Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

An Efficient Strategy for Real-Time Dynamic Optimization based on Parametric Sensitivities

Lynn Würth Ralf Hannemann Wolfgang Marquardt

*RWTH Aachen University, Process Systems Engineering, Turmstr. 46,
52064 Aachen, Germany, (*lynn.wuerth@avt.rwth-aachen.de,
ralf.hannemann@avt.rwth-aachen.de,
wolfgang.marquardt@avt.rwth-aachen.de*).*

Abstract: The optimal operation of chemical processes is challenged by frequent transitions and by the influence of process or model uncertainties. Under uncertainties, it is necessary to quickly update the optimal trajectories in order to avoid the violation of constraints and the deterioration of the economic performance of the process. Although an economically optimal operation can be ensured by online dynamic optimization, the high computational load of dynamic optimization associated with nonlinear and complex models is often prohibitive in real-time applications. To reduce the computational time required for online computation of the optimal trajectories in the neighborhood of the optimal solution under uncertainty, different strategies have been explored recently. If the operation is affected by small perturbations, efficient techniques for updating the nominal trajectories based on parametric sensitivities are applied, which do not require the solution of the rigorous optimization problem. However for larger perturbations, the linear updates obtained by the neighboring extremal solutions are not sufficiently accurate, and the solution of the nonlinear optimization problem requires further iterations with updated sensitivities to give a feasible and optimal solution. In this work, the sensitivity-based approach of Kadam and Marquardt (2004) is extended with a fast computational method for second-order derivatives based on composite adjoints. The application of the method to a simulated semi-batch reactor demonstrates that fast and optimal trajectory updates can be obtained.

Keywords: Real-Time Dynamic Optimization; Parametric Sensitivity Analysis; Uncertainty; Hessian; Composite Adjoints.

1. INTRODUCTION

Many chemical processes are operated in a flexible mode involving frequent transitions, which take place, for example, in batch processes or in continuous processes with frequent changes in product specifications. The profit achieved in transient processes is increased by the computation and tracking of optimal control trajectories, which are obtained through the solution of a dynamic optimization problem. However if the process operation is affected by process or model uncertainties, online updates of optimal trajectories are required.

The research interest in developing efficient strategies for solving dynamic optimization problems in online applications has grown in the past years. One approach consists of reducing the complexity by decomposing the optimal control problem on different time scales. Dünnebier et al. (2005) generate optimal control trajectories, that are tracked by an underlying MPC-Controller at a fast sampling rate. However, the solution of the dynamic optimization problem with large-scale models involves a high computational load and therefore renders online trajectory updates impractical.

A fast update strategy for optimal trajectories, based on sequential dynamic optimization, has been introduced by Kadam and Marquardt (2004). The authors apply parametric sensitivities to compute fast trajectory updates in the neighborhood of the optimal solution, even in the case of a changing active set. The computation of first-order approximations is based on the theory of neighboring extremals (Pesch, 1989), where an optimal feedback law is derived by linearization of the system along the optimal control trajectory. The feedback law allows an immediate correction of the optimal control trajectory for small deviations in the state vector. The optimal control problem and the approximation of the perturbed solution are set up as a boundary value problem, which is difficult to derive and the solution requires good estimates of the adjoint variables. In the recent literature, direct optimization methods have been applied to overcome these problems.

Real-time strategies for computing optimal control trajectories, which are based on multiple-shooting and simultaneous optimization methods, have also been presented by Diehl et al. (2005) and more recently by Zavala et al. (2007). A major feature of these algorithms, similar to the strategy presented in this paper, consists of the exploitation of sensitivity knowledge already available from the solution of previous optimization problems. The usage of this information allows to prepare the solution of the optimization problem at the next sampling interval to a large extent and to compute the trajectory update almost instantaneously once the measurements of the state variables are available. Diehl et al. (2005) perform only one iteration per sampling time in order to reduce the delay of the feedback response. Although the required computational time per update is short, a loss in performance might result if the system is affected by large uncertainties.

