Maximum Entropy Bootstrap for Time Series: The meboot R Package

Hrishikesh D. Vinod
Fordham University

Javier López-de-Lacalle
Universidad del País Vasco

Abstract

This introduction to the R package **meboot** is a (slightly) modified version of Vinod and López-de-Lacalle (2009), published in the *Journal of Statistical Software*.

The maximum entropy bootstrap is an algorithm that creates an ensemble for time series inference. Stationarity is not required and the ensemble satisfies the ergodic theorem and the central limit theorem. The **meboot** R package implements such algorithm. This document introduces the procedure and illustrates its scope by means of several guided applications.

Keywords:˜time series, dependent data, bootstrap, R.

1. Introduction

This paper illustrates the use of the **meboot** R package for R (R Development Core Team 2008). The package **meboot** implements the maximum entropy bootstrap algorithm for time series described in Vinod (2004, 2006). The package can be obtained from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=meboot`.

In the traditional theory, an ensemble Ω represents the population from which the observed time series is drawn. The maximum entropy (ME) bootstrap constructs a large number of replicates ($J = 999$, say) as elements of Ω for inference using a seven-step algorithm designed to satisfy the ergodic theorem (the grand mean of all ensembles is close to the sample mean). The algorithm's practical appeal is that it avoids all structural change and unit root type testing involving complicated asymptotics and all shape-destroying transformations like detrending or differencing to achieve stationarity. The constructed ensemble elements retain the basic shape and time dependence structure of the autocorrelation function (ACF) and the partial autocorrelation function (PACF) of the original time series.

This discussion collects relevant portions of Vinod (2004, 2006) as templates for users of the **meboot** package. Let us begin with some motivation. Wiener, Kolmogorov and Khintchine (WKK, Wiener 1930; Kolmogorov 1931; Khintchine 1934), among others, developed the stationary model in the 1930's where the data x_t arise from the Ω mentioned above. Stationary time series are integrated of order zero, I(0). Many real world applications involve a mixture of I(0) and nonstationary I(d) series, where the order of integration d can be different for different series and even fractional, and where the stationarity assumptions are difficult to verify. The situation is much worse in the presence of regime switching structural changes and other jump discontinuities occurring at arbitrary times. The WKK theory mostly needs the zero

memory I(0) white noise type processes, where some WKK results are true only for circular processes, implying that we can go back in history, (e.g., undo the US Securities and Exchange Commission, the Federal Communications Commission, or go back to horse and buggy, pre 9/11 days, etc.). Irreversibility is an important property of most economic time series, making the assumption of zero memory I(0) process quite unrealistic. Actually, social science systems are often dynamic, complex and adaptive leading to irreversible, non-stationary and sometimes rather short time series. Hence Economists often need: (i) 'non-standard' Dickey-Fuller type sampling distributions for testing regression coefficients (with severe inference problems for panel data), and (ii) detrending and differencing to convert such series to stationarity. The motivation then is to achieve greater flexibility and realism by avoiding both (i) and (ii).

Vinod (2004, 2006) offers a computer intensive construction of a plausible ensemble created from a density satisfying the maximum entropy principle. The ME bootstrap algorithm uses quantiles $x_{j,t}$ for $j = 1, \ldots, J$ ($J = 999$, say), of the ME density as members of Ω from the inverse of its 'empirical' cumulative distribution function (CDF). The algorithm guarantees the satisfaction of the ergodic theorem (grand mean of all $x_{j,t}$ representing the ensemble average equals the time average of x_t) and the central limit theorem.

Some authors try to bring realism by testing and allowing for finite 'structural changes', often with *ad hoc* tools. However, the notion of infinite memory of the random walk I(1) is unrealistic because the very definitions of economic series (e.g., quality and content of the gross domestic product, names of stocks in the Dow Jones average) change over finite (relatively short) time intervals. Changing definitions are generally not a problem in natural sciences. For example, the definition of water or the height of an ocean wave is unchanged over time.

2. Maximum entropy bootstrap

The bootstrap studies the relation between the sample and the (unknown) population by a comparable relation between the sample at hand and appropriately designed (observable) resamples. If the observed sample is independent and identially distributed (iid), $x_1, ...x_T$ are iid random variables with a common original density: F. The joint density of the sample is given by a T-fold product: F^T. If $\hat{\theta}_T$ estimates a parameter θ, the unknown sampling distribution of $(\hat{\theta}_T - \theta)$ is given by the conditional distribution of its bootstrap version $(\theta^* - \hat{\theta}_T)$, Lahiri (2003). This section describes the ME bootstrap algorithm and indicates how it extends the traditional iid bootstrap to nonstationary dependent data.

2.1. The algorithm

An overview of the steps in Vinod's ME bootstrap algorithm to create a random realization of x_t is provided in this subsection. The reader should consult the toy example of the next subsection for concreteness.

1. Sort the original data in increasing order to create order statistics $x_{(t)}$ and store the ordering index vector.

2. Compute intermediate points $z_t = (x_{(t)} + x_{(t+1)})/2$ for $t = 1, \ldots, T-1$ from the order statistics.

3. Compute the trimmed mean m_{trm} of deviations $x_t - x_{t-1}$ among all consecutive observations. Compute the lower limit for left tail as $z_0 = x_{(1)} - m_{\text{trm}}$ and upper limit for right tail as $z_T = x_{(T)} + m_{\text{trm}}$. These limits become the limiting intermediate points.

4. Compute the mean of the maximum entropy density within each interval such that the 'mean-preserving constraint' (designed to eventually satisfy the ergodic theorem) is satisfied. Interval means are denoted as m_t. The means for the first and the last interval have simpler formulas.

5. Generate random numbers from the $[0, 1]$ uniform interval, compute sample quantiles of the ME density at those points and sort them.

6. Reorder the sorted sample quantiles by using the ordering index of step 1. This recovers the time dependence relationships of the originally observed data.

7. Repeat steps 2 to 6 several times (e.g., 999).

2.2. A toy example

The procedure described above is illustrated with a small example. Let the sequence $x_t = (4, 12, 36, 20, 8)$ be the series of data observed from the period $t = 1$ to $t = 5$ as indicated in the first two columns in Table 1. We jointly sort these two columns on the second column and place the result in the next two columns (Table 1 columns 3 and 4), giving us the ordering index vector in column 3.

