Nonlinear Dynamic Data Reconciliation: In-depth Case Study

James H. Taylor & Rocio del Pilar Moreno
Dept. of Electrical & Computer Engineering
University of New Brunswick
Fredericton, N. B., Canada, E3B 5A3
jtaylor@unb.ca, pilar.moreno01@gmail.com

***Abstract*— Data reconciliation is a well-known method in on-line process control engineering aimed at estimating the true values of corrupted measurements under constraints. Early nonlinear dynamic data reconciliation (NDDR) studies considered models that were simple and of low order. In such cases the ability to run the NDDR algorithm in real time for relatively slow processes is not a serious problem, despite the heavy computational burden imposed by NDDR. In this study a much more difficult process was treated and the method presented by Laylabadi and Taylor [1], [2] was explored, refined and extended to increase efficiency (reduce computation time).**

In addition, a new hybrid NDDR method is proposed and a demonstrative example performed to show the promise of this approach in reducing the computational burden and handling industrial processes for which a realistic dynamic model does not exist. This contribution makes NDDR more feasible for a wider variety of applications.

I. Introduction

Data from real industrial and chemical processes may be degraded by several effects. **Noise** is the random corruption of sensor measurements due to high-frequency pick-up, low resolution, errors in transmission and conversion (including A/D and D/A conversion), among other effects. Noise is often assumed to be randomly distributed with zero mean. In addition, **gross errors** may result in large discrepancies, e.g., data drop-out or large A/D conversion errors, that may occur at random times. In either case, process dynamic behavior may not be matched, the laws of conservation of energy and mass may not be fulfilled, and trends may be obscured, producing errors in process diagnosis, identification and control. Reliable data is very important in order to achieve a high-quality controlled process. We consider two important aspects of data processing to improve the performance of the control system: Data Reconciliation (DR), and, as an ancillary procedure, Steady-State Detection (SSD).

DR is a technique used to adjust the measurements according to process constraints and conservation laws. Several techniques have been developed, extending the scope to deal with concerns such as dynamic and nonlinear behavior and gross error detection. Thus two categories of DR are prominent, steady-state or static DR (SDR) and dynamic DR (DDR). Note that the main difference between DDR and other filtering techniques is that DDR explicitly uses the process dynamic model as constraints to find estimates of process variables by adjusting the measurements so that the resulting estimates meet these constraints as closely as possible. Therefore the reconciled estimates are less corrupted than the measurements and, more importantly, are consistent with the relationships between process variables defined by the process static and dynamic constraints. In contrast, standard filtering methods (e.g., using a low-pass filter to reduce noise) introduces spurious dynamics.

II. Background

Despite the significant body of literature on DR processing methods showing their multiple advantages, there are few applications of these techniques to realistic large-scale industrial processes. There are significant obstacles to overcome, such as the inavailability of nonlinear mathematical models and the difficulty of running computationally intensive model-based algorithms in real time. This work is focused on the extension, application and assessment of the NDDR technique applied to a highly realistic, large-scale three-phase gravity separator model [4] representing the first stage in the oil production process. In addition to this, a multivariable steady-state detector algorithm has been developed to aid in the NDDR process. These two developments are part of the multi-agent system for Intelligent Control and Asset Management (ICAM) [5], [6].

Data Reconciliation: Kuehn and Davidson (1961 [7]) introduced static data reconciliation and applied it to steady-state chemical engineering processes. Their method involved the solution of an optimization problem that minimizes a weighted least-squares objective function of the error between measured and estimated values of the process variables under static material and energy balance constraints. Mah *et al*. (1976 [8]) treated the general linear data reconciliation problem; they demonstrated that data reconciliation improves data accuracy (compared with standard filtering methods).

In 1992 Liebman *et al*. [3] developed a new approach for NDDR, which used simultaneous optimization and solution techniques associated with a finite data window. Subsequently, Liebman and Edgar [9] demonstrated the advantage of using NLP techniques over conventional steady-state DR methods. They included variable limits and nonlinear algebraic constraints, improving the performance of the reconciliation. Laylabadi and Taylor [1] devised an ANDDR method based on [3] that includes the application to processes with unknown statistical models. They also created and demonstrated a new approach for Gross Error Detection and Correction (GEDC).

While dynamic data reconciliation has been widely studied, there are few applications to real industrial systems, due to the difficulty of solving large NLP problems in real time. McBrayer *et al*. (1998 [10]) reported the successful application of the NDDR algorithm developed by Liebman, to reconcile actual plant data from an Exxon Chemicals process. The model used in this application was developed using several simplifications such as constant pressure (to eliminate fast dynamics) and only one mass balance equation. The present study applies NDDR to a model with complete mass balance and very fast pressure dynamics. It was found, after many attempts, that ANDDR could not run in real time (as it must). This motivated the development of a new hybrid algorithm to make the NDDR process more efficient for application with complex models. The algorithm implemented here covers nonlinear, dynamic and static DR.

Steady-State Detection: Steady-State Detection (SSD) plays an important role in process control and is critical for the application of other functionalities such as Fault Detection Identification and Accommodation (FDIA [15]) and Dynamic Linearized Model Identification (LMId [11]); these require the system to be in steady state in order to produce the best results. In addition, SSD can be used to reduce the computational burden of NDDR, as will be shown.

The majority of methods for SSD are based on calculating either the mean, variance or regression slope over a data window, and comparing them with results over the previous window using statistical tests. Refer to [12] for three typical solutions to the problem of identifying steady-state conditions automatically which were published in *Control for the Process Industries*, 1994.

