A two-stage maximum entropy approach for time series regression[1]

Pedro Macedo

Center for Research & Development in Mathematics and Applications, Department of Mathematics, University of Aveiro, 3810-193 Aveiro, Portugal (pmacedo@ua.pt)

Abstract

The maximum entropy bootstrap for time series is a technique that creates a large number of replicates, as elements of an ensemble, for inference purposes, which satisfies the ergodic and the central limit theorems. As an alternative to the use of traditional techniques, this work proposes generalized maximum entropy for the estimation of parameters in all the replicated models. An empirical application and a simulated example illustrate the advantages of this two-stage maximum entropy approach for time series regression modeling, where maximum entropy is used both in data replication and in parameter estimation.

Keywords: bootstrap, ill-conditioned models, info-metrics, time series regression.
MSC 2020: 62M10, 91B84, 94A17

[1]Final version of this paper is published in **Communications in Statistics – Simulation and Computation, 53(1), 518-528, 2024** (`https://doi.org/10.1080/03610918.2022.2057540`).

1 Introduction

The maximum entropy bootstrap for time series (Vinod, 2004, 2006), with the corresponding *meboot* package in R (Vinod and López-de-Lacalle, 2009), has become popular in the last decade. It is a powerful technique that allows statistical formulations free of restrictive and unnecessary assumptions, by avoiding all structural changes and unit root type testing, and all the usual shape-destroying transformations like detrending or differencing to achieve stationarity. The technique creates a large number of replicates, as elements of an ensemble, which retain the shape of the original time series, as well as the time dependence structure of the autocorrelation and the partial autocorrelation functions. However, traditional functions from R for parameter estimation (e.g., the *lm* and *dynlm* functions, both currently using the QR decomposition) that are usually used to obtain estimates for a predefined parameter of interest may compromise inference analysis in ill-conditioned models (i.e., models affected by collinearity[2]) due to the instability of the estimates. Since ill-conditioning affects the precision of the estimates (inflating the variance and possibly affecting its signs) and the *meboot* provides a large number of replicates allowing to compute confidence intervals, this means that wider confidence intervals for the parameters (e.g., using the percentile method) may be obtained if ill-conditioned models are generated, and the null hypothesis of the corresponding parameter is equal to zero is not rejected more frequently. Thus, a stable estimation procedure should be considered as an alternative to the use of traditional techniques.

The generalized maximum entropy (GME) estimator proposed by Golan et al. (1996), as the name itself indicates, is a generalization of the maximum entropy (ME) principle established by Jaynes (1957a,b), based on Shannon (1948) entropy, and it can be seen as a particular case of the generalized cross entropy (GCE) estimator. These and other information-theoretic estimation techniques were recently embedded into a new area of research entitled info-metrics, a research area at the intersection of statistics, computer science and decision theory (Golan, 2018). The advantages of generalized maximum/cross entropy estimation in regression analysis are well-known: it represents a stable estimation procedure

[2]See Belsley et al. (2004, pp. 85–98) for an important discussion regarding this notion of collinearity.

when the design matrix is ill-conditioned (collinearity problems), when the number of unknown parameters to be estimated exceeds the number of observations (under-determined models), and when only samples of small size are available (micronumerosity problems). Some of these advantages are particularly attractive in time series regression analysis and this is the main reason to propose a two-stage maximum entropy approach in this work, where the second stage consists of the estimation of parameters in all the replicated models obtained by *meboot* (considered here as the first stage).

The remaining of the paper is organized as follows: in Section 2, the techniques are briefly presented. An empirical application and a simulated example are discussed in Section 3. Some conclusions and topics for future research are given in Section 4.

2 Some background on the techniques

2.1 Maximum entropy bootstrap

The maximum entropy bootstrap for dependent time series was firstly proposed by Vinod (2004). Later, Vinod (2006) simplifies the algorithm, extends it to panel data, and discusses underlying assumptions and properties. Its implementation in R (*meboot* package) was carried out by Vinod and López-de-Lacalle (2009). Following Vinod (2006), pp. 959–963, the *meboot* algorithm for a random replicate of a time series x_t, $t = 1, 2, \ldots, T$, is briefly presented next for reader's convenience.

1. Some plausible limits (lower and upper) for the time series are established by extrapolation and specified as $x_t \in [x_{LO}, x_{UP}]$. Usually, a trimmed mean of consecutive distances is used for this purpose: (1) absolute distances between consecutive data points, $d_t = |x_t - x_{t-1}|$, $t = 2, 3, \ldots, T$, are computed; (2) the absolute value of a $n\%$ trimmed mean of these d_t distances, denoted by $d_{trm,n}$, is obtained; (3) finally, x_{LO} will be the minimum value of the time series minus $d_{trm,n}$, and x_{UP} will be the maximum value of the time series plus $d_{trm,n}$. Additionally, a $(T \times 2)$ sorting matrix S_1 is created,

with the index set $T' = \{1, 2, \ldots, T\}$ in the first column and the time series x_t in the second column.

