Computing Generalized Empirical Likelihood and Generalized Method of Moments with R

Pierre Chaussé

Abstract

I present in this paper the package **gmm** that I wrote for the object oriented and open source statistical package R to compute the generalized method of moments (GMM) and the generalized empirical likelihood (GEL) estimators. There were no statistical package until now that included procedures to implement GEL, and those which offer a GMM procedure are not very flexible. The package presented here includes most of the recent development in this area. Its implementation is presented through both theoretical and applied examples.

Keywords: generalized empirical likelihood, generalized method of moments, empirical likelihood, continuous updated estimator, exponential tilting, exponentially tilted empirical likelihood, R.

1. Introduction

Usually, when one needs to estimate parameters using the generalized method of moments (GMM) of Hansen (1982), they face very few options. Softwares that offer this estimation procedure are in general quite restrictive. For example, they do not have many options related to the computation of the heteroskedasticity and autocorrelation covariance matrix (HAC) of the moment conditions. And most of them offer only the two step or the iterated GMM. They don't include a procedure to implement the continuous updated estimator even if recent theoretical studies demonstrate its superiority at least asymptotically. That is why several econometricians rely on homemade algorithms, written in Matlab, Gauss or other commercial statistical packages, which are written exclusively for their specific models. Although there may be some advantages to work with our own algorithm, it does raise the question of the reliability of the results. Everyone have heard the story of a scholar not being able to reproduce someone else's results. It is not easy to write a reliable algorithm and these stories may be the result of researchers falling in one the many numerical pitfalls. Even commercial algorithms often produce different results and there is no way to know why because their sources are note open.

Asymptotic properties of GMM and generalized empirical likelihood (GEL) are now well established in the econometric literature. Newey and Smith (2004) and Anatolyev (2005) have come a little closer to finite sample by exploring second order asymptotic properties, but it is still only asymptotics. These results are important but they may be far away from actual finite sample porterties. In order to analyze the behavior of GMM and GEL in small samples, there are no other ways than to rely on numerical simulations. Because of the

rapid technological progress in computing, it is getting much faster to simulate complicated models. We just need to make sure our results are reliable. Guggenberger (2008) contradicts some of the asymptotic results which show the superiority of empirical likelihood over GMM using simulations. I tried to reproduce his results without success. When I presented results from simulations at the Canadian Economic Association meeting to compare GMM for a continuum of moment conditions of Carrasco and Florens (2000) and GEL for a continuum of moment conditions of Chaussé (2008), one of the question I got was: "Are you sure you did it correctly?". It is a legitimate question. We can easily see if a mathematical proof is wrong, but detecting errors in simulations is much harder. Methods like continuous updated estimator (CUE) or GEL require to solve highly nonlinear system of equations. We therefore need to be careful when writing programs to implement them.

The solution I offer makes it possible to everyone to test the algorithm either by doing simulations, model estimations or by exploring the codes. It is the most interesting aspect of the open source world. There can be many contributors who help improving the package. And the more popular it will become, the more reliable it will be. We will then be able to spend our precious time understanding and interpreting the simulation results instead of trying to find what went wrong in the programming. I could have chosen another open source software like Octave (a pseudo clone of Matlab) or Scilab, but R is more adapted to econometrics and statistics. Furthermore, its object oriented approach allows very efficient programming.

The paper is organized as follows. Section 2 covers the GMM method while section 3 presents the GEL. In each section, we start with a brief summary of the method and then show how to implement it trough examples. Section 4 concludes.

2. Generalized method of moments

Before going to the estimation procedure, let us make a quick summary of what this method is about. Those who are not too familiar with this method and want a more detailed presentation should go through Hansen (1982) and Hansen, Heaton, and Yaron (1996) or consult a textbook such as Hamilton (1994).

2.1. The method

We want to estimate a vector of parameters $\theta \in \mathbb{R}^p$ from the following $q \times 1$ vector of unconditional moment conditions:

$$E[g(\theta, x_t)] = 0 \tag{1}$$

One of the biggest advantage of GMM is that it requires very few assumptions and can be seen as a method which embeds many others such as least squares (LS), maximum likelihood (ML) or instrumental variables (IV). For example, the following linear model:

$$Y = X\beta + u$$

can be estimated by LS which would imply the following moment conditions:

$$E(X_t u_t) = 0$$

estimated from a sample of T observations by the sample moment conditions:

$$\frac{1}{T}X'u = 0$$

The same model can be estimated by ML in which case the moment conditions would be:

$$E\left[\frac{dl_t(\beta)}{d\beta}\right] = 0$$

where $l_t(\beta)$ is the density of u_t. The last example, IV, is the method that most of the time is associated with GMM. The linear model is estimated using the conditional moment conditions $E(u_t|H_t) = 0$, where H_t is a set of instruments. From these conditional moment conditions, we can construct unconditional moment conditions like $E(H_t u_t) = 0$[1] with their sample counterparts:

$$\frac{1}{T}H'u = 0$$

This is probably the most attractive feature of GMM. Relying only on conditional moments to estimate a model allows us, in theory, to estimate fragments of a complete system of equations. It is quite appealing knowing that all our models are fragments of the real world. However, there is no such thing as free lunch as we all know. Therefore, the advantages of GMM come with a cost. The properties of its estimators depend heavily on the choice of the instruments. A bad choice can create very bad estimators. A lot of research is being made to improve our understanding of GMM. That is why we need a common numerical tool so that results can be compared without worrying about the reliability of the algorithm.

The above examples, by its linear framework, do not enlighten the needs of a well written algorithm. There are three features of GMM that makes it hard to implement. Firstly, the moment conditions $E(g(\theta, x_t)) = 0$ don't need to be linear functions of θ. Secondly, the number of conditions is not limited by the dimension of θ which forces us to replace the requirement of solving the sample moment conditions:

$$\bar{g}(\theta) \equiv \frac{1}{T}\sum_{t=1}^{T} g(\theta, x_t) = 0$$

by the necessity of minimizing the quadratic function $\bar{g}(\theta)'W\bar{g}(\theta)$, where W is a positive definite and symmetric $q \times q$ matrix. Finally, the optimal matrix W as been shown to be:

$$W^* = \left\{\lim_{T\to\infty} Var(\sqrt{T}\bar{g}(\theta)) \equiv \Omega(\theta)\right\}^{-1} \tag{2}$$

This optimal matrix can be estimated in general by replacing the asymptotic variance by an HAC matrix like the one proposed by Newey and West (1987a). The general form is:

$$\hat{\Omega} = \sum_{s=-(T-1)}^{T-1} k_h(s)\hat{\Gamma}_s(\theta^*) \tag{3}$$

[1] In fact any unconditional moment conditions of the form $E(u_t f(H_t)) = 0$ are also valid.

where, $k_h(s)$ is a kernel, h is the bandwidth which can be chosen using a procedure such as the ones proposed by Newey and West (1987a) or Andrews (1991),

$$\hat{\Gamma}_s(\theta^*) = \frac{1}{T}\sum_t g(\theta^*, x_t)g(\theta^*, x_{t+s})'$$

and θ^* is a convergent estimate of θ. The GMM estimator $\hat{\theta}$ is therefore defined as:

$$\hat{\theta} = \arg\min_\theta \bar{g}(\theta)'\hat{\Omega}(\theta^*)^{-1}\bar{g}(\theta) \tag{4}$$

The asymptotic properties of the GMM estimators do not depend on the way we estimate the optimal matrix but their finite sample properties do. Furthermore, they are certainly affected by the numerical algorithm used to obtain the estimates.

