Towards Dynamic Optimization with Partially Updated Sensitivities ⋆

Niels van Duijkeren * **Goele Pipeleers** * **Jan Swevers** *
Moritz Diehl **

* *Dept. of Mechanical Engineering, KU Leuven, Belgium.*
(e-mail: {`niels.vanduijkeren`, `goele.pipeleers`, `jan.swevers`}`@kuleuven.be`*).*
** *Dept. of Microsystems Engineering & Dept. of Mathematics, University of Freiburg, Germany.*
(e-mail: `moritz.diehl@imtek.uni-freiburg.de`*).*

Abstract: In nonlinear model predictive control (NMPC), a control task is approached by repeatedly solving an optimal control problem (OCP) over a receding horizon. Popularly, the OCP is approximated with a finite-dimensional nonlinear program (NLP). Since computing the solution of an NLP can be a complex and time-consuming task, tailored optimization algorithms have emerged to (approximately) solve the NLPs. Most methods rely on repeatedly solving a quadratic approximation of the NLP. Since computing this approximation is generally computationally demanding, it can form a bottelenck in obtaining a real-time applicable control law. This paper proposes DOPUS, a novel update scheme for the quadratic approximation of the NLP. DOPUS exploits the structure of the NLP and the repeated nature at which it is solved, to reduce the number of computations at the price of a small reduction of the convergence speed. Foreseen application areas include (economic) NMPC for fast-changing control tasks and fast time-varying systems. The convergence properties of DOPUS are studied and the performance is illustrated in a numerical case study considering a control task for a planar robot arm.

Keywords: Nonlinear predictive control; real-time optimization.

1. INTRODUCTION

Nonlinear model predictive control (NMPC) is an optimization based feedback control strategy. Among the advantages of NMPC are the capability to explicitly handle equality and inequality constraints, its support for nonlinear (possibly implicit) system dynamics and the flexibility in formulating the control objective. A continuous-time optimal control problem (OCP) collects the objective, the process model and the constraints. From this OCP a nonlinear program (NLP) can be derived using for instance the direct multiple-shooting method (Bock and Plitt, 1984). For a successful implementation of NMPC it is crucial that the control law can be computed in time. This depends on the required control rate, the complexity of the control law and the hardware available to perform the necessary computations. Because solving the NLP at every time instant is often too time consuming, various approximate solution schemes have been developed. A family of such schemes are the real-time iteration (RTI) algorithms (see e.g. Diehl et al., 2005). RTI schemes exploit the repetitive nature at which the control law is computed in NMPC. The condition to find a (local) optimum every control instance is relaxed, and considered something to be achieved in the long run as the loop is closed using suboptimal intermediate solutions. More specifically, the NLP is solved using a full-step sequential quadratic programming (SQP) approach which performs one iteration of the SQP scheme per control decision. An important requirement for these suboptimal solution strategies is fast convergence of the NLP over the iterations, and fast recovery from perturbations to the solution induced by e.g., disturbances or a change of the NLP parameters. SQP methods can in principle approach quadratic convergence rates. Though, this requires the availability of accurate sensitivities. For tracking-NMPC with a least-squares objective, the generalized Gauss-Newton method has proven to be a cheap, yet effective Hessian approximation. And tailored solvers have been developed for the purpose of generating sensitivities of the (implicit) system dynamics efficiently (Quirynen et al., 2014).

For certain applications, doing a quadratic approximation of the NLP remains computationally very demanding. Examples include the control of systems governed by large scale DAEs, systems controlled at very high sampling rates, and the application of economic-NMPC. Several solutions have been proposed to address this problem, among which are variants of the so-called multi-level iteration scheme (cf. Bock et al., 2007; Albersmeyer et al., 2009;

⋆ NvD is fellow of the TEMPO Initial Training Network. The work is supported by the EU via ERC-HIGHWIND (259 166), ITN-TEMPO (607 957), and ITN-AWESCO (642 682). The support from Flanders Make, DFG via the project "Numerische Methoden zur optimierungsbasierten Regelung zyklischer Prozesse, the KU Leuven-BOF PFV/10/002 Centre of Excellence: Optimization in Engineering (OPTEC), and project G0A6917N of the Research Foundation - Flanders (FWO-Flanders), is gratefully acknowledged.

Kirches et al., 2010, 2012) and the mixed-level iteration scheme (Frasch et al., 2012). Due to their relevance to this paper, both methods are introduced in more detail in Section 1.2.

This paper introduces DOPUS, which stands for Dynamic Optimization with Partially Updated Sensitivities. DOPUS belongs to the family of real-time iteration schemes and constitutes an alternative to the multi/mixed-level iteration schemes to save time in generating sensitivities. It aims at being the preferred choice for NMPC of fast time-varying systems or fast-changing control tasks. The main idea of DOPUS is to exploit the solution approach of RTI schemes in order to reuse the most accurate parts of the quadratic approximation. One of the key advantages of DOPUS is that the user can trade-off the number of iterations required for convergence and the computation time per iteration.