2. The original data is sorted in increasing order to create order statistics, $x_{(t)}$, and the ordering index vector is stored. This is accomplished by ordering matrix S_1 in increasing order with respect to the values in its second column, while carrying along the values in its first column. This yields the order statistics, $x_{(t)}$, in the second column and a vector I_{rev} of sorted T' in the first column. Next, are defined $z_0 = x_{LO}$, $z_T = x_{UP}$, $z_t = 0.5(x_{(t)} + x_{(t+1)})$, $t = 1, 2, \ldots, T-1$, and half-open intervals, $I_t = (z_{t-1}, z_t]$, of points around each $x_{(t)}$ from which the elements of the ensemble are selected. There are T intervals, each one containing an $x_{(t)}$ with probability $1/T$. Some notes:

    - *meboot* ensures the mass-preserving constraint by giving an equal probability to each half-open interval, I_t, of being included in the resample;
    - let f be the density function of x_t. The mean-preserving constraint on order 1 moments of f is $\sum_{t=1}^{T} x_t = \sum_{t=1}^{T} x_{(t)} = \sum_{t=1}^{T} m_t$, where m_t represents the mean of f within the interval I_t. To satisfy this constraint, *meboot* requires that the mean m_t in the interval I_t is equal to a weighted sum of the order statistic $x_{(t)}$, namely

$$\begin{gathered} f(x) = \frac{1}{z_1 - z_0}, x \in I_1, \\ m_1 = 0.75x_{(1)} + 0.25x_{(2)}; \end{gathered} \tag{1}$$

$$\begin{gathered} f(x) = \frac{1}{z_k - z_{k-1}}, x \in (z_{k-1}, z_k], \\ m_k = 0.25x_{(k-1)} + 0.50x_{(k)} + 0.25x_{(k+1)}, k = 2, 3, \ldots, T-1; \end{gathered} \tag{2}$$

$$\begin{gathered} f(x) = \frac{1}{z_T - z_{T-1}}, x \in I_T, \\ m_T = 0.25x_{(T-1)} + 0.75x_{(T)}. \end{gathered} \tag{3}$$

    Thus, from $x_{(t)}$, the intervals I_t and the means m_t are computed.

3. T uniform pseudorandom numbers, p_s, in the interval $[0, 1]$ are created, and is identified the range $R_t = (t/T, (t+1)/T]$, $t = 0, 1, \ldots, T-1$, wherein each p_s falls.

4. Each R_t is matched with I_t. If $p_s \in R_0$ or if $p_s \in R_{T-1}$, then (1) or (3) are used, respectively. Otherwise, linear interpolation is used to obtain a set of T values, $\{x_{j,t,me}\}$, as the jth resample. As mentioned by Vinod (2006), p. 961, these are the usual quantiles from the inverse cumulative distribution function of the ME density.

5. A $(T \times 2)$ sorting matrix S_2 is created. The T values of the set $\{x_{j,t,me}\}$ for the jth resample obtained in the previous step are sorted in increasing order of magnitude, as $\{x_{j,(t),me}\}$, and inserted in the first column of S_2. The sorted vector I_{rev} of step 2 is inserted in the second column of S_2.

6. Matrix S_2 is sorted with respect to its second column. The jointly sorted values of the first column of S_2, denoted by $\{x_{j,t}\}$, represent the ME resample.

7. Steps 2 to 6 are repeated for a large number of times, $j = 1, 2, \ldots, J$ (in this study are considered $J = 1000$ replications).

Additional details and advantages of the algorithm, including the ones over the traditional bootstrap, can be found in Vinod (2006) and Vinod and López-de-Lacalle (2009).

2.2 Generalized maximum/cross entropy estimation

Considering a linear regression model defined as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{e}, \tag{4}$$

where $\boldsymbol{y}$ denotes a $(N \times 1)$ vector of noisy observations, $\boldsymbol{\beta}$ is a $(K \times 1)$ vector of unknown parameters to be estimated, $\boldsymbol{X}$ is a known $(N \times K)$ design matrix of explanatory variables, and $\boldsymbol{e}$ is the $(N \times 1)$ vector of random errors, Golan et al. (1996) proposed its reformulation as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{Z}\boldsymbol{p} + \boldsymbol{V}\boldsymbol{w}, \tag{5}$$