III. Paper Outline

A multi-agent system for integrated control and asset management of petroleum production facilities [5], [6], [13] and the application (a pilot plant model for crude oil processing [4]) are overviewed in section IV, to establish the context for the research described here. Next, in section V, the theory, methodology and results obtained for the steady-state detection agent are presented. The NDDR approach is described in section VI, as well as a number of modifications and refinements to improve computational efficiency. A new hybrid solution method to tackle the data reconciliation problem is presented in section VII. Finally, in section VIII a summary and conclusions are discussed.

IV. Context: the ICAM System and Application

PAWS, Petroleum Applications of Wireless Systems, was a major research project pursued by several universities in Atlantic Canada for oil and gas applications. The overall project was divided in two major areas: One, led by Cape Breton University, focused on using wireless sensor networks to replace wired data cables in refineries and offshore oil rigs, and the University of New Brunswick (UNB) portion worked on the intelligent management and control of data and processes. For more information about the UNB part of the PAWS project see [14].

A. ICAM System Overview

Extensive supervisory monitoring and control is required to obtain accurate, reliable and efficient control of a modern industrial process. A number of actions may have to be executed, including steady-state detection; linearized model identification; dynamic data reconciliation; fault detection, isolation and accommodation; and supervisory control. ICAM [6], [13] is a multi-agent system, capable of integrating, supervising and managing all these tasks efficiently. Such a system reduces maintenance and production costs, improves utilization of manufacturing equipment, enhances safety and improves product quality. The ICAM infrastructure was interfaced with a pilot plant simulator, described below.

B. The ICAM System Prototype

Figure 1 depicts the original ICAM prototype [6], [13]. Data are obtained in real time either from an external plant or from a simulation model. It is stored in the database and also passed to the SSD Agent, which determines if the plant is either in steady or transient state, and to the ANDDR Agent, which reduces the noise and removes outliers. Processed data are also stored in a real-time database. The Linearized Model Identification (LMId) Agent is invoked if there is no model available or if a significant change in the process operating point occurs. This agent uses generalized binary noise signals as test inputs to perturb the process and collects the corresponding output signals to be used for LMId. Once a new model is obtained, the linearized process model parameters are updated and loaded into the ANDDR [1], [2] and FDIA Agents [11], [15].

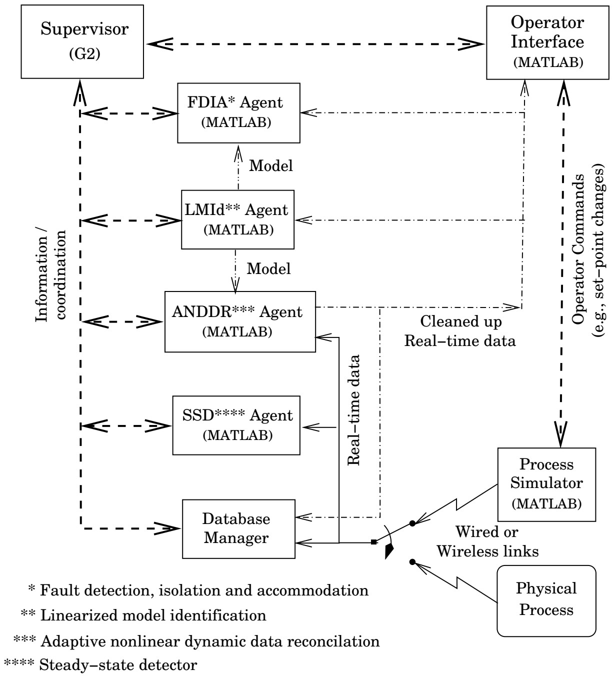

Fig. 1. ICAM system prototype

The supervisor is alerted about every event that occurs as it monitors, observes, and controls the system. An operator interface receives the data and the information from the supervisor relative to the different agents. This allows the operator to take decisions according to the system status and requirements. The external plant for this particular project represents an oil production facility, which separates crude oil from the well into petroleum, water, and gas. ICAM was supposed to interface with a pilot plant (three-phase separator) at the College of North Atlantic (CNA); however, due to logistical difficulties all PAWS research at UNB used a realistic model of this process [4].

C. Application: a Pilot Plant Model

Oil production facilities exhibit very complex and challenging dynamic behavior. The application treated in this paper is a *three-phase separator*, consisting of two horizontal tanks, the first called a “group separator” in which most of the gas is separated from oil and water, followed by a “treater” where residual gas, oil and water are all separated to the extent possible. Each phase’s dynamics are modeled; the hydrodynamics of oil-water separation is modeled based on the American Petroleum Institute design criteria, which involves solving an internal optimization problem, and the oil and gas phases' dynamic behaviors are modeled assuming gas-liquid phase equilibrium at the oil surface. The resulting model has states that are quite slow (liquid levels) and very fast (gas pressures). This oil production facility model was implemented in MATLAB®to produce our simulator.

V. Steady State Detection

Steady-state detection is an important functionality in process performance assessment, optimization, and control, and plays a significant role in ICAM. In particular, the LMId and FDIA agents require the system to be at steady state before they can start working. This paper presents a method for steady state detection based on linear regression over a moving data window. A description of the algorithm is given, followed by results obtained when applying this method to the pilot plant model.

A. SSD Algorithm

Several approaches were studied in order to find an algorithm that offered good performance using noisy data from the pilot plant model. The method finally adopted performs a least square linear regression over a moving data window. The purpose of this is to find the equation of the best-fitting line (in a least squares sense) and to analyze the rate-of-change of the line reflected in its slope. Given that the method was to be applied to the pilot plant which is a complex multi-variable process, every output is analyzed separately, and when all the variables fulfill the condition for steady state, the overall system is declared to be in steady state.

