TP : Finite difference method for HJB equations
M2 Modélisation aléatoire - Université Denis-Diderot
EDP en Finance et Méthodes Numériques

The aim is to test different schemes for HJB equations. A first part concerns first order HJB equations, related to deterministic control, where it is more easy to study the numerical schemes, the stability and monotonicity of the schemes. A second part concerns the approximamtion of a second order HJB equation (uncertain volatility model).

1 An eikonal equation

We look for a numerical approximation of $v = v(t,x)$ solution of the following equation, for $t \in (0,T)$, $x \in \Omega := (S_{min}, S_{max})$:

$$\begin{cases} v_t(t,x) + c|v_x(t,x)| = 0, \quad t \in (0,T),\ x \in (S_{min}, S_{max}) \\ v(t, S_{min}) = v_\ell(t) \quad t \in (0,T) \\ v(t, S_{max}) = v_r(t) \quad t \in (0,T) \\ v(0,x) = v_0(x) \quad x \in (S_{min}, S_{max}). \end{cases} \tag{1}$$

This PDE is called an "eikonal equation". It is a particular case of Hamilton-Jacobi-Bellman (HJB) equations.

We will consider the following parameters:

$$c = 1,\ T = 1,$$

and the following initial data

$$v_0(x) = -(\max(1-x^2, 0))^2 \tag{2}$$

and the following boundary

$$(S_{min}, S_{max}) := (-3, 3)$$

with zero Dirichlet boundary conditions: $v_r(t) = v_\ell(t) = 0$ (other boundary conditions could be used for different data).

In particular, we aim at computing $v(t,x)$ at final time $t = T$.

We introduce a discrete mesh as usual: $h := \frac{S_{max}-S_{min}}{I+1}$, $\Delta t := \frac{T}{N}$ and

$$x_j := S_{min} + jh, \quad j = 0, \dots, I+1 \text{ (mesh points)}$$
$$t_n = n\Delta t, \quad n = 0, \dots, N \text{ (time mesh)}$$

We are looking for U_j^n, an approximation of $v(t_n, x_j)$.

1.1 EE scheme

We remark that $c|v_x| = max(cv_x, -cv_x)$. We assume $c \geq 0$. Hence A first possible "Euler Forward scheme" (or Explicit Euler scheme), using the upwind and downwind approximations, for stability reasons, is as follows:

$$\begin{aligned} &\frac{U_j^{n+1} - U_j^n}{\Delta t} + \max\left(c\frac{U_j^n - U_{j-1}^n}{h},\ -c\frac{U_{j+1}^n - U_j^n}{h}\right) = 0 \\ &\qquad\qquad\qquad\qquad\qquad n = 0, \ldots, N-1, \\ &\qquad\qquad\qquad\qquad\qquad j = 1, \ldots, I \\ &U_0^n = v_\ell(t) \equiv 0, \quad U_{I+1}^n = v_r(t) \equiv 0 \quad n = 0, \ldots, N \\ &U_j^0 = v_0(x_j) \quad j = 1, \ldots, I \end{aligned} \tag{3}$$

1/ Compute the consistency error of the scheme.[1]

2/ Program the scheme. We advise to program matrix D^- and vector function $q^-(t)$ such that

$$(D^- U^n + q^-(t))_{1\leq i\leq I} = (c\frac{U_i^n - U_{i-1}^n}{h})_{1\leq i\leq I}.$$

and $D^+, q^+(t)$ in the same way:

$$(D^+ U^n + q^+(t))_{1\leq i\leq I} = (c\frac{U_{i+1}^n - U_i^n}{h})_{1\leq i\leq I}.$$

One can furthermore use the command `np.maximum(v1,v2)` (resp. `np.minimum(v1,v2)`) to maximize (resp. minimize) componentwise two vectors of type `numpy.array`

Remarque. 1 *In order to improve the effiency of the code, it is possible to precompute D^- and D^+ as sparse matrices.*[2]

---

[1] In order to obtain the consistency error, denote $V_j^n = \varphi(t_n, x_j)$ where φ is a sufficiently regular function, write the scheme in abstract form as follows: $\mathcal{S}(t_{n+1}, x_j, U_j^{n+1}, [U]) = 0$, (where $[U] = (U_\ell^k)$). Let $\mathcal{H}$ corresponds to the PDE equation:

$$\mathcal{H}(\varphi)(t,x) := \varphi_t(t,x) + c|\varphi_x(t,x)|$$

so that the PDE for φ reads $\mathcal{H}(\varphi)(t,x) = 0$. Then a consistency error can be defined as ϵ_j^n such that

$$\mathcal{S}(t_{n+1}, x_j, V_j^{n+1}) = \mathcal{H}(\varphi)(t_{n+1}, x_j) + \epsilon_j^n.$$

Then show that the consistency error satisfies

$$|\epsilon_j^n| \leq C(\Delta t + h).$$

This means that the consistency error is of order one in time and of order one in space.

