Review of Real Time Optimization in the Chemical Process Industries

M. R. Naysmith and P. L. Douglas*

Department of Chemical Engineering, University of Waterloo, Waterloo, Ontario, CANADA N2L 3G1

Real Time Optimization (RTO), or On-Line Optimization, is the continuous reevaluation and alteration of operating conditions of a process so that the economic productivity of the process is maximized subject to operational constraints. The purpose of this paper is to review the available literature in the area of real time optimization, with particular attention to the methodologies being used at present. The general components which make up a real time optimizer, the process model, objective function, optimization algorithm, data reconciliation and gross error detection, are also reviewed. Finally implementation of real time optimizers is covered including user interfacing and on-going maintenance of the system.

Introduction

The terms On-Line Optimization and Real Time Optimization (RTO) are used to indicate the continuous reevaluation and alteration of operating conditions of a process so that the economic productivity of the process is maximized subject to operational constraints [1]. Optimization of process operations is of considerable interest in industry due to increasing global competition and tightening product requirements [2, 3]. On-line optimization is an appealing concept because it is at the level of the control hierarchy at which business decisions are integrated into the operation. Advances in the speed and power of computers at lower costs are making on-line optimization a more cost effective method of improving plant performance.

Latour [4] lists the benefits from steady-state optimization as:

- Improved product yields and/or quality.
- Reduced energy consumption and operating costs.
- Increased capacity of equipment, stream factors.
- Less maintenance cost and better maintenance of instrumentation.
- More efficient engineers and operators as a tool for process troubleshooting and operation.
- Tighter, lower cost process designs if new plant designs include a real time optimizer.

Components of a Real Time Optimizer

A general structure for real time optimization based on the one shown in Hanmandlu et al. [5] is given in Figure 1. This structure is also similar to those presented in several other papers [1,3,6-12].

* *Author for correspondence.*

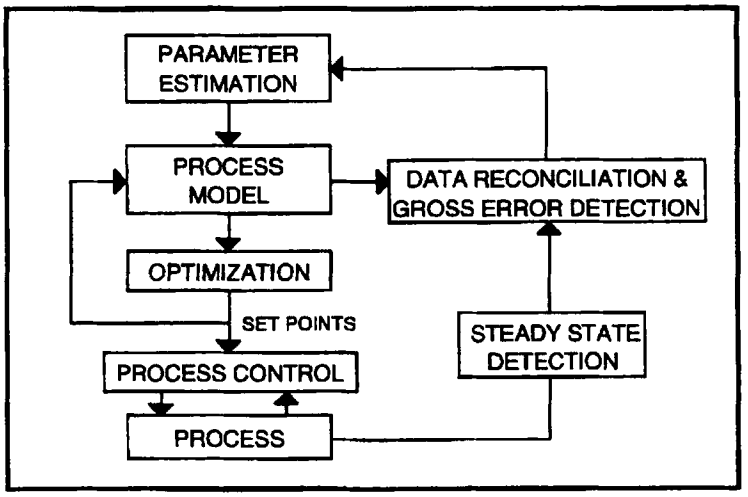

Figure 1. General structure for a model based real time optimization.

The general steps taken in one cycle of the real time optimizer shown in Figure 1 are:

1) *Steady-State Detection:* This detects when the process is close enough to steady-state to allow the real time optimizer to model the process with reasonable accuracy. When the plant is determined to be at steady-state, data from the process is passed onto the optimizer.
2) *Data Reconciliation and Gross Error Detection:* All measured data are subject to error. Data reconciliation adjusts the measured variables (e.g. energy, mass, etc.) and, if possible, estimates any unmeasured variables for the process so that they satisfy the balance constraints. Gross error detection takes suitable corrective actions to rectify any gross errors found in the process data [13]. This adjusted data from the plant is then passed onto the next step.
3) *Parameter Estimation:* The process of updating the model parameters to the adjusted process data available. These model parameters are then used in the process model for the optimization of the set points.
4) *Process Model:* The model of the process which represents the effects of changes made to the process by the optimization algorithm while it is iterating to an optimum solution.
5) *Optimization:* The optimization algorithm used to determine the optimum set points for the process given an objective function and process constraints.
6) *Updating of Process Set Points:* The optimum set points for the process found by the optimizer are returned to the plant. Once the plant has returned to steady-state, another cycle of the real time optimizer is performed to keep the plant operating at its optimum.

Some of the above steps are further detailed in the following subsections.

Process Model

The process model used in real time optimization must represent changes in the manipulated variables of the process given the updated set points calculated by the

optimization algorithm. The model approximates the effect of these variables on the dependent variables, which are usually those that are constraints or those that have a significant economic impact, such as yields, material balance or utility usage. In virtually all practical cases, the model is based on steady-state (rather than dynamic) relationships and is non-linear [3]. The choice of measurements to be used for matching the model parameters to the plant operations must be made carefully to avoid an infeasible problem. Bailey et al. [6] chose their variables so that each piece of equipment had enough associated specifications for the available degrees of freedom. Development of process models for optimization is covered in detail by Edgar and Himmelblau ([14], Chapter 2).

Fatora and Ayala [7] discuss the application of open equation-based models. Instead of posing an equation in its closed-form, for example Equation (1) below, the equation should be formulated in its open residual format as shown in Equation (2).

$$Q=mC_p(T_2 - T_1) \tag{1}$$

$$R=Q - mC_p(T_2 - T_1) \tag{2}$$

A simultaneous equation solving and optimization package will manipulate the unknowns such that the residual terms, R, are driven to zero. This method gives the same results as the closed-form equation. The advantage of open-form equations is that the model equations are completely separate from the convergence algorithm unlike the closed-form models which have a tailored solution algorithm built into the model. Such models are easily modified to account for process changes in the plant since the convergence scheme is separated from the model. The maintenance staff need not recode the convergence scheme each time the model is changed. The open equation-based approach to process simulation, while less robust with respect to finding a solution than closed-form equations models, has been used successfully in industry for several years. For these reasons, open-form equations are the most commonly used in real time optimization applications.

Two papers by Forbes et al. [8, 9] give some criteria for helping choose a model adequate for real time optimization. The quality of the model has a large effect on the success of the real time optimization system. They give a point-wise model adequacy criterion as the ability of the model to have an optimum that coincides with the true plant optimum. Analytical methods for checking the point-wise model adequacy were developed using reduced space optimization theory, and from these more practical numerical methods were developed for use on industrial problems [9].

Bailey et al. [6] briefly cover some modeling issues affecting their non-linear optimization of a hydrocracker fractionation plant. They found that large equation-based models are sensitive to scaling. However choosing appropriate units avoided numerical problems, even if the resulting units were unusual (e.g. teraJ per day). Numerical stability of the model can also be a problem and care should be taken to avoid division as an operator. Also, during the optimization, bounds must be placed on certain variables to prevent the solution from moving too far from the starting point. This is because it takes a finite amount of time to move the plant to a new operating point, therefore the setpoints should not be moved a great deal on each run of the optimizer. They suggest choosing bounds on the variables to reflect the allowable change in 1 to 3 hours. The maximum change should depend on the length

of time it would take the control system to take the maximum step and bring the plant back to steady-state.

