PS8 Finley

Caden Finley

April 2023

1 Code

library(nloptr) set.seed(100)

Dimensions N ¡- 100000 K ¡- 10

X matrix with first column of 1's and remaining columns normally distributed X ¡- matrix(rnorm(N * K), ncol = K) X[,1] ¡- 1 Create epsilon vector with N random numbers distributed sigma ¡- 0.5 eps ¡- rnorm(N, mean = 0, sd = sigma)

beta vectors beta ¡- c(1.5, -1, -0.25, 0.75, 3.5, -2, 0.5, 1, 1.25, 2) Generate Y as X * beta + epsilon Y ¡- X

OLS estimate of beta beta$_{h}at < -solve(t(X)Comparewithrealbetavaluesbeta_{h}atbetaThebetasfrombetaand$

6/gradient descent

learning rate and max iterations learning$_{r}ate < -0.0000003alpha < -learning_{r}atemax_{i}ter < -10000$

gradient ¡- function(x) return(solve(t(X)initialize a value to x x ¡- floor(runif(1)*10) vector for all xs for all steps x.All ¡- vector("numeric",max$_{i}ter)gradientdescentmethodtofindtheminimumfor($ $max_{i}ter)x < -x - alpha * gradient(x)x.All[i] < -xprint(x)print(paste("minoff(x)is", x, sep = ""))$

7/nloptr

objfun ¡- function(beta, y, X) return (as.vector(-2*t(X) Gradient objective function gradient ¡- function(beta, y, X) return (as.vector(-2*t(X) initial values beta0 ¡- runif(dim(X)[2])

L-BFGS algorithm options ¡- list("algorithm"="NLOPT$_{L}D_{L}BFGS", "xtol_{r}el" = 1.0e-6, "maxeval" = 1e3)Optimizeresult < -nloptr(x0 = beta0, eval_{f} = objfun, y = Y, X = X, eval_{g}rad_{f} = gradient, opts = options)print(result)$

Nelder-Mead options1 ¡- list("algorithm"="NLOPT$_{L}N_{N}ELDERMEAD", "xtol_{r}el" = 1.0e-6, "maxeval" = 1e4)beta0 < -runif(dim(X)[2])result1 < -nloptr(x0 = beta0, eval_{f} = objfun, eval_{g}rad_{f} = gradient, opts = options1, Y = Y, X = X)print(result1)Resultshouldbethesame$

8 gradient ¡- function(theta,Y,X) grad ¡- as.vector(rep(0,length(theta))) beta ¡- theta[1:(length(theta)-1)] sig ¡- theta[length(theta)] grad[1:(length(theta)-1)] ¡- -t(X)$^{3})return(grad)$

result ¡- nloptr(x0 = theta0, eval$_{f} = objfun, eval_{g}rad_{f} = gradient, lb = c(rep(-Inf, K), 0), ub = c(rep(Inf, K), Inf), opts = opts, Y = Y, X = X)$

9 est ¡- lm(Y   X-1) library(modelsummary) modelsummary(est, output = ”simplereg.tex”)