where

$$\boldsymbol{\beta} = \boldsymbol{Z}\boldsymbol{p} = \begin{bmatrix} \boldsymbol{z}_1' & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{z}_2' & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{z}_K' \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \vdots \\ \boldsymbol{p}_K \end{bmatrix}, \tag{6}$$

with $\boldsymbol{Z}$ a $(K \times KM)$ matrix of support spaces (closed and bounded intervals in which each parameter is restricted to belong) and $\boldsymbol{p}$ a $(KM \times 1)$ vector of unknown probabilities to be estimated, and

$$\boldsymbol{e} = \boldsymbol{V}\boldsymbol{w} = \begin{bmatrix} \boldsymbol{v}_1' & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{v}_2' & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{v}_N' \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_1 \\ \boldsymbol{w}_2 \\ \vdots \\ \boldsymbol{w}_N \end{bmatrix}, \tag{7}$$

with $\boldsymbol{V}$ a $(N \times NJ)$ matrix of support spaces (closed and bounded intervals in which each error is restricted to belong) and $\boldsymbol{w}$ a $(NJ \times 1)$ vector of unknown probabilities to be estimated. In this reformulation, each β_k, $k = 1, 2, \ldots, K$, and each e_n, $n = 1, 2, \ldots, N$, are viewed as expected values of discrete random variables z_k and v_n, respectively, with $M \geq 2$ and $J \geq 2$ possible outcomes, within the lower and upper bounds of the corresponding support spaces.

Thus, considering the linear regression model expressed in (4), the generalized cross entropy (GCE) estimator is given by

$$\underset{\boldsymbol{p},\boldsymbol{w}}{\operatorname{argmin}} \left\{ \boldsymbol{p}' \ln \left(\frac{\boldsymbol{p}}{\boldsymbol{q}_1} \right) + \boldsymbol{w}' \ln \left(\frac{\boldsymbol{w}}{\boldsymbol{q}_2} \right) \right\}, \tag{8}$$

subject to the model (data consistency) constraints,

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{Z}\boldsymbol{p} + \boldsymbol{V}\boldsymbol{w}, \tag{9}$$

the additivity constraints for $\boldsymbol{p}$,

$$\boldsymbol{1}_K = (\boldsymbol{I}_K \otimes \boldsymbol{1}_M')\boldsymbol{p}, \tag{10}$$

and the additivity constraints for $\boldsymbol{w}$,

$$\boldsymbol{1}_N = (\boldsymbol{I}_N \otimes \boldsymbol{1}_J')\boldsymbol{w}, \tag{11}$$

where $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ are vectors with prior information concerning the parameters and the errors of the model, respectively, and $\otimes$ represents the Kronecker product. As mentioned previously, the GCE estimator is identical to the generalized[3] maximum entropy (GME) estimator when the prior information is expressed as a uniform distribution (vectors $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$). However, the GME estimator can be simply established by stipulating the objective function as

$$\underset{\boldsymbol{p},\boldsymbol{w}}{\operatorname{argmax}} \left\{ -\boldsymbol{p}' \ln \boldsymbol{p} - \boldsymbol{w}' \ln \boldsymbol{w} \right\}, \tag{12}$$

with the same above restrictions (9) – (11). Both estimators generate the optimal probability vectors $\widehat{\boldsymbol{p}}$ and $\widehat{\boldsymbol{w}}$ from the previous numerical optimization problems, which can be used to form point estimates of the unknown parameters and the unknown errors, through the reparameterizations defined previously, (6) and (7). The uniqueness of the GCE and GME solutions is demonstrated in Golan et al. (1996), pp. 89–92. For simplicity, this work will just illustrate the use of the GME estimator (i.e., the GCE estimator with uniform priors).

Some concerns regarding the GCE/GME estimation are usually related to the specification of the support spaces, which remains an unsettled issue in terms of a formal statistical procedure. Further research on this topic is needed. Some discussions and guidelines for these choices are provided, for example, in Golan et al. (1996), pp. 137–142, Preckel (2001), Caputo and Paris (2008) and Golan (2018), pp. 262–264 and pp. 380–382. However, mainly motivated by empirical and simulation works, and although there are exceptions in specific scenarios, for practitioners' convenience some general principles are provided next: (1) the number of support points should be equal or greater to two, but it is usually between three and seven because there is likely no significant improvement in the estimation with more points; (2) the supports should be symmetric about zero; (3) the supports should be defined with equally spaced points between the lower and upper bounds; (4) for the error component are usually considered three points in supports and the bounds are usually defined by the three-sigma rule, considering the standard deviation of the noisy observations (observed

[3]The "generalized" derives from the fact that the maximum entropy principle is usually presented as $\underset{\boldsymbol{p}}{\operatorname{argmax}} \{-\boldsymbol{p}' \ln \boldsymbol{p}\}$, also subject to the corresponding model and additivity constraints; see Jaynes (1957a,b).