[2] Once a numpy array/matrix `D` has been defined, it possible to use simply `D=sparse(D)` with the sparse module loaded as described in TP1, Exercise 1.

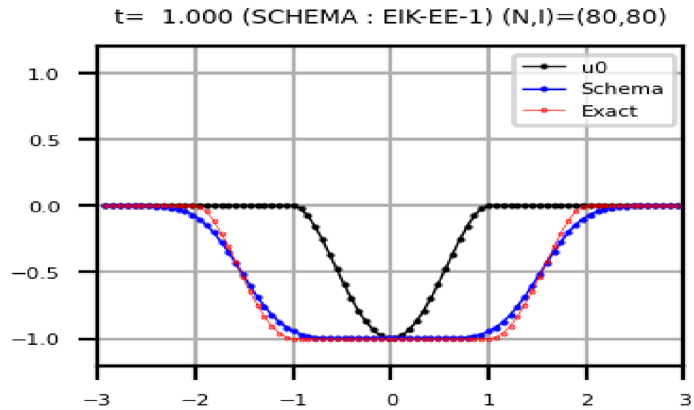

Figure 1: Eikonal equation, with $N = I = 80$.

3/ Intuitively guess the exact solution of the problem (the formula may change depending on the initial condition). [3]

4/ Show that the scheme is stable and monotone[4] for an appropriate (sufficient) CFL condition on the mesh steps to be found. Test in particular with $(I, N) = (80, 80)$, and $(I, N) = (80, 8)$. Are the results coherent with the CFL condition ?

5/ (Program) Estimate the error at the point $S_{val} = 1.5$ and $T = 1$, and the order of convergence.Draw an error table (for instance in the same way as in TP1). [5] It is possible to use for instance $N = I$ in a list of the form 10×2^k, $k = 0, 1, \ldots$. (Other couples (N, I) can be used in order to test the approximation in time or in space).

6/ (Program) Test also the scheme on the following initial data

$$v_0(x) = (\max(1 - x^2, 0))^2 \quad \text{and with } T = 0.4. \tag{4}$$

Typical results are shown in Figure 1.

1.2 Improving the order of consistency

1/ (Program) Show that the basic attempt to improve the accuracy with the following scheme

[3] Note that for the equation $v_t + cv_x = 0$ (with $v(0, x) = v_0(x)$, $x \in \mathbb{R}$), the exact solution would be $v(t, x) = v_0(x - ct)$. In the same way, for the equation $v_t - cv_x = 0$ (with $v(0, x) = v_0(x)$, $x \in \mathbb{R}$), the exact solution would be $v(t, x) = v_0(x + ct)$. For the eikonal equation, with $c \geq 0$, the general formula is $v(t, x) = \min_{y \in [x-ct, x+ct]} v_0(y)$. If $v_0 \downarrow$ on $(-\infty, 0]$ and $\uparrow$ on $[0, \infty)$, then $v(t, x) = v_0(x + ct)$ for $x \leq -ct$, $v_0(0)$ for $x \in [-ct, ct]$ and $v_0(x - ct)$ for $x \geq ct$.

[4] An explicit scheme of the form $U_j^{n+1} = F(U_{j-1}^n, U_j^n, U_{j+1}^n)$ is called "monotone" if F is an increasing function of all its arguments, i.e. $a \to F(a, b, c) \uparrow$, $b \to F(a, b, c) \uparrow$, and $c \to F(a, b, c) \uparrow$.

[5] For this point $S_{val} = 1.5$, $T = 1$, and for the present problem, it can be shown that the exact value at time T is also given by $v_0(x - ct)$ with $x = S_{val}$ and $t = T$.

$$\frac{U_j^{n+1}-U_j^n}{\Delta t} + c\left|\frac{U_{j+1}^n - U_{j-1}^n}{2h}\right| = 0 \tag{5}$$

$$n = 0,\dots,N-1,$$
$$j = 1,\dots,I$$
$$U_0^n = v_\ell(t_n) \equiv 0, \quad U_{I+1}^n = v_r(t_n) \equiv 0 \quad n = 0,\dots,N$$
$$U_j^0 = v_0(x_j) \quad j = 1,\dots,I$$

is not working (take for instance $N = I = 400$, $N = I = 800$ ...).

