# Adaptive Model Predictive Control Based on Fixed Point Iteration

HAMZA KHAN
Óbuda University
Doctoral School of Applied
Informatics and Applied Mathematics
Bécsi út 96/B, H-1034 Budapest
HUNGARY
ameer.hamza22@gmail.com

JÓZSEF K. TAR
Óbuda University
Antal Bejczy Center for Intelligent Robotics
(ABC iRob)
Bécsi út 96/B, H-1034 Budapest
HUNGARY
tar.jozsef@nik.uni-obuda.hu

IMRE J. RUDAS
Óbuda University
Antal Bejczy Center for Intelligent Robotics
(ABC iRob)
Bécsi út 96/B, H-1034 Budapest
HUNGARY
rudas@uni-obuda.hu

GYÖRGY EIGNER
Óbuda University
Physiological Controls
Research Center
Bécsi út 96/B, H-1034 Budapest
HUNGARY
eigner.gyorgy@nik.uni-obuda.hu

*Abstract: Nonlinear Programming* provides a practical, reduced-complexity solution for the realization of *Model Predictive Controllers* in which a cost function representing contradictory limitations is minimized under the constraints that express the dynamical properties of the system under control. For nonlinear system models and non-quadratic cost functions the solution over a finite time-grid can be obtained by the use of *Lagrange's Reduced Gradient Method* that needs complicated numerical calculations. In this paper it is shown that under not too limiting conditions this procedure can be replaced by a simple fixed point seeking iteration based on Banach's Fixed Point Theorem. The simplicity of the proposed algorithm widens the possibility for the practical applications of the Receding Horizon Control method. The same algorithm is used for adaptively and precisely tracking the "optimized trajectory" that can be constructed by the use of a dynamic model of "overestimated" parameters in order to evade dynamical overloads in the control process. To illustrate the efficiency of the method the Receding Horizon Control of a strongly nonlinear, oscillating system, the van der Pol oscillator is presented. In the simulations three different parameter settings are considered: one of them produces the trajectory to be tracked, the second one is used for the optimization, and the third one serves as the model of the controlled system.

*Key–Words:* Nonlinear Programming, Model Predictive Control, Receding Horizon Controller, Adaptive Control, Fixed Point Transformation

## 1 Introduction

The classical realization of the *Model Predictive Controllers (MPC)* controllers [1, 2] applies the mathematical framework of *Optimal Control (OC)* in which a cost function constructed of the nonnegative contributions of normally contradictory restrictions is minimized under the constraints that represent the dynamic properties (i.e. the model) of the controlled system. It is widely used for the control of nonlinear plants in traffic control [3, 4], chemistry (e.g. [5, 6]), life sciences (e.g. [7]), web transport systems (e.g. [8]) etc.

The most general approach considers the problem in analogy with the minimization of the action functional in *Classical Mechanics*. The so obtained *Hamilton-Jacobi-Bellman Equations* are complicated, and the *Dynamic Programming (DP)* applied for their solution generally needs high computational power [9, 10]. A more practical approach tackles the problem by calculating the variables in the discrete points of a finite time-grid that is considered as a "horizon" (*Nonlinear Programming (NP)*). The *Receding Horizon Controllers (RHC)* [11, 12] work with finite horizon lengths and for the compensation of the effects of modeling errors the horizon is frequently redesigned from the actual state of the controlled system. The optimization under constraints happens by NP that implements *Lagrange's Reduced Gradient Method* [13]. In the special case of the LTI system models and quadratic cost functions the problem is considerably reduced: the so obtained *Linear Quadratic Regulator (LQR)* [14] technically can be realized over a finite horizon by solving the *Riccati Differential Equation* with a *terminal condition* for a matrix function that provides an inhomogeneous part for the equation of motion of the system state satisfying an *initial condition*. In more general cases such a clear separation of the variables cannot be realized and one has to work with *Time-dependent Riccati Equations*. (In the survey paper [15] a huge number of applications was referred to in connection with this problem.)

In a formally more general case the RG method can be numerically implemented. The *MS EXCEL's Solver Package* (provided by an external firm Frontline Systems, Inc.) in combination with a little programming efforts in *Visual Basic (VB)* in the background serves as an excellent solution if the size of the problem is not too big. The problem conveniently can be formulated by functional relationships between the contents of the various cells of the worksheets. For this purpose *User Defined Functions* can be created in VB. Then for the Solver a "model" can be specified by giving the cell that contains the cost to be minimized, the location of the independent variables and the constraints in the worksheets, and the parameter settings of this optimization package. The so defined "model" can be saved somewhere in one of the worksheets. Following that a small program can be written in VB that declares the model parameters as global variables, reads their actual values from the worksheets, loads the "model" for the Solver, and for the horizons under consideration cyclically a) fills in the cells with the data of the nominal trajectory to be tracked, the initial values of the variables to be optimized, and the control forces, b) calls the Solver with the options that it must stop optimization if the prescribed limits in the time or step numbers have been achieved, keeps the so obtained results, and c) writes the optimized results in certain cells of a worksheet.

