COMBINED ON-LINE AND RUN-TO-RUN OPTIMIZATION OF BATCH PROCESSES WITH TERMINAL CONSTRAINTS

C. Welz, B. Srinivasan and D. Bonvin

Laboratoire d'Automatique, École Polytechnique Fédérale de Lausanne CH-1015 Lausanne, Switzerland

Abstract: This paper describes the optimization of batch processes in the presence of uncertainty and constraints. The optimal solution consists of keeping certain path and terminal constraints active and driving the sensitivities to zero. The case where the terminal constraints have a larger bearing on the cost than the sensitivities is considered, for which a two-time-scale methodology is proposed. The problem of meeting the active terminal constraints is addressed *on-line* using trajectory tracking, whilst pushing the sensitivities to zero is implemented on a *run-to-run* basis. The paper also discusses the run-to-run improvement of trajectory tracking via iterative learning control. The proposed methodology is illustrated in simulation on a batch distillation system.

Keywords: Dynamic optimization, Measurement-based optimization, Batch processes, Iterative learning control, Run-to-run control, Batch distillation.

1. INTRODUCTION

A frequent objective in batch process optimization is the maximization of product yield at final time while satisfying path and terminal constraints. The standard approach in the presence of uncertainties is to implement a conservative open-loop strategy that leads to feasible but sub-optimal operation. However, the measurements available during the batch or at batch end could be used to improve the performance by reducing this conservatism (Srinivasan *et al.*, 2003*a*). Depending on how these measurements are used, two methodologies can be distinguished:

(1) *Explicit optimization schemes* update a model of the process using the available measurements, and the refined model is used in the subsequent optimization step. This method suffers from the typical conflict between parameter estimation and optimization (Roberts and Williams, 1981; Srinivasan and Bonvin, 2002).
(2) *Implicit optimization schemes* directly use the measurements in a feedback control law to determine the optimal inputs (Srinivasan *et al.*, 2003*a*). This method, which has the advantage of easy and robust implementation, is the subject of this paper.

The implicit optimization scheme that consists of tracking the necessary conditions of optimality (NCO-tracking scheme) is considered here (Srinivasan *et al.*, 2003*a*). The main emphasis is on meeting the active constraints since much can be gained by keeping them active. Keeping the *path constraints* active is fairly straightforward using on-line feedback, while meeting the *terminal constraints* is less intuitive.

One possibility to enforce the active terminal constraints is via run-to-run control by using only batch-end measurements (Francois *et al.*, 2002).

However, this approach is slow and does not use the information available during the batch. Also, it cannot handle disturbances occurring during the batch. Hence, the problem addressed here is the use of *on-line measurements* to guarantee the satisfaction of *terminal constraints*, thereby ensuring feasible and near-optimal operation.

Meeting terminal constraints using mid-course corrections requires a model to predict the values of the constrained quantities at final time (Yabuki and MacGregor, 1997). Since such a model is rarely available in batch processing, this paper proposes to use trajectory tracking instead.

The more important problem in terms of performance, i.e meeting the terminal constraints, is addressed on-line using trajectory tracking. Also, the performance of trajectory tracking is improved using Iterative Learning Control (ILC) techniques (Moore, 1993; Lee *et al.*, 2000). The sensitivity aspect is addressed via trajectory adaptation on a run-to-run basis. Thus, the proposed scheme encompasses the following two parts:

(1) *Tracking of reference trajectories (on-line + run-to-run):* This part consists of choosing and tracking some feasible reference trajectories whose main purpose is to steer the process towards the terminal constraints. For this, an on-line feedback strategy is used. In addition, ILC can help improve the feedforward inputs for the current run using measurements from the previous run by exploiting the repetitive nature of batch processes.
(2) *Adaptation of the reference trajectories (run-to-run):* Though the terminal constraints can be met by steering the process along appropriate trajectories, the shape of these trajectories is important for optimality. Here, the reference trajectories are parameterized, and the corresponding parameters are adapted on a run-to-run basis in order to drive the cost sensitivities to zero.

The paper is organized as follows: In Section 2, the concept of measurement-based optimization is briefly revisited. Section 3 describes how trajectory tracking can be used to handle terminal constraints. Section 4 deals with improved trajectory tracking using ILC, whereas the run-to-run adaptation of the reference trajectories is presented in Section 5. As an illustration, the optimization of a batch distillation system is evaluated in Section 6, and conclusions are presented in Section 7.