The remainder of the paper is organized as follows. First, the problem setting is introduced and the multi-level and mixed-level iteration schemes are briefly summarized. Subsequently, the DOPUS algorithm is described and first insights into its theoretical convergence properties are discussed by analyzing a related unconstrained optimization problem. A prototype implementation of DOPUS is applied in a numerical case-study: an application of economic NMPC to a robotic manipulator.

1.1 Problem Formulation

Consider the implicit system dynamics

$$0 = f\left(t, \dot{x}, x, u\right), \tag{1}$$

with the state vector x and the control input vector u. NMPC is a control strategy which computes the desired input u for (1) by solving an optimal control problem at every discrete time instant. Using a direct multiple shooting strategy an infinite dimensional OCP is transcribed into a structured NLP of the following form:

$$\begin{aligned}
\underset{w}{\text{minimize}} \quad & F(w) := \sum_{k=0}^{N-1} \ell_k\left(x_k, u_k\right) + E\left(x_N\right) && \text{(2a)}\\
\text{subject to} \quad & 0 = x_{k+1} - \phi_k\left(x_k, u_k\right), k = 0 \dots N-1 && \text{(2b)}\\
& 0 \geq h_k\left(x_k, u_k\right), \qquad k = 0 \dots N-1 && \text{(2c)}\\
& 0 = x_0 - \hat{x}_0 && \text{(2d)}\\
& 0 \geq r\left(x_N\right). && \text{(2e)}
\end{aligned}$$

Let us define some shorthand notations. The concatenation $[x_k^\intercal \; u_k^\intercal]^\intercal =: w_k$ for $k = 0, \dots, N-1$. The collection of all optimization variables $w := \left[w_0^\intercal \; \dots \; w_{N-1}^\intercal \; x_N^\intercal\right]^\intercal$. Let us stack the constraints in vectors:

$$G(w, \hat{x}_0) = \begin{bmatrix} x_0 - \hat{x}_0 \\ x_1 - \phi_0(w_0) \\ \vdots \\ x_N - \phi_{N-1}(w_{N-1}) \end{bmatrix}, \; H(w) = \begin{bmatrix} h_0(w_0) \\ \vdots \\ h_{N-1}(w_{N-1}) \\ r(x_N) \end{bmatrix},$$

and define the Lagrangian:

$$\mathcal{L}(w) = F(w) + \lambda^\intercal G(w) + \mu^\intercal H(w). \tag{3}$$

Using a full-step SQP approach for NLP (2), a single iteration corresponds to:

$$\begin{aligned}
w^{[i+1]} &= w^{[i]} + \Delta w^{\mathrm{QP}} \\
\lambda^{[i+1]} &= \lambda^{\mathrm{QP}}, \quad \mu^{[i+1]} = \mu^{\mathrm{QP}},
\end{aligned} \tag{4}$$

where $\left(\Delta w^{\mathrm{QP}}, \lambda^{\mathrm{QP}}, \mu^{\mathrm{QP}}\right)$ is the solution of:

$$\begin{aligned}
\underset{\Delta w \in \mathbb{R}^{n_w}}{\text{minimize}} \quad & \frac{1}{2} \Delta w^\intercal \boldsymbol{A} \Delta w + \boldsymbol{a}^\intercal \Delta w \\
\text{subject to} \quad & \boldsymbol{B} \Delta w = \boldsymbol{b}(w^{[i]}, \hat{x}_0) \\
& \boldsymbol{C} \Delta w \leq \boldsymbol{c}(w^{[i]}).
\end{aligned} \tag{5}$$

Using exact sensitivities constitutes choosing the matrices

$$\boldsymbol{A} := \nabla_w^2 \mathcal{L}(w^{[i]}, \lambda^{[i]}, \mu^{[i]}), \tag{6}$$

$$\boldsymbol{B} := \nabla_w^\intercal G(w^{[i]}), \tag{7}$$

$$\boldsymbol{C} := \nabla_w^\intercal H(w^{[i]}), \tag{8}$$

and the vectors

$$\boldsymbol{a} := \nabla_w F(w^{[i]}), \tag{9}$$

$$\boldsymbol{b}(w^{[i]}, \hat{x}_0) := -G(w^{[i]}, \hat{x}_0), \tag{10}$$

$$\boldsymbol{c}(w^{[i]}) := -H(w^{[i]}). \tag{11}$$

The structure of the exact sensitivity matrices is

$$\boldsymbol{A} = \begin{bmatrix} A_0 & & \\ & \ddots & \\ & & A_N \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} C_0 & & \\ & \ddots & \\ & & C_N \end{bmatrix}$$

$$\text{and } \boldsymbol{B} = \begin{bmatrix} \mathbb{I} & & & & \\ -\Phi_0^{w_0} & \mathbb{I} & & & \\ & -\Phi_1^{w_1} & \mathbb{I} & & \\ & & \ddots & \ddots & \\ & & & -\Phi_{N-1}^{w_{N-1}} & \mathbb{I} \end{bmatrix},$$

where $\Phi_k^{w_k}$ denotes (an approximation of) $\nabla_{w_k}^\intercal \phi_k(w_k)$ and $\mathbb{I}$ denotes an appropriately sized identity matrix. The evaluation of A_k and $\Phi_k^{w_k}$ is in many practical applications computationally the most demanding step and is therefore to be addressed first in case real-time computation times need to be attained.

