Semi and Non-parametric Econometrics Project

Bruno BJAI, Robin FUCHS and Martin MUGNIER

December 2018

1 Presentation of the papers

1.1 He Hu (2002) : Markov Chain Marginal Bootstrap

Bootstrap methods are deemed to be particularly efficient tools to construct reliable confidence intervals when asymptotic normality lacks efficiency or that some assumptions such as independence and identical distribution are violated. However, it usually comes at the price of an important computational cost which can be due to both the numerous resampling implemented and the problem to be solved. It is especially the case for classical methods when performing M-estimation.

The canonical paper under study is from He and Hu (2002) and suggests a new method, the Markov Chain Monte Carlo Marginal Bootsrap (MCMB), which aims to provide faster computation in the framework of multiparameter M-estimation. The trick is done by reducing the dimension of the problem solved at each iteration. This variant has a broad range of applications in linear models inference but also in quantile regression as we will see more in details through the analysis of the second paper.

Because the paper is from 2004, it may be also interesting to look at recent developments.

First we present the method along with the main theoretical and empirical results. Then we provide ...

Framework Consider independent observations $Y = (Y_1, \dots, Y_n)$. The corresponding M-estimator typically solves the following :

$$S(Y, \theta) = \sum_{i=1}^{n} g_i(Y_i, \theta) = 0 \tag{1}$$

where $\theta \in \mathbb{R}^d$ and g_i maps the space of (y, θ) to $\mathbb{R}^d$.

First of all, it is important to note that, in this setting, the parameter is exactly identified since we have as many conditions as parameters. Furthermore, by assuming that Y is a sequence of independent random vectors whose distributions depend on the parameter $\theta \in \Theta$ and such that $\mathbb{E}_\theta[S(Y, \theta)] = 0$, then equation (1) becomes an *unbiased estimation equation*. The resulting estimators, called M-estimators, include among others the so-called MLE and GMM estimators depending on whether one considers log-likelihood first-order conditions or moment conditions. What for quantile regression ? (minimization $\rightarrow$ adapt).

The goal for building confidence interval based on bootstrap is to find a resampling and estimation method that preserves the correlation structure of the true parameter (e.g. variance-covariance matrix approximated by the second moments of a Markov Chain) and such that the joint distribution of $n^{-\frac{1}{2}}(\hat{\beta}-\beta)$ is well approximated. Conventional MCMC methods would sample randomly B sub-samples of equal size from the observations and solve repeatedly (1), a p-dimensional problem. By denoting P_n the empirical density of the sample Y, it means that $\mathcal{L}\left(n^{-\frac{1}{2}}(\hat{\beta}_n^{MCMC}-\hat{\beta}_n)\middle| P_n\right)$ is considered as a correct estimator of $\mathcal{L}\left(n^{-\frac{1}{2}}(\hat{\beta}_n-\beta)\right)$. Rather than working with these types of conditional distributions of $\hat{\theta}^*$, the MCMB algorithm samples randomly from each coordinate of g_i and solves marginally, coordinate by coordinate equation (9). Under some suitable conditions fulfilled by a general class of parametric models, this method is shown to be asymptotically consistent.

For the sake of presentation, it is assumed that the function g_i can be decomposed as follow : $g_i(Y_i,\hat{\theta}) = a_i z_i$ for $\hat{\theta}$ an estimator of $\theta = (\theta_1,\dots,\theta_d)'$.

Algorithm 1 MCMB

```
 1: procedure INITIALIZE
 2:     $\hat{\theta}^{(0)} \leftarrow \hat{\theta}$ and $k \leftarrow 1$
 3: top:
 4:     for each $j \in \{1,\dots,p\}$ do
 5:         Draw a bootstrap sample $\{z_{1j}^{*(k)},\dots,z_{nj}^{*(k)}\}$ from $\{z_1,\dots,z_n\}$
 6:     for each $j \in \{1,\dots,p\}$ do
 7:         Solve for $\hat{\theta}_j^{(k)}$ from $S_j\left(Y,\hat{\theta}_1^{(k)},\dots,\hat{\theta}_{j-1}^{(k)},\hat{\theta}_j^{(k)},\hat{\theta}_{j+1}^{(k-1)},\dots,\hat{\theta}_p^{(k-1)}\right) = S_j^{*(k)}$,
 8:         where $S_j$ is the $j$-th component of $S(Y,\theta)$, and $S_j^{*(k)}$ is the $j$-th component of
 9:         $\sum_{i=1}^n a_i z_{ij}^{*(k)}$
10:     if $k > \overline{K}$ then Stop
11:     $k \leftarrow k+1$
12:     go to top.
```

Results The main general result is to provide sufficient conditions such that the empirical first two moments of the resulting chain for an asymptotically normal estimator $\hat{\theta}$ (converging in distribution towards a $\mathcal{N}(0,\Sigma)$) will converge in probability towards $(0,\Sigma)$. Three theorems are proved to demonstrate that M-estimators with smooth score functions, Minimum L_q distance estimators, least-absolute-deviation (LAD) regression and generalized M-estimators verify these conditions and are thus *MCM bootstrappable*. Another theorem under additional assumptions states the *MCM bootstrappability* for MLEs.

