Dynamic Real-time Optimization

Jitendra Kadam, Wolfgang Marquardt, Martin Schlegel
Lehrstuhl für Prozesstechnik
RWTH Aachen

INCOOP Workshop
Düsseldorf, January 23 – 24, 2003
Operational strategies – the status

- plant in isolation
- steady-state operation
- limited flexibility
- disturbance handling
- largely autonomous

Production planning system

Operation support system

Corporate objectives

Supply chain

Market

Environment

... and its operation support system

Supplier

Customer

Purchasing & procurement

Marketing & sales

Dynamic real-time optimization
synchronous or asynchronous rolling window prediction adaptation

the plant as part of the supply chain

feedback

operational strategies reengineering the plant

performance indicators

the requirements

flexible production
smooth dynamics
lean inventories
high capital productivities
sustainable production
...
General operational objectives

- Optimal operation of chemical processes
- Constraints:
  - Equipment, safety, environment
  - Capacity, quality, reproducibility

Optimal profiles

- Manipulated variables
- State variables

Why should they be constant over time?

Process model:

\[ 0 = f(x, x, u, p, d) \]
\[ x(t_0) = x_0 \]
\[ y = g(x) \]
Optimization-based control

\[ F, h \]

measurements \[ \tilde{y}_r \] \rightarrow \text{optimal feedback control system} \rightarrow \text{manipulated variables} \[ u_c \]

2 coupled problems:

**dynamic data reconciliation**

\[
\min_{x_{r,0}, d_r} \Phi_r(y_r, \eta, x_{r,0}, d_r, t_c, t_f)
\]

s.t.

\[
0 = f(\dot{x}_r, x_r, u_r, d_r)
\]

\[
y_r = g(x_r)
\]

\[
x_r(t_f) = x_{r,0}
\]

\[
u_r = U(u_c(\cdot))
\]

\[
0 \geq h_r(x_r, d_r)
\]

\[
t \in [t_r, t_c]
\]

**optimal control**

\[
\min_{u_c} \Phi_c(x_c, u_c, t_c, t_f)
\]

s.t.

\[
0 = f(\dot{x}_c, x_c, u_c, d_c)
\]

\[
y_c = g(x_c)
\]

\[
x_c(t_c) = x_p(t_c)
\]

\[
d_c = D(d_r(\cdot))
\]

\[
0 \geq h_c(x_c, u_c)
\]

\[
t \in [t_c, t_f]
\]
Dynamic real-time optimization

Direct solution approach

- solution of optimal control reconciliation problems at controller sampling frequency
- computationally demanding
- model complexity limited (INCOOP benchmarks ⇒ large models!)
- lack of transparency, redundancy and reliability

( Terwiesch et al., 1994; Helbig et al., 1998; Wisnewski & Doyle, 1996; Biegler & Sentoni, 2000 )
Horizontal decomposition

- decentralization typically oriented at functional constituents of the plant
- coordination strategies enable approximation of "true" optimum
- not adequately covered in optimization-based control and operations yet

( Mesarovic et al., 1970; Findeisen et al., 1980; Morari et al., 1980; Lu, 2000 )
Dynamic real-time optimization

Vertical decomposition

optimizing feedback control system

- generalizes steady-state real-time optimization and constrained predictive control
- requires (multiple) time-scale separation, e.g. $d(t) = d_0(t) + \Delta d(t)$ with trend $d_0(t)$ zero mean fluctuation $\Delta d(t)$
Dynamic real-time optimization

INCOOP reference architecture

1. **Estimation**
   - Measurements
   - Process (including base control)

2. **D-RTO trigger**
   - State and disturbance estimates (fast)

3. **Dynamic Real Time Optimisation**
   - Optimal reference trajectories
   - Market and environmental conditions

4. **Model Predictive Control**
   - Control setpoints

**Dynamic optimization problems**

- **decision maker**
Goal: determine optimal transition trajectories using an economic objective function.

