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:

- Lee, S., Liao, Y., Seo, M.H. and Shin, Y., 2022. Fast and robust online inference with stochastic gradient descent via random scaling. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, No. 7, pp. 7381-7389). https://doi.org/10.1609/aaai.v36i7.20701.
- Lee, S., Liao, Y., Seo, M.H. and Shin, Y., 2023. Fast Inference for Quantile Regression with Tens of Millions of Observations. arXiv:2209.14502 [econ.EM] https://doi.org/10.48550/arXiv.2209.14502.

We begin by calling the SGDinference package.

```r
library(SGDinference)
```

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 contructed 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"
print(results)
```

```
#>                 estimate       2.5 %      97.5 %
#> (Intercept)  0.58114741  0.52705757  0.63523726
#> edu          0.12710477  0.12329983  0.13090971
#> exp          0.03108721  0.02877637  0.03339806
#> exp2        -0.04498841 -0.05070846 -0.03926835
```

Estimating the Mean Regression Model Using SGD

We now estimate the same model using SGD.

```
 mincer_sgd = sgdi_lm(y ~ edu + exp + exp2)
 print(mincer_sgd)
#> Call:
#> sgdi_lm(formula = y ~ edu + exp + exp2)
#> 
#> Coefficients:
#>              Coefficient    CI.Lower    CI.Upper
#> (Intercept)  0.58636390  0.51619856  0.65652924
#> edu          0.12657139  0.12286313  0.13027965
#> exp          0.03152237  0.02788735  0.03515739
#> exp2        -0.04599261 -0.05558258 -0.03640264
#> 
#> Significance Level: 95 %
```

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.

```
 mincer_sgd = sgd_lm(y ~ edu + exp + exp2)
 print(mincer_sgd)
#> Call:
#> sgd_lm(formula = y ~ edu + exp + exp2)
#> 
#> Coefficients:
#>              Coefficient
#> (Intercept)  0.58777036
#> edu          0.12646982
#> exp          0.03151513
#> exp2        -0.04600389
```

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`.

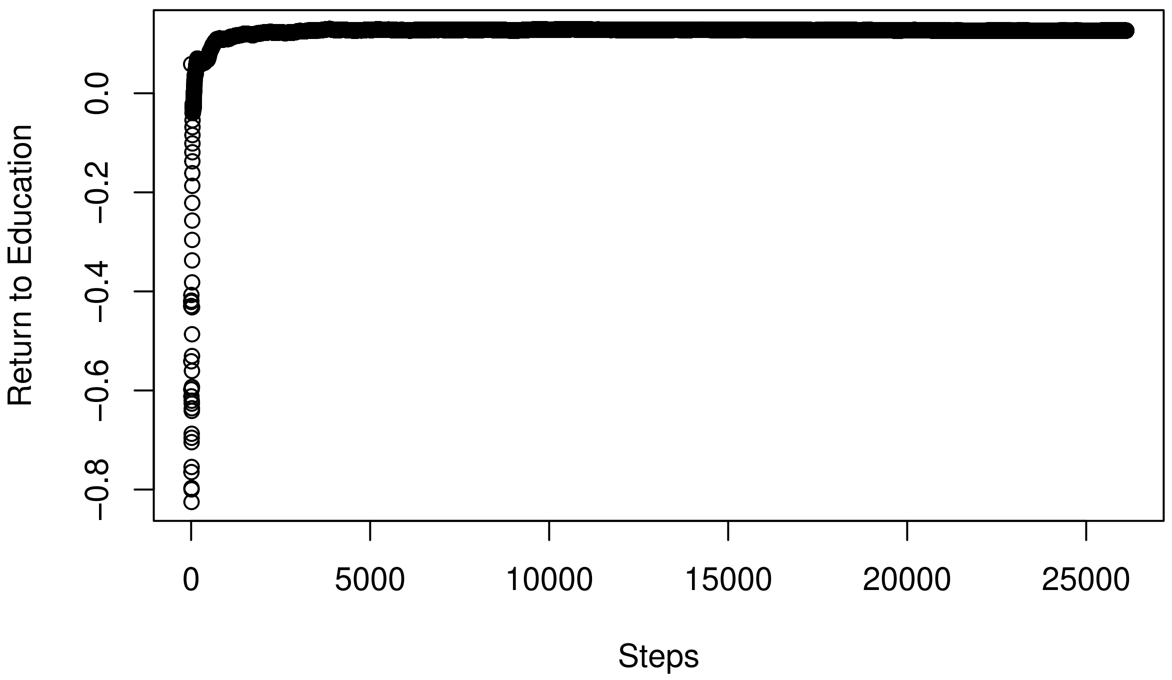
```
library(microbenchmark)
res <- microbenchmark(sgd_lm(y ~ edu + exp + exp2),
                      sgdi_lm(y ~ edu + exp + exp2),
                      times=100L)
print(res)
#> Unit: milliseconds
#>                            expr      min       lq     mean   median       uq      max neval
#>    sgd_lm(y ~ edu + exp + exp2) 3.487460 3.787149 4.681248 3.891556 4.199528 11.51559   100
#>   sgdi_lm(y ~ edu + exp + exp2) 4.422342 4.593722 6.570060 4.789599 8.510083 69.55187   100
```

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