- Heteroskedasticity (link with Kogerschinsky)
- $m \leq K_n$ condition
- non proved for GMM

Results from the empirical part :

1.2 Kocherginsky et al. (2005): Application to quantile regression and MCMB-A

1.2.1 Quick reminder about quantile regression

Useful or not ?

1.2.2 Inference in quantile regression

The method of Monte Carlo Marginal Bootstrap presents an important interest in the framework of inference for quantile regressions. Indeed, the construction of confidence interval for quantile regression estimators is particularly challenging. Several methods exist in order to estimate the variance of the estimator, but each one has its drawbacks, either in terms of accuracy or computational complexity. We can distinguish between three main categories of inference methods for quantile regression:

1. **Direct estimation of V_τ (sparsity):** under the iid assumption, there exists an analytical form for the variance-covariance matrix of the estimator, that requires the estimation of $f(0)$, with f the probability density function of the errors. Kernel smoothing methods can be used to estimate the distribution of the errors. In the non-iid case, asymptotic unbiased estimator of $f_i(0)$ exist. These methods can work well in large samples, but they exhibit unstable finite samples properties. There is also the issue of selection of the smoothing parameter. For high dimensional data, the computation time can also be prohibitively high ;
2. **Rank-score method:** this methods relies on the inversion of a rank score test. Its advantages is that it does not require to choose any smoothing parameter. Unfortunately, the method is highly time consuming for large datasets or high-dimension problems, and it does not provide the variance covariance matrix of β_τ (only separate confidence intervals for each component);
3. **Resampling:** we can distinguish between two families of resampling methods: bootstrapping of pairs and bootstrapping of residuals, which is valid only in the iid framework. The bootstrapping of pairs is rather effective to estimate the covariance matrix, but it requires to recompute the quantile estimator at each draw, which can be time consuming. A rule of thumb indicates that at least 50 draws are needed to obtain a decent estimate, which is already quite a lot. Alternatively, confidence intervals can be estimated using the percentile on the bootstrap draws, but it requires many more draws, and is not considered here;

In the end, the major problem for the estimation of confidence intervals for quantile regression estimator is the computation complexity, whatever the method considered. The MCMB algorithm provides an alternative way to perform inference at a moderate computational cost for a moderate amount of observations (in the end, asymptotic estimation remains the best method for the biggest datasets).

- How to compare confidence intervals ? => Coverage and length (trade-off) + computational complexity (computational time) ;
- n = 10 000 and p=50 ;

1.2.3 Application of MCMB to quantile regression

As explained in He Hu (2002), MCMB has several advantages compared to alternative methods. The main feature of the algorithm is that it transforms a p-dimensional equation into p one-dimensional ones at each of its iteration, enabling significant computational efficiency gains. This reminds of usual Monte Carlo Markov Chain (MCMC) algorithms such as Gibbs samplers as it turns high dimensional problems into series of one dimensional ones. However, it does not use conditional distributions to draw elements but solves linear estimating equations. Also, notice that the resampling is done for the computation of each element, at each step. It produces a sequence of $\beta^{(k)}$ that follows a multivariate AR(1) process.

- Validity of MCMB is assured if K (length of the chain) grows with n (number of observations) in a controlled manner ;

In the framework of the quantile regression, this is how the algorithm is applied. Recall that the estimator for quantile regression is given by:

$$\hat{\beta}_\tau = \arg\min_\beta \left(\sum_{i=1}^{n} \rho_\tau (y_i - x_i'\beta) \right) \tag{2}$$

So the first order condition is given by:

$$\sum_{i=1}^{n} \psi_\tau (y_i - x_i'\hat{\beta}_\tau) x_i' = 0 \tag{3}$$

with $\psi_\tau(x) = \tau \mathbb{1}(x > 0) + (\tau - 1)\mathbb{1}(x > 0)$ the derivative of the check function.

