Efficient Numerics for Nonlinear Model Predictive Control

Christian Kirches[1], Leonard Wirsching[1], Sebastian Sager[1], and Hans Georg Bock[1]

Interdisciplinary Center for Scientific Computing (IWR)
Ruprecht–Karls–Universität Heidelberg,
Im Neuenheimer Feld 368, 69120 Heidelberg, Germany
{christian.kirches|leonard.wirsching|
sebastian.sager|bock}@iwr.uni-heidelberg.de

Summary. We review a closely connected family of algorithmic approaches for fast and real–time capable nonlinear model predictive control (NMPC) of dynamic processes described by ordinary differential equations or index-1 differential-algebraic equations. Focusing on active–set based algorithms, we present emerging ideas on adaptive updates of the local quadratic subproblems (QPs) in a multi–level scheme. Structure exploiting approaches for the solution of these QP subproblems are the workhorses of any fast active–set NMPC method. We present linear algebra tailored to the QP block structures that act both as a preprocessing and as block structured factorization methods.

1 Introduction

Nonlinear model predictive control has become an increasingly popular control approach, and is both theoretically and computationally well-established. However, its application to time-critical systems requiring fast feedback is still a major computational challenge. We review a closely connected family of algorithmic approaches for fast and real–time capable NMPC of dynamic processes described by ordinary differential equations (ODEs) or differential-algebraic equations (DAEs). We start with the discretization of the optimal control problems (OCPs), focus on active–set based algorithms for the solution of the resulting nonlinear programs (NLPs), and present emerging ideas on adaptive updates of the local quadratic subproblems in a multi–level scheme. Structure exploiting approaches for the solution of these QP subproblems are the workhorses of any fast active–set NMPC method. Here, we present linear algebra tailored to the QP block structures that act both as a preprocessing and as block structured factorization methods. An introduction to a new block structured active set QP method concludes our review.

1.1 Direct Optimal Control in Nonlinear Model Predictive Control

We consider the following class of optimal control problems which typically arise in nonlinear model predictive control.

$$\min_{x(\cdot),u(\cdot)} \; J(x(t),u(t);p) = \int_{t_0}^{t_\mathrm{f}} L\left(x(t),u(t);p\right)\,\mathrm{d}t + E\left(x(t_\mathrm{f};p)\right) \tag{1a}$$

$$\text{s.t.} \qquad x(t_0) = x_0, \tag{1b}$$

$$\dot{x}(t) = f\left(t,x(t),u(t);p\right), \qquad \forall t \in [t_0,t_\mathrm{f}] \tag{1c}$$

$$0 \leq h_\mathrm{path}\left(x(t),u(t);p\right), \qquad \forall t \in [t_0,t_\mathrm{f}] \tag{1d}$$

$$0 \leq h_\mathrm{end}(x(t_\mathrm{f});p). \tag{1e}$$

The OCPs are formulated on a fixed and finite time horizon $\mathcal{T} := [t_0,t_\mathrm{f}]$ which is called the *prediction horizon*. We denote by $x(t) \in \mathbb{R}^{n_\mathrm{x}}$ the state vector of the dynamic process, and by $u(t) \in \mathbb{R}^{n_\mathrm{u}}$ the vector of continuous controls influencing the dynamic process. In the following, we drop the explicit time dependency and write x and u as shorthands for $x(t)$ and $u(t)$.

The state trajectory is determined by the initial value problem (IVP) (1b)-(1c), where x_0 is the current state of the process and $f\left(t,x(t),u(t);p\right)$ describes the dynamic process model. In this paper we consider process models described by ordinary differential equations to keep the presentation clear. However, the approach can naturally be extended to models described by differential-algebraic equations (see [22]). States and controls may be subject to path constraints $h_\mathrm{path}\left(x(t),u(t);p\right)$ and the final state may be restricted by an end-point constraint $h_\mathrm{end}(x(t_\mathrm{f});p)$.

The objective function is of Bolza type with a Lagrange term $L\left(x,u;p\right)$ and a Mayer term $E\left(x(t_\mathrm{f});p\right)$. An important and frequently occurring choice for the Lagrange term are *least-squares* objective functions of the form

$$L\left(x,u;p\right) = \|l(x,u;p)\|_2^2, \tag{2}$$

where l is the least-squares residual vector. A typical example is the *tracking-type objective*

$$L\left(x,u;p\right) = \left(x-\bar{x}\right)^T Q(t)\left(x-\bar{x}\right) + \left(u-\bar{u}\right)^T R(t)\left(u-\bar{u}\right), \tag{3}$$

with $\bar{x}$ and $\bar{u}$ are given reference trajectories for x and u, and $Q(t)$ and $R(t)$ are suitable positive definite weighting matrices. A typical choice for the Mayer term is the quadratic cost

$$E\left(x(t_\mathrm{f});p\right) = \left(x(t_\mathrm{f}) - \bar{x}(t_\mathrm{f})\right)^T P\left(x(t_\mathrm{f}) - \bar{x}(t_\mathrm{f})\right), \tag{4}$$

with a suitable weighting matrix P. The Mayer term can be used — typically in conjunction with the end–point constraint h_end — to design feedback control schemes that guarantee stability of the closed-loop system. For a detailed discussion of nominal stability for NMPC see, e.g., [24].

The problem may also depend on time-independent model parameters $p \in \mathbb{R}^{n_\mathrm{P}}$, but they are not included as degrees of freedom for the optimization. In practice, it may happen that some of the parameters change their value during the runtime of the process. This gives rise to the important area of online state and parameter estimation (see [27, 11]). However, in this work we assume the parameters to be known and constant over time, and we will drop them in the following presentation.

1.2 The Principle of Model Predictive Control

Model predictive control schemes generate feedback by repetitively performing the following actions:

1. Obtain the process state x_0 at the current sampling time t_0.
2. Solve OCP (1) for the current x_0 to obtain optimal state and control trajectories $x^\star(\cdot; x_0)$ and $u^\star(\cdot; x_0)$.
3. Feed back the first part of $u^\star(\cdot; x_0)$ as feedback control to the process during the current sampling period $[t_0, t_0 + \delta]$.

Advantages of this approach are the possibility to use a sophisticated process model to predict the behavior of the process, the flexibility in the choice of an optimization criterion and a natural incorporation of the process constraints.

However, solving an OCP for each sampling time is computationally challenging. The fact that OCP (1) depends parametrically on x_0 has to be exploited by carefully using the results from the last problem to solve the current problem.