The original version of GMM proposed by Hansen (1982) is called two-step GMM (2SGMM). It obtains θ^* by minimizing $\bar{g}(\theta)'\bar{g}(\theta)$. The algorithm is therefore simple:

1- Compute $\theta^* = \arg\min_\theta \bar{g}(\theta)'\bar{g}(\theta)$

2- Compute the HAC matrix $\hat{\Omega}(\theta^*)$

3- Compute the 2SGMM $\hat{\theta} = \arg\min_\theta \bar{g}(\theta)'\left[\hat{\Omega}(\theta^*)\right]^{-1}\bar{g}(\theta)$

In order to improve the properties of 2SGMM, Hansen *et al.* (1996) suggest two other methods. The first one is the iterative version of 2SGMM (ITGMM) and can be computed as follows:

1- Compute $\theta_0 = \arg\min_\theta \bar{g}(\theta)'\bar{g}(\theta)$

2- Compute the HAC matrix $\hat{\Omega}(\theta_0)$

3- Compute the $\theta_1 = \arg\min_\theta \bar{g}(\theta)'\left[\hat{\Omega}(\theta_0)\right]^{-1}\bar{g}(\theta)$

4- If $\|\theta_0 - \theta_1\| < tol$ stops, else $\theta_0 = \theta_1$ and go to 2-

5- Define the ITGMM estimator $\hat{\theta} = \theta_1$

where *tol* can be set as small as we want to increase the precision.

In the other method, no preliminary estimate is used to obtain the HAC matrix. The latter is treated as a function of θ and is allowed to change when the optimization algorithm computes the numerical derivatives. It is therefore continuously updated as we move toward the minimum. For that, it is called the continuous updated estimator (CUE). Because of the complexity of the HAC matrix, CUE is highly nonlinear. We therefore need to be careful when choosing the starting values. A good choice could be to use the 2SGMM. The algorithm is simple but with a much bigger numerical challenge.

1- Compute θ_0 using 2SGMM

2- Compute the CUE estimate defined as

$$\hat{\theta} = \arg\min_\theta \bar{g}(\theta)'\left[\hat{\Omega}(\theta)\right]^{-1}\bar{g}(\theta)$$

using θ_0 as starting values.

According to Newey and Smith (2004) and Anatolyev (2005), 2SGMM and ITGMM are second order asymptotic equivalent. In other words, the choice of the convergent estimate θ^* in $\hat{\Omega}(\theta^*)$ does not matter. Iterating will just improve the efficiency of θ^*. However, this is an asymptotic result. It does not say how it affects the finite sample properties. On the other hand, they showed that the second order asymptotic bias of CUE is smaller. Unfortunately this method is not available in any popular statistical package.

2.2. GMM with R

The package **gmm** can be found on the comprehensive R archive network (CRAN, http://CRAN.R-project.org/) with any other R packages. Once installed, it can be loaded the usual way.

```
> require(gmm)
```

Many options are available but in many cases they can be set to their default values. We start with a general presentation in which moment conditions do not come from a linear model. The minimum requirement is to define a function that produces a $T \times q$ matrix $g(\theta)$ with typical element $[g(\theta)]_{ti} = g_i(\theta, x_t)$. Here is an example that was proposed by Dieter Rozenich, a student from Vienna University of Economics and Business Administration in Austria. It is not something we want to do in practice, but it shows well how to implement the `gmm()` algorithm.

Estimating the parameters of a normal distribution

For the two parameters of a normal distribution (μ, σ) we have the following three moment conditions[2]:

$$\begin{aligned} m_1 &= \mu - x_i \\ m_2 &= \sigma^2 - (x_i - \mu)^2 \\ m_3 &= x_i^3 - \mu(\mu^2 + 3\sigma^2) \end{aligned}$$

(m_1, m_2) can be directly obtained by the definition of (μ, σ). The third moment condition comes from the third derivative of the moment generating function evaluated at 0.

First we generate normally distributed random numbers and compute the two parameters:

```
> set.seed(123)
> n <- 1000
> x1 <- rnorm(n, mean = 4, sd = 2)
> mean(x1)

[1] 4.032256

> sd(x1)

[1] 1.98339
```

[2] I want to thank Dieter Rozenich for providing this example. Most of what is written in this section come from him.

After, we define the moment function

```
> g <- function(tet, x) {
+     m1 <- (tet[1] - x)
+     m2 <- (tet[2]^2 - (x - tet[1])^2)
+     m3 <- x^3 - tet[1] * (tet[1]^2 + 3 * tet[2]^2)
+     f <- cbind(m1, m2, m3)
+     return(f)
+ }
```

We then run `gmm()` using the starting values $(\mu_0, \sigma_0^2) = (0, 0)$

```
> res <- gmm(g, x1, c(0, 0))
> res$par

Theta[1] Theta[2]
4.037008 1.976157
```

It is however strongly suggested to provide the gradient $G(\theta) = D_\theta \bar{g}(\theta)$ which is a $q \times p$ matrix with typical element $G_{ij} = \partial \bar{g}_i(\theta)/\partial \theta_j$. In our example:

$$\frac{\partial g(\theta, x_t)}{\partial \theta} = \begin{pmatrix} 1 & 0 \\ 2(x_t - \mu) & 2\sigma \\ -3(\mu^2 + \sigma^2) & -6\mu\sigma \end{pmatrix},$$

Which implies that

$$G \equiv \frac{\partial \bar{g}(\theta)}{\partial \theta} = \begin{pmatrix} 1 & 0 \\ 2(\bar{x} - \mu) & 2\sigma \\ -3(\mu^2 + \sigma^2) & -6\mu\sigma \end{pmatrix},$$

and the following function computes it:

```
> Dg <- function(tet, x) {
+     jacobian <- matrix(c(1, 2 * (-tet[1] + mean(x)), -3 * tet[1]^2 -
+         3 * tet[2]^2, 0, 2 * tet[2], -6 * tet[1] * tet[2]), nrow = 3,
+         ncol = 2)
+     return(jacobian)
+ }
```

We can then use it in the `gmm()` algorithm:

```
> res2 <- gmm(g, x1, c(0, 0), grad = Dg)
> res2$par

Theta[1] Theta[2]
4.037008 1.976157
```

The result is not different because the gradient is only used to estimate the asymptotic covariance matrix of $\sqrt{T}(\hat{\theta} - \theta)$ which is $(G'\Omega^{-1}G)^{-1}$. We can compare them using the `summary()` method:

```
> summary(res)$par

         Estimate Std. Error  t value Pr(>|t|)
Theta[1]  4.03701    0.06117 65.99483        0
Theta[2]  1.97616    0.04259 46.40320        0

> summary(res2)$par

         Estimate Std. Error  t value Pr(>|t|)
Theta[1]  4.03701    0.06117 65.99483        0
Theta[2]  1.97616    0.04259 46.40320        0
```

In a simple model like this one, it does not make any difference whether the gradient is provided or not. But it should always be provided when it is possible. It is more reliable.