As a consequence, we are here in the usual setting of M estimation, with $g_i(Y_i, \theta) = \psi_\tau(Y_i, \theta)x_i$. We can use the MCMB algorithm, recall the decomposition of g_i that becomes:

$$\begin{aligned}
&\sum_{i=1}^{n} g_i(Y_i, \theta) = \sum_{i=1}^{n} a_i z_i \\
\Leftrightarrow &\sum_{i=1}^{n} \psi_\tau (y_i - x_i'\hat{\beta}_\tau) x_i = \sum_{i=1}^{n} a_i z_i \\
\Rightarrow &\sum_{i=1}^{n} \psi_\tau (y_i - x_i'\hat{\beta}_\tau) x_i = \sum_{i=1}^{n} z_i && \text{if we fix } a_i = 1
\end{aligned}$$

We take the centered $z_i = \psi_\tau(y_i - x_i'\hat{\beta}_\tau)x_i - \bar{z}$ in order to improve performance.

1.2.4 The MCMB-A method

The basic MCMB algorithm give a chain $\beta^{(k)}$ which has high autocorrelation. That is a bas property as, for a given chain length, independent elements will give a better estimate of the covariance matrix...but it can be corrected with MCMB-A.

The MCMB-A method is a simple affine transformation of the parameter space that applies a standardisation of the X by $\tilde{X} = (X'X)^{-1/2}X$ before performing the usual MCMB, as detailed in part 3.2 of Kocherginsky et al (2005). However, this transformation seems erroneous. Indeed, for a matrix X of dimension n $\times$ p, we have $(X'X)^{-1/2}$ of dimension p $\times$ p, which can not be multiplied by X. Following part 3 of Kocherginsky and He (2007), we adopt the correct transformation $\tilde{X} = X(X'X)^{-1/2}$.

The estimator obtained is then destandardised: $\beta^{(k)} = (X'X)^{-1/2}\tilde{\beta}^{(k)}$. This simple trick not only removes the asymptotic correlations among the components of $\tilde{\beta}$, but also the autocorrelations between the elements of the generated sequence. Moreover, the estimator has the desirable property of being robust to heteroscadasticity.

Notice in addition that this transformation does not imply additional computational costs: just one transformation at the beginning and one at the end. This is true as long as p (the number of dependent variables) remains moderate, because this transformation implies to compute the square root of a $p \times p$ matrix.

Proof:
Recall that $\beta^{(k)}$ is asymptotically a multivariate AR(1) process:

$$n^{-1/2}(\beta^{(k)} - \hat{\beta}_\tau) = A_n n^{-1/2}(\beta^{(k)} - \hat{\beta}_\tau) + u_{n,k} \tag{4}$$

We can determine the analytical expression of A_n:

$$A_n = -\left(L(X'X)\right)^{-1}\left(X'X - L(X'X)\right) \tag{5}$$

So, if we replace X by $\tilde{X} = X(X'X)^{-1/2}$, we notice that:

$$\begin{aligned}
\tilde{X}'\tilde{X} =& \left(X(X'X)^{-1/2}\right)' X(X'X)^{-1/2} \\
=& \left((X'X)^{-1/2}\right)' X'X(X'X)^{-1/2} \\
=& \left(((X'X)^{1/2})^{-1}\right)' (X'X)^{1/2} \\
=& \left(((X'X)^{1/2})'\right)^{-1} (X'X)^{1/2} \\
=& (X'X)^{-1/2}(X'X)^{1/2} \\
=& I
\end{aligned}$$

Which implies,

$$\begin{aligned}
A_n =& -\left(L(\tilde{X}'\tilde{X})\right)^{-1}\left(\tilde{X}'\tilde{X} - L(\tilde{X}'\tilde{X})\right) \\
=& -\left(L(I)\right)^{-1}\left(I - L(I)\right) \\
=& -I^{-1}\left(I - I\right) = 0
\end{aligned}$$

So, our estimator verifies:

$$n^{-1/2}(\beta^{(k)} - \hat{\beta}_\tau) = u_{n,k} \tag{6}$$

1.2.5 Time-saving strategy

At each iteration, the algorithm is supposed to solve the following p-dimensional problem:

$$\sum_{i=1}^{n+1} \psi_\tau(y^* - x_{i,j}b)x_{i,j} = c^* \tag{7}$$

Which is actually (3.4) in Kocherginsky,

$$\arg\min_b \sum_{i=1}^{n} \rho_\tau(y^* - x_{i,j}b)x_{i,j} \tag{8}$$

with,

$$\begin{aligned}\sum_{i=1}^{n+1} \rho_\tau(y^* - x_{i,j}b) &= \sum_{i=1}^{n+1}(y^* - x_{i,j}b)(\tau - \mathbb{1}\{y^* - x_{i,j}b < 0\}) \\ &= \sum_{i=1}^{n+1}(z_i - b)(\tau - \mathbb{1}\{x_{i,j}(z_i - b) < 0\})x_{i,j} \qquad \text{we define } z_i = y^*/x_{i,j}\end{aligned}$$

It can be proved (see Appendix) that this is equivalent to the following problem:

$$\arg\min_b \sum_{i=1}^{n+1} |x_{i,j}|\rho_{\tau^*}(z_i - b) + \sum_{i=1}^{n+1} |x_{i,j}|c(x_{i,j}, z_i)$$

with $c(x_{i,j}, z_i)$ not depending on b.