1.2 Multi-level & Mixed-level Iterations

Multi-level iteration schemes (Bock et al., 2007) and mixed-level iteration schemes (Frasch et al., 2012) address the problem of the high computational cost to evaluate the sensitivity at every control interval by employing inexact Hessians, Jacobians, gradients and constraint residuals. Both methods break down the update scheme of quadratic program (QP) (5) into four levels, where *level A* consumes the least computational resources and *level D* the most. *Level A* implements a linear MPC, relying on a QP definition provided by higher levels. *Level B*, in addition, computes the constraint residuals and an approximation of the linear objective. Since *level C* and *level D* iterations are more important to the results in this paper, they are introduced in more detail.

Level C: Optimality Improvement. Keep the approximations of $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{C}$ fixed, evaluate equality and inequality constraints $\boldsymbol{b}(w^{[i]}, \hat{x}_0)$ and $\boldsymbol{c}(w^{[i]})$. Compute a modified gradient according to:

$$\boldsymbol{a}(w^{[i]}, \lambda^{[i]}, \mu^{[i]}) := \nabla_w \mathcal{L}(w^{[i]}, \lambda^{[i]}, \mu^{[i]}) - \boldsymbol{B}^\intercal \lambda^{[i]} - \boldsymbol{C}^\intercal \mu^{[i]}. \tag{12}$$

Solve the QP with state estimate $\hat{x}_0$, perform (4) and return feedback action $u_0^{[i+1]}$.

When the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are computed exactly, (12) is equivalent to $\nabla_w F(w^{[i]})$. However, when matrices $\boldsymbol{B}$ and

$\boldsymbol{C}$ are approximated, the use of $\boldsymbol{a} := \nabla_w F(w^{[i]})$ leads to convergence to a suboptimal point of (2) where

$$\nabla_w F(w^{[i]}) + \boldsymbol{A}\Delta w^{\mathrm{QP}} + \boldsymbol{B}^{\intercal}\lambda^{\mathrm{QP}} + \boldsymbol{C}^{\intercal}\mu^{\mathrm{QP}} = 0.$$

Whereas using the modified gradient (12), on convergence, the first-order optimality condition for (2) holds:

$$\nabla_w \mathcal{L}^{[i]} + \boldsymbol{A}\Delta w^{\mathrm{QP}} + \boldsymbol{B}^{\intercal}(\lambda^{\mathrm{QP}} - \lambda^{[i]}) + \boldsymbol{C}^{\intercal}(\mu^{\mathrm{QP}} - \mu^{[i]}) = 0.$$

See Griewank and Walther (2002); Diehl et al. (2010) for more details on this topic. Furthermore, Zanelli et al. (2016) investigates the consequences of omitting the gradient correction.

Level D: Full RTI. Recompute $\boldsymbol{A}, \boldsymbol{a}, \boldsymbol{B}, \boldsymbol{b}, \boldsymbol{C}$ and $\boldsymbol{c}$. Solve the QP with state estimate $\hat{x}_0$, perform (4) and return feedback action $u_0^{[i+1]}$. This essentially amounts to standard real-time iterations.

Multi-level iteration schemes implement at least two approximations to execute in parallel on a digital computer. Less computational demanding levels are executed at higher rates. The higher levels receive priority if their solution is available and broadcast information to lower levels. Mixed-level iterations extend upon this by not only choosing different approximation levels at different sampling rates, but also allowing different approximation levels over the prediction horizon.

Level D iterations can, for some applications, require too many resources to compute at sufficiently high rates. In these cases the approximations of the sensitivities become too outdated for good control performance. Therefore, the DOPUS algorithm is introduced, a method which situates between the *level C* and *level D* iterations.

2. THE DOPUS ALGORITHM

The algorithm central in this paper, DOPUS, exploits the structure of (2) for a partial linearization strategy. To reduce computations for matrices $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{C}$, the sensitivities for each interval are generated once as they appear in the horizon. Like RTI schemes shift the intermediate solution, DOPUS shifts the sensitivities backwards between iterations. In addition, DOPUS employs two optimality improvement steps of limited and scalable computational complexity.

Even when operating close to the optimum of the NLP (2), backward shifting of the sensitivities introduces errors. Firstly, the closed-loop trajectories of a system controlled by NMPC do not exactly correspond to the open-loop predictions. Secondly, factors such as model mismatch and exogenous influences to the controlled system disturb the NLP. The first optimality improvement step exploits the partial separability (Griewank and Toint, 1984) of the NLP. The quality of the linearization at each interval k is characterized by a norm-criterion that is based on the computation of the gradient of the Lagrangian and recomputing the quadratic approximation for the worst interval(s).

