STAMP 8.30 Batch and Ox code generator

Siem Jan Koopman

March 5, 2010

1 Introduction

The new update of the STAMP program is given version 8.30 and it contains the facility to generate Ox code for the model that is estimated in STAMP. It complements the Batch code generator in STAMP. It is particularly useful for those who use Ox regularly as their programming environment for analysing time series. In this short overview of the Batch and Ox code generators in OxMetrics for the STAMP program, we will present some examples of their use. We will consider the basic structural time series model with unobserved components (UC) trend, seasonal and irregular and use it for the analysis of the time series with missing entries "ofuELl" from the STAMP data set "ENERGYmiss.in7". The decomposition of the series is graphically presented in Figure 1.

2 Generating batch code

After STAMP has estimated the parameters of the BSM, the standard output is sent to the Text/Results and the Graphics/Model windows. The graphical output is as presented in Figure 1. The batch option is activated from the Model/Batch option (Alt-B). It shows the window which presents the Batch code as given by

```
// Batch code for UC( 1)
module("STAMP");
package("UCstamp");
usedata("ENERGYmiss.in7");
system
{
    Y = ofuELl;
```

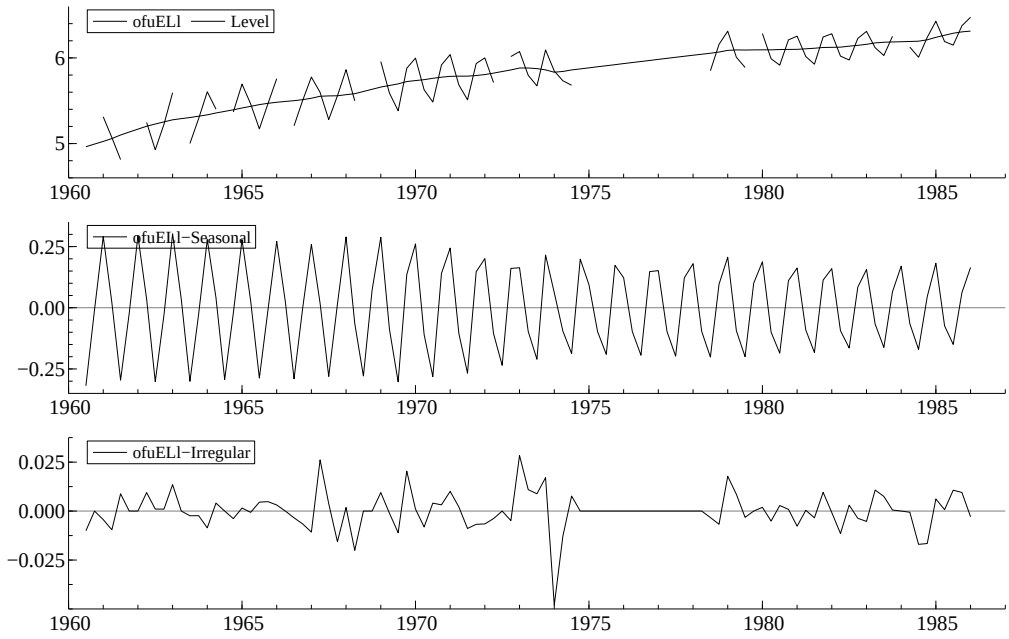

Figure 1: Quarterly UK electricity consumption between 1960 and 1986: millions of useful therms for "Other final users". The decomposition from STAMP is based on a basic structural time series model with trend, seasonal and irregular components. Data source: UK Department of Energy

```
}
setcmp("level", 1, 0.000387097, 0, 0);
setcmp("slope", 1, 1.87978e-006, 0, 0);
setcmp("seasonal", 4, 0.00017307, 0, 0);
setcmp("irregular", 0, 0.000524284, 0, 0);
setmodel();
estimate("ML", 1960, 3, 1986, 1);
```

The code represents the model in STAMP after maximum likelihood estimation. The commands "module" and "package" are required to start the STAMP module in OxMetrics. The command "usedata" loads the data file "ENERGYmiss.in7" while the command "system" assigns the variable ofuELl as the dependent variable that we want to analyze. The unobserved components model is constructed by the commands "setcmp" which introduces the components level with slope (trend), seasonal and irregular. The third arguments are the estimated variances for the components. The model formulation for our unobserved components model is completed by the command "setmodel". Although the variances in the "setcmp" commands are the ones estimated by STAMP (using the method of maximum likelihood), the Batch facility offers the command "estimate" to (re-)estimate the variances by maximum likelihood ("ML") or expectation-maximization ("EM") or another STAMP estimation method. To run the Batch code, click on the Run button (Alt-R) and the STAMP program will re-estimate the UC model