A moving data window approach is used for this algorithm. Although the concept is similar to the moving window used in data reconciliation, VI-B, the size of the window is different. The criteria for choosing this parameter for SSD depends upon the time constants of the variables, unlike in DDR where the size of the window depends upon the amount of noise reduction desired. The pilot plant model has five output variables; every variable has a different time constant, some very long (liquid levels) and some very short (gas pressures). Therefore, every signal is treated independently using a different window size. The advantage of using a data window is that it reduces computation time and the need to store data.

Once enough data is obtained to fill the window, linear regression is performed and the slope is compared with a threshold T. If the slope is smaller than the threshold for several samples (D samples), steady state can be confirmed. Figure 2 illustrates the concept of the method adopted to detect steady state; it shows the volume on the separator when a setpoint change of 10% of its nominal operating value is applied at time $t = 0 \ sec$. The noise standard deviation is 1% of each variable's nominal operating value in all examples herein. When the signal is in the transient state, the slope of the line is large (m1). At the maximum point the slope is small (m2), but this condition changes in a few samples. The closer the signal is to steady state, the smaller the slope (m3) and this condition continues. The threshold T is not a constant; it depends upon the set point change, SP_i, and the standard deviation of the noise, σ_i, both of which are assumed to be known. The formula for this parameter was obtained by doing a series of experiments, running the algorithm for different combinations of SP_i and σ_i and performing a multiple regression to fit the different outcomes of the tests. The empirical equation for the threshold vector is:

$$T = a_0 + a_1\sigma + a_2 SP \qquad (1)$$

where a_i are the coefficients of the threshold model.

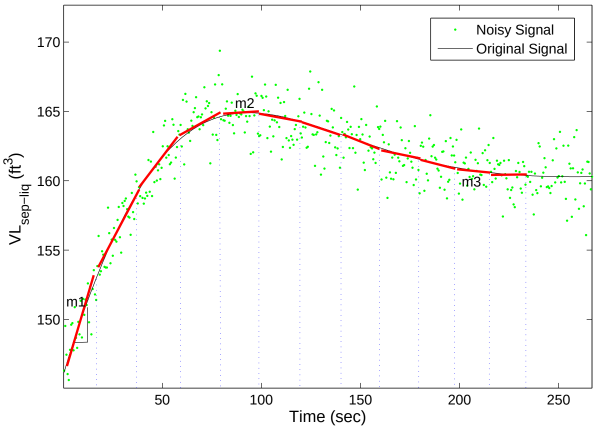

Fig. 2. Steady state detection example

B. SSD Algorithm Results

Measurements were simulated and noise added. The noise was assumed to be Gaussian with zero mean, and the time step used is 0.15 seconds. Figure 3 shows the response of the SSD algorithm for the volume in the separator $V_{sep-liq}$. In the simulation, the plant is working at its nominal operating point for the first $100 sec$. At that time a set-point change of 5% of the nominal operating value is applied. The upper plot displays the linear regression slope and the lower shows the noisy signal and the original signal without noise. In both plots is possible to observe the SSD flag, which informs the supervisor about the state of the plant. If the SSD flag is low the signal is in unsteady state, and if it is at the high the variable has reached steady state. The high and low values of the steady state flag sent to the supervisor are unity and zero; however, in figures those values are modified in order to make the flag's value comparable with the associated variable. This figure portrays the volume of the liquid in the separator ($V_{sep-liq}$) along with the regression slope and SSD flag. Note that this is a slow variable, so it takes a long

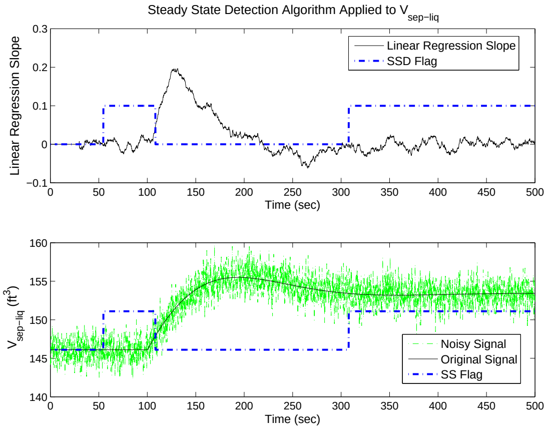

Fig. 3. Steady state detection for $V_{sep-liq}$

time to reach steady state. There are also fast signals such as the pressure in the separator ($P_{sep-vap}$) and in the treator ($P_{treat-vap}$), which take shorter times to reach steady state. When the algorithm is applied to the complete plant there are two kinds of flags: an individual flag for every variable, as illustrated in figure 3, and an overall flag which indicates the state of the entire system.

VI. Nonlinear Dynamic Data Reconciliation

Modern chemical plants, petrochemical processes and refineries work by measuring and controlling several variables such as flow rates, temperatures, pressures, levels and compositions. Sensed values of these variables are subject to corruption by random and systematic errors. Due to these errors, the relationship between the measured inputs and outputs of a system may not match with the process dynamics or conservation laws (process constraints). As was explained in section I, NDDR improves the accuracy of process data by adjusting the measured values so that they satisfy process constraints.

Based on the method used by Laylabadi and Taylor [1], [2], we implemented, refined, and assessed the ANDDR technique by applying it to the PAWS pilot plant model. The general formulation for the NDDR problem introduced by Liebman *et al.* [3] is presented below; the adaptive feature in the ANDDR algorithm was not the focus here. Rather, we address the refinements needed to tackle the problem while implementing NDDR for a much more complex model.

A. NDDR Formulation

The general NDDR formulation can be expressed as follows [3]:

$$\min_{\hat{z}(t)} \Phi[\tilde{z}, \hat{z}(t); \sigma], \tag{2}$$

subject to

$$\Psi(\frac{d\hat{z}(t)}{dt}, \hat{z}) = 0, \tag{3}$$

$$h[\hat{z}(t)] = 0, \tag{4}$$

$$g[\hat{z}(t)] \geq 0, \tag{5}$$

where

z = true (noise free) measurements,

$\tilde{z}$ = corrupted measurements,

$\hat{z}$ = estimated (reconciled) measurements,

Φ = objective function,

σ = measurement noise standard deviations,

Ψ = process dynamic constraints,

h = energy and/or material balance constraints,

g = process variable limits.