1.3 Direct Multiple Shooting Discretization

Approaches to solve OCP (1) divide up in *indirect* methods which first set up optimality conditions for the OCP and then discretize and solve these conditions (see [8]) and *direct* methods which first discretize the OCP and then setup und solve optimality conditions for the arising nonlinear program. In this work, we will consider the *Direct Multiple Shooting* method, first described by [26] and [7] and extended in a series of subsequent works (see, e.g., [23]). With the optimal control software package MUSCOD-II an efficient implementation of this method is available. For the use of other direct methods such as *Single Shooting* and *Collocation* in the context of online optimization we refer to the recent survey [10] and the references therein.

For a suitable partition of the horizon $[t_0, t_\mathrm{f}]$ into N subintervals $[t_i, t_{i+1}]$, $0 \leq i < N$, we set

$$u(t) = \varphi_i(t, q_i), \quad \text{for } t \in [t_i, t_{i+1}] \tag{5}$$

where φ_i are given basis functions parametrized by a finite dimensional parameter vector q_i. The functions φ_i may be for example vectors of polynomials; a common choice for NMPC are piecewise constant controls

$$\varphi_i(t, q_i) = q_i \quad \text{for } t \in [t_i, t_{i+1}]. \tag{6}$$

Note that for this particular choice of basis functions bounds on the control u transfer immediately to bounds on the parameter vectors q_i and vice versa.

Furthermore, we introduce additional variables s_i that serve as initial values for computing the state trajectories independently on the subintervals

$$\dot{x}_i(t) = f(t, x_i(t), \varphi_i(t, q_i)),\ x_i(t_i) = s_i,\ t \in [t_i, t_{i+1}],\ 0 \le i < N.$$

To ensure continuity of the optimal trajectory on the whole interval $[t_0, t_\mathrm{f}]$ we add matching conditions to the optimization problem

$$s_{i+1} = x_i(t_{i+1}; t_i, s_i, q_i),\ 0 \le i < N \tag{7}$$

where $x_i(t; t_i, s_i, q_i)$ denotes the solution of the IVP on $[t_i, t_{i+1}]$, depending on s_i and q_i. This method allows using state-of-the-art adaptive integrators for function and sensitivity evaluation, cf. [2, 25]. The path constraints (1d) are enforced in the shooting nodes t_i.

1.4 Sequential Quadratic Programming

From the multiple shooting discretization we obtain the NLP

$$\begin{aligned}
\min_{s,q} \quad & \sum_{i=0}^{N-1} L_i\left(s_i, q_i\right) + E\left(s_N\right) && && \text{(8a)}\\
\text{s.t.} \quad & 0 = s_0 - x_0, && && \text{(8b)}\\
& 0 = s_{i+1} - x_i(t_{i+1}; t_i, s_i, q_i), && 0 \le i < N, && \text{(8c)}\\
& 0 \le h_\mathrm{path}(s_i, \varphi_i(t_i, q_i)), && 0 \le i < N, && \text{(8d)}\\
& 0 \le h_\mathrm{end}(s_N), && && \text{(8e)}
\end{aligned}$$

where

$$L_i\left(s_i, q_i\right) = \int_{t_i}^{t_{i+1}} L(x(t), \varphi_i(t, q_i))\ \mathrm{d}t. \tag{9}$$

This NLP depends parametrically on x_0 and can be written in the generic form

$$\min_w\ \phi(w) \text{ s.t. } c(w) + \Lambda x_0 = 0,\ d(w) \ge 0, \tag{10}$$

where $\Lambda = (-I_{n_x}, 0, 0, \dots)$ and $w = (s_0, q_0, \dots, s_{N-1}, q_{N-1}, s_N)$ is the vector of all unknowns.

We choose to solve this NLP using a Newton–type framework. The various structural features such as the separable Lagrangian, the block diagonal Hessian, and the block structure of the Jacobians of the matching constraints (7) can be extensively exploited by tailored linear algebra. In particular using block–wise high–rank updates of the Hessian and a structure–exploiting algorithm for the solution of the arising QP subproblems as presented in section 4 improves convergence speed and computational efficiency.

Starting with an initial guess (w^0, λ^0, μ^0), a full step sequential quadratic programming (SQP) iteration is performed as follows

$$w^{k+1} = w^k + \Delta w^k, \quad \lambda^{k+1} = \lambda^k_{\mathrm{QP}}, \quad \mu^{k+1} = \mu^k_{\mathrm{QP}} \tag{11}$$

where $(\Delta w^k, \lambda^k_{\mathrm{QP}}, \mu^k_{\mathrm{QP}})$ is the solution of the QP subproblem

$$\min_{\Delta w} \quad \tfrac{1}{2}\Delta w^T B^k \Delta w + b^{kT}\Delta w \tag{12a}$$

$$\text{s.t.} \quad 0 = C^k \Delta w + c(w^k) + \Lambda x_0, \tag{12b}$$

$$0 \leq D^k \Delta w + d(w^k). \tag{12c}$$

Here, B^k denotes an approximation of the Hessian of the Lagrangian of (8), and b^k, C^k and D^k are the objective gradient and the Jacobians of the constraints c and d.

2 SQP based Model–Predictive Control

2.1 Initial Value Embedding and Tangential Predictors

The key to a performant numerical algorithm for NMPC is to reuse information from the last QP subproblem to initialize the new subproblem. This is due to the fact that subsequent problems differ only in the parameter x_0 of the linear embedding Λ. Given that the sampling periods are not too long and that the process does not behave too different from the prediction by the model, the solution information of the last problem can be expected to be a very good initial guess close to the solution of the new subproblem.

In [9] and related works it has been proposed to initialize the current problem with the full solution of the previous optimization run, i.e., control *and* state variables. Doing so, the value of s_0 will in general *not* be the value of the current state. By explicitly including the initial value constraint (8b) in the QP formulation, we can guarantee that the constraint is satisfied after the first full Newton–type step due to its linearity in x_0. This is called the *initial value embedding* technique.

On the other hand, by using the full solution of the last problem as initialization of the new problem, the first full Newton–type step already gives us a first order approximation of the solution of the new problem, even in the presence of an active set change. This motivates the idea of *real–time iterations*, which perform only one Newton–type iteration per sample, and is at the same time the main reason for our preference of active set methods over interior–point techniques. We refer to [10] for a detailed survey on the topic of initial value embeddings and the resulting first order tangential predictors.

2.2 Real–Time Iterations

Using the initial value embedding also has an important algorithmical advantage. We can evaluate all derivatives and all function values except the initial value constraint prior to knowing the current state x_0. Consequently, we can also presolve a major part of QP (12). This allows to separate each real–time iteration into the following three phases.