Next, the four intermediate points in Column 5 are seen to be simple averages of consecutive order statistics. We need two more (limiting) 'intermediate' points. These are obtained as described in Step 3 above. Using 10% trimming, the limiting intermediate values are $z_0 = -11$ and $z_T = 51$. With these six z_t values we build our five half open intervals: $U(-11, 6] \times U(6, 10] \times U(10, 16] \times U(16, 28] \times U(28, 51]$. The maximum entropy density of the ME bootstrap is defined as the combination of T uniform densities defined over (the support of) T half open intervals.

Time	x_t	Ordering vector	Sorted x_t	Intermediate points	Desired means	Uniform draws	Preliminary values	Final replicate
1	4	1	4	6	5	0.12	5.85	5.85
2	12	5	8	10	8	0.83	6.70	8.90
3	36	2	12	16	13	0.53	8.90	23.95
4	20	4	20	28	22	0.59	10.70	10.70
5	8	3	36		32	0.11	23.95	6.70

Table 1: Example of the ME bootstrap algorithm.

The ME density is shown in Figure˜1 along with the five (half-open) intervals. Note that these intervals join all intermediate points z_t (those in column 5 plus two limiting ones) without gaps.

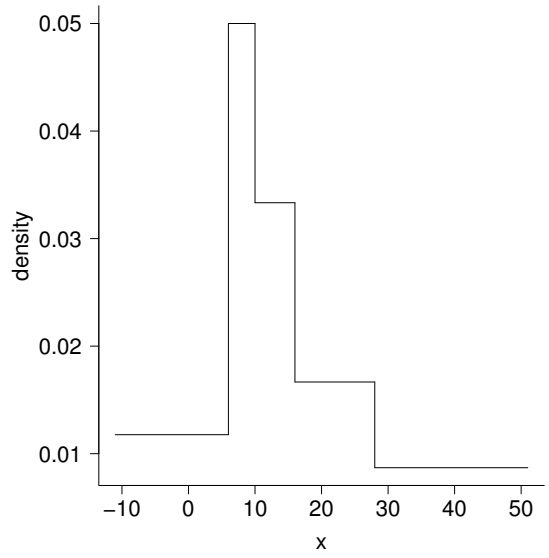

Figure 1: Maximum entropy density for the $x_t = 4, 12, 36, 20, 8$ example.

The uniform densities are also designed to satisfy the 'mean-preserving constraint', by making sure that the interval means for the uniform density, m_t, satisfy the following relations:

$$\begin{aligned} m_1 &= 0.75x_{(1)} + 0.25x_{(2)}\,, \quad \text{for the lowest interval,} \\ m_k &= 0.25x_{(k-1)} + 0.50x_{(k)} + 0.25x_{(k+1)}\,, \quad \text{for } k = 2, \ldots, T-1 \\ m_T &= 0.25x_{(T-1)} + 0.75x_{(T)}\,, \end{aligned}$$

where $x_{(t)}$ are the order statistics. The desired means using these formulas for the toy example are reported in column 6.

Finally, random numbers from the $[0, 1]$ uniform intervals are independently drawn to compute quantiles of the ME density. (See left side plot in Figure˜2.) The ME density quantiles obtained in this way provide a monotonic series. The final replicate is obtained after recovering the original order sorting column 8 according to the index order given in column 3. (See right side plot in Figure˜2.)

2.3. Contrast with traditional iid bootstrap

Singh (1981) used Edgeworth expansions to confirm the superiority of iid boot. He also proved that iid-boot fails for dependent data. See Davison and Hinkley (1997, Chapter˜8) and Lahiri (2003) for more recent results. A modification of the iid boot for stationary m-dependent data called the 'block bootstrap' is extensively discussed by Lahiri (2003). However, if the evolutionary data are non-stationary, one cannot always use 'differencing' operations to render them stationary. The ME bootstrap algorithm is more general, since it does not assume stationarity and does not need possibly 'questionable' differencing operations.

In addition to avoiding stationarity, Vinod (2004, 2006) mentions that it is desirable to avoid the following three properties of traditional iid bootstrap.

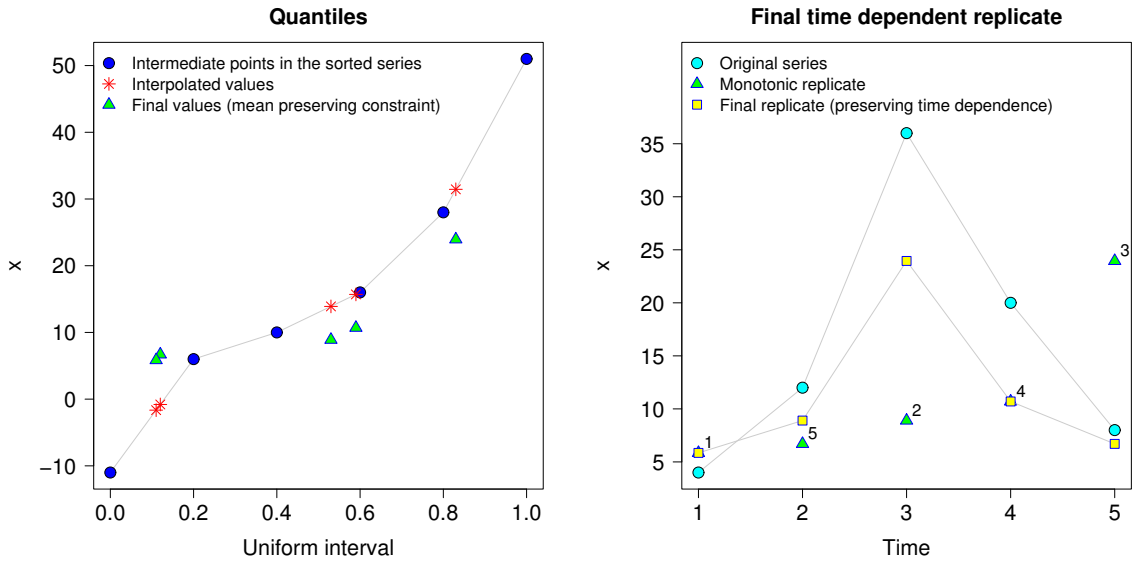

Figure 2: Example of the ME bootstrap algorithm.

