Hamburger Beiträge
zur Angewandten Mathematik

Coupling MPC and DP methods for an efficient solution of optimal control problems

A. Alla, G. Fabrini, and M. Falcone

Nr. 2015-38
October 2015

Coupling MPC and DP methods for an efficient solution of optimal control problems.

A. Alla, G. Fabrini, and M. Falcone

University of Hamburg, Department of Mathematics, Hamburg, Germany
(e-mail: alessandro.alla@uni-hamburg.de),
Università di Genova, DIME, Genova, Italy
and Laboratoire Jacques-Louis Lions, Sorbonne Universités, UPMC Univ Paris 06
(e-mail: fabrini@dime.unige.it),
La Sapienza Università di Roma, Dipartimento di Matematica, Roma, Italy
(e-mail: falcone@mat.uniroma1.it)

Abstract. We study the approximation of optimal control problems via the approximation of a Hamilton-Jacobi equation in a tube around a reference trajectory which is first obtained solving a Model Predictive Control problem. The coupling between the two methods is introduced to improve the initial local solution and to reduce the computational complexity of the Dynamic Programming algorithm. We present some features of the method and show some results obtained via this technique showing that it can produce an improvement with respect to the two uncoupled methods.

Keywords: Optimal control, Dynamic Programming, Model Predictive Control, finite difference schemes, semi-Lagrangian schemes.

1 Introduction

The numerical solution of partial differential equations obtained by applying the Dynamic Programming Principle (DPP) to nonlinear optimal control problems is a challenging topic that can have a great impact in many areas, e.g. robotics, aeronautics, electrical and aerospace engineering. Indeed, by means of the DPP one can characterize the value function of a fully–nonlinear control problem (including also state/control constraints) as the unique viscosity solution of a nonlinear Hamilton–Jacobi equation, and, even more important, from the solution of this equation one can derive the approximation of a feedback control. This result is the main motivation for the PDE approach to control problems and represents the main advantage over other methods, such as those based on the Pontryagin minimum principle. It is worth to mention that the characterization via the Pontryagin principle gives only necessary conditions for the optimal trajectory and optimal open-loop control. Although from the numerical point of view the control system can be solved via shooting methods for the associated two point boundary value problem, in real applications a good initial guess for the co-state is particularly difficult and often requires a long and tedious trial-and-error procedure to be found. In any case, it can be interesting to obtain a local version of the DP method around a reference trajectory to improve a sub-optimal strategy. The reference trajectory can be obtained via the Pontryagin principle (with open-loop controls), via a Model Predictive Control (MPC) approach (using feedback sub-optimal controls) or simply via the already known engineering experience. The application of DP in an appropriate neighborhood of the reference trajectory will not guarantee the global optimality of the new feedback controls but could improve the result within the given constraints.

In this paper we focus our attention on the coupling between the MPC approach and the DP method. Although this coupling can be applied to rather general nonlinear control problems governed by ordinary differential equations we present the main ideas of this approach using the *infinite horizon optimal control*, which is associated to the following Hamilton-Jacobi-Bellman equation:

$$\lambda v(x) + \max_{u\in U}\{-f(x,u)\cdot Dv(x) - \ell(x,u)\} = 0, \qquad \text{for } x \in \mathbb{R}^d .$$

For numerical purposes, the equation is solved in a bounded domain $\Omega \subset \mathbb{R}^d$, so that also boundary conditions on $\partial\Omega$ are needed. A rather standard choice when one does not have additional information on the solution is to impose state constraints boundary conditions. It is clear that the domain Ω should be large enough in order to contain as much information as possible. It is, in general, computed without any information about the optimal trajectory. Here we construct the domain Ω around a reference trajectory obtained by a fast solution with a Model Predictive Control (MPC). MPC is a receding horizon method which allows to compute optimal solution for a given initial condition by solving iteratively a finite horizon open-loop problem (see [5, 7]).

