

How to use the files in reform.zip

Andrey Launov^(*) and Klaus Wälde^(*),¹

^(*)University of Mainz, Université catholique de Louvain and CESifo

May 21, 2010

This file provides information on how to work with the matlab code contained in `reform.zip`. It describes how to install the files, start the programme and obtain numerical solutions in section 1. Section 2 describes how to create figures using existing numerical solutions. Section 3 discusses some robustness issues.

The description is relatively short. More detailed explanations can be provided should questions arise.

1 Obtaining numerical solutions

1.1 Technical preparation

Unzip the files by preserving the structure of the subdirectories. Lets assume the directory in which you save the files and the subdirectory is called `reform`. Then open matlab and go into the directory `reform`.

1.2 Starting the programme

Type `hyb` in its appropriate structure:

```
% structure of hyb(.) is as follows
%
% hyb('a', 'b', c, d)
% where a needs to equal either bUA, sBar or bUA_sBar
% b needs to equal 81a or 81b
% 0 < c < 82 and d = .2 or d = .8
```

An example is `hyb('bUA_sBar', '81a', 8, .8)`. The programme should now be running.

1.3 Possible results

The programme creates many figures in the directory `paras\graphics` and - for later analysis - data files in the directory `paras`.

- Data files

Typical data files are named

`reduce_bUA_81a_81.mat`

`reduce_bUA_sBar_81a_81.mat`

`reduce_bUA_sBar_81b_81.mat`

or any other structure which is described by the command

¹Both authors are at the Mainz School of Management and Economics, University of Mainz, Jakob-Welder-Weg 4, 55131 Mainz, Germany. Fax + 49.6131.39-23827. andrey.launov@uni-mainz.de, phone + 49.6131.39-23233. klaus.waelde@uni-mainz.de, www.waelde.com, phone + 49.6131.39-20143.

```
WhereToSave = sprintf('%s%s%s%s%s%g%s', 'reduce_', WhatRed, '_', ...
BootstrapSelector, '_', maxLoop, '.mat');
at the end of the file hyb.m
```

- Figures

When `hyb()` has been started, many figures will be created automatically at the end of the programme by the commands

```
microdynamics(WithGrowth, WhatRed, BootstrapSelector, trick, MicroFigures, ...
PRSS_x, PRSS_dmx, PRSS_eta, PRSS_phi, PRSS_mue, PRSS_mue0, PRSS_mueb1, PRSS_Vbs, PRSS_y0)
ComparativeStaticPlots(WithGrowth, WhatRed, BootstrapSelector, redB, ...
DataSelector, trick, PRSS_U, go_sbars, go_welfare.', go_U_rate.', go_phi_0.', ...
go_wage.', go_th.', go_taxrate.', go_V_b1_0.', go_V_OneYear.', go_V_w.', ...
go_firmvalue.', go_b2s, go_mue_bar.', go_x(:, maxLoop))
```

These functions create the figures, display them on the screen and save them in `paras\graphics`. The function `microdynamics` does not execute further functions at its end, the function `ComparativeStaticPlots` starts `PlotsWithConfInt` (i.e. plots with confidence intervalls). After this function, the programme stops.

1.4 Creating 'complete' datasets

- The dataset

Due to the physical structure of our server (see <http://www.macro.economics.uni-mainz.de/221.php>), `hyb()` is written such that it is most useful to run e.g.

```
hyb('bUA_sBar', '81a', 81, .8) and, simultaneously on the other processor,
hyb('bUA_sBar', '81b', 81, .8).
```

This implies two datasets

```
reduce_bUA_sBar_81a_81.mat
reduce_bUA_sBar_81b_81.mat
```

These datasets need to be combined by calling the script `zusammen.m` which is in the sub-directory '3 analyse'. Make sure that `WhatRed` and `BootstrapSelector` is appropriately chosen in the first lines. Towards the end of `zusammen.m`, the command

```
save(sprintf('%s%s%s', 'reduce_', WhatRed, '_gesamt.mat'))
```

saves the resulting file. In this case, we would obtain

```
reduce_bUA_sBar_gesamt.mat
```

This file contains raw data in the sense that no normalization took place.

