# MPC for nonlinear systems: a comparative review of discretization methods

Guido Sánchez*, Marina Murillo*, Lucas Genzelis*, Nahuel Deniz*, Leonardo Giovanini*

\* Instituto de Investigación en Señales, Sistemas e Inteligencia Computacional, sinc(i),
UNL, CONICET, Ciudad Universitaria UNL, 4to piso FICH, (S3000) Santa Fe, Argentina
{gsanchez,mmurillo,lgenzelis,ndeniz,lgiovanini}@sinc.unl.edu.ar

*Abstract*—This work provides a comparative review of three different numerical methods generally used to discretize continuous-time non-linear equations appearing in model predictive control problems: direct multiple shooting, direct collocation and successive linearizations. An overview of the characteristics of each method is given and the performance of each method is evaluated through the simulation of two test cases.

*Index Terms*—discretization methods, model predictive control, real-time optimization, non-linear systems

## I. Introduction

In this work we review some of the most popular methods used to discretize non-linear continuous-time equations that describe the system dynamics in model predictive control (MPC) algorithms.

MPC is a control technique that uses models of the plant in order to predict the future behaviour of the system over a prediction horizon. Usually, these models are expressed mathematically as ordinary differential equations (ODE). As MPC is, in general, a discrete-time control technique, these ODEs need to be transformed into discrete-time difference equations using a proper discretization method.

The work by Rao [1] describes the most important methods that have been developed over the years for solving general optimal control problems, in particular, the two broad classes of indirect and direct methods are discussed. Diehl et al. [2] review numerical approaches to solve non-linear optimal control problems, and discuss three direct optimal control approaches in detail: (i) single shooting, (ii) collocation, and (iii) multiple shooting. Probably, the most commonly used method to discretize continuous-time non-linear equations is the direct single-shooting method due to its simplicity and because it works well for simple problems. In this method the controls are discretized and then a numerical routine is used to sequentially obtain the states as an initial value problem for the complete horizon. Single shooting is very simple; however, when the systems are highly non-linear or unstable it is known to cause problems [3].

In order to improve direct single shooting, multiple shooting adds more degrees of freedom and partitions the time interval into additional grid points, leading to the generation of many initial value problems, which improves the convergence and prevents the growth of the error introduced by poor initial data [4], [5].

Another method that works well is direct collocation, which add a series of *collocation points* and approximates the model equations using a polynomial basis. The method ensures that the model equation is satisfied on this intermediate points [6], [7]. The addition of these variables adds even more degrees of freedom than multiple shooting, but we are rewarded with less nonlinearity.

An alternative discretization method is successive linearization, which was presented by the authors in [8], [9]. With this discretization method the continuous-time system equation can be transformed into a set of discretized linear time-varying (LTV) equations.

In this work we mainly revisit the last three methods: i) direct multiple shooting, ii) direct collocation, and iii) successive linearization. This work is organized as follows: in Section II the problem formulation is presented. In Section III an overview of the three methods is given. In order to compare the discretization methods, two test simulation examples are given in Section IV. Finally, in Section V the conclusion of this work is stated.

## II. Problem formulation

In the following we consider continuous-time dynamical systems described by the following equation

$$\dot{x}(t) = f(x(t), u(t)), \tag{1}$$

where $x(t) \in \Re^{n_x}$ is the differential state and $u(t) \in \Re^{n_u}$ is the control. Function $f$ is assumed to be twice differentiable and maps into $\Re^{n_x}$. Given the dynamical system from Eq. (1), the MPC problem is defined as follows: determine the state $x_k \in \Re^{n_x}$ and the control $u_k \in \Re^{n_u}$ that solves the following optimization problem

$$
\begin{aligned}
\min_{(x,u)} \quad & \sum_{k=i}^{i+N-1} \ell_k(x_k, u_k) + \ell_N(x_{k+N}) \\
\text{s.t.} \quad & x_i - \bar{x}_i = 0, \\
& x_{k+1} - f_k(x_k, u_k) = 0, \\
& h_k(x_k, u_k) \leq 0, \\
& \forall k \in [i, \dots, i+N-1],
\end{aligned}
\tag{2}
$$

where $N$ is the control horizon, $\ell_k$ is the stage cost and $\ell_N$ is the terminal cost (a popular choice is the use of quadratic costs [10]). The parameter $\bar{x}_i$ is the fixed initial value, the function $f_k$ is the discrete version of Eq. (1), and the function

$h_k$ is the set of state and control constraints. In order to solve Problem (2) we need to find a method to discretize Eq. (1), thus allowing to solve a non-linear program (NLP) on the free variables $(x, u)$.

