6520 Project

Minjia Jia and Joia Zhang

Fall 2023

```
rm(list=ls())
set.seed(6520)
library(ggplot2)
library(expm) # for sqrtm
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'expm'
```

```
## The following object is masked from 'package:Matrix':
##
##     expm
```

```
# setwd("/Users/jwz34/Documents/Github/6520project/onlinegrad")
# devtools::install()
library(onlinegrad)
# .rs.restartR() # if ".rdb is corrupt"
```

Simulate data for regression and classification

```
# simulate data: regression
n = 100 # sample size
p = 200 # number of predictors

# beta
k = round(0.05*p, 0) # number of nonzero coefficients
sd_beta = 0.01
nonzero_indexes = sample.int(n=p, size=k)
beta = rep(0, p)
beta[nonzero_indexes] = rnorm(n=k, mean=100, sd=sd_beta)
sum(which(beta !=0) != sort(nonzero_indexes)) # test that we made the right indexes nonzero
```

```
## [1] 0
```

```r
beta = as.matrix(beta)

# x
X = matrix(rnorm(n=n*p, mean=0, sd=5), nrow=n)

# epsilon
E = matrix(rnorm(n=n, mean=0, sd=1), nrow=n)

# y

Y = X%*%beta + E

# note that in the online setting, each t^th row of X and Y is for time t
```

```r
# simulate data: classification
# X, beta same as above
probs = 1/(1+exp(-X%*%beta))
Y = rbinom(n=n, size=1, prob = probs) # Bernoulli
```

Analysis of $\hat{\beta}$'s

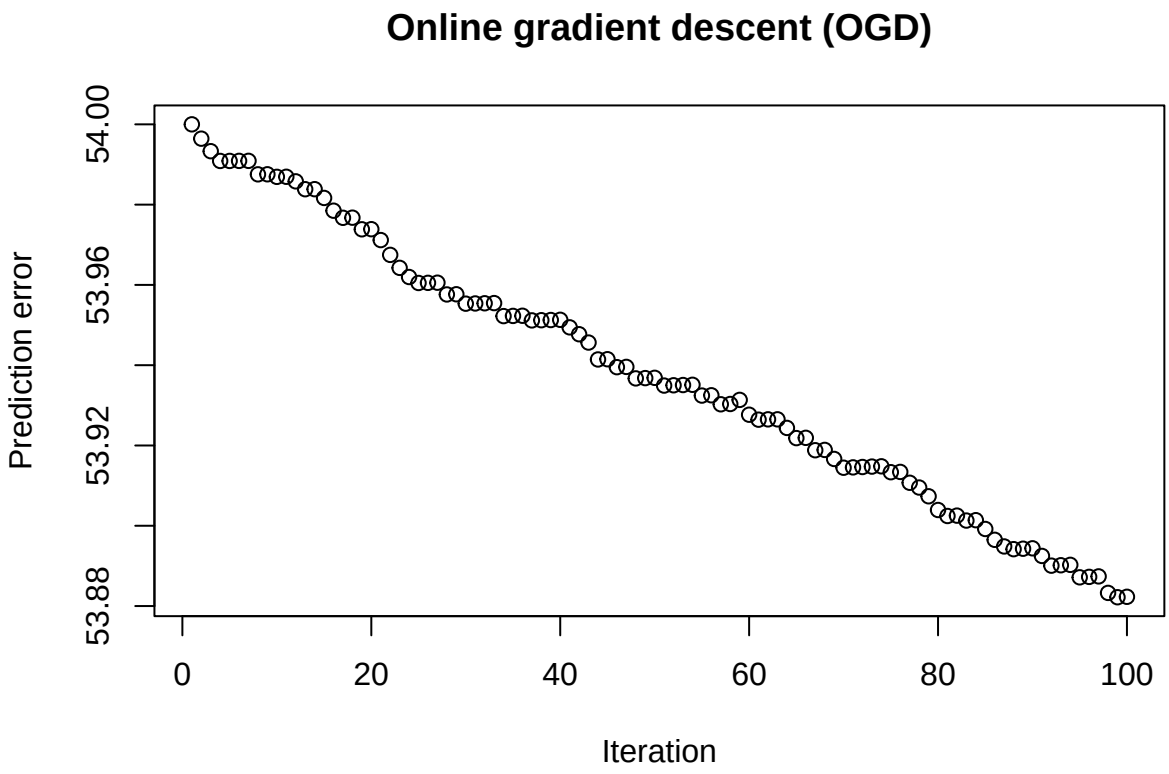
Plots + Prediction error vs iterations + Estimation error vs iterations + Betahats for each dimension, nonzero vs zero indexes - Comparison of different learning rates - Run time of full vs diagonal Adagrad + Run time of OGD, Adagrad, adam - Variance of betahats across iterations?

```r
# plot prediction or estimation error
# X: rows are observations, columns are predictors
# Y: response variable
# betahats: n x p matrix where each ith row is the coefficients for the ith iteration and the columns a
# beta: true beta coefficient px1 vector
# title: string for the title of the plot
# type: "prediction" or "estimation" for prediction error or estimation error
plot_prediction_error = function(betahats, beta, X, Y, title, type) {
  n = nrow(X)
  p = ncol(X)
  if ((type!="prediction") && (type!="estimation")) {
    stop("type parameter must be 'prediction' or 'estimation'")
  }
  if (type=="prediction") {
    # prediction error
    err = colSums((X%*%t(betahats) - matrix(rep(Y, n), nrow=n, ncol=n, byrow=F))^2) # row of the inside
    ylab = "Prediction error"
  } else {
    # estimation error
    beta = t(beta)
    beta = matrix(rep(beta, n),nrow=n, byrow=T) # row combine n number of t(beta)'s
    err = sqrt(rowSums((betahats - beta)^2))
    ylab = "Estimation error"
  }
  err = as.matrix(err)
  plot(err, xlab="Iteration", ylab=ylab, main=title)
```

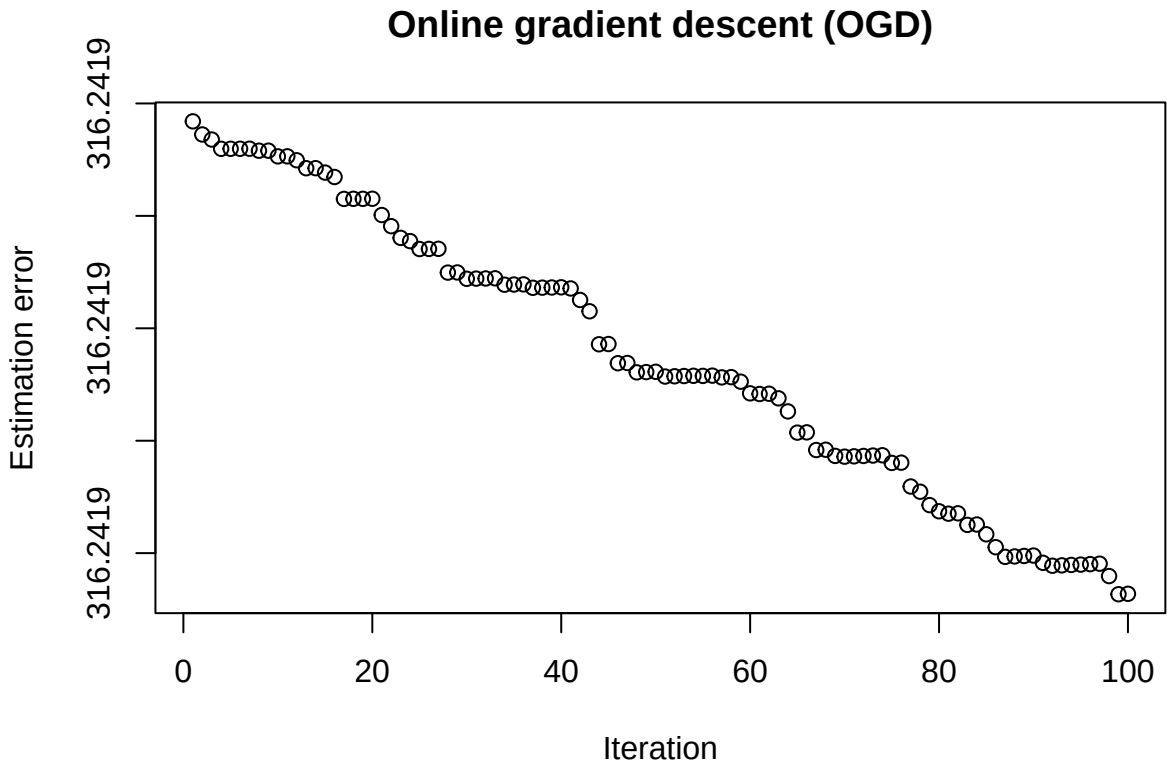
```
  return(err)
}
```

```
# plot last iteration of betahat for nonzero vs zero indexes, true beta overlaid
# nonzero_indexes: kx1 vector of indexes where the k-sparse vector true beta has nonzero values
# nonzero: boolean, if true plot only nonzero indexes (k indexes of the true k-sparse beta), otherwise
plot_betas = function(betahats, beta, nonzero_indexes, nonzero=T) {
  n = nrow(betahats)
  p = ncol(betahats)
  dat = data.frame("p"=1:p, "bethat_n"=betahats[n, ], "beta"=beta)
  if (nonzero) {
    dat = dat[nonzero_indexes, ]
    title = "Last iteration of bethat (orange) and true beta (black) at nonzero indexes"
  } else {
    dat = dat[-nonzero_indexes, ]
    title = "Last iteration of bethat (orange) and true beta (black) at zero indexes"
  }
  ggplot(dat) + geom_point(aes(x=p, y=bethat_n), color="orange") +
    geom_point(aes(x=p, y=beta), color="black", shape=4) +
    xlab("p") +
    ylab("") +
    ggtitle(title)
}
```

