3 Bootstrapping Dependent Data

Suppose we have dependent data $\boldsymbol{y} = (y_1, \ldots, y_n)$ generated from some unknown distribution $\boldsymbol{F} = \boldsymbol{F_Y} = F_{(Y_1,\ldots,Y_n)}$.

$Y_1, \ldots, Y_n$ no longer independent

could be time series or spatial for example

Goal:

To approximate dsn of a statistic

$$\theta = T(\underline{y}).$$

Challenge:

Since $Y_1, \ldots, Y_n$ dependent it is inappropriate to use the iid bootstrap.

Bootstrapped samples would no longer reproduce data generating process.

We will consider 2 approaches

1. model-based approach
2. block bootstrap (2 types).
   ↳ nonparametric.

3.1 Model-based approach

Example 3.1 Suppose we observe a time series $\boldsymbol{Y} = (Y_1, \ldots, Y_n)$ which we assume is generated by an AR(1) process, i.e., "Auto-regressive"

$$Y_t = \alpha Y_{t-1} + \varepsilon_t \qquad t = 1, \ldots, n$$

turn our problem into an iid bootstrap!

$$|\alpha| < 1 \quad \text{and} \quad \varepsilon_1, \ldots, \varepsilon_n \overset{iid}{\sim} (0, \sigma^2)$$

mean 0 variance $= \sigma^2$

If we assume an AR(1) model for the data, we can consider a method similar to bootstrapping residuals for linear regression.

(1) Estimate $\hat{\alpha}$ from original data (fit the AR(1) model).

observed data

(2) Define estimate innovations $\hat{\varepsilon}_t = Y_t - \hat{\alpha} Y_{t-1}$, $t = 2, \ldots, n$.

and their mean $\bar{\hat{\varepsilon}} = \frac{1}{n-1} \sum_{t=2}^{n} \hat{\varepsilon}_t$

(3) Define the residuals (assumed iid.) of the model as centered innovations.

$$e_t = \hat{\varepsilon}_t - \bar{\hat{\varepsilon}}$$

(4) For $b = 1, \ldots, B$,

a) create the bootstrap sample $e_0^*, \ldots, e_n^*$ by independently sampling w/ replacement $n+1$ values from the $n-1$ values e_t, $t = 2, \ldots, n$.

b) construct bootstrap data $Y^* = (Y_1^*, \ldots, Y_n^*)$ from

$$Y_0^* = e_0^*, \quad Y_t^* = \hat{\alpha} Y_{t-1}^* + e_t^*, \quad t = 1, \ldots, n.$$

c) get estimate $\hat{\alpha}^*$ from fitting AR(1) model on $(Y_1^*, \ldots, Y_n^*)$.

(5) The dsn of $\hat{\alpha}^{*(1)}, \ldots, \hat{\alpha}^{*(B)}$ is used as the sampling dsn of $\hat{\alpha}$.

Model-based – the performance of this approach depends on the model being appropriate for the data.

This may not always be a good assumption.

end of course material

3.2 Nonparametric approach

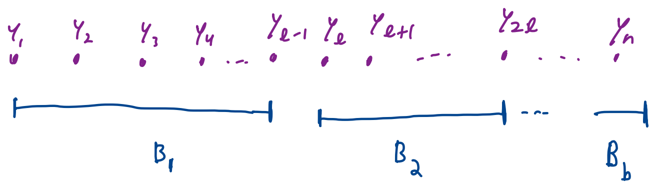
To deal with dependence in the data, we will employ a nonparametric *block* bootstrap.

Idea:

resample data in blocks to preserve the dependence structure within the blocks.

3.2.1 Nonoverlapping Blocks (NBB)

Consider splitting $\boldsymbol{Y} = (Y_1, \dots, Y_n)$ in b consecutive blocks of length ℓ.



