gretl working papers

Bayesian Averaging of Classical Estimates (BACE) for **gretl**

Marcin Błażejowski

WSB University in Torun

marcin.blazejowski@wsb.torun.pl

Jacek Kwiatkowski

Nicolaus Copernicus University in Toruń

jkwiat@umk.pl

working paper #6

August 31, 2018

Abstract

This paper presents a software package that implements Bayesian Averaging of Classical Estimates **BACE** ver. 1.1 for **gretl** (the GNU regression, econometrics and time-series library).

1 Introduction

The article Xavier Sala-i-Martin and Gernot Doppelhofer and Ronald I. Miller (2004) suggests a technique for averaging the parameter estimates – Bayesian Averaging of Classical Estimates, which enables to measure the importance of particular potential regressors. This approach is alternative to the familiar and earlier-applied Bayesian Model Averaging (BMA) technique, from which it differs mainly because of its less restrictive *a priori* assumptions. The authors showed that a weighting method may be understood as a limiting case of Bayesian analysis when the prior information is dominated by the data. Averaging of the parameter estimates is performed across all possible combinations of models obtained by means of Classical Ordinary Least Squares (OLS) estimation.

The BACE approach has several interesting features in relation to the techniques such as BMA or robustness-checking methods. First of all, it is not necessary to make assumptions concerning the *a priori* distribution for parameters, and only one simple assumption, on the expected model size.

By applying diffuse priors we obtain the posterior odds ratio for two regression models:

$$\frac{P(M_0|y)}{P(M_1|y)} \approx \frac{P(M_0)}{P(M_1)} \left(\frac{|A_0|/|A_0 + X_0'X_0|}{|A_1|/|A_1 + X_1'X_1|} \right)^{-1/2} \left(\frac{SSE_0 + Q_0}{SSE_1 + Q_1} \right)^{-T/2}, \tag{1}$$

where

- $P(M_i)$ and $P(M_i|y)$ are the prior and posterior probabilities, respectively, for model M_i;
- X_0 and X_1 are matrices with the observations for the independent variables for M_0 and M_1; SSE_i is the OLS sum of squared errors for model i;
- T is the sample size;
- A_0^{-1} and A_1^{-1} are covariance matrices;
- Q_i is a quadratic form of estimated parameters.

If we use Zellner's g-prior Zellner (1986) for $A_0 = gX_0{}'X_0$ and $A_1 = gX_1{}'X_1$ and assume that $g \to 0$ and $X^TX \to \infty$ we will finally obtain the following posterior odds ratio:

$$\frac{P(M_0|y)}{P(M_1|y)} = \frac{P(M_0)}{P(M_1)} T^{(k_1-k_0)/2} \left(\frac{SSE_0}{SSE_1} \right)^{-T/2}, \tag{2}$$

where k_i is the number of parameters in model M_i. We can calculate posterior probability of $P(M_j|y)$ via the following formula:

$$P(M_j|y) = \frac{P(M_j)T^{-k_j/2}SSE_j^{-T/2}}{\sum_{i=1}^{2^K} P(M_i)T^{-k_i/2}SSE_i^{-T/2}}. \tag{3}$$

where 2^K denotes the total number of potential combinations of the K independent variables.

Using BACE we can also easily evaluate the mean and variance of the posterior distribution of slope parameters (see Leamer (1978)):

$$E(\beta|y) = \sum_{i=1}^{2^K} P(M_j|y)E(\beta|y, M_j) \tag{4}$$

Equation (4) can be rewritten as:

$$E(\beta|y) = \sum_{i=1}^{2^K} P(M_j|y)\hat{\beta} \tag{5}$$

where $\hat{\beta} = E(\beta|y, M_j)$ is the OLS estimate of β from model M_j. The posterior variance of β is given by:

$$Var(\beta|y) = \sum_{i=1}^{2^K} P(M_j|y)Var(\beta|y, M_j) + \sum_{i=1}^{2^K} P(M_j|y)\left(\hat{\beta}_j - \sum_{i=1}^{2^K} P(M_j|y)\hat{\beta}_j\right)^2. \tag{6}$$

An example of the application of the BACE technique for forecasting macroeconomic indicators can be found, among others, in the articles van Dijk (2004); Białowolski et al. (2014); Albis and Mapa (2014).