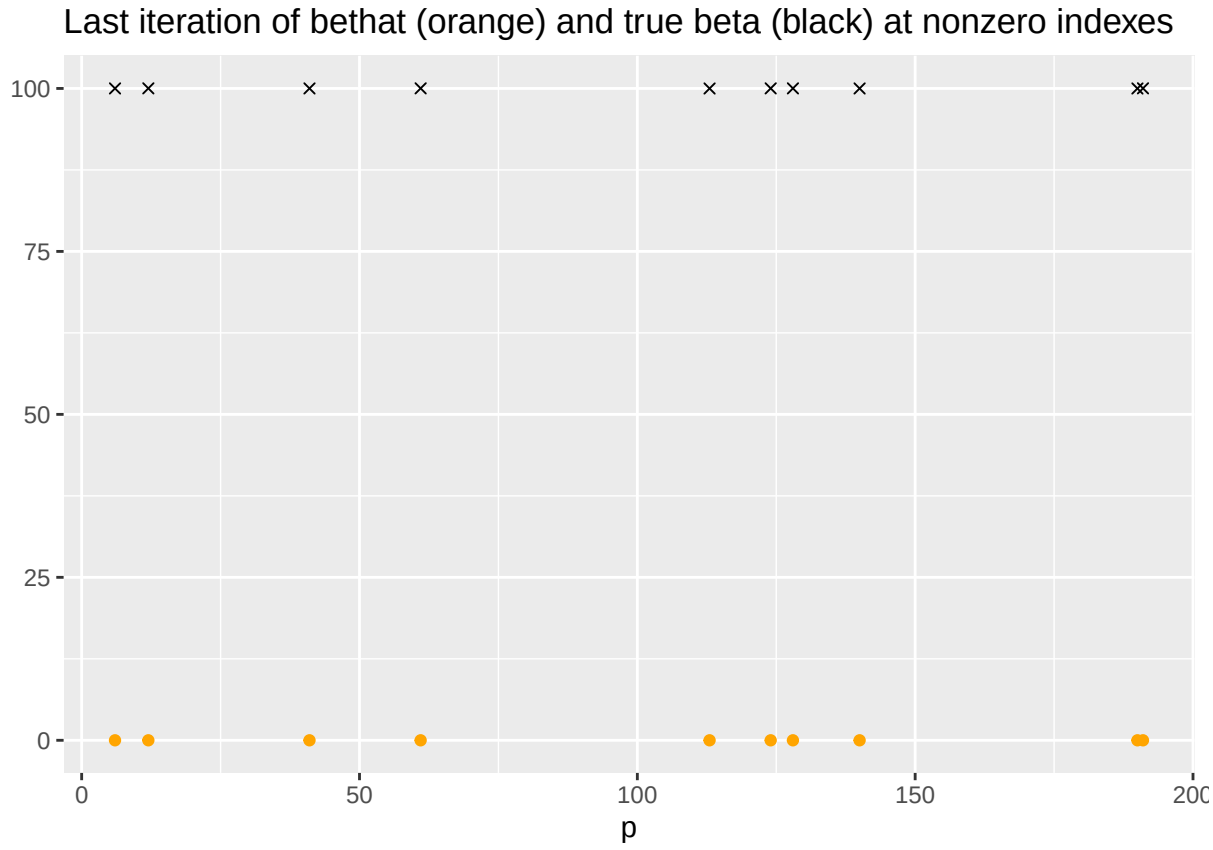
```
runtime_df = data.frame("n"=1:n)
pred_df = data.frame("n"=1:n) # prediction error
est_df = data.frame("n"=1:n) # estimation error

# OGD
temp = my_OGD(X=X, Y=Y, lr=0.0000001, beta_0=rep(0, p))
betahats = temp[[1]]
runtimes = temp[[2]]
runtime_df$ogd = runtimes

pred_df$ogd = plot_prediction_error(betahats, beta, X, Y, title="Online gradient descent (OGD)", type="
```



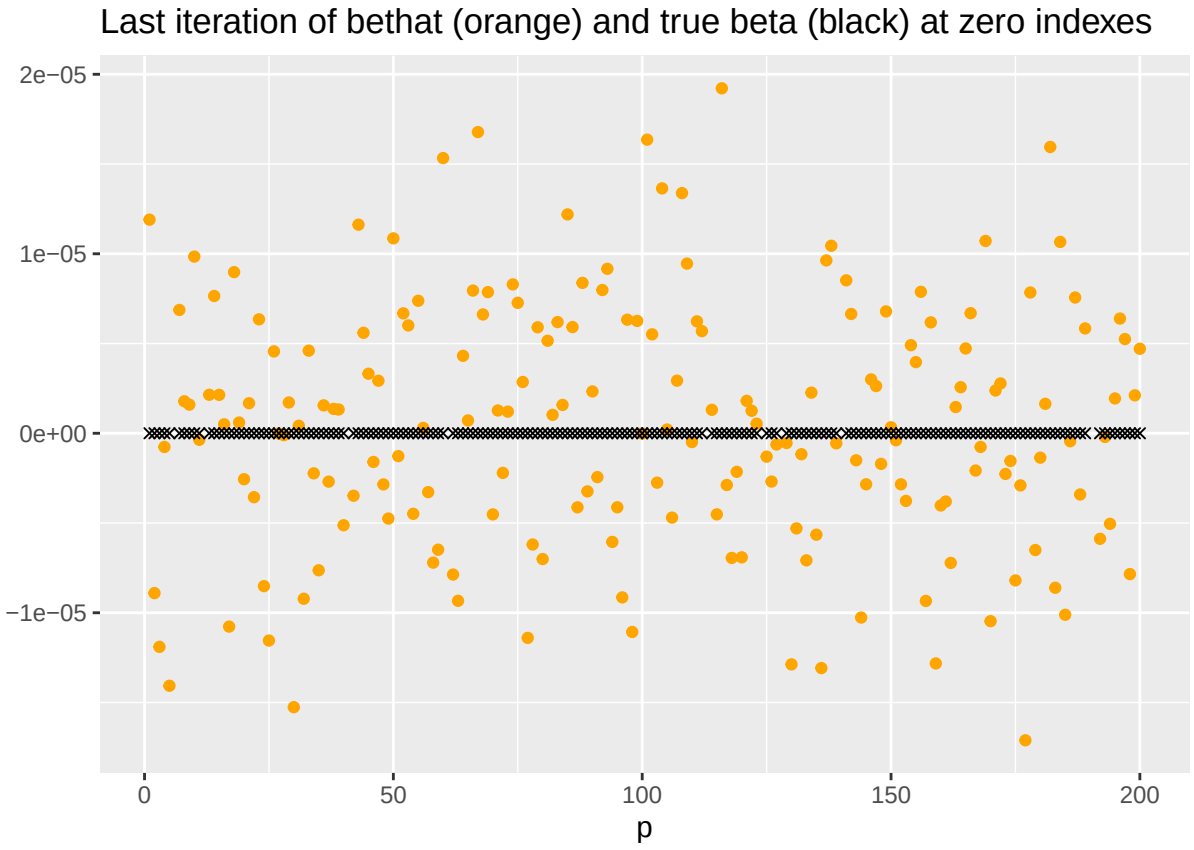

```
est_df$ogd = plot_prediction_error(betahats, beta, X, Y, title="Online gradient descent (OGD)", type="es
```