The lengths of z, $\tilde{z}$, $\hat{z}$ and σ are equal to the total number of variables (states and inputs), i.e., $z = [x \mid u]^T$.

B. Solution Strategy

We narrowed the definitions of the objective function and dynamic constraints (equations 2,3) to define the algorithm used here. Then there are three important strategies adopted to facilitate the solution of the general NDDR problem: using a process of discretization, a moving horizon data window, and an adaptive noise model.

Objective function, dynamic constraints and discretization: Most applications use weighted least-squares as the objective function in equation (2), and the dynamic constraint in equation (3) is usually that the process differential equation must be satisfied. Then $\hat{z}$ is adjusted until the difference between the measurements $\tilde{z}$ over the window and integrating the system differential equation over that window is minimized in the mean-square sense.

$$\Phi = \sum_{i=0}^{n_i+n_s} \eta_i \sum_{j=c-H}^{c} (\frac{\tilde{z}_{ij} - \hat{z}_{ij}}{\sigma_i})^2, \tag{6}$$

subject to

$$\Psi = \frac{d\hat{z}}{dt} - f(\hat{x}) = 0, \tag{7}$$

where η is a vector of weights, and n_i and n_s are the number of inputs and states. The estimation scheme is maximum likelihood if the weights η_i are all equal. If $\dot{x} = f(z)$ is a physics-based nonlinear model then the material and/or energy balance conditions are satisfied and equality constraints are not required. The inequality constraints may include limits on process variables; for example, the separator input and output flows cannot be negative.

Finally, the dynamic constraint, equation (7), needs to be discretized in order to solve the NLP problem defined by the equations above. To achieve this, the differential equation set $\dot{x} = f(z)$ is solved numerically over the window using a fixed-step integrator with the step equal to the sampling time.

Moving Horizon Window: Liebman *et al.* [3] introduced a moving time window approach in order to decrease the size of the optimization problem. If t_c is defined as the present time, the *history horizon* is established from $t_c - (H-1)\Delta t$ to t_c, where Δt is the step size; it is important to choose an appropriate horizon length *H*. If H is too small, the information available may not be enough to perform a good reconciliation, but if it is too large, the NLP problem can become excessively large. The steps for NDDR can be summarized as:

1) Acquire process measurements at time $t = t_c$
2) Minimize Φ, equation 6, under the constraints in equations 7, 4 and 5 over the window $(t_c - (H-1)\Delta t \leq t \leq t_c)$
3) Save $\hat{y}$ at time t_c as the reconciled signal for online control purposes
4) Repeat at the next step, t_{c+1}

The advantages of the moving window approach are that it reduces the size of the NLP problem and the only tuning parameter is the size of the history horizon *H*, if the weights η_i in equation 6 are equal.

Adaptive Noise Model: Taylor and Laylabodi [2] addressed systems with an unknown noise model. They also used a moving window approach; however the window should be substantially longer than that used for ANDDR. They estimated the noise model $\hat{\sigma}$ as the sample standard deviation over the window; in order for this estimate to be statistically significant this window should contain at least 50 to 100 points, and the true (noise-free) signals z must vary slowly over that window. If a sufficiently large number of points is used then the random variable $\hat{\sigma}$ is approximately gaussian with variance inversely proportional to the number of points. For a more thorough discussion of the statistics of $\hat{\sigma}$ refer to Taylor [16].

C. ANDDR Results

The ANDDR algorithm was applied to the pilot plant model to assess the performance of this method in a large scale, realistic model with fast dynamics (gas pressures), and as a first step to develop an agent capable of working within the ICAM system. For this test, five inputs and five outputs were estimated. True values were obtained by simulating the nonlinear model with a time step of $\Delta t = 0.15\ sec.$, and measurements were created by adding Gaussian noise to the true values; they were assumed to be gross error free. There were no setpoint changes, all variables are at their nominal operating point value, and the window size *H* was set to 10.

The approach here mimics that developed by Laylabadi and Taylor [1], [2]; only the simulator was different and much more challenging. Optimization was executed using the unconstrained nonlinear method `fminsearch`; this is a direct search method that does not use numerical or analytic gradients. Initial guesses for $\hat{z}_i$ in the optimization problem were set to the previous estimates $\hat{z}_{i-1}$ and all the weights $\eta_i = 1$.

Applying this approach to the pilot plant model consumed a huge amount of computation time. The optimization routine often could not find a minimum; it kept iterating until the maximum number of iterations or maximum number of evaluations were reached. ANDDR execution in this case consumed 1,049 times real time (simulated), which is entirely unacceptable. This is due to the fact that the pilot plant model is extremely complex, with an internal optimization problem needing to be solved to balance oil, water and gas separation at each time step. The small sampling time is also a factor; it is based on the rapid gas pressure dynamics. Finally, `fminsearch` converges very slowly for searches of three or more dimensions [17]. The pilot plant model includes a total of ten parameters to be estimated, so `fminsearch` is unsuitable for the problem addressed here.

We tried several strategies to reduce run time: using a different minimization routine, varying the window length, adjusting the weights in the objective function and modifying the initial guesses; here we can only summarize the results.

Data Reconciliation Results Using fminunc: Although the results of the original ANDDR algorithm [2] exhibit good noise and RMSE reduction, the computation time is an important issue. The first attempt at reducing the computation time was to use a different minimization routine, `fminunc` [17], which is an efficient large-scale algorithm.