dependent variable), i.e., $[-3\hat{\sigma}_y, 0, 3\hat{\sigma}_y]$; (5) the bounds and the center of the parameters' supports are problem specific and should be chosen with care. Usually the center is defined at zero if no prior information exists (even about the relevance of the corresponding variable). In a non-consensual way, usually the bounds are defined based on theoretical constraints, data from previous works, confidence intervals with high confidence levels from other estimation techniques. A sensitivity analysis can also be accomplished to evaluate the impact on the estimates by using different supports, or similar to traditional algorithms that provide results for a set of regularization parameters (e.g., from the least absolute shrinkage and selection operator family), an "optimal" support can be identified by the solution that corresponds, for example, to the minimum mean squared error (or any other loss function of interest) obtained by cross-validation.

Additional details on maximum entropy estimation, properties, asymptotic theory, simulation studies and examples can be found in Golan et al. (1996), Mittelhammer et al. (2013), Henderson et al. (2015), Golan (2018) and Macedo (2020).

2.3 Code for maximum entropy bootstrap with GME estimation

Given the theoretical discussion of the GME estimator in Section 2.2, some general steps to implement the estimator are briefly summarized next.

1. The number of observations, N, and variables, K, are stored.

2. The supports for the parameters, $\boldsymbol{z}'_k$, $k = 1, 2, \ldots, K$, are established, including the number of points, M, in each support, and matrix $\boldsymbol{Z}$ is created.

3. The supports for the errors, $\boldsymbol{v}'_n$, $n = 1, 2, \ldots, N$, are established, including the number of points, J, in each support, and matrix $\boldsymbol{V}$ is created.

4. Depending on the programming language or software used: $\boldsymbol{p}$ and $\boldsymbol{w}$ are initialized (usually both as uniform priors); objective function (12), and restrictions (9)–(11) are implemented in the numerical optimization structure; and vector $\widehat{\boldsymbol{p}}$ is obtained.

5. The estimated parameters are computed as $\widehat{\boldsymbol{\beta}} = \boldsymbol{Z}\widehat{\boldsymbol{p}}$.

To illustrate the previous steps, suppose a simple linear regression model with only five observations, $\{(2.8, 6.4), (2.5, 7.7), (3.9, 0.4), (3.1, 4.5), (3.5, 2.9)\}$, and consider the support spaces as $[-1000, 1000]$ and $[-2, 2]$, respectively for all the parameters and all the errors, with $M = 5$ and $J = 3$. Thus, with symmetric supports centered on zero and equally spaced points, the GME estimator is given by

$$\underset{\boldsymbol{p},\boldsymbol{w}}{\operatorname{argmax}} \left\{-\boldsymbol{p}' \ln \boldsymbol{p} - \boldsymbol{w}' \ln \boldsymbol{w}\right\}, \tag{13}$$

subject to the model constraints,

$$\begin{bmatrix} 2.8 \\ 2.5 \\ 3.9 \\ 3.1 \\ 3.5 \end{bmatrix} = \begin{bmatrix} 1 & 6.4 \\ 1 & 7.7 \\ 1 & 0.4 \\ 1 & 4.5 \\ 1 & 2.9 \end{bmatrix} \times \begin{bmatrix} -1000 & -500 & 0 & 500 & 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1000 & -500 & 0 & 500 & 1000 \end{bmatrix} \times$$

$$\times \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{15} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{25} \end{bmatrix} + \begin{bmatrix} -2 & 0 & 2 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 & 0 & 2 & \dots & 0 & 0 & 0 \\ & & & & & & \ddots & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & -2 & 0 & 2 \end{bmatrix} \times \begin{bmatrix} w_{11} \\ w_{12} \\ w_{13} \\ w_{21} \\ w_{22} \\ w_{23} \\ \vdots \\ w_{51} \\ w_{52} \\ w_{53} \end{bmatrix}, \tag{14}$$

the additivity constraints for $\boldsymbol{p}$,

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{15} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{25} \end{bmatrix}, \tag{15}$$

and the additivity constraints for $\boldsymbol{w}$,

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & \dots & 0 & 0 & 0 \\ & & & & & & \ddots & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} w_{11} \\ w_{12} \\ w_{13} \\ w_{21} \\ w_{22} \\ w_{23} \\ \vdots \\ w_{51} \\ w_{52} \\ w_{53} \end{bmatrix}. \tag{16}$$

Using, for example, the *fmincon* nonlinear programming solver from MATLAB (see details below) in the previous step 4, the result[4] is