This work extends the sensitivity-based update strategy (Kadam and Marquardt, 2004) with a fast computation method for second-order derivatives, which allows online updating of the Hessian. The solution of the neighboring extremal problem requires exact second-order derivatives of the Lagrangian. The exact Hessian of the Lagrangian is computed using a second-order adjoint method (Hannemann and Marquardt, 2007), which is efficient especially if the optimization problem involves a large number of optimization parameters. The second-order sensitivities were computed using finite differences in our previous work (Kadam and Marquardt, 2004). However, the calculation by finite differences is too time-consuming to compute an online update of the second-order sensitivities. If the system is affected by large perturbations, the first-order update computed with parametric sensitivities might not provide a feasible and optimal solution because of strong nonlinearities in the model. Therefore, updates of the sensitivities are required for further iterations of the nonlinear optimization problem to obtain a converging and accurate solution. The exact Hessian provides fast convergence of the optimization problem in the neighborhood of the optimal solution.

2. REAL-TIME DYNAMIC OPTIMIZATION PROBLEM FORMULATION

The moving horizon formulation of the dynamic optimization problem is similar to the formulation used in nonlinear model-predictive control, although an economic objective is chosen to provide an economically optimal operation at all times. The moving horizon problem is defined as follows:

$$\min_{\boldsymbol{u}^j(t),t_f^j} \Phi(\boldsymbol{x}(t_f^j)) \tag{1}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}^j(t), \hat{\boldsymbol{d}}^j(t))\,, \tag{2}$$

$$\boldsymbol{y}(t) = \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}^j(t), \hat{\boldsymbol{d}}^j(t)) \tag{3}$$

$$\boldsymbol{x}(t^j) = \hat{\boldsymbol{x}}^j\,, \tag{4}$$

$$\boldsymbol{0} \geq \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{y}(t), \boldsymbol{u}^j(t))\,, \tag{5}$$

$$\boldsymbol{0} \geq \boldsymbol{e}(\boldsymbol{x}(t_f^j))\,, \tag{6}$$

$$t \in [t^j, t_f^j]\,, \tag{7}$$

$$t^j := t^{j-1} + \Delta t\,. \tag{8}$$

$\boldsymbol{x}(t) \in \mathbb{R}^{n_x}$ are state variables with the initial conditions $\hat{\boldsymbol{x}}^j$; $\boldsymbol{y}(t) \in \mathbb{R}^{n_y}$ are algebraic output variables. The dynamic process model (2) is given by $\boldsymbol{f}(\cdot)$. The time-dependent control variables $\boldsymbol{u}(t) \in \mathbb{R}^{n_u}$ and possibly the final time t_f^j are the degrees of freedom of the optimization problem. The optimization problem is solved on the time horizon $[t^j, t_f^j]$ at every sampling instant t^j; the optimization horizon is then shifted by the sampling interval Δt. Equations (5) and (6) describe the path constraints $\boldsymbol{h}(\cdot)$ on the input and state variables and the endpoint constraints $\boldsymbol{e}(\cdot)$ on the state variables. Process operation is determined by economic decision criteria, which enter into the definition of the objective function $\Phi(\boldsymbol{x}(t_f^j))$. These economic decision criteria are valid for a certain production campaign. Furthermore, uncertainties with different dynamics $\boldsymbol{d}(t)$ affect the process.

3. FAST TRAJECTORY UPDATES UNDER UNCERTAINTY

A rigorous solution of the dynamic optimization problem presented in Section 2 can be very time-consuming. Instead, a strategy based on parametric sensitivities is applied to provide fast, first-order updates of the control trajectory.

3.1 Nominal solution

The nominal optimization problem is solved using the dynamic optimization software DyOS (Schlegel et al., 2005), which adopts a control vector parameterization strategy. After discretizing the control variables, the infinite-dimensional optimal control problem (1) is converted into

$$\min_{\boldsymbol{z}} \Psi(\boldsymbol{z}, \boldsymbol{p}) := \Phi(\boldsymbol{z}, \boldsymbol{p}) \tag{9}$$

$$\text{s.t. } \boldsymbol{g}(\boldsymbol{z}, \boldsymbol{p}) \geq 0. \tag{10}$$

The objective function and the constraints depend on the discretized control variables $\boldsymbol{z}$ as well as on the parametric uncertainties collected in $\boldsymbol{p}$. The nonlinear program (9,10) is solved using a sequential quadratic programming (SQP) strategy, while the objective function, the constraints, and the gradients are computed by simultaneous integration of the model (2) and the associated sensitivity equations.