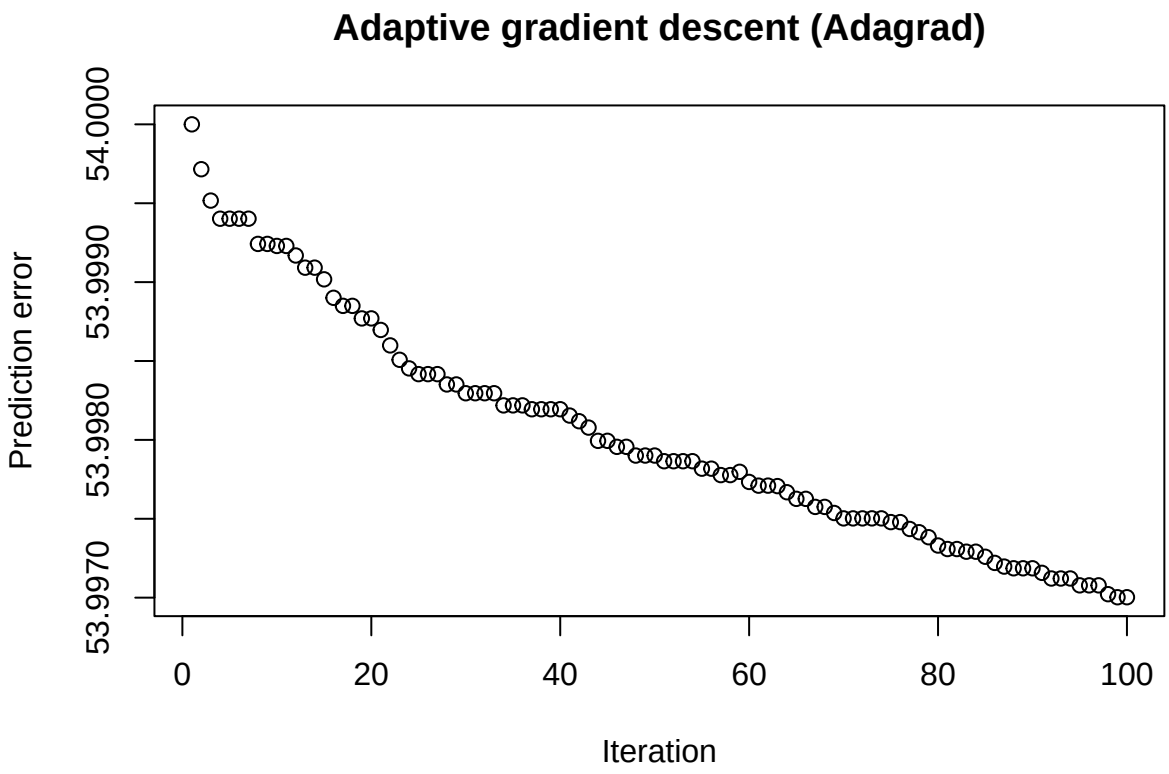

```
plot_betas(betahats, beta, nonzero_indexes, nonzero=T) # nonzero indexes
```



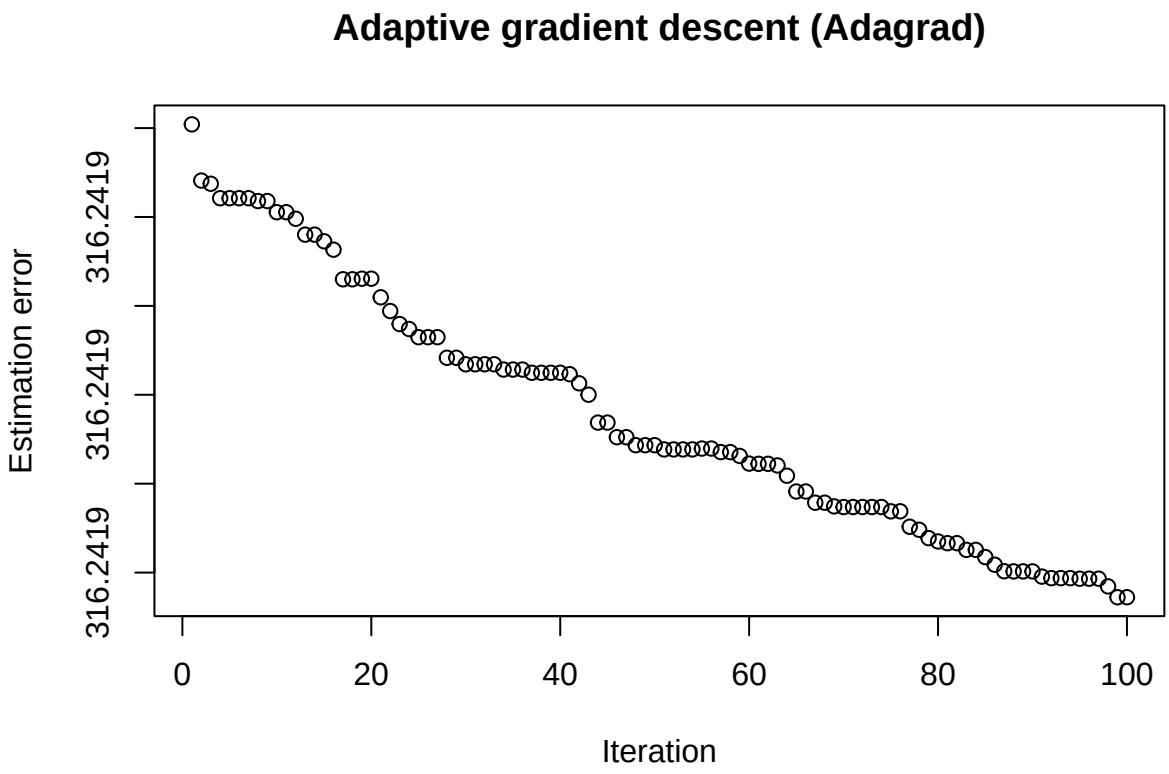

```
plot_betas(betahats, beta, nonzero_indexes, nonzero=F) # zero indexes
```



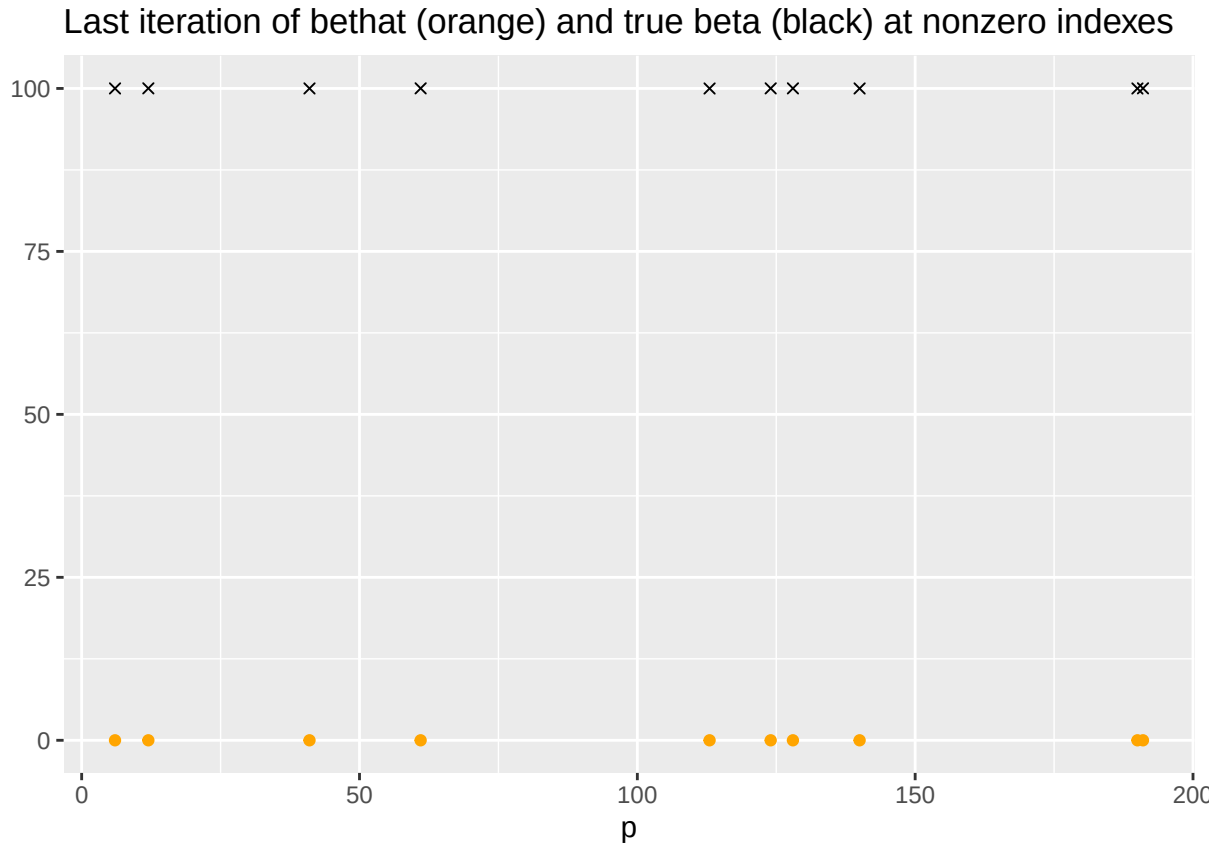

```
# Adagrad
temp = my_adagrad(X=X, Y=Y, lr=0.0000001, beta_0=rep(0, p), full=F)
betahats = temp[[1]]
runtimes = temp[[2]]
runtime_df$adagrad = runtimes

pred_df$adagrad = plot_prediction_error(betahats, beta, X, Y, title="Adaptive gradient descent (Adagrad
```