The second step modifies the linear objective in the underlying QP in order to account for the linearization error in the remaining intervals. This step is identical to the adjoint-gradient correction (12) employed in the *level C* updates of the multi-level iteration schemes. This modification comes at limited additional cost, since the majority of the computations for $\nabla_w \mathcal{L}$ is already performed by the re-linearization strategy.

2.1 Re-linearization Strategy

Regard the reordered Lagrangian (3):

$$\begin{aligned}
&\mathcal{L}(w, \lambda, \mu) = \\
&\qquad \lambda_0^{\intercal}(x_0 - \hat{x}_0) + \sum_{k=1}^{N} \lambda_k^{\intercal} x_k + E(x_N) + \mu_N^{\intercal} r(x_N) + \qquad (13) \\
&\qquad \sum_{k=0}^{N-1} \underbrace{\ell_k(x_k, u_k) - \lambda_{k+1}^{\intercal}\phi_k(x_k, u_k) + \mu_k^{\intercal} h_k(x_k, u_k)}_{=:\mathcal{L}_k(w_k, \lambda_{k+1}, \mu_k)},
\end{aligned}$$

and, with $\mathcal{L}_k^{[i]} = \mathcal{L}_k(w_k^{[i]}, \lambda_{k+1}^{[i]}, \mu_k^{[i]})$, further define

$$\mathcal{F}_k^{[i]} := \nabla \mathcal{L}_k^{[i]}, \tag{14}$$

$$\mathcal{H}_k^{[i]} := \begin{bmatrix} A_k & -(\Phi_k^{w_k})^{\intercal} & C_k^{\intercal} \\ -\Phi_k^{w_k} & & \\ C_k & & \end{bmatrix} \approx \nabla^2 \mathcal{L}_k^{[i]}. \tag{15}$$

The main idea of the re-linearization strategy is to assess the compliance to the secant condition of $\mathcal{F}_k$ between two subsequent SQP iterations. Consider the norm criterion

$$\rho_k := \left\| \mathcal{F}_{k+1}^{[i-1]} + \mathcal{H}_{k+1}^{[i-1]} \Delta\omega_{k+1}^{[i-1]} - \mathcal{F}_k^{[i]} \right\| / \|\Delta\omega_{k+1}^{[i-1]}\|, \tag{16}$$

where $\Delta\omega_{k+1}^{[i-1]} = \left[\Delta w_{k+1}^{\intercal} \; \Delta\lambda_{k+2}^{\intercal} \; \Delta\mu_{k+1}^{\intercal}\right]^{\intercal [i-1]}$. Inspired by quasi-Newton methods, this norm condition is used to gradually improve the matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ in the direction of the steps. For the M stages corresponding to the largest values of ρ_k in iteration i, the Jacobian $\mathcal{H}_k^{[i]}$ is recomputed by evaluating $\nabla^2 \mathcal{L}_k^{[i]}$. In order to save computations, the Jacobian approximation of the remaining $N-M-2$ stages are shifted backwards from the previous iteration.

Note that the evaluation of $\mathcal{F}_k^{[i]}$ involves computing the Jacobian multiplied by the Lagrange multipliers. Using backward algorithmic differentiation, the cost of this step is limited to at most five times the cost of evaluating the Lagrangian (Griewank and Walther, 2008).

2.2 Algorithm Outline

Algorithm 1 depicts a pseudo-code representation of DOPUS. During the initialization phase, before the control task is commenced, all QP data is initialized for an initial guess of the decision variables (w, λ, μ). In addition, $\mathcal{F}_k$ is computed for every interval. The controller starts by performing a single feedback step, i.e., one QP is solved and using (4) a new control action is determined and fed to the process. The primal variables are shifted backwards and the newly unknown last prediction node is initialized by forward simulation. In addition, the dual variables for the system dynamics constraints, the inequality constraints and the related sensitivities are shifted backwards as well. The Jacobians and Hessian for the last interval are computed exactly. The algorithm proceeds by performing the two optimality improvement steps, and the process is repeated by solving the new QP as soon as measurement data is available.