2. MEASUREMENT-BASED OPTIMIZATION VIA NCO-TRACKING

The following terminal-cost optimization problem for the kth batch is considered:

$$\begin{aligned} \min_{u_k(t)} \quad & \phi(x_k(t_f),\theta) \qquad (1)\\ s.t.\ \ & \dot{x}_k = F(x_k,u_k,\theta)+v_k(t),\ x_k(0)=x_k^{ic}\\ & y_k = H(x_k,u_k,\theta)+w_k(t)\\ & S(x_k,u_k,\theta)\le 0, \quad T(x_k(t_f),\theta)\le 0 \end{aligned}$$

where ϕ is the scalar cost function, x_k the states with the known initial conditions x_k^{ic}, u_k the inputs, y_k the outputs, and t_f the final time. F are the functions describing the system dynamics, S the path constraints, T the terminal constraints, θ the uncertain parameters, v_k the process noise, and w_k the measurement noise.

Applying Pontryagin's Maximum Principle (PMP) to (1) results in the following Hamiltonian and adjoint equations:

$$H_k = \lambda_k^T F + \mu_k^T S \qquad (2)$$

$$\dot{\lambda}_k^T = -\frac{\partial H_k}{\partial x_k}, \quad \lambda_k^T(t_f) = \left.\frac{\partial \phi}{\partial x_k}\right|_{t_f} + \nu_k^T \left.\frac{\partial T}{\partial x_k}\right|_{t_f} \qquad (3)$$

where $\lambda_k(t) \neq 0$ are the adjoint states, $\mu_k(t) \geq 0$ the Lagrange multipliers for the path constraints, and $\nu_k \geq 0$ the Lagrange multipliers for the terminal constraints. The first-order necessary conditions of optimality (NCO) can be partitioned as follows (Srinivasan *et al.*, 2003*b*):

	Path	Terminal
Constraints	$\mu_k^T S = 0$	$\nu_k^T T = 0$
Sensitivities	$\frac{\partial H_k}{\partial u_k} = 0$	$\left.\frac{\partial(\phi+\nu_k^T T)}{\partial x_k}\right\|_{t_f} - \lambda_k^T(t_f) = 0$

(4)

Note that the NCO are made of two parts (constraints and sensitivities), each with two components (path and terminal). These NCO remain valid also in the presence of uncertainty.

The optimal solution is typically discontinuous and characterized by various intervals that are either constraint- or sensitivity-seeking (Srinivasan *et al.*, 2003*b*). The NCO-tracking scheme enforces the four conditions in (4), some on-line and the others over successive batches.

3. MEETING TERMINAL CONSTRAINTS USING TRAJECTORY TRACKING

While enforcing the NCO, the natural partitioning of the tasks is to deal with the path conditions (first column of (4)) within the run and handle the terminal conditions (second column of (4)) on a run-to-run basis (Francois *et al.*, 2002). This seems natural since the measurements related to the terminal quantities are only available at the end of the run. However, the idea used here is to deal with all the constraints (first row of (4)) within the run and handle the sensitivities (second row of (4)) on a run-to-run basis. This way, on-line tracking is used to meet the path and terminal constraints and run-to-run update to drive the sensitivities to zero. Without loss of generality, assume that all terminal constraints are active, the non-active ones being simply discarded.

The advantage of this partitioning is that each batch is feasible, though the operation may not be optimal. Also, from an optimization perspective, there is often considerably more to gain by meeting the active constraints compared to pushing the sensitivities to zero (Srinivasan *et al.*, 2003*a*). Hence, the more important problem is addressed on-line, and the rest is worked out over successive batches.

Unfortunately, on-line measurements do not directly provide information regarding the terminal constraints, and some sort of prediction or extrapolation is needed (Yabuki and MacGregor, 1997). Since such a prediction is not always accurate due to model mismatch and disturbances, the idea used here is to track feasible reference trajectories, $T_{ref}[0, t_f]$, whose main purpose is to guarantee meeting the terminal constraints, i.e. $T(x(t_f), \theta) = 0$ (Welz *et al.*, 2002).

In accordance with the notation $T(x(t_f), \theta) = 0$, let $T(t)$ represent the values at time t of the quantities that are constrained at final time. The tracking presented in this paper requires that $T(t)$ be measured not only at final time but also during the run. Then, let $T_k[0, t_f]$ be the trajectories during the interval $[0, t_f]$ in the kth run. The goal of tracking is to push $T_k[0, t_f]$ towards $T_{ref}[0, t_f]$. Various aspects involved in the tracking of these trajectories will be discussed in the next section. Also, in the presence of measurement noise, safety margins or backoffs need to be introduced so that the terminal constraints are not violated.

It is interesting to note the twist in concept: Instead of building a *model* capable of predicting accurately the values $T(x(t_f), \theta)$ and adjusting the model for the prediction to match the reality, "appropriate" *trajectories* $T_{ref}[0, t_f]$ are proposed and the inputs adjusted so that the reality matches the proposed trajectories. Another difference arises from the fact that, when a model is used to predict the final values, the inputs need to be computed via optimization to enforce $T(x(t_f), \theta) = 0$. Here, this correction step is part of the trajectory tracking mechanism since $T_{ref}(x(t_f)) = 0$ by construction.

