High-order Approximation of the Finite Horizon Control Problem via a Tree Structure Algorithm [⋆]

A. Alla [*] M. Falcone [**] L. Saluzzi [***]

[*] *PUC-Rio, Rua Marques de Sao Vicente, 225, Gávea - Rio de Janeiro, RJ - Brasil - 22451-900, (e-mail: alla@mat.puc-rio.br).*
[**] *Sapienza Universitá di Roma - Piazzale Aldo Moro 5, 00185 Roma, (e-mail: falcone@mat.uniroma1.it)*
[***] *Gran Sasso Science Institute - Viale F. Crispi 7, 67100 L'Aquila, (e-mail: luca.saluzzi@gssi.it)*

Abstract: Solving optimal control problems via Dynamic Programming is a difficult task that suffers for the "curse of dimensionality" and this limitation has reduced its practical impact in real world applications since the construction of numerical methods for nonlinear PDEs in very high dimension is practically unfeasible. Recently we proposed a new numerical method to compute the value function eliminating the construction of a space grid and the need for interpolation techniques. The method is based on a tree structure that mimics the continuous dynamics and allows to solve optimal control problems in high-dimension. This is particularly useful to attack control problems with PDE constraints. We present a high-order approximation scheme based on a tree structure and show some numerical results.

Keywords: dynamic programming, optimal control, Hamilton-Jacobi-Bellman equation, tree structure algorithm, high-order approximation, error estimates

1. INTRODUCTION

The Dynamic Programming (DP) approach has been applied to several deterministic and stochastic optimal control problems. This approach has been revitalized by the development of a theory of weak solutions for Hamilton-Jacobi equations, the so-called viscosity solutions, introduced by Crandall and Lions in the middle of the 80s (see the monograph Bardi et al. (1997) and the list of references therein). The theory related to this approach is now rather complete and established giving a complete characterization of the value function as the unique viscosity solution of a nonlinear partial differential equation (the Hamilton-Jacobi-Bellman (HJB) equation). Tipically this equation has to be solved on a space grid and this is the bottleneck for numerical methods in high-dimension. Several efforts have been made to mitigate the limitations due to the *curse of dimensionality*, an obstacle that is particularly relevant in the framework of optimal control problems with PDE constraints. Let us just mention that an interesting approach is based on model order reduction techniques (e.g. Proper Orthogonal Decomposition) which allow to obtain a low dimensional version of the dynamics by orthogonal projection. Once a low dimensional approximation of the dynamics (e.g. $d \approx 5$) has been obtained the problem can be solved via the standard DP approach. We refer to the pioneering work Kunisch et al. (2004) for the coupling between model reduction and HJB equations and to Alla et al. (2017) for some a-priori error estimates related to the POD-HJB method. Note that the above papers refer to a *discretization in space and time* of the HJB equation. More recently we proposed a new method based on a time discretization of the dynamics which allows to mimic the continuous dynamics in high-dimension via a tree structure Alla et al. (2018). We refer to Capuzzo Dolcetta et al. (2001) for previous a-priori error estimates based on a time discretization of an infinite horizon control problem; these results were coupled to the space discretization in Falcone (1987). Later high–order error estimates have been obtained in Falcone et al. (1994) always for the infinite horizon problem. Here we deal with a finite horizon control problem and in the discretization the tree structure replaces the space grid allowing to increase the dimension of the state space. Furthermore, a pruning technique has been implemented to reduce the number of branches in the tree obtaining rather accurate results and a-priori error estimates Alla et al. (2018b). Here we improve the method showing how it can be extended to high-order approximation schemes always using only a time discretization. In the last section we present two tests. The first test shows that by applying the new tree structure algorithm (TSA) based on the Heun method for the approximation of the dynamics we obtain a second order approximation of the value function as expected (in this test we know the exact value function). The second test shows how the method can be applied to the control of a system governed by the advection equation.

[⋆] The second and third authors are currently members of the national group GNCS-INDAM and were partially supported for this research.

