Terminal condition on the variations of the jumping variables instead of their levels in deterministic models

Stéphane Adjemian*

Université du Maine and GAINS

August, 2013

This note describes a new feature in Dynare that can be used to simulate perfect foresight models when the steady state is unknown or when the model is very persistent. In the sequel we first present the new option and its implementation, and we propose an application to a small Real Business Cycle (RBC) model with a very persistent shock.

We propose a new option in the dynare syntax that replaces all (by default) the forward looking variables, say x_{t+1}, by the sum of the current level and expected variation:

$$x_{t+1} = x_t + \Delta x_{t+1}$$

In the long run, if the endogenous variable x reaches the constant state $x^\star$ then, by definition, the variation will be zero. To solve a perfect foresight model, we only need to impose an initial condition for the state variables and a terminal condition for the jump (leaded) variables. Using this transformation allows us to pin down the equilibrium path to the steady state without knowing the steady state level $x^\star$, because the steady state of the leaded variables, Δx_{t+1} after transformation, which we need to impose is trivially zero[1]. Note however that in practice we generally consider an initial condition in deviation to the steady state, to obtain a more readable equilibrium path, so the steady state is generally still needed to run the simulation.

This transformation is triggered by the option `differentiate_forward_vars`. Behind the scene, the preprocessor automatically creates one auxiliary variable for each endogenous variable appearing with a lead in the model. More precisely, if the model contains `x(1)`, then a variable `AUX_DIFF_VAR` will be created such that `AUX_DIFF_VAR=x-x(-1)`, and any occurence of `x(1)` will be replaced by `x+AUX_DIFF_VAR(1)`. By default this transformation is applied to all the endogenous variables appearing at time $t+1$. The transformation can be applied to a subset of these variables, if a list of endogenous variables, say `X`, `Y` and `Z`, is declared after the option:

*`stepan@dynare.org`

[1]We implicitly assume here that there is no growth in the long run, but this assumption could be relaxed (only the existence of a balanced growth path is needed).

```
model(differentiate_forward_vars = ( X Y Z ));

    blablabla

end;
```

Because we change the terminal condition by reparameterizing the leaded variables, all the algorithms available to solve a perfect foresight model can be used with this new option.

As an application, we consider an RBC model with a CES technology and very persistent productivity shock. The code is available upon request. We set the autoregressive parameter of this exogenous productivity to 0.999, so that in period 400 the level of productivity, after an initial one percent shock, is still 0,67% above its steady state level (1). We consider three scenarii:

1. Solve the perfect foresight model imposing a return to the steady state in period 8000 using the standard terminal condition.

2. Solve the perfect foresight model imposing a return to the steady state in period 400 using the standard terminal condition.

3. Solve the perfect foresight model imposing a return to the steady state in period 400 and using the alternative terminal condition (zero growth).

The first scenario will serve as a benchmark because it is more reasonable to impose a return to the steady state in period 8000 (productivity level is then 0.0003% above its steady state level, which is under the arbitrarily chosen level of precision for the nonlinear solver, 1e-5 by default). Figure (1) shows the equilibrium paths of output under scenarii 1 and 2. The most striking feature is that the difference between the two paths becomes relatively important near the end of the simulation (after period 300). Figure (2) shows the equilibrium paths under scenarii 1 and 3. Again we observe, though much smaller, an increasing difference between the two paths at the end of the simulation period. Finally, figure (3) compares the solutions under scenarii 2 and 3. Not surprisingly, by imposing a terminal condition on the variation instead of the level, we produce a much smoother equilibrium path for the last periods. In this case, the solution under scenarii 3 is much closer to the true equilibrium path (*i.e.* the one obtained under scenario 1). For models charaterized by higher nonlinearities, the "jump" observed during the last periods, with the terminal condition on the level, can result in a failure of the nonlinear solver. In these situations, imposing a terminal condition on the variation instead of the level can help. However, if the the nonlinear solver does not fail, as it happens here, the difference between the equilibrium paths is very small for the first periods. In our application, the maximum difference between scenarii 1 and 2 for periods 1 to 100 is 2.1×10^{-7}, while the maximum difference between scenarii 1 and 3 is 5.4×10^{-9}. These numbers are much smaller than the tolerance parameter of the nonlinear solver (1e-5), in this sense these numbers are not significantly different. The difference between the equilibrium paths becomes greater than the tolerance parameter only after period 204. If we are only interested in the first periods, the definition of the terminal condition makes no