Preparation

All functions and derivatives that do not require knowledge of x_0 are evaluated using the iterate of the previous step (w^k, λ^k, μ^k). Due to its special structure, the variables $(\Delta s_1, \ldots, \Delta s_N)$ can be eliminated from QP (12), cf. section 4.

Feedback

As soon as x_0 is available, Δs_0 can be eliminated as well and a small QP only in the variables $(\Delta q_0, \ldots, \Delta q_{N-1})$ is solved. The variable $q_0^{k+1} = q_0^k + \Delta q_0^k$ is then given to the process, allowing to compute the feedback control $\varphi_0(t, q_0^{k+1})$. Thus, the actual feedback delay reduces to the solution time of the QP resulting from both eliminations. The affine-linear dependence of this QP on x_0 via Λ can further be exploited by parametric quadratic programming as described in section 2.3.

Transition

Finally, the eliminated variables are recovered and step (11) is performed to obtain the new set of NLP variables $(w^{k+1}, \lambda^{k+1}, \mu^{k+1})$.

2.3 Parametric Quadratic Programming

Both the structured NLP (8) and the QP subproblems (12) derived from it depend parametrically on x_0. This linear dependence on x_0 is favourably exploited by parametric active set methods for the solution of (12), cf. [4] and [12]. The idea here is to introduce a linear affine homotopy in a scalar parameter $\tau \in [0, 1] \subset \mathbb{R}$ from the QP that was solved in iteration $k-1$ to the QP to be solved in iteration k:

$$\min_{\Delta w} \quad \tfrac{1}{2} \Delta w^T B^k \Delta w + b^T(\tau) \Delta w \tag{13a}$$

$$\text{s.t.} \quad 0 = C^k \Delta w + c(\tau) + \Lambda x_0(\tau), \tag{13b}$$

$$0 \leq D^k \Delta w + d(\tau), \tag{13c}$$

with initial values $x_0(0) = x_0^{k-1}$, $x_0(1) = x_0^k$. Linear affine gradient and constraint right hand sides on the homotopy path,

$$b(\tau) = (1-\tau)b(w^{k-1}) + \tau b(w^k), \tag{14}$$

$$c(\tau) = (1-\tau)c(w^{k-1}) + \tau c(w^k), \tag{15}$$

$$d(\tau) = (1-\tau)d(w^{k-1}) + \tau d(w^k), \tag{16}$$

allow for an update of the QP's vectors in iteration k by one of the multi–level scheme's modes, cf. section 3. From the optimality conditions of QP (13) in $\tau = 0$ and $\tau = 1$ it is easily found that an update of the QP's matrices is possible as well, without having to introduce matrix–valued homotopies.

Using this approach to compute the SQP algorithm's steps has multiple advantages. First, a phase I for finding a feasible point of the QP is unnecessary, as we can start the homotopy in a trivial QP with zero vectors and known optimal solution. Second, we can monitor the process of solving the QP using the distance $1 - \tau$ to the homotopy path's end. Intermediate iterates are physically meaningful and optimal for a known QP on the homotopy path. Thus, intermediate control feedback can be given during the ongoing solution process. Finally, premature termination of the QP solution process due to computing time constraints becomes possible, cf. [12].

3 The Multi–Level Iteration Scheme

A novel and promising algorithmic approach to SQP based nonlinear model predictive control is the multi–level iteration method, first proposed in [6, 5].

The multi–level iteration method aims at providing feedback very fast, while updating the data of the feedback-generating QP with information from the process on different levels. We distinguish four levels or modes, from which multi–level iteration schemes can be combined.

3.1 Mode A: Feedback Iterations

For Mode A, we assume that QP (12) is given with a Hessian approximation $\overline{B}$, objective gradient $\overline{b}$, constraint values $\overline{c}$, $\overline{d}$, and Jacobians $\overline{C}$, $\overline{D}$, and working on a reference solution $(\overline{w}, \overline{\lambda}, \overline{\mu})$. The aim of Mode A is to compute feedback by resolving the QP for new given current states x_0 and returning the control parameters $\overline{q}_0 + \Delta q_0^k$ to the process as quickly as possible. Mode A is essentially a linear model predictive controller (LMPC). In contrast to LMPC which uses linearizations of a steady state model, Mode A works on linearizations provided by higher modes of the multi–level scheme, which may include transient phases of the nonlinear process.

3.2 Mode B: Feasibility Improvement Iterations

In Mode B, we assume that we have a Hessian approximation $\overline{B}$, a reference objective gradient $\overline{b}$, Jacobians $\overline{C}$, $\overline{D}$ and a reference solution $(\overline{w}, \overline{\lambda}, \overline{\mu})$. Furthermore, Mode B holds its own variables w_{B}^k, which are initially set to $\overline{w}$.

To finish the setup of QP (12), we evaluate new function values $c(w^k)$ and $d(w^k)$ and approximate the QP gradient by $b(w^k) = \overline{b} + \overline{B}\left(w_{\mathrm{B}}^k - \overline{w}\right)$, so that we come up with the following QP

$$\begin{aligned} \min_{\Delta w_{\mathrm{B}}^k} \quad & \frac{1}{2}\,\Delta w_{\mathrm{B}}^{k\,T}\,\overline{B}\,\Delta w_{\mathrm{B}}^k + b(w^k)^T \Delta w_{\mathrm{B}}^k \\ \text{s.t.} \quad & \overline{C}\,\Delta w_{\mathrm{B}}^k + c(w^k) + \Lambda x_0 = 0 \\ & \overline{D}\,\Delta w_{\mathrm{B}}^k + d(w^k) \geq 0. \end{aligned}$$

Once we have solved the QP, we return the control parameters $q_{\mathrm{B},0}^k + \Delta q_{\mathrm{B},0}^k$ to the process and iterate by setting $w_{\mathrm{B}}^{k+1} = w_{\mathrm{B}}^k + \Delta w_{\mathrm{B}}^k$.

When performing Mode B iterations with a fixed x_0, one can show that w_{B}^k converges locally to a suboptimal but feasible point of NLP (8), thus Mode B iterations are also referred to as *feasibility improvement iterations*. Optimality is approximately treated by the gradient updates. In comparison to Mode A, the additional computational cost for a Mode B iteration are evaluations of the constraints c and d, and condensing of the constraint vectors and the approximated gradient. Since the QP matrices are fixed, no new matrix decompositions are required during QP solving.