2. DYNAMIC PROGRAMMING ON A TREE STRUCTURE

In this section we will recall the *finite horizon control problem* and its approximation by the tree structure algorithm (TSA) (see Alla et al. (2018) for a complete description of the method). Let the system be driven by

$$\begin{cases} \dot{y}(s) = f(y(s), u(s), s), & s \in (t, T], \\ y(t) = x \in \mathbb{R}^d. \end{cases} \tag{1}$$

We will denote by $y : [t, T] \to \mathbb{R}^d$ the solution, by u the control $u : [t, T] \to \mathbb{R}^m$, by $f : \mathbb{R}^d \times \mathbb{R}^m \times [t, T] \to \mathbb{R}^d$ the dynamics and by

$$\mathcal{U} = \{u : [t, T] \to U, \text{measurable}\}$$

the set of admissible controls where $U \subset \mathbb{R}^m$ is a compact set. We assume that there exists a unique solution for (1) for each $u \in \mathcal{U}$.

The cost functional for the finite horizon optimal control problem will be given by

$$J_{x,t}(y, u) := \int_t^T L(y(s, u), u(s), s)e^{-\lambda(s-t)}\, ds + +g(y(T))e^{-\lambda(T-t)}, \tag{2}$$

where $L : \mathbb{R}^d \times \mathbb{R}^m \times [t, T] \to \mathbb{R}$ is the running cost and $\lambda \geq 0$ is the discount factor. The optimal control problem reads:

$$\min_{u \in U} J_{x,t}(y, u), \tag{3}$$

Where $y(t)$ satisfies (1) for the control u. To guarantee existence and uniqueness of the control problem (3) we assume that the functions f, L, g are bounded the functions f, L are Lipschitz-continuous with respect to the first variable and the cost g is also Lipschitz-continuous.

We are interested in a control in feedback form, therefore we define the value function

$$v(x, t) := \inf_{u \in \mathcal{U}} J_{x,t}(u) \tag{4}$$

which satisfies the DPP, i.e. for every $\tau \in [t, T]$:

$$v(x, t) = \inf_{u \in \mathcal{U}} \left\{ \int_t^\tau L(y(s), u(s), s)e^{-\lambda(s-t)} ds + v(y(\tau), \tau)e^{-\lambda(\tau-t)} \right\}. \tag{5}$$

Due to (5) we can derive the HJB for every $x \in \mathbb{R}^d$, $s \in [t, T)$:

$$\begin{cases} -\dfrac{\partial v}{\partial s}(x, s) + \lambda v(x, s) + \\ \qquad \max\limits_{u \in U} \{-L(x, u, s) - \nabla v(x, s) \cdot f(x, u, s)\} = 0, \\ v(x, T) = g(x). \end{cases} \tag{6}$$

Finally, the computation of the feedback control is straightforward, assuming the value function is known:

$$u^*(t) := \arg\max_{u \in U} \{-L(x, u, t) - \nabla v(x, t) \cdot f(x, u, t)\}. \tag{7}$$

Analytical solution of Equation (6) is hard to find due to its nonlinearity and numerical methods should be able to handle discontinuities in the gradient (see Falcone et al. (2013) and the references therein). Here, we describe the time discretization of (6) with a time step $\Delta t := [(T - t)/\overline{N}]$ where $\overline{N}$ is the total number of steps. Thus, for $n = \overline{N} - 1, \ldots, 0$ and every $x \in \mathbb{R}^d$ we have

$$V^n(x) = \min_{u \in U} [\Delta t\, L(x, u, t_n) + e^{-\lambda \Delta t} V^{n+1}(x + \Delta t f(x, u, t_n))], \tag{8}$$

where $t_n = t + n\Delta t$, $t_{\overline{N}} = T$, and $V^n(x) := V(x, t_n)$. The above iterative scheme is coupled with the terminal condition

$$V^{\overline{N}}(x) = g(x). \tag{9}$$

Note that usually the term $V^{n+1}(x + \Delta t f(x, u, t_n))$ is computed by interpolation on a grid since $x + \Delta t f(x, u, t_n)$ is usually not a grid point. To avoid this interpolation we build a non-structured grid on a tree structure where we compute all the possible combinations of the term $x + \Delta t f(x, u, t_n)$ as follows.

To begin with we consider a discretize version of the control domain, say $U = \{u_1, ..., u_M\}$ with M controls. Then, we will denote the tree by $\mathcal{T} := \cup_{j=0}^{\overline{N}} \mathcal{T}^j$, where each $\mathcal{T}^j$ contains the nodes of the tree at time t_j. The first level $\mathcal{T}^0 = \{x\}$ is simply given by the initial condition x. Then, starting from x, we consider all the nodes obtained following the dynamics (1) discretized using e.g. an explicit Euler scheme with different discrete controls $u_j \in U$

$$\zeta_j^1 = x + \Delta t\, f(x, u_j, t_0), \qquad j = 1, \ldots, M.$$

Therefore, we have $\mathcal{T}^1 = \{\zeta_1^1, \ldots, \zeta_M^1\}$. Note that our goal is to obtain a high order method and this is the purpose of the next section.