So the original p-dimensional problem is equivalent to the determination of the weighted τ^*-th quantile of order of z_i.

Restriction on the interval on which β is searched at each iteration after a few iteration of the chain.

1.2.6 MCMB-AB

Another refinement of the MCMB method is developed in Kocherginsky and He (2007). It consists in adding another transformation of the variable space, called B. This second type of transformation aimed at broadening the applicability of the method to frameworks in which the hypotheses necessary for applicability are not fully verified.

More precisely, this transformation insures that the empirical estimates $\frac{1}{n}\sum_{i=1}^{n}[\psi_i(y_i,\theta)\psi_i(y_i,\theta)^T]$ and $\frac{1}{n}\sum_{i=1}^{n}[\frac{\partial}{\partial\theta}\psi_i(y_i,\theta)]$ converge to the Fisher information matrix. The B matrix involved is the following: $-\mathbb{E}_\theta[\frac{\partial}{\partial\theta}\psi(y,\theta)]^T \cdot [Var(\psi(y,\theta))]^{-1}$.

Even in our quantile regression framework, the addition of the B transformation could lead to more consistent estimates. However, several shortcomings deserve to be mentioned. First, this B transformation requires the estimation of the probability density function of the residuals, for instance by kernel techniques. This is a major drawback as it involves the selection of the bandwidth parameter, which sends us back to the shortcomings of the direct estimation of the variance matrix. Moreover, this B transformation does nothing more to remove autocorrelation, and consequently represents a very limited improvement in terms of accuracy from the previous MCMB-A method. Those drawbacks lead us to discard the use of this additional transformation for our simulations.

2 Simulations

We provide in the following section a series of simulations to assess the performance of the algorithm in the construction of confidence intervals for quantile regression parameters. In addition, we add several features in order to improve the computational efficiency or the speed of the baseline form of algorithm.

2.1 Simulations settings

The first improvement added to the basic version of the MCMB-A algorithm is the option of *sample spacing*. Indeed, as underlined several times in the papers studied, the chain generated from the algorithm follows an autoregressive process and is subject to a risk of high autocorrelation. This can be the origin of a loss of efficiency. The A-transformation is an attempt to overcome this issue. Sample spacing is an alternative strategy and consists in selecting elements of the chain only each time a given number of iterations have been completed. As autocorrelation fades over time, non-consecutive elements of the chain are less or not correlated. Sample spacing is a common strategy in the Monte Carlo Markov Chain (MCMC) framework but comes at the price of computational burden, as a significant share of the iterations results are just thrown away.

A second strategy used in order to improve the efficiency of the algorithm is parallelization. This trick relies on the use of the full computational power of the machine. Indeed, modern computers are now equipped with several cores (usually two or four), that can run simultaneously. Parallelization consists in running several operations at the same time on several cores. This strategy can provide huge gains in terms of computation speed. However, a drawback in our setting is that the algorithm is inherently sequential (each operation can be performed only after the previous one has been completed). As a result, parallelization here implies a slight tweaking of the baseline algorithm, as the elements of the estimated parameters are updated in groups, instead of sequentially. A full description of the tweaked algorithm is provided in the appendix. In the end, parallelization speed up the process, at the cost of a loss of efficiency because of the group updating. UNCLEAR HERE IF THE PROPERTIES OF THE ESTIMATOR ARE STILL THE SAME

- Length and coverage as benchmark for comparison ;
- Autocorrelograms and A-transformation (and sample spacing);

2.2 Simulations results

3 Appendix

3.1 Proof of lemma left to the reader

We give the details for some parts of proofs not detailed in the paper.

