

# Nonlinear Dynamic Data Reconciliation and Bias Estimation of Process Measurements in an Adiabatic Stirred-Tank Reactor

Karl Ezra Pilario and Jose Co Munoz

*Department of Chemical Engineering, University of the Philippines - Diliman*

**Abstract**— When process data is taken from the sensors of a plant, errors of varying degrees are inherent. Measured variables will most likely violate dynamic process models. Because of this, large volumes of data may be unreliable for process control, monitoring, and optimization. This paper describes a new method for simultaneous Nonlinear Dynamic Data Reconciliation and Gross Error Detection (NDDR-GED) which conditions raw sensor measurements and estimates bias in faulty sensors. The problem is formulated as a dynamic NLP, solved using a hybrid Nelder-Mead Simplex Particle Swarm Optimization (NM-PSO) algorithm and a moving horizon approach. The use of NM-PSO warrants the transfer of solutions, embedded in each elite particle, from one horizon NLP to the next, thereby promoting smoother profiles and faster convergence. This new feature is seen to be a learning mechanism of the method across time. Discretization of ODEs was done using orthogonal collocation on finite elements. Using simulated data from the nonlinear process model of an adiabatic CSTR, the resulting profiles were both smooth (with a percent standard deviation reduction in measurement error of 80-90%) and accurate to the process model within 10<sup>-7</sup>. Also, large biases were corrected accordingly, if the faulty sensor was known a priori.

**Keywords**— *Data Reconciliation, Bias Estimation, Adiabatic, Stirred Tank Reactor, Error Detection*

## 1. INTRODUCTION

Modern plants around the world consist of two main interacting systems: the physical plant and its information system [1]. The physical plant consists of the machines, equipment, hardware, utilities, control elements, process materials, among others. The information system, sometimes in the form of a Distributed Control System (DCS), acts as the brain of the plant where measurements from the physical plant are processed and integrated to achieve the best productivity. One of the roadblocks to achieving the best productivity is the way these process data are obtained and used.

When process data are acquired from the sensors of the physical plant, errors of varying degrees are inherent. Measured variables such as flow rates, compositions, temperatures, and pressures will most likely violate steady-state and/or dynamic process models. Data Reconciliation (DR) and Gross Error Detection (GED) are the 2 available technologies for adjusting the raw process data so that they satisfy known process models while preserving as much information from the plant as possible [2].

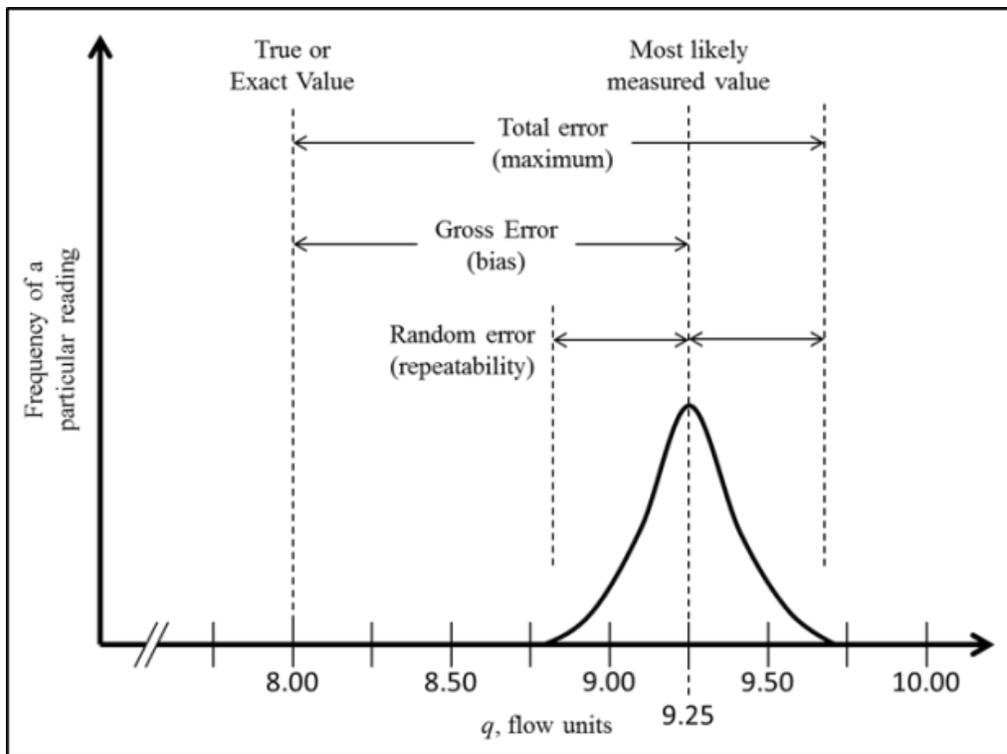
Historically, Kuehn and Davidson (1961) first posed the problem of linear steady-state data reconciliation [3]. From their work, hundreds of publications have appeared in the last half-century about DR, GED, optimal sensor placement, and fault detection and diagnosis.

According to Narasimhan and Jordache (2000) [4], a measured variable is said to be contaminated with 2 types of errors: random errors and gross errors. Random errors, or measurement noise, are small fluctuations of the measured variable from the true value; they can be electrically induced or process-induced [5]. Process-induced noise can arise from disturbances during incomplete mixing, turbulence,

vibrations, and other random events. Electrically induced noise is brought about by the instrument hardware. Gross errors, on the other hand, are large deviations arising from poor sensor calibration, improper sensor installation, instrument malfunction, or fouling. A type of sensor fault which is consistent in magnitude over the observation time is called a bias, denoted by  $\mu$ .