Example with iid moment conditions

The second example is a linear model with an endogeneity problem. It is the same model used by Carrasco (2007) to compare several methods which deal with the many instruments problem. We want to estimate δ from:

$$y_t = \delta W_t + \epsilon_t$$

with $\delta = 0.1$ and

$$W_t = e^{-x_t^2} + u_t,$$

where $(\epsilon_t, u_t) \sim \ iidN(0, \Sigma)$ with

$$\Sigma = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

Any power of x_t can be used as instrument because it is clearly orthogonal to ϵ_t and correlated with W_t. For the exercise, (x_t, x_t^2, x_t^3) will be the selected instruments. This simple example will allow us to see what `gmm()` can do without worrying about selecting the right parameters for the HAC matrix.

The first step is to generate the model (with $T = 400$):

```
> require(mvtnorm)
> sig <- matrix(c(1, 0.5, 0.5, 1), 2, 2)
> t <- 500
> e <- rmvnorm(t, sigma = sig)
> x2 <- rnorm(t)
> w <- exp(-x2^2) + e[, 1]
> y <- 0.1 * w + e[, 2]
```

where `rmvnorm()` is a multivariate normal distribution random generator which is included in the package **mvtnorm**. For a linear model, we do not need to write a function $g(\theta, x)$. We only need to give the instruments and a formula that characterizes the linear model (see `help(lm)` for more details). In the following, h is a 200×3 matrix of instruments.

```
> h <- cbind(x2, x2^2, x2^3)
> fct <- y ~ w
```

The moment conditions are:

$$E\begin{pmatrix}(y_t-\alpha-\delta W_t)x_t\\(y_t-\alpha-\delta W_t)x_t^2\\(y_t-\alpha-\delta W_t)x_t^3\end{pmatrix}=0$$

where the intercept α is always included. In order the remove the intercept, the option "intercept=FALSE" must be added. Since we should never exclude the intercept, this case will not be considered.

The moment conditions of this example are iid. Therefore, we can add the option `vcov="iid"`. This option tells `gmm()` to estimate the covariance matrix of $\sqrt{T}\bar{g}(\theta^*)$ as follows:

$$\hat{\Omega}(\theta^*)=\frac{1}{T}\sum_{t=1}^{T}g(\theta^*,x_t)g(\theta^*,x_t)'$$

Once `gmm()` executed, the method `summary()` can be applied to the object created by the algorithm. It gives details about the estimation.

```
> res <- gmm(fct, x = h)
> summary(res)

$met
GMM method
 "twoStep"

$kernel
kernel for cov matrix
 "Quadratic Spectral"

$algo
NULL

$par
            Estimate Std. Error  t value Pr(>|t|)
(Intercept) -0.01700    0.10673 -0.15932  0.87342
w            0.22094    0.15388  1.43573  0.15108

$J_test
[1] Test-J degrees of freedom is 2

$j
                 J-test    Pz(>j)
Test E(g)=0:   0.767123 0.6814302

attr(,"class")
[1] "summary.gmm"
```

Notice that the gradient was not provided. In the case of linear models with the function $g(\theta, x)$ replaced by a formula, the computation of the analytical gradient is embedded in `gmm()`. It is therefore note required. Along with the information on the parameters estimates, `summary()` computes the J-test of overidentifying restrictions. It tests the hypothesis $H_0 E(g(\theta, x_t)) = 0$ using the statistics $T\bar{g}(\hat{\theta})'[\hat{\Omega}(\theta^*]^{-1}\bar{g}(\hat{\theta}) \sim \chi^2_{q-p}$.

By default, the 2SGMM is computed. Other methods can be chosen by modifying the option "type". The second possibility is ITGMM:

```
> res2 <- gmm(fct, x = h, type = "iterative")
> summary(res2)$par

            Estimate Std. Error  t value Pr(>|t|)
(Intercept) -0.01035    0.10688 -0.09687  0.92283
w            0.21157    0.15405  1.37335  0.16964

> summary(res2)$j

                 J-test    Pz(>j)
Test E(g)=0:   0.743245 0.6896145
```

where the option `crit=` (default is 10^{-7}) can be set as small as we want to increase the precision and `maxiter=` increased if ITGMM fails to converge. The third method is CUE. As you can see, the estimates from ITGMM is used as starting values. However, the starting values is required only when a function $g(\theta, x)$ is provided instead of a formula. So it is not necessary.

```
> res3 <- gmm(fct, x = h, res2$par, type = "cue")
> summary(res3)$par

            Estimate Std. Error  t value Pr(>|t|)
(Intercept) -0.01216    0.10683 -0.11380  0.90940
w            0.21452    0.15400  1.39303  0.16361

> summary(res3)$j

                 J-test   Pz(>j)
Test E(g)=0:   0.742832 0.689757
```

It is possible to produce confidence intervals by adding the option `interval=a`, where a is the level of confidence. It can be extracted from `summary()` this way:

```
> summary(res3, interval = 0.95)$interval

            Theta_lower Theta_upper
(Intercept)  -0.2215397    0.197226
w            -0.0873064    0.516349
```

It is also possible to do hypothesis tests of the form $H_0 : R\theta = c$ using `lintest()` that comes with the package. It produces a Wald test. For example, if we want to test $H_0 : \alpha = 0, \delta = 0.1$ we would proceed this way:

```
> R <- diag(2)
> c <- matrix(c(0, 0.1), 2, 1)
> lintest(res3, R, c)

[[1]]
[1] "Wald test for H0: R(Theta)=c"

$H0
     Null Hypothesis
[1,] (Intercept) = 0
[2,] w = 0.1

$result
          Statistics   P-Value
Wald test   2.740553 0.2540367
```

A last comment before we go to the next example. When we provide a linear formula to `gmm()`, ITGMM and 2SGMM are computed using an analytical solution. That's why a starting value is not required. However, there exists no analytical solution for CUE. If no starting values are provided, then the analytical solution with the identity as matrix of weights is used by default. However, I believe it is always a good idea to give the best possible starting values, especially with CUE which is known to be a very unstable method. It is also possible to add options to the `optim()` algorithm for better convergence. This is done by adding the option `control=list()`. For example, you can keep track of the convergence with `control=list(trace=TRUE)` or increase the number of iterations with `control=list(maxit=1000)`. You can also choose another optimization algorithm with `method="BFGS"`, for example. See `help(optim)` for more details.