3.2 Sensitivity-based updates

Solving the optimal control problem on a moving horizon involves a sequence of related dynamic optimization problems. If the parametric uncertainties are small, it is not always necessary to solve a new optimization problem. In the neighborhood of the optimal solution, parametric sensitivities computed at the solution of the preceding optimization problem can be used to predict a first-order update of the optimal solution (Fiacco, 1983). The computational effort is reduced, since the required first and second-order sensitivities are evaluated beforehand at the nominal parameter values.

The parametric sensitivities of the control variables with respect to the uncertain parameters are obtained through differentiation of the first order necessary conditions of optimality (Karush-Kuhn-Tucker conditions) at the nominal parameter values, leading to the following system of linear equations:

$$\begin{bmatrix} L_{zz}(z_0,\lambda_0,p_0) & -g_z^T(z_0,p_0) \\ g_z(z_0,p_0) & 0 \end{bmatrix} \begin{bmatrix} z_p \\ \lambda_p^a \end{bmatrix} = - \begin{bmatrix} L_{zp}(z_0,\lambda_0,p_0) \\ g_p(z_0,p_0) \end{bmatrix} \tag{11}$$

where the Lagrangian is defined as $L(z, p, \lambda) = \Psi(z, p) - \lambda^T g(z, p)$. Under uncertainties, a first-order update of the control trajectory is rapidly computed from a first-order Taylor expansion around the nominal solution:

$$z(p) = z_0 + z_p(p_0)(p - p_0) \tag{12}$$

For the existence of a local minimum in the neighborhood of the nominal optimal solution, the strict complementarity slackness condition must hold, the active constraints' gradients must be linearly independent, and an important condition states that the strong second order sufficient conditions of optimality must be fulfilled (Fiacco, 1983). This can be checked by verifying the positive definiteness of the Hessian matrix projected on the null space of the Jacobian of the active constraints. Furthermore, it is assumed that the active constraint set does not change with uncertainties, which is quite restrictive.

A quadratic programming formulation, which allows updating the optimal solution while determining the new active set, has been reported by Ganesh and Biegler (1987):

$$\min_{\Delta z} \quad \frac{1}{2}\Delta z^T L_{zz}(p_0)\Delta z + \Delta p^T L_{pz}(p_0)\Delta z + f_z(p_0)\Delta z \tag{13}$$

$$\text{s. t.} \quad g(p_0) + g_z(p_0)\Delta z + g_p(p_0)\Delta p \geq 0\,, \tag{14}$$

Kadam and Marquardt (2004) have applied this formulation for real-time optimization and control of a semi-batch reactor to provide fast updates under uncertainties while accounting for active set changes. An optimal solution update can be rapidly computed by solving just one QP, if the perturbations are small and the updated solutions are close to the reference ones. For quadratic programs, the update corresponds to the exact optimal solution. However, for nonlinear problems, the error of the solution update is increasing substantially with the size of the perturbation. In that case, the linear update is not sufficient to provide the updated optimal solution.

3.3 Real-time algorithm

In order to account for larger uncertainties, the solution obtained by linear first-order updates is subsequently analyzed by a feasibility and an optimality criterion, which verifies whether the necessary conditions of optimality are satisfied. If this is not the case, a rigorous reoptimization is triggered (Kadam and Marquardt, 2004), that is computationally expensive. In this work, this approach is extended by computing a fast update of the first and second-order sensitivities and by triggering further iterations of the quadratic program similarly to an SQP algorithm. However, the first-order update may still be used as an initial guess for the next iteration of the optimization problem. The trigger evaluates the error in the Lagrange sensitivity (ϵ_{opt}) and the nonlinear constraint infeasibility (ϵ_{infs}) of the updated trajectories with respect to uncertainty and is defined as follows:

$$\epsilon_{opt} = \frac{\|L_z(z, p, \lambda)\|_\infty}{\|\lambda\|_2}, \epsilon_{infs} = \frac{\|g(z, p)\|_\infty}{\|z\|_2}. \tag{15}$$

