Dynare User Guide to LQ Approximations for RE Models in Dynare[*]

Paul Levine
University of Surrey

Joseph Pearlman
Loughborough University University

Bo Yang
University of Surrey

October 26, 2011

Introduction

Below is the addition to the document **Dynare.pdf** that describes how to use the new software for linear quadratic approximations to nonlinear optimal policy problems under rational expectations, with full or partial information.

4.22 LQ Approximations

Description

The problem is to maximize $E_0 \sum \beta^t u(Y_t, W_t)$ such that

$$E_t g(Y_t, Y_{t+1}, W_t, \varepsilon_t) = 0 \qquad h(Y_t, Y_{t-1}, W_t, \varepsilon_t) = 0 \tag{1}$$

We write the problem in this way so that, for convenience, there are no 2nd order derivatives in Y_{t+1} and Y_{t-1}; thus the main constraint is that there are no nonlinear interactions between Y_{t+1} and Y_{t-1}. If there are, then just define a new set of lagged variables $Ylag_{it} = Y_{i,t-1}$, and append the latter equations to $h(\ ,\ )$ and the new variables to Y_t.

This particular part of the software does not yet recognise lags in instruments. In order to use a lagged instrument in the simple rule, one needs to define a variable that is set equal to the instrument, and then use the lag of that variable.

The software first calculates the steady state of the optimum of the deterministic problem, then linearizes the constraints, and finds the quadratic approximation to utility about this steady state. The latter is obtained by calculating the second derivative (Hessian)

[*]We acknowledge financial support for this research from the EU Framework Programme 7 project MONFISPOL.

matrix of the Lagrangian. All LQ approximations are obtained for proportional deviations about the steady state (except when the steady state is zero, in which case deviations are used). It is recommended that variables like inflation and interest rates are expressed in gross terms in the nonlinear setup, not as deviations about the steady state of 1.

Next the software computes policy for the deviations model for three scenarios: fully optimal, time consistent and optimized simple rules. This can be done for both full and partial information. There are two further extensions: (a) robust simple rules can also be calculated by estimating the model using data, and then sampling the the draws; (b) zero lower bound (ZLB) constraints for the interest rate can be incorporated into policy design by subtracting a term $\frac{1}{2}w_r(R-\hat{R})^2$ from the utility function. The software iterates until it finds values of w_r and $\hat{R}$ that optimally satisfy the probability bound on failing the ZLB constraint.

Currently the software is not completely integrated into Dynare, so that the directory, *PI_ACES*, that contains the LQ approximation software must be placed at the top of the path when using Matlab's *setpath* directive.

The changes to the *modfile* that are required are dealt with in the order they appear. Some changes are under modification to streamline them better.

The parameters for use in ZLB calculations, w_r, and $\hat{R}$, must appear, if used, in that order in a separate list of parameters.

The policy rule that appears in the *model* equations must be preceded by a tag:

[tag1 = 'aceslq_sim_rule']

Options

options_.useACES=1;

This triggers the use of the interface with the Fortran package ACES for calculation of the rules.

options_.usePartInfo=1

This triggers the partial information software, and must be used in conjunction with the *varobs* command that lists the variables that agents observe.

global lq_instruments;

Indicates that there are exogenous instruments

lq_instruments.lambda_w=... ;

Defines the weight for the (under-)relaxation method (ideally ¡0.5). The latter is used for iterating to the optimal ZLB parameters

lq_instruments.maxiter=...;

maximum number of relaxation iterations.

lq_instruments.seedvar=strvcat('*varname*');

The name of the variable for which a seed value is used for calculating the steady state. Gross inflation is a good choice.

lq_instruments.optvar=strvcat('*varname*', '*varname*',...);

This must include variables in the policy rule, and any other variables of interest to the user. If all variables are included, then almost certainly none of the rules will solve.

lq_instruments.sim_lb=[, ,...];

Lower bounds for feedback coefficients according to the order listed in optvar

lq_instruments.sim_ub=[, ,...];

Upper bounds for feedback coefficients according to the order listed in optvar

lq_instruments.mhdraws=... ;

Number of random draws taken from MH iterations (outputs generated from estimation)

lq_instruments.aces=[];

performs all control exercises

lq_instruments.aces=strvcat('OPT');

performs optimal rule

lq_instruments.aces=strvcat('TCT');

performs time-consistent rule

lq_instruments.aces=strvcat('SIM');

performs simple rule with the given parameters

lq_instruments.aces=strvcat('SIM2');

performs optimized simple rule

lq_instruments.zlb_wr=...;

weight on squared instrument term

lq_instruments.zlb_prob=...;

permitted probability of hitting ZLB

lq_instruments.zlb_iter=...;

intermediate iterations for updating zlb_wr

lq_instruments.zlb_step=...;

step size for updating zlb_wr

lq_instruments.zlb_algo=1;

to be deleted in later versions

lq_instruments.zlb_outeriter...;

maximum number of outer iterations when updating weight and rhat - actual number is reported at the end of run

planner_objective

utility function - contains '-weight/2*(R-rhat)**2' term when ZLB is used

ramsey_policy(planner_discount=..., instruments=(*varname*), irf=30);

discount value, name of instrument, number of periods (currently this is required to be 30) for impulse response function.