Algorithm 1 Pseudo-code for DOPUS algorithm. Index k in appropriate range for every variable, e.g. for x_k, $k \in \{0, \ldots, N\}$, while for u_k, $k \in \{0, \ldots, N-1\}$.

```
// Initialization
Initialize $w$ and compute (6)-(11)
For all k: compute (14)
// Control task initiated
while (true) {
  Await measurement/estimate $\hat{x}_0$
  Solve QP (5)
  Full step update of decision variables (4)
  Apply control $u_0$ to system
  For all k>0: shift $x_k, u_k, \mu_k$ backwards
  For all k>1: shift $\lambda_k$ backwards
  For all k>0: shift $A_k.C_k, \Phi_k^{w_k}$ backwards
  For k=N-1: compute (14) and (15)
  For all k<N-1: compute (14) and (16)
  For k according to M largest $\rho_k$: compute (15)
  Compute QP vectors (10)-(12)
}
```

Multi/mixed-level iteration schemes are usually combined with condensing, a technique to eliminate the linearized equality constraints from (5). In contrast to e.g., multi-level iterations, DOPUS does not directly benefit from the same kind of simplifications for the traditional condensing approaches. However, opportunities lie in combining DOPUS with the recently proposed block condensing methods (Axehill, 2015).

2.3 First Insights in Local Convergence Behavior

In order to gain understanding of the local convergence behavior of DOPUS a simplified problem is analyzed, namely a similarly structured unconstrained optimization problem. Note that the interplay between the optimization algorithm and the controlled process is neglected, i.e. $\hat{x}_0$ is kept constant and shifting of variables and sensitivities in between iterations is not considered. Consider

$$\underset{w \in \mathbb{R}^{n_w}}{\text{minimize}} \; F(w) := \sum_{k=0}^{N-1} \ell_k(x_k, x_{k+1}), \tag{17}$$

where $w = [x_0^\intercal \; x_1^\intercal \; \cdots \; x_N^\intercal]^\intercal$ and $\ell_k \colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{R}$. Further assume

- that $\ell_k \in \mathcal{C}^2$ and Lipschitz continuity of $\nabla_w^2 \ell_k$,
- strong convexity of (17), i.e. $\nabla^2 F(w) \succ 0 \,\forall w$, and
- that $w^\star$ is an optimizer for (17).

Due to the absence of constraints, DOPUS iterations simplify to:

$$w^{[i+1]} = w^{[i]} - (\boldsymbol{A}^{[i]})^{-1} \nabla_w F(w^{[i]}), \tag{18}$$

where $\nabla_w F(w) = \sum_{k=0}^{N-1} \nabla_w \ell_k(x_k, x_{k+1})$, and $\boldsymbol{A} = \sum_{k=0}^{N-1} \overline{A}_k$, with $\overline{A}_k$ an approximation of $\nabla_w^2 \ell_k(x_k, x_{k+1})$.

Assume local asymptotic convergence (i.e., $(w^{[i]}) \to w^\star$, $w^{[i+1]} \neq w^{[i]}$) of (18). Using a first-order Taylor approximation for $\nabla_w F$ state

$$\begin{aligned} &\nabla_w F(w^{[i]}) + \boldsymbol{A}^{[i]} \Delta w - \nabla_w F(w^{[i+1]}) \\ &= \left(\boldsymbol{A}^{[i]} - \nabla_w^2 F(w^{[i]}) \right) \Delta w^{[i]} + \mathcal{O}(\|\Delta w^{[i]}\|^2) = \\ &\sum_{k=0}^{N-1} \left(\overline{A}_k^{[i]} - \nabla_w^2 \ell_k(x_k^{[i]}, x_{k+1}^{[i]}) \right) \Delta w^{[i]} + \mathcal{O}(\|\Delta w^{[i]}\|^2). \end{aligned} \tag{19}$$

The Dennis and Moré (1974) characterization states that if and only if

$$\lim_{i \to \infty} \frac{\left\| \left(\boldsymbol{A}^{[i]} - \nabla_w^2 F(w^{[i]}) \right) \Delta w^{[i]} \right\|}{\left\| \Delta w^{[i]} \right\|} = 0. \tag{20}$$

holds, (18) converges Q-superlinearly. By triangle inequality, (19) and because $\|\Delta w_k^{[i]}\| \leq \|\Delta w^{[i]}\|$

$$\begin{aligned} &\left\| \left(\boldsymbol{A}^{[i]} - \nabla_w^2 F(w^{[i]}) \right) \Delta w^{[i]} \right\| / \|\Delta w^{[i]}\| \\ &\leq \sum_{k=0}^{N-1} \left\| \left(\overline{A}_k^{[i]} - \nabla_w^2 \ell_k(x_k^{[i]}, x_{k+1}^{[i]}) \right) \Delta w^{[i]} \right\| / \|\Delta w_k^{[i]}\| \\ &= \sum_{k=0}^{N-1} \rho_k^{[i]} + \mathcal{O}(\|\Delta w^{[i]}\|). \end{aligned}$$

Hence, for unconstrained optimization problems, DOPUS exhibits local superlinear convergence if both the following conditions hold: $\lim_{i \to \infty} \Delta w^{[i]} = 0$ and $\lim_{i \to \infty} \sum_{k=0}^{N-1} \rho_k^{[i]} = 0$. Since in each iteration the intervals according to the $M > 0$ largest values ρ_k are recomputed, and $\rho_k \to 0$ as $\|\Delta w_k\| \to 0$, it follows that (20) holds in case (18) converges.