To illustrate how random and gross errors manifest in a measured value, Fig. 1 plots the frequency of measurement values for a particular flow rate in steady-state,  $q$ . In the figure, the true value of  $q$  is  $x^e=8.00$  units. Because sensor bias is also present, it shifts the most likely measured value to 9.25 units. In addition, a Gaussian probability distribution of possible measured values is found around 9.25, indicating noise with a known standard deviation of  $\sigma = 0.15$ . Noise is evident from non-repeatable measurements of a steady-state variable. If, say, the measured value at one point is  $x^m=9.18$ , it actually harbors a random error of  $\varepsilon=-0.07$  and a bias of  $\mu=+1.25$ , on top of the true value  $x^e=8.00$ .

**Figure 1.** Random and gross errors in a measured variable (reproduced from [5])



DR aims to reduce random errors, producing new reconciled values, called estimates, which are approximations of the exact value,  $x^e$ . Note that the value of  $x^e$  can never be truly known, unless an infinite set of measurement data is available and there is no mismatch between the actual process and its model. On the other hand, the GED method for bias is called bias estimation. Bias estimation aims to determine the sign and magnitude of the persistent measurement deviation. In DR-GED, models are seen to be trusted to validate measured data. This is in contrast to a lab experiment, wherein measured data are trusted to validate a model.

Nowadays, steady-state DR-GED is a mature technology. As listed by Bai and Thibault (2010) [1], it has been widely applied in processing industries such as chemical, biochemical, petrochemical, pulp-and-paper, and mineral processing. However, applications in dynamic process models must still undergo a lot of research. This paper describes a new method for simultaneous nonlinear dynamic data

reconciliation and bias estimation (NDDR-GED). The next section presents a literature review that will give sense to the choice of methods in this work. Section 3 outlines the problem statement and objectives; Section 4 presents the methodology; and, lastly, Section 5 discusses an illustrative case of an adiabatic continuously stirred-tank reactor.

## 2. LITERATURE REVIEW

The Data Reconciliation (DR) procedure is based on the fact that: in adjusting sensor measurements to satisfy the process models, the loss of information must be a minimum.

From information theory, one criterion that quantifies the loss of information is the Akaike Information Criterion (AIC). The DR-GED problem becomes an optimization problem by minimizing the AIC, which is the objective function. The branch-and-bound technique was used by Yamamura et al. (1988) to solve the resulting MINLP (mixed-integer nonlinear program) from AIC minimization for steady-state linear DR-GED [6].

Before the use of AIC, the traditional method for nonlinear dynamic measurement noise reduction uses the Extended Kalman Filter (EKF) [7]. Digital data filters simply aim to produce minimum variance estimates from state-space models. Stanley and Mah (1977) [8] pioneered the use of the Kalman Filter for linear dynamic DR-GED. Later on, optimization-based nonlinear DR (AIC minimization), when extended to dynamic process models, has proven to be a superior estimator than EKF [9] [10].

The work of Liebman et al. (1992) [11] is one of the most cited works in the use of Nonlinear Dynamic Data Reconciliation (NDDR) for on-line estimation and noise reduction. They have employed a moving horizon approach that captures a time-window of measurements, wherein DR is performed only once per window. Vacchani et al. (2005) [12] also devised a recursive NDDR, which adopted the recursive nature of the EKF approach. They argued that EKF is still competitive to the NDDR moving horizon approach, with the right modifications.

The most recent developments for NDDR-GED are the use of novel particle filtering by Zhang and Chen (2014) [13], the correntropy robust estimator for NDDR-GED by Zhang and Chen (2015) [14], an adaptive NDDR by Taylor and Laylabadi, (2006) [15], support vector regression by Miao et al. (2011) [16], and the robust auto-associative neural network or AANN-NDDR by Bai et al. (2007) [17]. In the last two references, DR-GED is achieved by data-driven methods instead of model-based methods.

### 2.1 Moving Horizon Estimation

This paper improves upon the approach of Liebman et al. [11] to NDDR, which they call the moving horizon estimation (MHE). In the next paragraphs, the details of MHE are discussed.

On the basis of minimum AIC, the NDDR problem is formulated as a nonlinear program (NLP).

$$\min_{\mathbf{x}^r(t)} \sum_{j=1}^H \sum_{k=1}^N \left( \frac{x_k^m(t_j) - x_k^r(t_j)}{\sigma_k} - \frac{\mu_k}{\sigma_k} \right)^2 \quad (1)$$

$$\text{such that } \mathbf{f} \left[ \frac{d\mathbf{x}^r(t)}{dt}, \mathbf{x}^r(t), \mathbf{u}(t) \right] = 0$$

$$\mathbf{h}[\mathbf{x}^r(t), \mathbf{u}(t)] = 0$$

$$\mathbf{g}[\mathbf{x}^r(t), \mathbf{u}(t)] \leq 0$$

In Eq. (1),  $\mathbf{f}$  is a set of nonlinear ODEs,  $\mathbf{x}^m$  is the set of measured values,  $\mathbf{x}^r$  is the set of reconciled estimates (decision variables),  $\boldsymbol{\mu}$  is the set of biases,  $\boldsymbol{\sigma}$  is the set of standard deviations,  $\mathbf{u}$  is the set of unmeasured variables,  $\mathbf{N}$  is the number of sensors, and  $\mathbf{H}$  is the number of discrete time steps in a horizon. Collectively, sets  $\mathbf{f}$  and  $\mathbf{h}$  constitute a system of differential-algebraic equations (DAE).