3.3 Mode C: Optimality Improvement by Adjoint SQP Iterations

In Mode C, we assume that we have a Hessian approximation $\overline{B}$, Jacobians $\overline{C}$, $\overline{D}$ and a reference solution $(\overline{w}, \overline{\lambda}, \overline{\mu})$. Furthermore, Mode C holds its own variables $(w_{\mathrm{C}}^k, \lambda_{\mathrm{C}}^k, \mu_{\mathrm{C}}^k)$, which are initially set to $(\overline{w}, \overline{\lambda}, \overline{\mu})$. To finish the setup of QP (12), we have to evaluate new function values $c(w^k)$ and $d(w^k)$, and we compute a modified gradient by

$$b(w^k) = \nabla\phi(w^k) + \left(\overline{C}^T - C^{k\,T}\right)\,\lambda^k + \left(\overline{D}^T - D^{k\,T}\right)\,\mu^k, \tag{17}$$

where C^k and D^k are the Jacobians of the constraints c and d at w^k. However, the Jacobians need not to be calculated completely, but rather the adjoint derivatives $C^{k\,T}\lambda^k$ and $D^{k\,T}\mu^k$. This can be done efficiently by the reverse mode of automatic differentiation, cf. [17]. After solving the following QP

$$\begin{aligned} \min_{\Delta w_{\mathrm{C}}^k} \quad & \frac{1}{2}\,\Delta w_{\mathrm{C}}^{k\,T}\,\overline{B}\,\Delta w_{\mathrm{C}}^k + b(w^k)^T \Delta w_{\mathrm{C}}^k \\ \text{s.t.} \quad & \overline{C}\,\Delta w_{\mathrm{C}}^k + c(w^k) + \Lambda x_0 = 0 \\ & \overline{D}\,\Delta w_{\mathrm{C}}^k + d(w^k) \geq 0, \end{aligned}$$

we return the control parameters $q_{\mathrm{C},0}^k + \Delta q_{\mathrm{C},0}^k$ to the process and iterate by setting

$$w_{\mathrm{C}}^{k+1} = w_{\mathrm{C}}^k + \Delta w_{\mathrm{C}}^k, \quad \lambda_{\mathrm{C}}^{k+1} = \lambda_{\mathrm{QP}}^k, \quad \mu_{\mathrm{C}}^{k+1} = \mu_{\mathrm{QP}}^k, \tag{18}$$

where λ^k_{QP} and μ^k_{QP} are the multipliers obtained from the QP solution.

When performing Mode C iterations with a fixed x_0, one can show local convergence of the sequence $(w^k_{\mathrm{C}}, \lambda^k_{\mathrm{C}}, \mu^k_{\mathrm{C}})$ to a KKT–point of NLP (8), cf. [31], thus Mode C iterations are also referred to as *optimality improvement* iterations. In comparison to Mode B, the additional computational cost for a Mode C iteration are evaluations of the adjoint derivatives $C^{k\,T}\lambda^k$ and $D^{k\,T}\mu^k$ which can be obtained at no more than five times the cost of the respective constraint evaluation [17]. Again, no new matrix decompositions are required during QP solving.

3.4 Mode D: Forward SQP Iterations

Mode D iterations are essentially standard real–time iterations, i.e. full SQP iterations. Mode D holds its own variables $(w^k_{\mathrm{D}}, \lambda^k_{\mathrm{D}}, \mu^k_{\mathrm{D}})$ and in each Mode D iteration, we evaluate the constraints $c(w^k)$ and $d(w^k)$, the objective gradient $b(w^k)$, and the constraint Jacobians $C(w^k)$ and $D(w^k)$, and build a new Hessian approximation $B(w^k)$. After solving QP (12) the control parameters $q^k_{\mathrm{D},0} + \Delta q^k_{\mathrm{D},0}$ are given to the process and we iterate by setting

$$w^{k+1}_{\mathrm{D}} = w^k_{\mathrm{D}} + \Delta w^k_{\mathrm{D}}, \quad \lambda^{k+1}_{\mathrm{D}} = \lambda^k_{\mathrm{QP}}, \quad \mu^{k+1}_{\mathrm{D}} = \mu^k_{\mathrm{QP}}, \tag{19}$$

where λ^k_{QP} and μ^k_{QP} are the multipliers obtained from the QP solution. In each Mode D iteration we have to evaluate the full constraint Jacobians, which amounts to the computational cost of the number of degrees of freedom times the cost for a constraint evaluation. Furthermore, a full initial decomposition has to be performed for the solution of the QP, cf. section 4, which depending on the chosen block structured QP method may have a computational complexity of up to $O(N^2 n^3)$.

3.5 Assembling Multi-level Iteration Schemes

From the four modes described above, we can assemble multi-level iteration schemes in various ways. A sequential approach is outlined in the following:

Algorithm 1: Sequential Multi-level Iteration Scheme

Choose initial $\overline{B}$,$\overline{C}$, $\overline{D}$,$\overline{b}$,$\overline{c}$, $\overline{d}$ and $(\overline{w}, \overline{\lambda}, \overline{\mu})$ for all modes;
while *Process running* **do**
  Determine mode;
  case *mode A*
    Perform calculations described in subsection 3.1;
  endsw
  case *mode B*
    Perform calculations described in subsection 3.2;
    Update b, c, d in mode A with the new values from mode B;
    Update $\overline{w}$ in mode A with w_{B};
  endsw
  case *mode C*
    Perform calculations described in subsection 3.3;
    Update b, c, d in mode A and B with the new values from mode C;
    Update $\overline{w}$ in mode A and w_{B} with w_{C};
  endsw
  case *mode D*
    Perform calculations described in subsection 3.4;
    Update B, C, D, b, c, d in mode A, B and C with the new values from mode D;
    Update $\overline{w}$ in mode A and w_{B}, w_{C} with w_{D} and $(\lambda_{\mathrm{c}}, \mu_{\mathrm{c}})$ with $(\lambda_{\mathrm{D}}, \mu_{\mathrm{D}})$;
  endsw
endw

However, a parallel implementation would be an even more natural choice, starting all modes at one time and then performing the updates described above whenever one of the modes has finished one calculation cycle. Of couse, one has to consider the issue of synchronization, so that the faster modes are updated only after finishing their current feedback calculation.

Multi-level iteration schemes do not need to employ all modes described above. An example application of a sequential multi-level iteration scheme using modes A and D to a vehicle model is presented in [1].

3.6 Euler Steps

In some cases the limiting factor for feedback generation is the sampling rate of the system states x_0, e.g., if the current states are obtained from a measurement procedure with limited throughput.