[4]The vector $\widehat{\boldsymbol{p}}$ is rounded here to four decimals, but the product $\boldsymbol{Z}\widehat{\boldsymbol{p}}$ is computed with the maximum available precision in MATLAB.

$$\begin{bmatrix} \widehat{\beta}_1 \\ \widehat{\beta}_2 \end{bmatrix} = \begin{bmatrix} -1000 & -500 & 0 & 500 & 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1000 & -500 & 0 & 500 & 1000 \end{bmatrix} \times \begin{bmatrix} 0.1984 \\ 0.1992 \\ 0.2000 \\ 0.2008 \\ 0.2016 \\ 0.2001 \\ 0.2000 \\ 0.2000 \\ 0.2000 \\ 0.1999 \end{bmatrix}$$

$$\approx \begin{bmatrix} 4.0022 \\ -0.1923 \end{bmatrix}. \tag{17}$$

Both procedures can be implemented in R or Python, two very attractive languages for statistics. However, using resources already available in R (Vinod and López-de-Lacalle, 2009) and MATLAB (Macedo, 2017), the maximum entropy bootstrap with GME estimation can be computed straightforwardly in MATLAB. The matrix with the J replications from *meboot* in R are imported, and a simple loop syntax in MATLAB is implemented to obtain the estimates of parameters for each replicated model, using the matrix structure of the GME estimator discussed previously and available in Macedo (2017). Then, confidence intervals (e.g., percentile method) are easily computed, along with some possible specific statistics. Given the large number of possible theoretical configurations in time series regression modeling (including different variables, lags, etc.), only a very general code structure to be implemented in MATLAB is provided next.[5] Additional details are available upon request to the author.

[5]The code is provided with absolutely no warranty. Its users assume all the responsibility when using it.

Code Structure: Maximum entropy bootstrap with GME estimation.

```
/* The package writexl in R and the function readmatrix in MATLAB may be useful to
   exchange data.                                                               */
```

Input: matrix with replications of the original time series from *meboot* in R.

Data: dimension of the series; number of replications/models; number of parameters to estimate; matrix of zeros (e.g., with name *estimates*) to collect the estimates for each of the models.

for $r = 1$ **to** *number of replications/models*

(Inside this loop should exist the)

- theoretical configuration of the time series regression model to use the data from the matrix in inputs, using Y for the response and X for the design matrix;
- number of rows (n) and columns (k) of the design matrix (e.g., $[n, k] = size(X);$);
- matrix structure of the GME estimator; lines 71-117 (from *intg* to *b1*) in Macedo (2017);
- supports for the parameters in *intg* accordingly to the theoretical configuration;
- matrix *estimates* to be updated at each iteration.

end

Output: compute confidence intervals (e.g., percentile method) and possible specific statistics from the matrix *estimates* using, for example, the *prctile* function in MATLAB.

```
/* Other changes can be made; e.g., (1) the number of points in the supports,
   which are 5 and 3, by default, in lines 74 (m = 5) and 86 (j = 3); (2) the
   strategy to define the supports for the error component.                     */
```

3 Empirical application and simulated examples

In this section, the empirical application used by Vinod and López-de-Lacalle (2009) is replicated for comparison purposes. Additionally, simulated examples with samples of small size and with different values of condition number[6] are presented to highlight the advantages of GME, when compared to the use of the *lm* and *dynlm* functions in R, in the estimation of all the models' parameters generated by the maximum entropy bootstrap.

3.1 Empirical application

The example used by Vinod and López-de-Lacalle (2009) is described by

$$c_t = \beta_1 + \beta_2\, c_{t-1} + \beta_3\, d_{t-1} + e_t, \tag{18}$$

where c represents the logarithm of the United States (US) consumption, d represents the logarithm of the disposable income, and t is the time period, which is from 1948 to 1998. The data is available in the R package as "USconsum". Since the purpose is to discuss a Keynesian consumption function on the basis of the hypothesis of permanent income, the hypothesis test of interest is

$$H_0 : \beta_3 = 0 \quad vs. \quad H_1 : \beta_3 \neq 0. \tag{19}$$

Table 1 presents the results provided by maximum entropy bootstrap (*meboot*) with *dynlm* function, considering 1000 replications of the original series, and by maximum entropy bootstrap with GME estimation (*gmeboot*). The GME estimator is performed with two different wide supports for all the parameters, considering minimal prior information about the problem: $[-100, 100]$ in $gmeboot_{100}$; and $[-1000, 1000]$ in $gmeboot_{1000}$. The supports are defined symmetric about zero, with five equally spaced points between the lower and upper bounds. For each error support (symmetric about zero, with three equally spaced points) is used the three-sigma rule, considering the standard deviation of the noisy observations; see

[6]Ratio of the largest with the smallest singular value of the design matrix; $cond(\boldsymbol{X})$.