All the nodes can be characterized by their n−th *time level*, as follows

$$\mathcal{T}^n = \{\zeta_i^{n-1} + \Delta t f(\zeta_i^{n-1}, u_j, t_{n-1})\}_{j=1}^M \quad i = 1, \ldots, M^{n-1}.$$

All the nodes of the tree can be shortly defined as

$$\mathcal{T} := \{\zeta_j^n\}_{j=1}^{M^n}, \quad n = 0, \ldots \overline{N},$$

where the nodes ζ_i^n are the results of the dynamics at time t_n with the controls $\{u_{j_k}\}_{k=0}^{n-1}$:

$$\begin{aligned} \zeta_{i_n}^n &= \zeta_{i_{n-1}}^{n-1} + \Delta t f(\zeta_{i_{n-1}}^{n-1}, u_{j_{n-1}}, t_{n-1}) \\ &= x + \Delta t \sum_{k=0}^{n-1} f(\zeta_{i_k}^k, u_{j_k}, t_k), \end{aligned}$$

with $\zeta^0 = x$, $i_k = \left\lfloor \dfrac{i_{k+1}}{M} \right\rfloor$ and $j_k \equiv i_{k+1} \bmod M$. We note that $\zeta_i^k \in \mathbb{R}^d, i = 1, \ldots, M^k$.

Although the tree structure allows to solve high dimensional problems, its construction might be expensive since:

$$|\mathcal{T}| = \sum_{i=0}^{\overline{N}} M^i = \frac{M^{\overline{N}+1} - 1}{M - 1},$$

where M is the number of controls and $\overline{N}$ the number of time steps. The cost of the problem increases exponentially and it is clear that the algorithm might be unfeasible due to the huge amount of memory allocations, if M or $\overline{N}$ are too large. To mitigate this problem we introduce a pruning rule based on the fact that the numerical value function is Lipschitz continuous (Alla et al. (2018)), therefore $\zeta_i^n \approx \zeta_j^n$ implies $V^n(\zeta_i^n) \approx V^n(\zeta_j^n)$ for $i \neq j$ and $n = 1, \ldots \overline{N}$. Defining a threshold $\varepsilon_{\mathcal{T}} > 0$ based on the distance between ζ_i^n and ζ_j^n we can cut off several branches of the tree and merge the nodes

$$\|\zeta_i^n - \zeta_j^n\| \leq \varepsilon_{\mathcal{T}}, \text{ for } i \neq j \text{ and } n = 0, \ldots, \overline{N}. \tag{10}$$

The pruning rule (10) helps to save a huge amount of memory Alla et al. (2018b). Later, we will show how to choose this threshold to guarantee the same order of convergence of the numerical method used to get an approximation of (1).

The knowledge of the tree $\mathcal{T}$ allows to drop every kind of interpolation in (8). Another advantage of the method is that we avoid to define an arbitrary numerical domain which is usually hard to choose and needs artificial boundary conditions. Furthermore, we gain the possibility to work with $d \gg 10$ and to deal also with the control of PDEs which is usually hard to attack.

The numerical value function $V(x,t)$ will be computed on the tree nodes in space as

$$V(x, t_n) = V^n(x), \quad \forall x \in \mathcal{T}^n, \tag{11}$$

where $t_n = t + n\Delta t$. Then, the computation of the value function follows directly from the DPP. The TSA defines a grid $\mathcal{T}^n = \{\zeta_j^n\}_{j=1}^{M^n}$ for $n = 0, \ldots, \overline{N}$, we can write a time discretization for (6) as follows:

$$\begin{cases} V^n(\zeta_i^n) = \min\limits_{u \in U}\{e^{-\lambda \Delta t} V^{n+1}(\zeta_i^n + \Delta t f(\zeta_i^n, u, t_n)) + \\ \qquad \Delta t\, L(\zeta_i^n, u, t_n)\}, \\ \qquad\qquad \zeta_i^n \in \mathcal{T}^n, n = \overline{N}-1, \ldots, 0, \\ V^{\overline{N}}(\zeta_i^{\overline{N}}) = g(\zeta_i^{\overline{N}}), \qquad\qquad \zeta_i^{\overline{N}} \in \mathcal{T}^{\overline{N}}. \end{cases} \tag{12}$$

We recall the set of controls U is discrete and we compute the minimization by comparison. A detailed comparison and discussion about the classical method and tree structure algorithm can be found in Alla et al. (2018).

