Simulation Example: Simulated Count Outcome

Kyle Lin Wu

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"))
```

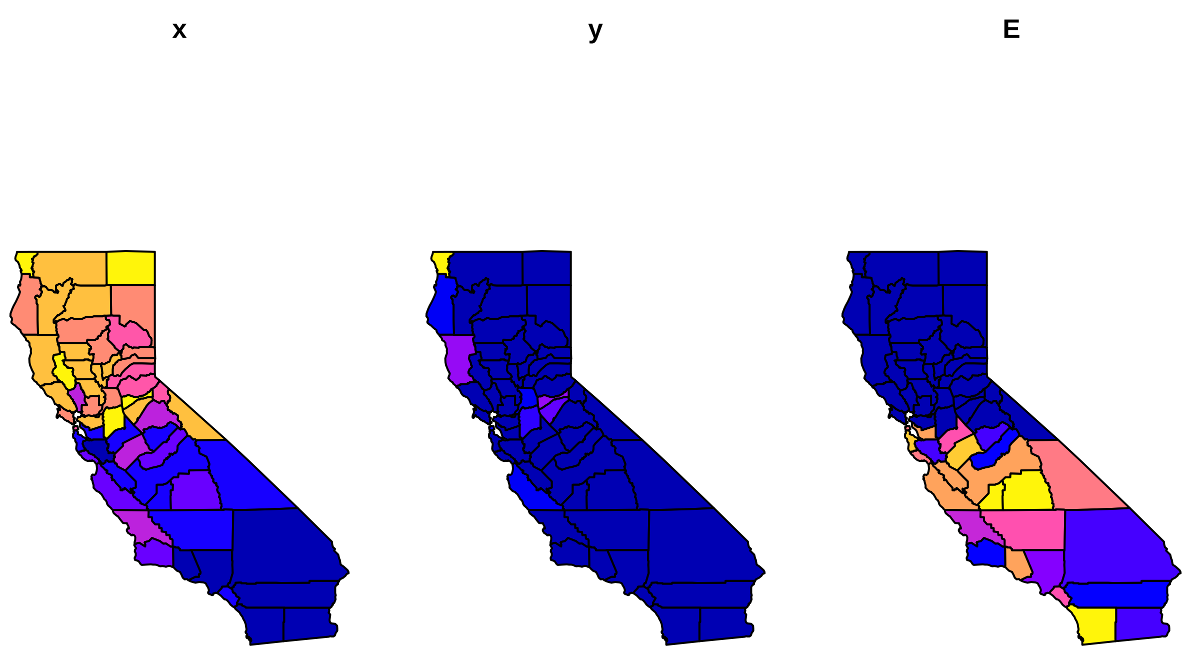
2 Data Generation

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))
```

```r
# restrict to connected county map
county_sp <- county_sp[!no_neighbors,]
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
```

```r
Sigma_analysis <- chol2inv(chol(Q_analysis))
scaling_factor <- exp(mean(log(diag(Sigma_analysis))))
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
)
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: 'Apple clang version 14.0.3 (clang-1403.0.22.14.1)'
## using SDK: 'MacOSX13.3.sdk'
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Framew
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
## #include <cmath>
##          ^~~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
```

```r
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 0.67    -6.48    -5.45    -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    -1.82    -0.69    -0.11     0.48
## phi[2]       0.15    0.00 0.84    -1.51    -0.41     0.15     0.71
## phi[3]       0.31    0.00 0.85    -1.36    -0.26     0.32     0.89
## phi[4]      -0.49    0.01 0.84    -2.14    -1.06    -0.50     0.07
## phi[5]       0.24    0.00 0.84    -1.43    -0.32     0.24     0.80
## phi[6]      -0.55    0.01 0.87    -2.23    -1.13    -0.55     0.03
## phi[7]      -0.34    0.00 0.87    -2.05    -0.93    -0.34     0.23
## phi[8]       0.28    0.01 1.07    -1.88    -0.43     0.29     0.99
## phi[9]       0.28    0.01 0.87    -1.46    -0.30     0.28     0.87
## phi[10]     -0.01    0.00 0.80    -1.57    -0.55    -0.01     0.53
## phi[11]     -0.59    0.01 0.88    -2.29    -1.18    -0.59     0.00
## phi[12]     -0.08    0.01 0.94    -1.93    -0.71    -0.08     0.55
## phi[13]      1.44    0.01 1.15    -0.87     0.68     1.44     2.20
## phi[14]      0.44    0.00 0.87    -1.25    -0.15     0.44     1.02
## phi[15]      0.60    0.01 0.84    -1.05     0.04     0.60     1.17
## phi[16]      0.11    0.01 0.88    -1.60    -0.49     0.10     0.70
## phi[17]     -0.54    0.01 0.86    -2.21    -1.13    -0.55     0.03
## phi[18]     -0.31    0.01 0.91    -2.11    -0.92    -0.31     0.29
## phi[19]      1.12    0.01 0.97    -0.81     0.47     1.13     1.77
## phi[20]     -0.42    0.01 0.87    -2.11    -1.01    -0.43     0.15
```

```
## phi[21]         0.07     0.01 1.19      -2.32     -0.70      0.09      0.86
## phi[22]        -0.21     0.00 0.88      -1.93     -0.80     -0.21      0.39
## phi[23]        -0.19     0.01 0.86      -1.89     -0.77     -0.19      0.38
## phi[24]        -0.35     0.01 0.82      -1.94     -0.91     -0.36      0.20
## phi[25]        -0.48     0.01 0.97      -2.38     -1.13     -0.48      0.17
## phi[26]         0.17     0.00 0.84      -1.46     -0.39      0.18      0.74
## phi[27]         0.38     0.00 0.85      -1.28     -0.19      0.38      0.95
## phi[28]        -0.18     0.00 0.89      -1.94     -0.78     -0.18      0.41
## phi[29]        -0.01     0.01 0.94      -1.88     -0.64      0.00      0.62
## phi[30]         1.11     0.01 0.99      -0.86      0.45      1.11      1.78
## phi[31]        -0.10     0.00 0.85      -1.77     -0.67     -0.10      0.47
## phi[32]        -0.34     0.01 0.84      -1.99     -0.91     -0.35      0.22
## phi[33]         1.33     0.01 1.03      -0.73      0.65      1.34      2.02
## phi[34]         0.09     0.00 0.79      -1.47     -0.44      0.09      0.62
## phi[35]         0.20     0.00 