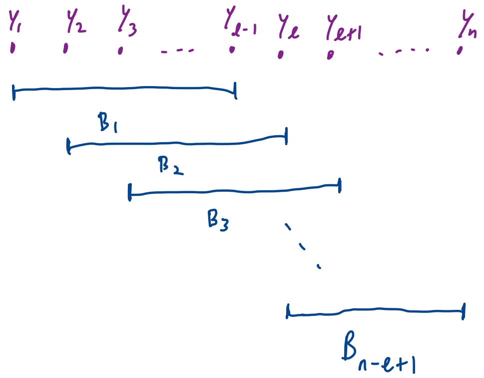
We can then rewrite the data as $\boldsymbol{Y} = (\boldsymbol{B}_1, \dots, \boldsymbol{B}_b)$ with $\boldsymbol{B}_k = (Y_{(k-1)\ell+1}, \dots, Y_{k\ell})$, $k = 1, \dots, b.$, $b = \lfloor \frac{n}{\ell} \rfloor$ ✓ "floor function" = round down

1. Sample nonoverlapping blocks $B_1^*, \dots, B_b^*$ independently from $(B_1, \dots, B_b)$ w/ replacement to form the bootstrap dataset $Y^* = (B_1^*, \dots, B_b^*)$
2. Statistic of interest θ is estimated from Y^* to create $\hat{\theta}^*$.
3. 1~2 repeated R times to obtain $\hat{\theta}^{*(1)}, \dots, \hat{\theta}^{*(R)}$ with which to estimate the dsn of $\hat{\theta}$.

Note, the order of data within the blocks must be maintained, but the order of the blocks that are resampled does not matter.

3.2.2 Moving Blocks (MBB)

Now consider splitting $\boldsymbol{Y} = (Y_1, \ldots, Y_n)$ into overlapping blocks of adjacent data points of length ℓ.



now we have more blocks to choose from!

We can then write the blocks as $\boldsymbol{B}_k = (Y_k, \ldots, Y_{k+\ell-1})$, $k = 1, \ldots, n-\ell+1$.

(1) Create a bootstrap dataset by sampling $Y^* = (B_1^*, \ldots, B_b^*)$ independently w/ replacement from $B_1, \ldots, B_{n-\ell+1}$

still $b = \lfloor \frac{n}{\ell} \rfloor$ the blocks

(2) Calculate $\hat{\theta}^*$ from Y^*

(3) Repeat 1–2 R times to obtain $\hat{\theta}^{*(1)}, \ldots, \hat{\theta}^{*(R)}$

3.2.3 Choosing Block Size

If the block length is too short,

The resampling cannot capture the dependence ($\ell = 1$ is the iid bootstrap!)

If the block length is too long,

Not many blocks to sample (does not resemble data generation)

Asymptotic result: block length should increase w/ length of time series.