- The traditional bootstrap sample obtained from shuffling with replacement repeats some x_t values while not using as many others. It never admits nearby data values in a resample. We are considering applications where there is no reason to believe that values near the observed x_t are impossible. For example, let $x_t = 49.2$. Since 49.19 or 49.24, both of which round to $x_t = 49.2$, there is no justification for excluding all such values.

- The traditional bootstrap resamples must lie in the closed interval $[\min(x_t), \max(x_t)]$. Since the observed range is random, we cannot rule out somewhat smaller or larger x_t. Note that the third step of our algorithm implies a less restrictive and wider range $[z_0, z_T]$.

- The traditional bootstrap resample shuffles x_t such that any dependence information in the time series sequence $(x_1, \ldots, x_t, x_{t+1}, \ldots, x_T)$ is lost in the shuffle. If we try to restore the original order to the shuffled resample of the traditional bootstrap, we end up with essentially the original set x_t, except that some dropped x_t values are replaced by the repeats of adjacent values. Hence, it is impossible to generate a large number J of sensibly distinct resamples with the traditional bootstrap shuffle without admitting nearby values.

2.4. Shape retention

The j-th ME boot resample $\{x_{j,t}\}$ retains the shape, or local peaks and troughs, of the original time series x_t, by being 'strongly dependent' on it. We now imagine that the time series x_t represents a set (or bundle) of levels of 'utility' enjoyed by someone. Economic theorists do not like to make interpersonal comparisons of utility, since two persons can never really 'feel' exactly the same level of satisfaction. Yet economists must compare utilities to make policy

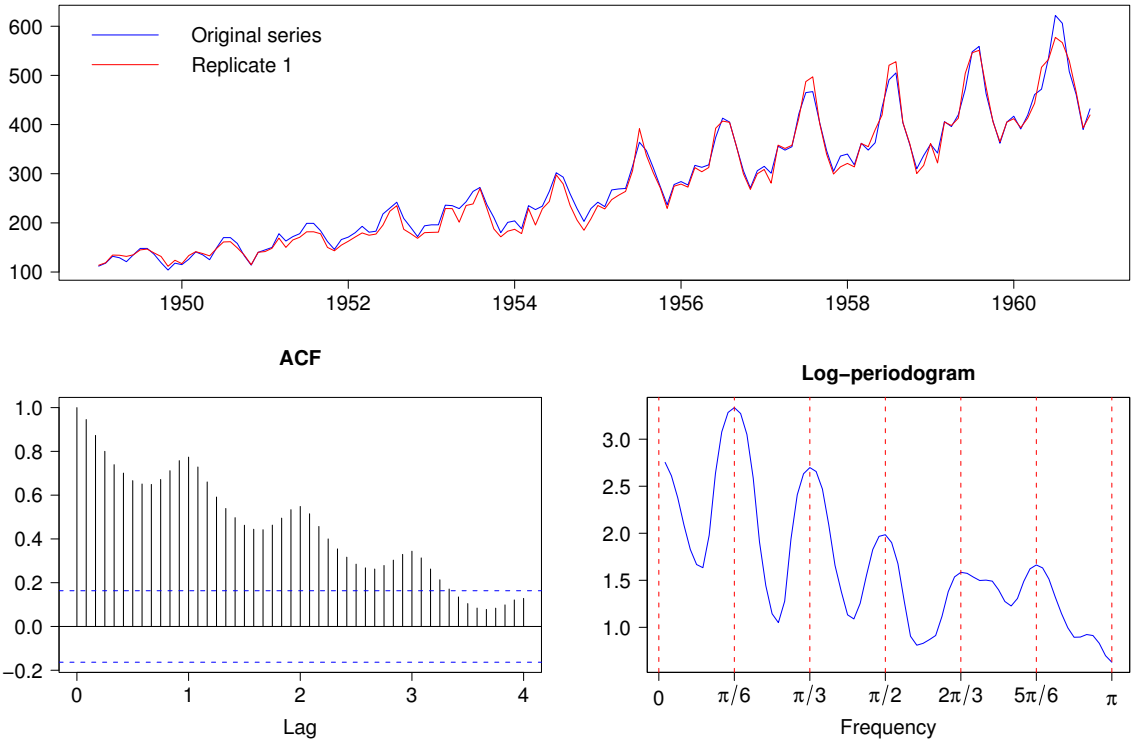

Figure 3: Replicate for the `AirPassengers` time series.

recommendations by considering preference orderings based on 'ordinal utility theory,' which says that utilities experienced by two individuals can be made comparable to each other, provided the two utility bundles satisfy a common partial ordering. Indeed our ME boot resamples do satisfy a common partial ordering, since their ranks match perfectly.

Imagine that the original $\{x_t\}$ represents the evolving time path for an individual's income, sensitive to initial resources at birth and intellectual endowments with a corresponding path of utility (enjoyment) levels. Our ME boot algorithm creates reincarnations of these paths ensuring that ordinal utilities are comparable across reincarnations, retaining just enough of the basic shape of x_t. See Henderson and Quandt (1980) for a discussion of multi-period consumption and ordinal utility.

Next we provide an example of how ME boot retains the shape as well as the periodicity of the original series by using the `AirPassengers` time series available in R.

Example: `AirPassengers` *time series*

Figure˜3 displays the `AirPassengers` time series along with a replicate of the series. An animation showing different replicates is available as a supplemental AVI file along with Vinod and López-de-Lacalle (2009). The autocorrelation function and the log-periodogram are shown for each replicate. One can see that, retaining the shape of the original series, the replicates remain close to the time and frequency domain properties of the series, without imposing any parametric restrictions.

3. Applications

3.1. Consumption function

This example describes how to carry out inference through the ME boot ensemble in the following regression:

$$c_t = \beta_1 + \beta_2\, c_{t-1} + \beta_3\, y_{t-1} + u_t, \tag{1}$$

for the null hypothesis $\beta_3 = 0$.

