Fast Nonlinear Model Predictive Control Algorithms and Applications in Process Engineering

Moritz Diehl,
Optimization in Engineering Center (OPTEC) & Electrical Engineering Department (ESAT)
K.U. Leuven,
Belgium

INRIA-Rocquencourt,
May 30, 2007
Outline of the Talk

- K.U.Leuven’s Optimization in Engineering Center OPTEC
- Nonlinear Model Predictive Control (NMPC)
- How to solve dynamic optimization problems?
- Four crucial features for fast NMPC algorithms
- Application to a Distillation Column
OPTEC Aim: Connect Optimization Methods & Applications

Applications: Smart problem formulations allow efficient solution (e.g. convexity)

Methods: New developments are inspired and driven by application needs

$$\min_{x_0, \ldots, x_N} \sum_{i=0}^{N-1} (x_i^T Q r_i + a_i^T R u_i)$$

s.t.

$$x_{k+1} = Ax_k + Bu_k,$$

$$x_0 \text{ given},$$

$$c \leq Cx_k \leq \bar{c},$$

$$d \leq D u_k \leq \bar{d},$$

$$e_T \leq C_T x_N.$$
Optimization in Engineering Center OPTEC

**Five year project, from 2005 to 2010,**
500,000 Euro per year, about 20 professors, 10 postdocs, and 60 PhD students involved in OPTEC research

**Promoted by four departments:**
- Electrical Engineering
- Mechanical Engineering
- Chemical Engineering
- Computer Science

Many real world applications at OPTEC...
Quarterly Stevin Lecture: Everyone Invited!

Quarterly „Simon Stevin Lecture on Optimization in Engineering“:

- Dec 6: Larry Biegler, CMU Pittsburgh
- Apr 18: Stephen Boyd, Stanford
- July 9: Steve Wright, Madison, Wisconsin
- Oct 24: Manfred Morari, ETH Zurich
- Dec X: David Mayne, Imperial, London

Lecture and following Reception in Arenberg Castle, Leuven

Simon Stevin, 1548-1620)
Outline of the Talk

- K.U.Leuven‘s Optimization in Engineering Center OPTEC
- Nonlinear Model Predictive Control (NMPC)
- How to solve dynamic optimization problems?
- Four crucial features for fast NMPC algorithms
- Online MPC of a combustion engine
First Principle Dynamic System Models

**E.g. some equations modelling a distillation column (in Stuttgart)**

\[
\begin{align*}
\dot{n}_{N+1} &= V_N - D - L_{N+1}, \\
0 &= \dot{n}_c^m = V^m(X_{\ell}, T_{\ell})\dot{n}_{\ell} + \frac{\partial V^m}{\partial (X, T)}(\dot{X}_{\ell}, \dot{T}_{\ell})^T n_{\ell}, \\
L_{\text{vol}} &= V^m(X_{N+1}, T_C)L_{N+1}, \\
\dot{X}_0 n_0 + X_0 \dot{n}_0 &= -V_0 Y_0 + L_1 X_1 - BX_0, \\
\dot{X}_{\ell} n_{\ell} + X_{\ell} \dot{n}_{\ell} &= V_{\ell-1} Y_{\ell-1} - V_{\ell} Y_{\ell} + L_{\ell+1} X_{\ell+1} - L_{\ell} X_{\ell} \\
&\quad \ell = 1, 2, \ldots, N_F - 1, N_F + 1, \ldots, N \\
\dot{k}_{N_F} n_{N_F} + X_{N_F} \dot{n}_{N_F} &= V_{N_F-1} Y_{N_F-1} - V_{N_F} Y_{N_F} \\
&\quad + L_{N_F+1} X_{N_F+1} - L_{N_F} X_{N_F} + FX_F,
\end{align*}
\]

- Nonlinear differential algebraic equations (DAE)
- often in modeling languages like gPROMS, SIMULINK, Modelica
- typical order of magnitude: some hundreds to thousands variables
- difficulties: stiffness, discontinuities, high index