0.86      -1.49     -0.38      0.20      0.78
## phi[36]         0.80     0.01 0.93      -1.02      0.17      0.80      1.42
## phi[37]         1.14     0.01 1.08      -0.99      0.43      1.14      1.85
## phi[38]        -0.04     0.01 1.21      -2.47     -0.82     -0.03      0.75
## phi[39]        -0.06     0.00 0.79      -1.61     -0.59     -0.06      0.48
## phi[40]         0.43     0.01 0.91      -1.35     -0.18      0.42      1.03
## phi[41]        -0.30     0.01 0.97      -2.19     -0.95     -0.30      0.35
## phi[42]         0.66     0.01 0.97      -1.28      0.02      0.67      1.32
## phi[43]        -0.26     0.00 0.81      -1.85     -0.81     -0.26      0.29
## phi[44]         0.08     0.00 0.89      -1.68     -0.52      0.08      0.67
## phi[45]        -0.49     0.01 0.87      -2.18     -1.07     -0.49      0.09
## phi[46]        -0.12     0.01 0.90      -1.90     -0.73     -0.12      0.48
## phi[47]        -0.38     0.01 0.91      -2.15     -0.99     -0.38      0.23
## phi[48]        -0.33     0.00 0.88      -2.05     -0.91     -0.33      0.26
## phi[49]        -0.33     0.01 0.92      -2.13     -0.95     -0.33      0.29
## phi[50]        -0.26     0.00 0.80      -1.83     -0.81     -0.26      0.27
## phi[51]        -0.43     0.00 0.83      -2.05     -0.99     -0.43      0.13
## phi[52]        -0.54     0.01 0.85      -2.19     -1.11     -0.55      0.03
## phi[53]        -0.50     0.01 0.89      -2.24     -1.10     -0.51      0.09
## phi[54]         0.29     0.01 0.90      -1.45     -0.31      0.29      0.89
## phi[55]         0.03     0.00 0.81      -1.56     -0.51      0.04      0.58
## phi[56]         0.61     0.01 0.99      -1.33     -0.05      0.61      1.27
## phi[57]        -0.28     0.00 0.83      -1.92     -0.84     -0.27      0.28
## phi[58]        -0.25     0.00 0.85      -1.91     -0.82     -0.26      0.31
## sigma2          2.69     0.01 0.76       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]       13.32     0.01 0.76      11.89     12.84     13.28     13.77
## alpha[2]        9.54     0.00 0.68       8.14      9.15      9.55      9.94
## alpha[3]       11.66     0.01 0.78      10.02     11.20     11.70     12.15
## alpha[4]        9.79     0.00 0.69       8.36      9.39      9.81     10.20
## alpha[5]       12.25     0.01 0.71      10.76     11.84     12.28     12.68
## alpha[6]        8.26     0.01 0.76       6.67      7.81      8.29      8.73
## alpha[7]        9.03     0.01 0.73       7.50      8.60      9.06      9.48
## alpha[8]       12.47     0.01 0.79      10.81     12.00     12.51     12.97
## alpha[9]       12.03     0.00 0.68      10.65     11.64     12.03     12.41
## alpha[10]      11.56     0.01 0.78      10.10     11.06     11.51     12.02
## alpha[11]       7.88     0.01 0.75       6.31      7.44      7.91      8.34
## alpha[12]      11.40     0.00 0.70       9.95     11.00     11.42     11.82
## alpha[13]      13.98     0.01 0.78      12.53     13.49     13.94     14.44
## alpha[14]      13.60     0.01 0.74      12.19     13.13     13.56     14.03
```

```