In the first step, the Solver tries to find a common point on the constraint surfaces by the use of the Newton-Raphson method [16]. (In the 2010 version various initial points can be used for this purpose.) Following that it computes the *Reduced Gradient (RG)* by calculation the appropriate *Lagrange Multipliers*, and realizes little steps in the direction of the RG. The algorithm stops when the RG takes zero. At this point the constraints do not allow more improvement of the cost. The Solver package numerically calculates the gradient values, can automatically set the appropriate step lengths. The calculation of the Lagrange multipliers in principle needs the calculation of a quadratic matrix that generally may be singular or ill-conditioned, therefore somehow it also has to tackle these problems.

It is a reasonable expectation that this complicated procedure can be evaded in the control of a system class in which a) the cost functions contain separate differentiable contributions for penalizing the tracking error and the too big control effort, and b) the mathematical form of the system's model under control is ab ovo known. In this case the appropriate gradients can be analytically calculated, and the EXCEL – VB programming background does not offer further convenience, especially if the RG algorithm can be replaced by a simpler one. This program is briefed in the next section.

## 2 The Basics of NLP

Consider the numerical approximation of the problem as follows: determine a *dense enough* discrete time-grid as $\{t_0,\, t_1 = t_0 + \Delta t,\, \ldots,\, t_{n+1} = t_n + \Delta t,\, \ldots,\, t_N\}$ in which $t_0$ and $t_N$ correspond to the *initial* and the *final time* of the considered motion. Let the function $\dot{x} = f(x, u)$ describe the equation of motion of the controlled system in which $x \in \mathbb{R}^n$ denotes the *state variable*, and $u \in \mathbb{R}^m$ is the *control signal* ($n, m \in \mathbb{N}$). The *nominal trajectory to be tracked* in the given time-grid takes the values $x^{Nom}(t_i) \equiv x_i^{Nom}$. In the control task this nominal trajectory cannot be exactly realized because *various restrictions* can be prescribed be the use of a *Cost Function* $J(x, u)$ in each point of the grid. The function $J(x, u) \geq 0$ may express various, *often contradictory requirements*. It can be constructed as the sum of various non-negative terms that *expediently are differentiable functions of the state variable and the control signal*. The use of large control signals can be "prohibited" in the cost function, too. For the *last term at* $t_N$ an extra *terminal condition can be prescribed* that depends only on $x_N$. In the *Optimal Control Approach* the above sum *has to be minimized*:

$$\sum_{i=0}^{N-1} J(x_i, u_i) + F(x_N) \ , \tag{1}$$

in which the last term $F(x_N)$ gives an "extra weight" to the last point of the trajectory. However, (1) cannot be *arbitrarily minimized*. The *dynamics of the system* expressed by the equation of state propagation *has to be taken into account as a constraint* in the minimization. This constraint can be processed by the use of the *Lagrange Multipliers* in the following manner: The time-derivative $\dot{x}$ has an expression from the *state propagation equation*, and the *numerical estimation* as $\frac{x_{i+1} - x_i}{\Delta t} \approx f(x_i, u_i)$. On this basis an "auxiliary function" can be introduced in which the Lagrange multipliers in the great majority of applications have clear physical meaning (e.g. [17]):

$$\Phi = \sum_{i=0}^{N-1} \left[ J(x_i, u_i) + \lambda_i^T \left( \frac{x_{i+1} - x_i}{\Delta t} - f(x_i, u_i) \right) \right] + F(x_N) \tag{2a}$$

in which $\Phi = \Phi(\{x\}, \{u\}, \{\lambda\})$. The *independent variables of the problem* are $\{x_1, \ldots, x_N\}$, $\{u_0 \ldots, u_{N-1}\}$, and $\{\lambda_0, \ldots, \lambda_{N-1} \in \mathbb{R}^n\}$ are the Lagrange multipliers. The auxiliary function $\Phi$ evidently is unbounded but it has *local saddle points* when its partial derivatives by its all variables are zeros. For $k \in \{1, 2, \ldots, N-1\}$ we get:

$$\frac{\partial \Phi}{\partial x_k} = \frac{\partial J(x_k, u_k)}{\partial x_k} + \frac{\lambda_{k-1}}{\Delta t} - \frac{\lambda_k}{\Delta t} - \frac{\lambda_k^T \partial f(x_k, u_k)}{\partial x_k} = 0, \tag{3}$$

for $k = N$:

$$\frac{\partial \Phi}{\partial x_N} = \frac{\lambda_{N-1}}{\Delta t} + \frac{\partial F(x_N)}{\partial x_N} = 0 \ , \tag{4}$$

