Weighted Bootstrap + Subsample SEs

Brice Green

2021-03-28

I was working on the `quantspace` package, and trying to implement a prediction method that included the bootstrapped samples, when I noticed that the subsample standard errors were linearly declining in sample size.[1]

```
# attach packages
library(quantspace)
library(ggplot2)
library(magrittr)
library(data.table)
library(stringr)

# data generating process
x = rnorm(500)*5
y = 0.2 + 0.8 * x + rnorm(500, sd = 3)
qplot(x, y)
```

```
# function for generating fits over range of subsample percentages
fitSubsamplePct <- function(ssPct, form, draw_weights = F, num_bs = 1000){
   qs(form, se_method = "boot", draw_weights = draw_weights,
      subsamplePct = ssPct, num_bs = num_bs)
}

ss_pcts <- seq(0.1, 0.9, by = 0.1)
# fit a bunch of stuff for a variety of subsampling percentages
fit_with_diff_ss_pcts <- lapply(ss_pcts, fitSubsamplePct, form = y ~ x)

# rename so that we can keep track of things
names(fit_with_diff_ss_pcts) <- ss_pcts

# process them for plotting purposes
plot_data <- process_fitted_models(fit_with_diff_ss_pcts)

# plot scale standard errors
ggplot(plot_data, aes(x = SubsamplingPercent, y = SE)) +
  geom_line() +
  xlab("Subsample Percent") +
  ylab("Standard Error") +
  ggtitle("Standard errors for different subsample percentages") +
  facet_wrap(~coef, scales = 'free_y')
```

[1] You can install all packages from CRAN except quantspace, which you can use if you run `devtools::install_github("be-green/quantspace")` after installing the devtools package.