2 Usage of the BACE package

2.1 BACE main function

The BACE package defines two public functions: `BACE_GUI()` and `BACE_Print()`. The `BACE_GUI()` takes following parameters:

- **Dependent variable** as series [required].
- **Y lags** as integer with zero (only for time-series).
- **List of independent variables (X)** as list [may be null].
- **Constant in model** as integer [required]:
  1. Never – all models without constant,
  2. Always – all models with constant,
  3. Can be dropped – constant may be removed from or added to any model.
- **Model prior** as integer [required]:
  1. Binomial – binomial distribution,
  2. Binomial-Beta – binomial-beta distribution[1].
- **Prior average model size** as scalar with zero [required].
- **Significance level for the initial model** as scalar from $(0, 1)$ [required].
- **Number of the top ranked models** as integer [required].
- **Jointness analysis** as integer with zero [required][2]:
  1. None – we do not perform any jointness analysis,
  2. Ley-Steel Measure – we perform jointness analysis with Ley-Steel measure,

[1] Note: this option may generate errors as this distribution follows Occam's razor principle in a very aggressive way.

[2] Values ±999.000000 should be interpreted as $\pm\infty$.

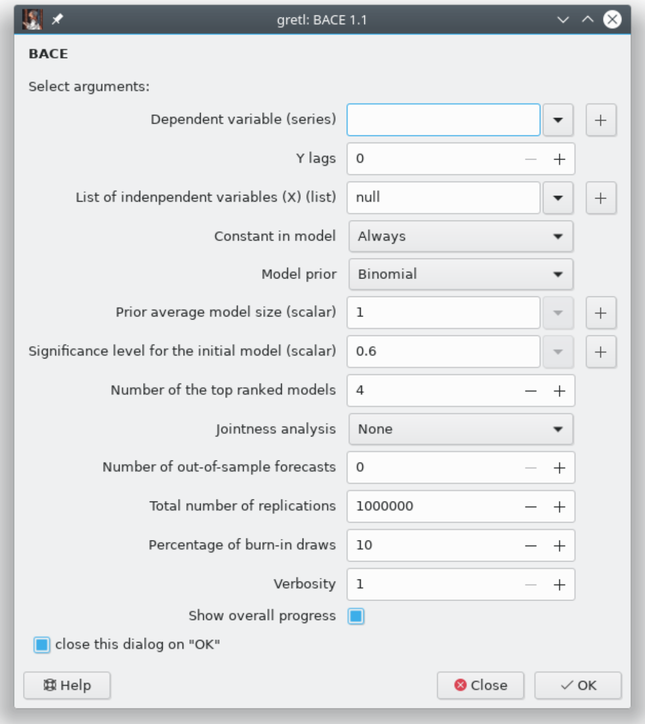
3. Doppelhofer-Weeks Measure – we perform jointness analysis with Doppelhofer-Weeks measure.

- **Number of out-of-sample forecasts** as integer with zero [required].
- **Total number of replications** as integer [required].
- **Percentage of burn-in draws** as integer from (0, 99) [required].
- **Verbosity** as integer [required].
- **Show overall progress** as Boolean.

The BACE_GUI() function returns a bundle with all information needed to print results anytime. One can print the results by calling BACE_Print(&RES), where RES is the bundle returned by the BACE_GUI() function.

2.2 The GUI way

Once you start **gretl**, you must open a data file and then you can load the relevant **BACE** package from the **gretl** server. In the main window, go to File > Function packages > On server heading. By selecting **BACE**, you will download the package on your local machine. Next, go to File > Function packages > On local machine and by selecting BACE from the list open a window similar to the one shown in Figure 1:



gretl: BACE 1.1

BACE

Select arguments:

Dependent variable (series)	
Y lags	0
List of indenpendent variables (X) (list)	null
Constant in model	Always
Model prior	Binomial
Prior average model size (scalar)	1
Significance level for the initial model (scalar)	0.6
Number of the top ranked models	4
Jointness analysis	None
Number of out-of-sample forecasts	0
Total number of replications	1000000
Percentage of burn-in draws	10
Verbosity	1
Show overall progress	☑

☑ close this dialog on "OK"

Figure 1: Main window for **BACE**.

