Simulation Examples for Generalized Linear Models

Package Setup

```r
if (!require("devtools", quietly = TRUE))
  install.packages("devtools")
```

```
Warning: package 'usethis' was built under R version 4.4.1
```

```r
devtools::install_github("anonstats123/SyNPar")
```

```
Skipping install of 'SyNPar' from a github remote, the SHA1 (e3765e99) has not changed
since last install.
  Use `force = TRUE` to force installation
```

```r
library(PRROC)
library(SyNPar)
library(MASS)
```

```
Warning: package 'MASS' was built under R version 4.4.1
```

```r
library(knockoff)
```

Simulation Data Generation

```r
n <- 500
p <- 100
nonzero_coefs <- 20
Amp <- 9
rho <- 0.6
Theta.8 <- toeplitz(rho^(0:(p - 1)))
X <- mvrnorm(n, rep(0, p), Sigma = Theta.8)
X = scale(X)/(sqrt(n))
beta <- rep(0, p)
beta[1:nonzero_coefs] <- sample(c(-Amp, Amp), nonzero_coefs, replace = TRUE)
Signal_index <- 1:nonzero_coefs
true_labels <- beta != 0
linear_predictor <- X %*% beta
prob <- 1 / (1 + exp(-linear_predictor))  # logistic function
y <- rbinom(n, 1, prob)
```

Statistical Metrics Function

```r
fdp = function(selected) sum(beta[selected] == 0) / max(1, length(selected))
power = function(selected) sum(beta[selected] != 0) / sum(beta != 0)
```

```r
aupr = function(statistic) pr.curve(scores.class0 = statistic,
                                    weights.class0 = true_labels,
                                    curve = FALSE)$auc.integral
```

SyNPar

```r
result_SyNPar <- synpar_filter(
  X, y, fdr_value = 0.1, best_lambda = NULL, B_reps = NULL, model_type = "glm"
)
```

```
Loading required package: Matrix

Warning: package 'Matrix' was built under R version 4.4.1

Loaded glmnet 4.1-8

Warning: package 'survival' was built under R version 4.4.1

Warning: package 'eha' was built under R version 4.4.1
```

```r
SyNPar_FDR <- length(which(result_SyNPar$statistic[setdiff(1:p, Signal_index)] >=
                             result_SyNPar$threshold)) / max(length(result_SyNPar$selecte
SyNPar_Power <- length(which(result_SyNPar$statistic[Signal_index] >=
                               result_SyNPar$threshold)) / length(Signal_index)
SyNPar_AUPR <- aupr(result_SyNPar$statistic)
cat("SyNPar FDR:", SyNPar_FDR, "\n")
```

```
SyNPar FDR: 0.07142857
```

```r
cat("SyNPar Power:", SyNPar_Power, "\n")
```

```
SyNPar Power: 0.65
```

```r
cat("SyNPar AUPR:", SyNPar_AUPR, "\n")
```

```
SyNPar AUPR: 0.8638521
```

Model-X

```r
mx_knockoff = create.second_order(X)
mx_statistic = stat.glmnet_coefdiff(X, mx_knockoff, y,
                                    family = "binomial")
mx_thres = knockoff.threshold(mx_statistic, fdr = 0.1)
mx_selected = which(mx_statistic >= mx_thres)
mx_FDR = fdp(mx_selected)
mx_Power = power(mx_selected)
```

```r
mx_AUPR = aupr(abs(mx_statistic))
cat("Model-X FDR:", mx_FDR, "\n")
```

```
Model-X FDR: 0
```

```r
cat("Model-X Power:", mx_Power, "\n")
```

```
Model-X Power: 0
```

```r
cat("Model-X AUPR:", mx_AUPR, "\n")
```

```
Model-X AUPR: 0.7902266
```

Fixed-X

```r
fx_knockoff = create.fixed(X)$Xk
fx_statistic = stat.glmnet_lambdasmax(X, fx_knockoff, y,
                                      family = "binomial")
fx_thres = knockoff.threshold(fx_statistic, fdr = 0.1)
fx_selected = which(fx_statistic >= fx_thres)
fx_FDR = fdp(fx_selected)
fx_Power = power(fx_selected)
fx_AUPR = aupr(abs(fx_statistic))
cat("Fixed-X FDR:", fx_FDR, "\n")
```

```
Fixed-X FDR: 0
```

```r
cat("Fixed-X Power:", fx_Power, "\n")
```

```
Fixed-X Power: 0
```

```r
cat("Fixed-X AUPR:", fx_AUPR, "\n")
```

```
Fixed-X AUPR: 0.6559715
```

Data Splitting

```r
source("./DS.R")
source("./analys.R")
result_ds <- DS(X, y, num_split = 1, q = 0.1)
ds_FDR <- fdp(result_ds$DS_feature)
ds_Power <- power(result_ds$DS_feature)
ds_AUPR = aupr(abs(result_ds$DS_statistic))
cat("Data Splitting FDR:", ds_FDR, "\n")
```

```
Data Splitting FDR: 0
```

```r
cat("Data Splitting Power:", ds_Power, "\n")
```

```
Data Splitting Power: 0
```

```r
cat("Data Splitting AUPR:", ds_AUPR, "\n")
```

```
Data Splitting AUPR: 0.5760418
```

Multiple Data Splitting

```r
source("./DS.R")
source("./analys.R")
result_mds <- DS(X, y, num_split = 50, q = 0.1)
mds_FDR <- fdp(result_mds$MDS_feature)
mds_Power <- power(result_mds$MDS_feature)
mds_AUPR = aupr(abs(result_mds$MDS_statistic))
cat("Multiple Data Splitting FDR:", mds_FDR, "\n")
```

```
Multiple Data Splitting FDR: 0.1428571
```

```r
cat("Multiple Data Splitting Power:", mds_Power, "\n")
```

```
Multiple Data Splitting Power: 0.3
```

```r
cat("Multiple Data Splitting AUPR:", mds_AUPR, "\n")
```

```
Multiple Data Splitting AUPR: 0.7622613
```