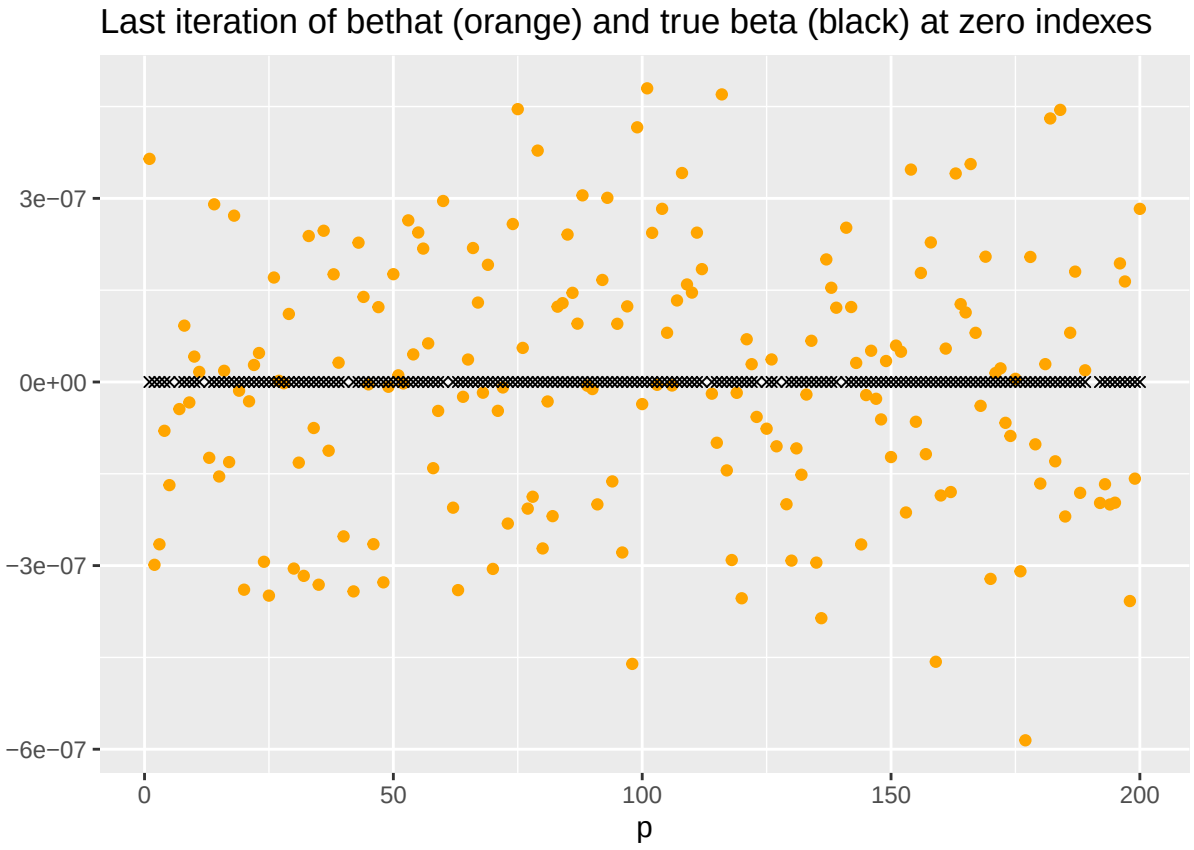

```
est_df$adagrad = plot_prediction_error(betahats, beta, X, Y, title="Adaptive gradient descent (Adagrad)")
```



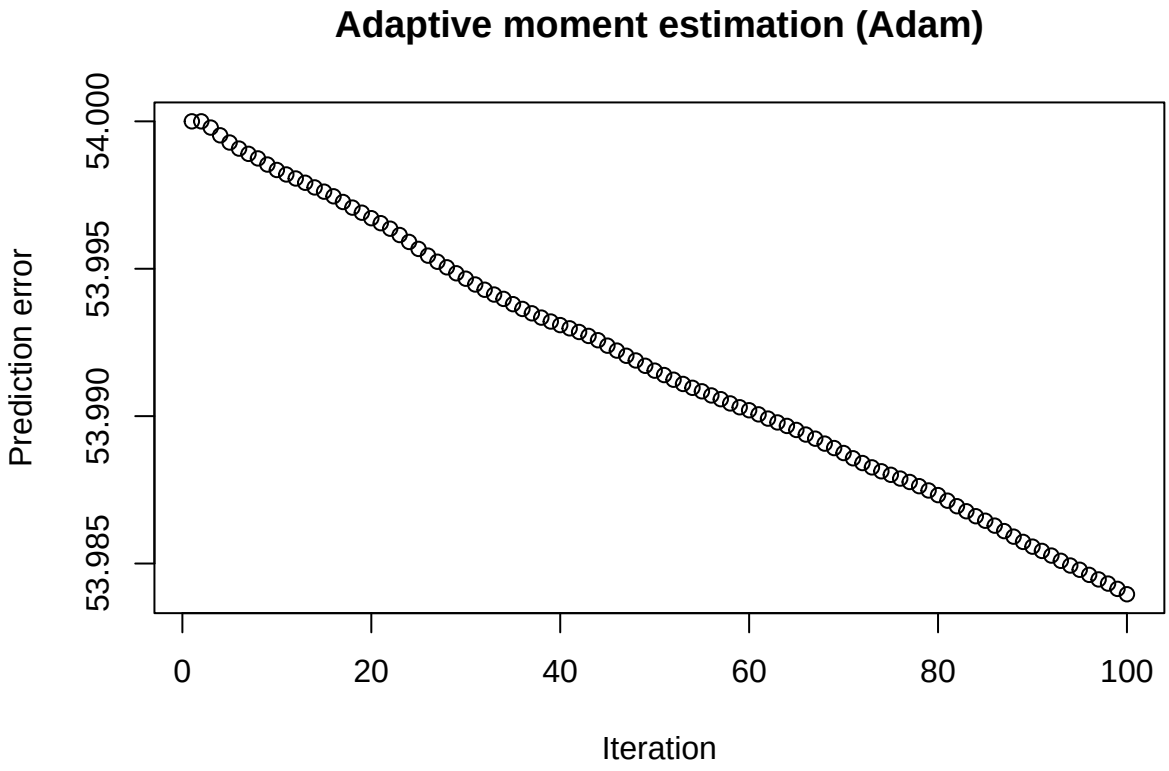

```
plot_betas(betahats, beta, nonzero_indexes, nonzero=T)
```