4. TRAJECTORY TRACKING USING ILC

This section addresses the problem of tracking given trajectories using information from the current and previous runs. Measurements from the current run are used for on-line *feedback* corrections, while those from earlier runs act on the *feedforward* part of the inputs. Thus, the inputs for the kth run have two parts, $u_k = u_k^{ff} + u_k^{fb}$, where the superscripts $(.)^{ff}$ and $(.)^{fb}$ are used for the feedforward and feedback parts, respectively .

The trajectory tracking problem using information from previous runs has been addressed in the literature under the name Iterative Learning Control (ILC). ILC was first proposed in the robotics community as a means to successively reduce the tracking errors in repetitive dynamic processes (Arimoto *et al.*, 1984). In the ILC literature, the run-to-run aspect was developed before studying its interaction with the on-line (within-run) adaptation. A similar line of presentation will be followed in this section.

4.1 Run-to-run adaptation in ILC

Consider a batch process with the inputs u and the outputs T. Let the input-output relationship for the kth run be, $T_k[0, t_f] = G\, u_k[0, t_f]$, where G is an operator describing the process, and $u_k[0, t_f]$ and $T_k[0, t_f]$ represent the time trajectories of u and T, respectively.

The objective of ILC is to compute $u_k[0, t_f]$ so that $T_k[0, t_f] \to T_{ref}[0, t_f]$ as $k \to \infty$. This eventually means system inversion, i.e. finding the inputs $u_\infty[0, t_f]$ that produce the outputs $T_{ref}[0, t_f]$: $u_\infty[0, t_f] = G^{-1} T_{ref}[0, t_f]$. The solution is straightforward if G is known and invertible. Unfortunately, this is rarely the case.

Instead of direct inversion, ILC represents an iterative inversion process that does not require a model since process data are utilized. Yet, an inverse must exist, i.e. it must be verified that the proposed trajectories $T_{ref}[0, t_f]$ are feasible so that the sought inputs do exist. The basic ILC algorithm updates the feedforward input trajectories from one run to the next using past control inputs and measurements:

$$u_{k+1}^{ff}[0, t_f] = u_k^{ff}[0, t_f] + K^{ff} e_k[0, t_f] \qquad (5)$$

where K^{ff} is an operator applied to the previous tracking errors, $e_k[0, t_f] = T_{ref}[0, t_f] - T_k[0, t_f]$, and possibly their derivatives. With the above update, the sufficient condition for convergence with zero tracking errors is:

$$\|I - K^{ff} G\| < 1 \qquad (6)$$

where $\|\cdot\|$ denotes the induced operator norm. For this condition to be verified, $K^{ff} G$ needs to have a causal inverse (Moore, 1993). This last condition can be interpreted as follows: (i) if G is causal with non-zero relative degree, then a non-causal K^{ff} is required for convergence (Lee *et al.*, 2000), or (ii) if G is of zero relative degree, then a causal K^{ff} with zero relative degree can be used.

Since exact system inversion may lead to ill-conditioning, the problem of approximate inversion has been studied, i.e. convergence with non-zero, yet small, residual errors (Moore, 1993). For this to happen, and in contrast to (5) that represents an integral law from a run-to-run view point, the ILC adaptation law does not need an integral term. It follows that the conditions for convergence are much less restrictive.

An extension to the update law (5) is to consider not only the last, but several previous error trajectories (Chen and Wen, 1999; Bien and Huh, 1989). However, recent results show that most higher-order ILC schemes can be reduced to first-order ones (Phan and Longman, 2002).

4.1.1. Types of ILC algorithms: In traditional ILC algorithms, the operator K^{ff} is either of the P-type (proportional) or D-type (derivative). The D-type algorithms typically use $K^{ff} = k\frac{d^r e_k}{dt^r}$, where r is the system relative degree and k a gain. Though the D-type algorithms are more sensitive to noise, from a theoretical perspective, they provide better convergence results since the algorithms are intrinsically non-causal.

The derivative action can be replaced by anticipation in time (Wang, 2000). For discrete-time systems, Amann *et al.* (1996) proposed to shift the error trajectories of the previous run backwards in time by rT_s (anticipation), where r is the system relative degree and T_s the sampling time. The same idea is also used to cope with varying initial conditions (Sun and Wang, 2003) or in the context of time-delay systems (Hideg, 1996; Park *et al.*, 1998).

Along the same lines, this paper uses anticipation in time. Instead of considering the process map $u_k^{ff}[0, t_f] \to T_k[0, t_f]$, the map $u_k^{ff}[0, t_f - \delta] \to T_k[\delta, t_f]$ is used, where δ is a delay term. For the inverse system, this delay corresponds to a prediction and can be viewed as the non-causality term needed in the controller K^{ff} for the sake of convergence with zero tracking errors. Larger delays make the control inputs less aggressive. The ILC control law becomes:

$$u_{k+1}^{ff}[0, t_f - \delta] = u_k^{ff}[\delta, t_f] + K^{ff} e_k[\delta, t_f] \quad (7)$$

The remaining interval $u_{k+1}^{ff}[t_f - \delta, t_f]$ is approximated as the constant value $u_k^{ff}(t_f)$.