2 A local version of DP via MPC models

Let us present the method for the classical *infinite horizon problem.* Let the controlled dynamics be given by

$$\begin{cases} \dot{y}(t) = f(y(t),u(t)), & t>0 \\ y(0) = x \end{cases} \tag{1}$$

where $y \in \mathbb{R}^d$, $u \in \mathbb{R}^m$ and $u \in \mathcal{U} \equiv \{u : \mathbb{R}_+ \to U, \text{ measurable}\}$. If f is Lipschitz continuous with respect to the state variable and continuous with respect to (x,u), the classical assumptions for the existence and uniqueness result for the Cauchy problem (1) are satisfied. To be more precise, the Carathéodory theorem (see [2]) implies that for any given control $u(\cdot) \in \mathcal{U}$ there exists a unique trajectory $y(\cdot;u)$ satisfying (1) almost everywhere. Changing the control policy the trajectory will change and we will have a family of infinitely many solutions of the controlled system (1) parametrized with respect to the control u.
Let us introduce the *cost functional* $J : \mathcal{U} \to \mathbb{R}$ which will be used to select the *optimal trajectory.* For the infinite horizon problem the cost functional is

$$J_x(u(\cdot)) = \int_0^\infty \ell(y(s),u(s))e^{-\lambda s}ds\,, \tag{2}$$

where ℓ is Lipschitz continuous in both arguments and $\lambda > 0$ is a given parameter. The function ℓ represents the running cost and λ is the discount factor which allows to compare the costs at different times rescaling the costs at time 0. From the technical point of view, the presence of the discount factor guarantees that the integral is finite whenever ℓ is bounded, i.e. $||\ell||_\infty \leq M_\ell$. In this section we will summarize the basic results for the two methods as they are the building blocks for our new method.

2.1 Hamilton–Jacobi–Bellman equations

The essential features will be briefly sketched, and more details in the framework of viscosity solutions can be found in [2, 4].
Let us define the value function of the problem as

$$v(x) = \inf_{u(\cdot)\in\mathcal{U}} J_x(u(\cdot)) \,. \tag{3}$$

It is well known that passing to the limit in the Dynamic Programming Principle one can obtain a characterization of the value function in terms of the following first order non linear Bellman equation

$$\lambda v(x) + \max_{u\in U}\{-f(x,u)\cdot Dv(x) - \ell(x,u)\} = 0, \qquad \text{for } x \in \mathbb{R}^d \,. \tag{4}$$

Several approximation schemes on a fixed grid G have been proposed for (4). To simplify the presentation, let us consider a uniform structured grid with constant space step $k := \Delta x$. We will use a semi-Lagrangian method based on a Discrete Time Dynamic Programming Principle, a first discretization in time of the original control problem leads to a characterization of the corresponding value function v^h (for the time step $h := \Delta t$) as

$$v^h(x) = \min_{u\in U}\{e^{-\lambda h} v_h\left(x + hf\left(x,u\right)\right) + h\ell\left(x,u\right)\}. \tag{5}$$

Then, we have to project on the grid and reconstruct the value $v_h\left(x + hf\left(x,u\right)\right)$ by interpolation (for example by a linear interpolation). Finally, we obtain the following fixed point formulation of the DP equation

$$w(x_i) = \min_{u\in U}\{e^{-\lambda h} w\left(x_i + hf\left(x_i,u\right)\right) + h\ell\left(x_i,u\right)\}, \text{ for } x_i \in G, \tag{6}$$

where $w(x_i) = v^{h,k}(x_i)$ is the approximation of the value function at the node x_i. Under appropriate assumptions, $v^{h,k}$ converges to $v(x)$ when $(\Delta t, \Delta x)$ goes to 0 (precise a-priori-estimates are available, e.g. [3] for more details). This method is referred in the literature as the *value iteration method* because, starting from an initial guess for the value function, it modifies the values on the grid according to the foot of the characteristics. It is well-known that the convergence of the value iteration can be very slow, since the contraction constant $e^{-\lambda\Delta t}$ is close to 1 when Δt is close to 0. This means that a higher accuracy will also require more iterations. Then, there is a need for an acceleration technique in order to cut the link between accuracy and complexity of the value iteration. One possible choice is the iteration in the policy space or the coupling between value iteration and the policy iteration in [1]. We refer the interested reader to the book [4] for a complete guide on the numerical approximation of the equation and the reference therein. One of the strength of this method is that it provides the feedback control once the value function is computed (and the feedback is computed at every node even in the fixed point iteration). In fact, we can characterize the optimal feedback control everywhere in Ω

$$u^*(x) = \arg\min_{u\in U}\{-f(x,u)\cdot Dv(x) - \ell(x,u)\}, \quad x \in \Omega,$$

where Dv is an approximation of the value function obtained by the values at the nodes.