The real-time optimization algorithm based on parametric sensitivities assuming a shrinking horizon as in batch optimization is as follows:

```
(1) Set counter $j$:=1;
(2) Solve the nominal dynamic optimization problem (1) on $t \in [0, t_{f_j}]$ to obtain the **reference solution** $z^j_{ref}$, $\lambda^j_{ref}$, $y^j_{ref}(t)$ computed at $p_0$. Assign $z^j_{ref}$, $y^j_{ref}$ as the reference trajectories for a lower level control system. Compute $L^j_{zz}, L^j_{zp}$ by the second-order adjoint method.
(3) **for** $j = 2, N$ (number of sample intervals) **do**
    (a) Implement the control $z^{j-1}_{ref}$ for one sampling interval $[t_{j-1}, t_j]$.
    (b) **Horizon shift:** Reduce the time horizon by one sampling interval $\Delta t$ i.e. $t_j$=$t_{j-1}$+$\Delta t$, $t_{f_j}$=$t_{f_{j-1}}$.
        On the reduced horizon, assemble the shifted reference discretized controls $\bar{z}^j_{ref}$, $\bar{\lambda}^j_{ref}$ $L^j_{zz}, L^j_{zp}$ from the corresponding quantities $z^{j-1}_{ref}$, $\lambda^{j-1}_{ref}$, $L^{j-1}_{zz}, L^{j-1}_{zp}$ on the previous horizon $[t_{j-1}, t_{f_{j-1}}]$.
        Prepare the next neighboring extremal control problem by formulating the QP given in eqns. (13) and (14).
    (c) **Process measurements:** At $t_j$ get the measurements $y^j$ and estimates $\hat{x}^j, \hat{d}^j$ after one sampling interval $\Delta t$ to update $p^j$.
    (d) **Fast update:** Solve eqns. (13) and (14) to obtain $\Delta z^j$,$\lambda^j$ and $\mathcal{G}^j$.
        Calculate the updated controls $z^j_{ref}$=$\bar{z}^j_{ref}$+$\Delta z^j$ and the Lagrange multipliers $\lambda^j_{ref}$=$\lambda^j$.
    (e) **Trigger:** Compute $y^j_{ref}$, $g^j$, and the sensitivities $f^j_z$, $g^j_z$ by doing one sensitivity integration of the DAE model using the updated control values $z^j_{ref}$. Compute the optimality error ($\epsilon_{opt}$, $\epsilon_{infs}$) of the updated controls and check the trigger criteria:
        **if** ($\epsilon_{opt}$>$\tau_{opt}$) and ($\epsilon_{infs}$>$\tau_{infs}$)
            update the 1st order sensitivities $f^j_z$, $g^j_z$ and second order sensitivities $L^j_{zz}, L^j_{zp}$ using the second-order adjoint method,
            Trigger another iteration of QP (13),(14) by setting $\Delta p$ = 0, until the feasibility and optimality criteria are satisfied.
        **end if**
        Get the new optimal references $z^j_{ref}$, $y^j_{ref}$, $\lambda^j$.
    (f) **Update sensitivities:** Update the reference sensitivities $L^j_{zz}, L^j_{zp}$, $f^j_z$, and $g^j_z$.
(4) **end for**;
```

The algorithm ensures that directly after getting the process measurements, the trajectory update is computed almost instantaneously by solving the neighboring extremal problem in QP (13),(14). The solution of the quadratic problem is only based on nominal sensitivity information, and on the current measurements or estimations of the process uncertainties. Therefore, the update of the reference sensitivities required to prepare the next neighboring extremal problem (step 3.f) as well as the shifting of the nominal solution (step 3.b) can be carried out in the background during the process sampling interval. The extension of the algorithm to a receding horizon problem is straightforward.