From another view point, the delay δ is the time required by the process to catch up with the reference trajectories. Since, due to unmatched initial conditions, it is usually not possible to follow $T_{ref}[0, t_f]$ right from the start, the update law (7) uses only $T_{ref}[\delta, t_f] - T_k[\delta, t_f]$ for adaptation.

The major difference with the results presented in the literature is that, in (7), u_k^{ff} is also shifted in time. Hence, there is no integral action from a run-to-run perspective, thereby leading to residual tracking errors, i.e. to only approximate inversion.

4.2 Within-run adaptation in ILC

The ILC laws (5) and (7) represent feedback on a run-to-run basis, but they are clearly open loop for the current run. Therefore, within-run disturbances cannot be compensated. By adding on-line feedback to ILC, deviations from the desired trajectories can be handled instantaneously in the current run. Since the resulting tracking errors are smaller than in conventional ILC, convergence is faster and more robust to uncertainty and noise. Furthermore, within-run stability can be guaranteed for unstable systems. The inputs then become:

$$u_{k+1}(t) = u_{k+1}^{ff}(t) + K^{fb} e_{k+1}(t) \quad (8)$$

For this update law, Kuc *et al.* (1991) provide convergence conditions based on Lyapunov theory. Since, due to feedback, the plant G is replaced by $(I + GK^{fb})^{-1}G$, the following sufficient condition for convergence results, compare (Moore, 1999):

$$\|(I - K^{ff}(I + GK^{fb})^{-1}G)\| < 1 \quad (9)$$

Two important sub-cases can be considered: (a) $K^{ff} = K^{fb}$: For such a case, Xu *et al.* (1995) have shown that, with a proportional controller and under mild conditions, convergence is independent of the gain. Consequently, a high gain can be used to obtain fast convergence. Also, Amann *et al.* (1996) propose a norm-optimal ILC scheme. (b) $K^{ff} = K^{fb} + \bar{K}^{ff}$: Writing the convergence condition of the ILC scheme in terms of $\bar{K}^{ff}$, Jang *et al.* (1995) have shown that, with a D-type $\bar{K}^{ff}$ and in the absence of zero dynamics, the feedback controller has no influence on the convergence condition but convergence may be faster.

5. RUN-TO-RUN TRAJECTORY UPDATE FOR OPTIMALITY

The last two sections have shown that, if the quantities $T(t)$ can be measured or reconstructed on-line from other measurements and if the tracking of $T_{ref}[0, t_f]$ is satisfactory, then the terminal constraints will be active. However, there remains an important question: From among the many feasible trajectories that can be proposed to enforce $T_{ref}(t_f) = 0$, how can $T_{ref}[0, t_f]$ be chosen in order to minimize the cost in the sense of Problem (1)?

Since the constraints are met via trajectory tracking, the sensitivities can be pushed towards zero by adjusting the reference trajectories on a run-to-run basis. Let the reference trajectories be parameterized as:

$$T_{ref}[0, t_f] = \mathcal{T}(\pi). \tag{10}$$

Note that the parameterization is done for the reference trajectories and not for the inputs as in numerical optimization. A parsimonious parameterization of the reference trajectories, i.e. with a small number of parameters, can usually be chosen on the basis of the results of numerical optimization. Approximations using either piecewise-constant or piecewise-linear intervals often suffice (Srinivasan *et al.*, 2003*b*).

The parameters π can be adapted on a run-to-run basis using a gradient-type update law that pushes the sensitivities to zero:

$$\pi_{k+1} = \pi_k - K_\pi \left.\frac{\partial \phi}{\partial \pi}\right|_k \tag{11}$$

where K_π is a gain matrix. The update law requires the knowledge of the sensitivities $\partial\phi/\partial\pi$, which can be evaluated using finite perturbations. Though this procedure is slow and experimentally intensive, it is acceptable here as explained next. Since the terminal constraints are met, all iterations lead to an acceptable product and, furthermore, the operation is fairly close to being optimal. Thus, the run-to-run update merely aims at the last few percent in performance, thereby making the number of experiments required to convergence rather immaterial.

The global optimization scheme is depicted in Figure 1. It has three parts: (i) on-line tracking of $T_{ref,k}(t)$, (ii) run-to-run update of $u_k^{ff}[0, t_f]$, and (iii) run-to-run update of $T_{ref,k}[0, t_f]$.