If it is still desired to update the feedback control with a higher frequency, a possible remedy is to use the model to predict the next x_0 by an Euler step

$$x_0^{\mathrm{new}} = x_0 + hf(x_0, \varphi_0(t_0, q_0^k)) \tag{20}$$

with a small stepsize $h = t_0^{\text{new}} - t_0$ and use x_0^{new} to obtain a new feedback q_0^{k+1}. In addition, as the explicit Euler scheme generates a linear affine homotopy path for $x_0^{\text{new}}(t)$ starting in t_0, it can be readily combined with the parametric QP strategy of section 2.3. This allows system state predictions to enter the QP solution even before the solution process has been completed.

3.7 Computing the Local Feedback Law

Phase A iterations can even be used to generate a local feedback law which maps differences $\Delta x_0 = x_0^{\text{new}} - x_0$ to feedback updates and thus can be used as an explicit continuous feedback law betweeen two following QP solutions.

To see this, we consider the Karush-Kuhn-Tucker (KKT) system of the QP after a successful solution

$$\underbrace{\begin{pmatrix} B & -C^T & -D_{\mathbb{A}}^T \\ C & & \\ D_{\mathbb{A}} & & \end{pmatrix}}_{:=K} \begin{pmatrix} \Delta w \\ \Delta\lambda \\ \Delta\mu_{\mathbb{A}} \end{pmatrix} = - \begin{pmatrix} b \\ c + \Lambda x_0 \\ d_{\mathbb{A}} \end{pmatrix}, \tag{21}$$

where $\mathbb{A}$ is the optimal active set. Let $\mathbb{I}$ be the index set of Δq_0 within Δw. We can easily calculate the part of the inverse of K which gives us Δq_0 when applied to the right hand side by solving

$$K^T X_i = e_i, \quad i \in \mathbb{I}, \tag{22}$$

with e_i the i-th unity vector. Since a decomposition of K is available from the QP solver, this amounts to only n_{u} backsolves. Assuming that $\mathbb{A}$ keeps constant for small changes in x_0, we can determine an update for Δq_0 by building

$$X^T \begin{pmatrix} 0 \\ \Lambda\Delta x_0 \\ 0 \end{pmatrix}, \tag{23}$$

for which we actually need only a small part of the matrix X.

4 Structured Quadratic Programming

This final part of our survey is concerned with numerical methods for the efficient solution of the QPs that arise from a direct multiple shooting discretization of the model predictive control problem. The focus is put on methods that efficiently exploit the block structure of problem (24) by appropriate linear algebra. We present the condensing algorithm due to [26, 7] that works as a preprocessing step, mention Riccati recursion to exploit the block structure, and conclude with a block structured active set method.

4.1 The Block Structured Quadratic Subproblem

To gain insight into the direct multiple shooting structure of QP (12) we rewrite it to expose the individual block matrices. The matching conditions (7) are separated in (24b), and equality as well as inequality point constraints are collected in (24c):

$$\min_{\Delta w} \sum_{i=0}^{N} \left(\tfrac{1}{2}\Delta w_i^T B_i \Delta w_i + \Phi_i^T \Delta w\right) \tag{24a}$$

$$\text{s.t. } 0 = X_i \Delta w_i - \Delta w_{i+1} - h_i \qquad 0 \le i < N \tag{24b}$$

$$0 \leqq R_i \Delta w_i + r_i \qquad 0 \le i \le N \tag{24c}$$

4.2 Condensing and Dense Active Set Methods

The purpose of the following condensing algorithm that is due to [26] and [7], cf. also [23], is to exploit the block sparse structure of QP (24) in a preprocessing or *condensing* step that transforms the QP into a smaller and densely populated one.

Reordering the Sparse Quadratic Problem

We start by reordering the constraint matrix of QP (24) to separate the multiple shooting state values $\Delta v_1 = (\Delta s_1, \ldots, \Delta s_N)$ introduced in section 1.3 from the single shooting values $\Delta v_2 = (\Delta s_0, \Delta q_0, \ldots, \Delta q_{N-1})$ as shown in (25). Therein, we use partitions $X_i = (X_i^{\mathrm{s}}\ X_i^{\mathrm{q}})$ and $R_i = (R_i^{\mathrm{s}}\ R_i^{\mathrm{q}})$ of the jacobians X_i and R_i with respect to Δs and Δq.

$$\left(\begin{array}{cccc|cccc} X_0^{\mathrm{s}} & X_0^{\mathrm{q}} & & & -I & & & \\ & X_1^{\mathrm{q}} & & & X_1^{\mathrm{s}} & -I & & \\ & & \ddots & & & \ddots & \ddots & \\ & & & X_{N-1}^{\mathrm{q}} & & & X_{N-1}^{\mathrm{s}} & -I \\ \hline R_0^{\mathrm{s}} & R_0^{\mathrm{q}} & & & & & & \\ & R_1^{\mathrm{q}} & & & R_1^{\mathrm{s}} & & & \\ & & \ddots & & & \ddots & & \\ & & & R_{N-1}^{\mathrm{q}} & & & R_{N-1}^{\mathrm{s}} & \\ & & & & & & & R_N^{\mathrm{s}} \end{array}\right). \tag{25}$$

Elimination Using the Matching Conditions

We may now use the negative identity matrix blocks of the equality matching conditions as pivots to formally eliminate the state values $(\Delta s_0, \ldots, \Delta s_N)$ from system (25), analogous to the usual Gaussian elimination method for triangular matrices. From this elimination procedure the dense constraint matrix

$$\begin{pmatrix} \overline{X} & -I \\ \overline{R} & 0 \end{pmatrix} := \left(\begin{array}{ccccc|c} X_0^{\mathrm{s}} & X_0^{\mathrm{q}} & & & & -I & \\ X_1^{\mathrm{s}}X_0^{\mathrm{s}} & X_1^{\mathrm{s}}X_0^{\mathrm{q}} & X_1^{\mathrm{q}} & & & & -I \\ \vdots & \vdots & \vdots & \ddots & & & \ddots \\ \Pi_0^{N-1} & \Pi_1^{N-1}X_0^{\mathrm{q}} & \Pi_2^{N-1}X_1^{\mathrm{q}} & \cdots & X_{N-1}^{\mathrm{q}} & & & -I \\ \hline R_0^{\mathrm{s}} & R_0^{\mathrm{q}} & & & & & \\ R_1^{\mathrm{s}}X_0^{\mathrm{s}} & R_1^{\mathrm{s}}X_0^{\mathrm{q}} & R_1^{\mathrm{q}} & & & & \\ \vdots & \vdots & \vdots & \ddots & & & \\ R_N^{\mathrm{s}}\Pi_0^{N-1} & R_N^{\mathrm{s}}\Pi_1^{N-1}X_0^{\mathrm{q}} & R_N^{\mathrm{s}}\Pi_2^{N-1}X_1^{\mathrm{q}} & \cdots & R_N^{\mathrm{s}}X_{N-1}^{\mathrm{q}} & & \end{array}\right) \tag{26}$$

is obtained, with sensitivity matrix products Π_j^k defined to be

$$\Pi_j^k := \prod_{l=j}^{k} X_l^{\mathrm{s}}, \quad 0 \le j \le k \le N-1. \tag{27}$$