A common method for more rapidly computing the optimum is an "inside-out" algorithm first proposed by Boston and Britt [15] to solve a single stage flash algorithm. In this method the major iterations deal with updating the parameters for the simplified models and matching their properties with a rigorous model. Therefore, since the more computationally expensive model is only evaluated in the outer loop, considerably less effort is required than with a conventional procedure. However, Biegler et al. [16] and Forbes et al. [9] point out that when this method is applied to optimization problems, this approach based on simplified models can converge to a solution that may not necessarily correspond to the optimum predicted when only rigorous methods are used. They give two observations to help avoid this problem when the "inside-out" type of algorithm is used:

(a) A necessary condition for an appropriate simplified model for optimization is that it recognizes the rigorous model optimum as a Karush-Kuhn-Tucker (KKT) point, that is an optimal point.
(b) A sufficient condition for an appropriate simplified model is that it matches all the gradients of the rigorous model at all points of interest.

Several papers cover the use of commercially available process simulation packages used as the process model, although there is no evidence to date of this method being used in industry. Two types of simulation packages are available. The first are open-form, equation oriented or simultaneous process simulation packages, for example, SPEEDUP and MASSBAL. Although these simulators are more flexible, they generally require a high level of expertise to learn and use, have poor user interfaces, and are not widely used in industry. The second type is the closed-form or sequential modular process simulation packages, such as, ASPENPLUS, PROII and HYSIM. These packages are widely used in industry due to their proven reliability and ease of use, however, they are less efficient with respect to speed and computer power required [17]. In addition they require special techniques and insights to develop and change the optimization strategies for on-line operation due to the sequential modular architecture.

Gallier and Kisala [17] used ASPEN, a sequential modular package, in their study of real time optimization. They cover two methods where the simulator is treated as a "black-box" and as an infeasible path optimization using the Sequential Quadratic Programming (SQP) method. It was found that the latter is preferable as the constraints at each iteration need not be satisfied. This allows the optimization of the process model to be completed in as few as five simulation time equivalents. Macchietto et al. [11] used the equation-based SPEEDUP process simulation package as the process model and also to perform the data reconciliation, gross error detection and optimization steps. Dynamic real time optimization is also possible using the SPEEDUP package. Shewchuk and Morton [12] outline the use of SACDA's MASSBAL process simulation packages as the model in their optimizer package. This package is also capable of both steady-state and dynamic optimization.

Objective Function

The objective function is derived from the plant model, and is to be either maximized or minimized through changes in the independent variables. The objective function for real time optimization normally represents the economic model of the process, usually limited to variable effects and ignoring fixed costs such as labour, overheads, etc. A typical objective function given in Darby and White [3] is:

$$\text{Objective} = \text{Product value} - \text{Feed costs} - \text{Utility costs} + \text{Other variable economic effects} \tag{3}$$

Bailey et al. [6] cover objective function development in detail. Feed and product prices must be chosen carefully since different values are often associated with the same material. For example the product might have a contract price, spot market price and also be an intermediate stream sent to another part of the plant for further processing. The case of a stream being an intermediate stream and a product is particularly difficult to determine an accurate value for an inter-process stream, as many planning models only consider feeds and finished products. One remedy for this problem is to include a simple model of the finishing process in the objective function thereby giving an approximate value to the inter-process stream.

Since the optimizer will be estimating derivatives of the objective function with respect to operating variables, it is important that the objective function be continuous. Discontinuities arising from streams with multiple prices should be treated specially (e.g. the stream has a contractual price and a spot price). Often an average price for the stream is not acceptable, and the discontinuity caused needs to be handled by a method such as an integer programming technique.

Bailey et al. [6] also show how they developed their objective function for the real time optimization of a hydrocracker fractionation plant in some detail. Edgar and Himmelblau ([14], Chapter 3) cover the development of objective functions including methods for measuring profitability and cost estimations.

Optimization Algorithm

The optimization algorithm uses the process model and the objective function to solve for the new optimum set points for the plant. The important considerations in choosing an optimizing algorithm are computational requirements, number of objective function evaluations and robustness. The general form of the constrained optimization problem [7] using the process model and the objective function is:

$$\begin{aligned} &\min_{z} \text{ (or max) } F(z, p) = 0 \\ &\text{subject to} \quad f(z, p) = 0 \\ &\qquad\qquad z_l \le z \le z_u \end{aligned} \tag{4}$$

where $F(z,p)$ = the objective function;

$f(z,p)$ = the vector of n equality constraints (from the process model);

p = fixed variables or parameters (e.g. fouling factors, cost coefficients, etc.)

Changes in these variables will change the optimum values of z, where: z = the process variables which can be varied to optimize F. The number of these variables must be greater than n.

z_l, z_u = the lower and upper bounds on the variables z respectively (which give the inequality constraints on the model).

Methods on how to solve the above constrained optimization problem are covered in several texts [14,18,19]. A great deal of literature is available on optimization algorithms, and generally is considered to be a mature area of mathematics.

Presently, the most common type of algorithm reported in the literature for solving this type of problem are Sequential (or Successive) Quadratic Programming (SQP) techniques. The SQP method approximates the objective function by a quadratic function, and the constraints by linear functions, so that quadratic programming can be used recursively to find a search direction minimizing the objective function. At each outer iteration (or quadratic program solution) the SQP method constructs these approximations using information about the values of the variables and their derivatives with respect to the decision variables [12,14,19,20,21]. Edgar and Himmelblau ([14], p357) provide a table of the sources of computer codes for SQP methods. Lucia and Xu [20] provide an extensive review of the SQP algorithms available, particularly with respect to chemical processes and large-scale problems. Fatora and Ayala report the use of an SQP algorithm in their real time optimization with over 36,000 variables and 31,000 equations.

Bailey et al. [6] use the MINOS routine developed by Stanford. MINOS is a sequential quadratic programming technique which also makes use of the Simplex method and a projected Lagrangian algorithm. The MINOS method is detailed by Murtagh and Saunders [22], and has been shown to solve large problems of around 5000 variables and 4500 equality constraints successfully.

The DICOPT routine is discussed in a paper by Kocis and Grossman [23]. DICOPT is an outer-approximation/equality relaxation algorithm specifically designed for process systems involving mixed-integer nonlinear programming (MINLP) due to nonlinearities in the models. The algorithm was tested successfully using problems of up to 528 variables and 1171 constraints, including 74 binary variables.

Data Reconciliation and Gross Error Detection

The key features of the process data problem are as follows:

1) All measurements are subject to errors. Random errors are assumed to be normally distributed and have zero mean. Gross errors are caused by non-random events such as instrument biases, malfunctioning measuring devices, inaccurate or incomplete process models and process leaks [13,24,25]. Figure 2 gives an illustration of the probabilities involved for random and gross errors.
2) Not all process variables are measured for reasons of cost, inconvenience or technical infeasibility.
3) There is data redundancy in the sense that there are more measurements (or data) available than needed if the measurements were not subject to errors. This is also referred to as spatial redundancy [24,26,27].