Finally, we describe how we obtain the feedback control on a tree structure. We note that we use the same discretized set U used to approximate the value function. Therefore, during the computation of the value function, we store the control indices corresponding to the argmin in (12). Clearly, with the tree structure all the possible trajectories are already computed and we only need to store the indices of the tree that provide the optimal path starting from $\zeta_*^0 = x$ in the following way

$$\begin{aligned} u_n^* := \arg\min_{u \in U} \{&e^{-\lambda \Delta t} V^{n+1}(\zeta_*^n + \Delta t f(\zeta_*^n, u, t_n)) + \\ &\Delta t\, L(\zeta_*^n, u, t_n)\}, \end{aligned} \tag{13}$$

$$\zeta_*^{n+1} \in \mathcal{T}^{n+1} \text{ such that } \zeta_*^n \to^{u_n^*} \zeta_*^{n+1},$$

for $n = 0, \ldots, \overline{N}-1$, where the symbol $\to^u$ stands for the connection of two nodes by the control u.

3. HIGH-ORDER SCHEMES BASED ON THE TREE STRUCTURE

In the previous section we recalled the TSA using a forward Euler scheme which leads to a first order convergence as shown by the numerical tests inAlla et al. (2018). However, our approach can be easily extended to high order schemes. This is the goal of the section and the novelty of the current work.

Let us consider a high-order approximation scheme for the cost functional (2) and high-order scheme to solve numerically the dynamics (1) under the assumptions on L, g and f provided in Section 2. As already suggested in Falcone et al. (1994) for the infinite horizon problem, we introduce a one-step approximation for the dynamics (1) as follows

$$\begin{cases} y^{n+1} = y^n + \Delta t \Phi(y^n, \mathbf{U}, t_n, \Delta t), \\ y^0 = x, \end{cases} \tag{14}$$

where the admissible control matrices at time t_n is $\mathbf{U} = U \times U \ldots \times U \in \mathbb{R}^{M \times (q+1)}$ matrix with $U \subset \mathbb{R}^M$ the discretized control set defined in Section 2 and $q+1$ is the number of stages of the numerical method for the ODE (it is also possible to consider a time dependence of U as in Falcone et al. (1994) but we will avoid this complication here) . We denote by u_i^n the $i-$th control of U for the $n-$th column of $\mathbf{U}$. We further assume that the function Φ in (14) is consistent

$$\lim_{\Delta t \to 0} \Phi(x, \bar{\mathbf{u}}, t, \Delta t) = f(x, \bar{\mathbf{u}}, t), \tag{15}$$

where $\bar{\mathbf{u}} = (\bar{u}, \ldots, \bar{u}) \in \mathbf{U}$ for $\bar{u} \in U$ and Lipschitz continuous:

$$|\Phi(x, \mathbf{U}, t, \Delta t) - \Phi(y, \mathbf{U}, t, \Delta t)| \leq L_\Phi |x - y|, \tag{16}$$

for any admissible set U and $0 < \Delta t < \overline{\Delta t} < \infty$. Under these assumptions the scheme (14) is convergent. Then, we consider the approximation of the cost functional

$$\begin{aligned} J_{x,t}^{\Delta t}(\mathbf{U}) = \Delta t \sum_{m=n}^{N-1} \sum_{i=0}^{q} w_i L(y^{m+\tau_i}, u_i^m, t_m) e^{-\lambda t_m} \\ + g(y^N) e^{-\lambda T}, \end{aligned} \tag{17}$$

where τ_i and w_i are the nodes and weights of the quadrature formula satisfying:

$$0 \leq \tau_i \leq 1, \qquad \omega_i \geq 0, \qquad \sum_{i=0}^{q} w_i = 1.$$

Finally, we define the numerical value function as

$$V(t, x) = \inf_{\mathbf{U}} J_{x,t}^{\Delta t}(\mathbf{U}). \tag{18}$$

Following Falcone et al. (1994), it is possible to prove the extended DPP which reads:

$$\begin{aligned} V(t, x) = \inf_{\mathbf{U}} \Bigg\{ &\Delta t \sum_{i=0}^{q} w_i L(y^{n+\tau_i}, u_i^n, t_{n+\tau_i}) e^{-\lambda(t_{n+\tau_i} - t)} + \\ &V(t_{n+1}, y^{n+1}) e^{-\lambda(t_{n+1} - t)} \Bigg\}. \end{aligned} \tag{19}$$

Under our assumptions on L and f (see Section 2), it is easy to check that V is Lipschitz-continuous and bounded and this will guarantee the convergence of the numerical scheme.