The optimization scheme can be looked upon from two different view points: (a) *Trajectory tracking and trajectory adaptation:* The parts (i) and (ii) belong to the trajectory tracking task, while (iii) is concerned with trajectory adaptation. (b) *On-line and run-to-run implementation:* The part (i) is implemented on-line, while (ii) and (iii) are implemented off-line.

6. APPLICATION TO A BATCH DISTILLATION SYSTEM

6.1 Problem formulation

A binary batch distillation system is used to illustrate in simulation the methodological developments of the previous sections. The following assumptions are made: (1) Equimolar overflow, (2) Constant relative volatility, ideal vapor-liquid equilibrium, (3) Equilibrium stages, (4) Negligible vapor holdup, (5) Constant liquid holdup on stages and in condenser, (6) Total condenser, (7) Constant boilup rate.

Considering the column with p equilibrium stages and writing molar balance equations for the holdup in the reboiler and for the liquid on the various stages and in the condenser, the following model of order $(p+2)$ is obtained:

$$\dot{M}_1 = -f_d V \tag{12}$$

$$\dot{x}_1 = \frac{V}{M_1}\left(x_1 - y_1 + (1-f_d)\, x_2\right) \tag{13}$$

$$\dot{x}_i = \frac{V}{M_i}\left(y_{i-1} - y_i + (1-f_d)\left(x_{i+1} - x_i\right)\right) \tag{14}$$

$$\dot{x}_c = \frac{V}{M_c}\left(y_p - x_c\right) \tag{15}$$

with $i = 2, \ldots, p$, x_i the molar liquid fraction, y_i the molar vapor fraction, and M_i the molar holdup on Stage i. Stage 1 refers to the reboiler and Stage p to the top of the column. M_c is the holdup in the condenser. The ratio f_d of the distillate to boilup rate, $f_d = \frac{D}{V}$, is considered as the manipulated variable. The vapor-liquid equilibrium relationship is:

$$y_i = \frac{\alpha x_i}{1 + (\alpha - 1) x_i}, \quad i = 1, \cdots, p \tag{16}$$

where α is the relative volatility. The model parameters and the initial conditions are given in Table 1. The composition of the accumulated distillate, x_d, which is measured with the sampling time T_s, is given by:

$$x_d(t) = \frac{\sum_{i=1}^{p} x_i(t) M_i(t) - x_i(0) M_i(0)}{M_1(t) - M_1(0)} \tag{17}$$

Also, it is assumed that the amount of distillate available at final time, $M_1(t_0) - M_1(t_f)$, is measured.

p	10		T_s	0.1	h
α	1.6		$M_1(0)$	100	kmol
M_i	0.2	kmol	$x_1(0)$	0.5	kmol/kmol
M_c	2	kmol	$x_i(0)$	0.5	kmol/kmol
V	15	kmol/h	$x_c(0)$	0.5	kmol/kmol
t_f	10	h	$x_{d,des}$	0.9	kmol/kmol

Table 1. Model parameters and initial conditions, $i = 2, \cdots, p$

The optimization problem consists of maximizing the quantity of distillate subject to a quality constraint at final time:

$$\max_{f_d(t)} \; J = M_1(t_0) - M_1(t_f) \tag{18}$$

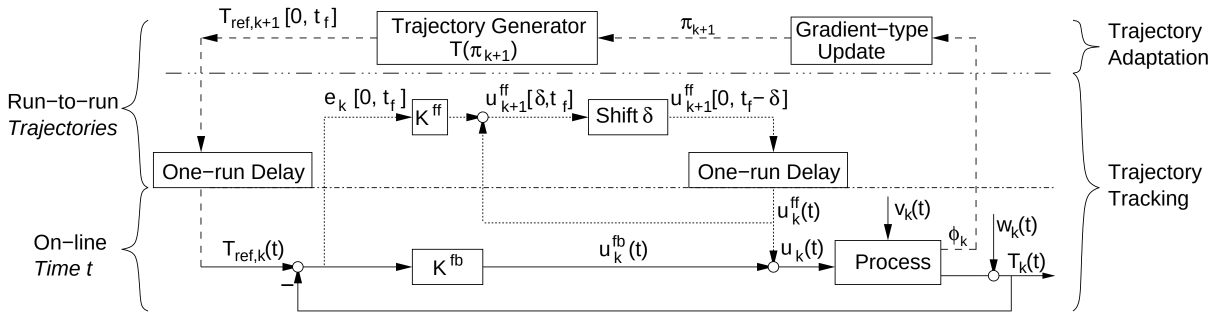

Fig. 1. Dynamic optimization scheme using within-run tracking (—), run-to-run update of feedforward inputs (···), and run-to-run trajectory update (- - -).

$$
\begin{aligned}
s.t. \quad & \text{Diff. Alg. Equations (12)-(17)} \\
& 0 \le f_d(t) \le 1 \\
& x_d(t_f) \ge x_{d,des}
\end{aligned}
$$

where $x_{d,des} = 0.9\ kmol/kmol$ is the desired distillate composition at the final time $t_f = 10\ h$.