4) Since most process data are being sampled continuously and regularly at great frequencies, there is also data redundancy if the process conditions are truly at steady-state. This is usually referred to as temporal redundancy.

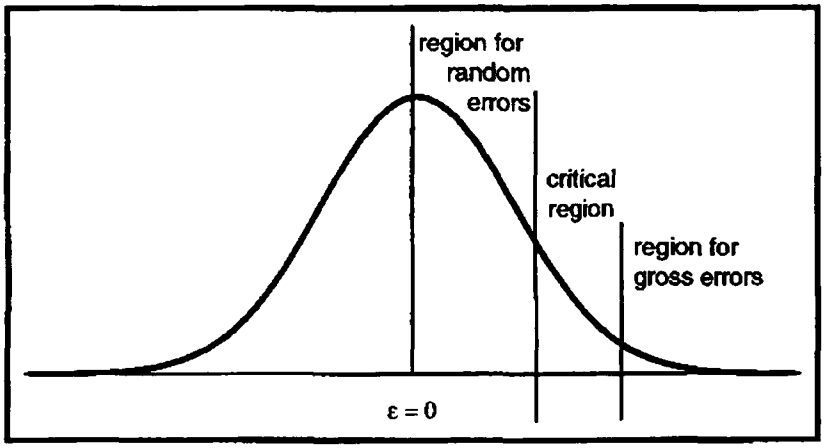

Figure 2. Error probability distribution for process measurements [13].

The adjustment of measured variables for random errors and, if possible, the estimation of unmeasured variables so that they satisfy the balance constraints is known as the data reconciliation problem. Detection of the presence of any gross errors so that suitable corrective actions can be taken is known as the gross error detection problem [28]. Collectively, these two problems are sometimes called data rectification [29]. The objectives of data rectification are to improve confidence in the measurements, estimate unmeasured streams, identify meter faults and to identify process losses [30]. A simplified view of the three basic steps to data rectification is shown in Figure 3. The first step, variable classification, involves determining which variables are observable and/or redundant [25]. In practice, steps two and three are often used iteratively to reconcile the data and remove gross errors.

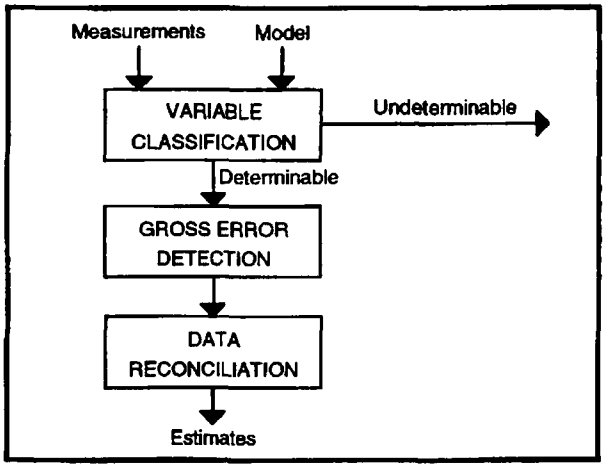

Figure 3. General steps for processing measurement data [25].

There is very little literature available comparing different methods of data reconciliation and gross error detection. Mah [24] provides probably the best review in this area to date covering several methods. However, development of better nonlinear and dynamic data reconciliation and gross error detection methods is still an on-going area of research.

I Linear, Steady-State Data Reconciliation

The simplest case is linear, steady-state data reconciliation. This is covered by Mah ([24], 9-1-2) and Crowe et al. [31]. The basic model used for the data reconciliation in the absence of gross errors [24] is:

$$\mathbf{y} = \mathbf{D}\mathbf{x} + \varepsilon \tag{5}$$

where $\mathbf{y}$ = $(s \times 1)$ vector of measured variables;
$\mathbf{D}$ = $(s \times p)$ measurement matrix which relates $\mathbf{y}$ to $\mathbf{x}$ (generally the measured variables will not be the same as the state variables);
$\mathbf{x}$ = $(p \times 1)$ vector of state variables;
ε = $(s \times 1)$ vector of random measurement errors.

The linear constraints given by stoichiometric constraint, energy, mass and other balances [24] are generalized into the form:

$$\mathbf{A}_1\mathbf{x} + \mathbf{A}_2\mathbf{u} = \mathbf{c} \tag{6}$$

where $\mathbf{A}_1$ = $(n \times p)$ matrix of coefficients of known constants;
$\mathbf{A}_2$ = $(n \times m)$ matrix of coefficients of known constants;
$\mathbf{u}$ = $(m \times 1)$ vector of parameters not directly related to the measurements through the basic model equation;
$\mathbf{c}$ = $(n \times 1)$ vector of known constants.

The data reconciliation problem is then the least squares estimation of $\mathbf{x}$ and $\mathbf{u}$ subject to the constraints equation. Crowe et al. [31] use a projection matrix to obtain a reduced set of balance equations, and also include a statistical test for the removal of gross errors. Mah ([24], 9-1-3) also covers the decomposition and solution of these equations in detail.

II Nonlinear, Steady-State Optimization

Nonlinear, steady-state data reconciliation is the extension of the above method to include nonlinear constraints. These occur when variables are measured indirectly by other physical properties, e.g. concentration by density, pH or thermal conductivity. Bilinear constraints occur when variables appear in two balances, e.g. temperatures measured along with concentrations will appear in both energy and component material balances. Bilinear constraints also occur when a stream is split into two or more streams of the same temperature and composition. No general analytical solution is available to bilinear and nonlinear data reconciliation problems. Nonlinear constraints also make it more difficult to determine whether unmeasured variables may be estimated, and how to decompose a problem with missing measurements ([24], 9-1-4).

Crowe [32] handles the nonlinear constraints by extending the method outlined in Crowe et al. [31] by applying two successive projection matrices to the equations

defining conservation laws and other constraints. Also, no restrictions are placed on the location of measurements, nor upon what is measured in any one stream. The construction of the projection matrices is direct and each requires the inversion of a matrix that is also needed for subsequent steps in the solution.

A paper by Serth et al. [33] adds details on the choice of regression variables used in nonlinear data reconciliation by defining primary and secondary variables. Total flow and component flows are examples of primary and secondary flows respectively. Since material balances are linear in secondary variables, they can be exploited to develop a particularly efficient data reconciliation procedure [32]. However, they do not yield maximum likelihood estimates for the true values of primary measured variables. This in turn could affect the error detection procedure, tending to increase the number of Type I errors (Definition of Type I error is given in the Gross Error Detection section).

Data reconciliation using process simulation software has been addressed in two papers, both of which used equation based simulation packages. Macchietto et al. [11] used both SPEEDUP and a package using simplified process models called DEBIL. SPEEDUP was found to be slower and use more computational resources, but was more flexible. Stephenson and Scewchuk [34] detail the methodology by which MASSBAL solves the data reconciliation problem. Applications to a pulp mill screening and cleaning system are described.