## III. DISCRETIZATION METHODS OVERVIEW

In this work we solve the MPC problem using numerical optimization methods, therefore the continuous-time dynamical system equations need to be discretized. During the analysis of the different methods, we will assume a piecewise constant control discretization

$$u(t) = u_k, \text{ for } t \in [t_k, t_{k+1}], \; k = i, \ldots, i + N - 1, \quad (3)$$

so we can focus on how the state trajectory discretization is handled.

### A. Multiple shooting

The direct multiple shooting method is due to Bock and Plitt [4] and proceeds as follows: first, the controls are discretized in a piecewise manner on a coarse grid as specified in Eq. (3). Then, each state $x_k$ is used inside the grid given by the discrete-time control horizon as a *shooting node* to solve the ordinary differential equation (ODE) given by Eq. (1) on each interval $[t_k, t_{k+1}]$, starting with the initial state $\bar{x}_i$. From the numerical solution of these initial value problems, we obtain the state trajectory

$$x_{k+1} = \Phi_k(x_k, u_k), \quad (4)$$

where $\Phi_k$ is a suitable integrator. Typically, $\Phi_k$ is a Runge Kutta 4 (RK4) approximation which is employed to solve each of the interval's ODE, where the user picks a step size $h > 0$ –usually equal to the sampling time– and then:

$$
\begin{aligned}
k_1 &= f(x_k, u_k), \\
k_2 &= f(x_k + \frac{h}{2}k_1, u_k), \\
k_3 &= f(x_k + \frac{h}{2}k_2, u_k), \\
k_4 &= f(x_k + hk_3, u_k), \\
x_{k+1} &= x_k + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4).
\end{aligned}
\quad (5)
$$

The RK4 method is reasonably simple and robust and is a good general candidate for numerical solution of differential equations.

### B. Direct Collocation

While multiple shooting only includes the state at the beginning of each control interval as a degree of freedom in the NLP (in addition to the discretized control and the parameters), in direct collocation the state at a set of *collocation points* (in addition to the beginning of the interval) are variables in the NLP. An example of these time points are the Legendre points of order $d = 3$:

$$\tau = [0, 0.112702, 0.500000, 0.887298]. \quad (6)$$

Keeping the same control discretization scheme, the complete list of time points inside the control horizon $N$, with $h_k := t_{k+1} - t_k$, is:

$$
\begin{aligned}
t_{k,j} &:= t_k + h_k\tau_j, \\
k &= i, \ldots, i + N - 1, \\
j &= 0, \ldots, d,
\end{aligned}
\quad (7)
$$

as well as the final time $t_{N,0}$. Also let $x_{k,j}$ denote the states at these time points. On each control interval, we shall define a Lagrangian polynomial basis:

$$L_j(\tau) = \prod_{r=0, r \neq j}^{d} \frac{\tau - \tau_r}{\tau_j - \tau_r}. \quad (8)$$

Since the Lagrangian basis satisfies:

$$L_j(\tau_r) = \begin{cases} 1, & \text{if } j = r \\ 0, & \text{otherwise} \end{cases}, \quad (9)$$

we can approximate the state trajectory approximation as a linear combination of these basis functions:

$$\tilde{x}_k(t) = \sum_{r=0}^{d} L_r\left(\frac{t - t_k}{h_k}\right) x_{k,r}. \quad (10)$$

In particular, we get approximations of the time derivative of the state at each collocation point (not including $\tau_0$):

$$\dot{\tilde{x}}_k(t_{k,j}) = \frac{1}{h_k}\sum_{r=0}^{d} \dot{L}_r(\tau_j)x_{k,r} := \frac{1}{h_k}\sum_{r=0}^{d} C_{r,j}x_{k,r} \quad (11)$$

as well as an approximation of the state at the end of the control interval:

$$\tilde{x}_{k+1,0} = \sum_{r=0}^{d} L_r(1)x_{k,r} := \sum_{r=0}^{d} D_r x_{k,r}. \quad (12)$$