In case we prefer a deterministic seasonal component, we can fix the seasonal variance by setting the variance to zero, that is 0.0, via the Batch code

```
...
setcmp("level", 1, 0.000387097, 0, 0);
setcmp("slope", 1, 1.87978e-006, 0, 0);
setcmp("seasonal", 4, 0.0, 0, 0);
setcmp("irregular", 0, 0.000524284, 0, 0);
setmodel();
estimate("ML", 1960, 3, 1986, 1);
```

When the variance is set to zero in the "setcmp" command, the STAMP program will treat the component as deterministic (the component is fixed over time). A part of the STAMP output in the Results window reports the maximum likelihood estimates of the variances:

```
Variances of disturbances:
                    Value    (q-ratio)
```

```
Level             0.000000  (   0.0000)
Slope         1.32900e-006  (0.0002261)
Seasonal          0.000000  (   0.0000)
Irregular       0.00587911  (    1.000)
```

It follows that the Level component is also estimated as zero which leads to a smooth trend component in STAMP.

Other useful Batch commands are "intervention" for including interventions in the model, "forecast" for generating forecasts from the model, "store" for storing residuals and estimated components from STAMP and "teststate" for printing the estimated state vector and related output. These commands are documented in the STAMP manual.

3 Generating Ox code

An alternative but a more flexible method of running STAMP in batch is offered in version 8.30 by means of the Ox code generator facility. It is activated by pressing Alt-O when STAMP is actived. In case a model is formulated in STAMP and parameters are estimated, the option Alt-O opens the menu window "Generate Ox code" in which the user has two options. The default choice is "Most recent model" and can be accepted. STAMP then outputs the following Ox code

```
#include <oxstd.h>
#import <packages/stamp/stamp_ox_uc>
main()
{
    //--- Ox code for UC( 1)
    decl model = new UCstamp();

    model.Load("ENERGYmiss.in7");
    model.Select(Y_VAR, {"ofuELl", 0, 0});
    model.SetSelSample(1960, 3, 1986, 1);
    model.SetMethod(M_ML);

    model.StartStamp();
    model.AddComponent(COM_LEVEL, 1, 0.000387097);
    model.AddComponent(COM_SLOPE, 1, 1.87978e-006);
    model.AddComponent(COM_SEASONAL, 4, 0.00017307);
    model.AddComponent(COM_IRREG, 0, 0.000524284);
    model.Estimate();
```

```
    delete model;
}
```

When STAMP is installed on the computer, the "stamp_ox_uc" library offers many Ox functions that are developed for STAMP. These functions are collected in the class "UCstamp" which is activated by the "new" command in Ox. The command "Load" reads in the data file "ENERGYmiss.in7" and "Select" takes the variable ofuELl as the dependent variable to analyze. In the Ox code, we can first select the sample and the estimation method using the commands "SetSelSample" and "SetMethod", respectively.

The command "StartStamp" initializes the settings for formulating an UC model. The next part of the Ox code is similar to the Batch code. The inclusion of a component is established by the command "AddComponent". In case of the seasonal component, the constant "COM_SEASONAL" indicates that the seasonal component is included in the model. The second constant indicates that we work with quarterly data (seasonal length is 4) and the value 0.00017307 is the value of the seasonal variance, as estimated by the STAMP program. The "Estimate" command (re-)estimates the variances (and, possibly, other parameters).

Other commands in the "stamp_ox_uc" library are "AddIntervention" for including interventions in the model, "GetForecast" for generating forecasts from the model, "Store" for storing residuals and estimated components from STAMP and "PrintState" for printing the estimated state vector.

4 Creating a link with SsfPack

For SsfPack users, the Ox code generator can facilitate a convenient link with the SsfPack functions. We can illustrate the link with the following extended example Ox code.

```
#include <oxstd.h>
#include <packages/ssfpack/ssfpack.h>
#import <packages/stamp/stamp_ox_uc>
main()
{
    decl model = new UCstamp();
    //...
    model.Estimate();

    decl dlik, dvar, vy, mphi, momega, msigma;
```

```
    vy = model.GetDataY();
    model.GetSsf(&mphi, &momega, &msigma);

    SsfLikEx(&dlik, &dvar, vy, mphi, momega, msigma);
    println("The SsfPack loglikelihood value is ", dlik);

    delete model;
}
```

The Ox code includes the SsfPack library of functions. All necessary variables for SsfPack can be obtained from the "stamp_ox_uc" library. The command "GetDataY" returns the data vector of the dependent variable (as a row vector). The command "GetSsf" returns the state space system matrices of the UC model. The system matrices are required for the SsfPack function. We illustrate it for the computation of the loglikelihood value using the SsfPack function "SsfLikEx". The output is given by

```
The SsfPack loglikelihood value is 87.6671
```

All other SsfPack functions can be used in this framework as well and we refer to the SsfPack manual for a full description of SsfPack.