Takiyama et al. [35] outline the use of SEBDARM (Sensor Based Data Reconciliation Method). The method is based on the direct use of measurement variables rather than using balanced variables (conventional method). This has the advantage of having the reconciled data not being influenced by the number of reduced balance equations. Their method, however, has only been tested on pilot plants to date.

III Dynamic Data Reconciliation

This is relatively new area of research in the field of data reconciliation. In many practical situations the process conditions are continuously undergoing changes and steady-state is never truly reached. Generally the data reconciliation problem in this case is reduced to a discrete Kalman filter for a quasisteady-state problem [36].

Darouach and Zasadzinski [36] develop a new dynamic algorithm based on steady-state reconciliation leading to a recursive scheme useful for real time processing of data. The method was shown to reduce computational problems associated with discrete Kalman filters, especially for singularities and round off error situations. Rollins and Devanatham [37] refine the method proposed by Darouch and Zasadzinski [36] to reduce the computational time required, while preserving its properties of unbiased estimation. Liebman et al. [25] use nonlinear programming to help solve systems in highly nonlinear regions where the Kalman filter method may be inaccurate. The method given was found to be a more robust means for reconciling dynamic data, especially in the presence of inequality constraints. The method is demonstrated on a reactor example.

IV Gross Error Detection

Gross errors in the data invalidate the statistical basis of reconciliation due to their non-normality. One gross error present in a constrained least squares reconciliation

will cause a series of small adjustments to other measured variables. For these reasons, gross errors need to be identified and removed before data reconciliation is carried out. The most common techniques for detecting gross errors are based on statistical hypothesis testing of the observed or measured data. For any gross error detection method to work, there must be at least two alternative ways of estimating the value of a variables, for instance, measured and reconciled values ([24], 9-2). Crowe [38] and Mah [24] give detailed accounts of typical statistical gross error detection algorithms.

Two measures of the effectiveness of a gross error detection scheme are the terms Type I and Type II errors made by the detection scheme. Type I errors refer to wrongly identifying truly random errors to be gross errors. Type II errors refer to wrongly finding truly gross errors to be random errors [38]. In devising the tests for a gross error, the probability of a correct detection must be balanced against the probability of mispredictions.

The Maximum Power (MP) test for Gross Errors is detailed in two papers by Crowe [39,40]. The maximum power test has the greatest probability of detecting the presence of a gross error without increasing the probability of Type I error than any other test would on a linear combination of the measurements. Crowe [39] only deals with the linear case, while Crowe [40] extends the method to the bilinear case and retests the original constraints used.

The Generalized Likelihood Ratio (GLR) approach is detailed by Narasimhan and Mah [41] and Mah ([24], 9-2-5). The previously discussed methods only deal with gross errors from measurement or sensor biases. The GLR approach provides a framework for identifying any type of gross errors that can be mathematically modeled. This can be very useful for identifying other process losses such as a leak. The method and examples of its use are detailed reasonably fully by Narasimhan and Mah [41]. Narasimhan [42] shows that the Maximum Power Test detailed above is equivalent to the GLR approach for simple steady-state models.

V Combined Data Reconciliation and Gross Error Detection Methods

Two papers by Narasimhan and Harikumar [43,44] give details of incorporating bounds into data reconciliation. Most approaches to data reconciliation do not impose inequality constraints because the solution can no longer be obtained analytically, and the statistical analysis of the measurement residuals for gross error detection is difficult. This leads to an iterative cycle of data reconciliation and gross error detection phases to complete the data rectification. Narasimhan and Harikumar [43] develop the algorithm for solving the data reconciliation problem with bounds and produce the constraint and measurement residuals. In the second paper [44], the gross error detection procedures are detailed and the whole method is compared to existing methods. Their results indicate that their method performed better than currently available methods.

Tjoa and Biegler [13] develop a simultaneous data reconciliation and gross error detection method. They minimize an objective function that is constructed using maximum likelihood principles to construct a new distribution function. This distribution takes into account both contributions from random and gross errors. This method gives unbiased estimates in the presence of gross errors. Therefore simultaneously a gross error detection test can be constructed based on the new

distribution functions without the assumption of linearity of constraints. This method is particularly effective for nonlinear problems.

Terry and Himmelblau [29] detail the use of Artificial Neural Networks (ANN) for data reconciliation and gross error detection. The ANN is "trained" or "learns" to perform the data rectification based on a given model and constraints. The method is still being researched and the results look promising compared to models built from first principles.

Methods of Real Time Optimization

(i) Global or Centralized Approach to Real Time Optimization

This is the most common approach found in the literature. It involves optimizing the plant as a whole using one objective function, subject to constraints, and a model of the entire plant.

Several examples in the literature use very similar techniques of taking a rigorous steady-state model of the process which is then optimized using a nonlinear objective function and constraints. In all the cases listed below, the model appeared to be developed specifically for the real time optimization implementation. Examples of this general form were performed on the following plants:

- Hydrocracker fractionation plant [6]. The optimization algorithm was carried out using MINOS 5.1 on nonlinear mass and energy balances models and nonlinear constraints. Comparison is made to a rank-null space decomposition SQP (Sequential Quadratic Programming) optimization technique. The problem involved 2836 equations, 2891 variables and 10 degrees of freedom. Modeling issues were found to be problems with scaling of the problem, numerical stability, particularly when using the division operator, and minimizing the number of relinearizations of the model constraints required to find the optimum. Bounds were placed on the maximum amount any variable could be changed in a run of the real time optimizer to prevent large set point changes destabilizing the plant. Objective function development was also covered in detail.
- Olefins plant [7]. Rigorous, fundamental chemical engineering models were developed for the olefins plant. All of the models were written in the open equation form and were optimized using an SQP algorithm. The resulting complex nonlinear optimization problem contained over 36,000 variables and 31,000 equations, and approximately 20 minutes is taken to complete a full cycle of the real time optimizer on an unspecified "minicomputer". The real time optimizer was commissioned in February 1991, and took a total of three weeks to fully implement. The payback was less than one year. The observed benefits from installing the real time optimizer were:
  - Improvements of 5% to 10% of the value added by the process.
  - Consistently holding the process at production targets.
  - Decreased product variability.
  - Optimal handling of utility versus yield trade-off.
  - Many perceived bottlenecks were eliminated.
  - Energy savings.
  - Increased productivity by trending unit performance and monitoring of key parameters.

— More accurate off-line planning.