Note that for $q = 0$ in (19) we obtain the standard formulation with Euler method:

$$V(t, x) = \min_{u \in U} \{\Delta t L(x, u, t) + V(t + \Delta t, x + \Delta t f(x, u, t))\}.$$

In the case of Heun's scheme, e.g. $q = 1$, equation (19) becomes

$$\begin{aligned} V(t, x) = \min_{(\bar{u}_0, \bar{u}_1) \in U \times U} \Bigg\{ &\frac{\Delta t}{2}(L(x, \bar{u}_0, t) + \\ &L(x + \Delta t \Phi(x, t, \{\bar{u}_0, \bar{u}_1\}, \Delta t), \bar{u}_1, t + \Delta t)) + \\ &+ V(t + \Delta t, x + \Delta t \Phi(x, t, \{\bar{u}_0, \bar{u}_1\})) \Bigg\}, \end{aligned} \tag{20}$$

where

$$\Phi(x,t,\{\bar u_0,\bar u_1\},\Delta t) = \frac{1}{2}\left(f(x,t,\bar u_0)\right.$$
$$\left. + f(x+\Delta t f(x,t,\bar u_0), t+\Delta t, \bar u_1)\right). \tag{21}$$

We further note that it is possible to deal with implicit numerical schemes in equation (19) using e.g. the trapezoidal rule which reads:

$$V(t,x) = \min_{(\bar u_0,\bar u_1)\in U\times U}\left\{\frac{\Delta t}{2}(L(x,\bar u_0,t)\right.$$
$$+L(y^{n+1}(\bar u_0,\bar u_1),\Delta t),\bar u_1, t+\Delta t))$$
$$\left.+V(t+\Delta t, y^{n+1}(\bar u_0,\bar u_1))\right\}, \tag{22}$$

where $y^{n+1}(\bar u_0,\bar u_1)$ is obtained solving

$$y^{n+1}(\bar u_0,\bar u_1) = x + \frac{\Delta t}{2}\left(f(x,t,\bar u_0)\right.$$
$$\left.+f(y^{n+1}(\bar u_0,\bar u_1), t+\Delta t, \bar u_1)\right). \tag{23}$$

Note that the cardinality of the tree $\mathcal{T}$ will significantly increase when dealing with high order schemes. Therefore, a pruning rule (10) is needed. It is possible to prove that if Φ is Lipschitz-continuous with constant L_Φ, then

$$|y^n(u) - \eta^n(u)| \le n\varepsilon_{\mathcal{T}} e^{L_\Phi(t_n-t)},$$

where $\eta^n(u)$ is the pruned trajectory (for more details on the definition of the pruned trajectory we refer to Alla et al. (2018b)). Furthermore, to guarantee a convergence of order p, the tolerance $\varepsilon_{\mathcal{T}}$ must satisfy

$$\varepsilon_{\mathcal{T}} \le C_{\varepsilon_{\mathcal{T}}}\Delta t^{p+1}.$$

4. NUMERICAL TESTS

In this section we are going to test the method and show some numerical results for two test cases. In the first, we deal with an ordinary differential equation and we show the order of convergence of the method for Euler and Heun's schemes. The second test deals with a linear Partial Differential Equation and we show the effectiveness of working with high-order methods. The numerical simulations reported in this paper were performed on a laptop with 1CPU Intel Core i5-3, 1 GHz and 8GB RAM. The codes are written in C++.

4.1 Test 1: Comparison with exact solution of the value function

In this test we compute the order and the errors of the TSA in an example where the exact value function is known analytically. We consider the following dynamics in (1)

$$f(x,u) = \begin{pmatrix} u \\ x_1^2 \end{pmatrix}, \; u \in U \equiv [-1,1], \tag{24}$$

where $x = (x_1, x_2) \in \mathbb{R}^2$ and $T = 1$. The cost functional in (2) is defined by the following choices:

$$L(x,u,t) = 0, \qquad g(x) = -x_2, \qquad \lambda = 0, \tag{25}$$

where we only consider the terminal cost g. The solution of the HJB equation is

$$v(x,t) = -x_2 - x_1^2(T-t) - \frac{1}{3}(T-t)^3 - |x_1|(T-t)^2. \tag{26}$$

In this example, we use the TSA for both forward Euler and Heun's scheme with and without the pruning criteria (10). We compare two different approximations according to ℓ_2-relative error with the exact solution on the tree nodes

$$\mathcal{E}_2(t_n) = \sqrt{\frac{\sum\limits_{x_i\in\mathcal{T}^n} |v(x_i,t_n) - V^n(x_i)|^2}{\sum\limits_{x_i\in\mathcal{T}^n} |v(x_i,t_n)|^2}}.$$