According to Figure 1, we can specify the following entries in the GUI **BACE** window

- **Dependent variable** – The dependent variable.
- **Y lags** – number of lags for dependent variable (for time-series only).
- **List of independent variables** – Loading variables from the database, which must have been opened previously. Note that by default we assume that you want to estimate an intercept; therefore, a constant is implicitly included in the list of the variables.
- **Constant in model** – Do we include intercept? It can be: always, never or can be dropped.
- **Model prior** – Indicates the choice of model prior. One can employ the binomial model prior or the binomial-beta model prior. Note that the uniform model prior is a special case of the binomial model prior. Therefore, in fact, our package allows for three types of priors.
- **Prior average model size** – Specifies the prior expected model size $\mathsf{E}(\Xi)$. The expected model size may range from 0 to K. The default value equals 1. It is the smallest reasonable prior expected model size. It means that we penalize large models and we assign high probability to small ones following Occam's razor principle.

  Note that for the binomial model prior and with $\mathsf{E}(\Xi) = 0.5K$ one can define the uniform prior on the model space. For example, for $K = 10$ regressors we can define the uniform prior with $\mathsf{E}(\Xi) = 5$. For $\mathsf{E}(\Xi) < 5$ we assign high probability to small models. The smaller the prior expected model size $\mathsf{E}(\Xi)$, the less probable are larger models.
- **Significance level for the initial model** – Defines the significance level which is used to build the initial model. An explanatory variable enters the initial model if its p value is less than the significance level. If the significance level equals 1, the initial model will be randomly chosen (with equal probability) from all available models. Note that if all available explanatory variables enter the initial model, you will get the following **gretl** error message "No independent variables were omitted".
- **Number of top ranked models** – Specifies the number of best models for which detailed information is stored.
- **Jointness analysis** – If `None` (the default), the jointness analysis is omitted. Alternatively, one can choose the jointness measures of Ley and Steel (2007) or Doppelhofer and Weeks (2009).
- **Number of out-of-sample forecasts** – Defines the total number of out-of-sample forecasts of the dependent variable.
- **Total number of replications** – Defines the total number of iteration draws to be sampled.
- **Percentage of burn-in draws** – Specifies the number of burn-in replications, calculated as the percentage of the total number of iteration draws.
- **Verbosity** – An integer value of 1 or 2; the default is 1, which allows to see the basic Bayesian Averaging of Classical Estimates results. If Verbosity equals 2, a more detailed description of the analysis is provided (initial model, speed of convergence, estimation results for top ranked models).

- **Show overall progress** – The output is flushed every 10% of MC^3 and the progress is printed. Note: this consumes a lot of CPU time.

2.3 The script way

2.3.1 General remarks

The **BACE** package can also be used inside `Hansl` scripts.

```
open greene9_1.gdt
include BACE.gfn
set seed 1000000

list greene = capital labor nfirm

scalar y_lags = 0
scalar with_const = 1
scalar model_prior = 1
scalar prior_avg_model_size = 1.5
scalar alpha = 0.6
scalar top_ranked_model = 4
scalar joint = 1
scalar forecasts = 0
scalar Nrep = 2000
scalar Nburn = 10
scalar verbosity = 1
scalar progress = 1

BACE_GUI(valadd, y_lags, greene, with_const, model_prior, prior_avg_model_size,
    alpha, top_ranked_model, joint, forecasts, Nrep, Nburn, verbosity, progress)
```

The above example consists of three blocks. The first block is just opening of the so called `greene9_1` dataset, which is bundled in every standard gretl installation. This dataset contains cross-sectional data on manufacturing of transportation equipment presented as Table 9.1 in Greene (1999)[3]. We also set the seed in order to exact reproduction of the results.

The second block is the definition of the `greene` list which contains three independent variables available in the `greene9_1` dataset.

