

# MATH-517: Assignment 3

Philipp Mayer

2025-10-05

## Theoretical exercise

Define the quadratic form

$$Q(\beta) = \sum_{i=1}^n K\left(\frac{X_i - x}{h}\right) (Y_i - \beta_0 - \beta_1(X_i - x))^2 = (Y - X\beta)^T W (Y - X\beta),$$

where  $W = \text{Diag}(W_1, \dots, W_n) = \text{Diag}\left(K\left(\frac{X_1 - x}{h}\right), \dots, K\left(\frac{X_n - x}{h}\right)\right)$ .

Using matrix calculus, we have that  $\frac{\partial Q(\beta)}{\partial \beta} = 2X^T W X \beta - 2X^T W Y$  so that the normal equations are given by  $X^T W X \hat{\beta} = X^T W Y$ . Assuming that  $X^T W X$  is invertible, we immediately obtain from the normal equations that  $\hat{\beta} = (X^T W X)^{-1} X^T W Y$ .

We will start off by compute some terms,  $X^T W X$  and  $X^T W Y$  separately.

$$\begin{aligned} X^T W X &= \begin{bmatrix} 1 & \cdots & 1 \\ X_1 - x & \cdots & X_n - x \end{bmatrix} \begin{bmatrix} W_1 & & \\ & \ddots & \\ & & W_n \end{bmatrix} \begin{bmatrix} 1 & X_1 - x \\ \vdots & \vdots \\ 1 & X_n - x \end{bmatrix} \\ &= \begin{bmatrix} 1 & \cdots & 1 \\ X_1 - x & \cdots & X_n - x \end{bmatrix} \begin{bmatrix} W_1 & W_1(X_1 - x) \\ \vdots & \vdots \\ W_n & W_n(X_n - x) \end{bmatrix} \end{aligned}$$

such that

$$X^T W X = \begin{bmatrix} \sum_{i=1}^n W_i & \sum_{i=1}^n W_i(X_i - x) \\ \sum_{i=1}^n W_i(X_i - x) & \sum_{i=1}^n W_i(X_i - x)^2 \end{bmatrix}$$

Define  $\tilde{S}_{n,k}(x) := nhS_{n,k}(x)$  for  $k = 0, 1, 2$ , where  $S_{n,k}(x)$  are defined as in the README. In view of this definition,  $X^T W X = \begin{bmatrix} \tilde{S}_{n,0} & \tilde{S}_{n,1} \\ \tilde{S}_{n,1} & \tilde{S}_{n,2} \end{bmatrix} = nh \begin{bmatrix} S_{n,0} & S_{n,1} \\ S_{n,1} & S_{n,2} \end{bmatrix}$ .

We now turn to  $X^T W Y$ .

$$\begin{aligned} X^T W Y &= \begin{bmatrix} 1 & \cdots & 1 \\ X_1 - x & \cdots & X_n - x \end{bmatrix} \begin{bmatrix} W_1 & & \\ & \ddots & \\ & & W_n \end{bmatrix} \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} W_1 & \cdots & W_n \\ W_1(X_1 - x) & \cdots & W_n(X_n - x) \end{bmatrix} \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix} \\ &= \begin{bmatrix} \sum_{i=1}^n W_i Y_i \\ \sum_{i=1}^n W_i (X_i - x) Y_i \end{bmatrix} \end{aligned}$$

Defining  $T_0 = \sum_{i=1}^n W_i Y_i$  and  $T_1 = \sum_{i=1}^n W_i (X_i - x) Y_i$ , we have that  $X^T W Y = \begin{bmatrix} T_0 & T_1 \end{bmatrix}^T$ .

We now invert  $X^T W X$ . Since  $X^T W X$  is a  $2 \times 2$  matrix, the inverse is

$$\begin{bmatrix} \tilde{S}_{n,0} & \tilde{S}_{n,1} \\ \tilde{S}_{n,1} & \tilde{S}_{n,2} \end{bmatrix}^{-1} = \frac{1}{D} \begin{bmatrix} \tilde{S}_{n,2} & -\tilde{S}_{n,1} \\ -\tilde{S}_{n,1} & \tilde{S}_{n,0} \end{bmatrix}, \text{ where } D = \tilde{S}_{n,0}\tilde{S}_{n,2} - \tilde{S}_{n,1}^2 \text{ is the determinant.}$$

Recall that we are interested in  $\hat{\beta}_0$  and in view of the above reasoning,

$$\hat{\beta} = \frac{1}{D} \begin{bmatrix} \tilde{S}_{n,2} & -\tilde{S}_{n,1} \\ -\tilde{S}_{n,1} & \tilde{S}_{n,0} \end{bmatrix} \begin{bmatrix} T_0 \\ T_1 \end{bmatrix} \text{ such that } \hat{\beta}_0 = \frac{1}{D} (\tilde{S}_{n,2}T_0 - \tilde{S}_{n,1}T_1) = \sum_{i=1}^n \frac{W_i (\tilde{S}_{n,2} - \tilde{S}_{n,1}(X_i - x))}{D} Y_i$$

Recall that  $\tilde{S}_{n,k} = nhS_{n,k}$ . It follows that  $D = (nh)^2 (S_{n,0}S_{n,2} - S_{n,1}^2)$ . It immediately follows that

$$\hat{\beta}_0 = \sum_{i=1}^n \frac{nh}{(nh)^2} \frac{W_i (S_{n,2} - S_{n,1}(X_i - x))}{S_{n,0}S_{n,2} - S_{n,1}^2} Y_i = \sum_{i=1}^n \frac{1}{nh} \frac{W_i (S_{n,2} - S_{n,1}(X_i - x))}{S_{n,0}S_{n,2} - S_{n,1}^2} Y_i = \sum_{i=1}^n w_{n,i} Y_i,$$

where

$$w_{n,i} = w_{n,i}(x) = \frac{1}{nh} \frac{K\left(\frac{X_i - x}{h}\right) (S_{n,2}(x) - S_{n,1}(x)(X_i - x))}{S_{n,0}(x)S_{n,2}(x) - S_{n,1}(x)^2}.$$

With this result, we answer both question (1) and (2) at the same time.