Can we use these models directly for optimization and feedback control?
Nonlinear Model Predictive Control (NMPC)

1. Estimate current system state $x_0$ (and parameters) from measurements.

2. Solve *in real-time* an optimal control problem:

$$\min_{x,z,u} \int_{t_0}^{t_0+T_p} L(x,z,u) dt + E(x(t_0+T_p)) \text{ s.t.}$$

$$x(t_0) - x_0 = 0,$$
$$\dot{x} - f(x,z,u) = 0, \quad t \in [t_0, t_0+T_p]$$
$$g(x,z,u) = 0, \quad t \in [t_0, t_0+T_p]$$
$$h(x,z,u) \geq 0, \quad t \in [t_0, t_0+T_p]$$
$$r(x(t_0+T_p)) \geq 0.$$

3. Implement first control $u_0$ for time $\delta$ at real plant. Set $t_0 = t_0 + \delta$ and go to 1.
Nonlinear Model Predictive Control When We Drive a Car

Always look a bit into the future!

Brain predicts and optimizes: e.g. slow down before curve

Main challenge for NMPC: fast and reliable real-time optimization!
NMPC applications, with decreasing timescales

Distillation column (with Univ. Stuttgart)

Polymerisation reactor (with BASF)

Combined Cycle Power Plant (with Univ. Pavia)

Chromatographic Separation (with Univ. Dortmund)

PET plant: Plant wide control project with Politecnico di Milano

Looping kites for power generation, with TU Delft, Politecnico di Torino

Robot arms (with Columbia Univ. & INRIA Grenoble)

Car Engines: EU Project with Univ. Linz, Stuttgart, Politecnico di Milano
Outline of the Talk

- K.U.Leuven‘s Optimization in Engineering Center OPTEC
- Nonlinear Model Predictive Control (NMPC)

**How to solve dynamic optimization problems?**

- Four crucial features for fast NMPC algorithms
- Application to a Distillation Column
Optimal Control Family Tree

Hamilton-Jacobi-Bellman Equation:
Tabulation in State Space

Indirect Methods, Pontryagin:
Solve Boundary Value Problem

Direct Methods:
Transform into Nonlinear Program (NLP)

Single Shooting:
Only discretized controls in NLP (sequential)

Collocation:
Discretized controls and states in NLP (simultaneous)

Multiple Shooting:
Controls and node start values in NLP (simultaneous)
Optimal Control Family Tree

Hamilton-Jacobi-Bellman Equation:
Tabulation in State Space

Indirect Methods, Pontryagin:
Solve Boundary Value Problem

Direct Methods:
Transform into Nonlinear Program (NLP)

Single Shooting:
Only discretized controls in NLP (sequential)

Collocation:
Discretized controls and states in NLP (simultaneous)

Multiple Shooting:
Controls and node start values in NLP (simultaneous)
Optimal Control Family Tree

Hamilton-Jacobi-Bellman Equation:
- Tabulation in State Space

Indirect Methods, Pontryagin:
- Solve Boundary Value Problem

Direct Methods:
- Transform into Nonlinear Program (NLP)

Single Shooting:
- Only discretized controls in NLP (sequential)

Collocation:
- Discretized controls and states in NLP (simultaneous)

Multiple Shooting:
- Controls and node start values in NLP (simultaneous)
Optimal Control Problem in Simplest Form

minimize $\int_0^T L(x(t), u(t)) \, dt + E(x(T))$

subject to

$x(0) - x_0 = 0,$  \hspace{1cm} \text{(fixed initial value)}

$\dot{x}(t) - f(x(t), u(t)) = 0,$  \hspace{1cm} $t \in [0, T], \text{ (ODE model)}$

$h(x(t), u(t)) \geq 0,$  \hspace{1cm} $t \in [0, T], \text{ (path constraints)}$

$r(x(T)) \geq 0$  \hspace{1cm} \text{(terminal constraints)}.$
Simplest Approach: Single Shooting

Discretize controls $u(t)$ on fixed grid

$$0 = t_0 < t_1 < \ldots < t_N = T,$$

regard states $x(t)$ on $[0, T]$ as dependent variables.