We use the annual data set employed in Murray (2006, pp.˜799–801) to discuss a Keynesian consumption function on the basis of the Friedman's permanent income hypothesis (PIH) and a simpler version of Robert Hall's model. The data are the logarithm of the US consumption, c_t, and disposable income, y_t, in the period 1948–1998. The packages **car** (Fox 2002) and **lmtest** (Zeileis and Hothorn 2002) will be useful to extract information from linear regression models. We use the interface in package **dynlm** (Zeileis 2008) for dynamic linear regression.

```
R> library("meboot")

[1] "kinship is loaded"

R> library("car")
R> library("lmtest")
R> library("dynlm")
R> data("USconsum")
R> USconsum <- log(USconsum)
R> lmcf <- dynlm(consum ~ L(consum, 1) + L(dispinc, 1), data = USconsum)
R> coeftest(lmcf)

t test of coefficients:

              Estimate Std. Error t value Pr(>|t|)
(Intercept)     0.0269     0.0261    1.03     0.31
L(consum, 1)    0.9697     0.1426    6.80  1.6e-08 ***
L(dispinc, 1)   0.0270     0.1439    0.19     0.85
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R> set.seed(135)
R> durbin.watson(model = lmcf, max.lag = 4)

 lag Autocorrelation D-W Statistic p-value
   1         0.14598         1.690   0.138
   2        -0.03521         2.018   0.986
   3        -0.08826         2.083   0.740
   4        -0.08850         2.078   0.622
 Alternative hypothesis: rho[lag] != 0
```

The residuals are serially uncorrelated since the p˜values of the generalized Durbin-Watson (DW) statistics up to order 4 are larger than the significance level 0.05. The seed was needed in the above code for a reproducible computation of p˜values for the DW statistics. The estimated coefficient of lagged income, $\hat{\beta}_3 = 0.027$, with the standard error $se = 0.1439$, is statistically insignificant. The 95% confidence interval $(-0.263, 0.316)$ has the zero inside this interval.

This result was initially interpreted as supporting Friedman's PIH. However, the large unit root literature argued that the sampling distribution of $\hat{\beta}_3$ is nonstandard, and that traditional inference based on the Student's t or asymptotic normal distributions may lead to spurious results. Hence, these days, one uses unit root tests to decide whether differencing or detrending of c_t and y_t would make all variables in a regression integrated of the same order, say $I(0)$. The critical values from a Dickey-Fuller type nonstandard density (originally obtained by a simulation) replace the usual Student's t critical values. Our bootstrap also reveals any nonstandard features of the sampling distribution and confidence intervals specific to the problem at hand, avoiding the use of critical values altogether. Thus we can cover a wide variety of situations beyond the one simulated by Dickey and Fuller.

Instead of resampling the residuals, our ME bootstrap resamples all time series in the regression themselves by following the 'resampling cases' bootstrap method. Three advantages of this method noted by Davison and Hinkley (1997, Section˜6.2.4) are: (a) This method does not use any simulated errors based on the assumed reliability of a parametric model. (b) It does not need to assume that the conditional mean of the dependent variable given a realization of regressors ($E(y|X = x)$ in standard notation) is linear. (c) It is robust against heteroscedastic errors.

Now we briefly describe the 'resampling cases' method in the context of time series regressions, where the 'case' refers to time. From (1) it is intuitively clear that we should resample only the two 'original' time series c_t and y_t, and then lag them as needed instead of blindly resampling (c_t, c_{t-1}, y_{t-1}) all three variables in the model. Our bootstrap inference will rely on a confidence interval for any function $\theta = f(\beta)$ of coefficients β. For example, $\theta = \beta_3$ for assessing the Friedman hypothesis based on (1).

```
R> theta <- function(y, x) {
+    reg <- dynlm(y ~ L(y, 1) + L(x, 1))
+    thet <- coef(reg)[3]
+    return(thet)
+  }
```

The above code represents our choice of simplicity over generality. It is intended to be used upon replacing the `y` by c_t and the `x` by y_t, for its use as $\theta = \beta_3$ in (1). For any other example it provides a template, needing modifications. If a researcher wishes to analyze the scale elasticity of a Cobb-Douglas type production function, Vinod (2008, pp.˜10–11), the regression becomes $y = \beta_1 x_1 + \beta_2 x_2$. Then the parameter of interest: $\theta = \beta_1 + \beta_2$, is a sum of two slope coefficients. The modified `theta` for this example denoted by `theta.cobbd` is given by the following code:

```
R> theta.cobbd <- function(y, x1, x2) {
+    reg <- lm(y ~ x1 + x2)
```

```
+    thet <- coef(reg)[2] + coef(reg)[3]
+    return(thet)
+  }
```

In general, a modification of `theta` can involve a nonlinear function of several coefficients. For example, Vinod (2008, Section˜3.2), if the parameter of interest is the long-run multiplier, it becomes a nonlinear function. The main point is that our θ refers to only one parameter of interest. Any researcher interested in two or more parameters can readily repeat our procedure as often as needed.

The following function called `bstar.consu` generates a large number of bootstrap single parameter estimates. The `bstar` in its name suggests that resamples of the third regression coefficient might be denoted as $\{b_3^*\}$. More important, it is a template, expecting modifications. Its initial arguments refer to data on all 'original' time series (not counting leads and lags as separate series) using the notation y for the dependent variable and x for the regressor. It is flexible, allowing the user to choose the confidence level (default: 95%), the R function `theta` (defining the parameter of interest must be predefined), size of resamples as `bigJ` (default: `bigJ = 999`) and the seed for random number generator as `seed1` (default: `seed1 = 135`).

```
R> bstar.consu <- function(y, x, theta,
+    level = 0.95, bigJ = 999, seed1 = 135) {
+    set.seed(seed1)
+    semy <- meboot(x = y, reps = bigJ)$ensemble
+    semx <- meboot(x = x, reps = bigJ)$ensemble
+    n <- NROW(y)
+    m <- length(theta(y, x))
+    if(m!=1) stop("too many parameters in theta")
+    bb <- matrix(NA, bigJ)
+    for(j in 1:bigJ) {
+      yy <- semy[,j]
+      xx <- semx[,j]
+      bb[j] <- theta(yy, xx)
+    }
+    return(bb)
+  }
```