- Olefins plant [10]. This reference gave very few specifics on what methods they actually used in the real time optimization. They do however outline a very generalized method. The model was based on a set of modular blocks of successive plant sections. The advanced local control techniques used in the plant were also detailed in the paper.
- Hot acid leaching and strong acid leaching residue treatment plant in a zinc refinery [45]. The purpose of this real time optimization was to show the effectiveness of the robust parameter estimation method as outlined in Krishnan et al. [46] and the development of the rigorous steady-state model required. However, the reference mentions very little about the optimization itself.
- Ammonia process [47]. A rigorous plant model built from first principles was used for the ammonia unit handling 1600 t/day. They found that statistical regression type models were found to have significant disadvantages to the rigorous model approach. The model was performed in Exxon Chemical's Equation Manager and Solver (EMS) package using 5500 equations with 160 tear streams taking two person years' effort to set up.
- Packed-bed immobilized-cell reactor [48]. Used a recursive least squares algorithm and high pass filtering of the data to estimate the parameters in the discrete-time dynamic model which was optimized by a full Newton optimization algorithm using the objective function's curvature information. The process was relatively small compared to most of the other plants listed, although did involve very noisy process data.
- Refinery power plant [21]. The MESA nonlinear simulator was selected to model the power plant unit as it was specifically developed for power plants. Mobil's proprietary SQP optimization technology, MOPT, was used as the optimization algorithm. They ran the real time optimization algorithm every 30 minutes, but have the option of execution on demand. The system averages 20 updates to the plant per day. From an audit of the on-line power plant optimizer, savings of approximately $2.1 million/year or a 3% saving in fuel consumption were found.
- Pulp refining process [49]. They found that the model based control technique was dependent on three main factors:
  1) Robust, mechanistic models which accurately describe the effects of process conditions over a wide range of operating conditions.
  2) Tuned parameters for the refiner models so that the equations accurately represent the real process.
  3) Optimization and control techniques which allow the refiner models to be applied in a systematic fashion.

  They found the weaknesses of their optimization system were maintaining the flow of information to the model and ensuring the correct information is available. Maintenance of the system was found to require a dedicated effort from both the mill and vendor.

Jang et al. [1] took a two-phase approach to the global real time optimization method. The problem of plant operation is based on the idea of a moving time horizon. The method uses the present measurements and the measurements taken during the past period of time together with a partial knowledge of the plants physical and chemical

principles governing the plant to give an optimal time plan for the manipulation of the set points for a future period of time. The plant is modeled by either a steady-state or dynamic model depending on how stable the plant is. The two phase approach is summarized in Figure 4. The identification phase of the method is only carried out when the process has been significantly disturbed from the previous steady-state.

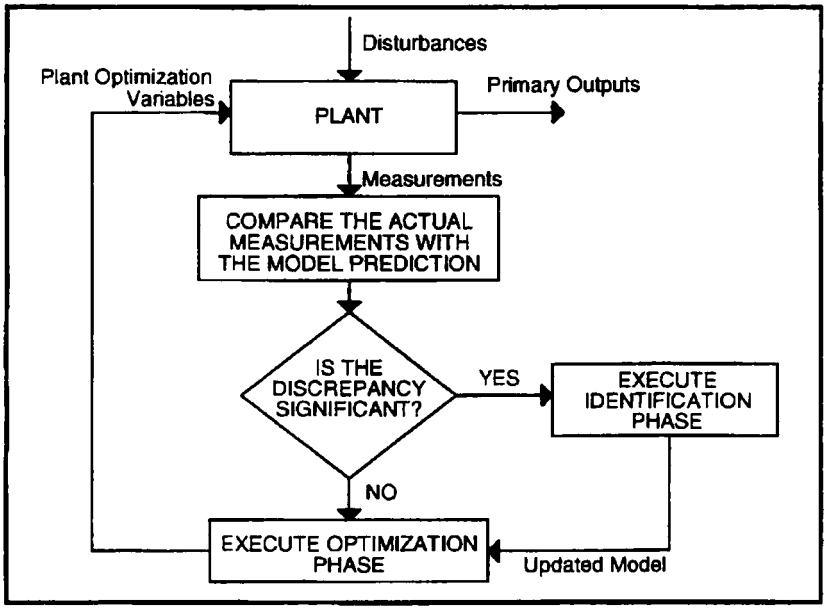

Figure 4. *Two-phase approach to on-line optimization [1].*

Gallier and Kisala [17] try to use process simulators as the process model using the SQP optimization algorithm to solve for the optimum. Their steps to create the real time optimizer, given the model of the plant, were to:

1) identify the objective function.
2) identify the degrees of freedom for the optimization.
3) identify the constraints.

They found the SQP method was very efficient at finding the optimum of the ammonia synthesis example they tested. However, they did not extend the method to a real plant situation nor mention the use of data reconciliation or gross error detection in their method.

Macchietto et al. [11] also used a process simulator in developing a real time optimization application. They used an equation based simulator, SPEEDUP, a data screening program called DEBIL and the SPEEDUP External Data Interface (EDI) as a communication mechanism. Figure 5 gives an outline of plant/control/simulator system. SPEEDUP has an advantage of having both successive quadratic programming (SQP) and MINOS optimization routines built into the simulator. They conclude that the SPEEDUP flowsheet optimizing package, using the built in external data interface, can be used successfully on-line with a plant control system. Only case studies are presented in the paper, and no evidence of an industrial application of this method of real time optimization was given.

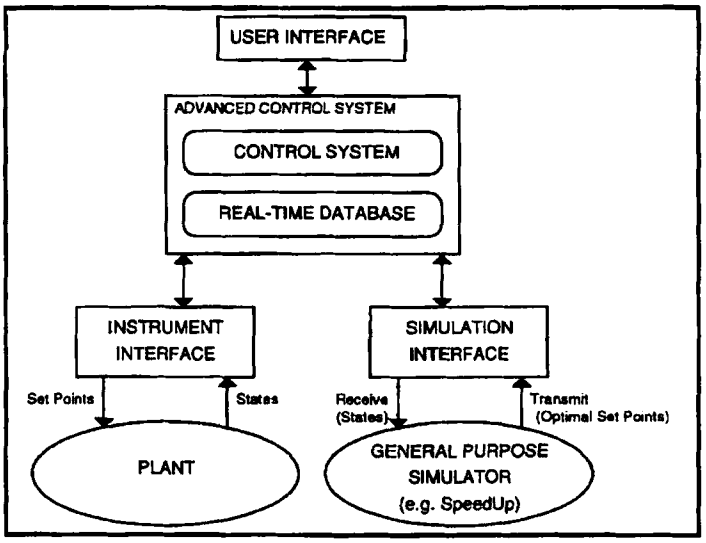

Figure 5. Integrated plant/control/simulator system [11].

(ii) Distributed Approach to Real Time Optimization

The distributed approach breaks down the overall optimization into several local optimizations which are coordinated by a unit optimization/coordination model. Figure 6 contrasts the distributed approach to the centralized or global approach [3]. The aim of the distributed approach is to decompose the large scale plant into subsystems thereby reducing the complexity of the original optimization problem [50]. The distributed approach is also sometimes referred to as modular or hierarchical optimization. Distributed real time optimization has been likened to a "bottom-up" implementation of optimization as opposed to the "top-down" centralized approach.