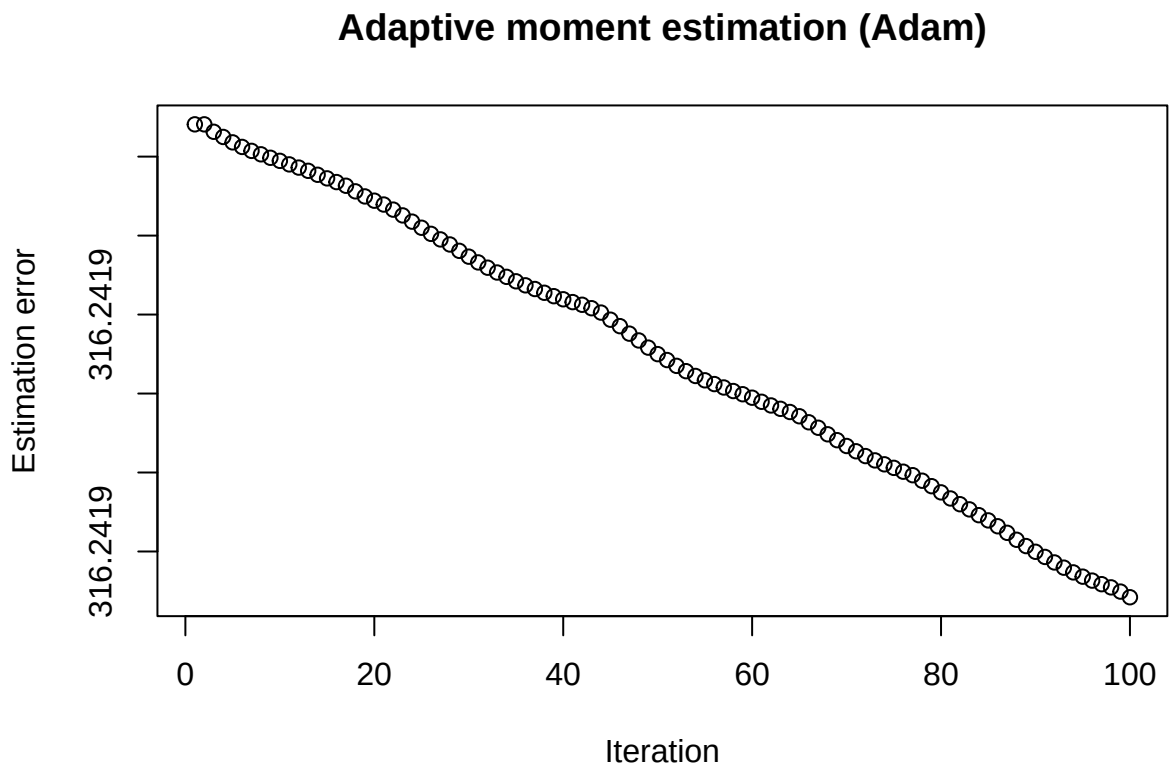

```
plot_betas(betahats, beta, nonzero_indexes, nonzero=F)
```



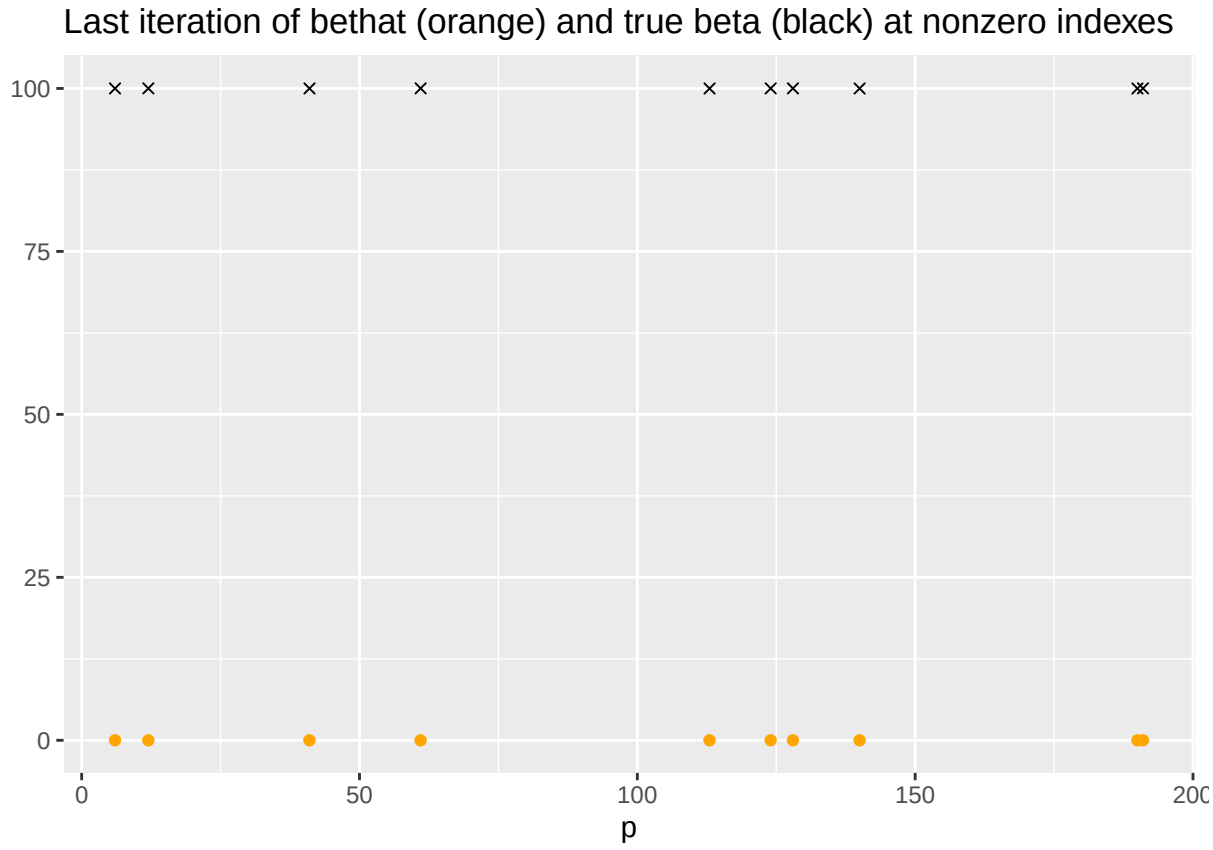

```
# Adam
temp = my_adam(X=X, Y=Y, lr=0.0000001, beta_0=rep(0, p), rho_1=0.9, rho_2=0.999, epsilon=1e-8)
betahats = temp[[1]]
runtimes = temp[[2]]
runtime_df$adam = runtimes

pred_df$adam = plot_prediction_error(betahats, beta, X, Y, title="Adaptive moment estimation (Adam)", ty
```



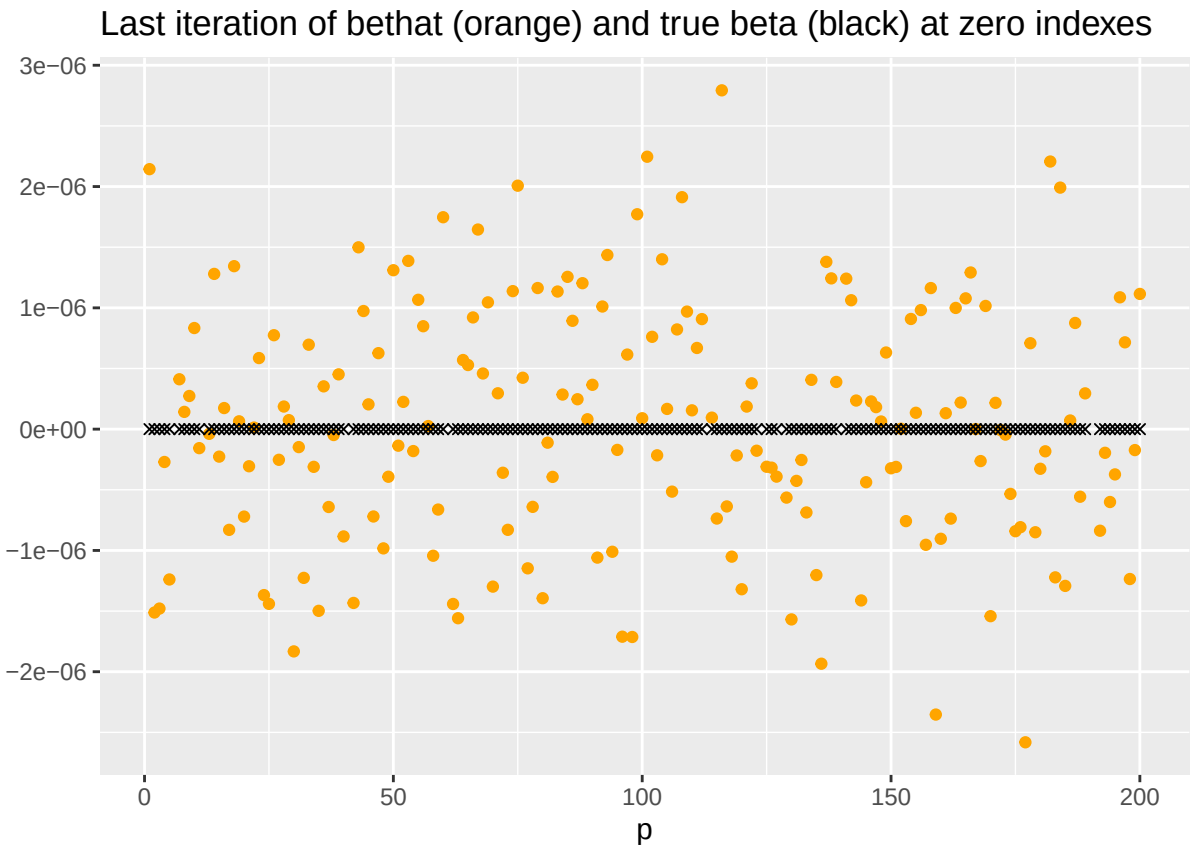

```
est_df$adam = plot_prediction_error(betahats, beta, X, Y, title="Adaptive moment estimation (Adam)", ty
```