Golan (2018), p. 380. The highest density regions (HDR) were adopted here to compute the confidence intervals (Hyndman, 1996). The estimate in Table 1 represents the median of the 1000 estimates obtained for β_3. All values are rounded to three decimals.[7]

Table 1: Results for meboot and gmeboot.

	meboot	$gmeboot_{100}$	$gmeboot_{1000}$
Estimate	0.126	0.128	0.126
$\mathrm{CI}_{99\%}(\beta_3)$	$(-0.156, 0.471)$	$(-0.111, 0.486)$	$(-0.114, 0.486)$
$\mathrm{CI}_{95\%}(\beta_3)$	$(-0.073, 0.340)$	$(-0.052, 0.355)$	$(-0.053, 0.352)$
$\mathrm{CI}_{90\%}(\beta_3)$	$(-0.041, 0.297)$	$(-0.028, 0.313)$	$(-0.030, 0.312)$

The results are similar between *gmeboot* (maximum entropy bootstrap using GME estimation) and *meboot* (using *dynlm* function from R that currently uses QR decomposition). The condition numbers of the 1000 design matrices, $cond(\boldsymbol{X}_i)$, $i = 1, 2, \ldots, 1000$, are approximately between 33 and 185, with a median of 80, which denotes (relatively) well-conditioned models. The null hypothesis in (19) is not rejected for significance levels less than or equal to 10%, supporting Friedman's permanent income hypothesis.

Figure 1 presents the HDR for the sampling distribution of the estimates of β_3 (with *meboot* on the left and $gmeboot_{1000}$ on the right).

3.2 Simulated examples

Ill-conditioned models (collinearity problems) may occur from the replications of maximum entropy bootstrap. To illustrate the performance of the GME in the estimation of parameters in these empirical scenarios, a time series regression model is defined as

$$y_t = \beta_1 + \beta_2\, x_{1t} + \beta_3\, x_{2t} + \beta_4\, x_{3t} + e_t, \tag{20}$$

[7]The R packages *meboot* (Vinod and López-de-Lacalle, 2009) and *hdrcde* (Hyndman et al., 2018) are used in this work. A general code structure to compute *gmeboot* is available above in Section 2.3.

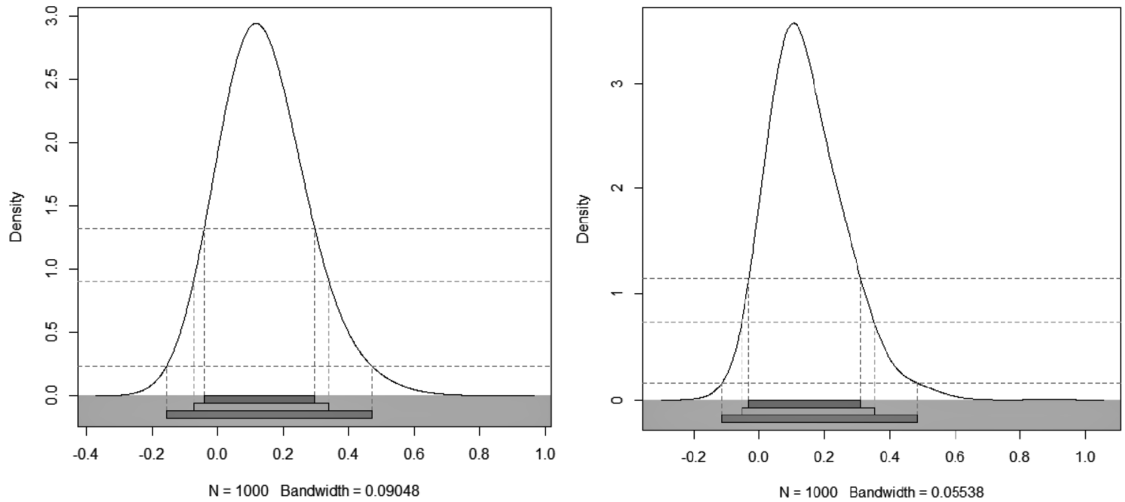

Figure 1: HDR for the sampling distribution of the estimates of β_3.

with $t = 1, 2, \ldots, T$. The model is constructed as follows: $x_1 \sim U(5, 25)$ and $x_2 \sim U(5, 25)$, both ordered; $x_3 = x_2/2 + N(0, \sigma^2)$; $e_t \sim N(0, 1)$; coefficients are defined as $\beta_1 = 0.10$, $\beta_2 = 0.95$, $\beta_3 = 0.25$ and $\beta_4 = 0.60$; finally, y_t is obtained accordingly to (20). The above strategy with a specific choice of structure and distributions represents a choice for simplicity over generality; it still remains valid to illustrate many other possible theoretical configurations. The design matrix is constructed in such a way that, decreasing the variance of the normal distribution used in the construction of x_3, the condition number increases (as expected) and the problem becomes increasingly ill-conditioned.[8] Three different sizes for time series are considered, namely $T = 20, 50, 100$. The mean squared error loss (MSEL), where $\text{SEL}(\widehat{\boldsymbol{\beta}}) = (\|\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\|_2)^2$, is the measure used to evaluate the performance of the estimators in 1000 trials.