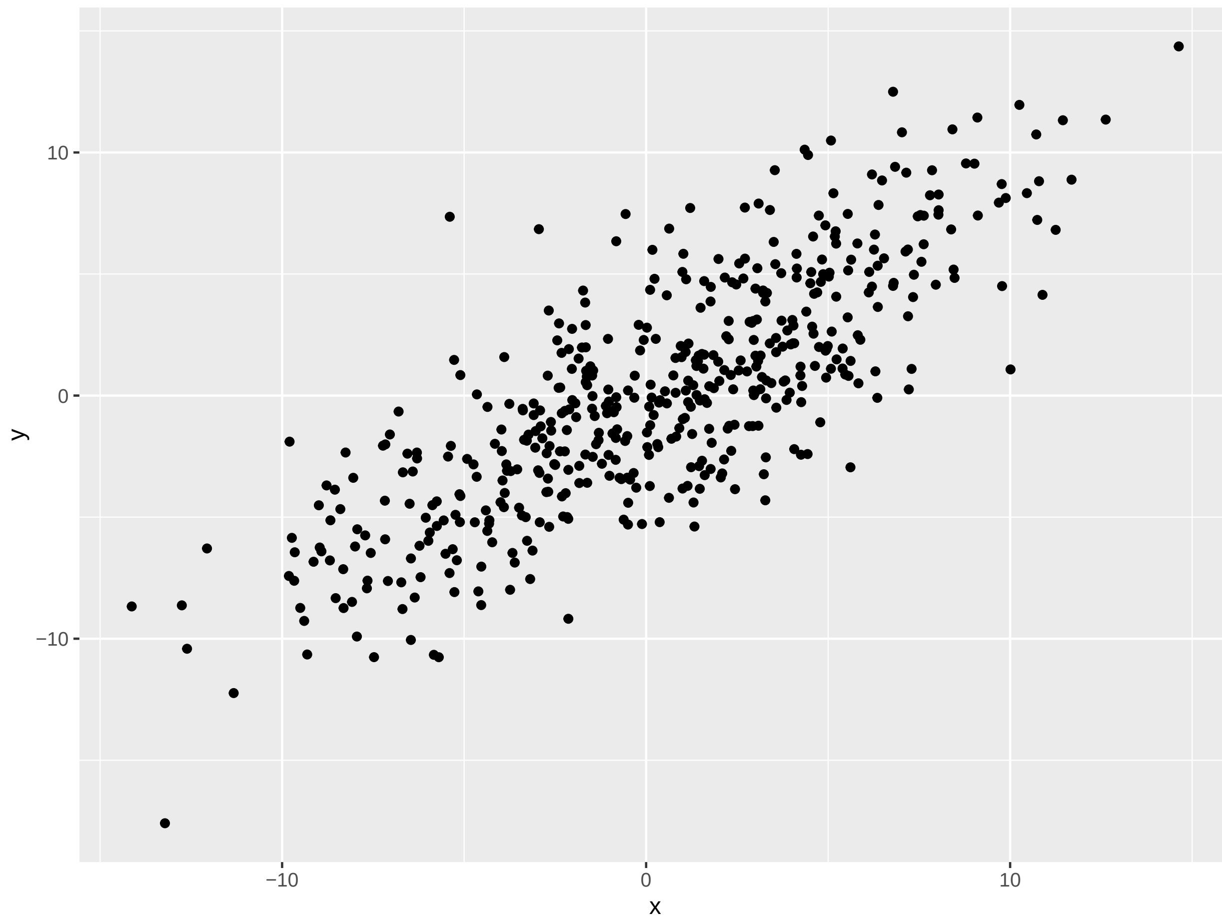

Figure 1: DGP Used for Testing

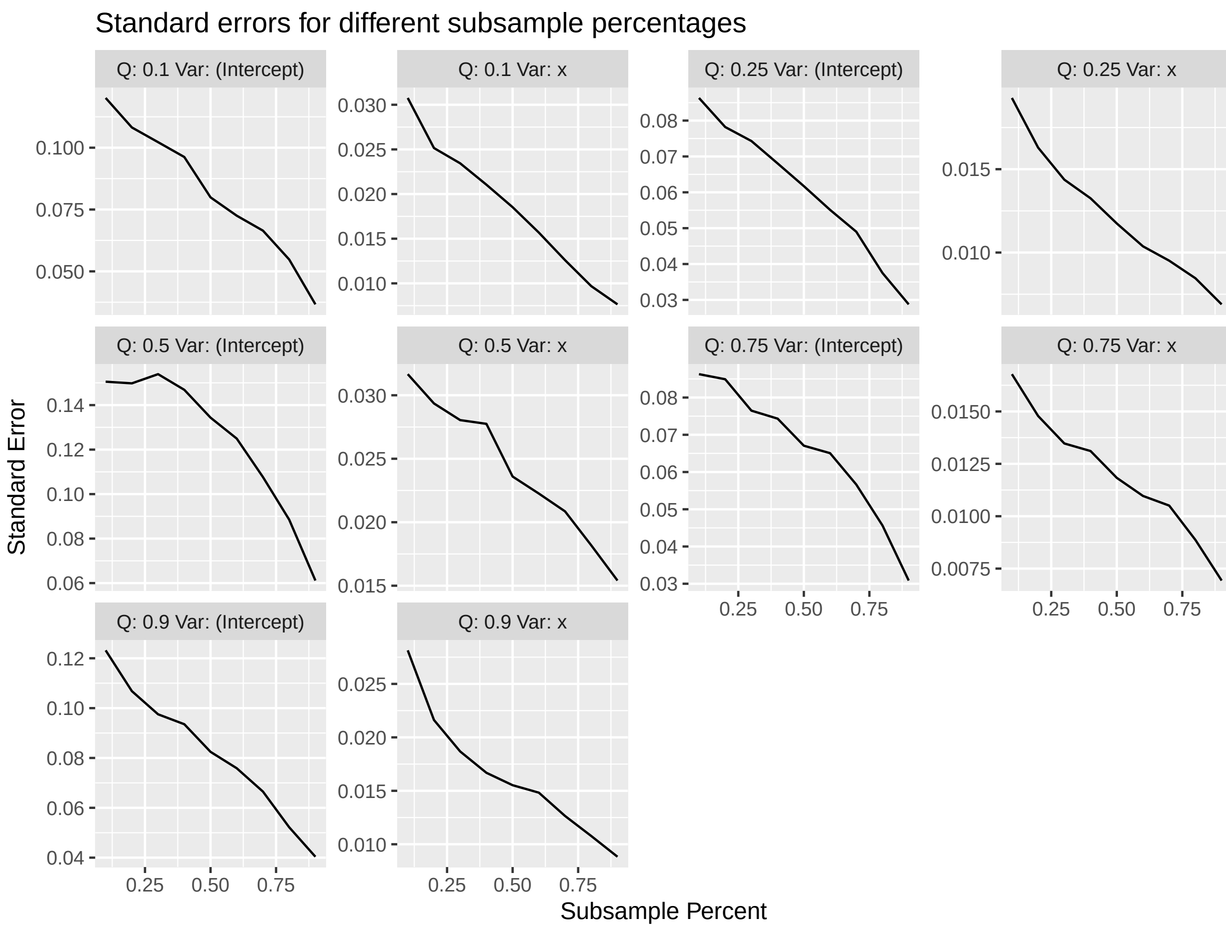

Figure 2: Subsampled Standard Errors

Right now this is how subsampling is scaled in the function `subsampleStandardErrors` which is doing most of the lifting, inside the `qs` function. `M` is the sub-sampling percentage.

```
quant_cov_mat <- stats::cov(fit$coef, use = "pairwise.complete.obs")
quant_cov_mat <- quant_cov_mat * (M)
```

This covariance matrix is passed to the `summary.qs` function, which calculates standard errors like so:

```
se = sqrt(diag(quant_cov_mat))
```

Which is equivalent to what I've plotted above–the standard error is scaled by the square root of the subsampling percentage. But even after this adjustment, the appropriately scaled standard errors are linearly decreasing in the subsampling percentage, even for a large number of bootstrap samples.

It's worth noting that the adjustment does improve things relative to the unadjusted standard errors. Here's the comparison against the raw covariances.

```
# plot scale standard errors
ggplot(plot_data, aes(x = SubsamplingPercent, y = SE)) +
  geom_line() +
  # unscale the SE
  geom_line(aes(y = SE/sqrt(SubsamplingPercent)), color = "red") +
  xlab("Subsample Percent") +