Using `fminunc` was the only change from the previous case. The results are shown in figure 4. Table I shows the quantitative results for this test, and a comparison between these results and those obtained with `fminsearch`. An important reduction in the computation time was observed: using `fminsearch` the computation time was $t_{comp} = 21,094\ sec.$ (1,049 times real time) and using `fminunc` the computation time was reduced to $t_{comp} = 1,368\ sec.$ (68 times the running time). This is a big improvement, but this run time is still too large,

The noise reduction achieved using `fminunc` is smaller that the reduction obtained using `fminsearch`, but still it is significant. The amounts of RMSE reduction are very similar between the two approaches. The reason for the lower noise reduction would require in-depth analysis of these two minimization methods, which was beyond the scope of this research. Perhaps the default tolerances in these algorithms are making `fminunc` less demanding.

Several more options were tested to speed up the ANDDR algorithm. These included modifying the size of the moving window, using scaling in the cost function, and varying the optimization tolerances. Data used for the different tests are the same as described previously, but the algorithms were executed for a shorter time ($t_f = 10\ sec.$ in every case), to shorten run times. The results and conclusions for these tests are presented in the following sections.

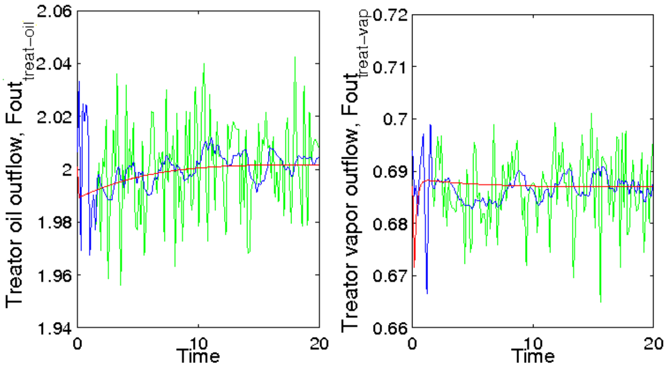

Fig. 4. Sample ANDDR results using `fminunc`

Variable		% Noise reduction		% RMSE reduction	
		`search`	`unc`	`search`	`unc`
Inputs	$Fout_{sep-liq}$	80.68	65.62	64.88	60.21
	$Fout_{sep-vap}$	74.69	57.49	56.88	55.60
	$Fout_{treat-wat}$	85.40	59.01	64.17	61.59
	$Fout_{treat-oil}$	81.25	74.63	53.57	54.10
	$Fout_{treat-vap}$	73.94	63.60	51.78	50.14
Outputs	$V_{sep-liq}$	76.15	72.35	60.47	62.75
	$P_{sep-vap}$	69.24	69.49	60.93	61.72
	$V_{treat-wat}$	69.03	80.24	65.97	70.33
	$V_{treat-oil}$	69.87	70.52	57.21	60.39
	$P_{treat-vap}$	64.52	54.59	49.59	45.25
Average		74.48	66.72	58.54	58.21

TABLE I

ANDDR PERFORMANCE USING `fminsearch` AND `fminunc`

Modifying the Window Size: One of the advantages of using a moving window as part of the solution strategy for ANDDR is that the length of the window acts as a convenient tuning parameter. Increasing *H* provides more information for the optimization algorithm and yields more noise reduction, although the optimization will take longer. Therefore it is necessary to find a balance between time consumed and noise reduction achieved.

Table II shows the percentage of noise reduction when the ANDDR algorithm is executed using different values of *H*. Note that the time consumed in executing the algorithm increases with the value of *H*, as does the noise and RMSE reduction. The level of noise reduction and RMSE reduction increase substantially with *H*, but only up to a certain value (in this case approximately *H*= 16); for larger values of *H*, the improvement in the estimates is less significant.

The window size chosen is $H = 8$, to obtain percentages of noise reduction and RMSE reduction above 50%. Using this value the computation time is reduced from 68 times real time to 48 times. An improvement in time is achieved while obtaining a decent reduction in noise; however, the computer time is still too large.

Using Unequal Weights: A second attempt was made with $H = 8$, by changing the scaling (weights η_i). Initially all the values of the weights were unity, to preserve the maximum-likelihood nature of the estimation scheme. However, it was conjectured that the optimization could be affected by the different ranges of the variables involved in the pilot plant model. To normalize the optimization function, the value of η_i in equation (6) was set to the inverse of the

H	Comp. (sec.)	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$
20	1011	89.8	91.3	95.4	95.0	93.5
16	839	90.3	94.0	94.0	95.8	93.1
12	673.9	75.2	80.1	67.8	83.1	67.3
10	618.9	67.2	77.2	59.0	80.0	62.7
8	481.4	69.8	74.8	61.7	74.4	55.0
6	375.4	60.6	56.8	53.9	71.4	47.9
4	267.1	48.3	46.3	42.1	62.1	37.9

TABLE II

PERCENTAGE OF NOISE REDUCTION *vs.* *H*

nominal operating point value of every variable.

Variable		NO Scaling		Scaling	
		% σ Reduct.	% RMSE Reduct.	% σ Reduct.	% RMSE Reduct.
Inputs	$Fout_{sep-liq}$	69.84	51.96	81.08	74.08
	$Fout_{sep-vap}$	74.82	70.69	67.08	63.61
	$Fout_{treat-wat}$	61.68	60.30	88.03	75.71
	$Fout_{treat-oil}$	74.43	74.38	79.72	77.66
	$Fout_{treat-vap}$	54.95	52.00	58.97	59.88
Outputs	$V_{sep-liq}$	66.49	64.45	73.68	72.42
	$P_{sep-vap}$	63.22	61.68	63.42	56.35
	$V_{treat-wat}$	74.06	67.77	51.91	44.50
	$V_{treat-oil}$	79.98	70.91	81.72	75.40
	$P_{treat-vap}$	62.12	48.34	66.73	31.21
Average		68.16	62.25	71.23	63.08

TABLE III

ANDDR RESULTS USING SCALING

Table III shows the outcome of the scaling test. The results show that using scaling provides, in general, more effective reconciliation. The computation time was $t_{comp} = 425.4\ sec.$, which is lower than in the previous test performed using $H = 8$ and without scaling ($t_{comp} = 481.36\ sec.$). In general the majority of the variables exhibit a greater reduction in noise and RMSE as well.