If the update does not fulfill the criteria (15), efficient adjoint-based methods are used to quickly update second-order sensitivities. Furthermore, the adjoint-based methods compute the exact Hessian matrix, which provides quadratic convergence close to the optimal solution and therefore significantly reduces the number of iterations required to obtain the optimal solution.

3.4 Computation of Hessian Matrix by Composite Adjoints

The second-order derivatives L_{zz} and L_{zp} are computed using the novel approach of composite adjoints presented by Hannemann and Marquardt (2007), that is based on the second-order adjoint sensitivity analysis recently investigated by Özyurt and Barton (2005). The classical second-order adjoint approach is not suited for path constrained optimal control problems since the Lagrangian consists of a linear combination of different ODE embedded functionals, which are evaluated at different points in time. The different points in time stem from the discretization of the path constraints on the grid $t_j < t_{j+1} < \cdots < t_N = t_f^j$:

$$\mathbf{0} \geq \boldsymbol{h}(\boldsymbol{x}(t_k; z, p), \boldsymbol{y}(t_k; z, p), \boldsymbol{u^j}(t_k; z, p)),\ k = j+1, \ldots, N.$$

In the Lagrangian, the path constraints occur as a linear combination associated with their corresponding Lagrange multipliers $\mu_k,\ k = j+1, \ldots, N$:

$$\sum_{k=j+1}^{N} \mu_k^T \boldsymbol{h}(\boldsymbol{x}(t_k; z, p), \boldsymbol{y}(t_k; z, p), \boldsymbol{u^j}(t_k; z, p)).$$

Following an ordinary adjoint approach, for each of the time points $t_k, k = j+1, \ldots, N$, an individual combined first- and second-order adjoint system has to be solved. The method of composite adjoints adapts the second-order adjoint sensitivity analysis for path constrained problems exploiting the linearity of the first- and second-order adjoint system. Instead of solving $(N - j)$ "single" adjoint systems, only one system is integrated to obtain the piecewise smooth composite adjoint vector, which is a linear combination of the "single" adjoint vectors. An illustration of the "single" and composite adjoints is given in Figure 1. Details of the algorithm have been reported by Hannemann and Marquardt (2007).

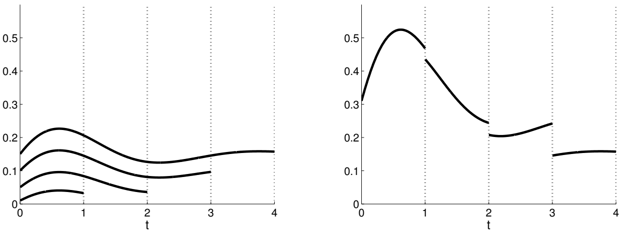

Fig. 1. "Single" adjoints (left) and composite adjoint (right)

Second-order adjoint sensitivity analysis and the method of composite adjoints require the storage of the solution of the state and first-order forwards sensitivity systems for the backwards integration of the first- and second-order adjoint systems. To avoid the necessity of interpolation, a second-order discrete adjoint approach Hannemann and Marquardt (2008) motivated by Hager (2000) is used. This discrete approach yields exact first- and second-order derivatives of the discretized initial value problem and is in this sense equivalent to automatic differentiation. In addition to the storage of the solution of the state and sensitivity equations also the first- and second-order composite adjoints are stored in the current implementation, doubling the total storage requirement. The storage requirement grows linearly in the number of integrations steps, the amount of intermediate storage per variable per integration step, the number of states, and the number of parameters.

For this case study, an s-stage Runge-Kutta method with $s = 4$ is used. On the largest horizon N integration steps with $N \approx 500$ steps and a total amount of $n_p = 66$ parameters are taken. The intermediate storage per variable per integration step is $s + 1$ This results in a storage requirement of $2n_x(n_p + 1)N(s + 1)$ variables, or about 24 megabytes of computer RAM, if 8-byte double precision variables are employed. Due to the availability of cheap computer RAM, this discrete composite adjoint approach is also feasible for medium size models with a few hundreds of states and a few hundreds of parameters. Of course, for really large-scale models with thousands of variables, a different implementation using checkpointing techniques has to be employed.