TSA easily provides higher order converging methods only modifying the numerical method that solves the ODEs and the quadrature formula for the cost functional. However, the case without pruning criteria becomes unfeasable for more than 10 time steps since it requires to store $O(M^{22})$ nodes applying Heun's scheme, whereas the application of pruning criteria (10) provides a real improvement. We are going to compute ℓ_2- error in time and in space

$$Err_{2,2} = \sqrt{\Delta t \sum_{n=0}^{N} \frac{\|V_{exact}(x_i,t_n) - V^n_{approx}(x_i)\|^2_{\ell^2(\mathcal{T}^n)}}{\|V_{exact}\|^2_{\ell^2(\mathcal{T}^n)}}}.$$

Figure 4.1 shows the order of convergence for forward Euler and Heun's method using different $\varepsilon_{\mathcal{T}}$. We note that we obtain first order of convergence when dealing with Euler scheme and $\varepsilon_{\mathcal{T}} = \Delta t^2$ and second order for Heun's approximation with $\varepsilon_{\mathcal{T}} = \Delta t^3$. We also show how crucial is the selection of the tolerance $\varepsilon_{\mathcal{T}}$.

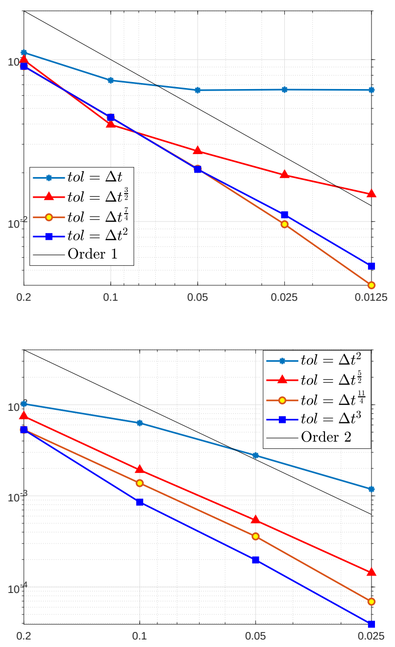


Fig. 1. Test 1: Comparison of the order of convergence for the pruned TSA with different tolerances (top) $\epsilon_{\mathcal{T}} = \{\Delta t, \Delta t^{3/2}, \Delta t^{7/4}, \Delta t^2\}$ with Euler method to approximate (1), (bottom) $\epsilon_{\mathcal{T}} = \{\Delta t^2, \Delta t^{5/2}, \Delta t^{11/4}, \Delta t^3\}$ with Heun method to approximate (1)

In Table 1 and Table 2 we present the results of the TSA applying the Euler scheme for $\varepsilon_{\mathcal{T}} = \{0, \Delta t^2\}$ respectively. We first note that the pruning criteria allows to solve the problem for a smaller temporal step size Δt since the cardinality of tree is smaller. The CPU time is then proportional to the cardinality of the tree. We also note that, as expected, the order of convergence is 1 in both cases.

Δt	Nodes	CPU	$Err_{2,2}$	$Order_{2,2}$
0.2	63	0.05s	9.0e-02	
0.1	2047	0.35s	4.4e-02	1.04
0.05	2097151	1.1s	2.2e-02	1.02

Table 1. Test 1: Error analysis and order of convergence for forward Euler scheme of the TSA without pruning rule ($\varepsilon_{\mathcal{T}} = 0$).

Δt	Nodes	CPU	$Err_{2,2}$	$Order_{2,2}$
0.2	42	0.05s	9.1e-02	
0.1	324	0.08s	4.4e-02	1.05
0.05	3151	0.6s	2.1e-02	1.04
0.025	29248	2.5s	1.1e-02	1.005
0.0125	252620	150s	5.3e-03	1.004

Table 2. Test 1: Error analysis and order of convergence for forward Euler scheme of the TSA with $\varepsilon_{\mathcal{T}} = \Delta t^2$.