Tables 2, 3 and 4 present the MSEL values from *lm* and *dynlm* functions in R (both currently using QR decomposition; the results are equal), and the results from GME, where the estimator is performed with two different supports for all the parameters: $[-10, 10]$ in GME_{10} and $[-100, 100]$ in GME_{100}. As before, all these supports are centered on zero with

[8]The traditional singular value decomposition to define a design matrix with a specific condition number is avoided here, given the usual characteristics of sample data in time series analysis.

five equally spaced points between the lower and upper bounds, and for each error support (centered on zero with three equally spaced points) is used the three-sigma rule, considering the standard deviation of the noisy observations, i.e., $[-3\hat{\sigma}_y, 0, 3\hat{\sigma}_y]$; see Golan (2018), p. 380. For a specific choice of σ^2 in the construction of x_3, the median (minimum and maximum) of the 1000 values of condition number is given by med (min and max) $cond(\boldsymbol{X})$. The values of $cond(\boldsymbol{X})$ are rounded to the nearest integer and the values of MSEL are rounded to three decimals.

Table 2: MSEL from *lm*/*dynlm* and GME ($T = 20$).

	lm/*dynlm*	GME_{10}	GME_{100}
med $cond(\boldsymbol{X}) \approx 82$ (min $cond(\boldsymbol{X}) \approx 53$; max $cond(\boldsymbol{X}) \approx 351$)	0.909	0.125	0.760
med $cond(\boldsymbol{X}) \approx 3005$ (min $cond(\boldsymbol{X}) \approx 1748$; max $cond(\boldsymbol{X}) \approx 5546$)	856.754	0.228	0.917
med $cond(\boldsymbol{X}) \approx 15033$ (min $cond(\boldsymbol{X}) \approx 8770$; max $cond(\boldsymbol{X}) \approx 30359$)	20778.694	0.232	0.877

Table 3: MSEL from *lm*/*dynlm* and GME ($T = 50$).

	lm/*dynlm*	GME_{10}	GME_{100}
med $cond(\boldsymbol{X}) \approx 79$ (min $cond(\boldsymbol{X}) \approx 57$; max $cond(\boldsymbol{X}) \approx 237$)	0.329	0.097	0.315
med $cond(\boldsymbol{X}) \approx 2798$ (min $cond(\boldsymbol{X}) \approx 1889$; max $cond(\boldsymbol{X}) \approx 4666$)	251.265	0.240	0.830
med $cond(\boldsymbol{X}) \approx 13966$ (min $cond(\boldsymbol{X}) \approx 10303$; max $cond(\boldsymbol{X}) \approx 20722$)	7328.837	0.237	0.476

As expected, the MSEL from *lm*/*dynlm* increases with the increase of $cond(\boldsymbol{X})$, but decreases with the increase of T. On the other hand, the results from GME reveal a remarkable

Table 4: MSEL from *lm*/*dynlm* and GME ($T = 100$).

	lm/*dynlm*	GME_{10}	GME_{100}
med $cond(\boldsymbol{X}) \approx 82$ (min $cond(\boldsymbol{X}) \approx 60$; max $cond(\boldsymbol{X}) \approx 163$)	0.179	0.084	0.175
med $cond(\boldsymbol{X}) \approx 2736$ (min $cond(\boldsymbol{X}) \approx 2185$; max $cond(\boldsymbol{X}) \approx 3621$)	121.211	0.237	0.998
med $cond(\boldsymbol{X}) \approx 13714$ (min $cond(\boldsymbol{X}) \approx 10835$; max $cond(\boldsymbol{X}) \approx 18114$)	3242.319	0.237	0.363

stability of the estimates with the increase of $cond(\boldsymbol{X})$. It is worth noting that regardless the supports considered, reflecting the existence of different levels of prior information about the problem, the GME estimator always has a similar performance or outperforms the QR decomposition currently used in *lm* and *dynlm* functions. These results clearly highlight the advantages of GME to estimate the parameters of all the models obtained from the replications of maximum entropy bootstrap, whether or not the models obtained are ill-conditioned.