## alpha[15]     13.92     0.01 0.73     12.54     13.48     13.89     14.34
## alpha[16]     11.51     0.01 0.75     10.09     11.03     11.47     11.94
## alpha[17]      7.98     0.01 0.79      6.32      7.51      8.01      8.47
## alpha[18]     10.34     0.00 0.68      8.93      9.95     10.35     10.73
## alpha[19]     15.10     0.01 0.81     13.60     14.57     15.05     15.58
## alpha[20]      8.33     0.01 0.75      6.85      7.87      8.32      8.77
## alpha[21]     11.01     0.01 0.71      9.54     10.60     11.04     11.44
## alpha[22]     11.48     0.01 0.77     10.04     11.00     11.44     11.93
## alpha[23]     12.19     0.01 0.73     10.65     11.76     12.22     12.63
## alpha[24]     10.35     0.00 0.69      9.00      9.95     10.34     10.74
## alpha[25]      9.78     0.01 0.76      8.18      9.33      9.82     10.25
## alpha[26]     10.70     0.01 0.72      9.19     10.28     10.73     11.14
## alpha[27]     14.56     0.01 0.71     13.19     14.13     14.52     14.96
## alpha[28]     11.26     0.00 0.68      9.91     10.87     11.25     11.65
## alpha[29]     10.34     0.00 0.69      8.91      9.94     10.36     10.75
## alpha[30]     13.74     0.01 0.74     12.33     13.28     13.70     14.17
## alpha[31]     10.45     0.00 0.68      9.06     10.06     10.45     10.83
## alpha[32]     10.16     0.00 0.68      8.78      9.77     10.16     10.54
## alpha[33]     13.51     0.01 0.80     12.03     13.00     13.47     13.99
## alpha[34]     11.97     0.00 0.68     10.56     11.58     11.99     12.37
## alpha[35]     14.61     0.01 0.77     13.17     14.12     14.56     15.06
## alpha[36]     11.61     0.01 0.85     10.02     11.05     11.56     12.12
## alpha[37]     13.58     0.01 0.79     12.12     13.08     13.54     14.05
## alpha[38]     13.01     0.00 0.67     11.64     12.62     13.01     13.39
## alpha[39]     10.63     0.01 0.79      8.96     10.15     10.67     11.13
## alpha[40]     12.77     0.00 0.69     11.42     12.37     12.75     13.16
## alpha[41]     11.60     0.01 0.77     10.15     11.12     11.56     12.05
## alpha[42]     12.40     0.01 0.71     11.03     11.97     12.36     12.80
## alpha[43]      9.75     0.01 0.83      8.19      9.23      9.72     10.26
## alpha[44]     13.23     0.01 0.72     11.84     12.79     13.19     13.64
## alpha[45]      8.89     0.01 0.74      7.33      8.45      8.93      9.35
## alpha[46]     10.74     0.00 0.68      9.34     10.36     10.76     11.14
## alpha[47]      9.35     0.01 0.72      7.84      8.93      9.38      9.78
## alpha[48]      8.79     0.00 0.69      7.37      8.39      8.80      9.19
## alpha[49]      8.92     0.01 0.74      7.36      8.48      8.96      9.38
## alpha[50]     10.45     0.01 0.78      8.98      9.95     10.41     10.91
## alpha[51]      8.93     0.01 0.76      7.34      8.49      8.97      9.40
## alpha[52]      9.02     0.01 0.71      7.53      8.61      9.04      9.44
## alpha[53]      8.66     0.01 0.72      7.15      8.24      8.69      9.10
## alpha[54]     13.40     0.01 0.72     12.02     12.96     13.36     13.81
## alpha[55]     11.84     0.00 0.67     10.49     11.45     11.83     12.22
## alpha[56]     12.69     0.01 0.84     11.14     12.15     12.64     13.19
## alpha[57]     10.45     0.01 0.72      8.94     10.03     10.48     10.89
## alpha[58]     10.50     0.01 0.72      9.00     10.08     10.53     10.94
## lp__      677222.62     0.07 9.42 677203.30 677216.46 677222.93 677229.14
##               97.5% n_eff Rhat
## beta[1]       -3.72 20659    1
## beta[2]        1.01 10906    1
## phi[1]         1.59 33265    1
## phi[2]         1.79 30134    1
## phi[3]         1.97 29547    1
## phi[4]         1.18 27279    1
## phi[5]         1.88 29243    1
## phi[6]         1.18 27970    1
```

```