It remains to show that  $\sum_{i=1}^n w_{n,i}(x) = 1$ . We proceed as follows:

$$\begin{aligned} \sum_{i=1}^n w_{n,i}(x) &= \frac{1}{nh} \sum_{i=1}^n \frac{W_i (S_{n,2}(x) - (X_i - x)S_{n,1}(x))}{S_{n,0}(x)S_{n,2}(x) - S_{n,1}(x)^2} \\ &= \frac{1}{nh} \frac{S_{n,2}(x)}{S_{n,0}(x)S_{n,2}(x) - S_{n,1}(x)^2} \sum_{i=1}^n W_i - \frac{1}{nh} \frac{S_{n,1}(x)}{S_{n,0}(x)S_{n,2}(x) - S_{n,1}(x)^2} \sum_{i=1}^n W_i (X_i - x) \end{aligned}$$

In view of the definitions of  $S_{n,0}(x)$  and  $S_{n,1}(x)$ , we obtain

$$\sum_{i=1}^n w_{n,i}(x) = \frac{S_{n,2}(x)S_{n,0}(x)}{S_{n,0}(x)S_{n,2}(x) - S_{n,1}(x)^2} - \frac{S_{n,1}(x)^2}{S_{n,0}(x)S_{n,2}(x) - S_{n,1}(x)^2} = 1,$$

which concludes the third (3) part of the exercise.

This concludes the theoretical exercise.

## Practical exercise

Unless stated otherwise, your answer to the practical part should include the following elements:

- Description of the aim of the simulation study.
- Description of the different quantities that intervene in your simulation study. Explain how these quantities (fixed or random) are defined and the reasoning behind the choices you made.
- Description of your findings using appropriate graphics and/or tables (with a caption!) that are well commented in the text.

The code should not appear in the PDF report, unless there is a specific reason to include it.

### Structure and details:

NB: To use the code, please read the section “**How to simply plot results**”. Each function is callable from the same `.py` file and terminal.

#### Aim of the simulation study:

The goal is to observe the impact of various parameters on the estimation of the IMSE-optimal bandwidth  $h_{IMSE}$ ,  $\sigma^2$  and  $\theta_{22}$ . In particular how it varies with respect to

- various values (combinations) of the Beta function,  $\alpha, \beta > 0$  (see below)

- various values (combinations) of sample size  $n$  and block size / number of blocks  $N$  (see below)
  - Do quantities depend on  $\alpha, \beta$
- What happens when we select  $N$  using Mallows  $C_p$

#### Fixed quantities:

- $\alpha, \beta > 0$  are the parameters of the Beta distribution, they're usually named **alpha** and **beta** in the code. We have a sample size  $n$  and a number of blocks  $N$ , they're usually denoted descriptively, for example **sample\_size** or **size** and **number\_of\_blocks**. Note that in the README, we consider the size of the blocks instead of the number, but they are pretty much the same in this case. Usually we'll fix  $\sigma^2$  to 1.0 and  $m$  (truth) is fixed.

#### Code architecture:

The technical functions are located in a file **functions.py** with usual alias **fct**. There, we have the following functions:

- **generate\_sample** takes inputs **alpha**, **beta**, **n\_samples**, **sigma\_2** which respectively represent  $\alpha$ ,  $\beta$ , the number of samples  $n$  and the variance of the error  $\epsilon$ ,  $\sigma^2$ . This function returns two one-dimensional arrays, **covariate** and **response** representing  $X$  and  $Y$  respectively.
- **estimate\_parameters** takes inputs **covariate**, **response**, **bandwith**, **p**, **reg**. They represent respectively  $X$ ,  $Y$ , a default bandwidth needed to perform an initial KDE and  $p$  which represents the order. Note that **p**=1 by default and **reg** is set to **1e-5**. It computes the estimated  $\hat{\beta}$  by solving the normal equation. To efficiently compute the solution, we use Cholesky decomposition and **np.linalg.solve** instead of **np.linalg.inv**, which is significantly more stable and efficient in this situation. For even more stability, we add a small regularization (**reg**) before using **np.linalg.solve**. Throughout the calls of the functions, the **reg** parameter is never changed and remains set to its default value of **1e-5**. The returns is a function **beta\_est** that, given a array of covariates, computes the fitted values.
- **estimate\_sigma\_theta** takes inputs **covariate**, **reponse**, **bandwith**, **N\_blocks** which respectively represent  $X, Y$ , a default bandwidth needed to perform the initial KDE and the number of blocks to compute estimates of  $\hat{\theta}_{22}$  and  $\hat{\sigma}^2$ . Note that in the README, it is usually mentioned the size of the blocks instead of the number. We decided to implement in terms of the number of blocks. This choice is due to (my) reasoning logic. The computation of the size, given the number of blocks is given by **size = sample\_size / N\_blocks**. To compute blocks, we compute the size, and then split the data in blocks [**B1** | ... | **Bk** | **Bextra**] **Bextra** is the remaining number of samples. This function returns an estimate of  $\theta_{22}$  and  $\sigma^2$ , following the procedure recommended in the README.