The third block contains the definition for Bayesian Averaging of Classical Estimates analysis: no lags for Y variable (not applicable since we have cross-sectional data), constant always included, binomial model prior, prior average model size set to 1.5 (which means that we have uniform model prior), significance level for the initial model set to 0.6, 4 top ranked models, jointness analysis with Ley-Steel Measure, without out-of-sample predictions, 10000 replications with 10% burn-in draws, basic output (verbosity set to 1) and progress overall progress is printed.

Suppose we want to run the same analysis but following Occam's razor principle. The simplest way is setting the prior average model size to 1. We also want to have ”random” constant, i.e. intercept can be dropped in any model. We also want to save results of the BACE analysis in the bundle `results_b`, the code should be as follows:

```
prior_avg_model_size = 1
with_const = 2
```

[3] This dataset is also available at link http://people.stern.nyu.edu/wgreene/Text/Edition7/TableF7-2.txt.

```
results_b = BACE_GUI(valadd, y_lags, greene, with_const, model_prior,
    prior_avg_model_size, alpha, top_ranked_model, joint, forecasts,
    Nrep, Nburn, verbosity, progress)
```

Suppose we want to calculate Doppelhofer-Weeks Measure instead of already computed Ley-Steel Measure. Since the type of jointness measure is controlled by `do_join` field in the result bundle, the only thing we have to do is to set the value of `do_join` to 2 and print BACE results again. The code should be as follows:

```
results_b.do_joint = 2
BACE_Print(&results_b)
```

2.4 Output

If you select the appropriate entries in the GUI **BACE** window, our package returns: Posterior Inclusion Probabilities (PIP), the averages of the posterior means and standard deviations of parameters ("Mean" and "Std.Dev.", respectively) and the posterior mean and standard deviation of each coefficient conditional on the variable being included in the model ("Cond.Mean" and "Cond.Std.Dev."). Furthermore, the package also returns the predictive results for the dependent variable ("Mean" and "Std.Dev.").

Let us consider the data used in previous example (`australia.gdt`). This dataset contains the Australian macro data used in Johansen (1995) and originally supplied by Tony Hall. The variables used in the text are:

- lpau – log of Australian CPI (dependent variable),
- lpus – log of US CPI,
- le – log of exchange rate (price of US dollars in Aus. dollars),
- iau2 – 5-year Treasury bond rate, Australia,
- ius2 – 5-year Treasury bond rate, USA.

Suppose we want to perform BACE analysis up to $ADL(2,2)$ specification, i.e. all possible combinations from $ADL(0,0)$ to $ADL(2,2)$. The example script would be as follows:

```
set verbose off
include BACE.gfn
open australia.gdt
genr time

# We create lagged values
adl_lags = 2
lags(adl_lags, ius2)
lags(adl_lags, iau2)
lags(adl_lags, e2)
lags(adl_lags, lpus)
lags(adl_lags, le)

# We define list with all X-es (including lags)
list X_list = time ius2 ius2_* iau2 iau2_* e2 e2_* lpus lpus_* le le_*
```

```
set seed 1000000
scalar y_lags = 2
scalar with_const = 2
scalar model_prior = 1
scalar prior_avg_model_size = (nelem(X_list) + y_lags + (with_const > 0)) / 2
scalar alpha = 0.6
scalar top_ranked_model = 4
scalar joint = 0
scalar forecasts = 4
scalar Nrep = 20000
scalar Nburn = 10
scalar verbosity = 2
scalar progress = 0

RES = BACE_GUI(lpau, y_lags, X_list, with_const, model_prior, prior_avg_model_size,
    alpha, top_ranked_model, joint, forecasts, Nrep, Nburn, verbosity, progress)
```