Lemma 1, c) We have that $\forall k, n : Z_{n,k} = \sum_{i=1}^{\infty} \Lambda_n^i u_{n,k-i}$. To prove $Z_{n,k}Z'_{n,k} \xrightarrow{\mathbb{P}} \Sigma_n$, we use the formulas :

$$Z_{n,k}Z'_{n,k} = \sum_{i=1}^{\infty} \Lambda_n^i u_{n,k-i}u'_{n,k-i}(\Lambda_n^i)' + \sum_{i=1}^{\infty} \Lambda_n^i u_{n,k-i}u'_{n,k-j}(\Lambda_n^j)' \tag{9}$$

$$\frac{1}{m}\sum_{k=1}^{m} Z_{n,k}Z'_{n,k} - \Sigma = \frac{1}{m}\sum_{k=1}^{m} Z_{n,k}Z'_{n,k} - \Sigma_n + (\Sigma_n - \Sigma) \tag{10}$$

and the fact that $\Sigma_n = \mathbb{E}\,||Z_{n,k}|\,|^2$.

Subtracting Σ_n in (9), summing over m and dividing by m yields :

$$\begin{aligned}\frac{1}{m}\left(\sum_{k=1}^{m} Z_{n,k}Z'_{n,k} - \Sigma_n\right) =& \sum_{i=1}^{\infty}\left(\frac{1}{m}\sum_{k=1}^{m} \Lambda_n^i u_{n,k-i}u'_{n,k-i}(\Lambda_n^i)'\right) \\ &+ \sum_{i=1}^{\infty}\left(\frac{1}{m}\sum_{k=1}^{m} \Lambda_n^i u_{n,k-i}u'_{n,k-j}(\Lambda_n^j)'\right) - \frac{1}{m}\sum_{k=1}^{m}\Sigma_n\end{aligned} \tag{11}$$

The triangle inequality applied to the $(1+\epsilon)$th moment of the expression above gives :

$$\begin{aligned}\mathbb{E}\left|\left|\frac{1}{m}\left(\sum_{k=1}^{m} Z_{n,k}Z'_{n,k} - \Sigma_n\right)\right|\right|^{1+\epsilon} &\le \frac{1}{m^{\epsilon}}\sum_{i=1}^{\infty}\gamma^{2i}\sup_n \mathbb{E}\,||u_{n,k-i}||^{2+\epsilon} + \frac{1}{m^{\epsilon}}\mathbb{E}\left(\left|\left|\mathbb{E}\,||Z_{n,k}||^2\right|\right|^{1+\epsilon}\right) \\ &\le C_3 m^{-\epsilon}\end{aligned} \tag{12}$$

where we used that for each n, $u_{n,k}$ are i.i.d random variables such that $\forall i \neq j$, $\mathbb{E}\left|\left|\Lambda_n^i u_{n,k-i}u'_{n,k-j}(\Lambda_n^j)'\right|\right|^{1+\epsilon} = 0$ and :

$$\begin{aligned}\mathbb{E}\left|\left|\sum_{i=1}^{\infty}\left(\frac{1}{m}\sum_{k=1}^{m} \Lambda_n^i u_{n,k-i}u'_{n,k-j}(\Lambda_n^j)'\right)\right|\right|^{1+\epsilon} &\le \sum_{i=1}^{\infty}\frac{1}{m^{\epsilon+1}}\sum_{k=1}^{m}\mathbb{E}\,\left|\left|\Lambda_n^i u_{n,k-i}u'_{n,k-j}\right|\right|^{\epsilon+1} \\ &\le \sum_{i=1}^{\infty}\frac{1}{m^{\epsilon}}\mathbb{E}\,||(XX')_i||^{\epsilon+1}, \text{ with } X_i = \Lambda_n^i u_{n,k-i}\end{aligned} \tag{13}$$

The upper bound in a) is due to the definition of operator-norm on matrices and maximum eigenvalue (σ_{max}). For a matrix $A \in \mathbb{R}^{n\times n}$ and $x \in \mathbb{R}^n$,

$$\sigma_{max}(A) := \max_{x\neq 0}\frac{||Ax||}{||x||}$$

So that, $\forall x \in \mathbb{R}^n : ||Ax|| \le \sigma_{max}(A)||x||$ (which make the summation of ρ out of the expectation). Now, by using the property of matrix norms ($\forall A, B, ||AB|| \le ||A||||B||$) and the fact that X' has the same eigenvalues that X, we obtain :

$$\mathbb{E}\left|\left|(XX')_i\right|\right|^{\epsilon+1} \leq \rho^{2(1+\epsilon)}\mathbb{E}\left|\left|u_{n,k-i}\right|\right|^{2(1+\epsilon)}$$

And thus,

$$\mathbb{E}\left|\left|\sum_{i=1}^{\infty}\left(\frac{1}{m}\sum_{k=1}^{m}\Lambda_n^i u_{n,k-i}u'_{n,k-j}(\Lambda_n^j)'\right)\right|\right|^{1+\epsilon} \leq \frac{1}{m^{\epsilon}}\sum_{i=1}^{\infty}\rho^{2(1+\epsilon)}\mathbb{E}\left|\left|u_{n,k-i}\right|\right|^{2(1+\epsilon)}$$

Equation (12) follows with $\gamma = \rho^{(1+\epsilon)} < 1$ and a well-chosen $\epsilon' = 2\epsilon > 0$ such that $\sup_n \mathbb{E}\left|\left|u_{n,k}\right|\right|^{2+\epsilon} < \infty$.