Plugging the approximation of the state derivative given by Eq. (11) into the ODE gives us a set of collocation equations that needs to be satisfied for every state at every collocation point:

$$
\begin{aligned}
h_k f(t_{k,j}, x_{k,j}, u_k) - \sum_{r=0}^{d} C_{r,j}x_{k,r} &= 0, \\
k &= i, \ldots, i + N - 1, \\
j &= 1, \ldots, d.
\end{aligned}
\quad (13)
$$

Then, the approximation of the end state given by Eq. (12) gives us a set of continuity equations that must be satisfied for every control interval:

$$
\begin{aligned}
x_{k+1,0} - \sum_{r=0}^{d} D_r x_{k,r} &= 0, \\
k &= i, \ldots, i + N - 1.
\end{aligned}
\quad (14)
$$

The set of Eq. (13) and (14) take the place of the integrator call that was made in direct multiple shooting. In direct collocation, simulation and optimization proceed simultaneously. The advantages of collocation methods are (i) a very sparse NLP is obtained (ii) we can use knowledge of the state trajectory

in the initialization (iii) it shows fast local convergence (iv) it can treat unstable systems well, and (v) it can easily cope with state and terminal constraints. Its major disadvantage is that adaptive discretization error control needs regridding and thus changes the NLP dimensions [11].

### C. Successive linearization

The method of successive linearizations is based on the work of Murillo et al. [8], [9]. The main idea is that the dynamic behavior of the non-linear system of Eq. (1) can be approximated as a linear time-varying (LTV) model

$$x_{k+1|k} = \hat{A}_{k|k} x_{k|k} + \hat{B}_{u_{k|k}} u_{k|k}, \tag{15}$$

where the matrices $\hat{A}_{k|k}$ and $\hat{B}_{u_{k|k}}$ are discretized versions of the Jacobian matrices $A_{k|k}$ and $B_{u_{k|k}}$ of the continuous time non-linear system given by Eq. (1), and they are defined as follows

$$
\begin{aligned}
A_{k|k} &= \left. \frac{\partial f(x_k, u_k, d_k)}{\partial x_k} \right|_{(*)}, \\
B_{u_{k|k}} &= \left. \frac{\partial f(x_k, u_k, d_k)}{\partial u(k)} \right|_{(*)},
\end{aligned} \tag{16}
$$

where $(*)$ stands for the reference trajectory given to the states and controls. The matrices obtained from Eq. (16), must then be discretized using a proper method [12]. As a result, the optimization Problem (2) can be rewritten as follows:

$$
\begin{aligned}
\min_{(x,u)} \quad & \sum_{k=i}^{i+N-1} \ell_k(x_k, u_k) + \ell_N(x_{k+N}) \\
\text{s.t.} \quad & x_i - \bar{x}_i = 0, \\
& x_{k+1|k} = \hat{A}_{k|k} x_{k|k} + \hat{B}_{uk|k} u_{k|k}, \\
& h_k(x_k, u_k) \leq 0, \\
& \forall k \in [i, \dots, i+N-1],
\end{aligned} \tag{17}
$$

and thus can be solved using quadratic program (QP) solvers instead of NLP solvers, given that the set of constraints $h_k$ are linear.

## IV. EXAMPLES

In the following examples we will assume that no perturbations are present and the model used by the controller matches the plant. We will minimize a quadratic cost, so the terms $\ell_k$ and $\ell_N$ will be defined as

$$
\begin{aligned}
\ell_k &= (x_k - x_k^s)^T Q (x_k - x_k^s) + (u_k - u_k^s)^T R (u_k - u_k^s) \\
\ell_N &= x_{k+N}^T Q_N x_{k+N}
\end{aligned} \tag{18}
$$

where the matrices $Q \in \Re^{n_x \times n_x}$, $R \in \Re^{n_u \times n_u}$ and $Q_N \in \Re^{n_x \times n_x}$ are symmetric definite positive and $x_k^s$ and $u_k^s$ are the desired state and control setpoints.