Note that aside from reconciled estimates,  $\mathbf{x}^r$ , the solution of Eq. (1) can also give estimates for the biases,  $\boldsymbol{\mu}$ , and unmeasured variables,  $\mathbf{u}$ . Unmeasured variables may fall in either of these 2 cases: they are process variables without sensors due to economic, technical, or practical reasons; or, they are unknown model parameters, e.g. reaction rate constants, transfer coefficients, mixture interaction parameters, and distribution coefficients. Thus, NDDR also offers a way to estimate model parameters on-line.

In MHE,  $H$  is defined as the *horizon length* or *window width*. The search space in Eq. (1) involves only  $N \times H$  dimensions (number of decision variables). The time steps are defined as follows:

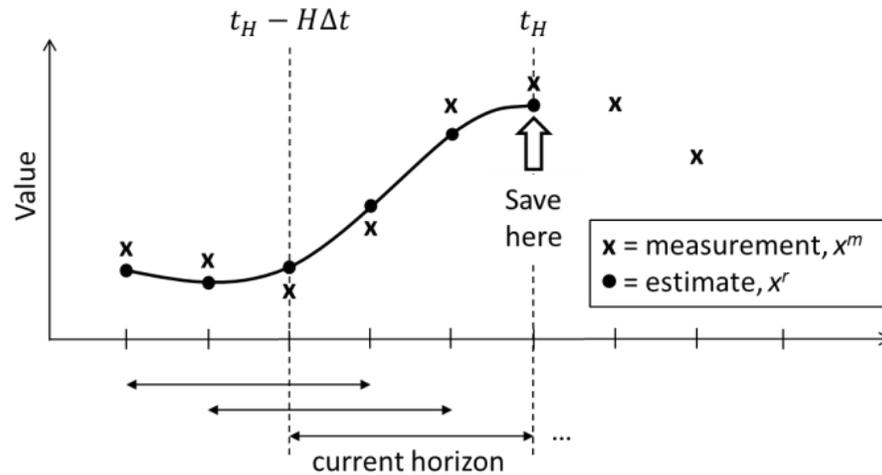
$$t_j = t_0 + j\Delta t \quad (j = 0, 1, 2 \dots H) \quad (2a)$$

$$\Delta t = \frac{t_H - t_0}{H} \quad (2b)$$

where  $t_0$  is the starting discrete time of the horizon and  $\Delta t$  is the duration of a time step in seconds. The horizon length,  $H$ , can be treated as a tuning parameter in MHE, which must be set beforehand. A larger  $H$  can increase model accuracy of the results, but can also propagate too many errors from earlier flawed estimates. Figure 2 illustrates how a sample horizon with  $H = 3$  moves in time.

The MHE has the following 4 steps:

1. Obtain measurements and set the value of  $H$ .
2. Optimize Eq. (1) for all  $\mathbf{x}^r(t)$  over  $t_0 \leq t \leq t_H$  where  $t_0 = t_H - H\Delta t$ .
3. Save only  $\mathbf{x}^r(t_H)$  for on-line data reconciliation.
4. Repeat step 2 at next time step  $t_H := t_H + \Delta t$ .



**Figure 2.** Horizon movement illustration for  $H = 3$

The algorithm indicates that the intended application is for on-line data reconciliation. This means that as  $N$  new measurements are collected in the next discrete sampling time,  $t_H$ , new estimates are produced for that sampling time,  $x^r(t_H)$ , by solving Equation (1) within a history of  $H$  time steps from the past. Thus, the profile of estimates remains as smooth as  $H$  permits. In practice, MHE is applied to raw measurements before feeding them to the Distributed Control System (DCS). Smooth profiles help attenuate excessive control action previously brought about by noisy and faulty measurements.

### 2.1 Optimization Methods for NDDR

Two widely used NLP techniques for DR in the industry are successive quadratic programming (SQP) and generalized reduced gradient (GRG) [4]. Liebman et al. (1992) used the SQP for NDDR [11]. Tjoa and Biegler (1991) used a hybrid SQP for steady-state data reconciliation [18]. In 1992, Sanchez et al. developed a software package called PLADAT for variable classification and steady-state plant DR using SQP [19].

Today's implementations of simultaneous DR and bias estimation necessitate population-based metaheuristic algorithms due to large process networks. Valdetarro and Schirru (2011) used particle swarm optimization (PSO) in the DR of measurements from a nuclear power reactor [20]. Prata et al. (2010) took real measurement data from an industrial polypropylene reactor, and used PSO for steady-state DR [21]. Wongrat et al. (2005) experimented on a modified genetic algorithm (GA) for nonlinear steady-state DR also [22].

Particularly interesting is the PSO implementation of Prata et al. (2009) to the NDDR problem of an industrial polypropylene reactor [23]. PSO is the chosen optimization solver for Eq. (1) due to its simple yet efficient handling of high-dimensional, multi-modal, nonlinear, discontinuous objective functions. In [23], PSO efficiently estimated 286 state variables and 10 parameters in the reactor model consisting of 9 nonlinear ODEs and 15 algebraic equations in every horizon movement.

Historically, PSO was introduced by Kennedy and Eberhart in 1995 [24]. The simplicity of the algorithm lies in the fact that it does not need to calculate gradients of the objective function. Parsopoulos et al. (2001) [25] were the first to study the use of PSO to solve steady-state DR problems.

In the mentioned DR applications of population-based metaheuristic algorithms above, the Gross Error Detection (GED) case was not handled. When the gross error location is not known, the problem becomes an MINLP, due to the binary decision of whether a gross error is present in a sensor or not. However, if the location is known a priori, Eq. (1) is applicable, and the solution can still be obtained using the standard PSO or GA. This NDDR-GED case is the problem addressed in this paper.

### 3. PROBLEM STATEMENT

This paper describes a new method for nonlinear dynamic data reconciliation and gross error detection (NDDR-GED), which is based on moving horizon estimation (MHE) and a chosen implementation of Particle Swarm Optimization.