Change in Optimization Tolerances: The next attempt in trying to reduce computation time for ANDDR was to modify some parameters that govern the behavior of `fminunc`. These changes included modifying the termination tolerances on the objective function value (default for $Tol_{\Phi} = 1e-3$) and on the estimates (default for $Tol_{\hat{y}} = 1e-3$). Several tests were performed in order to establish the effect of these parameters on the performance of the algorithm. Table IV shows the results for some of the tests executed. Performance in terms of percent reduction appears to be rather flat, but computation time is substantially reduced when using $1e-2$ for the tolerances.

Optimization Parameters	Comp. Time (sec.)	% Reduct.	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$
$Tol_{\hat{y}} = 1e-4$ $Tol_{\Phi} = 1e-4$	819	Noise	69.1	73.4	62.6	75.6	57.5
		RMSE	53.3	70.9	61.9	75.3	53.3
$Tol_{\hat{y}} = 1e-3$ $Tol_{\Phi} = 1e-3$	481	Noise	69.84	74.82	61.68	74.43	54.95
		RMSE	51.96	70.71	60.35	74.38	52.01
$Tol_{\hat{y}} = 1e-2$ $Tol_{\Phi} = 1e-2$	360	Noise	71.18	72.37	59.73	75.77	57.21
		RMSE	50.58	68.26	58.56	74.23	55.32

TABLE IV

ANDDR RESULTS USING DIFFERENT OPTIMIZATION TOLERANCES

D. ANDDR Results for a Set Point Change

The best compromise in performance was obtained for $H = 8$, using the previous estimates as initial guesses, applying scaling in the optimization routine, and using the default values for the optimization algorithm parameters. These conditions were selected in specifying the definitive ANDDR algorithm. The following scenario is presented in order to show its effectiveness.

This scenario runs for a real time of $t_f = 100\ sec.$ The plant starts working at the nominal operating point, and a positive setpoint change of 2% is applied at the time $t = 50\ sec.$ Table V and figure 5 present a sampling of the results obtained. This study required a computation time of $t_{comp} = 4828\ sec.$ or 48 times real time to complete reconciliation.

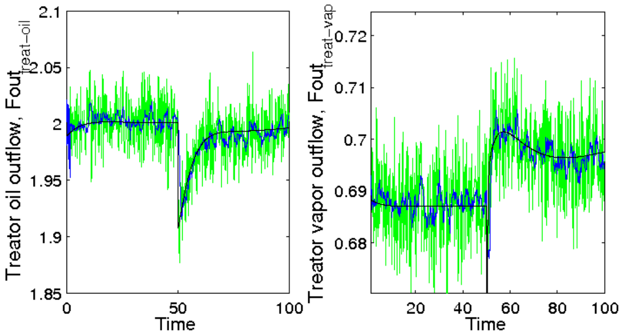

Fig. 5. ANDDR results for a positive setpoint change

Percent reduct.	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$
Noise	73.52	75.57	73.13	70.49	68.52
RMSE	73.30	75.06	73.85	64.89	70.10

TABLE V

ANDDR RESULTS FOR A POSITIVE SETPOINT CHANGE

VII. A HYBRID DDR APPROACH

The results presented in section VI show that the estimates obtained with ANDDR are significantly smoother than the corresponding measurements. However, the time consumed by the algorithm is too large. Another drawback to this approach is the need for a nonlinear process dynamic model, which is often not available or not practical to develop. In order to reduce the computation time and to address the dynamic process model availability issue, a new Hybrid DDR approach (HDDR) was developed and tested. The main idea is to use two different methods to perform the data reconciliation according to the condition of the system variables: transient or steady state.

It is possible to determine the state of the system, using the SSD algorithm explained in section V. During transient periods, data reconciliation is executed using the linearized model of the plant provided by the LMId Agent [11] operating on perturbation variables, *i.e.*, $\delta z = z - z_0$ where z_0 is the steady state operating point. This approach is valid for moderate changes in the set-points and maintains the goal of DDR, namely energy and material balance is retained in z_0. In steady-state intervals, data reconciliation is performed using an equilibrium finding approach instead of solving the system differential equations numerically over the window. It is faster to obtain the equilibrium point than to run the model for eight or ten samples every iteration of the optimization loop. However, using a dynamic model for equilibrium finding is still substantially less efficient than using a static process model directly. Thus the real promise of HDDR will be realized in applications for which a static model exists, which is not true for the pilot plant. The following is a feasibility study, to demonstrate that HDDR works effectively.

The nonlinear pilot plant model is defined in terms of the differential equation as $\dot{x} = f(x, u)$. For an input value u_0 and its corresponding equilibrium x_0, the output value is $y_0 = h(x_0, u_0)$. The perturbation variables can be defined as $\delta x = x - x_0$, $\delta u = u - u_0$, and $\delta y = y - y_0$. If the perturbations are small and if continuous partial derivatives exist at (x_0, u_0), the behavior of the original nonlinear system near x_0 is similar to that of:

$$\delta\dot{x} = A\delta x + B\delta u \tag{8}$$

$$\delta y = C\delta x + D\delta u \tag{9}$$

where $A = [\frac{\delta f}{\delta x}]_{x_0,u_0}$, $B = [\frac{\delta f}{\delta u}]_{x_0,u_0}$, $C = [\frac{\delta h}{\delta x}]_{x_0,u_0}$, and $D = [\frac{\delta h}{\delta u}]_{x_0,u_0}$ [18]. *A*, *B*, *C* and *D* are obtained by the LMId Agent [11], which is convenient.