### A. First example: Van der Pol oscillator

First, we will be driving a Van der Pol oscillator to the origin. The system equation is given by

$$
\begin{aligned}
\dot{x}_1 &= (1 - x_2^2) x_1 - x_2 + u, \\
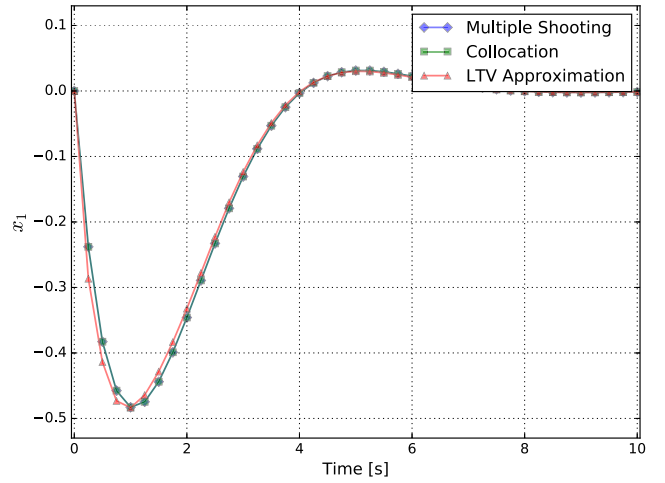\dot{x}_2 &= x_1,
\end{aligned} \tag{19}
$$



Fig. 1. State trajectory $x_1$ for the Van der Pol oscillator.

the initial condition is $x_1 = 0$, $x_2 = 1$ and the control input is subject to the following constraints

$$-0.75 \leq u \leq 1. \tag{20}$$

The chosen sampling time is $T_s = 0.25$ seconds, the horizon length is $N = 10$, and the weight matrices are $Q = Q_N = \text{diag}([1.0, 1.0, 1.0])$ and $R = 1$. The state trajectory obtained by each of the methods presented earlier can be seen in Figures 1 and 2, where the similarities between them can be observed.

The obtained control input (shown in Figure 3) is almost equal for multiple shooting and collocation, whereas the control input obtained by the successive linearization method presents some differences, mainly at the first iterations.

Finally, Figure 4 shows the execution time required to obtain the MPC solution at each discretization step. It can be seen that the successive linerarization method is faster than its counterparts. This is mainly because we used a QP solver and because we chose to use a receding horizon technique for the LTV model, updating only the last parameters of the corresponding LTV system and shifting the others. The mean time required to obtain a solution was 0.00205 seconds for multiple shooting, 0.00290 seconds for direct collocation and 0.00195 seconds for the successive linearizations method.

### B. Second example: Continuous stirred-tank reactor

In the second example, we will consider a continuous stirred-tank reactor (CSTR) taken from the work of Rawlings and Mayne [13]. Here, an irreversible, first-order reaction, $A \rightarrow B$, occurs in the liquid phase, and the reactor temperature is regulated with external cooling. The CSTR equation is given by

$$
\begin{aligned}
\dot{x}_1 &= F_0 \frac{c_0 - x_1}{\pi r^2 x_3} - k_0 x_1 \exp(-\frac{E}{R x_2}), \\
\dot{x}_2 &= F_0 \frac{T_0 - x_2}{\pi r^2 x_3} - \frac{\Delta H}{\rho C_p} k_0 x_1 \exp(-\frac{E}{R x_2}) \\
&\quad + 2 \frac{U}{r \rho C_p}(u_1 - x_2), \\
\dot{x}_3 &= \frac{F0 - u_2}{\pi r^2},
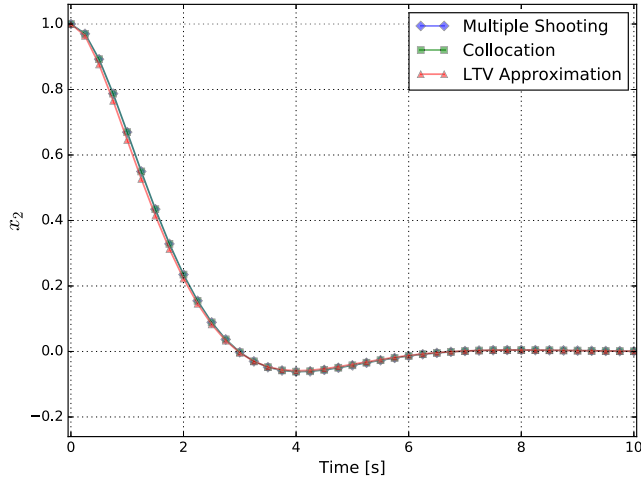\end{aligned} \tag{21}
$$