Estimating the AR coefficients of an ARMA process

Now we turn to a time series example in which the computation of the HAC matrix is required. We want to estimate the AR coefficients of the following process:

$$X_t = 1,4X_{t-1} - 0,6X_{t-2} + u_t$$

where $u_t = 0,6\epsilon_{t-1} - 0,3\epsilon_{t-2} + \epsilon_t$ and $\epsilon_t \sim iidN(0,1)$. In other words, X_t is an ARMA(2,2). We can estimate the AR coefficients by using X_{t-s} for $s > 2$ as instruments since they are uncorrelated with u_t. So we choose $(X_{t-3}, X_{t-4}, X_{t-5}, X_{t-6})$ and a sample size of 400. Here again, the choice of the instruments is arbitrary since it is not our goal to talk about the optimal choice. First, we can easily generate an ARMA process using the function `arima.sim()`:

```
> t <- 400
> x3 <- arima.sim(n = t, list(ar = c(1.4, -0.6), ma = c(0.6, -0.3)))
> xt <- x3[7:t]
```

```
> xt1 <- x3[6:(t - 1)]
> xt2 <- x3[5:(t - 2)]
> h2 <- cbind(x3[4:(t - 3)], x3[3:(t - 4)], x3[2:(t - 5)], x3[1:(t -
+     6)])
> fct2 <- xt ~ xt1 + xt2
> res <- gmm(fct2, x = h2)
> summary(res)$par

            Estimate Std. Error   t value Pr(>|t|)
(Intercept) -0.02578    0.06057  -0.42558  0.67041
xt1          1.44274    0.08269  17.44712  0.00000
xt2         -0.65185    0.06442 -10.11863  0.00000

> summary(res)$j

                 J-test    Pz(>j)
Test E(g)=0:   1.556592 0.4591879
```

As it is shown above, the optimal matrix, when moment conditions are based on time series, is an HAC matrix which is defined by equation (3). Usually, the default values work fine and do not need to be modified. But, for those who knows what they are doing and want to explore different options, we will give a brief review[3]. If the reader wants advice on which option is best, he should consult the literature about properties of HAC matrices. It is not the purpose of this article.

First, there are five choices of kernel: Truncated, Bartlett, Parzen, Tukey-Hanning and Quadratic spectral[4]. They can be selected by adding the option `kernel=`. By default, the Quadratic Spectral is used as it was shown to be the optimal kernel by Andrews (1991). However, they have their advantages. For example, the simplicity of the Bartlett kernel proposed by Newey and West (1987a) could be more stable numerically when dealing with highly nonlinear models, especially with CUE. In many cases, the choice does not seem to affect the estimates. We can see this by using the `HAC()` function that comes with the package. It is the function that computes the HAC covariance matrix of $\sqrt{T}\bar{g}(\theta^*)$ from the $T \times q$ matrix $g(\theta^*, x)$. We can also use it to compute the HAC matrix of any vector of sample means of weakly dependent processes. For example, we can estimate the variance of $\sqrt{T}\bar{x}$ of the above $ARMA(2,2)$:

```
> x3 <- matrix(x3, 400, 1)
> HAC(x3)

          [,1]
[1,] 47.68067
```

[3]Tools to compute that matrix are modified versions of functions included in the package **sandwich**. For a complete review of all options, see Zeileis (2006).

[4]The first three have been proposed by White (1984), Newey and West (1987a) and Gallant (1987) respectively and the last two, applied to HAC estimation, by Andrews (1991). But the latter gives a good review of all five.

We can see that the different kernels produce similar estimates of $\sqrt{(T)}\bar{x}$:

```
> HAC(x3, kernel = "Truncated")

        [,1]
[1,] 39.2354

> HAC(x3, kernel = "Bartlett")

         [,1]
[1,] 43.97312

> HAC(x3, kernel = "Parzen")

         [,1]
[1,] 45.11607

> HAC(x3, kernel = "Tukey-Hanning")

         [,1]
[1,] 45.86285
```

where only Truncated seems to deviate from the others. It will be shown below that the Truncated kernel is more useful with the GEL method to smooth the moment conditions. It is rarely used to compute HAC matrices.

The second choice is the bandwidth selection method. By default it is the automatic selection proposed by Andrews (1991). It is also possible to choose the automatic selection of Newey and West (1994) by adding `bw=bwNeweyWest2` (without quotes because bwNeweyWest2 is a function). There are few other options but I will present only the most important (See `help(gmm)` or `help(HAC)` for details on the other ones). A prewhitened kernel estimator can be computed using the option `prewhite=`. By default, it is set to `FALSE`. If you want à prewhitened estimator, you need to select the order of the VAR used to obtain it. For example, if you want a VAR(1), the option `prewhite=1` is required. The argument in favor of using this estimator is that it seems to improve the properties of hypothesis tests on parameters which are computed with the HAC matrix. For more details , see Andrews and Monahan (1992).

We conclude this section with an example on how to modify these options.

```
> res <- gmm(fct2, x = h2, kernel = "Bartlett", prewhite = 1, bw = bwNeweyWest2,
+     type = "iterative")
> summary(res)$par

            Estimate Std. Error  t value Pr(>|t|)
(Intercept) -0.01963    0.06483 -0.30274  0.76209
xt1          1.41697    0.07910 17.91461  0.00000
xt2         -0.63169    0.06401 -9.86811  0.00000
```