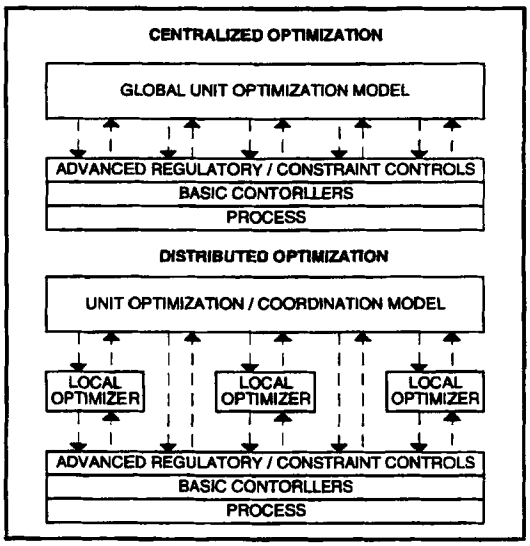

Figure 6. Centralized vs. distributed optimization [3].

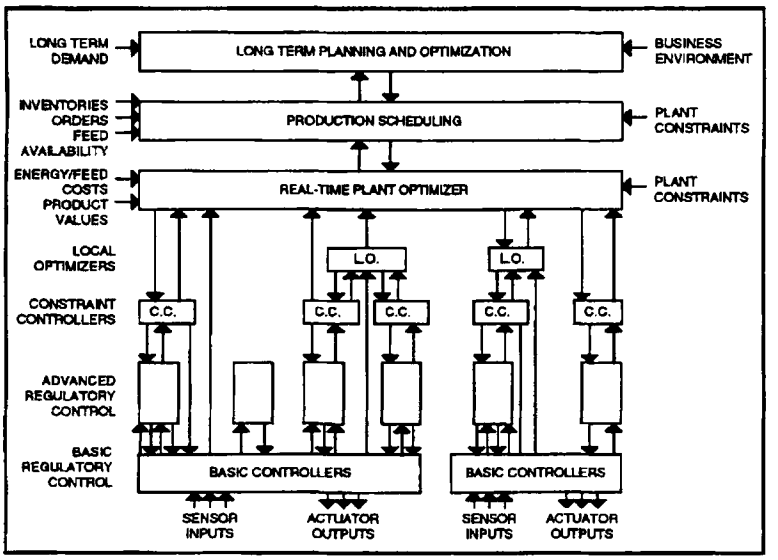

Figure 7. *Overall proces control systems structure [3].*

Darby and White [3] give a proposed overall control systems structure to deal with the distributed real time optimization shown in Figure 7. Some short examples are given to illustrate that the distributed approach will lead to the same answer as the global approach, but very little detail on implementation of the method is given.

Arkun and Stephanopoulos [50] give a very detailed method of how to perform optimization on a plant decomposed into its subsystems with functional uniformity and common objectives in terms of economics and operation. The subsystems are not independent of each other so interconnections between the subsystems are defined. Local optimizing controllers are designed for each subsystem, S_i, of the plant. If the plant has N subsystems, then the sub-problem: S_i, $i = 1,...,N$ is subject to the system constraints, as given by:

$$\min_{u_i m_i} l_i \quad = \Phi_i(x_i, m_i, u_i, d_i) - \lambda_i u_i + \sum_j \lambda_j^T Q_{ji}\, y_i \tag{7}$$

where d_i = vector of disturbances entering the subsystem S_i;
l_i = sub-Langrangian for a subsystem i;
m_i = vector of manipulated variables;
Q_i = incidence matrix denoting the interconnection among the subsystems;
u_i = vector of interconnection points;
x_i = vector of state variables;
y_i = vector of output variables;
λ_i = Lagrange multipliers for the interconnection constraints;
Φ_i = subobjective function.

The coordinator or overall optimizing problem for the plant attempts to satisfy the interaction between the subsystems by satisfying:

$$u_i = \sum_{j=1}^{N} Q_{ij}\, y_j \tag{8}$$

By performing this decomposition, and by properly formulating the subsystem control objectives, the selection of the controlled and manipulated variables can be made in a decentralized fashion. The method also deals with the sequencing of the set point changes using "Sequencing Trees" to ensure plant safety and bottlenecking are considered. The best route selected should also try to reduce the number of setpoint changes made and select the smoother and faster routes with respect to plant operation.

There is no agreement in the literature on whether distributed is better than global real time optimization or not. Darby and White [3] claim the following advantages for the distributed approach:

- The distributed optimization can be carried out more frequently than the global approach as the method only has to wait for steady-state in each subsystem rather than the plant as a whole.
- The local optimizers can model the subsystem more accurately and completely than a global optimizer can as the individual model dimensionalities will be less restricted than the global model. Also different models can have different optimization levels.
- Local optimizers permit the incorporation of on-line adaptation or on-line parameter estimation more easily.
- Local optimizers are easier to maintain as they are less complex and hence easier to understand than a large global optimizer.
- If problems are occurring in the modeling, the local optimizer causing the problem can be taken off-line while the rest of the optimizers continue to function. A global optimizer would have to be completely turned off to be restructured.

Bailey et al. [6], however, point out two major difficulties with the distributed approach:

- The constraint information being passed between local unit optimizers to prevent conflicts is not as effective as that in a global approach.
- It is possible to get inconsistencies in the update of the parameters when only parts of the process are considered in the local optimizers.

Unfortunately no actual case studies using the distributed approach were available to help determine which approach gives better results.

(iii) Direct Methods (Model free approaches)

The real time optimization is performed directly on the process without explicitly using any model. Direct methods are discussed by Arkun and Stephanopoulos [51] who conclude that the method is too slow and simplistic for on-line optimization. The method is an evolutionary operation, sometimes called an on-line hill-climbing gradient search, where after each trial of set point changes, the objective function is measured at steady-state and its sensitivity is used to readjust the setpoint. Large numbers of set point changes and on-line sensitivity measurements are required to observe the behaviour of the objective function, especially when process noise is present. The method is also slow to reach the optimum as it has to wait for steady-state after each set point change. In early attempts at real time optimization, this method was used and criticized for its slowness.

Implementation

Two papers by Latour [4,52] cover many of the issues relating to real time optimization from a business objective point of view. Tables 1 and 2 list some of these points.

Table 1. *Requirements for on-line optimization [4].*

- Select proper independent variables.
- Formulate model requirements.
- Formulate business objective functions.
- Select optimization algorithms appropriate to the nature of the problem (process and business). Always guarantee a feasible solution. Handle partial equipment failures.
- Specify sensor inputs and manual inputs.
- Difine human interface for objectives, defaults,economic parameters, physical limits, laboratory analyses. The system must inform people what it is doing and why.
- Specify the interface with regulatory controls, consider timing, move limits, output seuencing, process dynamics, stability and interactions.
- Specify computer hardware and software.
- Consider procedures for maintaining model fidelity, process and economic.

All process models are only approximations of real process behaviour. Therefore as operating conditions in the units shift with time, the model coefficients will require updating. Also new constraints may need to be added or old ones deleted at some point. Whether some model maintenance should be carried out manually or automatically is an important decision to be made in the implementation of the optimizer. Ongoing long term maintenance of the on-line optimizer must be carried out after implementation. There have been many cases of initial successes which later floundered due to an inadequate number of trained personnel assigned to the long term maintenance of the optimizer. This often results in the real time optimizer falling into disuse.