## phi[7]          1.38 30608    1
## phi[8]          2.37 27810    1
## phi[9]          1.98 28297    1
## phi[10]         1.57 30136    1
## phi[11]         1.16 27509    1
## phi[12]         1.74 28965    1
## phi[13]         3.69 24958    1
## phi[14]         2.12 30488    1
## phi[15]         2.24 28039    1
## phi[16]         1.84 29038    1
## phi[17]         1.17 27259    1
## phi[18]         1.47 27425    1
## phi[19]         2.99 24673    1
## phi[20]         1.29 25644    1
## phi[21]         2.41 32641    1
## phi[22]         1.51 31921    1
## phi[23]         1.48 29201    1
## phi[24]         1.27 26726    1
## phi[25]         1.44 28830    1
## phi[26]         1.82 32181    1
## phi[27]         2.03 29312    1
## phi[28]         1.57 32108    1
## phi[29]         1.83 30163    1
## phi[30]         3.04 26100    1
## phi[31]         1.55 30676    1
## phi[32]         1.32 28366    1
## phi[33]         3.32 23955    1
## phi[34]         1.63 29413    1
## phi[35]         1.87 31002    1
## phi[36]         2.63 27440    1
## phi[37]         3.28 25016    1
## phi[38]         2.33 32349    1
## phi[39]         1.49 32956    1
## phi[40]         2.21 29666    1
## phi[41]         1.64 32597    1
## phi[42]         2.57 30011    1
## phi[43]         1.35 31700    1
## phi[44]         1.82 31876    1
## phi[45]         1.25 27002    1
## phi[46]         1.63 28901    1
## phi[47]         1.43 27893    1
## phi[48]         1.41 31801    1
## phi[49]         1.51 30935    1
## phi[50]         1.33 30611    1
## phi[51]         1.21 28349    1
## phi[52]         1.15 27074    1
## phi[53]         1.26 27280    1
## phi[54]         2.06 30991    1
## phi[55]         1.61 31172    1
## phi[56]         2.56 30741    1
## phi[57]         1.36 31263    1
## phi[58]         1.41 29583    1
## sigma2          4.58  4938    1
## rho             0.74  4073    1
```

```
## alpha[1]       14.95 17038    1
## alpha[2]       10.93 20754    1
## alpha[3]       13.14 17541    1
## alpha[4]       11.18 20292    1
## alpha[5]       13.65 19407    1
## alpha[6]        9.71 18314    1
## alpha[7]       10.45 18994    1
## alpha[8]       13.96 17360    1
## alpha[9]       13.41 20776    1
## alpha[10]      13.23 17009    1
## alpha[11]       9.32 18709    1
## alpha[12]      12.79 19855    1
## alpha[13]      15.65 16513    1
## alpha[14]      15.18 17575    1
## alpha[15]      15.48 18093    1
## alpha[16]      13.11 17730    1
## alpha[17]       9.47 17547    1
## alpha[18]      11.72 20612    1
## alpha[19]      16.82 15866    1
## alpha[20]       9.88 22047    1
## alpha[21]      12.40 19643    1
## alpha[22]      13.12 17392    1
## alpha[23]      13.60 18853    1
## alpha[24]      11.80 20287    1
## alpha[25]      11.23 18011    1
## alpha[26]      12.11 19151    1
## alpha[27]      16.07 18730    1
## alpha[28]      12.68 20451    1
## alpha[29]      11.73 20234    1
## alpha[30]      15.33 17561    1
## alpha[31]      11.83 20846    1
## alpha[32]      11.55 20881    1
## alpha[33]      15.23 16051    1
## alpha[34]      13.35 20505    1
## alpha[35]      16.25 16793    1
## alpha[36]      13.43 15682    1
## alpha[37]      15.27 16361    1
## alpha[38]      14.39 20759    1
## alpha[39]      12.13 17237    1
## alpha[40]      14.22 19838    1
## alpha[41]      13.24 17322    1
## alpha[42]      13.90 19012    1
## alpha[43]      11.49 18582    1
## alpha[44]      14.78 18240    1
## alpha[45]      10.33 18614    1
## alpha[46]      12.13 20585    1
## alpha[47]      10.76 19377    1
## alpha[48]      10.18 20891    1
## alpha[49]      10.36 18566    1
## alpha[50]      12.10 17694    1
## alpha[51]      10.39 18132    1
## alpha[52]      10.41 19631    1
## alpha[53]      10.07 19334    1
## alpha[54]      14.93 18451    1
```