The optimal solution consists of 3 intervals (Welz *et al.*, 2002):

(1) Start-up phase with full reflux, $f_d = 0$.
(2) Distillation phase, $f_d \in (0, 1)$, where a compromise between quality and quantity is sought.
(3) No reflux ($f_d = 1$) in order to remove high purity distillate from the condenser.

The third interval, whose effect is negligible in the given example, is not considered here. Let t_s be the switching time between the first two intervals. f_d in the second interval is approximated by a linear profile with the parameters $f_d(t_s) = l_1$ and $f_d(t_f) = l_2$ (Figure 2).

In order to obtain a realistic test scenario, the following uncertainty is considered:

- Parametric uncertainty: Fixed but unknown relative volatility in the range $\alpha = [1.4\ 1.7]$.
- Perturbation: Boilup rate equally distributed in the range $V = [13\ 17]\ kmol/h$, changed every $0.5\ h$.
- Measurement noises: Product composition x_d with 5% multiplicative gaussian noise, distillate quantity $J(t_f)$ with 3% multiplicative gaussian noise.

The value $\alpha = 1.6$ is used in all simulations. However, this value is not disclosed to the various optimization schemes. In order to be representative, the measured values of J and $x_d(t_f)$ are averaged over 50 realizations of the perturbation and measurement noises. For each scheme, a backoff b is added to the reference trajectory $x_{d,ref}$ so that the terminal constraint is met in 99% of the cases. It creates an offset from the original trajectory towards a higher distillate purity. This backoff is computed iteratively as in Srinivasan *et al.* (2003*a*). The better the tracking performance and disturbance rejection, the smaller the backoff.

6.2 Robust optimization

A feasible, but conservative, solution is calculated as the optimal solution for the smallest value of α, i.e. $\alpha^{cons} = 1.4$ and the expected value of V, i.e. $\bar{V} = 15\ kmol/h$. The numerical values for this conservative solution are $t_s^{cons} = 1.415\ h$, $l_1^{cons} = 0.1020$, $l_2^{cons} = 0.1229$ and $J^{cons} = 14.48\ kmol$. For the sake of comparison, if the value of α were known and if there were no variations in V, the amount of distillate would be much higher, $J^* = 27.89\ kmol$.

6.2.1. Open-loop implementation of f_d^{cons} (OL): When the conservative input is applied open loop to the true system with $\alpha = 1.6$, the separation is easier, and purer distillate is produced, $x_d(t_f) = 0.967\ kmol/kmol$ (Figure 2). However, the amount of distillate is $J^{OL} = 14.34\ kmol$, which is quite far from the ideal value that could be obtained with $\alpha = 1.6$. Thus, it is possible to increase the quantity of distillate by reducing its quality.

Note that the difference between J^{cons} and J^{OL} is due to the difference in the values for V (constant at $\bar{V} = 15\ kmol/h$ vs. equally distributed in the range $[13\ 17]\ kmol/h$).

6.3 Measurement-based optimization

6.3.1. No feedback, run-to-run adaptation of f_d^{ff} (ILC): The idea is to track the conservative profile $x_{d,ref}^{cons}[0, t_f]$. This reference profile is approximated by a piecewise-linear signal $x_{d,ref}(t)$ with $x_{d,ref}(t_s) = x_{d,ref}^{cons}(t_s) + b$ and $x_{d,ref}(t_f) = x_{d,des} + b$, where b is the backoff. Since $x_d(t)$ can only be measured once distillate has been collected, feedback control is only applied after a delay of one sample in the second interval. The adaptation law is similar to (7):

$$f^{ff}_{d,k+1}[t_s, t_f - \delta] = f^{ff}_{d,k}[t_s + \delta, t_f] + K^{ff} e_k[t_s + \delta, t_f] \quad (19)$$

where $e(t) = x_{d,ref}(t) - x_d(t)$. The run-to-run gain $K^{ff} = 0.1\ kmol/kmol$ and the shift $\delta = 1\ h$ were determined as a compromise between robustness and performance. With the backoff $b^{ILC} = 0.0140\ kmol/kmol$, the distillate quantity $J^{ILC} = 24.05\ kmol$ is obtained after 30 runs.

6.3.2. On-line adaptation of f^{fb}_d (FB): Here, the reference profile $x_{d,ref}[t_s, t_f]$ is tracked using a PI controller :

$$f_d(t) = f^{cons}_d(t) - K_p \left(e(t) + K_i \int_{t_s}^{t} e(\tau) d\tau \right) \quad (20)$$

where the conservative input f^{cons}_d is used as the feedforward term. The tracking performance using the parameters $K_p = 8\ kmol/kmol$ and $K_i = 0.02\ 1/h$ is shown in Figure 2. The error does not go to zero since integral action is not sufficient to drive it to zero within the finite batch time. Also, increasing the gains for error reduction causes instability. With this strategy, the backoff $b^{FB} = 0.0042\ kmol/kmol$ has to be introduced to meet the terminal constraint in 99% of runs, and the distillate quantity $J^{FB} = 25.41\ kmol$ is obtained.

