Presentation of Exercise 3.A.1.

Mads Adrian Simonsen, William Scott Grundeland Olsen

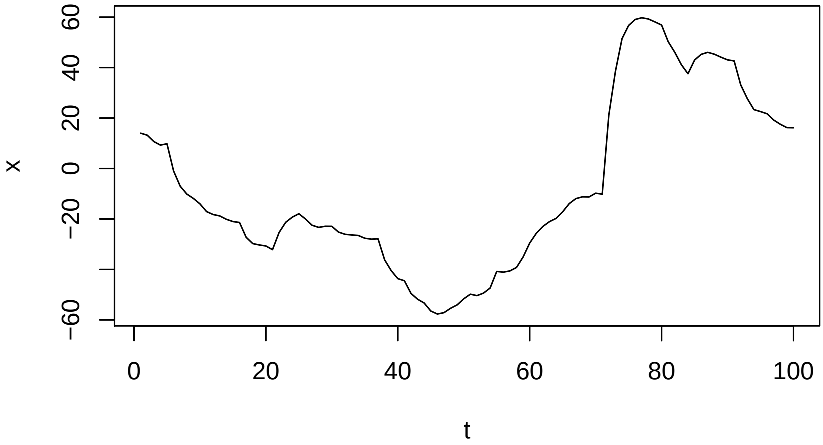
22 april, 2021

The Time Series and the Model

Assume an AR(2) model

$$x_t = \beta_1 x_{t-1} + \beta_2 x_{t-2} + e_t,$$

where $e_t \overset{\text{i.i.d.}}{\sim} (0, \sigma^2)$, for $t = 3, \ldots, T$.



The Problem

Use the residual resampling bootstrap method to evaluate the relative performance of the two parameter estimators $\hat{\beta}_{\mathrm{LS}}$ and $\hat{\beta}_{\mathrm{LA}}$, arising from minimizing

$$Q_{\mathrm{LS}}(\mathbf{x}) = \sum_{t=3}^{T}(x_t - \beta_1 x_{t-1} - \beta_2 x_{t-2})^2,$$

and

$$Q_{\mathrm{LA}}(\mathbf{x}) = \sum_{t=3}^{T}|x_t - \beta_1 x_{t-1} - \beta_2 x_{t-2}|,$$

respectively. Specifically, estimate the variance and bias of the two estimators. Is the LS estimator optimal for the AR(2) process?

Bootstrapping

- Use `ARp.beta.est` to calculate the estimated betas.

Bootstrapping

- Use `ARp.beta.est` to calculate the estimated betas.
- Use `ARp.resid` to find the corresponding residuals.

Bootstrapping

- Use `ARp.beta.est` to calculate the estimated betas.
- Use `ARp.resid` to find the corresponding residuals.
- Initialize matrices to store the bootstrap samples.

Bootstrapping

- Use `ARp.beta.est` to calculate the estimated betas.
- Use `ARp.resid` to find the corresponding residuals.
- Initialize matrices to store the bootstrap samples.
- Run the bootstrapping B times.

Bootstrapping

- Use `ARp.beta.est` to calculate the estimated betas.
- Use `ARp.resid` to find the corresponding residuals.
- Initialize matrices to store the bootstrap samples.
- Run the bootstrapping B times.
  - Sample from the calculated residuals with replacement.

Bootstrapping

- Use `ARp.beta.est` to calculate the estimated betas.
- Use `ARp.resid` to find the corresponding residuals.
- Initialize matrices to store the bootstrap samples.
- Run the bootstrapping B times.
  - Sample from the calculated residuals with replacement.
  - Calculate the AR(2) sequence where the initial values are chosen at random, using `ARp.filter`.

Bootstrapping

- Use `ARp.beta.est` to calculate the estimated betas.
- Use `ARp.resid` to find the corresponding residuals.
- Initialize matrices to store the bootstrap samples.
- Run the bootstrapping B times.
  - Sample from the calculated residuals with replacement.
  - Calculate the AR(2) sequence where the initial values are chosen at random, using `ARp.filter`.
  - Using again `ARp.beta.est`, calculate the estimated betas, and add them to the previously mentioned matrix.

Bootstrapping

- Use `ARp.beta.est` to calculate the estimated betas.
- Use `ARp.resid` to find the corresponding residuals.
- Initialize matrices to store the bootstrap samples.
- Run the bootstrapping B times.
  - Sample from the calculated residuals with replacement.
  - Calculate the AR(2) sequence where the initial values are chosen at random, using `ARp.filter`.
  - Using again `ARp.beta.est`, calculate the estimated betas, and add them to the previously mentioned matrix.
- Do inference on the betas obtained.

Results

Using the package `matrixStats` we can find the row-wise variance, and bias.

For the LS estimator

The variance is

```
## [1] 0.005492141 0.005369391
```

The bias is

```
## [1] -0.012957274  0.007564968
```

For the LA estimator

The variance is

```
## [1] 0.0003934763 0.0003875681
```

The bias is

```
## [1] -0.002130976  0.001541546
```

Conclusion

The estimated variance of $\hat{\beta}_{\mathrm{LA}}$ is smaller than that of $\hat{\beta}_{\mathrm{LS}}$, and this is also true for the estimated bias. This suggests that the LS estimator is not optimal for the AR(2) process.

Difficulties

- The function `filter()` in the help files had a name collision when using the `tidyverse` package. We therefore had to use `stats::filter()`.

Appendix – Code Part 1

The initialization:

```
library(matrixStats)

# Estimates for the betas:
betas <- ARp.beta.est(x, 2)
# Corresponding residuals:
res_LS <- ARp.resid(x, betas$LS)
res_LA <- ARp.resid(x, betas$LA)

# Bootstrap:
B <- 2000    # Number of bootstrap samples
n <- length(res_LS)    # The size of resampling
# Initialize to store samples:
boot_beta_LS <- matrix(NA, nrow = 2, ncol = B)
boot_beta_LA <- matrix(NA, nrow = 2, ncol = B)
```

Appendix – Code Part 2

The bootstrap run:

```
for(b in 1:B) {
  # Sample the residuals:
  sample_LS <- sample(res_LS, n, replace = TRUE)
  sample_LA <- sample(res_LA, n, replace = TRUE)
  # Calculate AR(2) sequence:
  # rep(., 2) replicates value from sample to a 2-vector
  x_LS <- ARp.filter(x[rep(sample(TT-1, 1), 2)+c(0, 1)],
                     betas$LS, sample_LS)
  x_LA <- ARp.filter(x[rep(sample(TT-1, 1), 2)+c(0, 1)],
                     betas$LA, sample_LA)
  # Append betas to the bootstrap matrices:
  boot_beta_LS[, b] <- ARp.beta.est(x_LS, 2)$LS
  boot_beta_LA[, b] <- ARp.beta.est(x_LA, 2)$LA
}
```