difference. If we reduce the number of periods by setting `periods=50` (instead of 400 as in the previous case) we obtain a diverging path with the standard terminal condition and the fit is much improved with the terminal condition on the variation. Actually, for the first period we hardly see any difference with the "true" path. We may conclude that the alternative terminal condition allows to reduce the number of periods (hence the computing time) without sacrifying the accuracy of the solution for the first period (this is crucial, for instance, in the case of the extended path algorithm). The problem is that if we reduce further the horizon of the perfect foresight model, by setting `periods=5` (!), we get a different ordering. In both cases the nonlinear solver converges, in both cases the accuracy errors are important, but this error is minimized when the terminal condition on the level is considered. So there is no clear cut conclusion about the advantage of the alternative terminal condition with respect to the computing time.

Figure 1: **Equilibrium paths for output.** Red and black curves are respectively the solutions under scenarii 1 and 2.

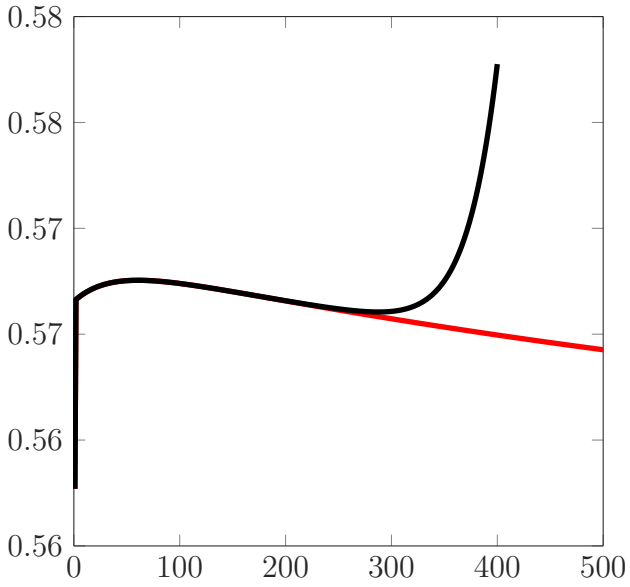

Figure 2: **Equilibrium paths for output.** Red and green curves are respectively the solutions under scenarii 1 and 3.

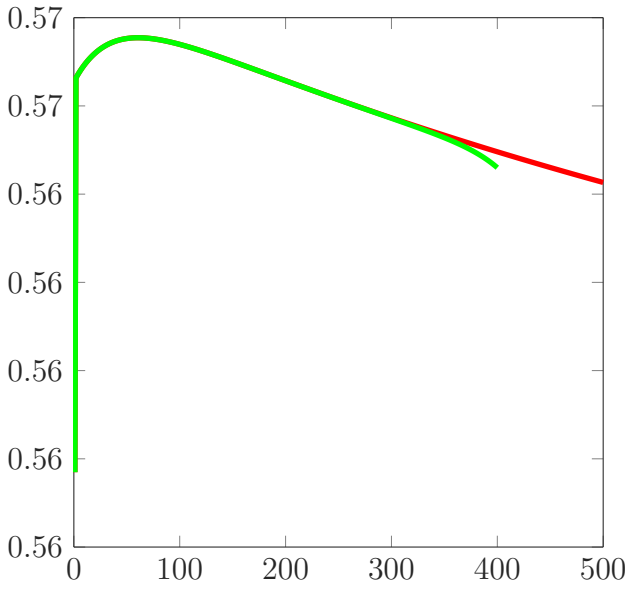

Figure 3: **Equilibrium paths for output.** Black and green curves are respectively the solutions under scenarii 2 and 3.