Testing the CAPM

The last example is an application in finance. We want to test one of the implications of the capital asset pricing model (CAPM). It is a simple example taken from Campbell, Lo, and Mackinlay (1996). It is a good example because it shows how to apply the GMM algorithm to estimate a system of equations. We want to estimate the following model:

$$(R_t - R_f) = \alpha + \beta(R_{mt} - R_f) + \epsilon_t,$$

where R_t is a $N \times 1$ vector of returns on stocks, R_{mt} if the return of the market portfolio (a proxy of it), R_f is the risk-free rate and ϵ_t if a vector error terms with covariance matrix Σ_t. So ϵ_t can be heteroskedastic and autocorrelated. The data are the daily returns of ten selected stocks taken from Yahoo-Finance and the risk-free rate and returns of the market portfolio that can be found on Kenneth R. French's web site from January 1993 to February 2009[5]. One implication of the CAPM is that the vector α should be zero. So we want to test the hypothesis $H_0 : \alpha = 0$. The instrument is simply $(R_{mt} - R_f)$ which implies that the model is just identified.

```
> data(Finance)
> r <- Finance[1:300, 1:10]
> rf <- Finance[1:300, "rf"]
> rm <- Finance[1:300, "rm"]
> z <- as.matrix(r - rf)
> t <- nrow(z)
> zm <- matrix(rm - rf, t, 1)
> res <- gmm(z ~ zm, x = zm)
> summary(res)$par
```

```
                Estimate Std. Error  t value Pr(>|t|)
WMK_(Intercept) -0.00467    0.05748 -0.08123  0.93526
UIS_(Intercept)  0.10235    0.11846  0.86401  0.38758
ORB_(Intercept)  0.14587    0.21227  0.68721  0.49195
MAT_(Intercept)  0.03590    0.10609  0.33833  0.73511
ABAX_(Intercept) 0.09174    0.27258  0.33657  0.73644
T_(Intercept)    0.02310    0.07582  0.30471  0.76059
EMR_(Intercept)  0.02993    0.05477  0.54641  0.58478
JCS_(Intercept)  0.11680    0.16275  0.71768  0.47296
VOXX_(Intercept) 0.02087    0.17444  0.11965  0.90476
ZOOM_(Intercept)-0.21914    0.19673 -1.11391  0.26532
WMK_zm           0.31719    0.13076  2.42575  0.01528
UIS_zm           1.26272    0.22784  5.54211  0.00000
ORB_zm           1.49391    0.42138  3.54524  0.00039
MAT_zm           1.01499    0.22899  4.43253  0.00001
ABAX_zm          1.08898    0.57801  1.88401  0.05956
T_zm             0.84898    0.16218  5.23482  0.00000
```

[5]The symbols of the stocks taken from http://ca.finance.yahoo.com/ are (WMK, UIS, ORB, MAT, ABAX, T, EMR, JCS, VOXX, ZOOM) and K. R. French's web site is http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html

```
EMR_zm            0.74079    0.10460  7.08231  0.00000
JCS_zm            0.95882    0.35882  2.67213  0.00754
VOXX_zm           1.48217    0.38027  3.89762  0.00010
ZOOM_zm           2.07770    0.31617  6.57140  0.00000

> R <- cbind(diag(10), matrix(0, 10, 10))
> c <- matrix(0, 10, 1)
> lintest(res, R, c)

[[1]]
[1] "Wald test for H0: R(Theta)=c"

$H0
      Null Hypothesis
 [1,] WMK_(Intercept) = 0
 [2,] UIS_(Intercept) = 0
 [3,] ORB_(Intercept) = 0
 [4,] MAT_(Intercept) = 0
 [5,] ABAX_(Intercept) = 0
 [6,] T_(Intercept) = 0
 [7,] EMR_(Intercept) = 0
 [8,] JCS_(Intercept) = 0
 [9,] VOXX_(Intercept) = 0
[10,] ZOOM_(Intercept) = 0

$result
          Statistics   P-Value
Wald test   4.118292 0.9418528
```

3. Generalized empirical likelihood

As for GMM we start by giving a brief review of the method. But since, in general, applied econometricians are not too familiar with it, the presentation will be more detailed. I am not aware of any applied works that use GEL. It is so recent that we are still studying its properties through simulations (see for example Guggenberger and Hahn (2005)) and asymptotic analysis (see Newey and Smith (2004) and Anatolyev (2005)).

3.1. The method

As for GMM, GEL is based on the moment conditions

$$E(g(\theta, x_t)) = 0.$$

In general we can write the sample moment conditions as follows:

$$\tilde{g}(\theta) = \sum_{t=1}^{T} p_t g(\theta, x_t) = 0,$$

instead of the usual $\bar{g}(\theta) = 0$, where p_t is the probability associated with x_t. That's the main difference between GMM et GEL. Because GEL does not restrict all the probabilities to be $(1/T)$, but instead select them in such a way that the moment conditions are satisfied exactly, sample means computed using these probabilities have better properties, at least in theory, under certain assumptions. That includes the sample mean of the Jacobian or the covariance matrix of the moment conditions which are part of the distribution of $\hat{\theta}$[6]. Asymptotically, GEL seems to have better properties than GMM, but Guggenberger and Hahn (2005)) and Guggenberger (2008) show, using simple linear simulations, that it may not be the case in finite sample. I think that most of the bad results come from numerical problems. In fact, I could not reproduce the results from the second paper. This is one of the reasons why I started this package project. With a difficult method like that, we need to have a common tool so that results can be compared. Furthermore, everyone can contribute to improve it since everything is open source and then transparent.

Another difference between GEL and GMM is how they deal with the fact that $g(\theta, x_t)$ can be a conditional heteroskedastic and weakly dependent process. We saw above that in this case GMM simply choose an appropriate covariance matrix estimator. For GEL, Smith (2001) proposes to replace $g(\theta, x_t)$ by:

$$g^w(\theta, x_t) = \sum_{s=-m}^{m} w(s) g(\theta, x_{t-s})$$

where $w(s)$ are kernel based weights that sum to one. This allows GEL to reach the same asymptotic efficiency as GMM (see also Kitamura and Stutzer (1997) and Smith (1997)). The sample moment conditions become:

$$\tilde{g}(\theta) = \sum_{t=1}^{T} p_t g^w(\theta, x_t) = 0 \tag{5}$$

Of course, the above equation has infinitely many solutions since we have (T+p) unknown. So we want to choose p_t as close as possible to (1/T). The metric used as distance is what characterizes the different methods that are part of the family of GEL. We first present the dual of GEL, which allow a better understanding. The estimator is defined as the solution to the following constraint minimization problem:

$$\begin{aligned}
\hat{\theta}_n &= \arg\min_{\theta, p_t} \sum_{t=1}^{T} h_T(p_t), && (6)\\
&\text{subject to} && (7)\\
&\sum_{t=1}^{T} p_t g^w(\theta, x_t) = 0 \quad \text{and} && (8)\\
&\sum_{t=1}^{T} p_t = 1, && (9)
\end{aligned}$$

where $h_T(p_t)$ has to belong to the following Cressie-Read family of discrepancies:

$$h_T(p_t) = \frac{[\gamma(\gamma+1)]^{-1}[(Tp_t)^{\gamma+1} - 1]}{T}.$$

[6]This a very simple way to present the advantages of GEL. see Newey and Smith (2004) but more details

In that case, Smith (1997) showed that the empirical likelihood method (EL) of Owen (2001) ($\gamma = 0$) and the exponential tilting of Kitamura and Stutzer (1997) ($\gamma = -1$) belong to the GEL family of estimators while Newey and Smith (2004) show that it is also the case for the continuous updating estimator (CUEGEL to differentiate with the CUE of gmm) of Hansen *et al.* (1996) ($\gamma = 1$). This family of estimator that we call GEL is defined as the solution to the following saddle point problem:

$$\hat{\theta} = \arg\min_{\theta} \left[\max_{\lambda} \frac{1}{T} \sum_{t=1}^{T} \rho\left(\lambda' g^w(\theta, x_t)\right) \right] \tag{10}$$

where in our case λ is the Lagrange multiplier associated with the constraint (8). $\rho(v)$ depends on the γ in the discrepancy function $h()$. It is a strictly concave function which is normalized so that $\rho'(0) = \rho''(0) = -1$. It can be shown that $\rho(v) = \ln(1 - v)$ corresponds to the EL method, $\rho(v) = -\exp(v)$ to the ET and to CUE if $\rho(v)$ is quadratic.