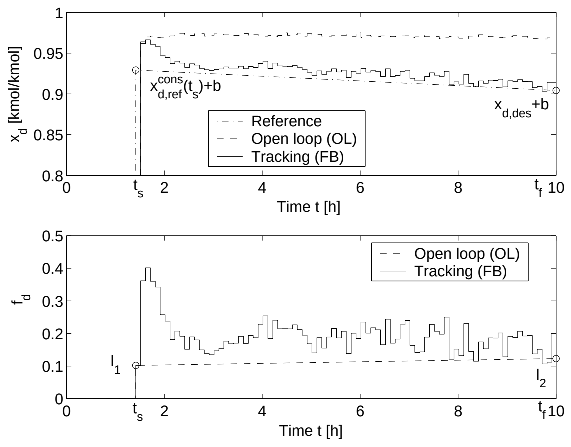

Fig. 2. Open-loop implementation (OL, - - -) and tracking of $x_{d,ref}$ (FB, —).

6.3.3. On-line and run-to-run adaptation of f_d (FB+ILC): Both the feedforward and feedback parts are adapted according to the laws (19) and (20), respectively. Since the effect of within-run disturbances are compensated by feedback, the backoff can be reduced to $b^{FB+ILC} = 0.0057\ kmol/kmol$ compared to $b^{ILC} = 0.0140\ kmol/kmol$ when feedback was not used. Also, the convergence is much faster with feedback, and the distillate quantity $J^{FB+ILC} = 25.53\ kmol$ is obtained after 4 runs. Figure 3 shows an improved tracking performance compared with that of Figure 2.

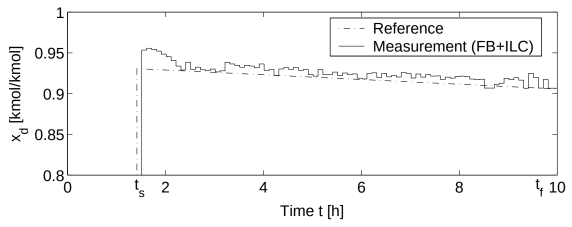

Fig. 3. On-line tracking of $x_{d,ref}$ upon convergence of the ILC iterations (FB+ILC).

6.3.4. On-line feedback and trajectory adaptation (FB+TAd): The reference trajectory $x_{d,ref}[t_s, t_f]$ is parameterized using t_s and $x_{d,ref}(t_s)$ since $x_{d,ref}(t_f)$ is fixed by the constraint $x_{d,des}+b$. Furthermore, since $x_{d,ref}(t_s)$ has very little influence on J, only t_s needs to be adjusted.

Starting from the conservative input f^{cons}_d, the sensitivity of J with respect to the switching time t_s is evaluated experimentally. The following update law is then used:

$$t_{s,k+1} = t_{s,k} - K_t \frac{J(t_{s,k}) - J(t_{s,k} - \Delta t_s)}{\Delta t_s} \quad (21)$$

where $\Delta t_s = 0.1\ h$ is the step size for gradient evaluation, and $K_t = 0.008\ h^2/kmol$ is the gain of the update law. With the backoff $b^{FB+TAd} = 0.0042\ kmol/kmol$, the distillate quantity $J^{FB+TAd} = 25.61\ kmol$ is obtained.

6.3.5. Global scheme (FB+ILC+TAd): The combination of the three adaptations (using the same parameter values in the update laws) results in the distillate quantity $J^{FB+ILC+TAd} = 25.78\ kmol$ after 4 runs.

6.4 Comparison of the various strategies

Table 2 provides a comparison of the various implementation strategies. It is seen that the improvement is considerable between the open-loop conservative strategy (OL) and the measurement-based approaches. This means that tracking of even the conservative reference $x^{cons}_{d,ref}[0, t_f]$ leads to fairly good results.

ILC without on-line feedback cannot compensate the effect of within-run perturbations, and a larger backoff is necessary, which limits the performance. Since, in addition, on-line feedback significantly increases the rate of convergence of ILC schemes, it is an important aspect of the global scheme. Finally, adaptation of the reference results in further improvement.

Strategy		J	b
		[kmol]	[kmol/kmol]
Open loop f_d^{cons}	(OL)	14.34	-
Adaptation of f_d^{ff}	(ILC)	24.05	0.0140
Adaptation of f_d^{fb}	(FB)	25.41	0.0042
Adaptation of f_d	(FB+ILC)	25.53	0.0057
Feedback + traj. ad.	(FB+TAd)	25.61	0.0042
Global scheme	(FB+ILC+TAd)	25.78	0.0057

Table 2. Comparison of the various optimization strategies in terms of cost J and necessary backoff b for a terminal constraint satisfaction of 99%.