2/ Look for a, b, c such that

$$\phi_x(x_j) = \frac{a\phi(x_j) + b\phi(x_{j-1}) + c\phi(x_{j-2})}{h} + O(h^2)$$

(where $x_i = S_{min} + ih$).[6]

3/ (Program) Let

$$\tilde{D}^- U_j^n = \frac{3U_j^n - 4U_{j-1}^n + U_{j-2}^n}{2h}, \quad \text{and} \quad \tilde{D}^+ U_j^n = -\frac{3U_j^n - 4U_{j+1}^n + U_{j+2}^n}{2h}$$

Program the following modified scheme:

$$\frac{U_j^{n+1}-U_j^n}{\Delta t} + \max\left(c\tilde{D}^- U_j^n,\ -c\tilde{D}^+ U_j^n\right) = 0 \tag{6}$$

$$n = 0,\dots,N-1,$$
$$j = 1,\dots,I$$
$$U_0^n = v_\ell(t_n) \equiv 0, \quad U_{I+1}^n = v_r(t_n) \equiv 0 \quad n = 0,\dots,N$$
$$U_j^0 = v_0(x_j) \quad j = 1,\dots,I$$

4/ (Program) Is the error improved with this scheme ? Draw error tables for the scheme. Show that the scheme is not monotone. Observe that the scheme still converges numerically towards the correct solution.

5/ (Program) RK2 variant. Let us denote $U^{n+1} = S^1(U^n)$ the scheme (6). Consider now the following scheme (RK2):

$$U^{n+1} = \frac{1}{2}(U^n + S^1(S^1(U^n))).$$

Show that the scheme corresponds to an RK2 scheme.[7] Observe that the scheme is numerically of order $(2,2)$, and is subject to some CFL condition for stability.

In conclusion: non-monotone schemes may be used, although convergence proof may not be established, or may be more demanding !

[6]Show that $a = \frac{3}{2}$, $b = \frac{-4}{2}$, $c = \frac{1}{2}$ is the only solution.

[7]For the approximation of $u_t = L(u)$, the RK2 scheme is $u^{n,1} = u^n n + \Delta t L(t_n, u^n)$ and $u^{n+1} = u^n + \frac{\Delta t}{2}(L(t_n, u^n) + L(t_{n+1}, u^{n,1}))$. It is consistent of order 2 in time.

1.3 IE scheme

The following implicit schemes are not to be programmed in this session. They are presented as examples of implicit non-linear schemes. They will be studied in the next programming session.

In order to get rid of the CFL condition, a natural implicit scheme is the following:

$$\frac{U_j^{n+1}-U_j^n}{\Delta t} + c\max\left(\frac{U_j^{n+1}-U_{j-1}^{n+1}}{h},\ -\frac{U_{j+1}^{n+1}-U_j^{n+1}}{h}\right) = 0 \qquad (7)$$

$$n = 0,\ldots,N-1,$$
$$j = 1,\ldots,I$$

$$U_0^n = v_\ell,\ U_{I+1}^n = v_r \quad n = 0,\ldots,N \qquad (8)$$

$$U_j^0 = v_0(x_j) \quad j = 1,\ldots,I \qquad (9)$$

- What is the consistency error of the scheme ?

- In order to program the scheme, use an equivalent matrix formulation of the scheme in the following form:

Let $A^- := \frac{1}{h} tridiag(-1,1,0)$ and $A^+ := \frac{1}{h} tridiag(0,1,-1)$ (where $T = tridiag(a_i,b_i,c_i)$ denotes the tridiagonal matrix with $T_{i,i-1} = a_i$, $T_{i,i} = b_i$ and $T_{i,i+1} = c_i$). The scheme can be written in vector form as follows:

$$\max_{\pm}\left(\frac{U^{n+1}-U^n}{\Delta t} + A^{\pm}U^{n+1} + q^{\pm}(t_{n+1})\right) = 0, \quad \text{in } \mathbb{R}^I$$

where $q^{\pm}$ are vectors taking into account the boundary conditions (here for the chosen boundary conditions we have $q^{\pm} \equiv 0$). Denoting

$$B^{\pm} := I + \Delta t A^{\pm}, \quad b^{\pm} := U^n - \Delta t q^{\pm},$$

notice that the scheme is also equivalent to find $x = U^{n+1} \in \mathbb{R}^I$ solution of

$$\max_{\pm}\left(B^{\pm}x - b^{\pm}\right) = 0, \quad \text{in } \mathbb{R}^I$$

1/ Implement[8] a ("semi-smooth") Newton's method for finding $x \in \mathbb{R}^I$ solution of

$$F(x) := \max(Bx - b, Cx - c) \equiv 0 \quad \text{for } x \text{ in } \mathbb{R}^I,$$

for given vectors $b, c \in \mathbb{R}^I$ and matrices $B, C \in \mathbb{R}^{I\times I}$. It will be shown that Newton's method converges always if for instance B and C are diagonally dominant M-matrices.