- `est_h_IMSE_support` takes inputs `sigma_2_hat`, `theta_22`, `size`, `covariate` as inputs which respectively represent  $\hat{\sigma}^2$ ,  $\hat{\theta}_{22}$  and the sample size  $n$ . It returns  $h_{IMSE}$  as given in the README. Note that `size` is not a necessary argument as `size=len(covariate)`. This choice was made for clarity. The `covariate` array is a necessary argument as the support of the realizations is given by `support=max(covariate)-min(covariate)` (note that this is always a nonnegative quantity and a positive quantity with probability one). This function returns an estimate of  $h_{IMSE}$ .
- `compute_mallow_C_p` takes inputs `covariate`, `response`, `fixed_bandwith`, `N_blocks` (as previously seen). It computes  $C_p(N)$  as suggested in the README and returns its value.
- `optimal_mallow` takes inputs `covariate`, `reponse`, `fixed_bandwith`, `max_N_blocks`. The first two inputs are as seen previously and the third is the same as `bandwith` seen previously. The last argument represents the maximal number of blocks on which the function computes the  $C_p$  statistics (for m “model” selection). The function returns the number of blocks that minimized the Mallow  $C_p$  statistic. The function calls `compute_mallow_C_p` for each block size.
- `h_IMSE_Cp_optimized` takes inputs `covariate`, `reponse`, `number_of_samples`, `default_bandwith`, `max_number_of_blocks`, which respectively represent  $X, Y, n$ , the fixed bandwidth as before (previously called `bandwith` or `fixed_bandwith` and the maximum number of blocks to test (previously called `max_N_blocks`). The function calls `optimal_mallow` and re-estimates  $\sigma^2$  and  $\theta_{22}$ , given the Mallow-optimal number of blocks (the return of `optimal_mallow`). This function returns an estimation of  $\sigma^2$  and  $\theta_{22}$  with Mallow-optimal selection of the number of blocks.
- `simulate` takes inputs `alpha`, `beta`, `number_of_samples`, `error_variance`, `default_bandwith`, `number_of_blocks`, which respectively represent  $\alpha, \beta, n, \sigma^2$ , a default bandwidth (as previously seen) and a number of blocks. It first calls `generate_sample` with  $\alpha, \beta, n, \sigma^2$ , which returns `covariate`, `response`. With that return, it calls `estimate_sigma_theta` with the default bandwidth and the number of blocks. Finally, it calls `est_h_IMSE_support` with the computed quantities. This function returns the estimations of  $h_{IMSE}, \sigma^2$  and  $\theta_{22}$ . We mostly use this function to ease the production of plots.

The plotting functions are accessible through a file called `plot_results.py`. There, one can use any of the functions below:

- `plot_simple_fit` takes inputs `covariate`, `response`, `fixed_bandwith`, `sigma_2`, representing  $X, Y$ , the fixed bandwidth as previously seen and  $\sigma^2$  (for the plot title). It returns a plot showing visually how the choice of bandwidth impacts the fits. To produce this plot simply, refer to the subsection below “How to simply plot results”. An example of call would be:

– `py plot_results.py simple_plot 0.1 0.2 10000 0.2 1`

- `plot_alpha_beta_impact` takes inputs `alphas`, `betas`, `error_variance`, `default_bandwidth`, `number_of_blocks`, which respectively represent a list of alphas  $[\alpha_1, \dots, \alpha_{n_\alpha}]$ , as list of betas  $[\beta_1, \dots, \beta_{n_\beta}]$ , the default bandwidth as seen previously and `error_variance` represents  $\sigma^2$ . This function produces three heat maps of the values of estimated  $h_{IMSE}$ ,  $\sigma^2$  and  $\theta_{22}$  for different values of  $\alpha_i, \beta_j$ . To produce this plot simply, refer to the subsection below **“How to simply plot results”**. An example of call would be:

– `py plot_results.py alpha_beta_impact 100 1.0 0.1 5`

- `plot_sample_blocks_impact` takes inputs `number_of_samples_range`, `number_of_blocks`, `alpha`, `beta`, `error_variance`, `default_bandwidth`, which respectively represent a list  $[n_1, \dots, n_{k_n}]$  of values for the number of samples  $n$ , a list  $[N_1, \dots, N_{k_N}]$  of values for the number of blocks  $N$ , fixed values of  $\alpha, \beta$  and  $\sigma^2$ . This function produces three heat maps of the values of estimated  $h_{IMSE}$ ,  $\sigma^2$  and  $\theta_{22}$  for different values of  $n_i$  and  $N_j$ . To produce this plot simply, refer to the subsection below **“How to simply plot results”**. An example of call would be:

– `py plot_results.py sample_blocks_impact 0.1 0.2 1.0 0.1`

- `plot_sample_size_impact_mallow` takes inputs `alpha`, `beta`, `max_number_of_samples`, `step`, `error_variance`, `default_bandwidth`, `max_number_of_blocks`, as seen previously. `max_number_of_blocks` is set to 10 by default. The additional new parameter represents the number of steps between two different sample sizes. The list of sample sizes to test are then given by `n_range = np.arange(start=default_start, stop=max_number_of_samples, step=step)`. This function produces three (sub) plots representing how the estimations of  $h_{IMSE}$ ,  $\sigma^2$  and  $\theta_{22}$  evolve when computed on  $n$  samples with Mallows-optimal number of blocks. To produce this plot simply, refer to the subsection below **“How to simply plot results”**. An example of call would be:

– `py plot_results.py sample_size_mallow_evolution 0.1 0.2 2000 10 1.0 0.1`

### **How to simply plot results:**

To call them, use the console with one of the following keywords `key`  $\in \{ \text{simple\_plot, alpha\_beta\_impact, sample\_blocks\_impact, plot\_sample\_size\_impact\_mallow} \}$  and call as follows: `py plot_results.py key parameter1 parameter2 parameter3 ...`

where `parameter1 parameter2 ...` are the functions parameters, detailed below:

- when `key=simple_plot`: The parameters are in the following order `alpha`, `beta`, `number_of_samples`, `fixed_bandwidth`, `variance`, representing respectively  $\alpha, \beta, n$ , the fixed bandwidth (as previously mentioned) and  $\sigma^2$ .

- when `key=alpha_beta_impact`: The parameters are in the following order `number_of_samples`, `error_variance`, `default_bandwidth`, `number_of_blocks`, representing respectively the number of samples  $n$  to test,  $\sigma^2$ , the default bandwidth (as previously mentioned) and the number of blocks.
- when `key=sample_blocks_impact`: The parameters are in the following order `alpha`, `beta`, `error_variance`, `default_bandwidth`, as previously seen.
- when `key=plot_sample_size_impact_mallow`: The parameters are in the following order `alpha`, `beta`, `max_number_of_samples`, `step`, `error_variance`, `default_bandwidth`.

When a call is successful, the console will display `TASK:<followed by the actual task it is performing>` and will print `TASK completed` when the plot has been produced.

### Findings and reports:

We start off with a simple visualization of the setup for fixed  $\alpha, \beta, n$ . The plot below showcases how different bandwidth choices visually impact the fit. In orange is the true  $m(x)$ . In red is a (relatively good fitting) line. We observe that as the bandwidth goes closer to zero, the fits become better. However, when too close to zero (the bandwidth), the fitted function appears to be too sensible and does not provide a robust fit to the truth.

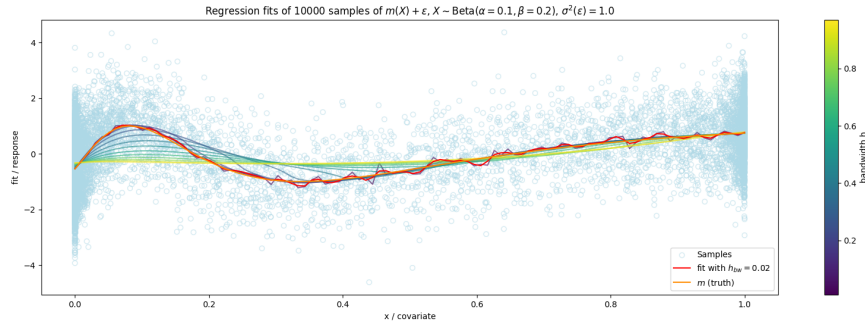


Figure 1: Visual plot of bandwidth impact, call of `py plot_results.py simple_plot 0.1 0.2 10000 0.2 1`

Here below is another plot, after calling `py plot_results.py simple_plot 10 10 10000 0.07 1`

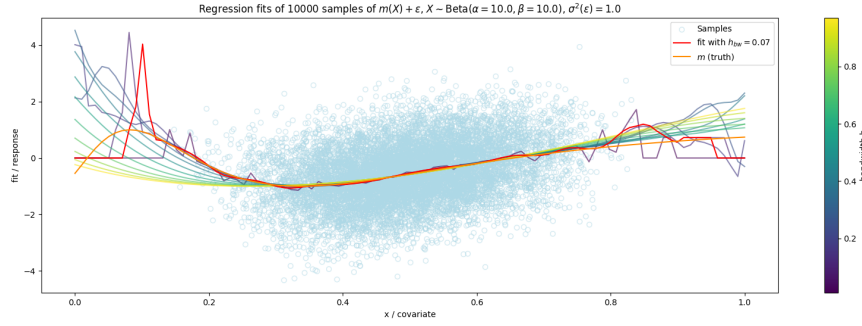


Figure 2: Visual plot of bandwidth impact, call of `py plot_results.py simple_plot 10 10000 0.07 1`

*NB: Note that to produce these variations, one must directly change export name of the .png files.*

- The choice of  $N$  should possibly depend on  $\alpha, \beta$  (and  $m$ ) as the concentrations of samples will vary in  $[0, 1]$ . Given  $m$  and depending on  $N$ , we may not be able to capture the volatility of  $\theta_{22}$  accurately.

Here below are heatmaps of the results of estimations of  $h_{IMSE}, \sigma^2, \theta_{22}$  against various values of  $\alpha, \beta$ . The  $x$ -axis represents  $\beta$  and the  $y$ -axis represents  $\alpha$ . Here the alphas and betas are chosen in `np.arange(0.1, 10, step = 0.5)`. We observe what seems to be a boundary effect when either  $\alpha$  or  $\beta$  is close to zero on the estimation of  $h_{IMSE}$  and  $\theta_{22}$  in particular. For  $h_{IMSE}$ , when  $\alpha$  or  $\beta$  (or both) are close to zero, the values of  $h_{IMSE}$  are of larger magnitude in comparison to the other cases. The inverted observation can be made for  $\theta_{22}$ 's estimation where the values are lower when  $\alpha, \beta$  are close to zero in comparison to the other case.