Challenges:
- Develop numerical solution methods which solve the problem robustly and sufficiently fast.
- Develop techniques for triggering a re-optimization based on external conditions.
- Implement software framework for enabling interaction with MPC and estimator.
A closer look on dynamic optimization

Mathematical problem formulation

\[
\begin{align*}
\min_{u(t), p, t_f} & \quad \Phi(x(t_f)) \\
\text{s.t.} & \quad \begin{aligned}
M \dot{x} &= F(x, u, p, t), & t \in [t_0, t_f], \\
0 &= x(t_0) - x_0, \\
0 &\geq P(x, u, p, t), & t \in [t_0, t_f], \\
0 &\geq E(x(t_f))
\end{aligned}
\end{align*}
\]

objective function (e.g. cost)

DAE system (process model)

path constraints (e.g. temp. bound)

endpoint constraints (e.g. prod. spec.)

Degrees of freedom:

\begin{align*}
\deg \text{ of freedom:} & \quad u(t) \quad \text{time-variant control variables} \\
p & \quad \text{time-invariant parameters} \\
t_f & \quad \text{final time}
\end{align*}
Example: Bayer Benchmark Process (I)

Dynamic real-time optimization

(From: Dünnebier & Klatt: Industrial challenges and requirements for optimization of polymerisation processes)
Polymerization process
- minimize time for load change
- three degrees of freedom
- path constraints on specifications

(From: Dünnebier & Klatt: Industrial challenges and requirements for optimization of polymerisation processes)
Solution approaches

Indirect solution methods
Necessary optimality conditions lead to multipoint boundary value problems:

- Highly accurate solutions with shooting techniques.
- Solution requires detailed a-priori knowledge of the optimal solution structure and appropriate estimates for adjoint variables.

Direct solution methods
Conversion of dynamic optimization problem into nonlinear programming problem (NLP) by discretization…

- …of state and control variables. (simultaneous methods, i.e. collocation, multiple shooting)
- …of control variables only. (sequential method, i.e. single shooting)

- Succesfully applied with large-scale process models
Solution algorithm

initial values: $u^0, p^0, t_f^0$
specify $\Phi$, $P$, $E$

Discretization

$z^0 = \{c^0, p^0, t_f^0\}$

Integration

$x^0(z^0) \Rightarrow \Phi^0, P^0, E^0$

$s^0(z^0) \Rightarrow \nabla\Phi^0, \nabla P^0, \nabla E^0$

QP solver

$\Phi^k, P^k, E^k + \nabla$

Convergence?

no

Line search

$z^k$

Integration

$x^k(z^k) \Rightarrow \Phi^k, P^k, E^k + \nabla$

Hessian update

yes

optimal solution: $c^*, p^*, t_f^*, \Phi^*, P^*, E^*$

optimal trajectories: $u(t), x(t)$

Potential improvements:

- most efficient integration
- as few NLP steps as possible

expensive! $> 90\%$ of CPU time
Sequential approach → single shooting

Control vector parameterization

\[ u_i(t) \approx \sum_{k \in \Lambda_i} c_{i,k} \phi_{i,k}(t) \]

Parameterization functions

\[ \phi_{i,k}(t) \]

Parameters

\[ c_{i,k} \]

⇒ Reformulation as nonlinear programming problem (NLP)

\[
\begin{align*}
\min_{c, p, t_f} \Phi(x(c, p, t_f)) \\
\text{s.t} \\
0 \geq P(x(c, p, t_i), \quad \forall t_i \in T, \\
0 \geq E(x(t_f))
\end{align*}
\]

DAE system solved by underlying numerical integration

• DAE system solved by underlying numerical integration
• Gradients for NLP solver typically obtained by integration of sensitivity systems

⇒ Numerical integration computationally most expensive (> 90 % of CPU time)
⇒ Computational effort strongly depends on size and complexity of process model
Algorithmic improvements – sequential approach

Sensitivity integration is expensive

Improve efficiency of sensitivity integration

New solver for sensitivity integration

State integration is expensive

Reduce number of sensitivity parameters

Control grid adaptation strategy

Reduce model complexity

Methods for model reduction
Dynamic optimization problem:

\[
\min_{z} \Phi(x, z, t)
\]

\[
s.t. \quad M \dot{x} = f(t, x, z)
\]

\[
0 = x(t_0) - x_0
\]

\[
0 \geq P(x, z, t)
\]

\[
0 \geq E(x(t_f))
\]

Typical solution approaches based on BDF-type integrators


New idea: Use one-step extrapolation method

- Based on LIMEX algorithm (Deuflhard et al. (1983,87))
- Extension for sensitivity computation: Schlegel et al. (2002)
Combined state and sensitivity system

\[ M \dot{x} = F(x, z, t) \]

\[ M \dot{s} = \left( \frac{\partial F}{\partial x} \right) s_i + \frac{\partial F}{\partial z_i} \quad i = 1, \ldots, n_z \]

\[ = A \]

\[ M \dot{X} = f(X, z, t) \]

with \( X = [x, s_1, \ldots, s_{n_z}]^T \)

Efficient solution of the combined system
- \( M \) is identical in both systems.
- \( A \) is already required for state integration.