These results for the unconstrained optimization do not directly extend to the original problem setting (2). The next section presents numerical results of the convergence behavior of DOPUS applied to a constrained optimization problem. A formal analysis of the convergence rate of DOPUS is a topic for future research.

3. NUMERICAL VALIDATION

Consider a planar robot arm, controlled by a predictive path-following controller. In predictive path-following, the objective is to follow geometric path without a prescribed timing (Faulwasser and Findeisen, 2016). The application of predictive path-following to robotic systems is an active research topic (Faulwasser et al., 2016; van Duijkeren et al., 2016) and a candidate application of DOPUS.

3.1 Planar Robot Model

Consider a planar two degree-of-freedom robot manipulator. The robot is modeled by a system of ODEs with four states $[\mathbf{q}^\intercal \; \dot{\mathbf{q}}^\intercal]^\intercal = [q_1 \; q_2 \; \dot{q}_1 \; \dot{q}_2]^\intercal$ and two inputs $\boldsymbol{\tau} = [\tau_1 \; \tau_2]^\intercal$:

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{q}} \\ B^{-1}(\mathbf{q}) \left(\boldsymbol{\tau} - C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} - F_\nu \dot{\mathbf{q}} - \boldsymbol{g}(\mathbf{q}) \right) \end{bmatrix}, \tag{21}$$

where $B \colon \mathbb{R}^2 \to \mathbb{R}^{2 \times 2}$ denotes the inertia matrix, $C \colon \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^{2 \times 2}$ is a matrix accounting for the Coriolis and centrifugal effects, $\boldsymbol{g} \colon \mathbb{R}^2 \to \mathbb{R}^2$ is the vector of torques due to gravity and $F_\nu \in \mathbb{R}^{2 \times 2}$ is the diagonal matrix of viscous friction coefficients in the two joints. The output map $\varphi \colon \mathbb{R}^2 \to \mathbb{R}^2$ represents the position of the tip of the second link in the Cartesian plane.

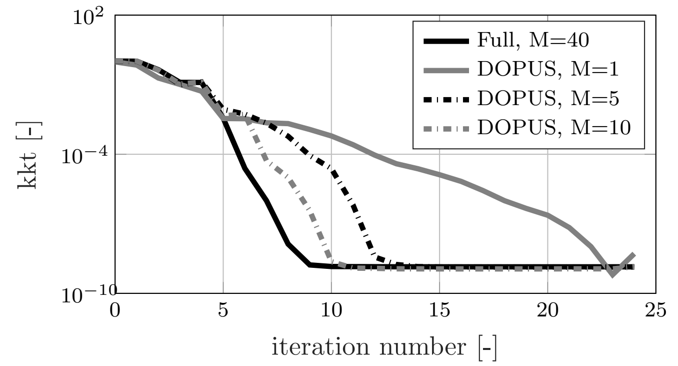

Fig. 1. The convergence behavior of full RTI in comparison to DOPUS for different values of M.

3.2 Path-following OCP

Consider a path $\theta \mapsto p(\theta) \in \mathbb{R}^2$. The path parameter θ and it time derivative $\dot{\theta}$ are introduced as virtual state and input respectively. The considered state and control vectors are therefore $x = [\mathbf{q}^\intercal\ \dot{\mathbf{q}}^\intercal\ \theta]^\intercal$ and $u = [\boldsymbol{\tau}^\intercal\ \dot{\theta}]^\intercal$. In the OCP below, the objective (22a) trades off traveled distance against path-following accuracy, alongside a small quadratic penalty term on the controls.

$$\underset{x(\cdot),u(\cdot)}{\text{minimize}} \int_{t_0}^{t_f} \left\| \begin{bmatrix} \varphi(\mathbf{q}(t)) - p(\theta(t)) \\ \boldsymbol{\tau}(t) \\ \dot{\theta}(t) \end{bmatrix} \right\|_W^2 \mathrm{d}t - \theta(t_f) \tag{22a}$$

$$\text{subject to} \quad (21) \qquad \forall t \in [t_0, t_f] \tag{22b}$$

$$\mathrm{d}\theta(t)/\mathrm{d}t = \dot{\theta}(t) \qquad \forall t \in [t_0, t_f] \tag{22c}$$

$$\|\boldsymbol{\tau}(t)\|_\infty \leq 10 \qquad \forall t \in [t_0, t_f] \tag{22d}$$

$$\hat{x}_0 = x(t_0) \tag{22e}$$

$$0 = \varphi(\mathbf{q}(t_f)) - p(\theta(t_f)). \tag{22f}$$

Equations (22b)-(22c) collect the augmented robot dynamics. Torque limits are enforced in (22d). Finally, (22f) is a terminal equality constraint on the position of the end-effector.