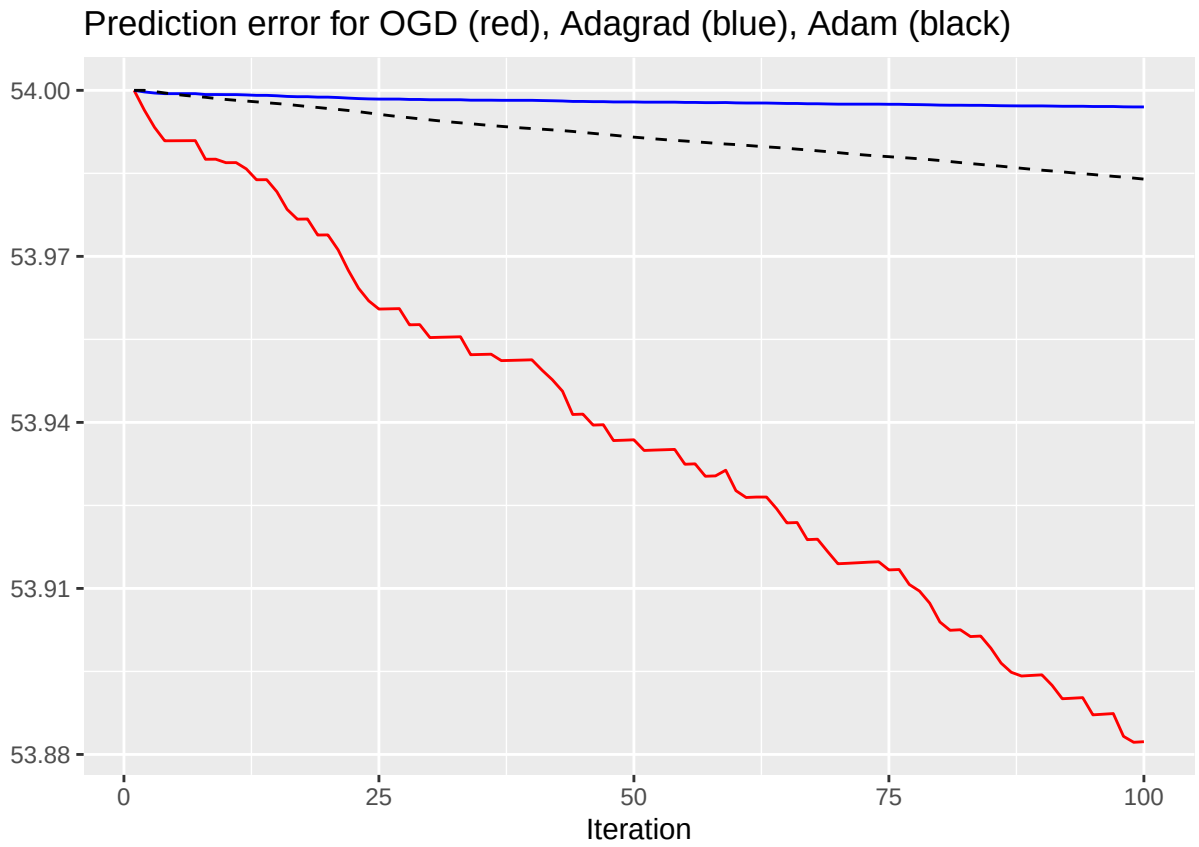

```
plot_betas(betahats, beta, nonzero_indexes, nonzero=T)
```



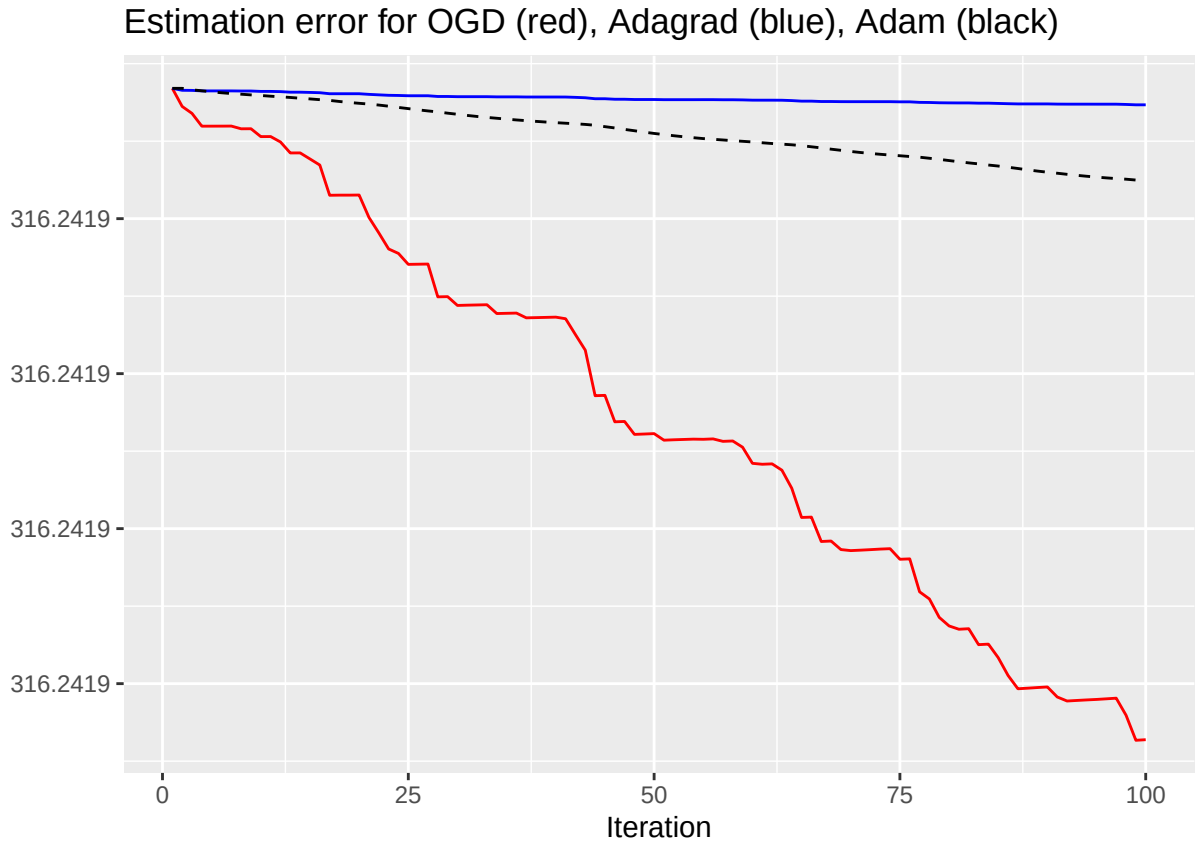

```
plot_betas(betahats, beta, nonzero_indexes, nonzero=F)
```