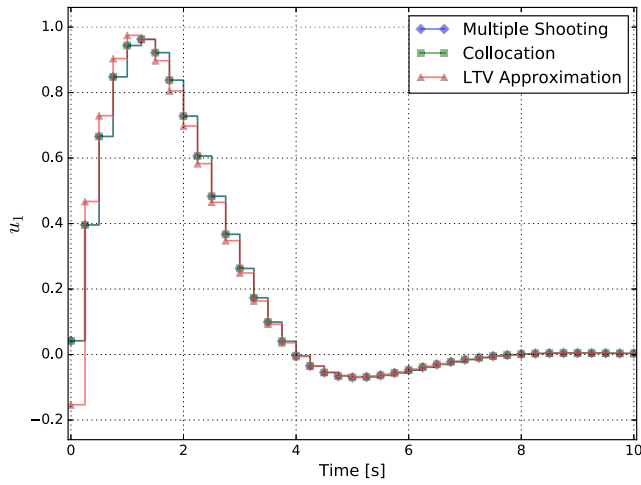Fig. 2. State trajectory $x_2$ for the Van der Pol oscillator.



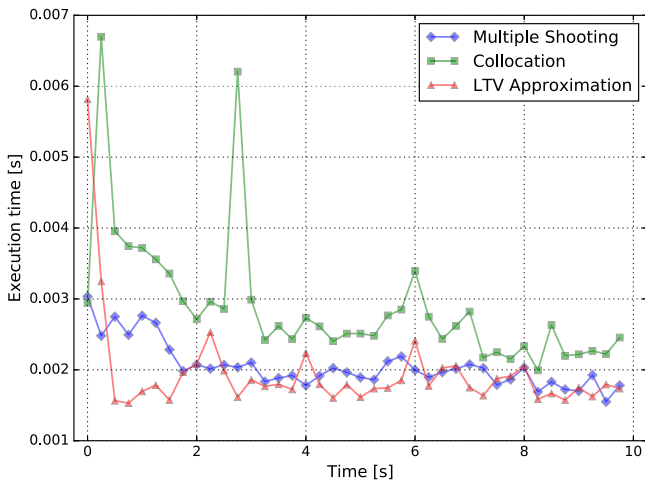Fig. 3. Control trajectory $u_1$ for the Van der Pol oscillator.



Fig. 4. Execution time of each of the methods for the Van der Pol oscillator.

| Parameter | Nominal value | Units |
|---|---|---|
| $F_0$ | 0.1 | $\mathrm{m}^3/\mathrm{min}$ |
| $T_0$ | 350 | K |
| $c_0$ | 1 | $\mathrm{kmol/m}^3$ |
| $r$ | 0.219 | m |
| $k_0$ | $7.2 \times 10^{10}$ | $\mathrm{min}^{-1}$ |
| $E/R$ | 8750 | K |
| $U$ | 54.94 | $\mathrm{kJ/(min\,m^2\,K)}$ |
| $\rho$ | 1000 | $\mathrm{kg/m}^3$ |
| $C_p$ | 0.239 | $\mathrm{kJ/(kg\,K)}$ |
| $\Delta H$ | $-5 \times 10^4$ | $\mathrm{kJ/kmol}$ |