```
## alpha[55]      13.25 20617    1
## alpha[56]      14.47 15582    1
## alpha[57]      11.87 19166    1
## alpha[58]      11.91 19250    1
## lp__       677240.10 16566    1
##
## Samples were drawn using NUTS(diag_e) at Thu Mar  6 10:34:58 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).
```

```
samps <- as.matrix(fit1)
#HDInterval::hdi(samps[,"rho"])
phi_samps <- samps[, paste0("phi[", seq_len(N), "]")]
sigma2_samps <- samps[, "sigma2"]
summary(sigma2_samps)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.095   2.168   2.549   2.693   3.052  10.345
```

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

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000242 0.1843572 0.3211718 0.3372529 0.4726903 0.9973008
```

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:

```
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
##   0.650   0.203   0.883
```

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

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

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

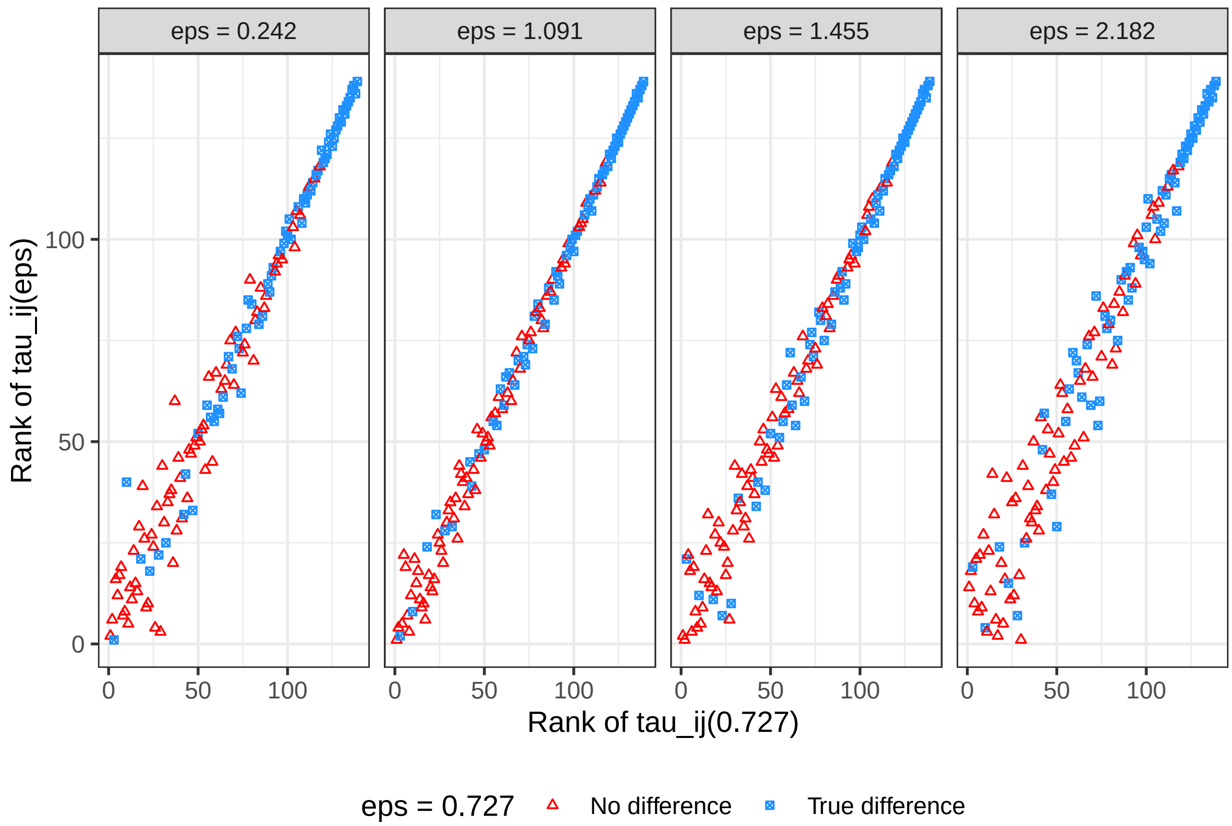
```
optim_e_vij <- ComputeSimVij(phi_diffs, epsilon = optim_e)
optim_e_vij_order <- order(optim_e_vij, decreasing = F)

# 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] 0.65
```

```
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)
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) +
```

```r
  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:

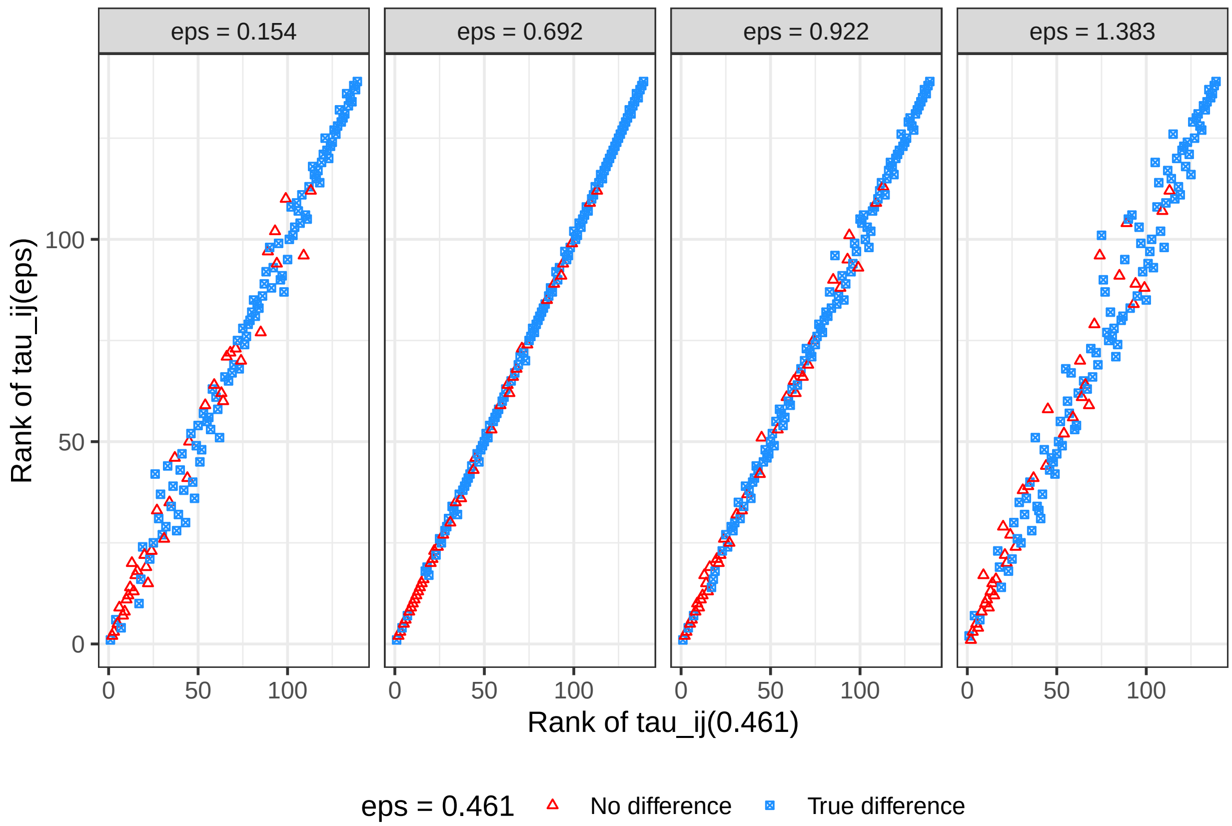
```r
# compute unstandardized difference probabilities
phi_diffs2 <- vapply(seq_len(k), function(pair_indx) {
  i <- adj_df[pair_indx,]$i
  j <- adj_df[pair_indx,]$j