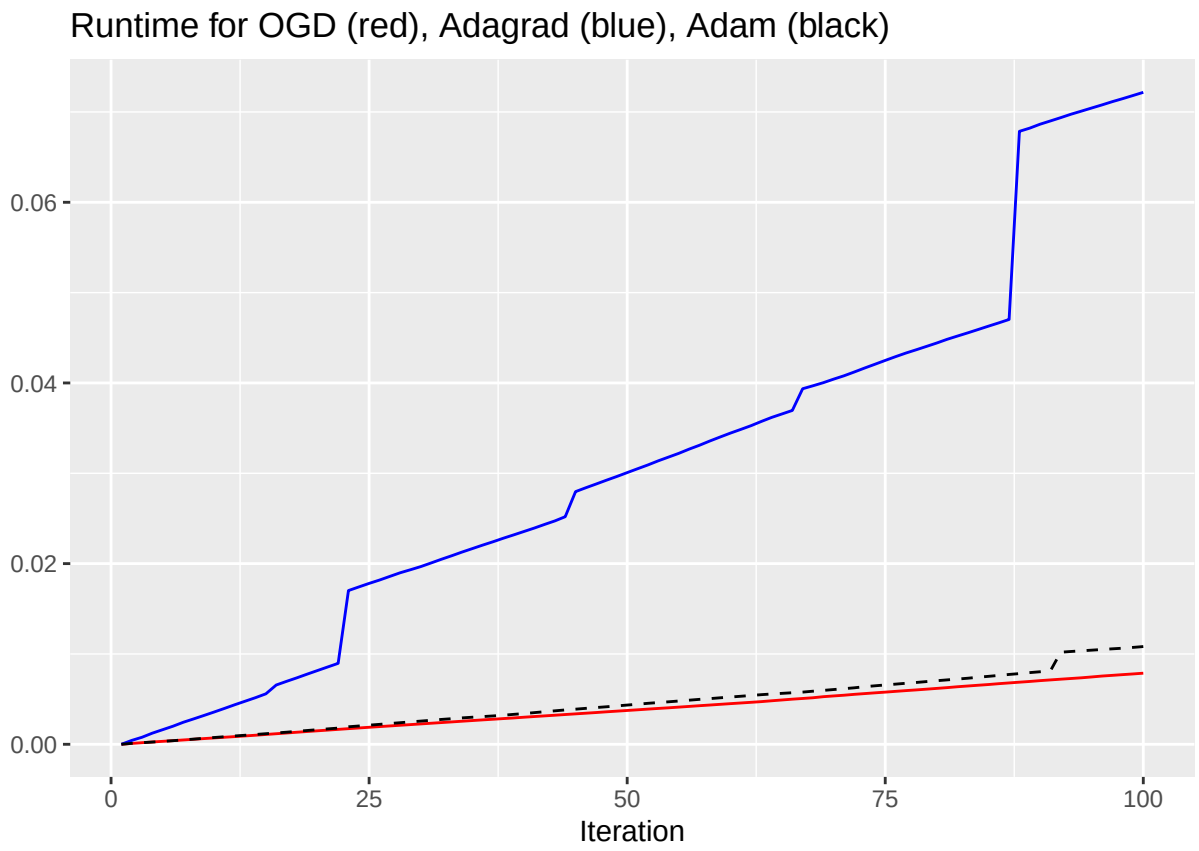

```r
# plot prediction error for ogd, adagrad, adam
ggplot(pred_df, aes(x=n)) +
  geom_line(aes(y=ogd), color="red") +
  geom_line(aes(y=adagrad), color="blue") +
  geom_line(aes(y=adam), linetype="dashed") +
  xlab("Iteration") +
  ylab("") +
  ggtitle("Prediction error for OGD (red), Adagrad (blue), Adam (black)")
```




```
# plot estimation error for ogd, adagrad, adam
ggplot(est_df, aes(x=n)) +
  geom_line(aes(y=ogd), color="red") +
  geom_line(aes(y=adagrad), color="blue") +
  geom_line(aes(y=adam), linetype="dashed") +
  xlab("Iteration") +
  ylab("") +
  ggtitle("Estimation error for OGD (red), Adagrad (blue), Adam (black)")
```



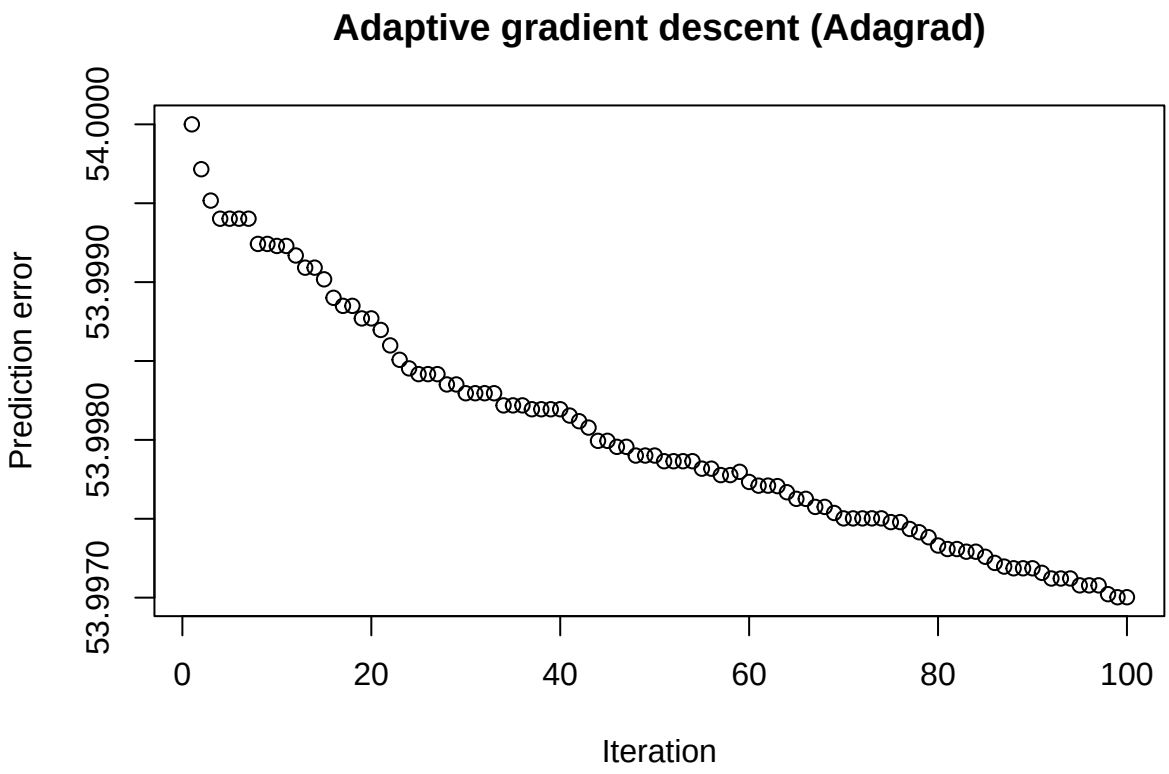

```
# plot runtime for ogd, adagrad, adam
ggplot(runtime_df, aes(x=n)) +
  geom_line(aes(y=ogd), color="red") +
  geom_line(aes(y=adagrad), color="blue") +
  geom_line(aes(y=adam), linetype="dashed") +
  xlab("Iteration") +
  ylab("") +
  ggtitle("Runtime for OGD (red), Adagrad (blue), Adam (black)")
```



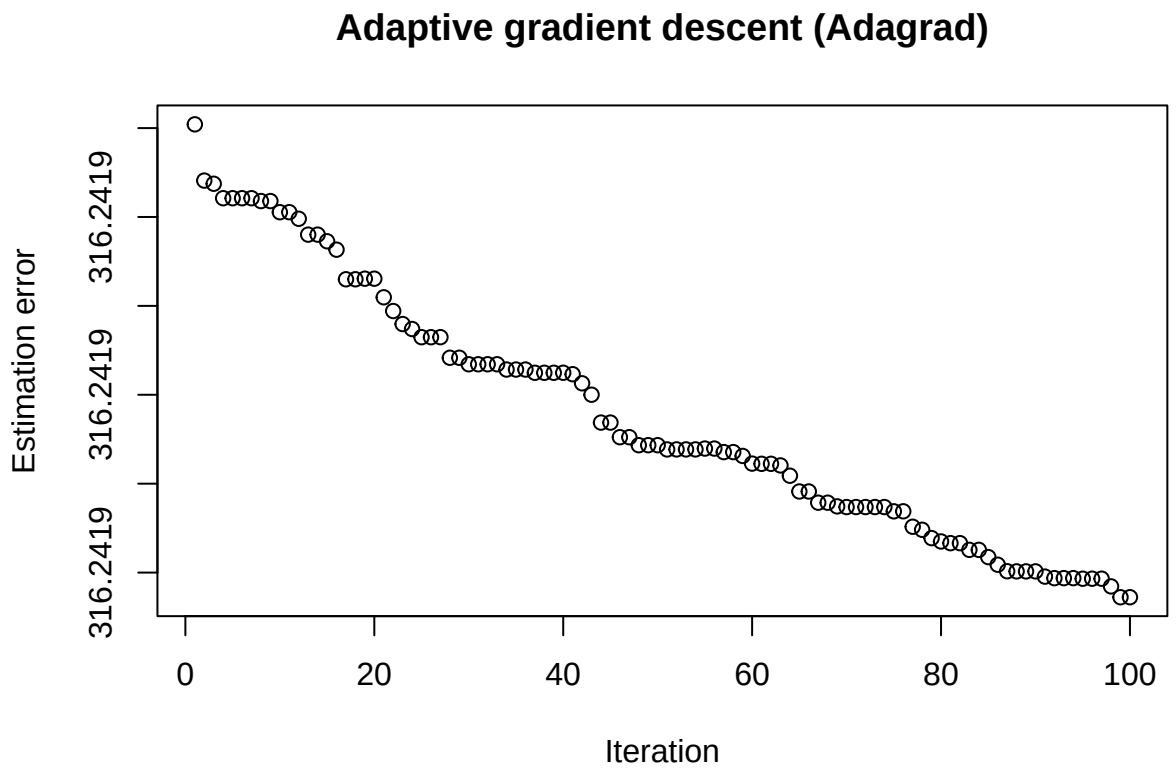

```
runtime_df = data.frame("n"=1:n)
pred_df = data.frame("n"=1:n) # prediction error
est_df = data.frame("n"=1:n) # estimation error

# comparing full and diagonal Adagrad
print("Diagonal")
```

```
## [1] "Diagonal"
```