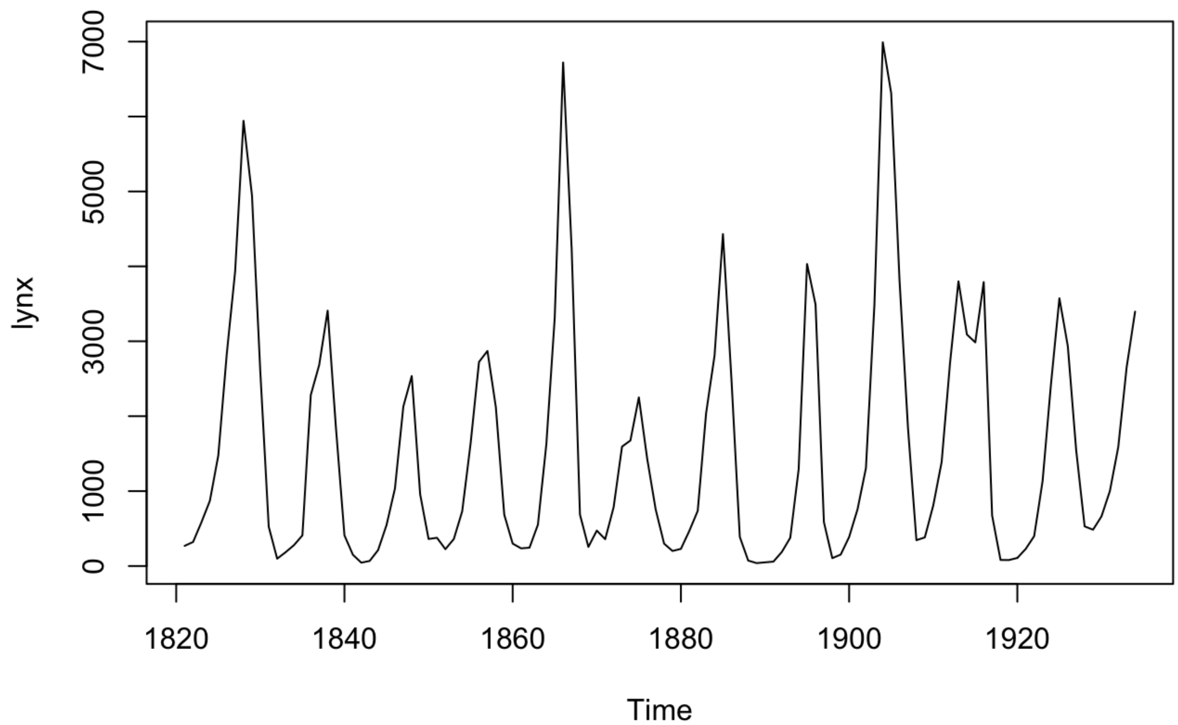
If so, MBB & NBB produce consistent estimators of moments, correct coverage probabilities for CIs and correct error rates for tests.

There are practical methods for choosing ℓ (Lahiri, 2003)

Your Turn

We will look at the annual numbers of lynx trappings for 1821–1934 in Canada. Taken from Brockwell & Davis (1991).

```
data(lynx)
plot(lynx)
```



Goal: Estimate the sample distribution of the mean $\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n} y_i$

```
theta_hat <- mean(lynx)
theta_hat
```

```
## [1] 1538.018
```

3.2.4 Independent Bootstrap

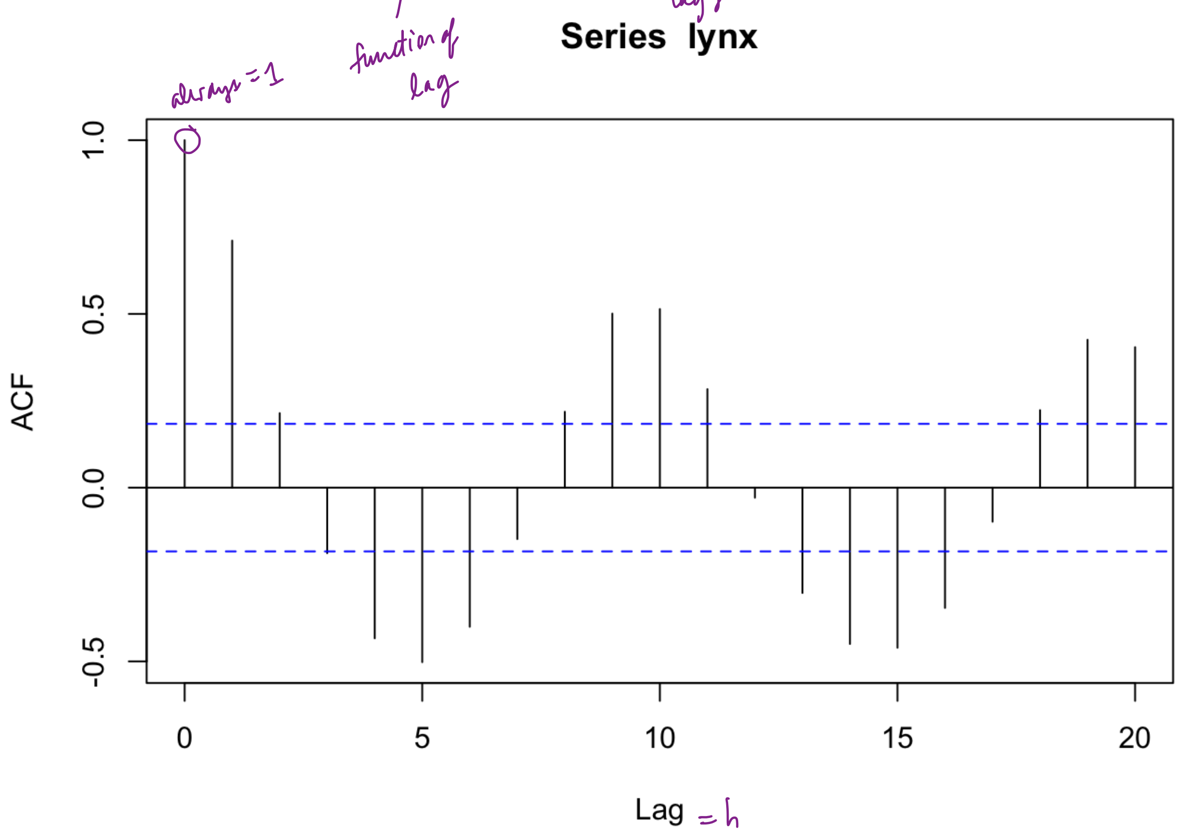
```
library(simpleboot)
B <- 10000

## Your turn: perform the independent bootstrap
## what is the bootstrap estimate se?
```