By the Markov inequality :

$$\mathbb{P}\left(\left|\left|\frac{1}{m}\left(\sum_{k=1}^{m}Z_{n,k}Z'_{n,k} - \Sigma_n\right)\right|\right| > \delta\right) \leq \frac{C_3\delta^{1+\epsilon}}{m^{\epsilon}}$$

for any $\delta > 0$ and some constant C_3.

Let us denote H_n the variance-covariance matrix of $u_{n,k}$. Now, from (P3) and the definiton of the AR(1) process $Z_{n,k}$, we have :

$$plim \quad \Sigma_n = plim \quad \Lambda_n\Sigma_n\Lambda'_n + \quad plim \quad H_n$$

$$\Longleftrightarrow plim \quad \Sigma_n - \quad \Lambda_n\Sigma_n\Lambda'_n = plim \quad H_n$$

since $Z_{n,k-1}$ converges in distribution towards a normal vector (see b) and ($X \sim \mathcal{N}(0,\Sigma)$ and $Y = AX) \Longrightarrow Y \sim \mathcal{N}(0, A\Sigma A^T)$.

Finally, since $\text{plim}H_n = \Sigma - \Lambda\Sigma\Lambda'$ and $\Lambda_n \to \Lambda$, it must be by identification that $\Sigma_n \overset{\mathbb{P}}{\to} \Sigma$. Combining this result with equation (10) completes the proof of c).

Lemma 1, c) To prove the convergence of $Z_{n,k}$ towards a $\mathcal{N}(0,\Sigma)$, let us write the characteristic function and find its limit when $n \to \infty$. Since for all n, the $(u_k)_{k\in\mathbb{Z}}$ are i.i.d and tends towards a $\mathcal{N}(0,H), \forall\lambda \in \mathbb{R}^p$:

$$\lim_{n\to\infty}\mathbb{E}\left[e^{i\langle(\sum_{i=1}^{\infty}\Lambda_n^i)\lambda, u_{n,k-i}\rangle}\right] = \exp\left\{i\langle\lambda, \left(\sum_{i=1}^{\infty}\Lambda_n^i\right)0_p\rangle - \frac{1}{2}\lambda'\left(\lim_{n\to\infty}\sum_{i=1}^{\infty}\Lambda_n^i\right)H\left(\lim_{n\to\infty}\sum_{i=1}^{\infty}\Lambda_n^i\right)'\lambda\right\} \tag{14}$$

Now we have :

$$\begin{aligned}
\lim_{n\to\infty}\left(\sum_{i=1}^{\infty}\Lambda_n^i\right)H\left(\sum_{i=1}^{\infty}\Lambda_n^i\right)' &= \left(\sum_{i=1}^{\infty}\Lambda^i\right)\Sigma\left(\sum_{i=1}^{\infty}\Lambda'^i\right)-\left(\sum_{i=1}^{\infty}\Lambda^i\right)\Lambda\Sigma\Lambda'\left(\sum_{i=1}^{\infty}\Lambda'^i\right)\\
&=\sum_{i=0}^{\infty}\Lambda^i\Sigma\Lambda'^i-\Sigma-\sum_{i=1}^{\infty}\Lambda^i\Sigma-\Sigma\sum_{i=i}^{\infty}\Lambda'^i+2\sum_{k\neq k',k,k'\geq 0}\Lambda^k\Sigma\Lambda'^k\\
&-\sum_{i=0}^{\infty}\Lambda^{i+1}\Sigma\Lambda'^{i+1}-2\sum_{k\neq k',k,k'\geq 1}\Lambda^k\Sigma\Lambda'^k+\Lambda\Sigma\Lambda'+\sum_{i=1}^{\infty}\Lambda^{i+1}\Sigma\Lambda'+\Lambda\Sigma\sum_{i=1}^{\infty}\Lambda'^{i+1}\\
&=\ldots
\end{aligned}\tag{15}$$

Try it another way using that :

$$\sum_{i=0}^{\infty}\Lambda^i=(I-\Lambda)^{-1},\ \text{because } \sigma_{max}(\Lambda)\leq\rho<1$$