Since the Cobb-Douglas model involves regressing y on x_1 and x_2, its function (called `bstar.cobbd` below) has an additional input `x2`. It calls the function `meboot` thrice for `y, x1` and `x2`. Also, the input to the function `theta.cobbd` needs both `xx1` and `xx2` instead of simply `xx`, in the code `bstar.consu` above. We believe that it is easy to make such changes to our simple and intuitive `bstar` type functions. The template `bstar.cobbd`, for the two regressor Cobb-Douglas case below, explicitly shows how to extend the function `bstar.consu` to two or more regressors by using `x2`, `x3`, `x4`, ... as needed.

```
R> bstar.cobbd <- function(y, x1, x2, theta = theta.cobbd,
+    level = 0.95, bigJ = 999, seed1 = 135) {
+    set.seed(seed1)
```

```
+    semy <- meboot(x = y, reps = bigJ)$ensemble
+    semx1 <- meboot(x = x1, reps = bigJ)$ensemble
+    semx2 <- meboot(x = x2, reps = bigJ)$ensemble
+    n <- NROW(y)
+    m <- length(theta.cobbd(y, x1, x2))
+    if(m!=1) stop("too many parameters in theta")
+    bb <- matrix(NA, bigJ)
+    for(j in 1:bigJ) {
+      yy <- semy[,j]
+      xx1 <- semx1[,j]
+      xx2 <- semx2[,j]
+      bb[j] <- theta.cobbd(yy, xx1, xx2)
+    }
+    return(bb)
+  }
```

Now we return to constructing an approximation to the sampling distribution of $\hat{\beta}_3$ in (1), without having to assume that the distribution is Student's t or Dickey-Fuller. That is, we use the output of the function `bstar.consu` to construct a confidence interval for β_3 to help decide whether $\hat{\beta}_3$ is statistically significantly different from zero. Assuming the earlier code is in the memory, let us begin by computing the simplest percentile interval, using the function `quantile` of R, while choosing `type = 8` (as recommended by Hyndman and Fan 1996, see also `help("quantile")`).

```
R> y <- USconsum[,2]
R> x <- USconsum[,1]
R> reg <- dynlm(y ~ L(y, 1) + L(x, 1))
R> su <- summary(reg)
R> se <- su$coefficients[3,2]
R> t0 <- theta(y, x)
R> b3s <- bstar.consu(y, x, theta)
R> simple.percentile <- quantile(b3s, c(0.025, 0.975), type = 8)
R> asymmetric.around.0 <- null.ci(b3s)
R> out <- list(t = b3s, t0 = t0, var.t0 = se^2, R = 999)
R> class(out) <- "boot"
R> library("boot")
R> boot.percentile <- boot.ci(out, type = "perc")$percent[4:5]
R> boot.norm <- boot.ci(out, type = "norm")$normal[2:3]
R> boot.basic <- boot.ci(out, type = "basic")$basic[4:5]
R> rbind(simple.percentile, asymmetric.around.0, boot.percentile,
+    boot.norm, boot.basic)
```

```
                       2.5%  97.5%
simple.percentile   -0.04485 0.3733
asymmetric.around.0 -0.06250 0.3221
boot.percentile     -0.04501 0.3739
boot.norm           -0.16256 0.2643
boot.basic          -0.20742 0.2115
```

The code above reports four intervals beyond the simplest one mentioned before. The **meboot** package includes the function `null.ci` (an elegant improvement by Achim Zeileis of our function `zero.ci`) which provides asymmetric confidence intervals arond a specified null value (=0, here). The names of three confidence intervals have the prefix boot to remind us that they come from the **boot** package (Canty and Ripley 2009). These are available only after `out` is defined with a suitable `list` and `boot.ci` function is called with appropriate options. These options provide some of the well known refinements to the percentile confidence interval from the bootstrap literature.

Statistical theory behind these refinements is mentioned at the beginning of Section 2. In the present context, bootstrap estimates of θ (see `b3s` above) are $\hat{\theta}^*_j$, with $j = 1, 2, \ldots, J$. If the standard error se of $\hat{\theta}$ is known, then $(\hat{\theta}^*_j - \hat{\theta})/se$ values provide a good approximation to the sampling distribution of $(\hat{\theta} - \theta)/se$, the Wald statistic. The code in `boot.ci` is designed to correct for bias and improve asymptotic properties of bootstrap confidence intervals.

Finally, let us consider sophisticated confidence intervals based on highest density regions (HDR) of sampling distributions, Hyndman (1996). If $f(\hat{\theta})$ is the density, and α is the type˜I error (= 0.05, say), then the $100(1-\alpha)\%$ HDR is the subset of the sample space of the random variable such that

$$HDR(f_\alpha) = \{\hat{\theta} : f(\hat{\theta}) \geq f_\alpha\}, \tag{2}$$

where f_α is the largest constant such that the following probability statement holds true: $Pr(\hat{\theta} \in HDR(f_\alpha)) \geq 1-\alpha$. Highest density means every point inside the HDR has probability density at least as large as every point outside the HDR. When the sampling distribution is bimodal or multimodal, HDR seems to be a reliable way of finding confidence regions. Hyndman (1996) discusses many advantages of HDR methods. We use the R package **hdrcde** (Hyndman 2008) to find HDR regions with graphics for a study of the sampling distribution of $\hat{\theta}$ under the null. It also reports the value of f_α appearing in equation (2).

```
R> library("hdrcde")

This is hdrcde 2.09

R> hdr.den(b3s, main = expression(Highest ~ density ~ region ~
+    of ~ beta [3] ~ estimates :  ~  Hall ~ model))

$hdr
        [,1]   [,2]
99% -0.09297 0.4310
95% -0.07104 0.3128
50%  0.01020 0.1571

$mode
[1] 0.1050

$falpha
    1%     5%    50%
0.1963 0.6207 3.1236
```

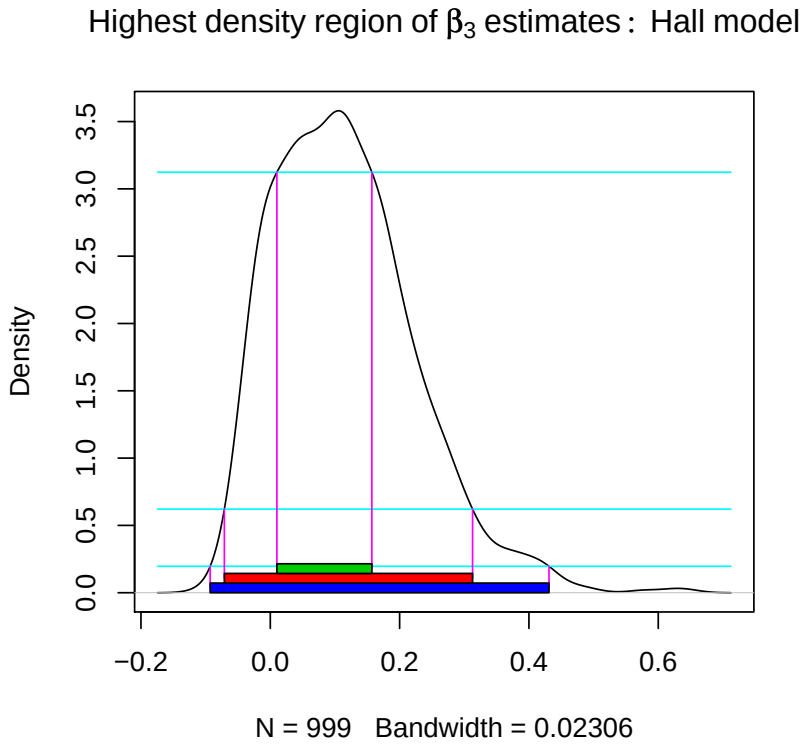

Figure 4: Highest density region for the sampling distribution of $\hat{\beta}_3$ using Hall's model.