- Figures

The script automatically starts the function `ComparativeStaticPlots` at the end. This function works with normalized data - see `normalize().m`. Again, the function `ComparativeStaticPlots` starts `PlotsWithConfInt` and stops afterwards. Hence, joining datasets automatically creates the figures used in the paper.

2 Access to data and creating figures again

If for any reason data needs to be checked, it can be opened by using the `open` or `load` command. Figures can be created or looked at again by loading data and starting the corresponding functions.

2.1 Plots for comparative statics

The following programme calls work

1. go to subdirectory `paras`
`load reduce_bUA_sBar_gesamt.mat`
go to subdirectory '3 analyse'
`trick = [1 1];`
`DataSelector = 3;`
`ComparativeStaticPlots(WithGrowth,WhatRed,BootstrapSelector,redB,...`
`DataSelector,trick,U,sbars,welfare,U_rate,phi_0,wage,th,taxrate,V_b1_0,...`
`V_OneYear,NaN,NaN,NaN,V_w,firmvalue,b2s,mue_bar,erste.x,NaN)`

Again, the function `ComparativeStaticPlots` starts `PlotsWithConfInt` and stops afterwards.

2. go to subdirectory `paras`
`load reduce_bUA_sBar_gesamt.mat`
go to subdirectory '3 analyse'
`GrossWage = wage./(1-taxrate);`
`wait = 0; WaitAtEnd = 1;`
`PlotsWithConfInt(WhatRed,GrossWage,welfare,U_rate,phi_0,wage,th,taxrate,...`
`V_b1_0,V_OneYear,V_w,firmvalue,mue_bar,wait,WaitAtEnd)`

This gives plots with confidence intervalls without normalization. This is useful e.g. for the unemployment rate, i.e. to see whether observed macro values are within the confidence band of micro estimates (see the discussion in the paper).

3. go to subdirectory `paras`
`load reduce_bUA_sBar_81a_81.mat`
go to subdirectory '3 analyse'
open the file `ComparativeStaticPlots.m` and set `NormalizeFlag` equal in around line 50 to, say, zero: `NormalizeFlag = 0;`
save the change and return to the Command Window of matlab. Then run
`DataSelector=3; trick = [1 1]; MicroFigures = 1;`
`ComparativeStaticPlots(WithGrowth,WhatRed,BootstrapSelector,redB,...`
`DataSelector,trick,PRSS_U,go_sbars,go_welfare.',go_U_rate.',go_phi_0.',...`
`go_wage.',go_th.',go_taxrate.',go_V_b1_0.',go_V_OneYear.',go_V_w.',...`
`go_firmvalue.',go_b2s,go_mue_bar.',go_x(:,maxLoop))`

This gives plots with confidence intervalls without normalization for the subset 81a. Be sure to change `NormalizeFlag` back to one: `NormalizeFlag = 1`. Don't be surprised that the net wage w and tightness θ seem normalized, they are not. A and γ is computed such that they equal the observed wage and tightness equal the predicted one in the numerical solution. (See the paper for more on this.)

2.2 Plots for micro dynamics

Micro figures resulting from `microdynamics()` can be created either automatically as described above or by opening a file having the structure `*_81a_81.mat`, i.e. the file must not contain 'gesamt' as the latter does not save the micro results. Here is an example:

1. go to subdirectory `paras`
`load reduce_bUA_sBar_81a_81.mat`
go to subdirectory '3 analyse'
`DataSelector=3; trick = [1 1]; MicroFigures = 1;`
`microdynamics(WithGrowth,WhatRed,BootstrapSelector,trick,MicroFigures,...`
`PRSS_x,PRSS_dmx,PRSS_eta,PRSS_phi,PRSS_mue,PRSS_mue0,PRSS_mueb1,PRSS_Vbs,PRSS_y0)`
`ComparativeStaticPlots(WithGrowth,WhatRed,BootstrapSelector,redB,...`
`DataSelector,trick,PRSS_U,go_sbars,go_welfare.',go_U_rate.',go_phi_0.',...`
`go_wage.',go_th.',go_taxrate.',go_V_b1_0.',go_V_OneYear.',go_V_w.',...`
`go_firmvalue.',go_b2s,go_mue_bar.',go_x(:,maxLoop))`

This gives plots for a subset (here 81a) of the entire data.