Note that the implementation schemes presented here do not use any prior knowledge of the value of the uncertain parameter, except for its range that is used to compute the conservative input.

7. CONCLUSION

This paper has considered the class of optimization problems where the optimal solution is characterized by active terminal constraints. The main idea is to track appropriate references on-line in order to meet the terminal constraints, and to adapt these references on a run-to-run basis using experimental sensitivity information. Furthermore, the tracking performance can be improved by updating the feedforward inputs using ILC. On the example of a simulated batch distillation column, the performance could be improved significantly compared to robust optimization, which is the approach typically used in industry.

REFERENCES

Amann, N., D.H. Owens and E. Rogers (1996). Iterative learning control for discrete-time systems with exponential rate of convergence. *IEE Proc.-Control Theory Appl.* **143**(2), 217–224.

Arimoto, S., S. Kawamura and F. Miyazaki (1984). Bettering operation of robots by learning. *J. Robotic Syst.* **1**(2), 123–140.

Bien, Z. and K.M. Huh (1989). Higher-order iterative control algorithm. *IEE Proc.-Control Theory Appl.* **136**(3), 105–112.

Chen, Y. and C. Wen (1999). *Iterative Learning Control.* Lecture Notes in Control and Information Sciences. Springer, London.

Francois, G., B. Srinivasan and D. Bonvin (2002). Run-to-run optimization of batch emulsion polymerization. In: *IFAC World Congress.* Barcelona, Spain. pp. 1258–1263.

Hideg, L.M. (1996). Stability and convergence issues in iterative learning control: Part II. In: *IEEE Int. Symp. Intelligent Control.* Dearborn, MI. pp. 480–484.

Jang, T.J., C.H. Choi and H.S. Ahn (1995). Iterative learning control in feedback systems. *Automatica* **31**(2), 243–248.

Kuc, T.Y., K. Nam and J.S. Lee (1991). An iterative learning control of robot manipulators. *IEEE Trans. Rob. Autom.* **7**(6), 835–842.

Lee, J.H., K.S. Lee and W.C. Kim (2000). Model-based iterative learning control with a quadratic criterion for time-varying linear systems. *Automatica* **36**(5), 641–657.

Moore, K.L. (1993). *Iterative Learning Control for Deterministic Systems.* Springer, London.

Moore, K.L. (1999). *Iterative learning control: An expository overview.* pp. 151–214. Applied and computational control, signals, and circuits. Birkhäuser, Boston.

Park, K.H., Z. Bien and D.H. Hwang (1998). Design of an iterative learning controller for a class of linear dynamic systems with time delay. *IEE Proc.-Control Theory Appl.* **145**(6), 507–512.

Phan, M.Q. and R.L. Longman (2002). Higher-order iterative learning control by pole placement and noise filtering. In: *IFAC World Congress.* Barcelona, Spain. pp. 1899–1904.

Roberts, P.D. and T.W.C. Williams (1981). On an algorithm for combined system optimization and parameter estimation. *Automatica* **17**, 199–209.

Srinivasan, B. and D. Bonvin (2002). Interplay between identification and optimization in run-to-run schemes. In: *American Control Conference.* Anchorage, Alaska. pp. 2174–2179.

Srinivasan, B., D. Bonvin, E. Visser and S. Palanki (2003*a*). Dynamic optimization of batch processes: II. Role of measurements in handling uncertainty. *Comp.Chem.Eng.* **27**, 27–44.

Srinivasan, B., S. Palanki and D. Bonvin (2003*b*). Dynamic optimization of batch processes: I. Characterization of the nominal solution. *Comp.Chem.Eng.* **27**, 1–26.

Sun, M. and D. Wang (2003). Initial shift issues on discrete-time iterative learning control with system relative degree. *IEEE Trans. Autom. Contr.* **48**, 144–148.

Wang, D. (2000). On D-type and P-type ILC designs and anticipatory approach. *Int. J. Control* **73**(10), 890–901.

Welz, C., B. Srinivasan and D. Bonvin (2002). Evaluation of measurement-based optimization schemes for batch distillation. In: *15th IFAC World Congress.* Vol. T–MO–M11. Barcelona, Spain.

Xu, J.X., X.W. Wang and L.T. Heng (1995). Analysis of continuos iterative learning control systems using current cycle feedback. In: *Proceedings of the American Control Conference.* Seattle, Washington. pp. 4221–4225.

Yabuki, Y. and J.F. MacGregor (1997). Product quality control in semibatch reactors using midcourse correction policies. *Ind.Eng.Chem.Res.* **36**, 1268–1275.