In the above analysis all variables (including dependent variable $lpau_t$) are lagged by 2 periods. Additionally we compute 4 forecasts:

```
Posterior moments (unconditional and conditional on inclusion):
                 PIP          Mean        Std.Dev.    Cond.Mean  Cond.Std.Dev
lpau_1     1.000000      0.898422      0.088271      0.898422      0.088271
  ius2     0.997000     -0.416877      0.099906     -0.418132      0.097400
lpus_2     0.809944     -0.314702      0.206717     -0.388548      0.155135
lpus_1     0.809722      0.375109      0.250558      0.463256      0.191565
  iau2     0.734667      0.220749      0.163020      0.300475      0.110535
  le_2     0.486278     -0.025183      0.043186     -0.051788      0.049573
 const     0.380944     -0.070362      0.121555     -0.184705      0.132918
  e2_2     0.313611      0.000893      0.031421      0.002848      0.056059
  le_1     0.289333     -0.023608      0.062658     -0.081595      0.094008
  e2_1     0.259778      0.015060      0.051931      0.057971      0.088847
  lpus     0.259333      0.030558      0.100080      0.117835      0.168338
    le     0.232389      0.010776      0.043065      0.046368      0.079563
iau2_1     0.231833      0.038012      0.094594      0.163963      0.133962
lpau_2     0.228389      0.034060      0.083090      0.149130      0.114319
  time     0.227500      0.000056      0.000259      0.000245      0.000498
ius2_1     0.177833      0.022368      0.075481      0.125778      0.137952
    e2     0.176944     -0.005384      0.035227     -0.030426      0.079064
ius2_2     0.140056     -0.010130      0.049214     -0.072329      0.113114
iau2_2     0.130111      0.005367      0.040834      0.041247      0.106466

----------------------------------

Posterior probability of models:
Model 1:        0.018944
Model 2:        0.016278
Model 3:        0.015056
Model 4:        0.011889
Total probability of the models in ranking (numerical): 0.062167
```

```
Correlation coefficient between the analytical
and numerical probabilities of the above models: 0.997955

----------------------------------

Predictive results:
                  lpau           Mean        Std.Dev.
1990:2        5.334649       5.340560       0.007958
1990:3        5.341856       5.356856       0.008412
1990:4        5.368310       5.378740       0.010150
1991:1        5.366443       5.395888       0.013844
```

References

Albis MLF, Mapa DS. 2014. Bayesian Averaging of Classical Estimates in Asymmetric Vector Autoregressive (AVAR) Models. MPRA Paper 55902, University Library of Munich, Germany.
URL https://ideas.repec.org/p/pra/mprapa/55902.html

Białowolski P, Kuszewski T, Witkowski B. 2014. Bayesian averaging of classical estimates in forecasting macroeconomic indicators with application of business survey data. *Empirica* **41**: 53–68.
URL https://ideas.repec.org/a/kap/empiri/v41y2014i1p53-68.html

Doppelhofer G, Weeks M. 2009. Jointness of Growth Determinants. *Journal of Applied Econometrics* **24**: 209–244.

Greene W. 1999. *Econometric Analysis*. Upper Saddle River: Prentice Hall, 4th edition.

Johansen S. 1995. *Likelihood-Based Inference in Cointegrated Vector Autoregressive Models*. Oxford University Press.

Leamer E. 1978. *Specification Searches*. New Jersey, USA: John Wiley & Sons. ISBN 0471015202.

Ley E, Steel M. 2007. Jointness in Bayesian Variable Selection with Applications to Growth Regression. *Journal of Macroeconomics* **29**: 476–493.

van Dijk D. 2004. Forecasting US Inflation Using Model Averaging. Econometric Society 2004 Australasian Meetings 143, Econometric Society.
URL https://ideas.repec.org/p/ecm/ausm04/143.html

Xavier Sala-i-Martin and Gernot Doppelhofer and Ronald I Miller. 2004. Determinants of Long-Term Growth: A Bayesian Averaging of Classical Estimates (BACE) Approach. *American Economic Review* **94**: 813–835.
URL https://ideas.repec.org/a/aea/aecrev/v94y2004i4p813-835.html

Zellner A. 1986. On Assessing Prior Distributions and Bayesian Regression Analysis with g-Prior Distributions. In Goel P, Zellner A (eds.) *Bayesian Inference and Decision Techniques: Essays in Honor of Bruno de Finetti*. Amsterdam, Holland: Elsevier. ISBN 978-04-4487-712-3.