```
mincer_sgd_path = sgdi_lm(y ~ edu + exp + exp2, path = TRUE, path_index = 2)
```

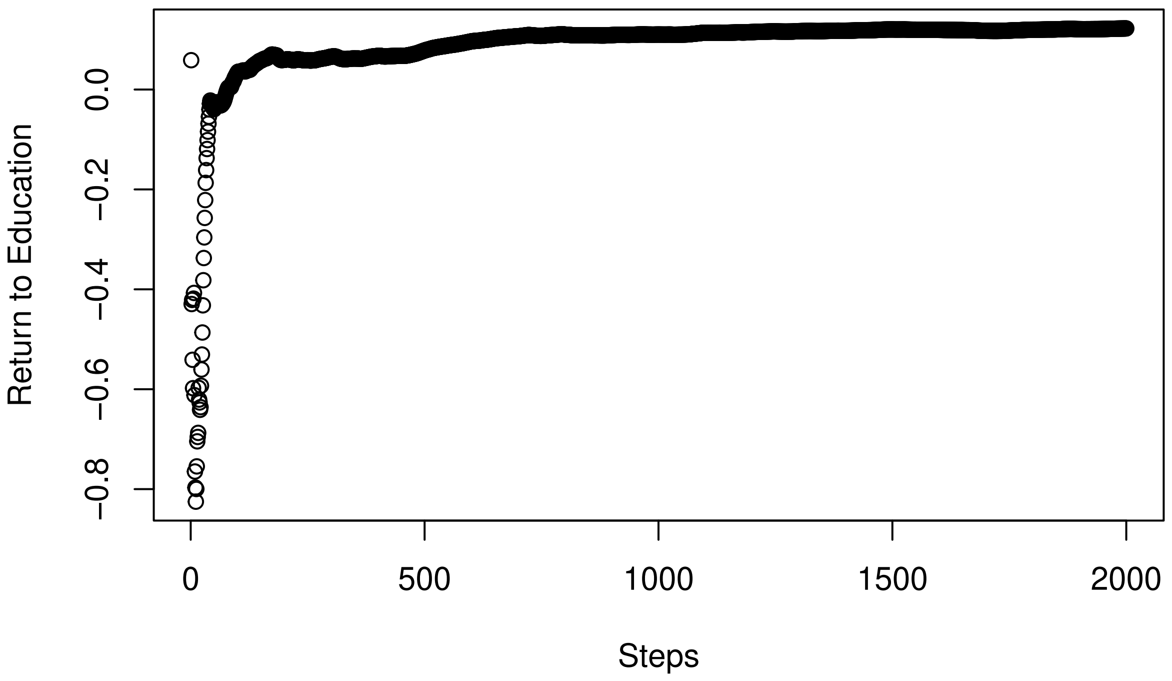
Then, we can plot the SGD path.

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



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")
```



```r
print(c("2000th step", mincer_sgd_path$path_coefficients[2000]))
#> [1] "2000th step"       "0.122324374 73292"
print(c("Final Estimate", mincer_sgd_path$coefficients[2]))
#> [1] "Final Estimate"     "0.126519441731705"
```

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.

```
mincer_sgd = sgdi_qr(y ~ edu + exp + exp2)
print(mincer_sgd)
#> Call:
#> sgdi_qr(formula = y ~ edu + exp + exp2)
#>
#> Coefficients:
#>              Coefficient    CI.Lower    CI.Upper
#> (Intercept)  0.39673700  0.35688283  0.43659117
#> edu          0.14085754  0.13808663  0.14362844
#> exp          0.03068952  0.02859980  0.03277924
#> exp2        -0.04441670 -0.05002659 -0.03880681
#>
#> Significance Level: 95 %
```

The default quantile level is 0.5, as seen below.

```
mincer_sgd = sgdi_qr(y ~ edu + exp + exp2)
print(mincer_sgd)
#> Call:
#> sgdi_qr(formula = y ~ edu + exp + exp2)
#>
#> Coefficients:
#>              Coefficient    CI.Lower    CI.Upper
#> (Intercept)  0.38699481  0.34079300  0.43319662
#> edu          0.14148189  0.13842986  0.14453392
#> exp          0.03092141  0.02910978  0.03273304
#> exp2        -0.04474268 -0.04977409 -0.03971127
#>
#> Significance Level: 95 %
mincer_sgd_median = sgdi_qr(y ~ edu + exp + exp2, qt=0.5)
print(mincer_sgd_median)
#> Call:
#> sgdi_qr(formula = y ~ edu + exp + exp2, qt = 0.5)
#>
#> Coefficients:
#>              Coefficient    CI.Lower    CI.Upper
#> (Intercept)  0.39178927  0.35236005  0.43121849
#> edu          0.14066766  0.13793570  0.14339963
#> exp          0.03162926  0.02975408  0.03350443
#> exp2        -0.04643939 -0.05316247 -0.03971631
#>
#> Significance Level: 95 %
```