The complexity of the process model also determines the sophistication of the maintenance required. Not all variables and constraints can be included in the process model. The art of optimization modeling is to select the variables that have the most

Table 2. *What on-line optimizers will not do [4].*

- Determine business objectives of the plant.
- Specify the process and economic models.
- Select criteria for finding and defining the plant optimum.
- Select the best set of independent variables.
- Specify mechanical or safety limits
- Determine market requirements for product quality.
- Verify product values, fuel and feed costs, and production rate limits.

significant effect on the economics and important constraints, while eliminating those of lesser significance [3].

It is difficult to convey to operators what the optimizer is doing and why. Generally, if the operator does not understand the results, the optimizer will be turned off at the first excuse. Simplicity and modularity in the interface to the operators will enhance understanding [3]. Campbell [53] and Hanmandlu et al. [5] both cover features of on-line optimization software. Figure 8 shows how an on-line optimizer interacts with other parts of the plant. Good training and documentation at both the user and system level are required to get the best results out of the optimizer.

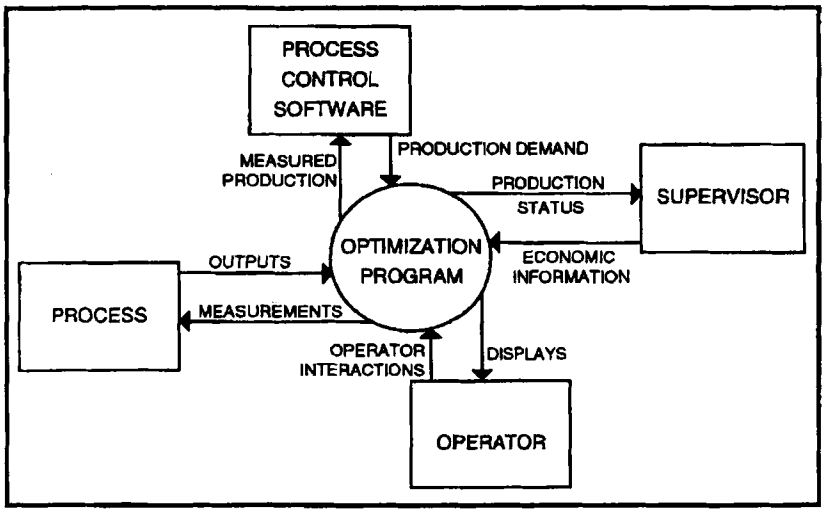

Figure 8. Interaction of optimizing program in the plant system [5,53].

The real time optimizer needs to be able to operate without some available process measurements. If the optimization is disabled every time a single input is off-line, then it will not be in use very much in a large plant. Methods for getting around this problem include estimating the missing input or being able to switch off that portion of the optimizer affected by the loss of the input [3].

Conclusions and Future Areas of Research

A review in the area of real time optimization has been presented. At present, real time optimization is not a well researched or documented field. Most of the development in real time optimization has been carried out by industry, and consequently most information is not easily accessible for study. Some areas of real time optimization, such as the optimization algorithm, data reconciliation and gross error detection do receive significant attention in the literature. Other areas including implementation, on going maintenance of a real time optimizer and model development (specifically for real time optimization) have very few references available. Questions addressing the issues of how close to the constraints should a real time optimizer be allowed to change the process set points, and what effect does the real time optimizer have on the stability of the process control system have not been addressed in the literature to date.

Companies using real time optimization in their processes are tending to move away from developing the optimizer themselves towards using vendor provided software. Extension of process simulation packages to include real time optimization would enable a lower level of expertise to create and maintain a real time optimizer, and also help to provide user friendly graphical interfaces for the software. The use of similar models in the real time optimizer to those used by the process engineer would also have benefits in both the understanding and maintenance of the plant process models. This may involve simplifying the rigorous design models for use in a real time optimizer. With the rapidly decreasing cost of computer power, real time optimization is becoming a more attractive method of reducing processing costs.

References

1. Jang, S., Joseph, B. and Mikai, H. (1987). On-line Optimization of Constrained Multivariable Chemical Processes. *AIChE J.*, 33(1):26-35.
2. Cutler, C.R. and Perry, R.T. (1983). Real Time Optimization with Multivariable Control is Required to Maximize Profits. *Comput. Chem. Eng.*, 7:663-667.
3. Darby, M.L. and White, D.C. (1988). On-Line Optimization of Complex Process Units. *Chem. Eng. Prog.*, 67(10):51-59.
4. Latour, P.R. (1979). Online Computer Optimization 2: Benefits and Implementation. *Hydrocarbon Process.*, 58(7):219-223.
5. Hanmandlu, M., Purkayastha, P. and Pal, J.K. (1985). On the Use of Nonlinear Programming in Real Time Control in Process Industries. *IFAC Control Applications of Nonlinear Programming and Optimization Conference, Capri, Italy,* Pergamon Press, Oxford, U.K., pp71-78
6. Bailey, J.K., Hyrmak, A.N., Treiber, S.S. and Hawkins R.B. (1993). Nonlinear Optimization of a Hydrocracker Fractionation Plant. *Comput. Chem. Eng.* 17:123-138.
7. Fatora, F.C. and Ayala, J.S. (1992). Successful Closed Loop Real-time Optimization. *Hydrocarbon Process.*, 71(6):65-68.
8. Forbes, J.F., Marlin, T.E. and MacGregor, J.F. (1992). Model Accuracy Requirements for Economic Optimizing Model Predictive Controllers - The Linear Programming Case. *Proceedings of the 1992 American Control Conference - Volume 2,* American Automatic Control Council, Green Valley, Arizona, pp1587-1593.
9. Forbes, J.F., Marlin, T.E. and MacGregor, J.F. (1994). Model Adequacy Requirements for Optimizing Plant Operations. *Comput. Chem. Eng.*, 18:497-510.
10. Lojek, R.J. and Whitehead, B.D. (1989). Integrated Advanced Control and Online Optimization in Olefins Plant. *Comput. Chem. Eng.*, 13:1219-1297.
11. Macchietto, S., Stuart, G., Perris, T.A. and Dissinger, G.R. (1989). Monitoring and On-line Optimization of Process Using SPEEDUP. *Comput. Chem. Eng.*, 13:571-576.
12. Shewchuk, C.F. and Morton, W. (1994). The Evolution of an On-line Model-based Optimization System. *Pulp Paper Canada,* 95(6):29-34.
13. Tjoa, I.B., and Beigler, L.T. (1991). Simultaneous Strategies for Data Reconciliation and Gross Error Detection of Nonlinear Systems. *Comput. Chem. Eng.*, 15:679-690.
14. Edgar, T.F. and Himmelblau, D.M. (1988). *Optimization of Chemical Processes,* McGraw-Hill, New York.
15. Boston, J.F. and Britt, H.I. (1978). A Radically Different Formulation and Solution of the Single Stage Flash Problem. *Comput. Chem. Eng.*, 2:109.