for $l \in \{0, 1, 2, \ldots, N-1\}$:

$$\frac{\partial \Phi}{\partial u_l} = \frac{\partial J(x_l, u_l)}{\partial u_l} - \frac{\lambda_l^T \partial f(x_l, u_l)}{\partial u_l} = 0, \tag{5}$$

and for $j \in \{0, 1, \ldots, N-1\}$

$$\frac{\partial \Phi}{\partial \lambda_j} = \frac{x_{j+1} - x_j}{\Delta t} - f(x_j, u_j) = 0 \ , \tag{6}$$

for the given initial value $x_0$. Evidently (3) states that *the reduced gradient is zero*, that is the set of the $\nabla\Phi = \mathbf{0}$ points contains the points where the above detailed algorithm stops, (4) is related to the *terminal condition*, (6) means that the solution must be on the constraints' common hypersurface, and (5) expresses the *condition for the control forces*. It worths noting that in general, if they exist, the local maximums of the cost function also satisfy the $\nabla\Phi = \mathbf{0}$ condition. However, in many practical applications (e.g. in Thermodynamics) it corresponds to the local minimum. The traditional approach consider these equations as starting point for developing the LQR controller for special cost functions and model structures.

Instead tracking the traditional route it is expedient to observe that if in the variable $X \in \mathbb{R}^K$ all the independent variables of $\Phi$ are collected, the function $\Psi(X) \stackrel{def}{=} \nabla\Phi(X) : \mathbb{R}^K \mapsto \mathbb{R}^K$ is a $K (\in \mathbb{N})$ dimensional vector function, and our goal is to drive the value of this function to zero from an ini0.1tial point. This task evidently is in strict analogy with the *Inverse Kinematic Task of Robots* in which the *Cartesian Workshop Coordinates* $x(q) \in \mathbb{R}^l$ as the functions of the *Joint Coordinates* $q \in \mathbb{R}^s$, $l, s \in \mathbb{N}$, and for a redundant robot $s > l$ describe the *Forward Kinematics* of the robot arm.

# 3 Analogy with The Solution of The Inverse Kinematic Task of Robots

The task is to find $q$ for a given $x^{Des}$ "desired position" has (normally ambiguous) closed form solution only for special arm constructions, e.g. in the case of a PUMA-type robot [18]. As a general possibility, the differential solution based on the use of the Jacobian $\frac{\partial x}{\partial q}$ in a function of a *scalar variable* $\xi \in \mathbb{R}$ as $x(\xi) = x(q(\xi))$ is considered in the equation

$$\frac{\mathrm{d}x_j}{\mathrm{d}\xi} = \sum_i \frac{\partial x_j}{\partial q_i} \frac{\mathrm{d}q_i}{\mathrm{d}\xi} \equiv \sum_i J_{ji} \frac{\mathrm{d}q_i}{\mathrm{d}\xi} \ , \tag{7}$$

where the *initial conditions* as $x(\xi_{ini}) = x_{ini}$ and $q(\xi_{ini})$ are known. The traditional solutions contain some *generalized inverse* as e.g. the *Moore-Penrose Pseudoinverse* [19, 20] that is singular in, and ill-conditioned in the vicinity of the kinematic singularities of the robot arm. The general problem is that such a solution generates huge joint coordinate time-derivatives therefore it is expedient to "tame" the original task to evade the numerical inconveniences, as e.g. in the method of *Damped Least Squares* [21]. As an alternative of the traditional approach in [22] the original task was transformed into a fixed point problem that subsequently was solved by simple iteration. Its special advantage is that it automatically shows stable solution in and in the vicinity of the kinematic singularities without the use of any "complementary trick", and automatically selects one of the ambiguous solutions. On this reason the use of this algorithm for driving $\nabla\Phi$ to zero in the novel RHC controller was suggested. The essence of the method is briefed below.

The idea of transforming our task into a fixed point problem and solving it via iterations, has very early roots in the $17^{th}$ century as the Newton-Raphson Algorithm, that has many applications even in our days (e.g. [23]). In 1922 Banach extended this way of thinking to quite wide problem classes [24]. According to his theorem, *in a linear, complete metric space (i.e. the "Banach Space") the sequence created by the contractive mapping* $\psi : \mathbb{R}^m \mapsto \mathbb{R}^m$, $m \in \mathbb{N}$ *as* $x_{s+1} = \psi(x_s)$ *is a Cauchy Sequence that converges to the fixed point of* $\psi$ *defined as* $\psi(x_\star) = x_\star$. (A map is contractive if $\exists 0 \leq H < 1$ so that $\forall x, y$ elements of the space $\|\psi(x) - \psi(y)\| \leq H\|x - y\|$.) In [25] the following transformation was used for this purpose: a real differentiable function $\varphi(\xi) : \mathbb{R} \mapsto \mathbb{R}$ was taken with an *attractive fixed point* $\varphi(\xi_\star) = \xi_\star$. It was used for the generation of a sequence of iterative signals as