  ylab("Standard Error") +
  ggtitle("Standard errors for different subsample percentages",
          subtitle ="Scaled (black) vs Unscaled (red)") +
  facet_wrap(~coef, scales = 'free_y')
```

Larry thought this was because the weights were not being drawn. But the weighted bootstrap standard errors seem to still have this issue.

```
# fit a bunch of stuff for a variety of subsampling percentages
fits_with_weights <- lapply(ss_pcts, fitSubsamplePct, form = y ~ x, draw_weights = T)
names(fits_with_weights) <- ss_pcts
plot_data_with_weights <- process_fitted_models(fits_with_weights)

ggplot(plot_data_with_weights, aes(x = SubsamplingPercent, y = SE)) +
  geom_line() +
  xlab("Subsample Percent") +
  ylab("Standard Error") +
  ggtitle("Standard errors for different subsample percentages",
          subtitle = "Using random exponential weights") +
  facet_wrap(~coef, scales = 'free_y')
```

Here it looks like the medians fair slightly better, but the others are still linearly decreasing in in the subsampling percentage.

```
comparison <- merge(plot_data, plot_data_with_weights,
                    by = c("coef", "SubsamplingPercent"),
              suffix = c("_subsampled","_weighted"))

ggplot(comparison, aes(x = SubsamplingPercent, y = SE_subsampled)) +
```