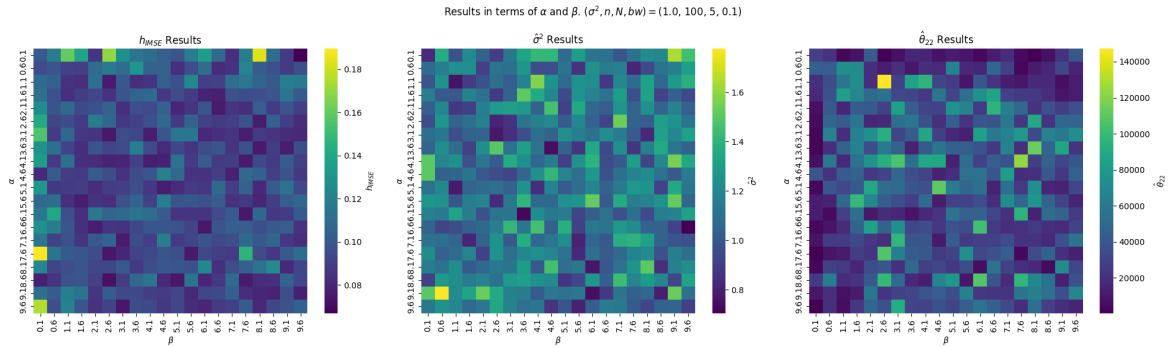


Figure 3: Impact of the choices of alpha and beta on estimations, call of `py plot_results.py alpha_beta_impact 100 1.0 0.1 5`



To better emphasize on this boundary effect, we recomputed the heatmaps for  $\alpha, \beta$  taking values in `np.arange(0.01, 5, step = 0.5)`. There, the boundary effect on the estimation of  $h_{IMSE}$  and  $\theta_{22}$  are even more noticeable.

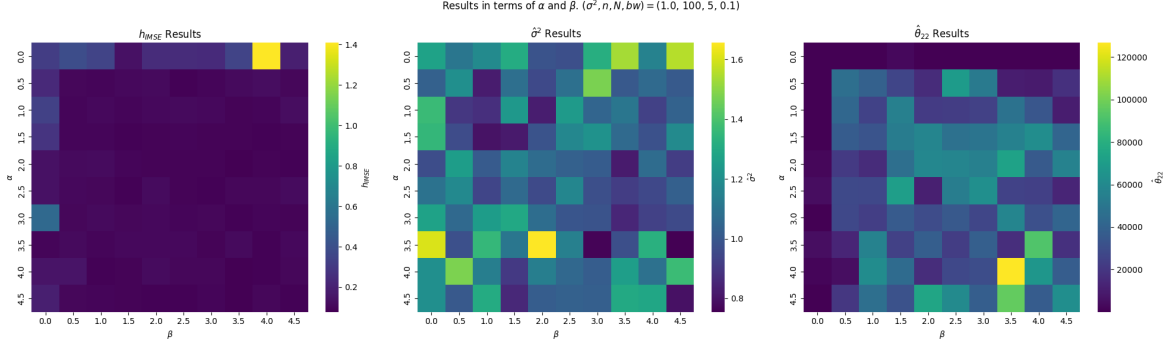


Figure 4: Impact of the choices of alpha and betas, close to zero

Note that in both cases, the estimations of  $\sigma^2$  remain relatively stable between 0.8 and 1.2 approximately.

To complete the observations around the impacts of  $\alpha, \beta$  on the estimations, we perform a last run for  $\alpha, \beta$  taking values in `np.arange(0.01, 1, step = 0.01)`. There, we observe a fade from the left and upper boundaries to the center for  $\theta_{22}$ , and a strong boundary effect on  $h_{IMSE}$ , as already expected. We additionally observe that the estimations of  $h_{IMSE}$  remain stable for values of  $\alpha, \beta$  far enough from zero (both at the same time) and that the estimations of  $\sigma^2$  are reasonably stable on average, though very “white-noise” looking.

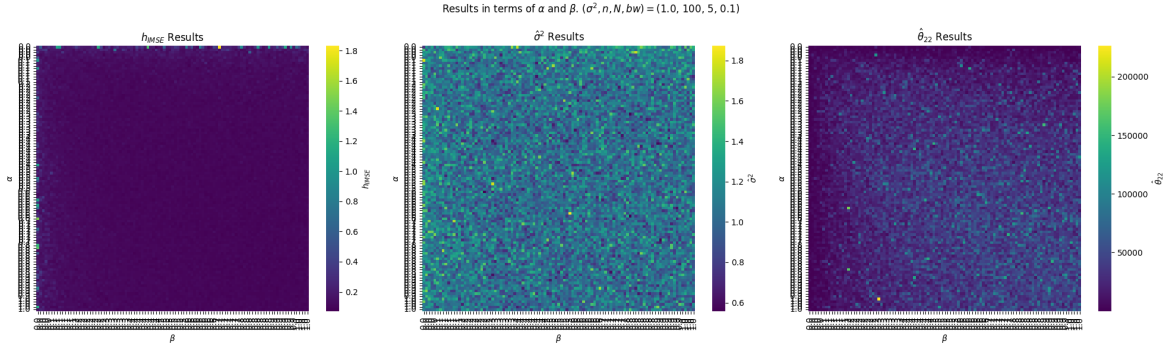


Figure 5: Impact of the choices of alpha and beta with increased granularity and closer to zero

In conclusion,

- The estimation of  $h_{IMSE}, \theta_{22}$  are more stable for values of  $\alpha, \beta$  far enough from zero (boundary effect). The estimation of  $h_{IMSE}$  has very few extreme values (so low variance) for values of  $\alpha, \beta$  large enough.
- The estimation of  $\sigma^2$  appears to not be affected a lot by changes of  $\alpha, \beta$ .
- initial estimations are that  $h_{IMSE}$  is below 0.2 on average,  $\sigma^2$  is, as expected, around 1.0 and  $\theta_{22}$  appears to be around 50000 on average.