Use numerical integration routine to obtain state as function $x(t;q)$ of finitely many control parameters

$$q = (q_0, q_1, \ldots, q_{N-1})$$
Nonlinear Program (NLP) in Single Shooting

- After control discretization, obtain NLP:

\[
\begin{align*}
\text{minimize} & \quad \int_0^T L(x(t; q), u(t; q)) \, dt + E(x(T; q)) \\
\text{subject to} & \\
& h(x(t_i; q), u(t_i; q)) \geq 0, \quad i = 0, \ldots, N, \quad \text{(discretized path constraints)} \\
& r(x(T; q)) \geq 0. \quad \text{(terminal constraints)}
\end{align*}
\]

- Solve with NLP solver, e.g. Sequential Quadratic Programming (SQP)
Sequential Quadratic Programming (SQP)

Summarize problem as

\[
\min_{q} F(q) \quad \text{s.t.} \quad H(q) \geq 0.
\]

Solve iteratively, start with guess \( q^0 \) for controls. Set \( k = 0 \).

1. Evaluate \( F(q^k), H(q^k) \) and derivatives (ODE solution).
   Obtain “Hessian matrix” \( A^k \) e.g. by updates.

2. Compute \( \Delta q^k \) that solves \textbf{Quadratic Program (QP)}:

\[
\min_{\Delta q} \nabla F(q^k)^T \Delta q + \frac{1}{2} \Delta q^T A^k \Delta q \quad \text{s.t.} \quad H(q^k) + \nabla H(q^k)^T \Delta q \geq 0.
\]

3. Perform step

\[
q^{k+1} = q^k + \alpha_k \Delta q^k
\]

with step length \( \alpha_k \in [0, 1] \) determined e.g. by line search. \( k = k + 1 \).
Toy Problem with One ODE for Illustration

minimize \int_0^3 (x(t)^2 + u(t)^2) \, dt

subject to

\begin{align*}
  x(0) &= x_0, & \text{(initial value)} \\
  \dot{x} &= (1 + x)x + u, & t \in [0, 3], & \text{(ODE model)} \\
  \begin{pmatrix}
    1 - x(t) \\
    1 + x(t) \\
    1 - u(t) \\
    1 + u(t)
  \end{pmatrix} & \geq
  \begin{pmatrix}
    0 \\
    0 \\
    0 \\
    0
  \end{pmatrix}, & t \in [0, 3], & \text{(bounds)} \\
  x(3) &= 0. & \text{(zero terminal constraint)}.
\end{align*}

Mildly nonlinear and unstable system.
Single Shooting

- Choose $N = 30$ equal control intervals.
- Initialize with steady state controls $u(t) \equiv 0$. 
Single Shooting: First Iteration
Single Shooting: Second Iteration

\[ u(t) \]

\[ x(t) \]

Graphs showing the evolution of \( u(t) \) and \( x(t) \) over time, with axes for time \( t \) and values ranging from \(-1\) to \(1\).
Single Shooting: Third Iteration
Single Shooting: 4th Iteration
Single Shooting: 5th Iteration
Single Shooting: 7th Iteration (Solution)
Single Shooting: Pros and Cons

+ Can use state-of-the-art ODE/DAE solvers.
+ Few degrees of freedom even for large ODE/DAE systems.
+ Active set changes easily treated.
+ Need only initial guess for controls $q$.
- Cannot use knowledge of $x$ in initialization (e.g. in tracking problems).
- ODE solution $x(t; q)$ can depend very nonlinearly on $q$.
- Unstable systems difficult to treat.
Alternative: Direct Multiple Shooting [Bock, Plitt 1981]

- Discretize controls piecewise on a coarse grid
  \[ u(t) = q_i \quad \text{for} \quad t \in [t_i, t_{i+1}] \]

- Solve ODE on each interval \([t_i, t_{i+1}]\) numerically, starting with artificial initial value \(s_i\):
  \[
  \begin{align*}
  \dot{x}_i(t; s_i, q_i) &= f(x_i(t; s_i, q_i), q_i), \quad t \in [t_i, t_{i+1}], \\
  x_i(t_i; s_i, q_i) &= s_i.
  \end{align*}
  \]

  Obtain trajectory pieces \(x_i(t; s_i, q_i)\).