3.3 Implementation

A prototype of DOPUS is implemented in MATLAB. The OCP is approximated with an NLP using the direct multiple-shooting method, dividing the horizon in $N = 40$ intervals of each 0.01s. CasADi (Andersson, 2013) is used for algorithmic differentiation and the interface to CVODES from the SUNDIALS integrator suite (Hindmarsh et al., 2005). DOPUS is compared to the full RTI scheme by their closed-loop trajectories and their convergence behavior. In the simulated scenario, the end-effector of the robot follows a circular path. After one second of simulation time, a disturbance is inflicted which brings the robot to a standstill and changes its orientation slightly.

3.4 OCP Convergence Results

Fig. 1 shows the convergence behavior of the full RTI scheme alongside different configurations of DOPUS solving one single NLP, i.e., for a fixed $\hat{x}_0$. Convergence is measured by the so-called KKT condition, a measure for first-order optimality of a candidate solution for an NLP,

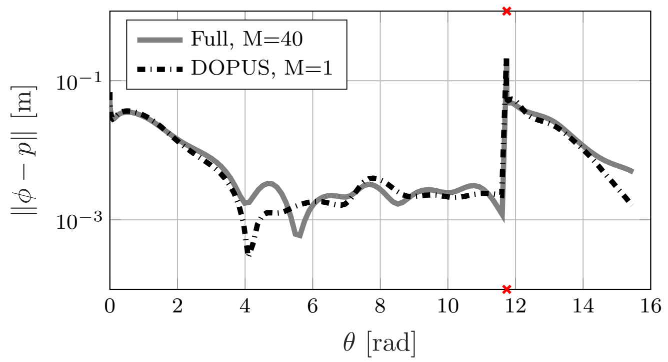

Fig. 2. The deviation from the circular reference path for full RTI and for DOPUS. The crosses indicate the time instance at which the process is disturbed.

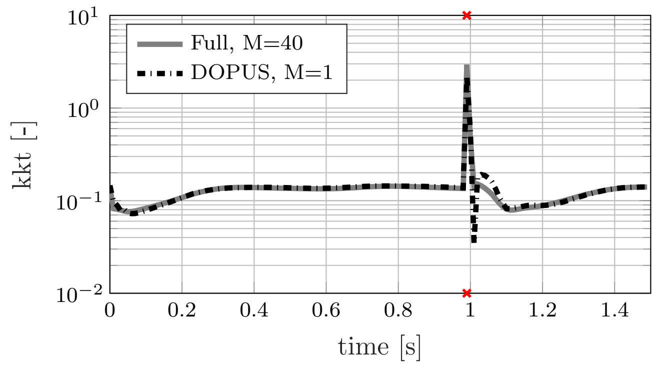

Fig. 3. Convergence behavior of the optimality conditions for full RTI and DOPUS. The crosses indicate the time instance at which the process is disturbed.

commonly used in SQP methods. It is observed that the value of M correlates to the convergence speed. Although important for a well-functioning optimization algorithm, a control engineer will not measure controller performance by the convergence rates of the optimization scheme. Let us therefore continue the analysis by closing the loop between the NMPC and a simulated model. In these closed-loop simulations, DOPUS is configured to recompute the Jacobian for one interval per iteration ($M = 1$).

3.5 Closed-loop Simulation Results

In Fig. 2 the deviation from the reference path is depicted for the full RTI scheme and DOPUS. The closed-loop path-following accuracy of the two simulations are similar up to the accuracy levels resulting from the trade-off with time-optimal motion. Furthermore, in Fig. 4 the closed-loop torque trajectories are depicted for simulations with DOPUS. As expected for an NMPC approximating time-optimal motion, torque bounds are active most of the time. Combined with Fig. 3 the results show that for the this non-trivial example the partial updating strategy has limited effect on the closed-loop performance and barely deteriorates the convergence behavior.

Two remarks are made regarding the presented results. Firstly, the prototype implementation of DOPUS is not suited for a meaningful comparison of the computational times of the full RTI scheme and DOPUS and therefore omitted. Secondly, one can observe that the KKT condi-

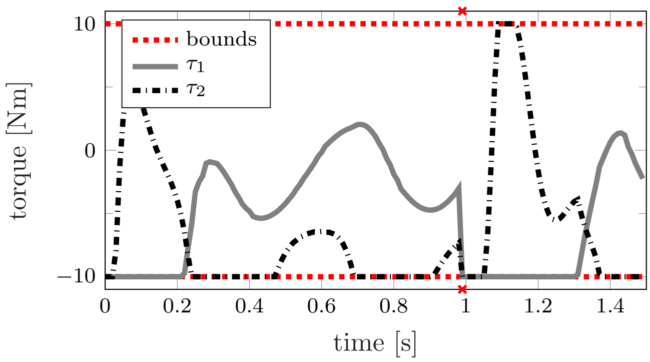

Fig. 4. Torque commands from control scheme for DOPUS. The crosses indicate the time instance at which the process is disturbed.

tion in Fig. 3 does not converge to near machine precision. It is noted that this behavior results from the receding horizon solution strategy and does not occur for a fixed initial condition $\hat{x}_0$.