Note that even the 50% confidence region (HDR) starts at nearly zero, while 95% region decidedly covers the zero. However the largest constants f_α are all positive. Our **meboot** results (including the HDR) support Friedman's PIH, since zero is inside all 95% confidence intervals for β_3.

3.2. Assessment of the Fed effect on stock prices using panel data

This example shows how the ME bootstrap can be employed for panel data analysis. Our example is from Vinod (2002) where the effect of monetary policy (interest rates) on prices and their 'turning points' in the stock market is evaluated in greater detail.

The 'Fed effect' discussed in the financial press refers to a rally in the S&P 500 index of stock prices a few days before the Federal Reserve Bank (Fed) policy makers' meeting and a price decline after the meeting. This example focuses on the longer term than daily price fluctuations by using the monthly data (May 1993 to November 1998 with $T = 67$) for stocks with ticker symbols: ABT, AEG, ATI, ALD, ALL, AOL, and AXP and regard this as a representative sample of the market containing $N = 7$ individual companies.

Note that when the Fed adjusts the Fed funds rate, it affects market expectations and, hence, the interest on 3-month Treasury bills (*Tb3*), the key short-run interest rate in the economy. Our simple model of monetary policy regresses the stock price (P) on the natural log of

market capitalization (LMV, as a control variable for the size of the firm) and the $Tb3$. We write:

$$P_{it} = \beta_1 + \beta_2\, LMV_{it} + \beta_3\, Tb3_{it} + \varepsilon_{it}\,, \tag{3}$$

where the subscript it refers to i-th individual (company) at time t and where ε_{it} are assumed to be iid. The Fed effect is present, if the coefficient of the variable $Tb3$ in equation (3) is statistically significant.

We use the R package **plm** (Croissant and Millo 2008) for basic estimation of panel data models before any bootstrap. It expects that the data be in the form of a `data.frame` object. Accordingly, the package **meboot** provides the data for this example as a `data.frame` object called `ullwan`. Let us replace the third column containing the 'market value of the firm' by its logarithms denoted by LMV within the data frame object. Since the data setup is critical, it is perhaps useful to illustrate our (slightly revised) data frame `ullwan` by displaying the initial and ending observations. Note that the first column entitled `Subj` contains the identification numbers 1 to 7, for the 7 ticker symbols included in the data set. Note that all time series for the first ticker symbol ABT are placed together at the beginning of the data set. These are viewed by using the command: `head(ullwan)`. Trailing data for observation numbers 464 to 469 for the last ticker symbol AXP are viewed by using the command: `tail(ullwan)` in the following code.

```
R> rm(list = ls())
R> library("plm")
R> data("ullwan")
R> attach(ullwan)
R> LMV <- log(MktVal)
R> ullwan[,3] <- LMV
R> names(ullwan)[3] <- "LMV"
R> head(ullwan)
```

```
                   Subj              Tim    LMV Price Pupdn  Tb3
1-1993.33333333333    1 1993.33333333333 10.099 14.81     0 3.08
1-1993.41666666667    1 1993.41666666667 10.095 14.75     1 3.02
1-1993.5              1           1993.5 10.029 13.81     0 3.21
1-1993.58333333333    1 1993.58333333333  9.987 13.31     0 3.52
1-1993.66666666667    1 1993.66666666667 10.051 14.19     1 3.74
1-1993.75             1          1993.75 10.102 14.94     0 4.19
```

```
R> tail(ullwan)
```

```
                   Subj              Tim   LMV Price Pupdn  Tb3
7-1998.41666666667    7 1998.41666666667 10.74 102.9     0 4.34
7-1998.5              7           1998.5 10.80 108.5     0 4.45
7-1998.58333333333    7 1998.58333333333 10.88 117.8     0 4.48
7-1998.66666666667    7 1998.66666666667 10.98 130.7     0 4.28
7-1998.75             7          1998.75 10.91 121.1     1 4.51
7-1998.83333333333    7 1998.83333333333 10.98 130.1     1 4.59
```

Pooled effects

Pooled regression means combining the cross-sectional data and time series data into one large set of $T \times N (= 67 \times 7 = 469$ here) observations. We note below that Student's t˜value and the corresponding p˜value from the pooled model suggest a highly significant *Tb3* regressor implying that the Fed announcement does have a statistically significant effect on the level of stock prices. The multiple R^2 is 0.497, which becomes 0.495 when adjusted for degrees of freedom.

```
R> summary(lm(Price ~ LMV + Tb3))

Call:
lm(formula = Price ~ LMV + Tb3)

Residuals:
   Min     1Q Median     3Q    Max
-35.41 -11.47  -2.92   4.82  70.91

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -125.596     11.209  -11.21   <2e-16 ***
LMV           18.848      0.886   21.26   <2e-16 ***
Tb3           -4.805      1.479   -3.25   0.0012 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 19 on 466 degrees of freedom
Multiple R-squared: 0.497,        Adjusted R-squared: 0.495
F-statistic:  230 on 2 and 466 DF,  p-value: <2e-16
```

The t˜value for the coefficient of *Tb3* from the pooled model suggests that the Fed does have a statistically significant effect on the level of stock prices in a pooled regression.