- Also compute integrals
  \[
  l_i(s_i, q_i) := \int_{t_i}^{t_{i+1}} L(x_i(t; s_i, q_i), q_i) \, dt
  \]
Nonlinear Program in Multiple Shooting

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=0}^{N-1} l_i(s_i, q_i) + E(s_N) \\
\text{subject to} & \quad s_0 - x_0 = 0, \quad \text{(initial value)} \\
& \quad s_{i+1} - x_i(t_{i+1}; s_i, q_i) = 0, \quad i = 0, \ldots, N-1, \quad \text{(continuity)} \\
& \quad h(s_i, q_i) \geq 0, \quad i = 0, \ldots, N, \quad \text{(discretized path constraints)} \\
& \quad r(s_N) \geq 0. \quad \text{(terminal constraints)}
\end{align*}
\]
Summarize NLP:

\[
\min_{w} F(w) \quad \text{s.t.} \quad \begin{cases} 
G(w) & = 0, \\
H(w) & \geq 0.
\end{cases}
\]

- Summarize all variables as \( w := (s_0, q_0, s_1, q_1, \ldots, s_N) \).
- In each iteration, solve Quadratic Program:

\[
\min_{\Delta w} \nabla F(w^k)^T \Delta w + \frac{1}{2} \Delta w^T A^k \Delta w \quad \text{s.t.} \quad \begin{cases} 
G(w^k) + \nabla G(w^k)^T \Delta w & = 0 \\
H(w^k) + \nabla H(w^k)^T \Delta w & \geq 0.
\end{cases}
\]

- Jacobians \( \nabla G(w^k)^T \), \( \nabla H(w^k)^T \) and Hessian \( A^k \) are block sparse.
Toy Example: Multiple Shooting Initialization
Multiple Shooting: First Iteration

\[ u(t) \]

\[ x_0(t) \]

\[ t \]

\[ 0 \quad 1 \quad 2 \quad 3 \]
Multiple Shooting: Second Iteration
Multiple Shooting: 3\textsuperscript{rd} Iteration (already solution!)
Multiple Shooting: 3\textsuperscript{rd} Iteration (already solution!)

Single shooting converged much slower!
Why Direct Multiple Shooting?

- uses **adaptive** DAE solvers
- but NLP has **fixed dimensions**
- treats **nonlinear, stiff, and unstable** systems well
- robust handling of **control and state constraints**
- easy to **parallelize**
- **multistage** optimal control problems in DAE can be treated in modular package MUSCOD-II (Leineweber, Schäfer, Diehl, Brandt-Pollmann, Sager, 1999-)
- Coupled to modeling languages like gPROMS, SIMULINK, also C, FORTRAN
The MUSCOD-II Developer Team [Heidelberg, Leuven, Madrid]
Outline of the Talk

- K.U.Leuven‘s Optimization in Engineering Center OPTEC
- Nonlinear Model Predictive Control (NMPC)
- How to solve dynamic optimization problems?
- **Four crucial features for fast NMPC algorithms**
- Online MPC of a combustion engine
NMPC Computation from 1998 to 2006

- **1998**: 5th order distillation model allows sampling times of only 5 minutes [Allgower, Findeisen, 1998]

- **2001**: 206th order distillation model, sampling times of 20 seconds [D. et al., 01]

\[ \frac{5 \times 60 \times 1000}{20} = 15000 \text{ times faster} \], due to Moore's law + Algorithm Development
1998: 5th order distillation model allows sampling times of only 5 minutes [Allgower, Findeisen, 1998]


2006: 5th order engine model, sampling times of 10-20 milliseconds [Ferreau et al. ‘06], [Albersmeyer, Findeisen ‘06]

\[ 5 \times 60 \times 1000 / 20 = 15 \,000 \, \text{times faster, due to Moore’s law + Algorithm Development} \]
Four Crucial Features for Fast NMPC