4. CASE STUDY

As a benchmark problem, the Williams-Otto semi-batch reactor, as introduced by Forbes (1994), is chosen. In the reactor the following reactions take place:

$$A + B \xrightarrow{k_1} C,\ C + B \xrightarrow{k_2} P + E,\ P + C \xrightarrow{k_3} G.$$

Reactant A is already present in the reactor, whereas reactant B is fed continuously to the reactor. During the exothermic reactions the products P and E as well as the side-product G is formed. The heat generated through the exothermic reaction is removed by a cooling jacket, which is controlled by manipulating the cooling water temperature. At the end of the batch, the conversion to the main products P and E should be maximized. During the batch, path constraints on the inlet flow rate of reactant B ($F_{B_{in}}$), the reactor temperature (T_r), the reactor volume (V) and the cooling water temperature (T_w) must be observed. The manipulated control variables of this process are $F_{B_{in}}$ and T_w. The dynamic model consists of nine differential equations.

The economic objective is to maximize the yield of the main products at the end of batch. The dynamic optimization problem is defined as follows:

$$\max_{F_{B_{in}}(t), T_w(t)} \Phi(t_f) = c_p n_p(t_f) + c_e n_e(t_f) \tag{16}$$

s.t. *process model*, and

$$0 \leq F_{B_{in}}(t) \leq 5.784 \frac{\text{kg}}{\text{sec}}, \tag{17}$$

$$V(t_f) \leq 5 \text{ m}^3, \tag{18}$$

$$20 \leq T_w(t) \leq 100 \text{ ℃}, \tag{19}$$

$$60 \leq T_r(t) \leq 90 \text{ ℃}. \tag{20}$$

The initial reactor temperature $T_{r,0}$ and feed temperature T_{in} are fixed at their nominally optimal values of 60 ℃ and 35 ℃, respectively. During the batch, two types of uncertainties are influencing the process. On the one hand, the inlet temperature decreases gradually from 35°C to 25°C during the time span of 200s. Furthermore, uncertainty is present in the model, since the reaction parameter b_1 in the reaction kinetics $k_1{=}a_1 \exp(\frac{b_1}{T_r + 273.15})$ deviates by +10% from its nominal value $b_1 = 6666, 7$. This parameter is assumed to be estimated on-line. We collect T_{in} and b_1 in the vector of uncertain parameters p. The uncertainties are represented in Figure 2.

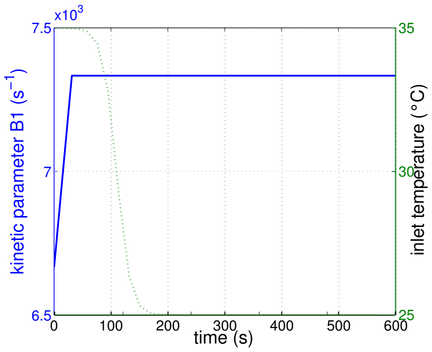

Fig. 2. Uncertainties: Model uncertainty in the kinetic parameters (solid line) and disturbance in the inlet temperature (dashed line)

The reference solution, which is shown in Figures 3-6 by solid lines, is first computed at the nominal parameter values. For online implementation, a sampling time of 31.25 s is chosen, whereas the total batch time is fixed at 1000 s. This results in a uniform discretization grid of both control variables with 32 parameters. A uniform grid is chosen here for simplicity. Adaptive grid refinement presented in (Schlegel et al., 2005) would reduce the number of optimization parameters and the computational solution time. The application of the real-time algorithm to optimize the process online is shown in Figures 3-6 by solid lines, whereas the sampling times are represented by dots. This solution is compared to a conventional rigorous solution strategy, where a dynamic optimization problem is iterated until full convergence using a sequential quadratic programming strategy with BFGS Hessian updates (dashed lines marked with a cross). A warm-start is also provided for the rigorous approach at each interval using the shifted solution of the previous problem. All computations are performed on a Windows XP PC with a 2.0 GHz AMD Athlon(TM) XP 2400+ processor and a memory of 2048 MB.