TABLE I
PARAMETERS OF THE CSTR

where $x_1$ is the molar concentration of $A$, $x_2$ is the reaction temperature in (K) and $x_3$ is the level of the tank in meters, and the controls are $u_1$ and $u_2$, the outlet flow rate in (L/min) and coolant liquid temperature, respectively. The rest of the CSTR parameters can be seen in Table I. In this example, we will start from the initial condition $x_1 = 0.04389$, $x_2 = 243.3730$, $x_3 = 0.3295$ and we wish to drive the system to the setpoint $x_1^s = 0.8778$, $x_2^s = 324.5$, $x_3^s = 0.659$. The control input is subject to the following constraints

$$
\begin{aligned}
285 &\leq u_1 \leq 315, \\
0.085 &\leq u_2 \leq 0.115, \\
-3 &\leq \Delta u_1 \leq 3, \\
-0.03 &\leq \Delta u_2 \leq 0.03,
\end{aligned}
\tag{22}
$$

and the control setpoint is expected to be $u_1^s = 300$ and $u_2^s = 0.1$. The chosen sampling time is $T_s = 15$ minutes, the horizon length is $N = 15$, and the weight matrices are $Q = \mathrm{diag}([6.488\,69 \times 10^{-1}, 4.748\,40 \times 10^{-6}, 1.15132])$, $R = \mathrm{diag}([2.222\,22 \times 10^{-5}, 200])$ and $Q_N = 10Q$. The state trajectory obtained by each of the methods is shown in Figures 5, 6 and 7, where it can be seen again the similarities between each of the reviewed methods.

The obtained control inputs (shown in Figures 8 and 9) are almost equal for multiple shooting and collocation, whereas –once again– the obtained control inputs of the successive linearization method present some differences, mostly noticeable at the first iterations of $u_1$.

Finally, Figure 10 shows the execution time required to obtain the MPC solution at each discretization step. It can be seen that in this case, the collocation method is faster than its counterparts. This is mainly because the sparser NLP structure obtained by the use of collocation is better suited for the NLP solver that CasADi [14] uses (IPOPT) and because, due to the non linearities of this system, we had to obtain a full linerization over the complete control horizon at each time step for the the LTV model of the successive linearization method. The mean time was $0.01517$ seconds for multiple shooting, $0.01236$ seconds for direct collocation and $0.01896$ seconds for the successive linearizations method.

## V. CONCLUSION

This work shows that current state of the art NLP solvers can nowadays cope with real-time requirements, at least in small to medium scale non-linear systems. The successive linearization
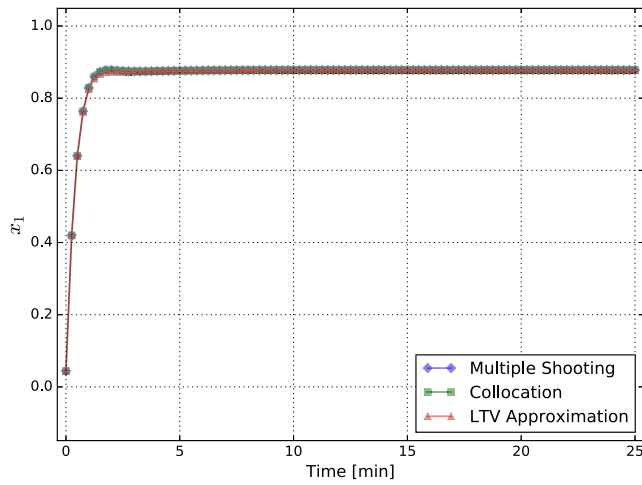
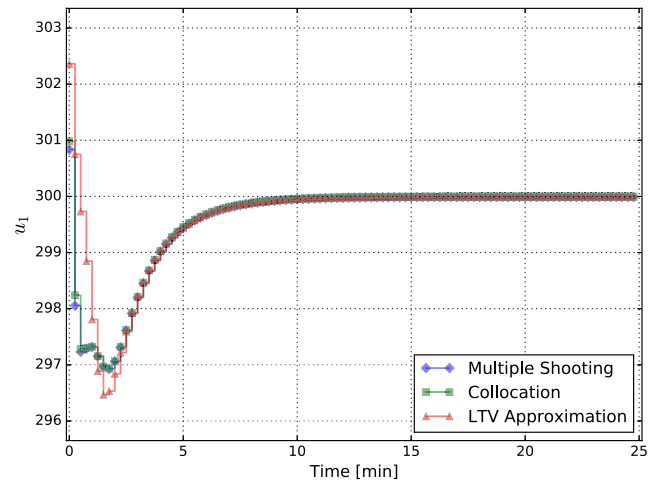Fig. 5. State trajectory $x_1$ for the CSTR.