2.2 Model Predictive Control

Nonlinear model predictive control (NMPC) is an optimization based method for the feedback control of nonlinear systems. It consists on solving iteratively a finite horizon open loop optimal control problem subject to system dynamics and constraints involving states and controls.
The infinite horizon problem, described at the beginning of Section 2, turns out to be computationally unfeasible for the open-loop approach therefore we solve a sequence of finite horizon problems. In order to formulate the algorithm we need to introduce the finite horizon cost functional:

$$J^N_{y_0}(u(\cdot)) = \int_{t_0}^{t_0^N} \ell(y(s),u(s))e^{-\lambda s}ds$$

where N is a natural number, $t_0^N = t_0 + N\Delta t$ is the final time, $N\Delta t$ denotes the length of the prediction horizon for the chosen time step $\Delta t > 0$ and the state y solves $\dot{y}(t) = f(y(t),u(t)),\ y(t_0) = y_0,\ t \in [t_0, t_0^N)$ and is denoted by $y(\cdot, t_0; u(\cdot))$. We also note that $y_0 = x$ at $t = 0$ as in equation (1).
The basic idea of NMPC algorithm is summarized at the end of sub-section.

The method works as follows: we store the optimal control on the first subinterval $[t_0, t_0 + \Delta t]$ together with the associated optimal trajectory. Then, we initialize a new finite horizon optimal control problem whose initial condition is given by the optimal trajectory $y(t) = y(t; t_0, u^N(t))$ at $t = t_0 + \Delta t$ using the sub-optimal control $u^N(t)$ for $t \in (t_0, t_0 + \Delta t]$. We iterate this process by setting $t_0 = t_0 + \Delta t$. Note that (7) is an open loop problem on a finite time horizon $[t_0, t_0 + N\Delta t]$ which can be treated by classical techniques, see e.g. [6]. The interested reader can find in [5] a detailed presentation of the method and a long list of references.

In general, the larger the prediction horizon, the better the feedback law one can obtain. However, one is interested in short prediction horizons (or even horizon of minimal length) while guaranteeing stabilization properties of the

MPC scheme (see [5]). The computation of this minimal horizon is related to a relaxed dynamic programming principle in terms of the value function for the finite horizon problem (7).

Start: `choose` $\Delta t > 0$, $N \in \mathbb{N}$, $\lambda > 0$.
for $n = 0, 1, 2, \ldots$
Step 1: `Compute the state` $y(t_n)$ `of the system at` $t_n = n\Delta t$,
Step 2: `Set` $t_0 = t_n = n\Delta t$, $y_0 = y(t_n)$ `and compute a global solution`,

$$u^N(t) := \arg\min_{u \in \mathcal{U}} J^N_{y_0}(u(t_0)). \tag{7}$$

Step 3: `Define the MPC feedback value` $u^N(t)$, $t \in (t_0, t_0 + \Delta t]$ `and use this control to compute the associated state` $y = y(t; t_0, u^N(t))$ `by solving the dynamical system in` (1) `on` $[t_0, t_0 + \Delta t]$.
end for
end

2.3 Coupling MPC with Bellman Equations

The idea behind the coupling is to combine the advantages from both methods as follows. Let us assume that we are interested only on the approximation of the control problem for a given initial condition x and we would like to use the knowledge of the value function. First of all we have to select a domain where we are going to compute the approximate value function.

The idea of MPC is to compute optimal solution starting by a given initial condition whereas the knowledge of the value function allows the reconstruction of feedback control for any initial condition in the domain Ω. For this reason MPC can give a quick and reasonable reference trajectory y^{MPC} in order to build the domain Ω_ρ centered around it. The choice of the prediction horizon N here is crucial, we will assume to select a *short* prediction horizon in order to have a fast approximation of the initial guess and then build $\Omega\rho$ where we are going to apply the DP approach. It is clear that MPC may provide inaccurate solutions due to this choice but it is relevant to have some rough information about the trajectory and we set the HJB equation in a tube around y^{MPC}. The tube is defined as

$$\Omega_\rho := \{x \in \Omega : dist(x, y^{MPC}) \leq \rho\} \tag{8}$$

and is computed via the eikonal equation, i.e. we solve the Dirichlet problem

$$|\nabla v(x)| = 1, \quad x \in \mathbb{R}^N \backslash \mathcal{T}, \qquad \text{with } v(x) = 0, \quad x \in \mathcal{T} \tag{9}$$

where $\mathcal{T} = y^{MPC}$ is our target. We just want to mention that for that equation several fast methods (Fast Marching [8] and Fast Sweeping [9])have been proposed so that this step can be solved very efficiently (we refer to [4] for details on the weak solutions of the eikonal equation and their numerical approximations).