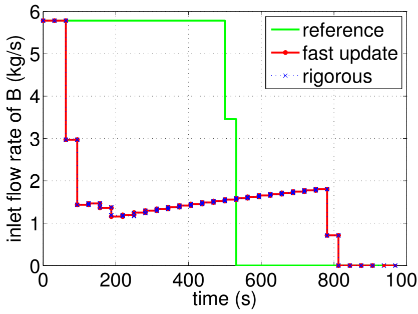

Fig. 3. Inlet Flowrate

In Figures 3 and 4 the plots of online optimized manipulated variables show that the solution obtained with the real-time algorithm are coinciding with the solution obtained with conventional optimization. Under the influence of uncertainties significant changes in the active set are observed, which are correctly detected in the real-time algorithm.

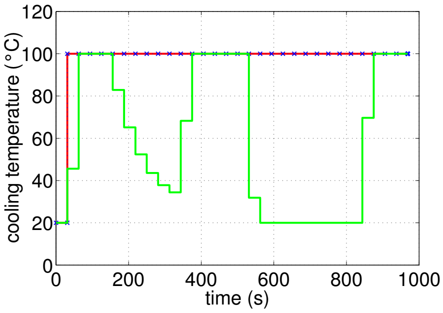

Fig. 4. Cooling Water Temperature

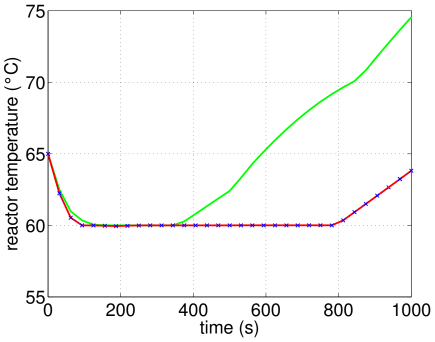

Fig. 5. Reactor Temperature

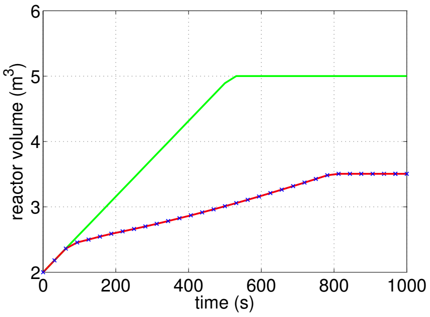

Fig. 6. Reactor Volume

The objective values, the number of iterations and the CPU times per interval are compared in table 1. The values of the objective are very close, even in the cases where only one iteration is performed using the parametric sensitivity-based solution of (13). Except on the first interval, the solution of the neighboring extremal problem is sufficiently accurate to compute the optimal update. However on the first interval, the uncertainty in the kinetic parameter is significantly perturbing the optimal solution. Therefore after the first solution of the quadratic program, further iterations are triggered because the trajectories obtained by the first-order update are violating the constraints and do not fulfil the feasibility and optimality criteria in eq. (15). Already after a small number of iterations, the optimal solution is obtained on the first sampling interval. Com-

	Sampling Time	0	31.25	62.5	93.75	125	156.25	187.5	218.75	250	281.25	312.5	343.75
Real-	Objective		1266.7	1265.9	1262.5	1248.9	1204.1	1128	1083.2	1070.1	1067.3	1066.6	1066.6
Time	Iterations		5	1	1	1	1	1	1	1	1	1	1
Alg	CPU Time		6.25	1.29	1.13	1.11	1.09	1.06	1.04	0.99	0.98	0.95	0.93
Rig.	Objective		1266.8	1265.9	1262.5	1249	1204.9	1130.4	1083.9	1070.3	1067.4	1066.7	1066.7
Opt.	Iterations		21	5	3	6	7	8	8	7	3	5	3
Alg	CPU Time		9.58	3.49	6.09	3.53	3.78	4.25	3.89	3.84	5.03	2.81	2.06

Table 1. Objective function, number of iterations, and computational time per sampling interval of real-time algorithm compared to a rigorous online solution of the optimization problem

pared to the second approach, the number of iterations per sampling instant is significantly smaller. This is due to the fact, that the exact Hessian matrix is computed in the real-time algorithm, whereas the dynamic optimization is solved in the second approach using a BFGS update, which has a much slower rate of convergence. The computational time required for the exact Hessian matrix is relatively low in this case study with 0.6 seconds per iteration. However, the total CPU time per iteration is lower in the second approach, since the cost of computing one BFGS update is very low.