The problem is formulated as a series of NLPs (Eq. (1)) solved every  $H$  discrete time steps, where  $H$  is the horizon length. A variant of the PSO, the hybrid Nelder-Mead Simplex and Particle Swarm Optimization (NM-PSO) method, taken from Zahara and Kao (2009) [26] is used to solve the NLPs. It is argued that augmenting the Nelder-Mead Simplex to PSO every iteration leads to faster convergence. Moreover, the use of PSO warrants the use of a learning mechanism in the MHE, where the elite particles of a swarm in the current horizon's NLP are transferred to the next. When applied, succeeding horizon NLPs converge even faster and smoother profiles are also achieved. This would work under the assumption that for smooth profiles, the optimal solution in the current horizon's NLP is just nearby the optimal solution in the next horizon's NLP.

Lastly, the DAE system serving as constraints in the NLP will be discretized by orthogonal collocation on finite elements (OCFE). OCFE was originally developed to solve the model for heat and mass transfer with chemical reaction in a catalyst pellet, which has steep gradients (Carey and Finlayson, 1975) [27]. This problem is of the same type as the adiabatic stirred-tank reactor test case to which the proposed method is to be evaluated.

The overall NDDR algorithm is entirely coded in MATLAB® R2013b. The adiabatic stirred-tank reactor test case is described by a nonlinear dynamic model having 2 state variables and 2 input variables. Gross errors were also doped to the 2 state variables one at a time. Performance is evaluated qualitatively using time-profiles, and quantitatively using % SDR (% std. dev. reduction of measurement error), which is a measure of response smoothness and accuracy.

### 4. METHODOLOGY

This section describes the NDDR-GED method and the details of NM-PSO, OCFE, and the test case.

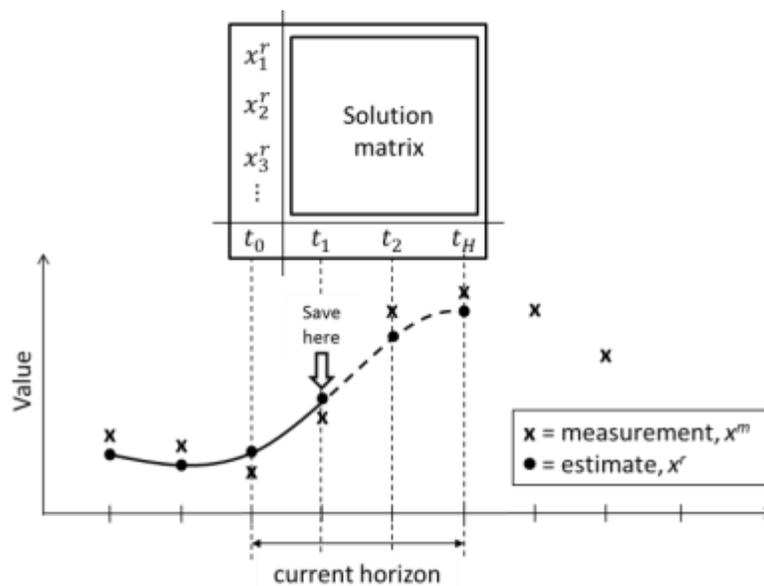
#### 4.1. The Proposed NDDR-GED Algorithm

Some assumptions must be made before presenting the proposed NDDR-GED algorithm. If ever gross error or bias is present in an instrument, the bias persists over time. This means that bias only needs to be estimated at the initial steady-state set of measurements, and its value is treated as a constant over the succeeding horizon NLPs. Furthermore, it is assumed that no other gross error will appear in the other instruments mid-simulation. Lastly, it is assumed that there is no precision degradation in the sensors

throughout the process duration. This means that the standard deviation of measurement noise,  $\sigma$ , does not change with time [28].

The proposed algorithm has the following steps:

1. Obtain process measurements and set  $H$ .
2. Optimize Eq. (1) via NM-PSO at initial steady-state only, to obtain  $\mathbf{x}^r(t_0)$  and  $\mu$ .
3. Start the current horizon from the initial time  $t_0$  to  $t_H = t_0 + H\Delta t$ .
4. Optimize Eq. (1) via NM-PSO for all  $\mathbf{x}^r(t)$  over  $t_1 \leq t \leq t_H$  using  $\mathbf{x}^r(t_0)$  as initial conditions to the DAE constraints.
5. Save only  $\mathbf{x}^r(t_1)$ .
6. Repeat steps 4-5 for the next horizon, where  $t_0 := t_0 + \Delta t$ .



**Figure 3.** Horizon movement for the proposed NDDR-GED MHE for  $H = 3$

The DAE constraints in Eq. (1) are recognized as an initial value problem (IVP). Thus, the recorded estimates at  $t_0 = t_H - H\Delta t$  from the previous horizon are treated as initial conditions to the current horizon. Also, the decision variables in the objective function of Eq. (1) do not include the estimates at  $t_0$ , i.e. the solution matrix on the current horizon has a size  $N \times H$  whose elements are  $\mathbf{x}^r(t_j) \forall j = 1, 2, 3 \dots H$ . In contrast to the work of Liebman et al., the proposed algorithm saves the result at  $\mathbf{x}^r(t_1)$  instead of  $\mathbf{x}^r(t_H)$ . This is illustrated in Figure 3.