From (26) we deduce that, after this elimination step, the transformed QP in terms of the two unknowns Δv_1 and Δv_2 reads

$$\min_{\Delta v} \quad \frac{1}{2}\begin{pmatrix} \Delta v_1 \\ \Delta v_2 \end{pmatrix}^T \begin{pmatrix} \overline{B}_{11} & \overline{B}_{12} \\ \overline{B}_{12}^T & \overline{B}_{22} \end{pmatrix} \begin{pmatrix} \Delta v_1 \\ \Delta v_2 \end{pmatrix} + \begin{pmatrix} \overline{\Phi}_1 \\ \overline{\Phi}_2 \end{pmatrix}^T \begin{pmatrix} \Delta v_1 \\ \Delta v_2 \end{pmatrix} \tag{28a}$$

$$\text{s.t.} \quad 0 = \overline{X}\Delta v_1 - \Delta v_2 - \overline{h} \tag{28b}$$

$$0 \leqq \overline{R}\Delta v_1 - \overline{r} \tag{28c}$$

wherein $\overline{B}$ and $\overline{\Phi}$ are reorderings of B and Φ, and $\overline{h}$ and $\overline{r}$ are appropriate right hand side vectors obtained by applying the Gaussian elimination steps to h and r.

Reduction to a Single Shooting Sized System

System (28) lends itself to the elimination of the unknown Δv_2. By this step we arrive at the final *condensed QP*

$$\min_{\Delta v_1} \quad \tfrac{1}{2}\Delta v_1^T \overline{\overline{B}}\Delta v_1 + \overline{\overline{\Phi}}^T \Delta v_1 \tag{29a}$$

$$\text{s.t.} \quad 0 \le \overline{R}\Delta v_1 - \overline{r} \tag{29b}$$

with the following dense Hessian matrix and gradient obtained from substitution of Δv_2 in the objective (28a)

$$\overline{\overline{B}} = \overline{B}_{11} + \overline{B}_{12}\overline{X} + \overline{X}^T\overline{B}_{12}^T + \overline{X}^T\overline{B}_{22}\overline{X}, \tag{30a}$$

$$\overline{\overline{\Phi}} = \overline{\Phi}_1 + \overline{X}^T\overline{\Phi}_2 - \overline{B}_{12}^T\overline{h} - \overline{X}^T\overline{B}_{22}\overline{h}. \tag{30b}$$

The required matrix multiplications are easily laid out to exploit the block triangular structure of $\overline{X}$ and the block diagonal structure of B. In addition, from the elimination steps described in the previous two paragraphs one obtains relations that allow to recover $\Delta v_2 = (\Delta s_1, \ldots, \Delta s_N)$ from the solution $\Delta v_1 = (\Delta s_0, \Delta q_0, \ldots, \Delta q_{N-1})$ of the condensed QP (29).

Solving the Condensed Quadratic Problem

The resulting condensed QP (29) no longer has a multiple shooting specific structure. It may thus be solved using any standard dense active–set method, which is what condensing ultimately aims for. Popular codes are the null space method `QPSOL` and its successor `QPOPT` [15]. The code `BQPD` [13] is even able to exploit remaining sparsity to some extent. An efficient code for parametric quadratic programming is `qpOASES` [12].

Condensing in Model–Predictive Control

The run time complexity of the condensing preprocessing step is $O(N^2)$ due to the elimination in (26). As all controls remain in the condensed QP, from which all states additionally introduced in section 1.3 are eliminated, condensing is a computationally favourable approach for model predictive control problems with a large number n_x of system states, few control parameters, and a limited number N of discretization points of the prediction horizon. The majority of condensing can be carried out in the preparation phase, cf. section 2.2, as the initial value x_0^new need not be known in advance. This reduces the control feedback delay to essentially the run time of the QP solver on the condensed QP (29).

4.3 Riccati Recursion

While the condensing algorithm acts as a preprocessing step on the block structured QP data, an alternative approach is to exploit this structure inside the QP solver, i.e. to solve block structured KKT systems. Riccati recursion, based on the dynamic programming principle, is a popular concept here. Starting with the last shooting node's cost function

$$\phi_N(\Delta s_N) = \tfrac{1}{2}\Delta s_N^T B_N \Delta s_N + \Phi_N^T \Delta s_N \tag{31}$$

the cost–to–go function ϕ_{N-1} of the previous node is found from tabulation of the optimal control step Δq_{N-1} for each admissible state step Δs_{N-1}. This procedure is repeated until the backwards recursion arrives at the first node $i = 0$, at which point the sequence of optimal control steps Δq can simply be obtained from a table look–up using the estimated or measured initial value x_0^new.

The optimal control steps Δq_i of nodes $i = N-1, \ldots, 0$ are found by solving the purely equality-constrained QP (32) using an appropriate factorization of the associated KKT system,

$$\phi_i(\Delta s_i) = \min_{\substack{\Delta s_{i+1} \\ \Delta q_i}} \frac{1}{2} \begin{pmatrix} \Delta s_i \\ \Delta q_i \end{pmatrix}^T \begin{pmatrix} B_i^\mathrm{ss} & B_i^\mathrm{sq} \\ B_i^\mathrm{qs} & B_i^\mathrm{qq} \end{pmatrix} \begin{pmatrix} \Delta s_i \\ \Delta q_i \end{pmatrix} + \begin{pmatrix} \Phi_i^\mathrm{s} \\ \Phi_i^\mathrm{q} \end{pmatrix}^T \begin{pmatrix} \Delta s_i \\ \Delta q_i \end{pmatrix} + \phi_{i+1}(\Delta s_{i+1}) \tag{32a}$$

$$\text{s.t.} \quad h_i = \begin{pmatrix} X_i^\mathrm{s} & X_i^\mathrm{q} \end{pmatrix} \begin{pmatrix} \Delta s_i \\ \Delta q_i \end{pmatrix} - \Delta s_{i+1}. \tag{32b}$$