In transient state, for small changes in the setpoint, the linearized model can be used as the constraints in equation (3). However, in steady state it is not accurate to use a linear approximation, given that the equilibrium point for the linearized model is different from that for the nonlinear model. Furthermore, the concept of data reconciliation is to adjust the measurements according to conservation law constraints. This requirement would not be satisfied by using the linearized model, because although the transient behavior of the original nonlinear system is similar to that of the linear model, the linear model does not provide the physical relationship between inputs and outputs.

A. HDDR Results

The scenario used to test HDDR starts at the nominal operating point and at the time $t = 50$ $sec.$ a positive set point change of 2% is applied to u; the final time is $t_f = 300$ $sec.$ Figure 6 shows the results for the test. The computation time is $t_{comp} = 7494$ $sec.$ which corresponds to 24 times the final time. This is much shorter than the time consumed in the examples presented in section VI (approximately 49 times real time).

Note that the estimates present less noise than the measurements, in both the transient and the steady state. However, outliers are introduced in some of the variables at the moment of the transition from transient to steady state ($t = 250$ $sec.$). The fastest variables ($P_{sep-vap}$ and $P_{treat-vap}$) are the most affected by the transition. It takes a window of *H* samples for the system to adapt again using the nonlinear dynamic model. Table VI shows the quantitative results for this test, confirming the results mentioned above. The reduced run time is due to the long transient period, where the linearized model is used for DDR. Using the steady-state approach with the current pilot plant model does not substantially improve the time consumed for the data reconciliation algorithm: If a simpler nonlinear static model based on material and energy balance were available, the equilibrium algorithm would be much more efficient and it could be used in HDDR to obtain a faster and more reliable data reconciliation routine. The derivation of the new nonlinear static model for the pilot plant is not part of the scope of this project; it is suggested as future work.

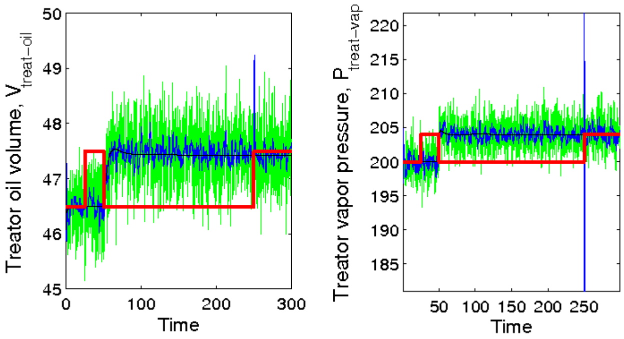

Fig. 6. Hybrid DDR results

Percent reduct.	Comp. Time (sec.)	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$
Noise	7494	70.2	67.1	68.9	63.0	50.8
RMSE	7494	69.9	67.1	68.9	63.1	50.9

TABLE VI
HYBRID DDR APPROACH RESULTS

B. Solving the Steady-state/Transient Transition Problem

A routine was designed to detect changes in the steady state flag and analyze the values of the estimates during the next window. The objective of this routine is to solve the problem of transitioning from transient state to steady state, eliminating outliers produced by switching between the models, as shown in figure 6. When the steady state flag changes from unity to zero or *vice versa*, the algorithm starts to compare the current estimate ($\hat{z}_c$) with the previous one ($\hat{z}_{c-1}$). If the difference between them is larger than five times the standard deviation of the previous data window, then the $\hat{z}_c$ is replaced by $\hat{z}_{c-1}$. Table VII shows the noise and RMSE reduction for this refined HDDR approach, and figure 7 shows the results for $P_{treat-vap}$ using HDDR with this routine. Note that the transition is conducted without the negative effects in the estimates that are apparent in figure 6, and noise reduction is comparable for the separator outputs and superior for the treater outputs.

Percent reduct.	Comp. Time (sec.)	$Fout_{sep-liq}$	$Fout_{sep-vap}$	$Fout_{treat-wat}$	$Fout_{treat-oil}$	$Fout_{treat-vap}$
Noise	7490	69.24	68.36	70.34	68.65	69.36
RMSE	7490	68.93	67.38	67.68	61.31	52.42

TABLE VII
HDDR RESULTS AFTER SOLVING TRANSITION PROBLEM

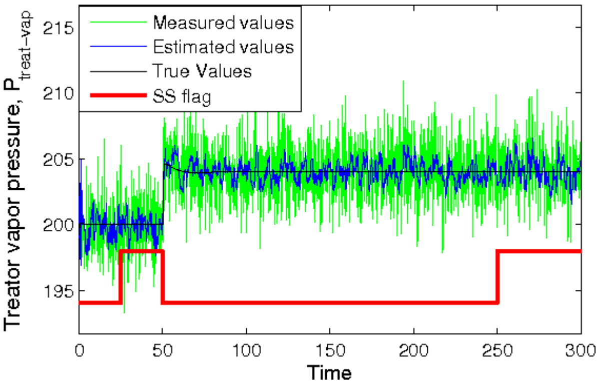

Fig. 7. Hybrid DDR results solving transition problem for $P_{treat-vap}$

VIII. Summary and Conclusions

A. Summary

The ANDDR algorithm was tested extensively on the simulated pilot plant model of Sayda and Taylor [4]. First we tried the ANDDR algorithm exactly as implemented by Laylabadi and Taylor [1], [2] and we found that it was extremely slow (run time = 1049 times real time) and thus totally unsuitable for this application. We performed an extensive case study in which we varied the optimization algorithm, changed the data window size, used scaling in the performance index, and varied the optimization routine's stopping tolerances. The best we could achieve was 18 times real time. We did not try converting the pilot plant model from an M-file to a MEX-file, which might have substantially improved the run time, because we decided to pursue another idea: HDDR.