The estimators are obtain by solving the following system of equations:

$$\sum_{t=1}^{T} p_t g^w(\theta, x_t) = 0, \tag{11}$$

$$\sum_{t=1}^{T} p_t \lambda' \left(\frac{\partial g^w(\theta, x_t)}{\partial \theta} \right) = 0, \tag{12}$$

with

$$p_t = \frac{1}{T} \rho'\left(\lambda' g^w(\theta, x_t)\right). \tag{13}$$

In the method gel() that comes with the package, the numerical algorithm optim() minimizes equation (10), where λ is substituted by the solution of condition (11) which is obtain by applying a Newton method. It gives more control to the user this way. But before playing with the control variables, it is important to know how the iterative method works[7]. The iterative procedure relating λ from iteration $i-1$ to iteration i is given by the following difference equation (g_t is defined as $g^w(\theta, x_t)$ for simplicity):

$$\lambda_i = \lambda_{i-1} - \left[\frac{1}{T} \sum_{t=1}^{T} \rho''(\lambda'_{i-1} g_t) g_t g_t' \right]^{-1} \left[\frac{1}{T} \sum_{t=1}^{T} \rho'(\lambda'_{i-1} g_t) g_t \right]$$

The iterative procedure starts at $\lambda_0 = 0$ because it is its asymptotic value. It stops when $\|\lambda_i - \lambda_{i-1}\|$ reaches a certain tolerance level or if the number of iterations passes a predetermined value. In the first case, we have normal convergence, while in the second we have no convergence.

In order to test the overidentifying restrictions, Smith (2004) proposes three tests which are all asymptotically distributed as a χ^2_{q-p} like the J-test of GMM. In fact one of them is the J-test defined as:

$$T \bar{g}^w(\hat{\theta})' [\hat{\Omega}(\hat{\theta})]^{-1} \bar{g}^w(\hat{\theta}),$$

[7] This Newton method for solving nonlinear system of equations is explained in any textbook on numerical methods.

the second is a Lagrange multiplier test (LM):

$$LM = T\hat{\lambda}'\hat{\Omega}(\hat{\theta})\hat{\lambda}$$

and the last is a likelihood ratio test (LR):

$$LR = 2\left[\sum_{t=1}^{T}\rho\left(\hat{\lambda}'g^w(\hat{\theta}, x_t)\right) - \rho(0)\right]$$

3.2. GEL with R

The same examples will be presented which will allow us to compare the results with those of GMM.

Estimating the parameters of a normal distribution

We start with this simple example. Since the observations are iid, we do not need to smooth the moment functions $g(\theta, x_t)$. It is the default value (`smooth=FALSE`). Notice that we should choose a good starting values. GEL is a very nonlinear method. So the choice of starting values is very important. The best choice is the sample mean and the standard deviation. By default the option `type=` is set to "EL". We need to modify it if we want the other methods.

```
> res_el <- gel(g, x1, c(mean(x1), sd(x1)))
> sres_el <- summary(res_el)
> sres_el$par

         Estimate Std. Error  t value Pr(>|t|)
Theta[1]   4.0347    0.06064 66.53706        0
Theta[2]   1.9799    0.04260 46.48072        0

> sres_el$lambda

           Estimate Std. Error  t value Pr(>|t|)
Lambda[1] -0.11803    0.11611 -1.01658  0.30936
Lambda[2] -0.02357    0.02499 -0.94353  0.34541
Lambda[3] -0.00195    0.00193 -1.00990  0.31254

> sres_el$test

        statistics   p-value
LR test  0.9861826 0.3206772
LM test  1.0159535 0.3134808
J test   0.9724292 0.3240751
```

Each Lagrange multiplier is the shadow price of the constraint implied by moment condition. An unbinding constraint will produce a multiplier close to zero. Therefore, its value informs us on the validity of the moment condition. In the above results, the three $\hat{\lambda}$ are not significant which is consistent with the three tests of overidentifying restrictions. We can compare the results with the other two methods. We start with ET:

```
> res_et <- gel(g, x1, c(mean(x1), sd(x1)), type = "ET")
> summary(res_et)$par

         Estimate Std. Error t value Pr(>|t|)
Theta[1]  4.03460    0.06066 66.5151        0
Theta[2]  1.97829    0.04263 46.4067        0
```

then with CUE:

```
> res_cue <- gel(g, x1, c(mean(x1), sd(x1)), type = "CUE")
> summary(res_cue)$par

         Estimate Std. Error  t value Pr(>|t|)
Theta[1]  4.03391    0.06069 66.46913        0
Theta[2]  1.97644    0.04266 46.32564        0
```

If you compare the last results with CUE (which is not reported here), they are very close. The CUEGEL is the easiest to compute, which is not surprising since in this case $\rho(v)$ is quadratic. On the other hand, EL is the hardest because the domain of $\rho(v)$ is constraint. Indeed, $\rho(v) = \log(1 - v)$, which implies that we need $\lambda' g(\theta, x_t) < 1$ for all t. Therefore, it is much harder to solve for $\hat{\lambda}$. In such problems, it is like trying to find the minimum inside a maze. Each time we get caught in a dead end, we have to go back. We can compare the execution time of each method:

```
> system.time(gel(g, x1, c(mean(x1), sd(x1))))

   user  system elapsed
  1.312   0.004   1.321

> system.time(gel(g, x1, c(mean(x1), sd(x1)), type = "ET"))

   user  system elapsed
  0.224   0.000   0.224

> system.time(gel(g, x1, c(mean(x1), sd(x1)), type = "CUE"))

   user  system elapsed
  0.572   0.004   0.579
```

However, it may be worth waiting, because theoretical results suggest that EL produces estimators with the best properties.