Riccati Recursion in Model–Predictive Control

By observing that the optimal cost–to–go functions $\phi_i(\Delta s_i)$ remain quadratic functions,

$$\phi_i(\Delta s_i) = \Delta s_i^T P_i \Delta s_i + p_i^T \Delta s_i + \pi_i, \tag{33}$$

and by inserting (32b) into (32a), the unknown Δs_{i+1} can be eliminated. Problem (32) then becomes an unconstrained minimization problem,

$$\phi_i(\Delta s_i) = \min_{\Delta q_i} \frac{1}{2} \begin{pmatrix} \Delta s_i \\ \Delta q_i \end{pmatrix}^T \begin{pmatrix} B_i^{\mathrm{ss}} + X_i^{\mathrm{s}T} P_{i+1} X_i^{\mathrm{s}} & B_i^{\mathrm{sq}} + X_i^{\mathrm{s}T} P_{i+1} X_i^{\mathrm{q}} \\ B_i^{\mathrm{qs}} + X_i^{\mathrm{q}T} P_{i+1} X_i^{\mathrm{s}} & B_i^{\mathrm{qq}} + X_i^{\mathrm{q}T} P_{i+1} X_i^{\mathrm{q}} \end{pmatrix} \begin{pmatrix} \Delta s_i \\ \Delta q_i \end{pmatrix} \tag{34}$$

$$+ \begin{pmatrix} \Phi_i^{\mathrm{s}} - X_i^{\mathrm{s}T} P_{i+1} h_i + X_i^{\mathrm{s}T} p_{i+1} \\ \Phi_i^{\mathrm{q}} - X_i^{\mathrm{q}T} P_{i+1} h_i + X_i^{\mathrm{q}T} p_{i+1} \end{pmatrix}^T \begin{pmatrix} \Delta s_i \\ \Delta q_i \end{pmatrix} \tag{35}$$

$$+ h_i^T P_{i+1} h_i - p_{i+1}^T h_i + \pi_{i+1} \tag{36}$$

From this, an explicit expression for the optimal Δq_i is easily obtained. Inserting it into the cost–to–go function finally allows for the direct computation of $\phi_i(\Delta s_i)$. The backwards recursion can thus be started with $P_N = B_N$, $p_N = \Phi_n$, $\pi_n = 0$, and carried out without knowledge of the true system state x_0^{new}. After the backward sweep has been completed, the feedback control step is available as

$$\Delta q_0 = K_0(x_0 - x_0^{\mathrm{new}}) + k_0 \tag{37}$$

with an $n_{\mathrm{q}} \times n_{\mathrm{x}}$ matrix K_0 and an n_{q}-vector k_0 obtained from the backward sweep eliminations. The feedback delay is as small as the time required for a matrix–vector–multiplication with K_0. A forward recursion starting with the known initial value $\Delta s_0 = x_0^{\mathrm{new}} - s_0$ is employed afterwards to recover the steps $\Delta q_1, \ldots, \Delta q_{N-1}$ and Δs which are not needed for the immediate control feedback.

Inequality Constraints

The applicability of Riccati recursion is restricted to purely equality constrained systems, i.e. KKT systems or QPs with only equality constraints. In order to treat inequality constraints, a Riccati recursion based KKT solver can be employed inside an active set method. The performance of such Riccati recursion based active set solvers suffers from the $O(Nn^3)$ runtime complexity of the KKT system solution, and the approach is thus more popular for interior point methods [18, 28, 30], where it has been successfully used in place of symmetric indefinite factorizations.

4.4 Block Structured Active Set Methods

A third possibility of solving QP (24) is to employ a block structured factorization of the KKT system inside an active set method. For efficiency, matrix updates for such factorizations should be available.

Block Structured Factorization

We present here a block structured factorization due to [19, 30] that is composed from step-wise reductions of the KKT matrix (38), in which all matrices and vectors are understood as restrictions onto the current active set,

$$\begin{pmatrix} B_0 & R_0^T & X_0^T & & & & & \\ R_0 & & & & & & & \\ X_0 & & & P_1 & & & & \\ & & P_1^T & B_1 & R_1^T & X_1^T & & \\ & & & R_1 & & & & \\ & & & X_1 & & & & \\ & & & & & & \ddots & \\ & & & & & & B_N & R_N^T \\ & & & & & & R_N & \end{pmatrix} \tag{38}$$

Similar to Ricatti recursion, the idea is to factorize this KKT matrix and exploit the inherent block structure while avoiding any fill–in.

Matrix Updates

Contrary to Ricatti recursion, though, we desire to derive a factorization that opens up the possibility of applying *matrix update* techniques, cf. [14, 16]. These allow to recover the factorization of the KKT matrix after an active set exchange in $O(n^2)$ time, while computing a factorization anew usually requires $O(n^3)$ time. An example is the Schur complement based dual active set method presented in [29, 3]. A factorization tailored to the block structure (38) that is based on a hybrid null–space range–space is given in [19]. Suitable updates are derived in [20], based on techniques by [14].

Runtime Complexity

This approach has $O(N)$ runtime complexity compared to $O(N^2)$ for the condensing approach of section 4.2, and is therefore suited problems that require longer prediction horizons or finer discretizations of the prediction horizon. As the factorization eliminates all controls from the system in the first step, problems with limited state dimension but with a large number of controls, e.g. in mixed–integer predictive control [21] or in online optimal experimental design, will benefit from this approach. Compared to the $O(n^3)$ runtime complexity of Riccati recursion techniques, the availability of matrix updates reduces the runtime complexity for all but the first iteration of the active set loop to $O(n^2)$.

5 Summary

We reviewed a collection of state–of–the–art numerical methods for efficient NMPC of nonlinear dynamic processes in ODE and DAE systems under real–time conditions. Our presentation started with a presentation of the discussed problem class and a brief introduction to direct multiple shooting for the discretization of the optimal control problem. We focussed on a Newton–type framework for the solution of the resulting nonlinear problem, relying on active set based methods. In combination with initial value embedding, the real–time iteration scheme provides an efficient first order tangential predictor of the optimal feedback control. A multi–level scheme featuring at least four distinct modes that provide adaptive updates to selected components of the quadratic subproblem is presented, and we mentioned theoretical results as well as computational effort of the different modes. Connections to emerging ideas such as parametric quadratic programming, Euler feedback steps, and the computation of the local linear feedback law providing microsecond control feedback opportunities are shown. The efficient solution of the arising quadratic subproblems is the core of all active–set based NMPC algorithms. Here we introduced the block structure that is due to direct multiple shooting, and reviewed the condensing preprocessing step as well as a Riccati recursion scheme. Both exploit the exhibited block structure, but also left room for improvements. Our survey concluded with mentioning block structured active set methods. These require matrix updates tailored to the block structure, but are able to reduce the run time of an active set iteration to an unmatched complexity of $O(Nn^2)$.