Linear-implicit Euler discretization

\[ X_{k+1} = X_k - \left[ A - \frac{M}{h_j} \right] \]

\[ \begin{bmatrix} A_{11} & \cdots & A_{1n_z} \\ \vdots & \ddots & \vdots \\ A_{n_z 1} & \cdots & A_{n_z n_z} \end{bmatrix} \begin{bmatrix} f(X_k, z) \\ \vdots \\ f(X_k, z) \end{bmatrix} \]

Reuse LU decomposition

effect is off-diagonal elements.
Extrapolation algorithm for simultaneous state and sensitivity integration

Compute  \( A_0 = \frac{\partial}{\partial y}(f(y_0, p)) \)

for  \( j=1, \ldots, j_{max} \) while convergence criterion not satisfied

\( h_j = H / j \)

\( LU = A_0 - \frac{B}{h_j} \)

for  \( k=0, \ldots, j-1 \)

\( y_{k+1} = y_k - (LU)^{-1} f(y_k, p) \)

\( s_{i,k+1} = s_{i,k} - (LU)^{-1} \left( A(y_k)s_{i,k} + \frac{\partial f}{\partial p_i}(y_k) \right), \quad i = 1, \ldots, n_z \)

\( T_{j,1} = Y_j \)

if  \( j>1 \) compute  \( T_{jj} \) and check convergence

\( X_{new} = X_{j,j} \)

(here simplified for  \( M = \text{const.} \))
Results

Small example problem, solved for two different tolerances
tol = 1.e-5
tol = 1.e-7

⇒ One-step extrapolation faster than multistep BDF with increasing level of discretization

⇒ Used as standard for optimization of INCOOP benchmark problems
Algorithmic improvements – sequential approach

Sensitivity integration is expensive

- Improve efficiency of sensitivity integration
  - New solver for sensitivity integration

State integration is expensive

- Reduce number of sensitivity parameters
  - Reduce model complexity
    - Methods for model reduction

Control grid adaptation strategy

Dynamic real-time optimization
Different **representations** of the same function ...

... for problem discretization:

\[ u = \sum_{(j,k) \in \Lambda_y} c_{j,k} \phi_{j,k}(t) \]

... for grid point elimination analysis:

\[ u = c_{0,0} \phi_{0,0}(t) + \sum_{(j,k) \in \Psi} d_{j,k} \psi_{j,k}(t) \]
Adaptive refinement algorithm

Mesh analysis
- Concepts from signal analysis
- Grid point elimination
- Grid point insertion

Repetitive procedure
- Re-optimize problem on refined mesh
- Profile from previous solution as initial guess
- Decouple optimization and adaptation

Dynamic real-time optimization
Elimination of parameterization functions

- Problem specification
- Discretization
- NLP
- Warm start
- A-posteriori analysis
  - Elimination
  - Insertion

Analysis of wavelet coefficients of control variables

Minimize number of parameterization functions in $u^*$ such that:

$$\frac{\|u^* - \tilde{u}^*\|}{\|u^*\|} \leq \varepsilon$$

Approximation: Norm equivalence

$$\|u\|_{L_2} \approx \|d\|_{l_2}$$

Discarding small $d_{j,k}$ causes only small changes in approximate representation
Insertion of parameterization functions

**Problem specification**

**Discretization** → **NLP**

**Warm start**

**A-posteriori analysis**
- Elimination
- Insertion

**Termination criterion**

**Introduction of new parameterization functions where** $\| d_{j,k} \| \geq \eta$

Analysis of wavelet coefficients of control variables

Dynamic real-time optimization
Example: Batch reactive distillation

**Objective:**
Minimize energy demand with given:
- Fixed batch time
- Amount of distillate $D \geq 6.0$ kmol
- Product purity $x_D \geq 0.46$