Notice that the method `summary()` produces different output when applied to an object of class "gel" or "gmm" which are produced respectively by `gel()` and `gmm()`. This is the beauty of object oriented programming. The same method can be applied to an object of class "lm", in which case it produces results from an ols estimation or to an object of class "vector" to obtain some statistical properties of the object. Here, the method `summary()` applied to a "gel" object reports the $\hat{\lambda}$, $\hat{\theta}$ along with their standard deviations and also several results about the quality of the solution that we need to understand. We can see all the variables computed by summary by typing `names(summary object)`:

```
> names(summary(res_el))

[1] "type"        "par"         "lambda"      "test"        "badrho"
[6] "conv_par"    "conv_pt"     "conv_moment" "conv_lambda"
```

The first value of interest is "badrho". As Newey and Smith (2004) say, There may be no λ that solves the first order conditions within the domain of $\rho(v)$ in finite sample. We can only say that the probability that it exists goes to one as T goes to infinity. The only method that is susceptible to create that problem is EL. When it happens, the observations for which $\lambda' g(\theta, x_t) \geq 1$ are dropped and "badrho" returns how many there are.

```
> summary(res_el)$badrho

Number_of_bad_rho
                0
```

The second value of interest is "conv_lambda" which returns a message about the convergence quality of the iterative Newton algorithm (see the function `get_lamb()` for details) used to solve for $\hat{\lambda}$. It speaks by itself, so no further explanations are required.

```
> summary(res_el)$conv_lamb

Convergence_code_for_lambda
       "Normal convergence"
```

If the convergence fails, there are options that can be modified such as the number of iterations or the tolerance level. See `help(gel)` and `help(get_lamb)` for more details. Another possible alternative is to let optim() do everything for you by setting the option `optlam="numeric"`. It takes in general more times to compute the results and it is more hazardous. But you have the choice.

The last three values of interest are "conv_par" that returns the convergence code of `optim()` which should be 0 (see `help(optim)`), "conv_pt", defined as $\sum_{t=1}^{T} \hat{p}_t$, which should be close to 1 and "conv_moment" gives $\sum_{t=1}^{T} \hat{p}_t g^w(\hat{\theta}, x_t)$ that should be a vector of zeros.

```
> summary(res_el)$conv_par

Convergence_code_theta
                    10
```

```
> summary(res_el)$conv_moment

     Sample_moment_with_pt
[1,]         -7.226885e-18
[2,]          4.696120e-17
[3,]         -3.946496e-17
```

```
> summary(res_el)$conv_pt
```

```
Sum_of_pt
        1
```

The results show that the solution satisfies all convergence criteria.

A fourth method is available. It is called the exponentially tilted empirical likelihood (ETEL) and was proposed by Schennach (2007). However, it does not belong to the family of GEL estimators. It solves the problem of misspecified models. In such models there may not exist any pseudo values to which $\hat{\theta}$ converges as the sample size increases. ETEL uses the $\rho()$ of ET to solve for λ and the $\rho()$ of EL to solve for θ. It is an appealing alternative to EL because she shows that ETEL shares the same properties as EL but without the difficulties associated with the computation of $\hat{\lambda}$. That's why I included it in the package so we can analyze it further. We conclude this example with this last method.

```
> res <- gel(g, x1, c(mean(x1), sd(x1)), type = "ETEL")
> summary(res)$par

         Estimate Std. Error  t value Pr(>|t|)
Theta[1]  4.05710    0.05972 67.92991        0
Theta[2]  2.00059    0.04232 47.26819        0

> system.time(gel(g, x1, c(mean(x1), sd(x1)), type = "ETEL"))

   user  system elapsed
  0.472   0.000   0.471
```

Example with iid moment conditions

We can cover this example quickly because there is not much to learn that we do not already know, except for the fact that, as for GMM, we can use a formula instead of a function when we estimate a linear model. Here again, we do not need to smooth the moment function because the observations are iid.

Lets start with a complete summary of the results from the EL method. We can use the GMM estimator with the identity matrix as starting values. We have not seen yet how to do it. We only need the define the option `wmatrix="ident"`:

```
> tet0 <- gmm(fct, x = h, wmatrix = "ident")$par
> res <- gel(fct, x = h, tet0)
> summary(res)

$type
Type of GEL
       "EL"

$par
            Estimate Std. Error  t value Pr(>|t|)
(Intercept) -0.01336    0.10267 -0.13014  0.89646
```

```
w            0.21597    0.15017  1.43819  0.15038

$lambda
          Estimate Std. Error  t value Pr(>|t|)
Intercept -0.00381    0.05963 -0.06384  0.94910
h1        -0.04414    0.07990 -0.55250  0.58060
h2         0.00515    0.03960  0.12994  0.89661
h3         0.02234    0.02453  0.91072  0.36244

$test
        statistics   p-value
LR test  0.7970740 0.6713014
LM test  0.8783238 0.6445764
J test   0.7459213 0.6886923

$badrho
Number_of_bad_rho
                0

$conv_par
Convergence_code_theta
                     0

$conv_pt
Sum_of_pt
        1

$conv_moment
     Sample_moment_with_pt
[1,]         -4.426594e-18
[2,]         -9.359714e-19
[3,]          2.777421e-18
[4,]         -2.978168e-18

$conv_lambda
Convergence_code_for_lambda
       "Normal convergence"

attr(,"class")
[1] "summary.gel"
```

The results are not very different from GMM and all the convergence codes are fine. Results from the other EL methods are comparable. We conclude this subsection by showing that the method `lintest()` can also be applied to an object of class "gel".

```
> R <- matrix(c(0, 1), 1, 2)
> c <- 0.1
```

```
> lintest(res, R, c)


[[1]]
[1] "Wald test for H0: R(Theta)=c"

$H0
     Null Hypothesis
[1,] w = 0.1

$result
          Statistics   P-Value
Wald test   0.596409 0.4399515
```

Estimating the AR coefficients of an ARMA process

Now we need to deal with weakly dependent observations, which requires us to smooth the sample moment function. First, we need to set the option `smooth=TRUE`, then we need to select the kernel. Before going to the estimation procedure, we need to understand the relationship between the smoothing kernel and the HAC estimator that will result from this choice. The reason why we need to smooth the moment functions is that GEL estimates the covariance matrix of $\bar{g}(\theta, x_t)$, as if we had iid observations, using the expression $(1/T)\sum_{t=1}^{T}(g_t g_t')$. We can show that substituting g_t by g_t^w in this expression results in an HAC estimator. But there is not a one to one relationship between the smoothing kernel and the kernel that appears in the HAC estimator. For example, we can show that if the smoothing kernel is Truncated, then the kernel in the HAC estimator is the Bartlett. Lets consider the truncated kernel with a bandwidth of 2. This implies that $w(s) = 1/5$ for $|s| \leq 2$ and 0 otherwise. Then, the expression for the covariance matrix becomes:

$$\begin{aligned}
\frac{1}{T}\sum_{t=1}^{T} g_t^w (g_t^w)' &= \frac{1}{T}\sum_{t=1}^{T}\left(\sum_{s=-2}^{2}\frac{1}{5}g_{t+s}\right)\left(\sum_{l=-2}^{2}\frac{1}{5}g_{t+l}'\right), \\
&= \frac{1}{25}\sum_{s=-2}^{2}\sum_{l=-2}^{2}\left(\frac{1}{T}\sum_{t=1}^{T} g_{t+s}g_{t+l}'\right), \\
&= \frac{1}{25}\sum_{s=-2}^{2}\sum_{l=-2}^{2}\hat{\Gamma}_{s-l}, \\
&= \frac{1}{25}\sum_{s=-4}^{4}(5-|s|)\hat{\Gamma}_s, \\
&= \sum_{s=-4}^{4}\left(\frac{1}{5}-\frac{|s|}{25}\right)\hat{\Gamma}_s, \\
&= \sum_{s=-T+1}^{T-1} k_5(s)\hat{\Gamma}_s,
\end{aligned}$$

where $k_5(s)$ is the Bartlett kernel with a bandwith of 5 which is defined as

$$K_5(s) = \begin{cases} 1/5 + |s|/25 & \text{if} \quad |s| \leq 5 \\ 0 & \text{otherwise} \end{cases}.$$

See Smith (2001) for more details. The model will therefore be estimated using the kernel Truncated. As for the HAC matrix in `gmm()`, it is also possible to choose the bandwidth selection method by modifying the option `bw`. Here are the results:

```
> tet0 <- gmm(fct2, x = h2, wmatrix = "ident")$par
> res <- gel(fct2, x = h2, tet0, smooth = TRUE, kernel = "Truncated")
> summary(res)

$type
Type of GEL
       "EL"

$par
            Estimate Std. Error  t value Pr(>|t|)
(Intercept) -0.02754    0.09584 -0.28734  0.77385
xt1          1.43489    0.11538 12.43664  0.00000
xt2         -0.64452    0.09007 -7.15588  0.00000

$lambda
          Estimate Std. Error  t value Pr(>|t|)
Intercept  0.00115    0.02794  0.04134  0.96702
h1         0.08469    0.02722  3.11084  0.00187
h2        -0.22600    0.05465 -4.13506  0.00004
h3         0.21403    0.04856  4.40765  0.00001
h4        -0.05674    0.02075 -2.73471  0.00624

$test
        statistics    p-value
LR test   4.579198 0.10130706
LM test   4.681219 0.09626897
J test    4.464897 0.10726548

$badrho
Number_of_bad_rho
                0

$weights
[1] 1 1

$conv_par
Convergence_code_theta
                     0
```

```
$conv_pt
Sum_of_pt
        1

$conv_moment
     Sample_moment_with_pt
[1,]         -5.163513e-18
[2,]          5.142845e-17
[3,]          5.522655e-18
[4,]         -4.787430e-18
[5,]         -1.016440e-17

$conv_lambda
Convergence_code_for_lambda
       "Normal convergence"

attr(,"class")
[1] "summary.gel"
```

We covered most of what the package can do. There are some more options available that we have not talked about. Those who are interested can consult the help that comes with the package.

4. Conclusion

A complete package to estimate models based on GMM and GEL has been presented. This package is the only one available to apply GEL methods. It also offers more options than most popular statistical packages to implement GMM. Because it is part of the open source software R, it makes it easier for econometricians to contribute to its improvement. If it gains in popularity, it would become a very useful and reliable tool to analyze and understand better GMM et GEL.

References

Anatolyev S (2005). "GMM, GEL, Serial Correlation, and Asymptotic Bias." *Econometrica*, **73**, 983–1002.

Andrews D (1991). "Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimation." *Econometrica*, **59**, 817–858.

Andrews W, Monahan JC (1992). "An Improved Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimator." *Econometrica*, **60**(4), 953–966.

Campbell JY, Lo AW, Mackinlay A (1996). *The Econometrics of Financial Markets*. Princeton University Press.

Carrasco M (2007). "A Regularization Approach to the Many Instruments Problem." *Incomplete*.

Carrasco M, Florens J (2000). "Generalization of GMM to a Continuum of Moment Conditions." *Econometric Theory*, **16**, 655–673.

Chaussé P (2008). "Generalized Emprical Likelihood for a Continuum of Moment Conditons."

Gallant AR (1987). *Nonlinear Statistical Models*. New York: Wiley.

Guggenberger P (2008). "Finite Sample Evidence Suggesting a Heavy Tail Problem of the Generalized Empirical Likelihood Estimator." *Econometric Reviews*, **26**, 526–541.

Guggenberger P, Hahn J (2005). "Finite Sample Properties of the Two-Step Empirical Likelihood Estimator." *Econometric Reviews*, **24**(3), 247–263.

Hamilton JD (1994). *Time Series Analisys*. Princeton University Press.

Hansen L (1982). "Large Sample Properties of Generalized Method of Moments Estimators." *Econometrica*, **50**, 1029–1054.

Hansen L, Heaton J, Yaron A (1996). "Finit-Sample Properties of Some Alternative GMM Estimators." *Journal of Business and Economic Statistics*, **14**, 262–280.

Kitamura Y, Stutzer M (1997). "An Information-Theoretic Alternative to Generalized Method of Moments Estimation." *Econometrica*, **65**(5), 861–874.

Newey W, Smith R (2004). "Higher Order Properties of GMM and Generalized Empirical Likelihood Estimators." *Econometrica*, **72**, 219–255.

Newey W, West K (1994). "Automatic Lag Selection in Covariance Matrix Estimation." *Review of Economic Studies*, **61**, 631–653.

Newey WK, West KD (1987a). "A Simple, Positive Semi-definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix." *Econometrica*, **55**, 703–708.

Owen AB (2001). *Empirical Likelihood*. Chapman and Hall.

Schennach Susanne M (2007). "Point Estimation with Exponentially Tilted Empirical Likelihood." *Econometrica*, **35**(2), 634–672.

Smith R (2004). "GEL Criteria for Moment Condition Models."

Smith RJ (1997). "Alternative Semi-Parametric Likelihood Approaches to Generalized Method of Moments Estimation." *The Economic Journal*, **107**, 503–519.

Smith RJ (2001). "GEL Criteria for Moment Condition Models." *Working Paper, University of Bristol*.

White H (1984). *Asymptotic Theory for Econometricians*. New York: Academic Press.

Zeileis A (2006). "Object-oriented Computation of Sandwich Estimator." *Journal of Statistical Software*, **16**(9), 1–16.

Affiliation:

Pierre Chaussé
Département des sciences économiques
Université du Québec à Montréal
315, Ste-Catherine Est, Montréal, (Québec), Canada
E-mail: pierre.chausse@uqam.ca
URL: http://www.er.uqam.ca/nobel/k34115/