$$q(i+1) = \left[\varphi(A\|x(q(i)) - x^{Des}\| + \xi_\star) - \xi_\star\right] \cdot \cdot \frac{x(q(i)) - x^{Des}}{\|x(q(i)) - x^{Des}\|} + q(i) \ , \tag{8a}$$

in which the Frobenius norm was used. In (8a) $A \in \mathbb{R}$ is an *adaptive parameter*. For $q(k) = q_\star$ that provides $x(q_\star) = x^{Des}$, (8a) yields that $q(k+1) = q(k)$, that means that $q_\star$, i.e. the solution of our task, is the fixed point of this function. The convergence of this sequence was investigated in [26] by making the first order Taylor series approximation of $\varphi(\xi)$ in the vicinity of $\xi_\star$ and that of $x(q)$ around $q_\star$. It was found that if the real part of each eigenvalue of the Jacobian $\frac{\partial x}{\partial q}$ *simultaneously* positive or negative, an appropriate parameter $A$ can be so chosen that it guarantees the convergence. This result for the *redundant robot arms* of non-quadratic Jacobians in [22] was so applied that instead of the original problem $x^{Des} = x(q)$ the modified one $J^T(q)x^{Des} = J^T(q)x(q)$ was solved. By the Taylor series approximation of $x(q)$ around $q_\star$ it can be shown that the convergence will be determined by the positive semidefinite matrix $J^T(q)J(q)$ that has non-negative eigenvalue. (The zeros eigenvalues cause "stagnation" instead of infinite velocities in the singularities.) For adaptively tracking the "optimized trajectory" a similar transformation into a fixed point problem was applied as it is briefed in the sequel.

# 4 Fixed Point Transformation in Adaptive Control

The idea of transforming an adaptive control task into a fixed point problem was risen in [27]. According to Fig. 1 it can be shortly expounded for the digital control of a second order system as follows: by applying an appropriate tracking error feedback in the "Kinematic Block" to calculate the "Desired Tracking Error Damping" [in the case of a PD-type controller it is $\ddot{q}^{Des}(t) = \ddot{q}^N(t) + 2\Lambda(\dot{q}^N(t) - \dot{q}(t)) + \Lambda^2(q^N(t) - q(t))$], for a constant $\Lambda > 0$ time-exponent by the use of this signal the elements of the sequence of the "Deformed Control Signals" $\ddot{q}^{Def}(t)$ are created by the function in (8a); this deformed signal is used as the input of the available "Approximate Model" of the controlled system for the calculation of the control force $Q(t)$ that is exerted on the actually controlled system that generates the realized response $\ddot{q}(t)$. (The symbolic integrations at the bottom of the figure are done by the dynamics of the controlled system in a real control situation, or, in the case of a simulation study, they have to be implemented numerically.) After converging to the fixed point the kinematically prescribed trajectory tracking error damping will be precisely realized. In [29] the same control was implemented in an EXCEL-Solver-Visual Basic environment. In the present research the same structure is used in the proposed adaptive RHC controller.

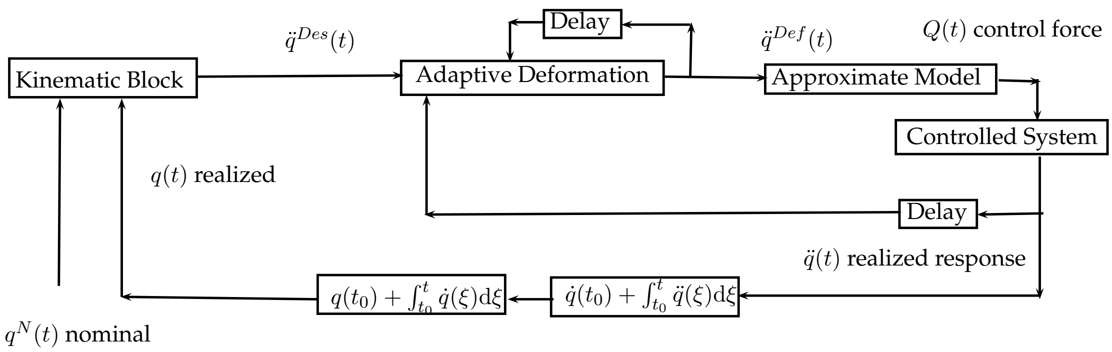


Fig. 1. Schematic structure of the "Fixed Point Transformation-based Adaptive Controller" taken from [28]

# 5 Simulation Investigations

The investigated strongly nonlinear 2nd order physical system was the van der Pol oscillator invented in 1927 [30]. Its equation of motion is given in (9)

$$\ddot{x} = \frac{-kx - b(x^2 - d)\dot{x} - cx^3 + eu}{m} \equiv f(x, \dot{x}, u) \ , \tag{9}$$

in which $u$ is the control force, $x$ and $\dot{x}$ are the state variables. Parameters $k > 0$ and $c > 0$ describe a spring that "strengthens" with increasing extension $x$, parameter $b > 0$ describes viscous damping if $x^2 > d$, and excitation for $x^2 < d$. Due to it the state $x \equiv 0$ is an unstable equilibrium point: the smallest disturbance brings about excitation and drives the system into nonlinear oscillation that is bounded by the dissipative nature of the term $-b(x^2 - d)\dot{x}$ for $x^2 > d$. Parameter $e$ describes the system's sensitivity for the control force $u$. The appropriate model parameters are given in Table 1.