  (phi_samps[,i] - phi_samps[,j])
}, numeric(n_s))

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])
```

```r
}, 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
##   0.958   0.372   1.449
```

```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) +
```

```r
  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.

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

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

```r
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            0.9617261
## 2: Unstandardized Difference            0.8569820
```

AUC of the ROC curve from each classification method:

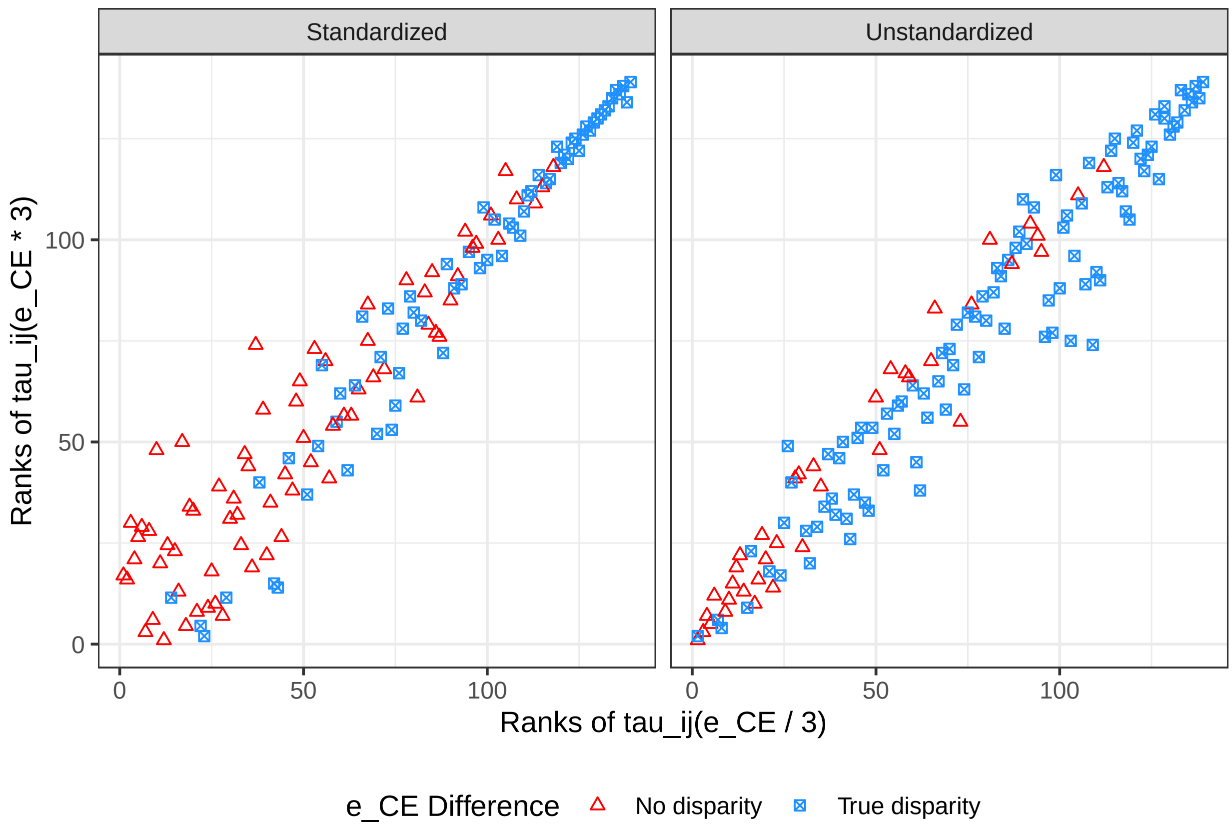
```r
roc_list <- list(
  "Standardized Difference" = pROC::roc(as.vector(true_diff), as.vector(optim_e_vij)),
  "Unstandardized Difference" = pROC::roc(as.vector(true_diff2), as.vector(optim_e_vij2))