*NB: Note that to produce these variations, one must directly comment / uncomment parts of the code as well as change export name of the .png files. The default computation is made for values of  $\alpha, \beta$  in `np.arange(0.1, 10, step = 0.5)`.*

We now take a look at how the number of blocks (or similarly the size of the blocks) and the sample size impact the estimations.

As already mentioned, we think of number of blocks instead of size of blocks. As the number of blocks decrease, the size increases and vice-versa. First notice that we obtain on-average inconsistent estimations of  $h_{IMSE}$  and  $\sigma^2$  when the sample size is low and especially when  $n$  is small and  $N$  is large. For  $\theta_{22}$  on the other hand, the computations appear to be inconsistent in comparison to the average when the number of blocks is small. It becomes more stable as the number of blocks increase. This is an observation that one can also make for  $h_{IMSE}$  and  $\sigma^2$ .

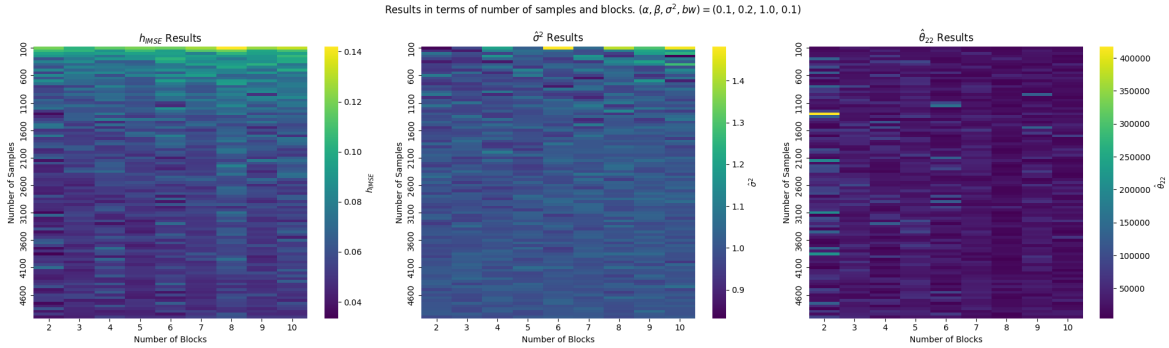


Figure 6: Impact of sample size and number of blocks on estimations (color map)

In conclusion, for this choice of  $(\alpha, \beta) = (0.1, 0.2)$ ,

- As expected, the sample size positively increases the on-average consistency of the estimations.
- The size of the blocks decreases (and the number of blocks increases) the estimations appear to become stable when  $n$  (the sample size) is large enough. However, this effect does not seem to be very pronounced as soon as the sample size is large enough.

Consider the case where  $(\alpha, \beta) = (10, 10)$  below.

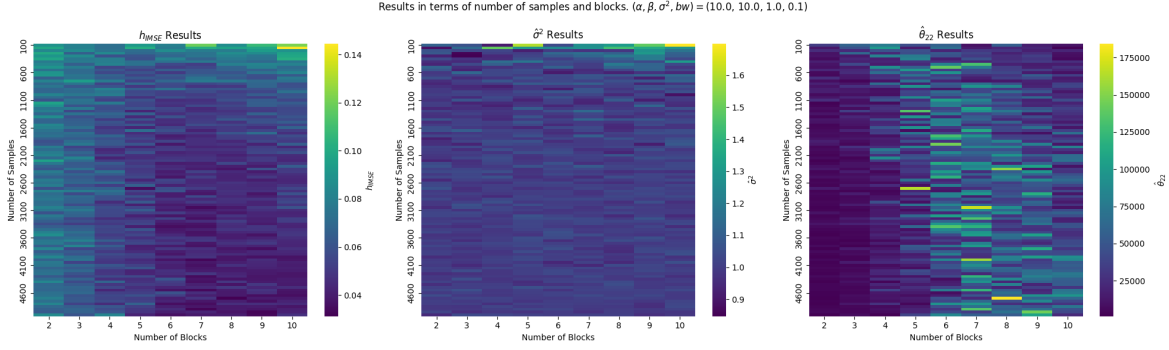


Figure 7: Impact of  $n$  and  $N$  when  $(\alpha, \beta) = (10, 10)$

We observe that

- Similarly to before, as  $n$  increases, the estimations become more stable in comparison to the average.
- On the other hand, the choice of  $N$  appears to impact the stability significantly (especially for  $\theta_{22}$  and  $h_{IMSE}$ ) in comparison to the previous case where  $(\alpha, \beta) = (0.1, 0.2)$ .  $N$  will change significantly how the estimates are captured when the density of samples move (this is controlled by  $\alpha, \beta$ . We can deduce that  $N$  should (or at least could) depend on  $\alpha, \beta$ .
- In this case the number of blocks shouldn't "split" the data in heteroskedastic sub-samples. Data should be *i.i.d.* for a LLN.

*NB: Note that to produce these variations, one must directly change export name of the .png files.*

Let us now take a look at how the estimations behave against the sample size when the number of blocks (or similarly the size) is selected using Mallow's  $C_p$  selection criterion. This is a reasonable way to compare the impact of sample sizes alone.