**Controls:**
- Reflux ratio $R(t)$
- Vapor rate $V(t)$

**Dynamic model:**
- 10 theoretical trays
- gPROMS model contains 418 DAEs
  (63 differential equations)
Results: Batch reactive distillation
Adaptive approach (Binder et al., 2000):
- numerical lower for the adaptive refinement approach
- intermediate solutions are available
  - back-up values in real-time environment
  - direct employment on the process at early time
Dynamic optimization software DyOS (LPT)

Dynamic real-time optimization
Collocation on finite elements

Simultaneous approach $\rightarrow$ collocation (I)

Dynamic real-time optimization
Simultaneous approach → collocation (II)

Conversion into NLP problem yields

\[
\min \psi(z_i, y_{i,q}, u_{i,q}, p, t_f) \\
\text{s.t. } \left( \frac{dz}{dt} \right)_{i,j} = F \left( z_{i-1}, \frac{dz}{dt}_{i,j}, z_i, y_{i,j}, u_{i,j}, p \right) \\
0 = G \left( z_{i-1}, \frac{dz}{dt}_{i,j}, z_i, y_{i,j}, u_{i,j}, p \right) \\
z_i = f \left( \frac{dz}{dt}_{i-1,j}, z_{i-1} \right)
\]

\[
z_0^o = z(0) \\
z_i^l \leq z_i \leq z_i^u \\
y_{i,j}^l \leq y_{i,j} \leq y_{i,j}^u \\
u_{i,j}^l \leq u_{i,j} \leq u_{i,j}^u \\
p^l \leq p \leq p^u
\]

Requires specially tailored solution techniques:
- advanced interior-point solver
- filter-line search techniques (implemented as IPOPT, Biegler et al., 2001)

large-scale NLP problem

\[
\min \ f(x) \\
\text{s.t. } c(x) = 0 \\
x^L \leq x \leq x^u
\]
Barrier function formulation

original formulation

\[\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad f(x) \\
\text{s.t.} & \quad c(x) = 0 \\
& \quad x \geq 0
\end{align*}\]

barrier approach

\[\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad \phi_\mu(x) = f(x) - \mu \sum_{i=1}^{n} \ln s_i \\
\text{s.t.} & \quad c(x) = 0 \\
& \quad s - x = 0
\end{align*}\]

\[\Rightarrow \text{as } \mu \to 0, \quad x^*(\mu) \to x^*\]
Solution of the barrier problem (I)

⇒ Newton Directions (KKT System)

\[ \nabla f(x) + A(x)\lambda - \nu = 0 \\
SVe - \mu e = 0 \\
c(x) = 0 \\
s - x = 0 \]

⇒ solve primal-dual version

\[
\begin{bmatrix}
H & 0 & A & -I \\
0 & S^{-1}V & 0 & I \\
A^T & 0 & 0 & 0 \\
-I & I & 0 & 0
\end{bmatrix}
\begin{bmatrix}
d_x \\
d_s \\
d_\lambda \\
d_v
\end{bmatrix}
= -
\begin{bmatrix}
\nabla f + A\lambda - \nu \\
v - \mu S^{-1}e \\
c \\
0
\end{bmatrix}
\]
⇒ Range Space Step

\[ A^T d_x + c = 0 \]

⇒ \( d_R = -C^{-1} c \)

⇒ Null Space Step (reduced QP)

\[
\min_{d_Q} \left( Q^T \nabla \varphi_{\mu} + Q^T (H + \Sigma) R d_R \right)^T d_Q + \frac{1}{2} d_Q^T Q^T (H + \Sigma) Q d_Q
\]

\[
d_Q = -\left[ Q^T (H + \Sigma) Q \right]^{-1} \left( Q^T \nabla \varphi_{\mu} + Q^T (H + \Sigma) R d_R \right)
\]

reduced Hessian \hspace{2cm} cross term
Dynamic real-time optimization

Illustration of filter concept

- Optimality (RG)
- Feasibility

\((\theta, \varphi)_k\)  Member of filter set

Area of optimal solution
Dynamic optimization software DYNOPC/IPOPT (CMU)

- DYNOPC
  - initial integration
  - NLP solver IPOPT
  - initial trajectory
  - optimal trajectory

- model server (e.g. gPROMS)
  - process model
  - CAPE-OPEN compliant software interface