5. CONCLUSIONS AND FUTURE WORK

The real-time algorithm based on neighboring extremals provides closed-loop solutions which are nearly identical to the solution obtained with rigorous dynamic online optimization. The application to a benchmark problem has shown that updates obtained by solving one quadratic program with parametric sensitivities are very fast and sufficiently accurate, if the process is perturbed by small disturbances. In order to update the first and second-order sensitivities online, an efficient computation method based on composite adjoints is employed. Furthermore, the performance is not deteriorating for larger uncertainties because further iterations are triggered in order to satisfy a certain feasibility and optimality criterion. Since in this paper a rather small benchmark problem is used, the approach will be applied to larger models of industrial processes in the next step, where the computation of the Hessian matrix will become more expensive. However, it is expected that by adapting the discretization of the control variables to the solution structure according to the approaches presented by Schlegel et al. (2005) and Hartwich et al. (2007), the number of optimization variables will be reduced, which will result in a further decrease in computational time.

ACKNOWLEDGEMENTS

This work was supported in part by the European Commission (MRTN-CT-2004-512441) and by the German Research Foundation DFG (MA 1188/28-1).

REFERENCES

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

G. Dünnebier, D. van Hessem, J. Kadam, K.-U. Klatt, and M. Schlegel. Optimization and control of polymerization processes. *Chem. Eng. Technology*, 28(5):575–580, 2005.

A.V. Fiacco. *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming.* Academic Press, New York, 1983.

J.F. Forbes. *Model Structure and Adjustable Parameter Selection for Operations Optimizations.* PhD thesis, McMaster University, Hamilton, Canada, 1994.

N. Ganesh and L.T. Biegler. A reduced Hessian strategy for sensitivity analysis of optimal flowsheets. *AIChE Journal*, 33(2):282–296, 1987.

W. Hager. Runge-Kutta methods in optimal control and the transformed adjoint system. *Numer. Math.*, 87:247–282, 2000.

R. Hannemann and W. Marquardt. Fast computation of the Hessian of the Lagrangian in shooting algorithms for dynamic optimization. In *Proceedings of the DYCOPS 8 conference*, Cancun, Mexico, June 2007.

R. Hannemann and W. Marquardt. Continuous and discrete composite adjoints for the Hessian of the Lagrangian in shooting algorithms for dynamic optimization. Submitted to: SIAM Journal of Scientific Computing, 2008.

A. Hartwich, M. Schlegel, L. Würth, and W. Marquardt. Adaptive control vector parameterization for nonlinear model-predictive control. *International Journal of Robust and Nonlinear Control*, 2007. in press.

J. Kadam and W. Marquardt. Sensitivity-based solution updates in closed-loop dynamic optimization. In *Proceedings of the DYCOPS 7 conference*, July 2004.

D.B. Özyurt and P.I. Barton. Cheap second order directional derivatives of stiff ODE embedded functionals. *SIAM J. Sci. Comput.*, 26(5), 2005.

H.J. Pesch. Real-time computation of feedback controls for constrained optimal control problems, part 2: A correction method based on multiple shooting. *Optimal Control Applications & Methods*, 10(2):147–171, 1989.

M. Schlegel, K. Stockmann, T. Binder, and W. Marquardt. Dynamic optimization using adaptive control vector parameterization. *Comp. & Chem. Eng.*, 29(8), 2005.

V.M. Zavala, C.D. Laird, and L.T. Biegler. Fast implementations and rigorous models: Can both be accommodated in NMPC? *International Journal of Robust and Nonlinear Control*, 2007. in press.