We proposed and tested an alternative approach to reduce the ANDDR run time, called HDDR. This involves using static DR when the process is in steady state and dynamic DR using a linearized model during transient conditions. We did reduce the computational burden, even though we were handicapped by not having a static model of the process. Instead we had to perform equilibrium finding, i.e., using a nonlinear equation solver to find the solution to $\dot{x} = 0$ which also imposes a large computational burden. We believe that a true static model would speed up the HDDR algorithm substantially. The hybrid method has the added benefit of eliminating the need for a nonlinear dynamic process model, which in most practical applications is not available.

Finally, we believe that the discovery or development of a more suitable optimization algorithm is key to making ANDDR and HDDR practical for large scale, realistic processes. Also, converting the process dynamic model to a `MEX` file or otherwise speeding up the numerical integration process in ANDDR or HDDR will be quite beneficial.

B. Contributions

The requirements demanded by control and automation techniques are continuously increasing due to the need for faster and more reliable results. The best control system performance can only be achieved by using accurate measurements. As a result, data reconciliation and gross error and steady-state detection have become crucial tools for data quality improvement in integrated control and asset management systems.

These methods have been widely studied, but there are few applications to real industrial models or processes, especially not to realistic nonlinear and dynamic processes. Implementing the ANDDR technique [1], [2] on real industrial processes requires facing new challenges that are not present in research using ideal models; this necessitates finding solutions and tools to overcome the difficulties inherent NDDR. The new hybrid method is an important and promising contribution to performing DR efficiently in systems with complex models. Modifying the algorithms in order to make them compatible with a multi-agent supervisory system [6] is another contribution that will facilitate future applications to industrial processes.

IX. Acknowledgments

This research was supported by the Atlantic Canada Opportunities Agency (ACOA) under the Atlantic Innovation Fund (AIF) program. The authors gratefully acknowledge that support, as well as the collaboration of the personnel at the Cape Breton University (CBU) and the College of the North Atlantic (CNA).

References

[1] M. Laylabadi and J. H. Taylor, "ANDDR with novel gross error detection and smart tracking system," *Proc. 12th IFAC Symp. on Information Control Problems in Manufacturing*, May 2006.

[2] J. H. Taylor and M. Laylabadi, "A novel adaptive nonlinear dynamic data reconciliation and gross error detection method," *IEEE Conf. on Control Applications*, October 2006.

[3] M. J. Liebman, T. F. Edgar, and L. S. Lasdon, "Efficient data reconciliation and estimation for dynamic processes using nonlinear programming techniques," *Computers Chem. Eng'ng.*, vol. 16, no. 10/11, pp. 963–986, 1992.

[4] A. Sayda and J. H. Taylor, "Modeling and control of three-phase gravity separators in oil production facilities," *Proc. American Control Conf.*, 2007.

[5] A. Sayda and J. H. Taylor "An implementation plan for integrated control and asset management of petroleum production facilities," *Proc. IEEE Conf. on Control Applications*, 2006.

[6] J. H. Taylor and A. Sayda, "Prototype design of a multi-agent system for integrated control and asset management of petroleum production facilities," *Proc. American Control Conf.*, June 2008.

[7] D. R. Kuehn and H. Davidson, "Computer control II : Mathematics of control," *Chem. Eng'ng. Prog.*, vol. 57, no. 44, 1961.

[8] R. S. H. Mah, G. M. Stanley, and D. M. Downing, "Reconciliation and rectification of process flow and inventory data," *Ind. and Eng'ng. Proc. Design and Dev.*, vol. 15, no. 1, pp. 175–183, 1976.

[9] M. J. Liebman and T. F. Edgar, "Data reconciliation for nonlinear process," *Proc. AIChE Annual Meeting*, 1998.

[10] K. F. McBrayer, T. A. Soderstrom, T. F. Edgar, and R. E. Young, "The application of nonlinear dynamic data reconciliation to plant data," *Computers and Chem. Eng'ng.*, vol. 22, no. 12, pp. 1907–1911, 1998.

[11] M. Omana, "Fault detection, isolation and accomodation using the generalized parity vector technique," Ph.D. dissertation, University of New Brunswick, 2009.

[12] J. Loar, S. L. Alekman, and G. Jubien & G. Bihary, *Control for the Process Industries*. Putman Publications, Chicago, IL, 1994, ch. Three articles by these authors.

[13] A. Sayda and J. H. Taylor, "A multi-agent system for integrated control and asset management of petroleum production facilites - part 1: Prototype design and development," *Proc. IEEE International Symp. on Intelligent Control*, September 2008.

[14] J. H. Taylor, "Intelligent control and asset management for wireless networked control systems," http://www.ece.unb.ca/jtaylor/PAWS.html 2010.

[15] J. H. Taylor and M. Omana, "Fault detection, isolation and accomodation using the generalized parity vector technique," *Proc. IFAC World Congress*, July 2008.

[16] J. H. Taylor, "Statistical performance analysis of nonlinear stochastic systems by the monte carlo method," *Trans. on Mathematics and Computers in Simulation*, vol. 23, April 1981.

[17] MathWorks, "Optimization toolbox for use with MATLAB®," 1990-2004.

[18] G. F. Franklin and J. D. Powell, *Digital Control of Dynamic Systems*. Addison-Wesley Pub. Co., 1980.