In Table 3 and Table 4 we present the results obtained by means of the Heun's method. Similar considerations to the tables which refer to Euler scheme hold true. However, we note that the order of convergence is improved as Heun's method is a higher order scheme.

Δt	Nodes	CPU	$Err_{2,2}$	$Order_2$
0.2	1365	0.29s	3.51e-03	
0.1	1398101	3.92s	8.59e-04	2.0316

Table 3. Test 1: Error analysis and order of convergence for Heun's scheme of the TSA without pruning ($\varepsilon_{\mathcal{T}} = 0$).

Δt	Nodes	CPU	$Err_{2,2}$	$Order_2$
0.2	160	0.35s	5.32e-03	
0.1	2895	0.61s	8.53e-04	2.65
0.05	58888	60s	1.98e-04	2.11
0.025	1018012	9051s	3.9e-05	2.34

Table 4. Test 1: Error analysis and order of convergence for Heun's scheme of the TSA with $\varepsilon_{\mathcal{T}} = \Delta t^3$.

Finally, we note that to reach an error of order $O(10^{-3})$ using Euler method with pruning rule needs $150s, \Delta t = 0.0125$ and $|\mathcal{T}| = 252620$, whereas Heun's with pruning requires only $\Delta t = 0.2, |\mathcal{T}| = 160$ in $0.35s$. This shows that the choice of the numerical scheme of higher order allows accurate approximations in reasonable time. However, one should also consider that the comparison just discussed presents different pruning criteria. Thus, Euler scheme has order of convergence $O(\Delta t)$, whereas Heun's $O(\Delta t^2)$, which means that the same order is applied when $\Delta t_{euler} = \Delta t_{heun}^2$, if we apply Δt_{heun} for Heun's method. On the other hand the tolerance for Euler scheme will be Δt_{heun}^4, while for Heun's Δt_{heun}^3. To summarize Heun's scheme requires a bigger tolerance to obtain the same error order and, clearly, lower CPU time.

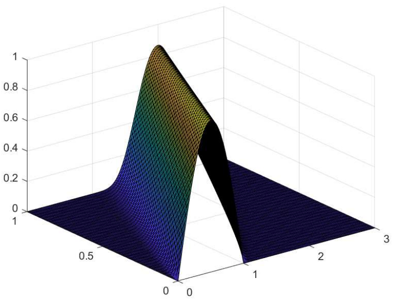
4.2 Test 2: Bilinear control for advection equation

In the second example we deal with advection equation:

$$\begin{cases} y_t + cy_x = yu(t) & (x,t) \in \Omega \times [0,T], \\ y(x,t) = 0 & (x,t) \in \partial\Omega \times [0,T], \\ y(x,0) = y_0(x) & x \in \Omega. \end{cases} \tag{27}$$

We consider a Finite Difference approximation for equation (27) and we can note that we fit into the abstract formulation (1). Here we use $\Delta x = 0.01$, $\Delta t = 0.025$, $\Omega = [0,3]$ and $c = 1.5$. The cost functional we want to minimize is of tracking-type:

$$J_{y_0,t}(u) = \int_t^T \left(\|y(s)\|_2^2 \, dx + 0.01|u(s)|^2 \right) ds + \|y(T)\|_2^2.$$



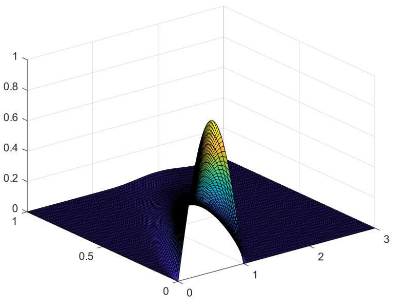


Fig. 2. Test 2: Uncontrolled (top) and controlled solution (bottom) using trapezoidal method.

To avoid narrow CFL conditions we are going to consider first and second order implicit schemes. In Figure 4.2 we show the results of the uncontrolled solution and the controlled solution using TSA and trapezoidal rule to approximate the dynamics. We note that the feedback has been built on the tree structure with the same control set as in the computation of the value function as explained in (13). A more sophisticated method to compute feedback control goes beyond the scopes of this paper and will be the focus of future works. As expected we can note that the controlled solution goes to zero faster than the uncontrolled one.