We must account for the dependence to obtain a correct estimate of the variance!

```
acf(lynx)
```

Auto correlation function, $\gamma(h) = Cor(Y_t, Y_{t+h})$

lagged observations



The acf (autocorrelation) in the dominant terms is positive, so we are *underestimating* the standard error.

3.2.5 Non-overlapping Block Bootstrap

```
# function to create non-overlapping blocks
nb <- function(x, b) {
  n <- length(x)
  l <- n %/% b

  blocks <- matrix(NA, nrow = b, ncol = l)
  for(i in 1:b) {
    blocks[i, ] <- x[((i - 1)*l + 1):(i*l)]
  }
  blocks
}

# Your turn: perform the NBB with b = 10 and l = 11
theta_hat_star_nbb <- rep(NA, B)
nb_blocks <- nb(lynx, 10)
for(i in 1:B) {
  # sample blocks
  # get theta_hat^*
}

# Plot your results to inspect the distribution
# What is the estimated standard error of theta hat? The Bias?
```

3.2.6 Moving Block Bootstrap

```
# function to create overlapping blocks
mb <- function(x, l) {
  n <- length(x)
  blocks <- matrix(NA, nrow = n - l + 1, ncol = l)
  for(i in 1:(n - l + 1)) {
    blocks[i, ] <- x[i:(i + l - 1)]
  }
  blocks
}

# Your turn: perform the MBB with l = 11
mb_blocks <- mb(lynx, 11)
theta_hat_star_mbb <- rep(NA, B)
for(i in 1:B) {
  # sample blocks
  # get theta_hat^*
```

```
}

# Plot your results to inspect the distribution
# What is the estimated standard error of theta hat? The Bias?
```

3.2.7 Choosing the Block size

```
# Your turn: Perform the mbb for multiple block sizes l = 1:12
# Create a plot of the se vs the block size. What do you notice?
```

4 Summary

Bootstrap methods are simulation methods for frequentist inference. as opposed to Bayesian inference, which you should take next semester.

Bootstrap methods are useful for

- many problem types
- especially when standard assumptions are invalid (like non-normal statistics)

Remember Bootstrap principal: The bootstrap dsn should approximate the sampling dsn of the statistic!

Bootstrap methods can fail when

We have extremes or heavy-tailed dsns

Can be computationally expensive (i.e. slow)

need to be careful w/ dependent data.