- **Direct, simultaneous** optimal control: Multiple Shooting
- Efficient *derivative generation* for ODE/DAE solvers
- Initialization by „*Initial Value Embedding*“
- *Real-Time Iterations* for fast tracking of optimal solutions
Example: Distillation Column (ISR, Stuttgart)

- Aim: to ensure product purity, keep two temperatures $(T_{14}, T_{28})$ constant despite disturbances

- least squares objective:
  \[
  \min_{t_0} \int_{t_0}^{t_0+T_p} \left\| \begin{array}{c}
  T_{14}(t) - T_{14}^{\text{ref}} \\
  T_{28}(t) - T_{28}^{\text{ref}}
  \end{array} \right\|_2^2 \, dt
  \]

- control horizon 10 min
- prediction horizon 10 h
- stiff DAE model with 82 differential and 122 algebraic state variables
- Desired sampling time: 30 seconds.
Distillation Online Scenario

- System is in steady state, optimizer predicts constant trajectory:

- **Suddenly**, system state $x_0$ is disturbed.
- What to do with optimizer?
Conventional Approach

- use offline method, e.g. MUSCOD-II with BFGS (Leineweber, 1999).
- initialize with \textbf{new} initial value \( x_0 \) and integrate system with \textbf{old} controls.
- iterate until convergence.

Initialization
Conventional Approach

- use offline method, e.g. MUSCOD-II with BFGS (Leineweber, 1999).
- initialize with **new** initial value $x_0$ and integrate system with **old** controls.
- iterate until convergence.

Initialization

<table>
<thead>
<tr>
<th>L</th>
<th>q</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Initialization Graph 1" /></td>
<td><img src="image2" alt="Initialization Graph 2" /></td>
</tr>
</tbody>
</table>

16th Iteration

<table>
<thead>
<tr>
<th>L</th>
<th>q</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="16th Iteration Graph 1" /></td>
<td><img src="image4" alt="16th Iteration Graph 2" /></td>
</tr>
</tbody>
</table>
Conventional Approach

- use offline method, e.g. MUSCOD-II with BFGS (Leinewebber, 1999).
- initialize with **new** initial value $x_0$ and integrate system with **old** controls.
- iterate until convergence.

**Initialization**

**16th Iteration**

**Solution (32nd Iteration)**
New Approach: Initial Value Embedding

- Initialize with old trajectory, accept violation of $s_0^x - x_0 = 0$
New Approach: Initial Value Embedding

- Initialize with **old** trajectory, accept violation of \( s_0^x - x_0 = 0 \)
New Approach: Initial Value Embedding

- Initialize with old trajectory, accept violation of \( s_x^0 - x_0 = 0 \)

Initialization  First Iteration  Solution (3rd Iteration)

First iteration nearly solution!
Never simulate a nonlinear system open-loop!

**Conventional:**

**Initial Value Embedding:**
Initial Value Embedding

- first iteration is tangential predictor for exact solution (for exact hessian SQP)
- also valid for active set changes
- derivative can be computed before $x_0$ is known: first iteration nearly without delay
Initial Value Embedding

- first iteration is tangential predictor for exact solution (for exact hessian SQP)
- also valid for active set changes
- derivative can be computed before $x_0$ is known: first iteration nearly without delay

Why wait until convergence and do nothing in the meantime?
Iterate, *while* problem is changing!

- tangential prediction after each change in $x_0$
- solution accuracy is increased with each iteration when $x_0$ changes little
- iterates stay close to solution manifold
Real-Time Iteration Algorithm:

1. **Preparation Step (costly):**
   Linearize system at current iterate, perform partial reduction and condensing of quadratic program.

2. **Feedback Step (short):**
   When new $x_0$ is known, solve condensed QP and implement control $u_0$ immediately.
   Complete SQP iteration. Go to 1.