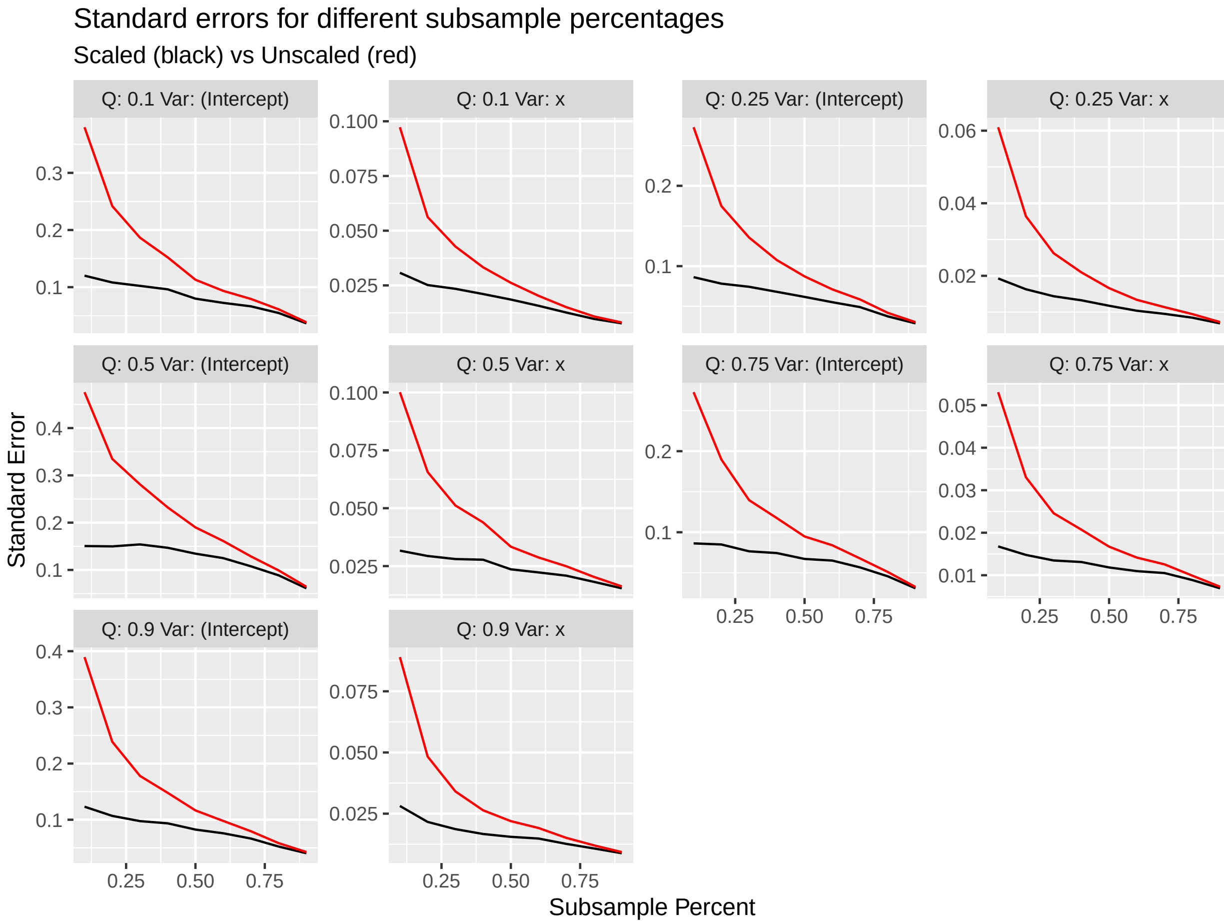

Figure 3: Subsampled Standard Errors, scaled vs. unscaled

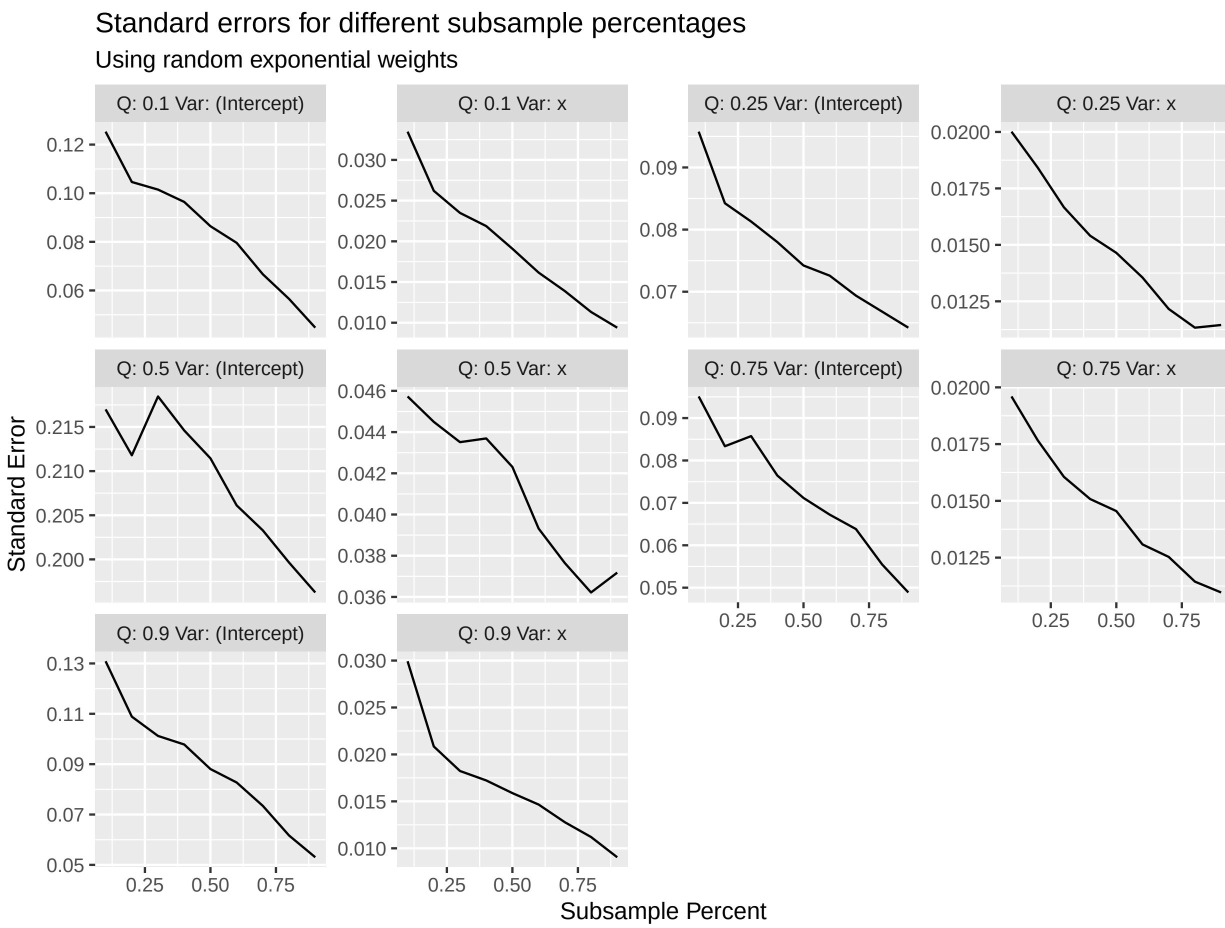

Figure 4: Weighted Bootstrap Standard Errors

```
  geom_line() +
  geom_line(aes(y = SE_weighted), color = "red") +
  facet_wrap(~coef, scales = "free") +
  ylab("Standard Error")+
  xlab("Subsampling Percentage") +
  ggtitle("Comparison of both standard errors, side by side",
          subtitle = paste("Subsampled (black) vs. Subsampled",
                           "w/ Random Exponential Weights (red)"))
```

The behavior seems worse in the tails and better in the middle, where the random exponential weights seem to flatten the standard errors considerably. Quantiles 0.1 and 0.9 are especially problematic. Larry thought the issue might be numerical precision type errors, but to have a standard error move from 0.03 to 0.01 or 0.12 to 0.04 seems like a pretty big jump to be a function of precision type mistakes.

```
n_tests = 500
tests <- list()

for(i in 1:n_tests) {
  # same dgp and N, different draws every time
  x = rnorm(500)*5
  y = 0.2 + 0.8 * x + rnorm(500, sd = 3)
  # use 20% of data in subsampling
  fit_with_subsampling <- qs(y ~ x, draw_weights = F,
```

```
                             subsamplePct = 0.2, num_bs = 100)

  # use 40% of data in subsampling, and random exponential weights
  fit_with_both <- qs(y ~ x, draw_weights = T,
                      subsamplePct = 0.4, num_bs = 100)

  # use all of the data, with random weights
  fit_with_weights <- qs(y ~ x, draw_weights = T,
                         subsamplePct = 1, num_bs = 100)

  # store the values
  tests[[i]] <- list(
    just_ss = fit_with_subsampling,
    both = fit_with_both,
    just_weights = fit_with_weights
  )

}
```