SGDinference: An R Vignette

Introduction

**SGDinference** is an R package that provides estimation and inference methods for large-scale mean and quantile regression models via stochastic (sub-)gradient descent (S-subGD) algorithms. The inference procedure handles cross-sectional data sequentially:

(i) updating the parameter estimate with each incoming “new observation”,
(ii) aggregating it as a Polyak-Ruppert average, and
(iii) computing an asymptotically pivotal statistic for inference through random scaling.

The methodology used in the SGDinference package is described in detail in the following papers:


We begin by calling the SGDinference package.

```r
library(SGDinference)
set.seed(100723)
```

Case Study: Estimating the Mincer Equation

To illustrate the usefulness of the package, we use a small dataset included in the package. Specifically, the **Census2000** dataset from Acemoglu and Autor (2011) consists of observations on 26,120 nonwhite, female workers. This small dataset is constructed from “microwage2000_ext.dta” at https://economics.mit.edu/people/faculty/david-h-autor/data-archive. Observations are dropped if hourly wages are missing or years of education are smaller than 6. Then, a 5 percent random sample is drawn to make the dataset small. The following three variables are included:

- `ln_hrwage`: log hourly wages
- `edyrs`: years of education
- `exp`: years of potential experience

We now define the variables.

```r
y = Census2000$ln_hrwage
edu = Census2000$edyrs
exp = Census2000$exp
exp2 = exp^2/100
```

As a benchmark, we first estimate the Mincer equation and report the point estimates and their 95% heteroskedasticity-robust confidence intervals.

```r
mincer = lm(y ~ edu + exp + exp2)
inference = lmtest::coefci(mincer, df = Inf, vcov = sandwich::vcovHC)
results = cbind(mincer$coefficients, inference)
colnames(results)[1] = "estimate"
```
We now estimate the same model using SGD.

```r
mincer_sgd <- sgd_lm(y ~ edu + exp + exp2)
print(mincer_sgd)
```

It can be seen that the estimation results are similar between two methods. There is a different command that only computes the estimates but not confidence intervals.

```r
mincer_sgd <- sgd_lm(y ~ edu + exp + exp2)
print(mincer_sgd)
```

We compare the execution times between two versions and find that there is not much difference in this simple example. By construction, it takes more time to conduct inference via `sgdi_lm`.

```r
library(microbenchmark)
res <- microbenchmark(sgd_lm(y ~ edu + exp + exp2),
                       sgdi_lm(y ~ edu + exp + exp2),
                       times=100L)
print(res)
```

To plot the SGD path, we first construct a SGD path for the return to education coefficients.
Then, we can plot the SGD path.

```r
plot(mincer_sgd_path$path_coefficients, ylab="Return to Education", xlab="Steps", type="l")
```

To observe the initial paths, we now truncate the paths up to 2,000.

```r
plot(mincer_sgd_path$path_coefficients[1:2000], ylab="Return to Education", xlab="Steps", type="l")
```

```r
print(c("2000th step", mincer_sgd_path$path_coefficients[2000]))
#> [1] "2000th step" "0.121832196962998"
```
It can be seen that the SGD path almost converged only after the 2,000 steps, less than 10% of the sample size.

Estimating the Quantile Regression Model Using S-subGD

We now estimate a quantile regression version of the Mincer equation.

```r
mincer_sgd = sgdi_qr(y ~ edu + exp + exp2)
print(mincer_sgd)
```

The default quantile level is 0.5, as seen below.

```r
mincer_sgd = sgdi_qr(y ~ edu + exp + exp2)
print(mincer_sgd)
```

We now consider alternative quantile levels.

```r
mincer_sgd_median = sgdi_qr(y ~ edu + exp + exp2, qt=0.5)
print(mincer_sgd_median)
```
As before, we can plot the SGD path.

```r
mincer_sgd_path = sgdi_qr(y ~ edu + exp + exp2, path = TRUE, path_index = 2)
plot(mincer_sgd_path$path_coefficients[1:2000], ylab="Return to Education", xlab="Steps", type="l")
```
print(c("2000th step", mincer_sgd_path$path_coefficients[2000])))
#> [1] "2000th step" "0.144993066450143"
print(c("Final Estimate", mincer_sgd_path$coefficients[2]))
#> [1] "Final Estimate" "0.141593421862818"