Table 1: The applied model parameters

| Param. | Exact | Approx. | Traj. generator |
|---|---|---|---|
| $m$ | 1.0 | 2.0 | 3.0 |
| $k$ | 100.0 | 130.0 | 140.0 |
| $b$ | 1.2 | 1.5 | 2.0 |
| $d$ | 1.0 | 1.3 | 3.0 |
| $c$ | 0.5 | 0.8 | 0.6 |
| $e$ | 2.0 | 1.5 | 1.0 |

For the *dynamic control* $\Lambda = 2.0\,s^{-1}$ was used, parameter $A$ in (8a) was $A_{dc} = -0.5$. For the purpose of the optimization various values were studied for $A_{opt}$. The time resolution of the grid was $\Delta t = 10^{-3}\,s$, the horizons consisted of $G = 10$ grid points, that, in the case of a 2nd order system corresponds to 8 independent state variables (the initial conditions correspond to two independent grid points at the beginning of the horizon), and on the same reason we have 8 independent Lagrange multipliers and 8 independent control signals that determine the system's motion over the grid. No special terminal cost was applied, and the "auxiliary function" had the structure as follows:

$$\Psi = \sum_{j=3}^{G} \left| \frac{x_j^N - x_j}{A_x} \right|_x^{\alpha} + B_u \sum_{j=1}^{G-2} \left| \frac{u_j}{A_u} \right|_u^{\alpha} + \sum_{j=1}^{G-2} \lambda_j \left[ x_{j+2} - 2x_{j+1} + x_j \right] + \sum_{j=1}^{G-2} \lambda_j \left[ -\Delta t^2 f(x_j, \dot{x}_j, u_j) \right] \quad , \tag{10a}$$

in which for $\alpha_x > 1$ and $\alpha_u > 1$ the tracking error and the control force are well tolerated if $|x^N - x| < A_x$ and $|u| < A_u$, respectively, but they are strongly penalized over these limits. In (10a) the term $f(x_j, \dot{x}_j, u_j)$ can be approximated as $f(x_j, \frac{x_{j+1}-x_j}{\Delta t}, u_j)$, and the terms in $\nabla\Psi$ and the Jacobian of the problem can be calculated in closed form for optimization. (For sparing room these terms are not given here.) In (8a) the function $\varphi(x) = \frac{x}{2} + 0.3$ was in use.

In the investigations the *trajectory generator* was excited with a constant force $F_\gamma = 300.0\,N$ that makes it settling down at the damped region. The control parameters were set as follows: $A_x = 0.5\,m$, $\alpha_x = 6.1$, $A_u = 100.0\,N$, $\alpha_u = 8.0$, $B_u = 100$. The RHC algorithm contained 100 internal iterations. The $A_{opt} = -1 \times 10^{-5}$ setting represents too slow iteration. Figure 2 reveals that the "optimized" trajectory is very far from the "nominal" one, and that the internal iteration did not result in good improvement of $\nabla\Psi$. The counterpart of Fig. 2 for $A_{opt} = -1 \times 10^{-2}$ is displayed in Fig. 3. It reveals that the tracking error is practically kept under $0.5\,m$ that is compatible with the setting $A_x = 0.5\,m$, $\alpha_x = 6.1$. It is also clear that the $\|\nabla\Psi\|$ went down from the value 15 to $\approx 0.13$, i.e. the inner iteration was really responsible for driving $\nabla\Psi$ towards zero.

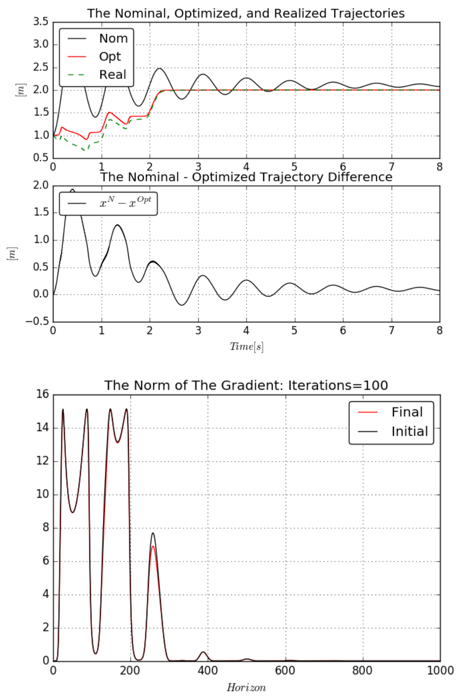


Figure 2: Trajectory tracking for too small adaptive parameter in the optimization ($A_{opt} = -1 \times 10^{-5}$)