#### 4.2. Hybrid Nelder-Mead Simplex and Particle Swarm Optimization

A valuable feature common between the simplex method and particle swarm optimization is that they are both gradient-free methods. The objective function need not to be differentiable in the search space. However, because the NLP is heavily constrained and accuracy is important, a gradient-based repair method was used in constraint handling. The main features of the adopted NM-PSO are:

*1. Initialization.* A population size of  $P$  particles is generated. Each particle is given a random starting position in the search space, and a random starting velocity. As suggested by Zahara and

Kao (2009),  $P = 41N + 1$ , where  $N$  is the number of search space dimensions or optimization decision variables.

2. *Swarming*. The velocity and position of the  $p$ th particle on the  $i$ th iteration is updated using:

$$V_p^{i+1} = c_0 V_p^i + c_1 \times \text{rand} \times (x_p^{\text{best}} - x_p^i) + c_2 \times \text{rand} \times (x_g^{\text{best}} - x_p^i) \quad (3a)$$

$$x_p^{i+1} = x_p^i + \begin{cases} \text{sgn}(V_p^{i+1}) \times V_{\max}, & |V_p^{i+1}| \geq V_{\max} \\ V_p^{i+1}, & |V_p^{i+1}| < V_{\max} \end{cases} \quad (3b)$$

where  $V_p^i$  is the current velocity,  $V_p^{i+1}$  is the updated velocity,  $x_p^i$  is the current position,  $x_p^{i+1}$  is the updated position,  $x_p^{\text{best}}$  is the particle's best position so far, and  $x_g^{\text{best}}$  is the global best particle position so far. Both  $c_1$  and  $c_2$  are set to 2.0, while  $c_0$  is linearly decreasing from 0.9 to 0.4, according to

$$c_0 = 0.9 - 0.5 \times \left( \frac{i-1}{I-1} \right), \quad 1 \leq i \leq I \quad (4)$$

where  $I$  is the number of iterations and  $c_0$  is computed every iteration,  $i$ . The linearly decreasing inertia weight allows global search at the onset of swarming (exploration), and a more local search towards the end (exploitation). The function  $\text{rand}()$  is a random value from a uniform distribution inside  $(0, 1)$ . The function  $\text{sgn}()$  returns -1, 0, or 1 depending on the sign of the input value.

3. *Gradient Repair Method*. This constraint handling technique, proposed by Chootinan and Chen (2006) [29], is used to direct the infeasible particles toward the feasible region after every velocity and position update. An infeasible particle at position  $\mathbf{x}^t$  is moved to a new position  $\mathbf{x}^{t+1}$  according to

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \nabla_{\mathbf{x}} \mathbf{V}^{-1} \times \Delta \mathbf{V} \quad (5a)$$

$$\Delta \mathbf{V} = \begin{bmatrix} \text{Min}\{0, -\mathbf{g}(\mathbf{x})\} \\ -\mathbf{h}(\mathbf{x}) \end{bmatrix}_{M \times 1} \quad (5b)$$

$$\nabla_{\mathbf{x}} \mathbf{V} = \begin{bmatrix} \nabla_{\mathbf{x}} \mathbf{g} \\ \nabla_{\mathbf{x}} \mathbf{h} \end{bmatrix}_{M \times N} \approx \frac{1}{e} \cdot \begin{bmatrix} \mathbf{g}(\mathbf{x}+e) - \mathbf{g}(\mathbf{x}) \\ \mathbf{h}(\mathbf{x}+e) - \mathbf{h}(\mathbf{x}) \end{bmatrix}_{M \times N} \quad (5c)$$

where  $\Delta \mathbf{V}$  is the degree of constraint violation,  $\nabla_{\mathbf{x}} \mathbf{V}$  is the derivative of the constraints with respect to the position vector  $\mathbf{x}$ ,  $e$  is a small perturbation factor to approximate the gradient,  $M$  is the total number of constraints, and  $N$  is the number of optimization decision variables. For the case when  $\nabla_{\mathbf{x}} \mathbf{V}$  is not a square matrix, the Moore-Penrose pseudoinverse is used for  $\nabla_{\mathbf{x}} \mathbf{V}^{-1}$ .

For nonlinear constraints, Eq. (5) does not guarantee repair in one application to an infeasible particle. It is repeated successively until the particle reaches the feasible region, but at most 5 times only. If the 5th application is not enough to repair the particle, it is arbitrarily deemed unrepairable.

4. *Constraint Fitness Ranking*. Because of the occurrence of unrepairable particles, all particles are ranked based on their constraint fitness  $\text{Cf}(\mathbf{x})$ . Let  $M$  be the total number of constraints and  $Q$  be the

number of inequality constraints. Inequality constraints are numbered 1 to  $Q$ , and equality constraints are numbered  $Q + 1$  to  $M$ . The constraint fitness of a particle in position  $\mathbf{x}$  is computed as

$$C_j(\mathbf{x}) = \begin{cases} 0, & \text{if } g_j(\mathbf{x}) \leq 0 \text{ and } j \in [1, Q] \\ g_j(\mathbf{x}), & \text{if } g_j(\mathbf{x}) > 0 \text{ and } j \in [1, Q] \\ |h_j(\mathbf{x})|, & \forall j \in [Q + 1, M] \end{cases} \quad (6a)$$

$$Cf(\mathbf{x}) = \sum_{j=1}^M w_j C_j(\mathbf{x}) \quad \text{such that} \quad \sum_{j=1}^M w_j = 1 \quad (6b)$$

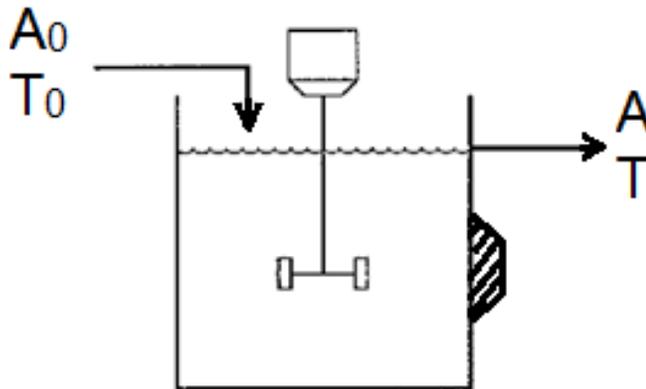
where  $w_j$  is a random weight from the interval  $[0, 1]$  assigned to the  $j$ th constraint. Particles inside the feasible region would have  $Cf(\mathbf{x}) = 0$ , while infeasible particles have  $Cf(\mathbf{x}) > 0$ . The higher the constraint fitness value, the farther the particle from the feasible region. When two particles have the same constraint fitness value, the one with lesser objective function value comes first.