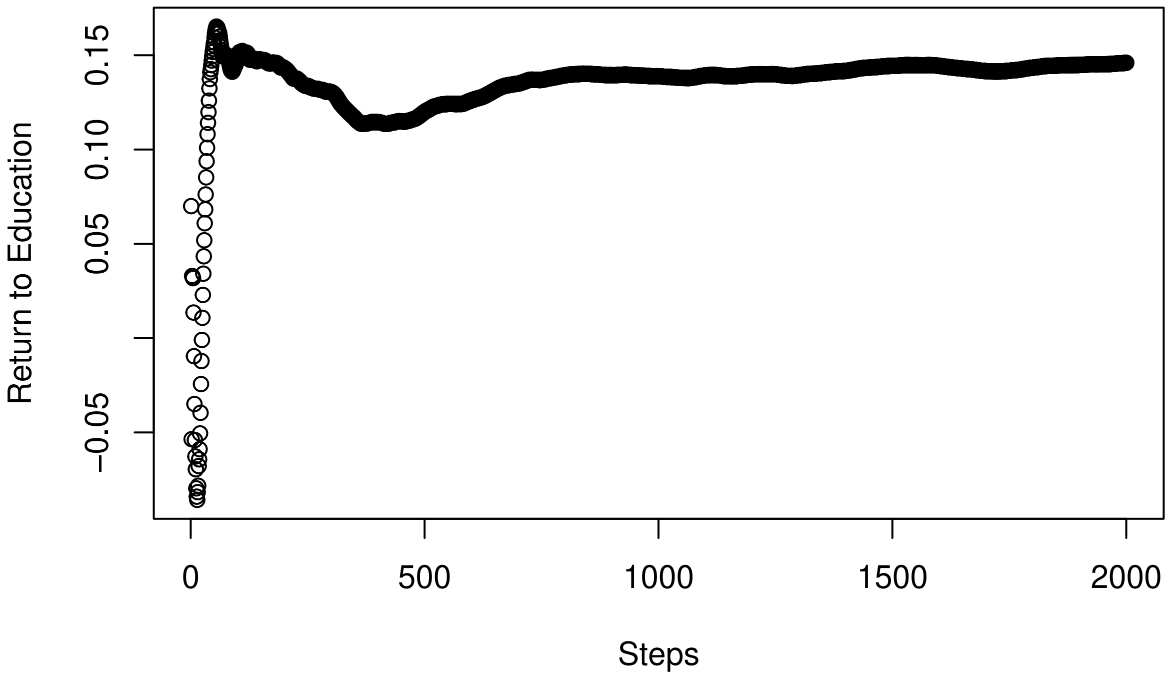
We now consider alternative quantile levels.

```
mincer_sgd_p10 = sgdi_qr(y ~ edu + exp + exp2, qt=0.1)
print(mincer_sgd_p10)
#> Call:
#> sgdi_qr(formula = y ~ edu + exp + exp2, qt = 0.1)
```

```
#>
#> Coefficients:
#>              Coefficient    CI.Lower    CI.Upper
#> (Intercept)  -0.2280177 -0.28995868 -0.16607671
#> edu           0.1286538  0.12278920  0.13451831
#> exp           0.0336624  0.02574478  0.04158003
#> exp2         -0.0544648 -0.07188785 -0.03704175
#>
#> Significance Level: 95 %
mincer_sgd_p90 = sgdi_qr(y ~ edu + exp + exp2, qt=0.9)
print(mincer_sgd_p90)
#> Call:
#> sgdi_qr(formula = y ~ edu + exp + exp2, qt = 0.9)
#>
#> Coefficients:
#>               Coefficient    CI.Lower    CI.Upper
#> (Intercept)  1.551897847  1.42033470 1.683460999
#> edu          0.114558071  0.10381224 0.125303905
#> exp          0.017722737  0.01409223 0.021353249
#> exp2        -0.008718438 -0.01880192 0.001365042
#>
#> Significance Level: 95 %
```

As before, we can plot the SGD path.

```
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")
```



```
print(c("2000th step", mincer_sgd_path$path_coefficients[2000]))
#> [1] "2000th step"       "0.145985907046015"
print(c("Final Estimate", mincer_sgd_path$coefficients[2]))
#> [1] "Final Estimate"    "0.141429120655089"
```