Regarding the adaptive tracking of the optimized trajectory it can be seen that in both cases the adaptivity that was switched on in the beginning of the 2nd horizon, produced good results. Figure 5 explains its reason: the "desired" 2nd time-derivatives are well approximated by the realized ones while they considerably differ from the "adaptively deformed" values. The significance of the dynamic adaptivity in trajectory tracking is also substantiated by Fig. 4, that describes the case in which this dynamic adaptivity was switched off: the realized trajectory even does not approach the optimized one.

Figure 6 explains the reason for the remnant part of $\nabla\Psi$: the minimal eigenvalue is 0, therefore the theoretically expected occurrence of "stagnation" was substantiated by the computations.

Figure 7 reveals great fluctuations in the Lagrange multipliers and the control forces. It worths noting that in [29] similar fluctuations were observed in con-

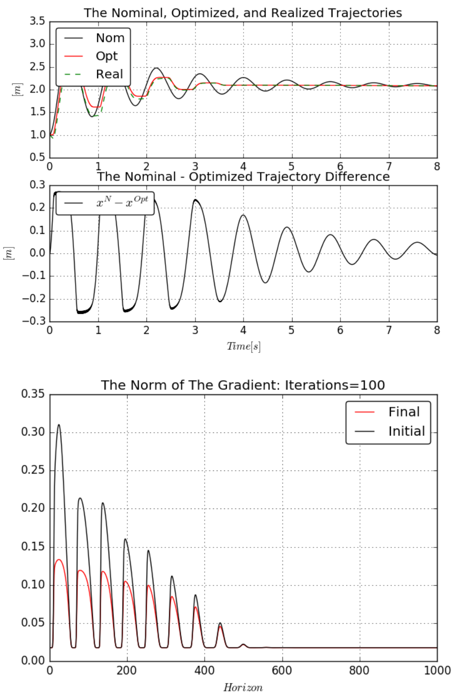


Figure 3: Trajectory tracking for appropriate adaptive parameter in the optimization ($A_{opt} = -1 \times 10^{-2}$)

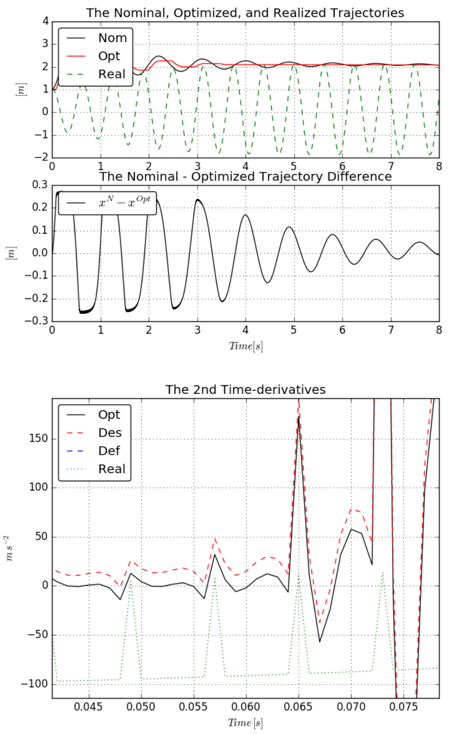


Figure 4: Trajectory tracking without dynamic adaptivity for appropriate adaptive parameter in the optimization ($A_{opt} = -1 \times 10^{-2}$)

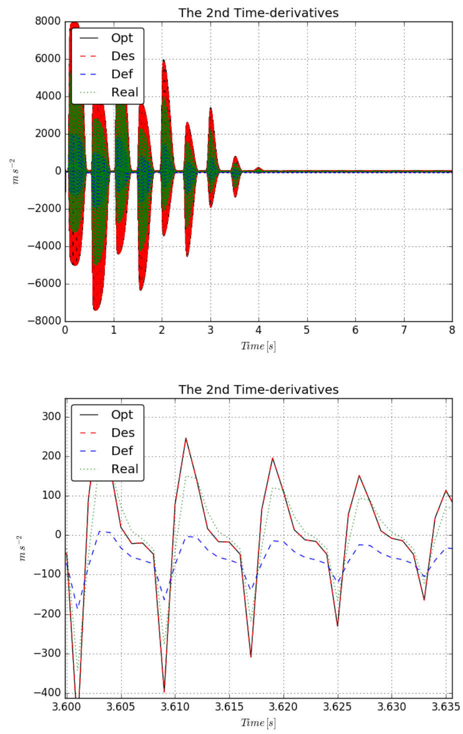


Figure 5: The second time-derivatives in the adaptive dynamic tracking of the optimized trajectory for appropriate adaptive parameter in the optimization ($A_{opt} = -1 \times 10^{-2}$)