Solving the eikonal equation (9) (in the viscosity sense) we obtain the distance function from the target. Then, we choose a radius $\rho > 0$ in order to build the tube Ω_ρ. In this way the domain of the HJB is not built by scratch but takes into account some information on the controlled system. To localize the solution in the tube we impose state constraints boundary conditions on $\partial\Omega_\rho$ penalizing in the scheme (6) the points outside the domain. It is clear that a larger ρ will allow for a better approximation of the value function but at the same time enlarging ρ we will lose the localization around our trajectory increasing the number of nodes (and the CPU time). Finally, we compute the optimal feedback from the value function computed and the corresponding optimal trajectories in Ω_ρ The algorithm is summarized below:

Start: `Inizialization`
Step 1: `Solve MPC and compute` y_x^{MPC} `for a given initial condition` x
Step 2: `Compute the distance from` y_x^{MPC} `via the Eikonal equation`
Step 3: `Select the tube` Ω_ρ `with distance` ρ `with respect to` y_x^{MPC}
Step 4: `Compute the constrained value function` v^{tube} `in` Ω_ρ `via HJB`
Step 5: `Compute the optimal feedbacks and trajectory using` v^{tube}.
End

3 Numerical tests

In this section we present two numerical tests for the infinite horizon problem to illustrate the performances of the proposed algorithm. However, the localization procedure can be applied to more general optimal control problems.
All the numerical simulations have been made on a MacBook Pro with 1 CPU Intel Core i5 2.4 Ghz and 8GB RAM. The codes used for the simulations are written in Matlab. The routine for the approximation of MPC is provided in [5].

Test 1: 2D Linear Dynamics Let us consider the following controlled dynamics:

$$\begin{cases} \dot{y}(t) = u(t) \quad t \in [0, T] \\ y(0) = x \end{cases} \tag{10}$$

where $u = (u_1, u_2)$ is the control, $y : [0, T] \to \mathbb{R}^2$ is the dynamic and x is the initial condition. The cost functional we want to minimize is:

$$J_x(u) := \int_0^\infty \min\{y(t;u)^2, (y(t;u) - y_2)^2 - 1\} e^{-\lambda t}\, dt \tag{11}$$

where $\lambda > 0$ is the discount factor.
In this example, the running cost has two local minima: in $y_1 = (0, 0)$ and in $y_2 = (2, 2)$ (where the running cost is -1). Note that we have included a discount factor λ, which guarantees the integrability of the cost functional $J_x(u)$ and the existence and uniqueness of the viscosity solution. The main task of the discount factor is to penalize long prediction horizons. Since we want to make a comparison we introduce it also in the setting of MPC, although this is not a standard choice. As we mentioned, MPC will just provide a first guess which is used to define the domain where we are solving the HJB equation.

In this test the chosen parameters are: $u \in [-1,1]^2$, $\rho = 0.2$, $\Omega = [-4,6]^2$, $\Delta t_{MPC} = 0.05 = \Delta t_{HJB}$, $\Delta x_{HJB} = 0.025$, $\Delta\tau = 0.01$ (the time step to integrate the trajectories). In particular, we focus on $\lambda = 0.1$ and $\lambda = 1$. The number of controls are 21^2 for the value function and 3^2 for the trajectories. Note that the time step used in the HJB approach for the approximation of the trajectory ($\Delta\tau$) is smaller than the one used for MPC: this is because with MPC we want to have a rough and quick approximation of the solution. In Figure 1, we show

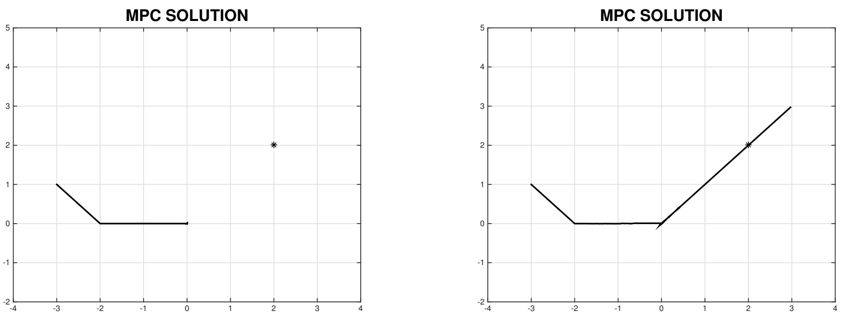

Fig. 1. Test 1: MPC solver with $\lambda = 0.1$ (left) and $\lambda = 1$ (right)

the results of MPC with $\lambda = 0.1$ on the left and $\lambda = 1$ on the right. As one can see, none of them is an accurate solution. In the first case, the solution goes to the local minimum $(0,0)$ and is trapped there, whereas when we increase λ the optimal solution does not stop at the global minimum y_2. On the other hand these two approximations help us to localize the behavior of the optimal solution in order to apply the Bellman equation in a reference domain Ω_ρ.
In Figure 2, we show the contour lines of value function in the whole interval Ω for $\lambda = 1$ and the corresponding value function in Ω_ρ. Finally, the optimal