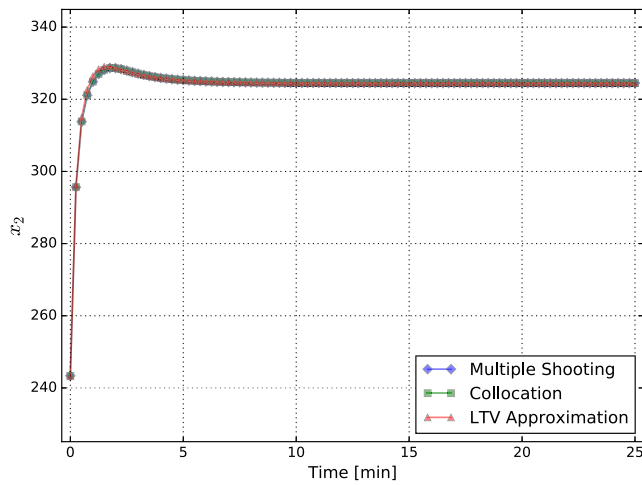Fig. 8. Control trajectory $u_1$ for the CSTR.



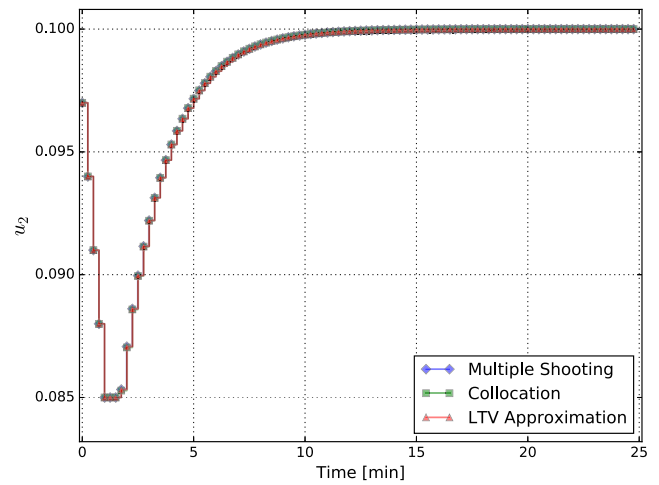Fig. 6. State trajectory $x_2$ for the CSTR.



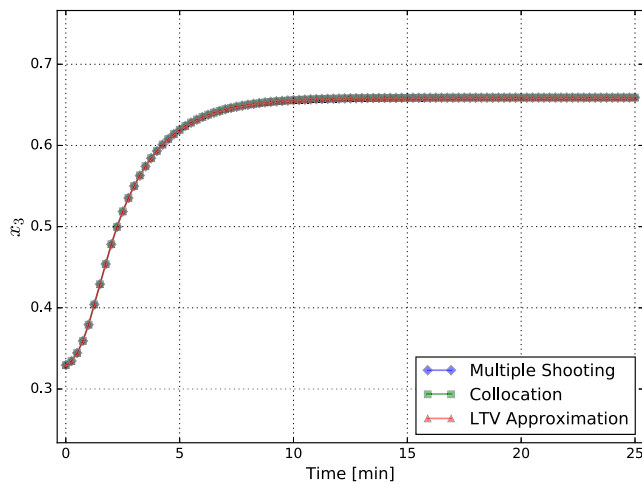Fig. 9. Control trajectory $u_2$ for the CSTR.