Since we do not know the value function in this case to show the effectiveness of the method we compare the values of the cost functionals in the top panel of Figure 4.2 for each time instance. As expected trapezoidal rule performs better than Euler method. In the bottom plot we show the final configuration at $T = 1$ for both controlled and uncontrolled solution with Euler and trapezoidal scheme. The comparison of the cost functional for the controlled dynamics may be not sufficient, since Euler scheme contains more numerical diffusion.

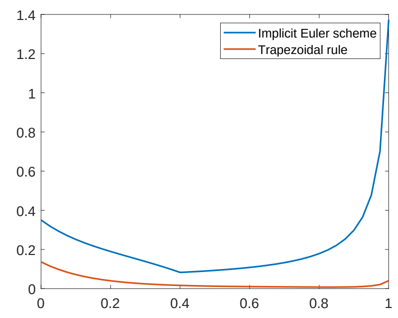


Fig. 3. Test 2: comparison normalized cost functionals

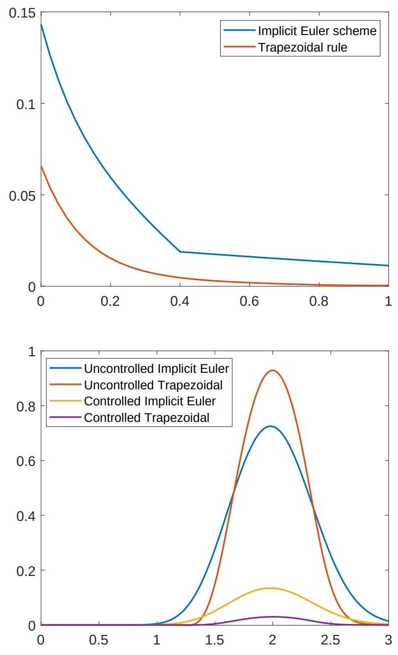


Fig. 4. Test 2: comparison cost functionals (top), dynamics at final time (bottom)

To obtain a fair comparison we introduce the following normalized cost functional

$$\bar{J}_{y_0,t}(u) = \frac{J_{y_0,t}(u)}{J_{y_0,t}(0)}$$

in order to show the real efficiency of trapezoidal rule with respect to Euler scheme.

5. CONCLUSION

In this work we have extended o high-order methods the first order approximation scheme based on a tree structure proposed in Alla et al. (2018,b) t. In particular, we have shown numerically that with a tree structure we can achieve the same order of convergence as the numerical method used in the discretization of the ODEs. We have also tested our algorithm with high dimensional problem to show the advantages of the proposed approach. In particular, dealing with an hyperbolic problem as the advection equation we were able to control the solution with first and second order methods. Again, we want to emphasize the possibility to deal with PDEs via dynamic programming which was not even possible less than a decade ago.

REFERENCES

A. Alla, M. Falcone and L. Saluzzi. *An efficient DP algorithm on a tree-structure for finite horizon optimal control problems*, submitted, 2018 `https://arxiv.org/pdf/1807.11008.pdf`

A. Alla, M. Falcone and L. Saluzzi. *Error estimates for a tree structure algorithm for dynamic programming equations*, submitted, 2018

A. Alla, M. Falcone and S. Volkwein, *Error analysis for POD approximations of infinite horizon problems via the dynamic programming approach*, SIAM J. Control Optim. **55**, 5, 3091–3115

M. Bardi and I. Capuzzo-Dolcetta. Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations. Birkhäuser, Basel, 1997.

E. Carlini, M. Falcone and R. Ferretti, *An efficient algorithm for Hamilton-Jacobi equations in high dimension*, Comput. Vis. Sc., **7** (2004), pp. 15–29.

I. Capuzzo Dolcetta, H. Ishii. *On the rate of convergence in homogenization of Hamilton-Jacobi equations*, Indiana University MAthematical Journal, **50**, 2001, 77-109.

M. Falcone. *A numerical approach to the infinite horizon problem of deterministic control theory*, Applied Mathematics and Optimization, **15**, 1987, 1-13.

M. Falcone, R. Ferretti, *Discrete time high-order schemes for viscosity solutions of Hamilton-Jacobi-Bellman equations*, Numerische Mathematik, **67**, 1994, 315–344.

M. Falcone and R. Ferretti. Semi-Lagrangian Approximation Schemes for Linear and Hamilton-Jacobi equations, SIAM, 2013.

K. Kunisch, S. Volkwein, and L. Xie. *HJB-POD based feedback design for the optimal control of evolution problems*, SIAM J. on Applied Dynamical Systems, **4**, 2004, 701–722.