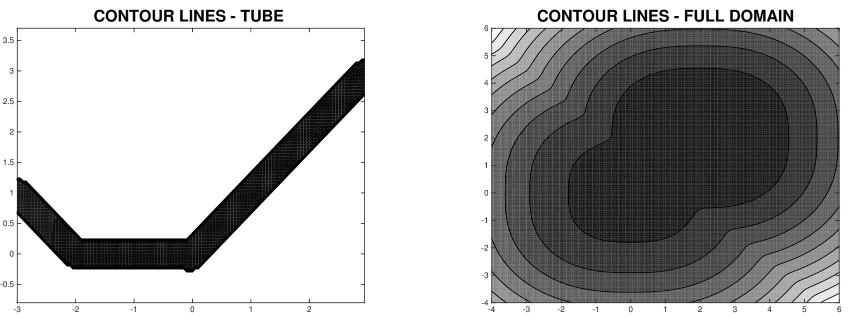

Fig. 2. Test 1: Contour lines of the value function in the tube Ω_ρ (left) and in Ω (right).

trajectories for $\lambda = 1$ are shown in Figure 3. On the right we propose the optimal solution obtained by the approximation of the value function in Ω whereas, on the left we can see the first approximation of the MPC solver (dotted line), the tube (solid lines) and the optimal solution via Bellman equation (dashed line). As you can see in the pictures the solutions provided from the DP approach in Ω and Ω_ρ are able to reach the global desired minimum y_2. In Table 1, we present

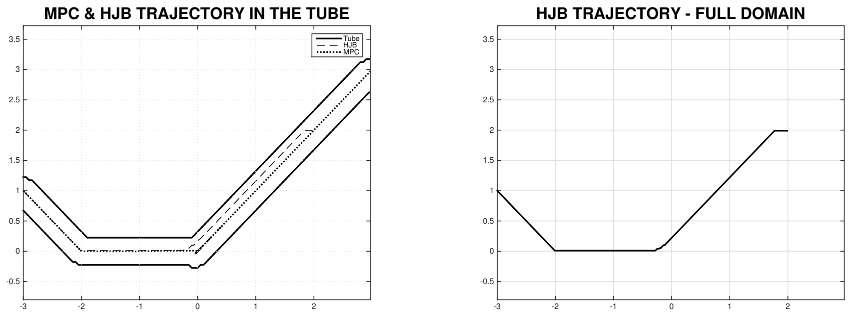

Fig. 3. Test 1: Optimal trajectory via MPC (dotted line) and via HJB (dashed line) in the tube (solid lines) (left), optimal trajectory via HJB in Ω (right).

$\lambda = 1$	**MPC N=5**	**HJB in Ω_ρ**	**HJB in Ω**
CPU	16s	239s	638s
$J_x(u)$	5.41	5.33	5.3

Table 1. A comparison of CPU time(seconds) and values of the cost functional.

the CPU time and the evaluation of the cost functional for different tests. As far as the CPU time is concerned, in the fourth column we show the global time needed to get the approximation of the value function in the whole domain and the time to obtain the optimal trajectory, whereas in the third column there is global time needed to compute the trajectory obtained via MPC, to build the tube, to compute the value function in the reduced domain and to compute the optimal trajectory. As we expected, the value of the cost functional is lower when we compute the value function in the whole domain (just because $\Omega_\rho \subset \Omega$). It is important to note that the approximation in Ω_ρ guarantees a reduction of the CPU time of the 62.5%.

