(e_CE * 3)") +
  theme_bw() +
  theme(legend.position = "bottom")

roc_plot <- pROC::ggroc(roc_list, aes = c("colour", "linetype"), linewidth = 0.8) +
  geom_abline(intercept = 1, slope = 1, color = "darkgrey", linetype = "dotted") +
  scale_color_discrete(name = "Model") +
  scale_linetype_discrete(name = "Model") +
  theme_bw() +
  theme(legend.position = "bottom") +
  labs(x = "Specificity", y = "Sensitivity")
roc_plot
```




```r
roc_plot_s <- pROC::ggroc(roc_list[1], linewidth = 0.8) +
  geom_abline(intercept = 1, slope = 1, color = "darkgrey", linetype = "dotted") +
  #scale_color_discrete(name = "Model") +
  #scale_linetype_discrete(name = "Model") +
  theme_bw() +
  theme(legend.position = "none") +
  labs(x = "Specificity", y = "Sensitivity")

fig <- ggpubr::ggarrange(rate_map, lograte_map, roc_plot_s, stability_plot2,
                         nrow = 1)
fig
```


42°N
40°N
38°N
36°N
34°N
32°N
120°W
118°W
116°W
114°W
mulated Rate
Simulated Log(Rate)
−7.5
Sensitivity
1.00
0.75
0.50
0.25
0.00
e_CE Difference
No disparity
Ranks of tau_ij(e_CE * 3)
Ranks of tau_ij(e_CE
0
50
100