The default use of the function `meboot` is illustrated in the earlier subsection in `bstar.consu` and `bstar.cobbd`, where the argument `x` represents a single vector of numbers (usually time series). In this subsection we require `meboot` to create J replicates over time, separately for all N individuals to suit the panel data. Since the **plm** package expects the panel data in the form of a `data.frame` object, our function `meboot` is designed to expect similarly organized data. For example, since the identifier for individual (company ticker) called `Subj` is located in the first column of the `ullwan` data frame, the `meboot` would expect `colsubj=1` as the argument. It is necessary to call the `meboot` function separately for each relevant column of the data frame identified by its column number denoted as the argument `coldata`. For example, we set `coldata = 3` for LMV since third column has data on LMV. Mere presence of non-null values for `colsubj` and `coldata` in the call to `meboot` triggers it to implement its panel data version.

The following code creates $J = 999$ ensembles for the $T = 67$ time series points for the $N = 7$ stocks separately for the three variables in our regression (P, LMV and $Tb3$). Each call

yields an ensemble of 1000 sets of $67 \times 7 = 469$ data points, upon including the original data as the first column and 999 additional columns of similarly evolving time series.

```
R> jboot <- 999
R> set.seed(567)
R> LMV.ens <- meboot(x = ullwan, reps = jboot, colsubj = 1, coldata = 3)
R> Price.ens <- meboot(x = ullwan, reps = jboot, colsubj = 1, coldata = 4)
R> Tb3.ens <- meboot(x = ullwan, reps = jboot, colsubj = 1, coldata = 6)
```

The purpose of the ME bootstrap here is to assess whether the effect of the Fed announcement continues to be significant for the pooled and other models described below. The slope coefficients based on the ME bootstrap ensembles (created above) can be computed as follows.

```
R> slopeTb3 <- slopeLMV <- rep(0, jboot)
R> for(j in 1:jboot) {
+    frm <- data.frame(Subj = ullwan[,1], Time = ullwan[,2],
+      Price = Price.ens[,j], Tb3 = Tb3.ens[,j], LMV = LMV.ens[,j])
+    frm <- plm.data(frm, 7)
+    gip <- coef(plm(Price ~ LMV + Tb3, model = "pooling", data = frm))
+    slopeTb3[j] <- gip[3]
+    slopeLMV[j] <- gip[2]
+  }
R> Percentile.Tb3 <- quantile(slopeTb3, c(0.025, 0.975), type = 8)
R> Refined.Tb3 <- null.ci(slopeTb3)
R> Percentile.LMV <- quantile(slopeLMV, c(0.025, 0.975), type = 8)
R> Refined.LMV <- null.ci(slopeLMV)
R> rbind(Percentile.Tb3, Refined.Tb3, Percentile.LMV, Refined.LMV)

                 2.5%  97.5%
Percentile.Tb3 -6.036 -3.046
Refined.Tb3    -5.814 -2.146
Percentile.LMV 17.269 20.968
Refined.LMV    16.392 20.655
```

The 95% ME bootstrap confidence intervals for the control variable LMV has all estimated slopes positive and the lower limit of of the asymmetric (refined) 95% interval is simply the smallest slope. It suggests that β_2 in (3) for the pooled model is statistically significantly positive. That is, it is worthwhile to include the control variable in the model.

The 95% ME bootstrap percentile confidence interval for the slope coefficient of *Tb3* is shown above. Since all estimated slopes by the ME bootstrap are negative with the null value zero, the upper limit of the asymmetric (refined) 95% interval is simply the largest slope. It is interesting to also report additional advanced confidence intervals from the package **boot** by using the function `boot.ci` with some preliminary code to conform with its protocol.

```
R> thetp <- plm(Price ~ LMV + Tb3, model = "pooling", data = ullwan)
R> varTb3 <- thetp$vcov[3,3]
R> plm1 <- coef(thetp)
```

```
R> t0Tb3 <- plm1[3]
R> t0LMV <- plm1[2]
R> out2 <- list(t = as.matrix(slopeTb3), t0 = t0Tb3,
+    var.t0 = varTb3, R = 999)
R> class(out2) <- "boot"
R> boot.percentile <- boot.ci(out2, type = "perc")$percent[4:5]
R> boot.norm <- boot.ci(out2, type = "norm")$normal[2:3]
R> boot.basic <- boot.ci(out2, type = "basic")$basic[4:5]
R> rbind(Percentile.Tb3, boot.percentile, boot.norm, boot.basic)

                  2.5%  97.5%
Percentile.Tb3  -6.036 -3.046
boot.percentile -6.040 -3.041
boot.norm       -6.599 -3.511
boot.basic      -6.568 -3.569
```

These results clearly suggests that for the pooled model β_3 in (3) is statistically significantly negative.

```
R> znp <- pvcm(Price ~ LMV + Tb3, data = ullwan, model = "within")
R> zplm <- plm(Price ~ LMV + Tb3, data = ullwan)
R> pooltest(zplm, znp)

        F statistic

data:  Price ~ LMV + Tb3
F = 63.4, df1 = 12, df2 = 448, p-value < 2.2e-16
alternative hypothesis: unstability
```

The function `pvcm` refers to panel variable coefficients models. If pooling is appropriate the coefficients for individual units do not significantly differ from one another. The high value of the F˜statistic in the output of the `pooltest` suggests that pooling may not be appropriate. Hence we need to consider a 'random effects' model next.

Random effects

The random effects model results are obtained by setting `model = "random"` in the arguments of the function `plm`.

```
R> gir <- plm(Price ~ LMV + Tb3, data = ullwan, model = "random")
R> coef(gir)

(Intercept)         LMV         Tb3
    -184.92       25.00       -4.96
```

Using the ensembles created above for all data variables in equation (3) we implement the ME bootstrap for the random effects model and print various confidence intervals as follows.

```
R> slopeTb3 <- slopeLMV <- rep(0, jboot)
R> for(j in 1:jboot) {
+    frm <- data.frame(Subj = ullwan[,1], Tim = ullwan[,2],
+      Price = Price.ens[,j], Tb3 = Tb3.ens[,j], LMV = LMV.ens[,j])
+    frm <- plm.data(frm, 7)
+    gip <- coef(plm(Price ~ LMV + Tb3, model = "random", data = frm))
+    slopeTb3[j] <- gip[3]
+    slopeLMV[j] <- gip[2]
+  }
R> Percentile.Tb3 <- quantile(slopeTb3, c(0.025, 0.975), type = 8)
R> Refined.Tb3 <- null.ci(slopeTb3)
R> Percentile.LMV <- quantile(slopeLMV, c(0.025, 0.975), type = 8)
R> Refined.LMV <- null.ci(slopeLMV)
R> rbind(Percentile.Tb3, Refined.Tb3, Percentile.LMV, Refined.LMV)
```

```
                 2.5%  97.5%
Percentile.Tb3 -6.168 -3.095
Refined.Tb3    -5.974 -2.303
Percentile.LMV 21.528 28.449
Refined.LMV    19.415 27.906
```