*5. Nelder-Mead Simplex.* If  $N$  is the number of optimization decision variables, then the top  $N+1$  particles in the ranking are fed to the Nelder-Mead simplex routine. A reflection coefficient of 1, contraction coefficient of 0.5, expansion coefficient of 2, and shrinkage coefficient of 0.5 were used.

*6. Termination.* The exit criterion is the set number of iterations. Tolerance is set to  $10^{-12}$ .

#### 4.3. The Adiabatic CSTR Model

The algorithm is tested on the adiabatic CSTR system taken from Seinfeld [30]. Figure 4 shows a diagram of the system. The process dynamic model is written as



**Figure 4.** Adiabatic stirred-tank reactor system

$$\frac{dA}{dt} = \frac{q}{V} (A_0 - A) - \alpha_d k A \quad (7a)$$

$$\frac{dT}{dt} = \frac{q}{V} (T_0 - T) + \alpha_d \left( \frac{-\Delta H_r}{\rho C_p} \right) k A - \frac{U A_c}{\rho C_p V} (T - T_c) \quad (7b)$$

where,

$k = k_0 \exp(-E_A/T)$	is an Arrhenius rate expression,
$A_0$	= feed conc., gmol/cm <sup>3</sup> ,
$T_0$	= feed temperature, K,
$A$	= tank conc., gmol/cm <sup>3</sup> ,
$T$	= tank temperature, K.

The CSTR system contains a first-order exothermic reaction, where the tank concentration ( $A$ ) and tank temperature ( $T$ ) are the *state variables*, and the feed concentration ( $A_0$ ) and feed temperature ( $T_0$ ) are the *input variables*. Other physical constants in the dynamic model are shown in Table 1. The initial steady-state of the CSTR system is at the following exact values  $\mathbf{x}^e$  of scaled variables:  $A_0 = 6.5$ ,  $T_0 = 3.5$ ,  $A = 0.152472$  and  $T = 4.609221$ . Refer to [11] for details of the scaling. State and input variables in this paper, henceforth, are scaled.

**Table 1.** Physical data for the Adiabatic CSTR

Parameter	Value	Units
$q$	10.0	cm <sup>3</sup> s <sup>-1</sup>
$V$	1000.0	cm <sup>3</sup>
$\Delta H_r$	-27,000.0	cal gmol <sup>-1</sup>
$\rho$	0.001	g cm <sup>-3</sup>
$C_p$	1.0	cal (g K) <sup>-1</sup>
$U$	$5.0 \times 10^{-4}$	cal (cm <sup>2</sup> s K) <sup>-1</sup>
$A_R$	10.0	cm <sup>2</sup>
$T_c$	340.0	K
$k_0$	$7.86 \times 10^{12}$	s <sup>-1</sup>
$E_A$	14,090.0	K
$\alpha_d$	1.0	-

To simulate transient behavior, the feed concentration  $A_0$  was abruptly increased from 6.5 to 8.5 at  $t = 100$  sec, and then to 5.5 at  $t = 150$  sec.  $T_0$  was held constant. The response of  $A$  and  $T$  are produced over a period of 250 seconds, with samples taken every  $\Delta t = 2.5$  seconds.

To generate noise, measured values were taken from a normal distribution, centered on the simulated true response, with a standard deviation of  $\sigma = 0.15$  for all variables. In addition, estimates were imposed to be bounded within  $\pm 3\sigma$  of the measured values. To simulate gross errors, a bias of  $+10\sigma$  is doped on the measured variables A and T.

4.4. Orthogonal Collocation on Finite Elements

OCFE solves the IVP in Eq. (7) at discrete time steps, and also at interior points within a time step for added accuracy. Each time step constitutes a finite element. A finite element is first scaled to  $0 \leq t \leq 1$ , where the interior points are set to the roots of some orthogonal polynomial, so chosen that the interpolated values at these points accurately represent the solution inside a time step. In this work, a 4<sup>th</sup>-order collocation method is done using the 2<sup>nd</sup>-order shifted Legendre polynomial,

$$P_2(x) = 6x^2 - 6x + 1 \tag{8}$$

whose roots are  $x_1 = (\sqrt{3} - 1)/(2\sqrt{3})$  and  $x_2 = (\sqrt{3} + 1)/(2\sqrt{3})$ . The discrete times  $[0, x_1, x_2, 1]$  are referred to as the *collocation points* per finite element. In OCFE, the ODEs in Eq. (7) are written as Eq. (9), and their solution is given by Eq. (10).