References

1. J. Albersmeyer, D. Beigel, C. Kirches, L. Wirsching, H. Bock, and J. Schlöder. Fast nonlinear model predictive control with an application in automotive engineering. In L. Magni, D. Raimondo, and F. Allgöwer, editors, *Lecture Notes in Control and Information Sciences*, volume 384, pages 471–480. Springer Verlag Berlin Heidelberg, 2009.
2. J. Albersmeyer and H. Bock. Sensitivity Generation in an Adaptive BDF-Method. In H. G. Bock, E. Kostina, X. Phu, and R. Rannacher, editors, *Modeling, Simulation and Optimization of Complex Processes: Proceedings of the International Conference on High Performance Scientific Computing, March 6–10, 2006, Hanoi, Vietnam*, pages 15–24. Springer Verlag Berlin Heidelberg New York, 2008.
3. R. Bartlett and L. Biegler. QPSchur: A dual, active set, schur complement method for large-scale and structured convex quadratic programming algorithm. *Optimization and Engineering*, 7:5–32, 2006.
4. M. Best. *An Algorithm for the Solution of the Parametric Quadratic Programming Problem*, chapter 3, pages 57–76. Applied Mathematics and Parallel Computing. Physica-Verlag, Heidelberg, 1996.

5. H. Bock, M. Diehl, E. Kostina, and J. Schlöder. Constrained Optimal Feedback Control for DAE. In L. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, and B. van Bloemen Waanders, editors, *Real-Time PDE-Constrained Optimization*, chapter 1, pages 3–24. SIAM, 2007.
6. H. Bock, M. Diehl, P. Kühl, E. Kostina, J. Schlëder, and L. Wirsching. Numerical methods for efficient and fast nonlinear model predictive control. In R. Findeisen, F. Allgöwer, and L. T. Biegler, editors, *Assessment and future directions of Nonlinear Model Predictive Control*, volume 358 of *Lecture Notes in Control and Information Sciences*, pages 163–179. Springer, 2005.
7. H. Bock and K. Plitt. A Multiple Shooting algorithm for direct solution of optimal control problems. In *Proceedings of the 9th IFAC World Congress*, pages 243–247, Budapest, 1984. Pergamon Press. Available at http://www.iwr.uni-heidelberg.de/groups/agbock/FILES/Bock1984.pdf.
8. A. Bryson and Y.-C. Ho. *Applied Optimal Control*. Wiley, New York, 1975.
9. M. Diehl, H. Bock, J. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *J. Proc. Contr.*, 12(4):577–585, 2002.
10. M. Diehl, H. Ferreau, and N. Haverbeke. Efficient numerical methods for nonlinear mpc and moving horizon estimation. In L. Magni, D. Raimondo, and F. Allgöwer, editors, *Nonlinear Model Predictive Control*, volume 384 of *Springer Lecture Notes in Control and Information Sciences*, pages 391–417. Springer-Verlag, Berlin, Heidelberg, New York, 2009.
11. M. Diehl, P. Kuehl, H. Bock, and J. Schlöder. Schnelle Algorithmen für die Zustands- und Parameterschätzung auf bewegten Horizonten. *Automatisierungstechnik*, 54(12):602–613, 2006.
12. H. Ferreau, H. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control*, 18(8):816–830, 2008.
13. R. Fletcher. Resolving degeneracy in quadratic programming. Numerical Analysis Report NA/135, University of Dundee, Dundee, Scotland, 1991.
14. P. Gill, G. Golub, W. Murray, and M. A. Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535, 1974.
15. P. Gill, W. Murray, and M. Saunders. *User's Guide For QPOPT 1.0: A Fortran Package For Quadratic Programming*, 1995.
16. G. Golub and C. van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 3rd edition, 1996.
17. A. Griewank. *Evaluating Derivatives, Principles and Techniques of Algorithmic Differentiation*. Number 19 in Frontiers in Appl. Math. SIAM, Philadelphia, 2000.
18. N. Haverbeke, M. Diehl, and B. de Moor. A structure exploiting interior-point method for moving horizon estimation. In *Proceedings of the 48th IEEE Conference on Decision and Control (CDC09)*, pages 1–6, 2009.
19. C. Kirches, H. Bock, J. Schlöder, and S. Sager. Block structured quadratic programming for the direct multiple shooting method for optimal control. *Optimization Methods and Software*, 2010. DOI 10.1080/10556781003623891.
20. C. Kirches, H. Bock, J. Schlöder, and S. Sager. A factorization with update procedures for a KKT matrix arising in direct optimal control. *Mathematical Programming Computation*, 2010. (submitted). Available Online: http://www.optimization-online.org/DB_HTML/2009/11/2456.html.

21. C. Kirches, S. Sager, H. Bock, and J. Schlöder. Time-optimal control of automobile test drives with gear shifts. *Optimal Control Applications and Methods*, 2010. DOI 10.1002/oca.892.
22. D. Leineweber. *Efficient reduced SQP methods for the optimization of chemical processes described by large sparse DAE models*, volume 613 of *Fortschritt-Berichte VDI Reihe 3, Verfahrenstechnik*. VDI Verlag, Düsseldorf, 1999.
23. D. Leineweber, I. Bauer, A. Schäfer, H. Bock, and J. Schlöder. An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization (Parts I and II). *Computers and Chemical Engineering*, 27:157–174, 2003.
24. D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: stability and optimality. *Automatica*, 26(6):789–814, 2000.
25. L. Petzold, S. Li, Y. Cao, and R. Serban. Sensitivity analysis of differential-algebraic equations and partial differential equations. *Computers and Chemical Engineering*, 30:1553–1559, 2006.
26. K. Plitt. Ein superlinear konvergentes Mehrzielverfahren zur direkten Berechnung beschränkter optimaler Steuerungen. Diploma thesis, Rheinische Friedrich–Wilhelms–Universität zu Bonn, 1981.
27. C. V. Rao, J. B. Rawlings, and D. Q. Mayne. Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations. *IEEE Transactions on Automatic Control*, 48(2):246–258, 2003.
28. C. Rao, S. Wright, and J. Rawlings. Application of interior-point methods to model predictive control. *Journal of Optimization Theory and Applications*, 99:723–757, 1998.
29. C. Schmid and L. Biegler. Quadratic programming methods for tailored reduced Hessian SQP. *Computers & Chemical Engineering*, 18(9):817–832, September 1994.
30. M. Steinbach. Structured interior point SQP methods in optimal control. *Zeitschrift für Angewandte Mathematik und Mechanik*, 76(S3):59–62, 1996.
31. L. Wirsching. An SQP algorithm with inexact derivatives for a direct multiple shooting method for optimal control problems. Diploma thesis, Universität Heidelberg, 2006.