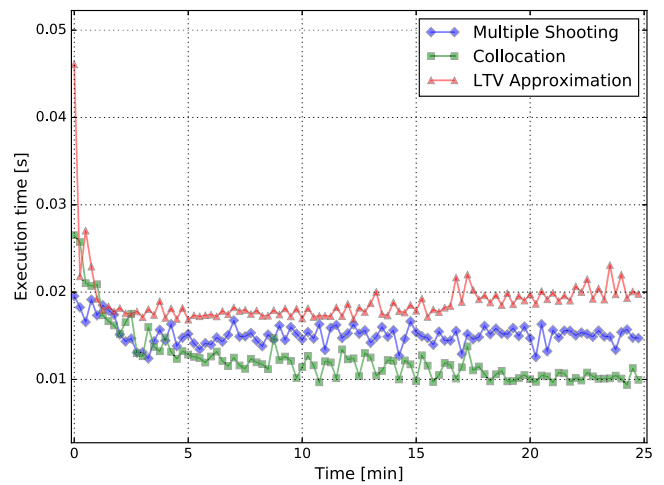Fig. 7. State trajectory $x_3$ for the CSTR.



Fig. 10. Execution time of each of the methods for the CSTR.

method performs as well and as fast as their counterparts and allows us to use analytical tools in order to analyze the stability, robustness and convergence of the controller. Besides, the possibility of using QP solvers guarantees that solution of the MPC control problem is obtained in each iteration.

Each of the examples was programmed in Python and was not optimized in any way. Since CasADi has C++ interfaces, we think that there is room for improvement. The source code for the examples is available at https://github.com/gmsanchez/mpc_comparison_rpic2017.

## REFERENCES

[1] A. Rao, "A survey of numerical methods for optimal control," *Advances in the Astronautical Sciences*, vol. 135, no. 1, pp. 497–528, 2009.

[2] M. Diehl, H. Bock, H. Diedam, and P.-B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," in *Fast motions in biomechanics and robotics*. Springer, 2006, pp. 65–93.

[3] C. Kirches, *Fast numerical methods for mixed-integer nonlinear model-predictive control*. Springer, 2011.

[4] H. Bock and K.-J. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems." Proceedings of the IFAC World Congress, 1984.

[5] D. Morrison, J. Riley, and J. Zancanaro, "Multiple shooting method for two-point boundary value problems," *Communications of the ACM*, vol. 5, no. 12, pp. 613–614, 1962.

[6] J. Ahlberg and T. Ito, "A collocation method for two-point boundary value problems," *Mathematics of Computation*, vol. 29, no. 131, pp. 761–776, 1975.

[7] F. Topputo and C. Zhang, "Survey of direct transcription for low-thrust space trajectory optimization with applications," in *Abstract and Applied Analysis*, vol. 2014. Hindawi Publishing Corporation, 2014.

[8] M. Murillo, A. Limache, P. Rojas Fredini, and L. Giovanini, "Generalized nonlinear optimal predictive control using iterative state-space trajectories: Applications to autonomous flight of uavs," *International Journal of Control, Automation and Systems*, vol. 13, no. 2, pp. 361–370, 2015.

[9] M. Murillo, G. Sánchez, and L. Giovanini, "Iterated non-linear model predictive control based on tubes and contractive constraints," {*ISA*} *Transactions*, vol. 62, pp. 120 – 128, 2016, sI: Control of Renewable Energy Systems.

[10] J. Maciejowski, *Predictive control: with constraints*. Pearson education, 2002.

[11] M. Diehl, H. Ferreau, and N. Haverbeke, "Efficient numerical methods for nonlinear MPC and moving horizon estimation," in *Nonlinear Model Predictive Control: Towards New Challenging Applications*, L. Magni, D. M. Raimondo, and F. Allgöwer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, vol. 384, ch. Efficient, pp. 391–417.

[12] R. DeCarlo, *Linear systems: A state variable approach with numerical implementation*. Prentice-Hall, Inc., 1989.

[13] J. Rawlings and D. Mayne, *Model predictive control: Theory and Design*. Nob Hill Publishing, LLC, 2009.

[14] J. Andersson, "A General-Purpose Software Framework for Dynamic Optimization," PhD thesis, Arenberg Doctoral School, KU Leuven, Department of Electrical Engineering (ESAT/SCD) and Optimization in Engineering Center, Kasteelpark Arenberg 10, 3001-Heverlee, Belgium, October 2013.

[15] M. Risbeck and J. Rawlings, "Mpctools: Nonlinear model predictive control tools for casadi (python interface)," https://bitbucket.org/rawlings-group/mpc-tools-casadi, 2015.