4. CONCLUSION

This paper introduced a novel real-time iteration scheme: DOPUS, short for Dynamic Optimization with Partially Updated Sensitivities. First steps in analyzing the local convergence behavior of DOPUS are presented. Q-superlinear convergence is shown for the application to unconstrained optimization problems. A prototype implementation of DOPUS is used to compare its performance to the traditional real-time-iteration scheme. Despite the potential to be computationally much lighter, the simulation results show very similar closed-loop behavior. Furthermore, results show that convergence speed can be effectively traded for computational load by choosing the number of stages to update. The optimal number of stages to update will highly depend on the parallel computing capabilities of the processing unit.

Planned future work includes obtaining formal results for the convergence properties of DOPUS and writing an efficient implementation for real-time experiments on a 6-DOF robot arm.

REFERENCES

Albersmeyer, J., Beigel, D., Kirches, C., Wirsching, L., Bock, H.G., and Schlder, J.P. (2009). Fast nonlinear model predictive control with an application in automotive engineering. *Nonlinear Model Predictive Control*, 471–480.

Andersson, J. (2013). *A General-Purpose Software Framework for Dynamic Optimization*. PhD Thesis, KU Leuven.

Axehill, D. (2015). Controlling the level of sparsity in MPC. *Systems & Control Letters*, 76, 1–7.

Bock, H.G. and Plitt, K.J. (1984). A multiple shooting algorithm for direct solution of optimal control problems. In *Proc. of the 9^{th} IFAC World Congress*, 242–247.

Bock, H.G., Diehl, M., Kostina, E., and Schlöder, J.P. (2007). Constrained optimal feedback control of systems governed by large differential algebraic equations. In *Real-Time PDE-Constrained Optimization*, 3–22. SIAM.

Dennis, J.E. and Moré, J.J. (1974). A characterization of superlinear convergence and its application to quasi-Newton methods. *Mathematics of Computation*, 28(126), 549–549.

Diehl, M., Bock, H.G., and Schlöder, J.P. (2005). A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on Control and Optimization*, 43(5), 1714–1736.

Diehl, M., Walther, A., Bock, H.G., and Kostina, E. (2010). An adjoint-based SQP algorithm with quasi-Newton jacobian updates for inequality constrained optimization. *Optimization Methods and Software*, 25(4), 531–552.

Faulwasser, T. and Findeisen, R. (2016). Nonlinear model predictive control for constrained output path following. *IEEE Transactions on Automatic Control*, 61(4), 1026–1039.

Faulwasser, T., Weber, T., Zometa, P., and Findeisen, R. (2016). Implementation of nonlinear model predictive path-following control for an industrial robot. *IEEE Transactions on Control Systems Technology*. In press.

Frasch, J.V., Wirsching, L., Sager, S., and Bock, H.G. (2012). Mixed-level iteration schemes for nonlinear model predictive control. In *Proceedings of the 4th IFAC Nonlinear Model Predictive Control Conference*, 138–144.

Griewank, A. and Toint, P.L. (1984). *Numerical Analysis*, chapter Numerical experiments with partially separable optimization problems, 203–220. Springer Science + Business Media.

Griewank, A. and Walther, A. (2002). On constrained optimization by adjoint based quasi-Newton methods. *Optimization Methods and Software*, 17(5), 869–889.

Griewank, A. and Walther, A. (2008). *Evaluating Derivatives: principles and techniques of algorithmic differentiation*. SIAM, 2 edition.

Hindmarsh, A.C., Brown, P.N., Grant, K.E., Lee, S.L., Serban, R., Shumaker, D.E., and Woodward, C.S. (2005). SUNDIALS. *ACM Transactions on Mathematical Software*, 31(3), 363–396.

Kirches, C., Wirsching, L., Bock, H.G., and Schlder, J.P. (2012). Efficient direct multiple shooting for nonlinear model predictive control on long horizons. *Journal of Process Control*, 22(3), 540–550.

Kirches, C., Wirsching, L., Sager, S., and Bock, H.G. (2010). Efficient numerics for nonlinear model predictive control. *Recent Advances in Optimization and its Applications in Engineering*, 339–357.

Quirynen, R., Vukov, M., Zanon, M., and Diehl, M. (2014). Autogenerating microsecond solvers for nonlinear mpc: A tutorial using acado integrators. *Optimal Control Applications and Methods*.

van Duijkeren, N., Verschueren, R., Pipeleers, G., Diehl, M., and Swevers, J. (2016). Path-following NMPC for serial-link robot manipulators using a path-parametric system reformulation. In *Proceedings of the 2016 European Control Conference*, 477–482. Aalborg, Denmark.

Zanelli, A., Quirynen, R., and Diehl, M. (2016). An efficient inexact NMPC scheme with stability and feasibility guarantees. In *Proceedings of the 10th IFAC Symposium on Nonlinear Control Systems*. Monterey, CA, USA.