We start by fixing  $(\alpha, \beta) = (0.1, 0.2)$  to be in the situation we have already extensively discussed.

We observe, as expected and previously observed, that small sample sized (even with optimal number of blocks) yield unreasonable estimations in comparison to the average. Moreover, the estimation of  $h_{IMSE}$  appears to average around the value of 0.07 after the elbow (see second figure for clear elbow). The estimation of  $\sigma^2$  also becomes stable and converges to, it appears close to, 1.0 (as expected). Furthermore, as already observed previously, the estimation of  $\theta_{22}$  is relatively unstable / changes a lot.

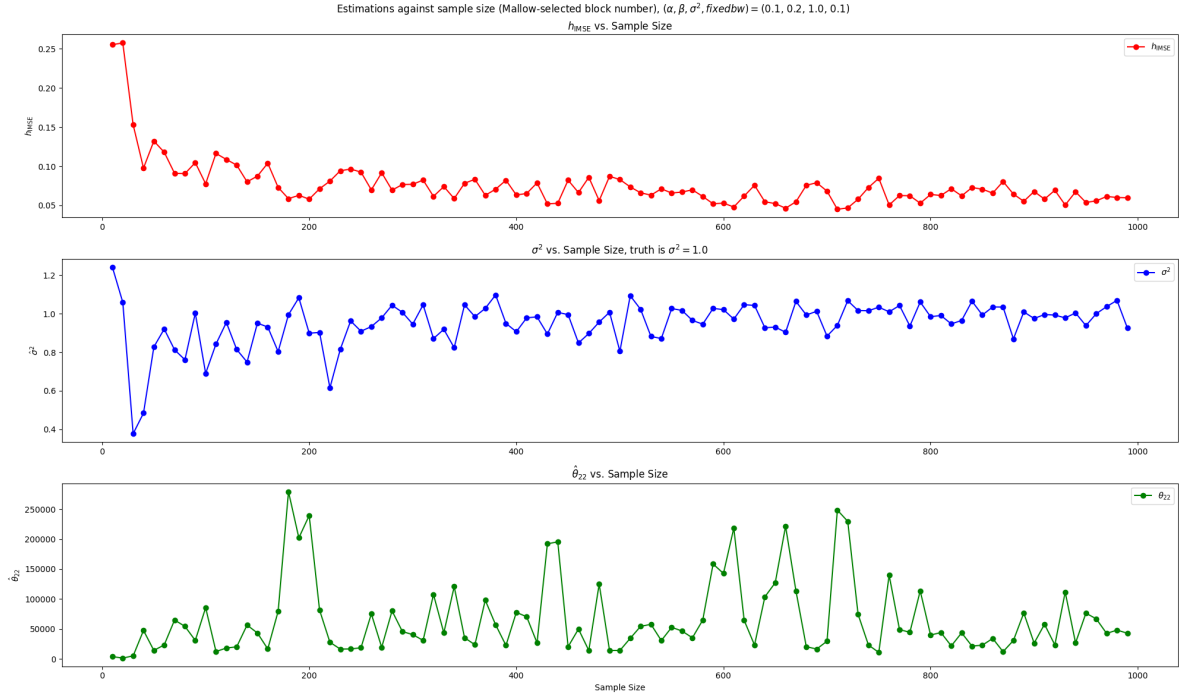


Figure 8: Estimations against sample size with Mallow-selected block number / size

For confirmation purposes, here is a version going up to 2000 samples:

