Solving Economics PDE Models

Matthieu Gomez *

September 3, 2016

This note details how Ψtc works

1 Solving Finite Difference Schemes

1.1 How Ψtc works

Denote Y the solution (encoded as an array) and denote $F(Y) = 0$ the finite difference scheme corresponding to a model. On can solve for Y using one of the two methods:

1. Non linear solver for $F(Y) = 0$. Updates take the form

$$0 = F(y_t) + J_F(y_t)(y_{t+1} - y_t)$$

   The method converges only if the initial guess is sufficiently close to the solution.

2. ODE solver for $F(Y) = \dot{Y}$. The solution of $F(Y) = 0$ is obtained with $T \to +\infty$ With a simple explicit Euler method, updates take the form

$$0 = F(y_t) - \frac{1}{\Delta}(y_{t+1} - y_t)$$

   Convergence conditions are given by the Barles-Souganadis theorem. Explicit schemes usually don't satisfy them.

I propose to use a fully implicit Euler method, which has better convergence properties than explicit schemes. Updates take the form

$$\forall t \leq T \qquad 0 = F(y_{t+1}) - \frac{1}{\Delta}(y_{t+1} - y_t)$$

Each time step is a non linear equation, which I solve using a Newton-Raphson method. These inner iterations take the form

$$\forall i \leq I \qquad 0 = F(y_t^i) - \frac{1}{\Delta}(y_t^i - y_t) + (J_F(y_t^i) - \frac{1}{\Delta})(y_t^{i+1} - y_t^i)$$

*I thank Valentin Haddad and Ben Moll for useful discussions.

As pointed above, the Newton-Raphson method converges when y_t is sufficiently close to y_{t+1}. Therefore I decrease Δ until the inner Newton-Raphson method converges.[1].
I accomodate algebraic equations by setting $\Delta = 0$ for these equations. This ensures that the PDEs are solved backward on a path that always satisfies the algebraic constraints.

How does the method relate to the two algorithms seen above? When $I = 1$ (i.e. with only one inner iteration) the update can be seen as the sum of a Newton-Raphson and an explicit time step

$$\forall t \leq T \qquad 0 = F(y_{t+1})(J_F(y_t) - \frac{1}{\Delta})(y_{t+1} - y_t)$$

The method can be also be seen as a dampened Newton-Raphson algorithm. As in the Levenberg-Marquardt method, the diagonal of the Jacobian is modified until the algorithm gets close to the solution.

At the start of the algorithm, the jacobian is computed using analytical differenciation. It is then updated with Broyden updates. The algorithm usally converges quickly: the convergence is quadratic around the solution, since the algorithm follows the Newton-Rapshon method around the solution.

1.2 Related Methods

- With $I = 1$ and constant Δ, we obtain the method in Achdou, Han, Lasry, Lions (2016) for a partial equilibrium consumption / saving problem with separable preference. I've found that allowing Δ to change and a $I > 1$ are important for the robustness of the algorithm. Moreover, Ψtc handles systems with implicit jacobians and algebraic equations.

- A similar algorithm is used in Fluid Dynamics. In this context, it is called Pseudo-Transient Continuation (denoted Ψtc).

2 Writing Finite Difference Schemes

- Write the finite difference scheme so that the implicit Euler method satisfies the convergence conditions of Barles-Souganadis theorem (as much as possible). In particular,
  - Upwind first derivatives to make the scheme monotonous (for instance see Achdou, Han, Lasry, Lions (2016))
  - Write the function F so that $\dot{Y}$ would appear as such (i.e. not multiplied by some parameters). For instance, a typical PDE for the price

[1] However, I cannot prove the convergence of the overall scheme when Δ depends on the step. The Barles-Souganadis theorem ensures that the implicit Euler scheme converges only with Δ fixed.

dividend ratio should be written

$$0 = p(\frac{1}{p} + E[\frac{dD}{D}] + E[\frac{dp}{p}] + \sigma[\frac{dp}{dp}]\sigma[\frac{dD}{dD}] - r - \kappa(\sigma[\frac{dp}{dp}] + \sigma[\frac{dD}{D}]))$$

- When solving for multiple functions, use the same economic quantities across the different equations. This ensures that the time step is comparable across different equations. For instance use the wealth / consumption of each agent in heterogeneous agent models.