2/ Program the EI scheme

3/ Check the stability of the scheme inconditionnally with respect to the mesh steps

[8] Typical scheme : start from a given x^0. Compute $x^{n+1} = x^n - F'(x^n)^{-1}F(x^n)$ until convergence. In this iteration, $F(x^n) = max(Bx^n - b, Cx^n - c)$ and $F'(x^n)$ is a square matrix that can be defined as follows:

$$\begin{aligned} F'(x^n)_{ij} &:= B_{ij} \quad \text{if } (Bx^n-b)_i \geq (Cx^n-c)_i, \\ &:= C_{ij} \quad \text{otherwise} \end{aligned}$$

1.4 Implicit second order variant

Program the following scheme. Observe a numerical order $(2, 2)$ inconditionnally on the mesh steps. For instance, one can compute the error tables with $N = I$ and $N = I/10$.

$$\frac{3U_j^{n+1} - 4U_j^n + U_j^{n-1}}{2\Delta t} + \max\left(c\tilde{D}^- U_j^{n+1},\ -c\tilde{D}^+ U_j^{n+1}\right) = 0, \qquad (10)$$

$$n = 1, \ldots, N-1,$$
$$j = 2, \ldots, I$$

$$U_0^n = v_\ell(t_n) \equiv 0, \quad U_{I+1}^n = v_r(t_n) \equiv 0 \quad n = 0, \ldots, N$$
$$U_j^0 = v_0(x_j) \quad j = 1, \ldots, I$$

Note: in order to compute the first step U^1, an implicit IE step can be used.

2 A simple uncertain volatility model

We consider an academic uncertain volatility model where the volatility of some asset can be controlled and can take any value $\sigma \in [0, 1]$ (i.e. $dS_\tau = \sigma_\tau dW_\tau$ with control $\sigma_\tau \in [0, 1]$). The maximisation of the terminal price (with payoff $v_0(.)$) under such controlled asset leads to an HJB equation of the form

$$v_t(t, x) + \min_{\sigma \in [0,1]} (-\frac{1}{2}\sigma^2 v_{xx}(t, x)) = 0, \quad t \in (0, T),\ x \in \mathbb{R}$$
$$v(T, x) = v_0(x)$$

After a time reversal, and imposing boundary conditions, we are led to the following second order HJB equation:

$$-v_t(t, x) + \min(0, -\frac{1}{2}v_{xx}(t, x)) = 0, \quad t \in (0, T),\ x \in (S_{min}, S_{max}) \qquad (11a)$$
$$v(0, x) = v_0(x) \quad x \in (S_{min}, S_{max}) \qquad (11b)$$

with the following boundary conditions:

$$v(t, S_{min}) = v_\ell \quad t \in (0, T) \qquad (12a)$$
$$v(t, S_{max}) = v_r \quad t \in (0, T) \qquad (12b)$$

We will consider the following parameters:

$$S_{min} = -3,\ S_{max} = 3, \quad \text{and} \quad T = 0.5,$$

with smooth initial data

$$v_0(x) := \frac{1}{2} sign(x)((max(1 - |x|, 0))^4 - 1);$$

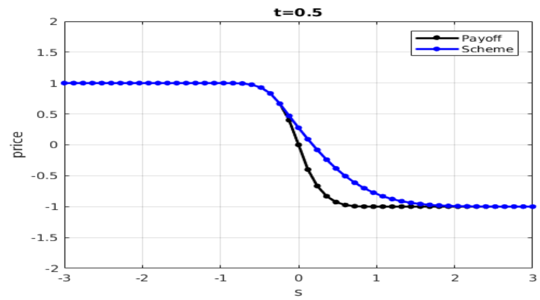

Figure 2: uncertain volatility test $T = 0.5$, $N = I = 50$.

and boundary conditions compatible the initial data: $v_\ell = 1$ and $v_r = -1$.

We introduce the second order approximation of $-u_{xx}$:

$$D^2 U_i := \frac{U_{i-1} - 2U_i + U_{i+1}}{h^2}.$$

1) Euler Explicit scheme.

A possible EE scheme for (11)-(12) is therefore:

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} + \min\left(0,\ -\frac{1}{2}\sigma^2 D^2 U_i^n\right) = 0 \tag{13}$$

$$n = 0, \ldots, N-1,$$
$$j = 1, \ldots, I$$

$$U_0^n = v_\ell,\ U_{I+1}^n = v_r \quad n = 0, \ldots, N$$
$$U_j^0 = v_0(x_j) \quad j = 1, \ldots, I$$

- What is the consistency error of the scheme ?
- Program and test the scheme

2) Euler Implicit scheme

- Propose a corresponding implicit Euler scheme (EI)
- Program the EI scheme and check its unconditional stability with respect to the mesh parameters.

3) Second order scheme Developpement : program the uncertain volatility Example 4.2, Section 4 of [1] (implement a second order scheme, such as BDF2, and compare with a first order scheme).

References

[1] O. Bokanowski, A. Picarelli, and C. Reisinger. High-order filtered schemes for time-dependent second order HJB equations. *ESAIM Math. Model. Numer. Anal.*, 52(1):69–97, 2018.