Now we use the function `boot.ci` after some preliminary code to conform with its protocol to obtain additional 'random effects' confidence intervals for the coefficient of $Tb3$.

```
R> thetr <- plm(Price ~ LMV + Tb3, model = "random", data = ullwan)
R> varTb3 <- thetr$vcov[3,3]
R> plm1 <- coef(thetr)
R> t0Tb3 <- plm1[3]
R> out3 <- list(t = as.matrix(slopeTb3), t0 = t0Tb3, var.t0 = varTb3, R = 999)
R> class(out3) <- "boot"
R> boot.percentile <- boot.ci(out3, type = "perc")$percent[4:5]
R> boot.norm <- boot.ci(out3, type = "norm")$normal[2:3]
R> boot.basic <- boot.ci(out3, type = "basic")$basic[4:5]
R> rbind(boot.percentile, boot.norm, boot.basic)
```

```
                  [,1]   [,2]
boot.percentile -6.170 -3.084
boot.norm       -6.855 -3.684
boot.basic      -6.836 -3.750
```

The random effects 95% ME bootstrap confidence intervals using the 999 replicates of data are essentially similar to the pooled data results, allowing us to conclude that both β_2 and β_3 in (3) are significantly different from zero. Since the 95% confidence intervals for β_3 do not cover the zero, we can conclude that the 'Fed effect' is significant for the random effects panel data model.

4. Concluding remarks

This paper illustrates the performance and usage of Vinod's maximum entropy bootstrap for dependent data by using econometric examples, including one involving panel (longitudinal) data. Besides econometrics, at least some time series in biology, engineering and social sciences are similarly state-dependent and subject to structural changes and discontinuities. All such series cannot be transformed into stationary series without impairing our understanding of underlying relations among them. The **meboot** R package not only fills a gap in the bootstrap toolkit, but is particularly useful as it permits simpler model specifications (allowing a direct use of one or more such time series without first making them stationary) and convenient statistical inference (avoiding non-standard Dickey-Fuller type sampling distributions).

Acknowledgments

We thank the referee, Achim Zeileis and Johanna Francis for helpful comments.

References

Canty A, Ripley BD (2009). ***boot***: *Bootstrap R (S-PLUS) Functions*. R˜package version˜1.2-35, URL http://CRAN.R-project.org/package=boot.

Croissant Y, Millo G (2008). "Panel Data Econometrics in R: The **plm** Package." *Journal of Statistical Software*, **27**(2), 1–43. URL http://www.jstatsoft.org/v27/i02/.

Davison AC, Hinkley DV (1997). *Bootstrap Methods and Their Applications*. Cambridge University Press, Cambridge.

Fox J (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Publications, Thousand Oaks, CA.

Henderson JM, Quandt RE (1980). *Microeconomic Theory: A Mathematical Approach*. 3rd edition. McGraw Hill, New York.

Hyndman RJ (1996). "Computing and Graphing Highest Density Regions." *The American Statistician*, **50**, 120–126.

Hyndman RJ (2008). ***hdrcde***: *Highest Density Regions and Conditional Density Estimation*. R˜package version˜2.09, URL http://CRAN.R-project.org/package=hdrcde.

Hyndman RJ, Fan Y (1996). "Sample Quantiles in Statistical Packages." *The American Statistician*, **50**, 361–365.

Khintchine AI (1934). "Korrelationstheorie der stationären stochastischen Prozesse." *Mathematische Annalen*, **109**, 604–615.

Kolmogorov AN (1931). "Über die analytischen Methoden in der Wahrscheinlichkeitsrechnung." *Mathematische Annalen*, **104**, 415–458.

Lahiri SN (2003). *Resampling Methods for Dependent Data*. Springer-Verlag, New York.

Murray MP (2006). *Econometrics: A Modern Introduction*. Addison-Wesley, New York.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Singh K (1981). "On the Asymptotic Accuracy of Efron's Bootstrap." *The Annals of Statistics*, **9**, 1187–1195.

Vinod HD (2002). "Econometric Applications of Generalized Estimating Equations for Panel Data and Extensions to Inference." In A˜Ullah, ATK Wan, A˜Chaturvedi (eds.), *Handbook of Applied Econometrics*, chapter˜26, pp. 553–574. Marcel Dekker, New York.

Vinod HD (2004). "Ranking Mutual Funds Using Unconventional Utility Theory and Stochastic Dominance." *Journal of Empirical Finance*, **11**(3), 353–377.

Vinod HD (2006). "Maximum Entropy Ensembles for Time Series Inference in Economics." *Journal of Asian Economics*, **17**(6), 955–978.

Vinod HD (2008). *Hands-On Intermediate Econometrics Using R: Templates for Extending Dozens of Practical Examples*. World Scientific, New Jersey.

Vinod HD, López-de-Lacalle J (2009). "Maximum Entropy Bootstrap for Time Series: The **meboot** R Package." *Journal of Statistical Software*, **29**(5), 1–19. URL http://www.jstatsoft.org/v29/i05/.

Wiener N (1930). "Generalized Harmonic Analysis." *Acta Mathematica*, **55**, 117–258.

Zeileis A (2008). ***dynlm**: Dynamic Linear Regression*. R˜package version˜0.2-0, URL http://CRAN.R-project.org/package=dynlm.

Zeileis A, Hothorn T (2002). "Diagnostic Checking in Regression Relationships." *R News*, **2**(3), 7–10. URL http://CRAN.R-project.org/doc/Rnews/.

Affiliation:

Hrishikesh D. Vinod
Fordham University
Department of Economics, Institute of Ethics and Economic Policy
Bronx, NY 10458, United States of America
E-mail: Vinod@fordham.edu
URL: http://www.fordham.edu/economics/vinod/

Javier López-de-Lacalle
Universidad del País Vasco
Facultad de Ciencias Económicas y Empresariales
48015 Bilbao, Spain
E-mail: javlacalle@yahoo.es
URL: http://www.bl.ehu.es/˜jedlobej/