- short cycle-duration (as **one** SQP iteration)
- negligible feedback delay ($\approx 1\%$ of cycle)
- nevertheless fully nonlinear optimization
Real-Time Iterations minimize feedback delay

\[ x_0(t_k) \]

\[ u_0(x_0(t_k)) \]

\[ t_{k-1} \rightarrow t_k \rightarrow t_{k+1} \]
Stability of System-Optimizer Dynamics?

- System and optimizer are coupled: can numerical errors grow and destabilize closed loop?
- Stability analysis combines concepts from both, **NMPC stability theory** and **convergence theory of Newton-type optimization**.
- Stability shown under mild assumptions (short sampling times, stable NMPC scheme) [Diehl, Findeisen, Allgöwer, 2005]
- Losses w.r.t. optimal feedback control are $O(\kappa^2\epsilon^2)$ after $\epsilon$ disturbance [Diehl, Bock, Schlöder, 2005]
Outline of the Talk

- K.U.Leuven’s Optimization in Engineering Center OPTEC
- Nonlinear Model Predictive Control (NMPC)
- How to solve dynamic optimization problems?
- Four crucial features for fast NMPC algorithms
- Application to a Distillation Column
Realization at Distillation Column

[D., Findeisen, Schwarzkopf, Uslu, Allgöwer, Bock, Schlöder, 2002]

- Parameter estimation using dynamic experiments
- Online state estimation with Extended Kalman Filter variant, using only 3 temperature measurements to infer all 82 system states
- Implementation of estimator and optimizer on Linux Workstation.
- Communication with Process Control System via FTP all 10 seconds.
- Self-synchronizing processes.
Computation Times During Application

**Preparation**

- Comp. time [s]
- Time [s]

**Feedback**

- Comp. time [ms]
- Time [s]
Experiments with a Real Distillation Column

Feedflow Change by 20%: Transient Phase (Comparison with PI-Controller)

Transient in 15 minutes instead of 2 hours!
Large Disturbance (Heating), then NMPC

- Overheating by manual control
- NMPC only starts at $t = 1500$ s
- PI-controller not implementable, as disturbance too large (valve saturation)
- NMPC: at start control bound active $\Rightarrow T_{28}$ rises further
- Disturbance attenuated after half an hour
Recent progress makes **Nonlinear** MPC with first principles models in millisecond range possible (now 15 000 x faster than 1998)

Emerging consensus for NMPC algorithms:
- employ direct, simultaneous methods
- use Initial Value Embedding (first order predictor)
- perform Real-Time Iterations to trace NMPC problem solution while data change
- Use SQP type method to track active set changes
13th Czech-French-German Conference on Optimization
Heidelberg, September 17-21, 2007

Topics:
Continuous Optimization (Smooth and Nonsmooth)
Numerical Methods for Mathematical Programming
Optimal Control and Calculus of Variations
Robust Optimization
Mixed Integer Optimization
Optimization with PDE
Differential Inclusions and Set-Valued Analysis
Stochastic Optimization
Multicriteria Optimization
Optimization Techniques for Industrial Applications

Confirmed Plenary Speakers:
Guillaume Carlier, Paris Dauphine
Roger Fletcher, University of Dundee
Roland Griese, Austrian Academy of Sciences
Pierre Maréchal, UPS Toulouse
Alexander Martin, TU Darmstadt
David Preiss, University College London
Carsten Scherrer, TU Delft
Zdenek Strakos, Czech Academy of Sciences
Emmanuel Trélat, Université d’Orléans
Michael Vašek, Czech Technical University, Prague
Luís Nunes Vicente, Universidade de Coimbra
Andreas Wächter, IBM, Yorktown Heights
Andrea Walther, TU Dresden

cfg07.uni-hd.de
4 PhD Positions in Numerical Optimization:

- Sequential Convex Programming Algorithms for Nonlinear SDP
- Large Scale & PDE Constrained Real-Time Optimization Algorithms
- Fast Model Predictive Control Applications in Mechatronic Systems
- Shape Optimization of Mechanical Parts under Inertia Loading

(deadline: June 21, 2007)