16. Biegler, L.T., Grossman, I.E. and Westerburg, A.W. (1985). A Note on Approximation Techniques Used for Process Optimization. *Comput. Chem. Eng.*, 9:201-206.
17. Gallier, P.W. and Kisala, T.P. (1987). Process Optimization by Simulation. *Chem. Eng. Prog.*, 83(8):60-66.
18. Gill, P.E., Murray, W. and Wright, M. (1981). *Practical Optimization.* Academic Press, New York.
19. Fletcher, R. (1987). *Practical Methods of Optimization* (2nd edition). John Wiley & Sons, Chichester, U.K.
20. Lucia, A. and Xu, J. (1990). Chemical Process Optimization Using Newton-like Methods. *Comput. Chem. Eng.*, 14:119-138.
21. Wellons, M.C., Sapre, A.V., Chang, A.I. and Laird, T.L. (1994). On-line Power Plant Optimization Improves Texas Refiner's Bottom Line. *Oil Gas J.*, 92(20):53-58.
22. Murtagh, B.A. and Saunders, M.A. (1982). A Projected Lagrangian Algorithm and its Implementation for Sparse Nonlinear Constraints. *Math Program Study* 16:84-117.
23. Kocis, G.R. and Grossman, I.E. (1989). Computational Experience with DICOPT Solving MINLP Problems in Process Systems Engineering. *Comput. Chem. Eng.*, 13:307-315.
24. Mah, R.S.H. (1990). *Chemical Process Structures and Information Flows.* Butterworths, Boston.
25. Liebman, M.J., Edgar, T.F. and Lasdon, L.S. (1992). Efficient Reconciliation and Estimation for Dynamic Processes Using Nonlinear Programming Techniques. *Comput. Chem. Eng.*, 16:963-986.
26. Meyer, M., Koehret, B. and Enjalbert, M. (1993). Data Reconciliation on Multicomponent Network Process. *Comput. Chem. Eng.*, 17:807-817.
27. Crowe, C.M. (1989). Observability and Redundancy of Process Data for Steady State Reconciliation. *Chem. Eng. Sci.*, 44:2909-2917.
28. Tamhane, A.C. and Mah, R.S.H. (1985). Data Reconciliation and Gross Error Detection in Chemical Process Networks. *Technometrics*, 27:409-422.
29. Terry, P.A. and Himmelblau, D.M. (1993). Data Rectification and Gross Error Detection in a Steady-state Process via Artificial Neural Networks. *Ind. Eng. Chem. Res.*, 32:3020-3028.
30. Lawrence, R.J. (1989). Data Reconciliation: Getting Better Information. *Hydrocarbon Process.*, 68(6):55-60.
31. Crowe, C.M., Garcia Campos, Y.A. and Hyrmak, A. (1983). Reconciliation of Process Flow Rates by Matrix Projection. *AIChE J.*, 29:881-888.
32. Crowe, C.M. (1986). Reconciliation of Process Flow Rates by Matrix Projection Part II: The Nonlinear Case. *AIChE J.*, 32:616-623.
33. Serth, R.W., Weart, M. T. and Heenan, W.A. (1989). On the Choice of Regression Variables in Nonlinear Data Reconciliation. *Chem. Eng. Commun.*, 79:141-152.
34. Stephenson, G.R. and Shewchuk, C.F. (1986). Reconciliation of Process Data with Process Simulation. *AIChE J.*, 32:247-254.
35. Takiyama, H., Naka, Y. and O'Shima, E. (1991). Sensor-based Data Reconciliation Method and Application to the Pilot Plant. J. *Chem. Eng. Japan*, 24:339-346.
36. Darouach, M. and Zasadzinski, M. (1991). Data Reconciliation in Generalized Linear Dynamic Systems. *AIChE J.*, 37:193-201.
37. Rollins, D.K. and Devanathan, S. (1993). Unbiased Estimation in Dynamic Data Reconciliation. *AIChE J.*, 39:1330-1334.

38. Crowe, C.M. (1988). Recursive Identification of Gross Errors in Linear Data Reconciliation. *AIChE J.*, 34:541-550.
39. Crowe, C.M. (1989). Tests of Maximum Power for Detection of Gross Errors in Process Constraints. *AIChE J.*, 35:869-872.
40. Crowe, C.M. (1992). The Maximum-power Test for Gross Errors in the Original Constraints in Data Reconciliation. *Canandian J. Chem. Eng.*, 70:1030-1036.
41. Narasimhan, S. and Mah, R.S.H. (1987). Generalized Likelihood Ratio Method for Gross Error Identification. *AIChE J.*, 33:1514-1521.
42. Narasimhan, S. (1990). Maximum Power Tests for Gross Error Detection Using Likelihood Ratios. *AIChE J.*, 36:1589-1591.
43. Narasimhan, S. and Harikumar, P. (1993). A Method to Incorporate Bounds in Data Reconciliation and Gross Error Detection - I. The Bounded Data Reconciliation Problem. *Comput. Chem. Eng.*, 17:1115-1120.
44. Harikumar, P. and Narasimhan, S. (1993). A Method to Incorporate Bounds in Data Reconciliation and Gross Error Detection - II. Gross Error Detection Strategies. *Comput. Chem. Eng.*, 17:1121-1128.
45. Krishnan, S., Barton, G.W. and Perkins, J.D. (1993). Robust Parameter Estimation in On-line Optimization - Part II. Application to an Industrial Process. *Comput. Chem. Eng.*, 17:663-669.
46. Krishnan, S., Barton, G.W. and Perkins, J.D. (1992). Robust Parameter Estimation in On-line Optimization - Part I. Methodology and Simulated Case Study. *Comput. Chem. Eng.*, 16:545-562.
47. Tsang, E.H. and Meixell, M.D. (1988). Large Scale Ammonia Process Optimization in Real-time. *Proceedings of the 1988 American Control Conference - Volume 3,* American Automatic Control Council, Green Valley, Arizona, pp1971-1974.
48. Hamer, J.W. and Richenburg, C.B. (1988). On-Line Optimizing Control of a Packed-Bed Immobilised-Cell Reactor. *AIChE J.*, 34:626-632.
49. Strand, W.C., Ferritsius, O. and Mokvist, A.V. (1991). Use of Simulation Models in the On-line Control and Optimization of the Refining Process. *Tappi J.*, 74(11):103-108.
50. Arkun, Y., and Stephanopoulos, G. (1981). Studies in the Synthesis of Control Structures for Chemical Processes. Part V: Design of Steady-State Optimizing Control Structures for Integrated Chemical Plants. *AIChE J.*, 27:779-793.
51. Arkun, Y. and Stephanopoulos, G. (1980). Optimizing Control of Industrial Chemical Processes: State of Art Review. *Joint Automatic Control Conference,* pWP5-A.
52. Latour, P.R. (1979). Online Computer Optimization 1: What It Is and Where to Do It. *Hydrocarbon Process.*, 58(6):73-82.
53. Campbell, B.D. (1984). Optimizing Software Systems can be Created Successfully. *Hydrocarbon Process.*, 63(11):109-112.

Received: 12 April 1995; *Accepted after revision:* 1 August 1995.