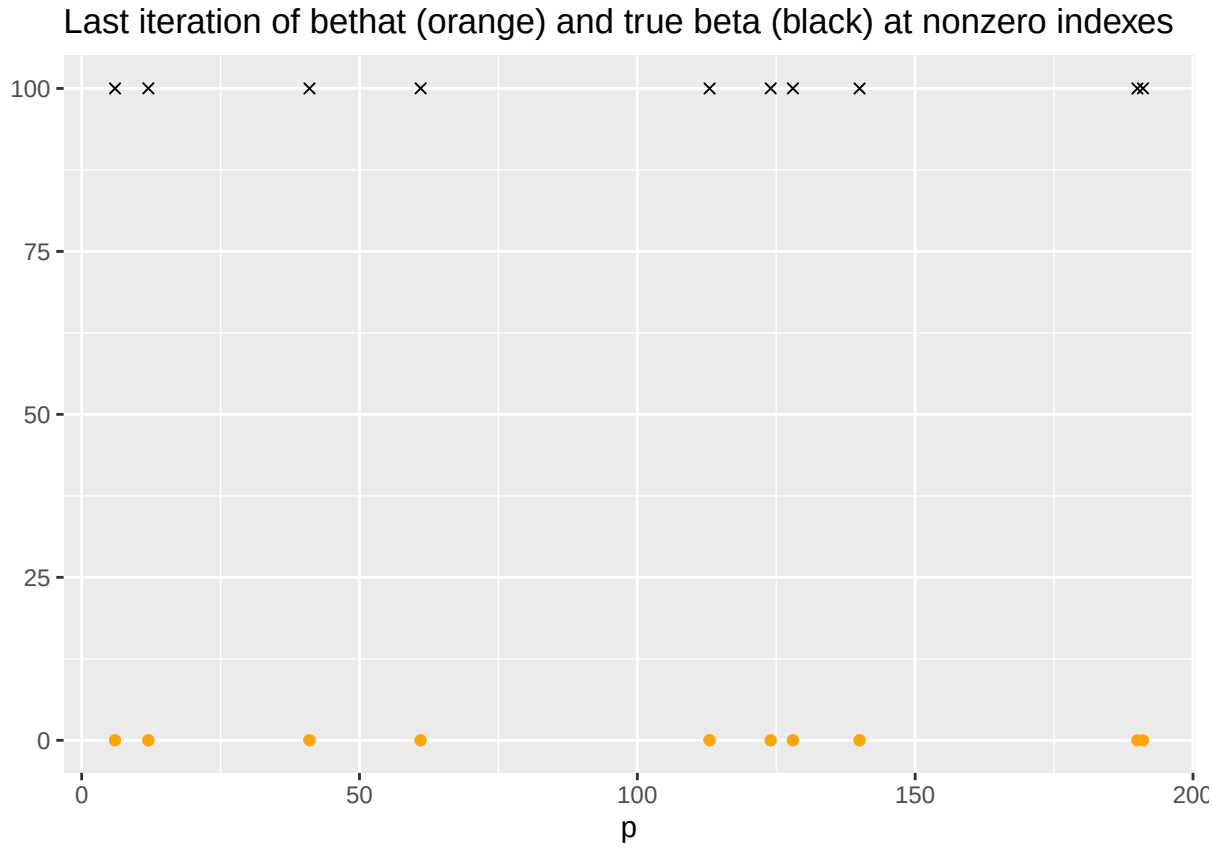
```
temp = my_adagrad(X=X, Y=Y, lr=0.0000001, beta_0=rep(0, p), full=F)
betahats = temp[[1]]
runtimes = temp[[2]]
runtime_df$diag = runtimes

pred_df$diag = plot_prediction_error(betahats, beta, X, Y, title="Adaptive gradient descent (Adagrad)",
```



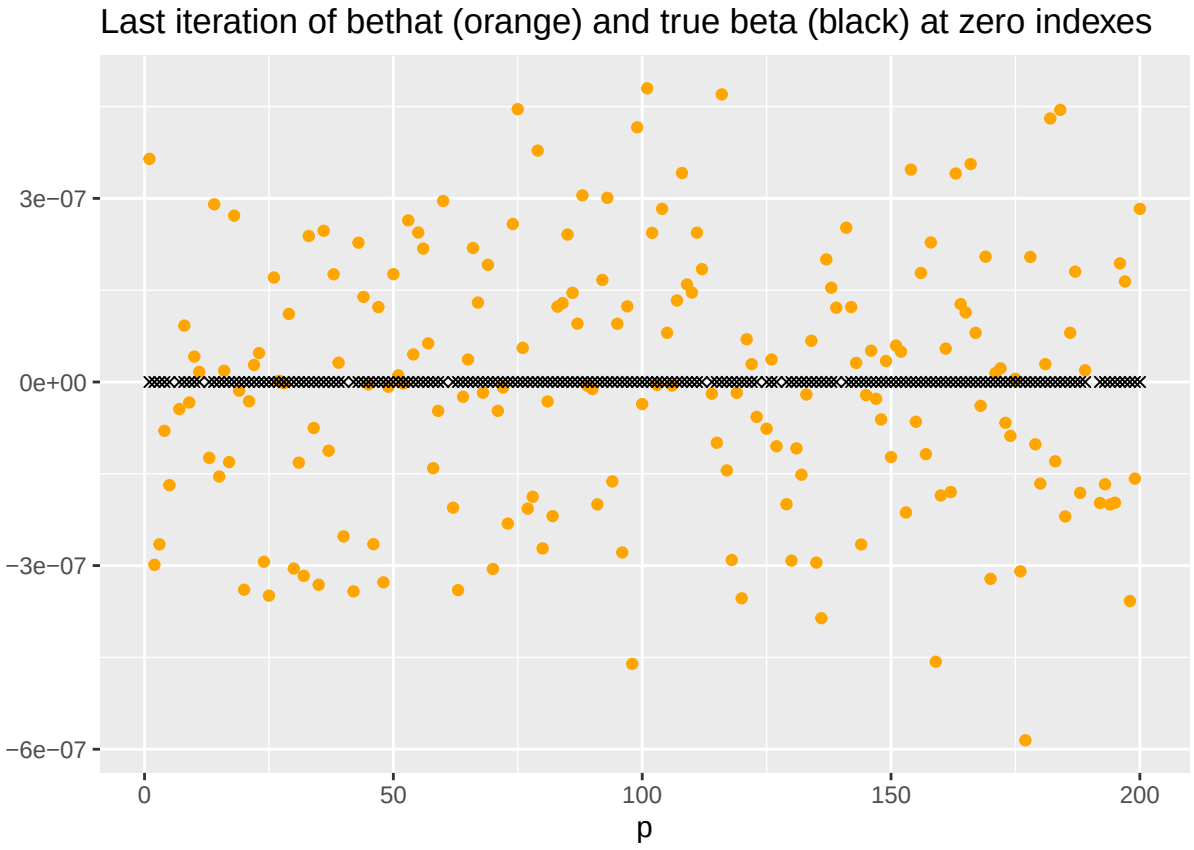

```
est_df$diag = plot_prediction_error(betahats, beta, X, Y, title="Adaptive gradient descent (Adagrad)", t
```




```
plot_betas(betahats, beta, nonzero_indexes, nonzero=T)
```




```
plot_betas(betahats, beta, nonzero_indexes, nonzero=F)
```




```
# print("Full")
# temp = my_adagrad(X=X, Y=Y, lr=0.0000001, beta_0=rep(0, p), full=T)
# betahats = temp[[1]]
# runtimes = temp[[2]]
# runtime_df$full = runtimes
#
# pred_df$full = plot_prediction_error(betahats, beta, X, Y, title="Adaptive gradient descent (Adagrad)
# est_df$full = plot_prediction_error(betahats, beta, X, Y, title="Adaptive gradient descent (Adagrad)"
#
# plot_betas(betahats, beta, nonzero_indexes, nonzero=T)
# plot_betas(betahats, beta, nonzero_indexes, nonzero=F)
```