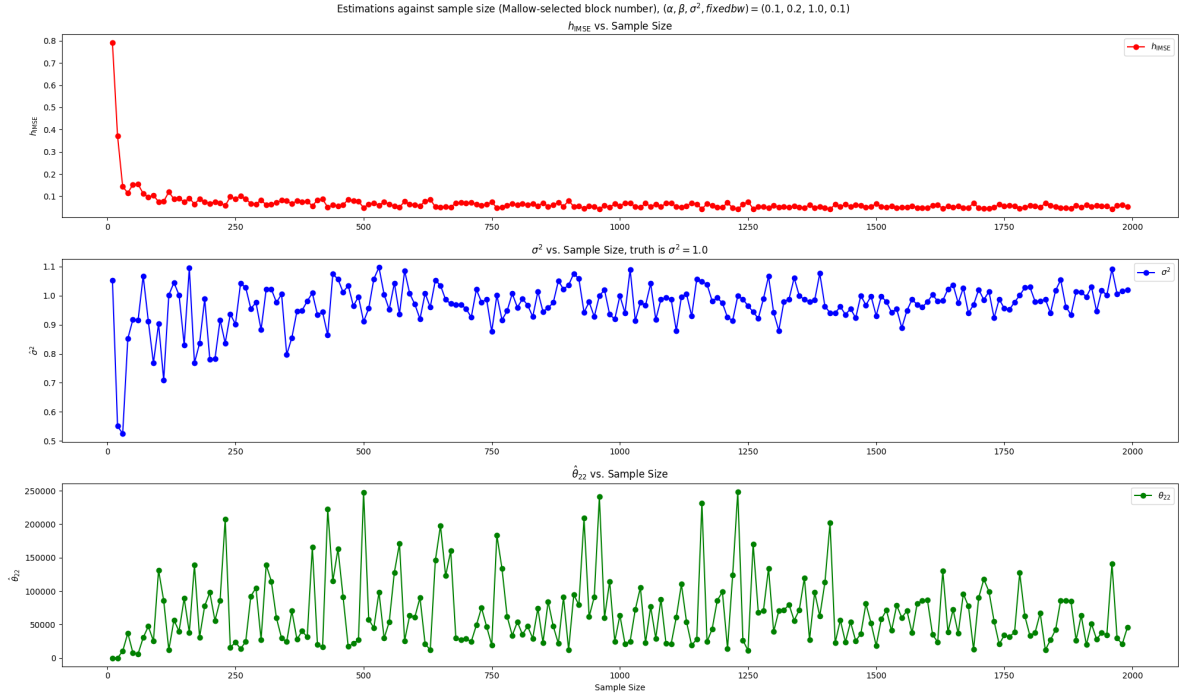


Figure 9: Mallow selected evolution up to 2000 samples

Here below is the call of `py plot_results.py simple_plot 0.1 0.2 10000 0.07 1` with the bandwidth of 0.7 mentioned above.

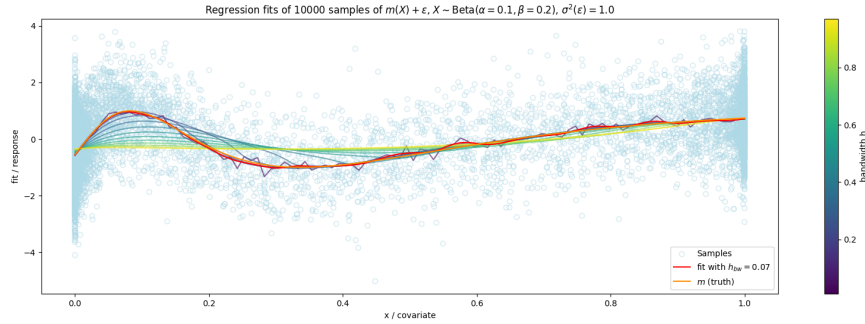


Figure 10: Plot with bandwidth of 0.7

Again, changing  $\alpha, \beta$  produces significantly different results, as shown here below when calling `py plot_results.py sample_size_mallow_evolution 20 10 1000 10 1.0 0.1`, that is we let  $(\alpha, \beta) = (20, 10)$ :

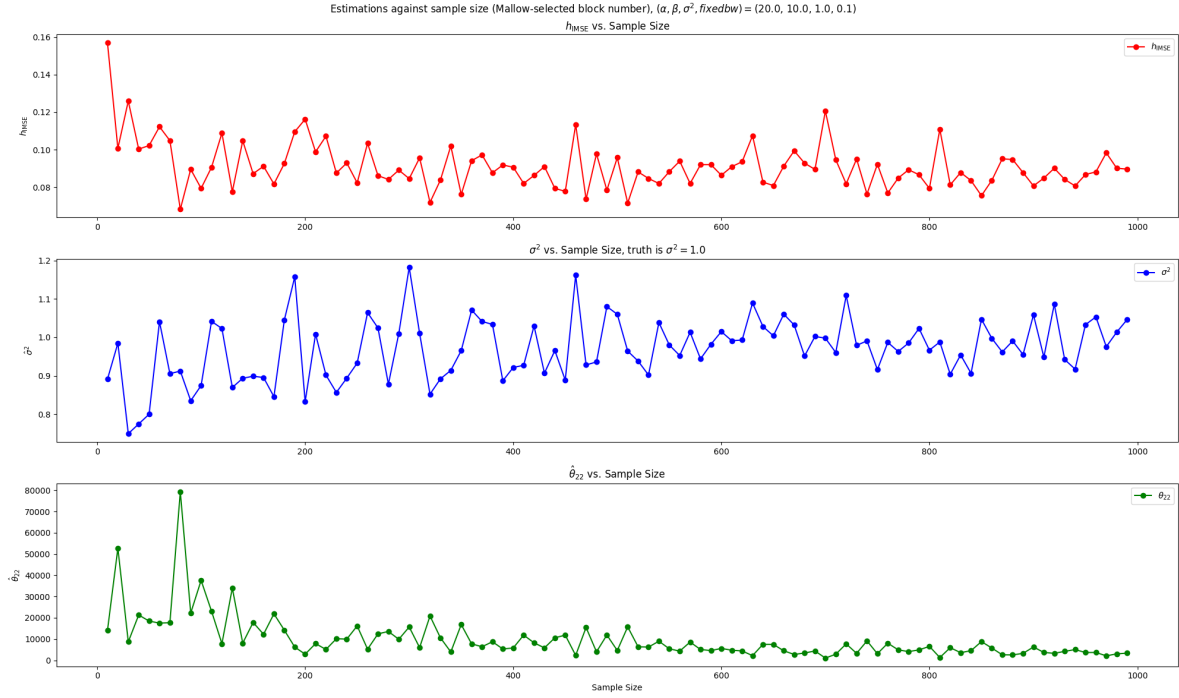


Figure 11: Call of `py plot_results.py sample_size_mallow_evolution 20 10 1000 10 1.0 0.1`

This time on the other hand, the computation of  $\theta_{22}$  becomes stable, the estimation of  $\sigma^2$  is still relatively stable, closely around 1.0. and  $h_{IMSE}$  is estimated around 0.09 after the elbow.  $\theta_{22}$  is close to 10000 after the computations stabilize.

- This again confirms the suspicion that  $\alpha, \beta$  significantly impact  $h_{IMSE}$  and the other estimations, especially  $\theta_{22}$ .

*NB: Note that to produce these variations, one must directly change export name of the .png files.*