2.3 Plot with growth

The section analysing the effect of growth on unemployment reduction shows a figure on “the effect of economic growth ...”. This figure is created as follows:

```
go to subdirectory '3 analyse'
run UrateWithAndWithoutGrowth.m
```

Looking at the vectors `UrateWithGrowth` and `UrateWithoutGrowth` resulting from this programme run gives the unemployment rate reductions as discussed in the paper.

3 Numerical implementation

3.1 Steplength $h = 0.8$ is quick and precise

The numerical solution for the evaluation in the section “Evaluating the labour market reforms” had to make some choices concerning step length (h in the matlab code). In all solutions reported in the paper, the step length is set equal to 0.8 months. The entitlement period for UI payments is set to $\bar{s} = 12$ (instead of the 12.22 in the data) as we need a number for \bar{s} which can be divided by h such that an integer results. Such an integer is required e.g. when computing the integrals in the Volterra equations.

A consequence of $h = .8$ is that reductions in the entitlement period can also be only in steps of h (see e.g. `HartzStep_s` in `go.m`). In the data, we observe a reduction from 12.22 to 10.86 following the reform. In the numerical implementation we reduce entitlement period from 12 to 10.4, i.e. by slightly more than 10.86 as the difference between 12 and 10.4 is $2h = 1.6$.

The main reason for not reducing h to .5 or .2 is computational time. While a typical solution for 0.8 takes .7 hours, a solution with .5 takes 8 hours and one with .2 takes 24.7 hours. As we need 80 solutions in order to obtain our confidence bands as visible in the figures of the section “Evaluating the labour market reforms”, time becomes crucial.

As we want to be sure that our results are not too strongly driven by these numerical compromises, we ran solutions without confidence bands also for $h = .2$. This allows us to reduce the entitlement length from 12.2 to 10.8. To do so, we adjusted `HartzStep_s` in `go.m` to 1.4 (instead of 1.6). \bar{s} is adjusted automatically in `Inputdata.m` to 12.2. We started the programme by `hyb('bUA_sBar','81a',1,.2)`. The (normalized) unemployment rate reduced from 1 to 97.51% instead of 97.35%. All results are available in `reduce_bUA_sBar_81a_1_h=0.2.mat`.

We consider this so small that we continued working with $h = 0.8$.

3.2 Changes in β to .3 or .7 do not have strong effects

The results of changing β to .3 are saved in `reduce_bUA_sBar_81a_8_beta=0.3.mat`. The (normalized) unemployment rate reduced from 1 to 97.35% as well. This needs to be the case as block 1 (see the appendix “Steady state solution” to the paper) fixes effort without using β . When effort is fixed, the unemployment rate is fixed, as is visible from the Volterra equations in the paper. When looking at the effect on the wage, there was no difference concerning the level after the reform (1174 in both cases). Only after 3 additional steps (at ‘-3’ in the figure), there was a difference of 1182.0 instead of 1182.1.

Similarly negligible effects hold for $\beta = .7$.

3.3 Control flags

There are various control flags in the programme. Most of them can be found in the master file `hyb.m` but also in other subsequent files, e.g. in `zusammen.m`. The master file `hyb.m` contains, inter alia, the flag “genau”. When `genau = 1`, the results are computed as they should be. With `genau = 0`, the programme runs much faster but results are wrong. A message will appear showing that this is a programming checking mode. Another flag is “WithGrowth” which allows to switch on and off any growth effect of TFP.

The file `zusammen.m` merges datasets from two simulation runs. Some simulations do not result in numerically convincing solutions. This applies currently to 5 of 161 solutions, i.e. around 3.1%. When the flag “MinusThreeGoAway” is set equal to 1, these numerically non-convincing solutions (error code -3 in matlab’s `fsolve`) are removed. Reducing the step-length between different evaluations would allow us to find solutions. We currently work on this. Setting `MinusThreeGoAway=0` does not remove these 5 solutions with error code -3. The only implication is that the confidence band around “labour market tightness” θ in our fig. 3 “Aggregate effects of UA payments `bUA` and entitlement length \bar{s} ” is somewhat less smooth. As we are convinced that this is a purely numerical issue, we neglected this at this point.

There are many other flags in the files and an explanation can be provided upon request.