$$\frac{dy(t)}{dt} = f(y(t), t) \tag{9}$$

$$y(t) = \sum_{i=0}^{N_c-1} \phi_i(t) y(t_i) \tag{10a}$$

$$\phi_i(t) = \prod_{\substack{m=0 \\ m \neq i}}^{N_c-1} \left( \frac{t - t_m}{t_i - t_m} \right) \tag{10b}$$

where  $y(t)$  is the vector  $[A(t) \ T(t)]^T$ ,  $N_c = 4$  is the number of collocation points,  $t_m$  and  $t_i$  are collocation points, and  $\phi_i(t)$  are called *basis functions*. Basis functions are defined like so in Eq. (10a) because it is the only way to define a rational polynomial function such that  $\phi_i(t_m) = 1$  when  $m = i$ , and  $\phi_i(t_m) = 0$  when  $m \neq i$ . Differentiating Eq. (10) gives the discretized derivatives as Eq. (11). The discretized form of Eq. (7) is written in Eq. (12) for all  $j = 0, 1, 2 \dots N_c$ ,

$$\frac{dy(t)}{dt} \cong \frac{1}{\Delta t} \sum_{i=0}^{N_c-1} \left[ \frac{d\phi_i(t)}{dt} y(t_i) \right] \tag{11}$$

$$\frac{1}{\Delta t} \sum_{i=0}^{N_c-1} \left[ \frac{d\phi_i(t)}{dt} \Big|_{t=t_j} A(t_i) \right] = f_A(A(t_j), T(t_j), A_0(t_j), t_j) \tag{12a}$$

$$\frac{1}{\Delta t} \sum_{i=0}^{N_c-1} \left[ \frac{d\phi_i(t)}{dt} \Big|_{t=t_j} T(t_i) \right] = f_T(A(t_j), T(t_j), T_0(t_j), t_j) \tag{12b}$$

where  $f_A$  is the right-hand side of Eq. (7a),  $f_T$  is the right-hand side of Eq. (7b),  $t_i$  and  $t_j$  are the collocation points, and  $\Delta t$  is the size of a time step. The  $\Delta t$  appears in Eq. (12) to scale the finite element from  $0 \leq t \leq 1$  back to  $t_j \leq t \leq t_j + \Delta t$ . Furthermore, it is imposed that the slope of the response functions at the end of one finite element,  $t_e$ , must be equal to slope at the start of the next finite element,  $t_s$ . This statement is written mathematically as Eq. (13).

$$\sum_{i=0}^{N_e-1} \left[ \frac{d\phi_i(t)}{dt} \Big|_{t=t_i} y(t_i) \right] = \sum_{i=0}^{N_e-1} \left[ \frac{d\phi_i(t)}{dt} \Big|_{t=t_i} y(t_i) \right] \quad (13)$$

Equations (12) and (13) constitute the system of purely nonlinear algebraic equations that takes the place of  $f$  and  $h$  in Eq. (1) as equality constraints during optimization. To demonstrate how large and how sparse the system is with respect to a given horizon length, Figure 5a shows an incidence matrix of the system for  $H = 3$ . For Figure 5 only, the notation  $t_j^{(i)}$  denotes the  $j$ th collocation point in the  $i$ th finite element (both zero-based). These points are depicted in the moving horizon in Figure 5b. For  $H = 3$ , there are 20 algebraic equations (rows) and 28 estimates (columns). Rows 4, 7, 14, and 17 are Eq. (13), and the rest correspond to Eq. (12).