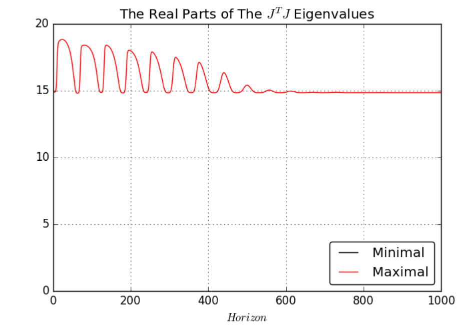


Figure 6: The maximal and minimal eigenvalues of $J^T J$ in the internal iterations for appropriate adaptive parameter in the optimization ($A_{opt} = -1 \times 10^{-2}$)

nection with a similar problem solved by the use of the EXCEL–Solver–Visual Basic apparatus.

# 6 Conclusion

In this paper an attempt is reported for replacing Lagrange's Reduced Gradient Method's EXCEL-Solver-Visual Basic-based implementation with a simple fixed point transformation-based adaptive solution

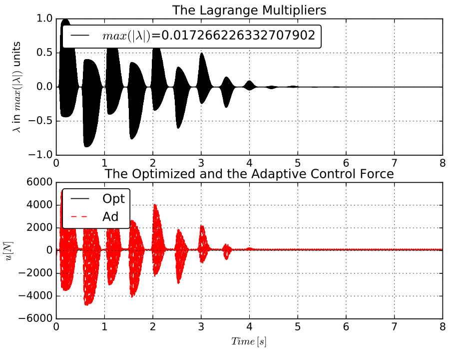


Figure 7: The Lagrange multipliers and the control signals for appropriate adaptive parameter in the optimization ($A_{opt} = -1 \times 10^{-2}$)

that easily can be implemented in arbitrary software environment for a wide class of problem classes in which the gradient of the "auxiliary function" as well as the gradient of this gradient can be determined in closed form formulation. The same type of fixed point transformation was applied for driving the gradient of the auxiliary function and adaptively tracking of the optimized trajectory by the actual system.

The applicability of the method was illustrated by presenting an example realizing the adaptive RHC control of a van der Pol oscillator. In this task three different parameter settings were applied: one of them was used for the generation of the nominal trajectory, another settings was used in the constraint terms of the optimization, and the third one represented the actual system under control. The simulations were made by a simple sequential code written in Julia language.

It definitely can be stated that the theoretical expectations were verified by the simulations.

Regarding our future plans, we wish to apply this approach for tackling adaptive RHC control realizations in biomedical applications.

**Acknowledgements:** The research was supported by the Doctoral School of Applied Informatics and Applied Mathematics of Óbuda University. Gy. Eigner was supported by the \'UNKP-17-4/I. New National Excellence Program of the Ministry of Human Capacities.

*References:*

[1] L. Grüne and J. Pannek, Nonlinear Model Predictive Control, *Springer* 2011.

[2] A. Grancharova and T.A. Johansen, Explicit Nonlinear Model Predictive Control, *Springer* 2012.

[3] T. Tettamanti and I. Varga, Distributed Traffic Control System based on Model Predictive Control, *Periodica Polytechnica ser. Civil Eng.* 54(1), 2010, pp. 3–9.

[4] S. Lin, B. De Schutter, Y. Xi and J. Hellendoorn, Fast model predictive control for urban road networks via MILP, *IEEE Transactions on Intelligent Transportation Systems* 12, 2011, pp. 846–856.

[5] J.W. Eaton and J.B. Rawlings, Feedback control of chemical processes using on-line optimization techniques, *Computers & Chem. Eng.* 14, 1990, pp. 469–479.

[6] N. Moldoványi, Model Predictive Control of Crystallisers (PhD Thesis), *Department of Process Engineering, University of Pannonia, Veszprém, Hungary* 2012.

[7] I. Naşcu, R. Oberdieck and E.N. Pistikopoulos, Offset-free explicit hybrid model predictive control of intravenous anaesthesia, *In: Proc. of the 2015 IEEE International Conference on Systems, Man, and Cybernetics, October 9-13, 2015, Hong Kong* 2015, pp. 2475–2480.

[8] N. Muthukumar, Seshadhri Srinivasan, K. Ramkumar, K. Kannan and V.E. Balas, Adaptive Model Predictive Controller for Web Transport Systems, *Acta Polytechnica Hungarica* 13(3), 2016, pp. 181–194.