- CORBA Object Bus
  - SetVariables
  - GetResiduals
  - ESO

Dynamic real-time optimization
Comparison of approaches

<table>
<thead>
<tr>
<th></th>
<th>Sequential approach</th>
<th>Simultaneous approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>size of NLP</td>
<td>small</td>
<td>large</td>
</tr>
<tr>
<td>DAE model fulfilled in each step?</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>initial guess required for...</td>
<td>controls</td>
<td>states and controls</td>
</tr>
</tbody>
</table>

Experience from solving INCOOP benchmark problems

- sequential approach more robust and capable of handling bigger problems
- simultaneous approach can be faster with good initial guess, but more sensitive to initial guess
- accuracy problems with simultaneous approach for stiff problems (error controlled integration vs. fixed-grid collocation)
Dynamic real-time optimization
Interplay between D-RTO and MPC

- Soft constraints can be moved from MPC to D-RTO
- Longer time horizon for D-RTO to ensure feasibility
- D-RTO trigger for a possible re-optimization
- Delta-mode MPC computes updates to the control profiles for tracking the process in the strict operation envelope: rejects fast frequency process disturbances
Lagrange function sensitivities w.r.t. all estimated disturbances

\[ S_j = \frac{dL_j}{dd_j} \mid_{t_0j}; \]

\[ L_j = \Phi(u_i^{\text{ref}}, \hat{d}_j) + \mu_i^T h(u_i^{\text{ref}}, \hat{d}_j) \]

- One sensitivity integration of process model at each sampling time \( \tilde{t}_{0j} \) using previous D-RTO results (and active constraint set) at \( \tilde{t}_{0j} \) is required
- Compute change in sensitivities (\( \Delta S_j = S_j - S_i \)) and Lagrange function (\( \Delta L_j = L_j - L_i \)) can be then calculated
D-RTO trigger (II)

Optimal solution sensitivities w.r.t. all estimated disturbances

\[ U_j = \frac{du_j^{ref}}{d\hat{d}_j} \bigg|_{\hat{r}_0} \]

and changed active constraint set

• Solution to QP problem:
  - using second order information (Hessian of Lagrange function) \( \Rightarrow \) optimal sensitivities
  - using first order information \( \Rightarrow \) feasible only sensitivities

• updates as \( u_j^{ref} = u_i^{ref} + U_i^T (\hat{d}_j - \bar{d}_i) \)

• If \( \Delta S_j \) and \( \Delta L_j \) are larger than a threshold value \( S_{th} \) and changed active constraint set is predicted, a re-optimization should be done

• Else linear updates based on optimal solution sensitivities \( U_j \) are sufficient
results with re-optimization and feasible updates

Dynamic real-time optimization
• Reaction parameters randomly perturbed between their bounds
• Re-optimization done only when necessary
  ⇒ steer to desired grade
• Feasible updates only possible up to +4% change in parameter 1

• D-RTO problem needs to be solved only necessary
• the hybrid integrated D-RTO and control with embedded sensitivity analysis is well suited for large-scale industrial process operation
Off-line dynamic optimization:
Today already many numerical and software techniques available
for efficient and convenient solution of such problems

but...

dynamic optimization still not state-of-the-art (especially not in industry):
• Though pure solution time for solving one mathematical problem only in the
  order of *hours*,
• overall engineering time to solve the real application problem in the order of
  *weeks* or *months*.
• Problems: Modeling issues, problem formulation, convergence problems, ...

*It is still not “pushing the button”.*
## Future perspectives

Experience from INCOOP: for large-scale process models application of dynamic optimization in real-time still time-critical

<table>
<thead>
<tr>
<th>Dynamic real-time optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>• further enhance sequential approach dynamic optimization</td>
</tr>
<tr>
<td>• more elaborate adaptation strategies</td>
</tr>
<tr>
<td>• interaction NLP solver / integrator</td>
</tr>
<tr>
<td>• adapt integration accuracy</td>
</tr>
<tr>
<td>• incorporate second order information</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Integration of control and optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>• further exploit re-optimization features</td>
</tr>
<tr>
<td>• apply adaptation strategies in real-time context</td>
</tr>
<tr>
<td>• gain speed by feasible-first optimizations</td>
</tr>
<tr>
<td>• interlink MPC and D-RTO by shifting the prediction to the D-RTO level</td>
</tr>
</tbody>
</table>