Test 2: Van der Pol dynamics. In this test we consider the two-dimensional nonlinear system dynamics given by the Van Der Pol oscillator:

$$\begin{cases} \dot{x}(t) = y(t) \\ \dot{y}(t) = (1 - x(t)^2)y(t) - x(t) + u(t) \\ x(0) = x_0,\, y(0) = y_0. \end{cases} \tag{12}$$

The cost functional we want to minimize with respect to u is:

$$J_x(u) := \int_0^\infty (x^2 + y^2)e^{-\lambda t}\, dt. \tag{13}$$

We are dealing with a standard tracking problem where the state we want to reach is the origin. The chosen parameters are: $\lambda = \{0.1, 1\}$, $u \in [-1, 1]$, $\rho = 0.4$, $\Omega = [-6, 6]^2$, $\Delta t_{MPC} = 0.05 = \Delta t_{HJB}$, $\Delta x_{HJB} = 0.025$, $\Delta\tau = 0.01$, $x_0 = -3$, $y_0 = 2$. We took 21 controls for the approximation of the value function and 3 for the optimal trajectory. In Figure 4, we present the optimal trajectory: on the

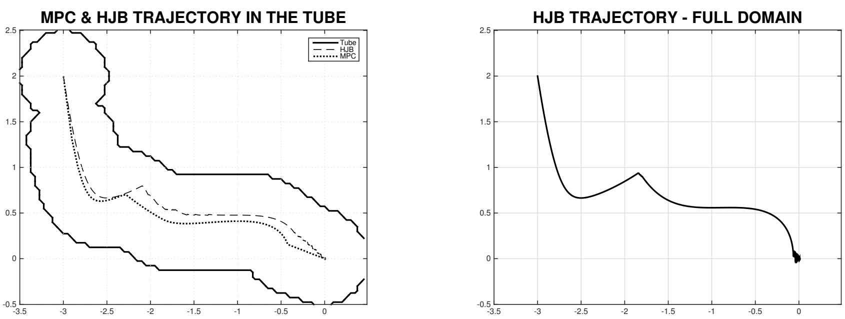

Fig. 4. Test 2: Optimal trajectory via MPC (dotted line) and via HJB (dashed line) in the tube Ω_ρ (left) and in Ω (right) for $\lambda = 0.1$.

right, the one obtained solving the HJB equation in the whole domain, on the left, the one obtained applying the algorithm we propose.
In Table 2 we present the CPU time and the evaluation of the cost functional with $\lambda = 0.1$ and $\lambda = 1$. In both case we can observe that the algorithm we popose is faster than solving HJB in the whole domain and the cost functional provides a value which improves the one obtained with the MPC algorithm.

4 Conclusions

We have proposed a local version of the dynamic programming approach for the solution of the infinite horizon problem showing that the coupling between

$\lambda = 0.1$	**MPC N=10**	**HJB in Ω_ρ**	**HJB in Ω**
CPU	79s	155s	228s
$J_x(u)$	14.31	13.13	12.41

$\lambda = 1$	**MPC N=10**	**HJB in Ω_ρ**	**HJB in Ω**
CPU	23s	49s	63s
$J_x(u)$	6.45	6.09	6.07

Table 2. Test 2: A comparison of CPU time (seconds) and values of the cost functional for $\lambda = \{0.1, 1\}$.

MPC and DP methods can produce rather accurate results. The coupling improves the original guess obtained by the MPC method and allows to save memory allocations and CPU time with respect to the global solution computed via Hamilton-Jacobi equations. An extension of this approach to other classical control problems and more technical details on the choice of the parameters λ and ρ will be given in a future paper.

References

1. A. Alla, M. Falcone and D. Kalise. An Efficient Policy Iteration Algorithm for Dynamic Programming Equations SIAM J. Sci. Comput. **37** (2015), no. 1, 181-200.
2. M. Bardi, I. Capuzzo Dolcetta. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations.* Birkhauser, Basel, 1997.
3. M. Falcone, Numerical solution of Dynamic Programming equations, Appendix A in the volume M. Bardi and I. Capuzzo Dolcetta, *Optimal control and viscosity solutions of Hamilton–Jacobi–Bellman equations*, Birkhäuser, Boston, 1997, 471-504.
4. M. Falcone, R. Ferretti. *Semi-Lagrangian Approximation Schemes for Linear and Hamilton-Jacobi Equations*, SIAM, 2013.
5. L. Grüne and J. Pannek. *Nonlinear Model Predictive Control*, Springer London, 2011.
6. J. Nocedal and S.J. Wright. *Numerical Optimization, second edition.* Springer Series in Operation Research and Financial Engineering, 2006.
7. J.B. Rawlings and D.Q. Mayne. *Model Predictive Control: Theory and Design.* Nob Hill Publishing, LLC, 2009.
8. J.A. Sethian, Level set methods and fast marching methods, Cambridge University Press, 1999.
9. H. Zhao A fast sweeping method for Eikonal equations Math. Comp., **74** (2005), 603-627.