[9] R.E. Bellman, Dynamic Programming and a new formalism in the calculus of variations, *Proc. Natl. Acad. Sci.* 40(4), 1954, pp. 231–235.

[10] R.E. Bellman, Dynamic Programming, *Princeton Univ. Press, Princeton, N. J.* 1957.

[11] J. Richalet, A. Rault, J.L. Testud and J. Papon, Model predictive heuristic control: Applications to industrial processes, *Automatica* 14(5), 1978, pp. 413–428.

[12] A. Jadbabaie, Receding Horizon Control of Nonlinear Systems: A Control Lyapunov Function Approach (PhD Thesis), *California Institute of Technology, Pasadena, California, USA* 2000.

[13] J.L. Lagrange, J.P.M. Binet and J.G. Garnier, Mécanique analytique (Eds. J.P.M. Binet and J.G. Garnier), *Ve Courcier, Paris* 1811.

[14] R.E. Kalman, Contribution to the Theory of Optimal Control, *Boletin Sociedad Matematica Mexicana* 5(1), 1960, pp. 102–119.

[15] Tayfun Çimen, State-Dependent Riccati Equation in Nonlinear Optimal Control Synthesis, *In the Proc. of the Special International Conference on Complex Systems:*

*Synergy of Control, Communications and Computing - COSY 2011, Hotel Metropol Resort,* *Ohrid, Republic of Macedonia, September, 16 – 20, 2011* 2011, pp. 321–332.

[16] Tjalling J. Ypma, Historical development of the Newton-Raphson method, *SIAM Review* 37(4), 1995, pp. 531–551.

[17] H. Karabulut, Physical meaning of Lagrange multipliers, *European Journal of Physics (physics.ed-ph); General Physics (physics.gen-ph)* 27, 2007, pp. 709–718.

[18] C.S.G. Lee and M. Ziegler, RSD-TR-1-83 A geometric approach is solving the inverse kinematics of PUMA robots, *The University of Michigan, Ann Arbor, Michigan 48109-1109* 1983.

[19] E.H. Moore, On the reciprocal of the general algebraDinevaPhD:2016ic matrix, *Bulletin of the American Mathematical Society* 26(9), 1920, pp. 394–395.

[20] R. Penrose, A generalized inverse for matrices, *Proceedings of the Cambridge Philosophical Society* 51, 1955, pp. 406–413.

[21] S. Chiaverini, O. Egeland and R.K. Kanestrom, Achieving User-Defined Accuracy with Damped Least Squares Inverse Kinematics, *In the Proc. of the 1991 IEEE International Conference on Robotics and Automation, June 19-22, 1991, Pisa, Italy* 1991.

[22] B. Csanádi, J.K. Tar and J.F. Bitó, Matrix Inversion-free Quasi-differential Approach in Solving the Inverse Kinematic Task, *In Proc. of the* $17^{th}$ *IEEE International Symposium on Computational Intelligence and Informatics (CINTI 2016), 17-19 November 2016, Budapest, Hungary* 2016, pp. 61–66.

[23] C.T. Kelley, Solving Nonlinear Equations with Newton's Method, no 1 in Fundamentals of Algorithms, *SIAM* 2003.

[24] S. Banach, Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales, *Fund. Math.* 3, 1922, pp. 133–181.

[25] A. Dineva, J.K. Tar, A. Várkonyi-Kóczy and V. Piuri, Adaptive Control of Underactuated Mechanical Systems Using Improved "Sigmoid Generated Fixed Point Transformation" and Scheduling Strategy, *In Proc. of the* $14^{th}$ *IEEE International Symposium on Applied Machine Intelligence and Informatics, January 21-23, 2016, Herl'any, Slovakia* 2016, pp. 193–197.

[26] A. Dineva, Non-conventional Data Representation and Control (PhD Thesis), *Óbuda University, Budapest, Hungary* 2016.

[27] J.K. Tar, J.F. Bitó, L. Nádai and J.A. Tenreiro Machado, Robust Fixed Point Transformations in Adaptive Control Using Local Basin of Attraction, *Acta Polytechnica Hungarica* 6(1), 2009, pp. 21–37.

[28] H. Redjimi and J.K. Tar, On the Effects of Time-Delay on Precision Degradation in Fixed Point Transformation-based Adaptive Control, *In the Proc. of the 2017 IEEE 30th Jubilee Neumann Colloquium, November 24-25, 2017, Budapest, Hungary* 2017, pp. 125–130.

[29] H. Khan, Á. Szeghegyi and J.K. Tar, Fixed Point Transformation-based Adaptive Optimal Control Using NLP, *In the Proc. of the 2017 IEEE 30th Jubilee Neumann Colloquium, November 24-25, 2017, Budapest, Hungary* 2017, pp. 35–40.

[30] B. Van der Pol, Forced oscillations in a circuit with non-linear resistance (reception with reactive triode), *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 7(3), 1927, pp. 65–80.