And thus :

$$\sum_{i=1}^{\infty}\Lambda^i=(I-\Lambda)^{-1}-I$$

CA REVIENT A FAIRE CE QU'IL Y A AU-DESSUS MAIS JE TROUVE TJRS PAS LA BONNE VARIANCE. JE RETOMBE SUR : $\Sigma-\Lambda\Sigma\Lambda'$

3.2 Time saving strategy of MCMB for quantile regression

Let's notice that:

$$\mathbb{1}\{x_{i,j}(z_i-b)<0\}=\frac{x_{i,j}(z_i-b)-|x_{i,j}||z_i-b|}{2x_{i,j}(z_i-b)}$$

So,

$$\begin{aligned}
&\sum_{i=1}^{n+1}(z_i-b)x_{i,j}\tau-(z_i-b)x_{i,j}\mathbb{1}\{x_{i,j}(z_i-b)<0\}\\
=&\sum_{i=1}^{n+1}(z_i-b)x_{i,j}\tau-(z_i-b)x_{i,j}\frac{x_{i,j}(z_i-b)-|x_{i,j}||z_i-b|}{2x_{i,j}(z_i-b)}\\
=&\sum_{i=1}^{n+1}(z_i-b)x_{i,j}\tau-\frac{x_{i,j}^2(z_i-b)^2}{2x_{i,j}(z_i-b)}+\frac{x_{i,j}(z_i-b)|x_{i,j}||z_i-b|}{2x_{i,j}(z_i-b)}\\
=&\sum_{i=1}^{n+1}(z_i-b)x_{i,j}\tau-\frac{x_{i,j}(z_i-b)}{2}+\frac{|x_{i,j}||z_i-b|}{2}\\
=&\sum_{i=1}^{n+1}(z_i-b)x_{i,j}(\tau-\frac{1}{2})+\frac{|x_{i,j}||z_i-b|}{2}\\
=&\frac{1}{2}\sum_{i=1}^{n+1}|x_{i,j}||z_i+b|-(z_i-b)x_{i,j}(2\tau-1)\\
\Leftrightarrow&\sum_{i=1}^{n+1}\left(|z_i-b||x_{i,j}|-(2\tau-1)(z_i-b)x_{i,j}\right) \qquad \text{in our setting}
\end{aligned}$$

Second step:

$$
\begin{aligned}
&\sum_{i=1}^{n+1} (|z_i - b||x_{i,j}| - (2\tau - 1)(z_i - b)x_{i,j})) \\
=&\sum_{i=1}^{n+1} |x_{i,j}| \left(|z_i - b| - (2\tau - 1)(z_i - b)\frac{x_{i,j}}{|x_{i,j}|} \right) \\
=&\sum_{i=1}^{n+1} |x_{i,j}| \left(|z_i - b| - (2\tau - 1)(z_i - b)\frac{\sum_{k=1}^{n+1} x_{k,j}}{\sum_{k=1}^{n+1} |x_{k,j}|} + (2\tau - 1)(z_i - b)\left[\frac{\sum_{i=1}^{n+1} x_{i,j}}{\sum_{i=1}^{n+1} |x_{i,j}|} - \frac{x_{i,j}}{|x_{i,j}|}\right] \right) \\
=&\sum_{i=1}^{n+1} |x_{i,j}| \left(|z_i - b| - (2\tau - 1)(z_i - b)\frac{\sum_{k=1}^{n+1} x_{k,j}}{\sum_{k=1}^{n+1} |x_{k,j}|} + (2\tau - 1)z_i\left[\frac{\sum_{k=1}^{n+1} x_{k,j}}{\sum_{k=1}^{n+1} |x_{k,j}|} - \frac{x_{i,j}}{|x_{i,j}|}\right] \right) \\
&- (2\tau - 1)b\sum_{i=1}^{n+1} |x_{i,j}| \left[\frac{\sum_{i=1}^{n+1} x_{i,j}}{\sum_{i=1}^{n+1} |x_{i,j}|} - \frac{x_{i,j}}{|x_{i,j}|}\right] \\
=&\sum_{i=1}^{n+1} |x_{i,j}| \left(|z_i - b| - (2\tau - 1)(z_i - b)\frac{\sum_{k=1}^{n+1} x_{k,j}}{\sum_{k=1}^{n+1} |x_{k,j}|} + (2\tau - 1)z_i\left[\frac{\sum_{k=1}^{n+1} x_{k,j}}{\sum_{k=1}^{n+1} |x_{k,j}|} - \frac{x_{i,j}}{|x_{i,j}|}\right] \right) \\
&- (2\tau - 1)b\left[\frac{\sum_{k=1}^{n+1} x_{k,j}\sum_{i=1}^{n+1} |x_{i,j}|}{\sum_{i=1}^{n+1} |x_{k,j}|} - \sum_{i=1}^{n+1} |x_{i,j}|\frac{x_{i,j}}{|x_{i,j}|}\right] \\
=&\sum_{i=1}^{n+1} |x_{i,j}| \left(|z_i - b| - (2\tau - 1)(z_i - b)\frac{\sum_{k=1}^{n+1} x_{k,j}}{\sum_{k=1}^{n+1} |x_{k,j}|} + (2\tau - 1)z_i\left[\frac{\sum_{k=1}^{n+1} x_{k,j}}{\sum_{k=1}^{n+1} |x_{k,j}|} - \frac{x_{i,j}}{|x_{i,j}|}\right] \right) \\
&- (2\tau - 1)b\left[\sum_{i=1}^{n+1} x_{i,j} - \sum_{i=1}^{n+1} x_{i,j}\right] \\
=&\sum_{i=1}^{n+1} |x_{i,j}| \left(|z_i - b| - (2\tau - 1)(z_i - b)\frac{\sum_{k=1}^{n+1} x_{k,j}}{\sum_{k=1}^{n+1} |x_{k,j}|} + (2\tau - 1)z_i\left[\frac{\sum_{k=1}^{n+1} x_{k,j}}{\sum_{k=1}^{n+1} |x_{k,j}|} - \frac{x_{i,j}}{|x_{i,j}|}\right] \right) \\
=&\sum_{i=1}^{n+1} |x_{i,j}| \left(|z_i - b| - (2(0.5 + (\tau - 0.5)\frac{\sum_{i=1}^{n+1} x_{i,j}}{\sum_{i=1}^{n+1} |x_{i,j}|}) - 1)(z_i - b) + c(x_{i,j}, z_i) \right) \\
=&\sum_{i=1}^{n+1} |x_{i,j}| (|z_i - b| - (2\tau^* - 1)(z_i - b) + c(x_{i,j}, z_i)) \\
=&\sum_{i=1}^{n+1} |x_{i,j}| (|z_i - b| - (2\tau^* - 1)(z_i - b)) + \sum_{i=1}^{n+1} |x_{i,j}|c(x_{i,j}, z_i) \\
=&\sum_{i=1}^{n+1} |x_{i,j}|\rho_{\tau^*}(z_i - b) + \sum_{i=1}^{n+1} |x_{i,j}|c(x_{i,j}, z_i)
\end{aligned}
$$

with $c(x_{i,j}, z_i) = (2\tau - 1)z_i\left[\frac{\sum_{k=1}^{n+1} x_{k,j}}{\sum_{k=1}^{n+1} |x_{k,j}|} - \frac{x_{i,j}}{|x_{i,j}|}\right]$

3.3 MCMB algorithm with parallelization

with c the number of cores, and $\lfloor x \rfloor$ the integer part of x.

Algorithm 2 MCMB with parallelization

```
1: procedure INITIALIZE
2:     $\hat{\theta}^{(0)} \leftarrow \hat{\theta}$ and $k \leftarrow 1$
3: *top*:
4:     for each $J \in \{\{1, \dots, c\}, \{c+1, \dots, 2c\}, \dots, \{\lfloor p/c \rfloor c + 1, \dots, p\}\}$ do
5:         Draw a bootstrap sample $\{z_{1J}^{*(k)}, \dots, z_{nJ}^{*(k)}\}$ from $\{z_1, \dots, z_n\}$
6:     for each $j \in J$ do
7:         Solve for $\hat{\theta}_j^{(k)}$ from $S_J j \left(Y, \hat{\theta}_1^{(k)}, \dots, \hat{\theta}_{J1-1}^{(k)}, \hat{\theta}_{J1}^{(k-1)}, \dots, \hat{\theta}_{Jj-1}^{(k-1)}, \hat{\theta}_{Jj}^{(k)}, \hat{\theta}_{Jj+1}^{(k-1)}, \dots, \hat{\theta}_p^{(k-1)}\right) =$
   $S_J j^{*(k)}$,
8:         where $S_J j$ is the $Jj$-th component of $S(Y, \theta)$, and $S_J j^{*(k)}$ is the $Jj$-th component of
9:         $\sum_{i=1}^{n} a_i z_{iJ}^{*(k)}$
10:    if $k > \overline{K}$ then Stop
11:    $k \leftarrow k + 1$
12:    go to *top*.
```