)
auc_values <- vapply(roc_list, function(x) x$auc, numeric(1))
auc_values
```

```
##   Standardized Difference Unstandardized Difference
##                 0.8039643                 0.7683689
```

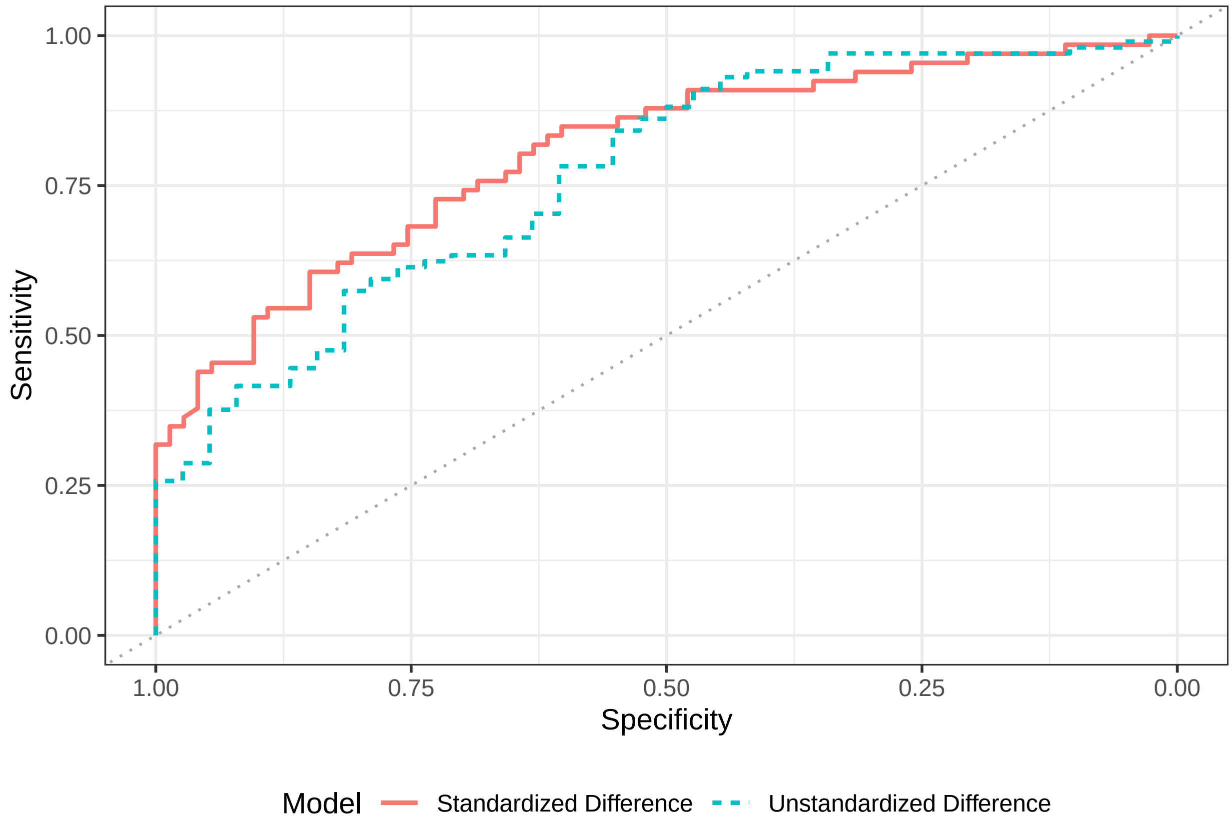
Rate map, log rate map, ROC curve of ϵ-difference method (standardized) and rejection path graph:

```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() +
```

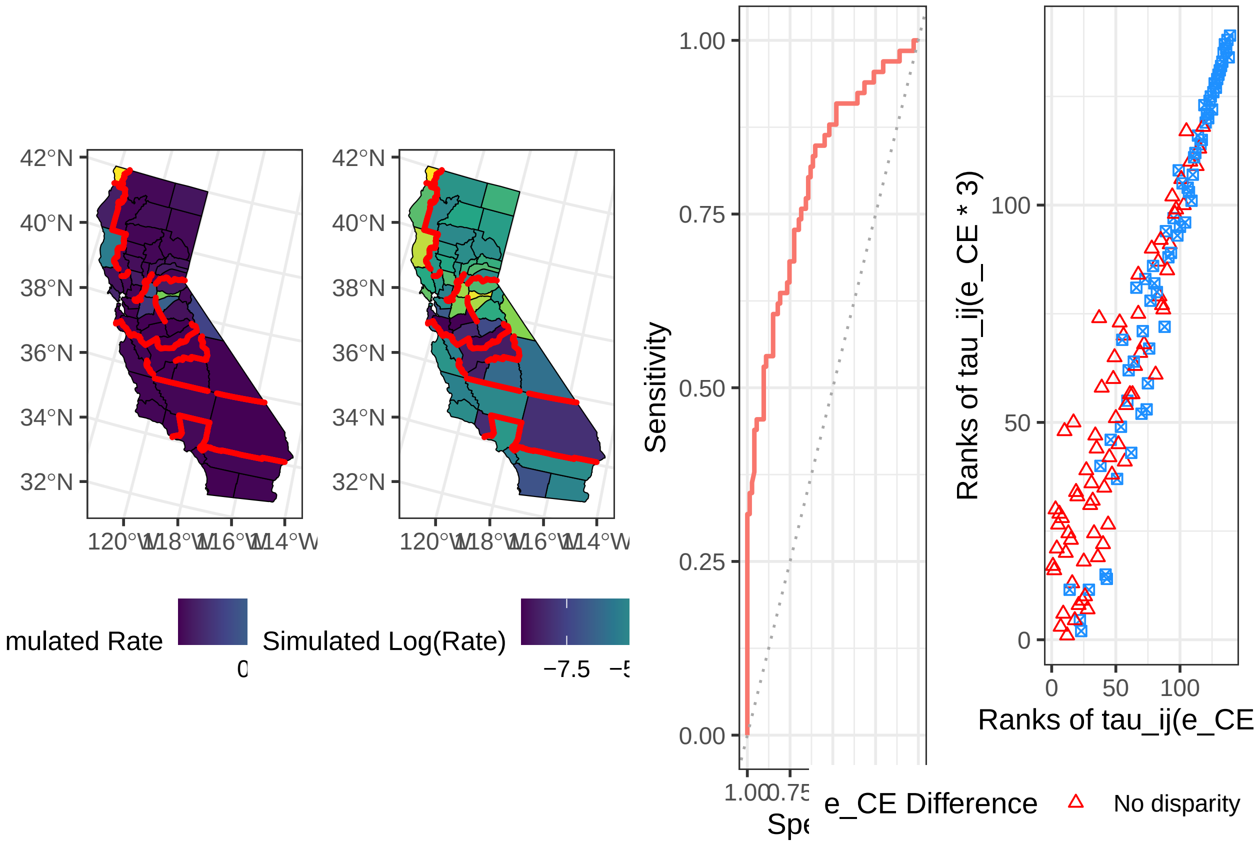
```r
  theme(legend.position = "bottom")
stability_plot
```




```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
```




```
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°W118°W116°W114°W
mulated Rate
0
Simulated Log(Rate)
−7.5 −5
Sensitivity
1.00
0.75
0.50
0.25
0.00
1.000.75
Spe
e_CE Difference
No disparity
Ranks of tau_ij(e_CE * 3)
100
50
0
Ranks of tau_ij(e_CE