Of course, and as mentioned before, the use of the GME estimator has also its difficulties, namely in the choice of the supports for the parameters that is always problem specific and should be carefully chosen in real problems (Golan, 2018). In this simulation, different supports appear to have a negligible impact on the estimates. In real-world scenarios, if prior knowledge does not exist, since increasing the amplitude decreases the impact of the supports, different wider bounds should be considered, and a sensitivity analysis on the estimates should be implemented. The guidelines provided in Section 2.2 may be helpful in solving these possible difficulties.

4 Conclusions

The discussion provided in this work intends to improve inference analysis from maximum entropy bootstrap for time series proposed by Vinod (2004, 2006). Although only the GME estimator is illustrated here, other information-theoretic methods can be implemented; e.g., W-GME from Wu (2009). Additionally, and although other methods can be used to estimate the parameters in the models obtained from the replications of maximum entropy bootstrap (e.g., from the least absolute shrinkage and selection operator family), the purpose of this work is to suggest and illustrate the application of a unique approach, where maximum entropy is used both in data replication and in parameter estimation. Illustrative applications were provided that suggest the GME estimator is competitive with traditional functions from R for parameter estimation in cases where data were reasonably well-conditioned, and exhibited superiority in ill-conditioned scenarios. While promising, the illustrative performance of the two-stage maximum entropy approach is clearly case-specific, and further research on maximum entropy bootstrap with information-theoretic methods for parameter estimation should include the performance in models with different theoretical configurations, possibly including outliers and other error structures.

Acknowledgements

I would like to express my deeply gratitude to the Editor and to the anonymous Referee. They offered extremely valuable suggestions for improvements. This work is supported by The Center for Research and Development in Mathematics and Applications (CIDMA) through the Portuguese Foundation for Science and Technology (FCT – Fundação para a Ciência e a Tecnologia), reference UIDB/04106/2020.

References

Belsley, D. A., Kuh, E., and Welsch, R. E. (2004). *Regression Diagnostics - Identifying Influential Data and Sources of Collinearity.* John Wiley & Sons, Hoboken, New Jersey.

Caputo, M. R. and Paris, Q. (2008). Comparative statics of the generalized maximum entropy estimator of the general linear model. *European Journal of Operational Research*, 185:195–203.

Golan, A. (2018). *Foundations of Info-Metrics: Modeling, Inference, and Imperfect Information*. Oxford University Press, New York.

Golan, A., Judge, G., and Miller, D. (1996). *Maximum Entropy Econometrics: Robust Estimation with Limited Data*. John Wiley & Sons, Chichester.

Henderson, H., Golan, A., and Seabold, S. (2015). Incorporating prior information when true priors are unknown: An information-theoretic approach for increasing efficiency in estimation. *Economics Letters*, 127:1–5.

Hyndman, R., Einbeck, J., and Wand, M. (2018). *hdrcde: Highest Density Regions and Conditional Density Estimation*. https://CRAN.R-project.org/package=hdrcde.

Hyndman, R. J. (1996). Computing and graphing highest density regions. *The American Statistician*, 50(2):120–126.

Jaynes, E. T. (1957a). Information theory and statistical mechanics. *Physical Review*, 106(4):620–630.

Jaynes, E. T. (1957b). Information theory and statistical mechanics. II. *Physical Review*, 108(2):171–190.

Macedo, P. (2017). Ridge regression and generalized maximum entropy: an improved version of the Ridge-GME parameter estimator. *Communications in Statistics - Simulation and Computation*, 46(5):3527–3539.

Macedo, P. (2020). Freedman's paradox: A solution based on normalized entropy. In *Theory and Applications of Time Series Analysis*, Contributions to Statistics, pages 239–252. Springer.

Mittelhammer, R., Cardell, N. S., and Marsh, T. L. (2013). The data-constrained generalized maximum entropy estimator of the GLM: Asymptotic theory and inference. *Entropy*, 15:1756–1775.

Preckel, P. V. (2001). Least squares and entropy: A penalty function perspective. *American Journal of Agricultural Economics*, 83(2):366–377.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

Vinod, H. D. (2004). Ranking mutual funds using unconventional utility theory and stochastic dominance. *Journal of Empirical Finance*, 11:353–377.

Vinod, H. D. (2006). Maximum entropy ensembles for time series inference in economics. *Journal of Asian Economics*, 17(6):955–978.

Vinod, H. D. and López-de-Lacalle, J. (2009). Maximum entropy bootstrap for time series: The meboot r package. *Journal of Statistical Software*, 29(5):1–19.

Wu, X. (2009). A weighted generalized maximum entropy estimator with a data-driven weight. *Entropy*, 11(4):917–930.