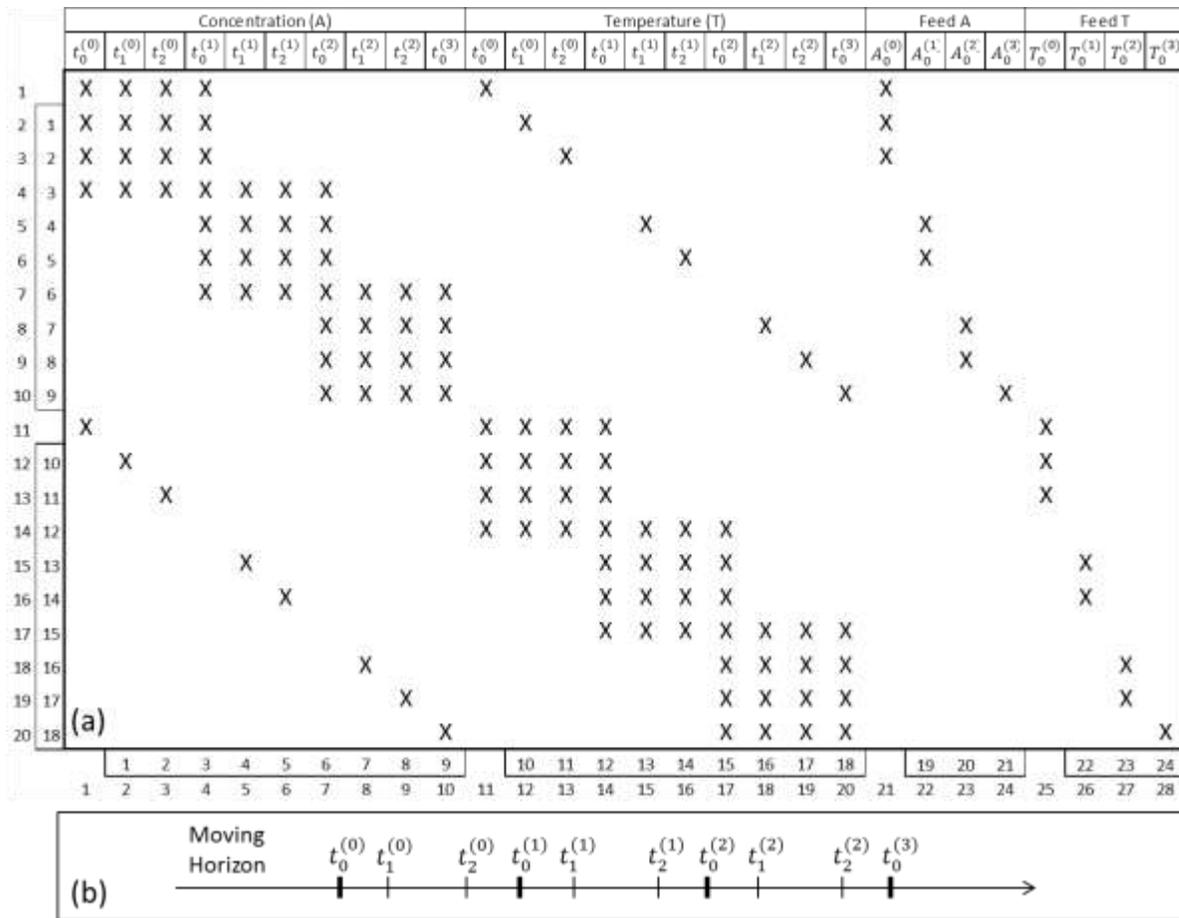


Figure 5. Discretized ODEs ( $H = 3$ ): (a) incidence matrix; (b) collocation points

In this work, input variables  $A_0$  and  $T_0$  are assumed to be constant on all collocation points within a finite element, implying that input step changes appear only at finite element boundaries.

It was noted in Step 4 of the proposed algorithm that saved values from the previous window are to be used as initial conditions in the next window. These initial conditions are  $A(t_0^{(0)})$ ,  $T(t_0^{(0)})$ ,  $A_0^{(0)}$ , and  $T_0^{(0)}$ . In effect, the NLP per window reduces to 24 estimates and 18 equations for  $H = 3$ . These correspond to the boxed row and column labels in Figure 5a. Note that the incidence matrix dimensions increase with increasing  $H$ .

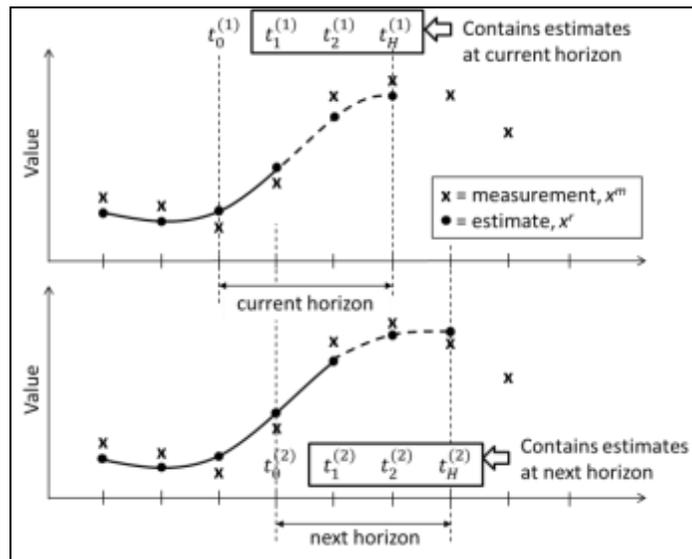
#### 4.5. A Novel Learning Mechanism

Formerly, the idea is to let NM-PSO solve the NLP at each horizon movement as if each NLP is unrelated. However, doing so requires a lot of computational burden. Most of the computing time will be spent on the gradient repair subroutine.

The gradient repair involves taking the pseudo-inverse of a 24 X 18 matrix (for  $H = 3$ ), and do it 5 times per particle, multiplied by around 200 particles, times 100 iterations, times 100 horizon movements. It would take days to finish a simulation run in this setup, what more at higher  $H$ .

The novel learning mechanism is a necessary solution to speed up the computation. Recognizing that the profile of estimates should be smooth, the solution to the NLP in the current horizon must be just around the vicinity of the solution to the NLP in the next horizon. Thus, it is proposed that  $N+1$  elite particles from the swarm of the current horizon are to be transferred to the initial swarm of the next horizon. This way, the reconciliation learns from its previous experiences.

The number of iterations in the NM-PSO can now be reduced from 100 to 10 (arbitrarily chosen) while still expecting smooth results, except for the first horizon NLP where the NDDR has no experience to learn from yet.



**Figure 6.** Discrete times in the current and the next horizon ( $H = 3$ )

On a final note, one must be careful in “transferring” the elite particles. Each elite particle carries its own solution matrix containing the estimates in the discrete times of the current horizon. However, as illustrated in Figure 6, the discrete times in the current horizon are different from the discrete times in the next horizon. Therefore, the solution matrix from the current horizon must be modified to correspond to the discrete times in the next horizon, prior to the “transfer of elites”.

Let  $t_j^{(i)}$  be the  $j$ th discrete time at the  $i$ th horizon, where  $i = 1$  is the current horizon. To properly “transfer” the elites:

- (1) the estimates at  $t = t_1^{(1)}$  must be removed;
- (2) the solution matrix is shifted one time step to the left; and,
- (3) the estimates at  $t = t_H^{(2)}$  must have an initial guess, which, in this work, is  $\mathbf{x}^r(t_H^{(2)}) = \mathbf{x}^r(t_H^{(1)})$ .

### 5. ILLUSTRATIVE CASE STUDY

In this section, the proposed NDDR-GED algorithm as applied to the measurements from an adiabatic CSTR is evaluated in two experiments: (a) No gross error; and (b) Single gross error. Performance evaluation is done both quantitatively and qualitatively.

The quantitative measure for performance is given by % SDR or percent standard deviation reduction of measurement error. Since the estimates are expectedly smoother than the measurements, the standard deviation of the measurement error must decrease after NDDR. Here, measurement error means the difference between the measurement and the exact values. Percent SDR is computed by

$$\% \text{ SDR} = \frac{\text{MESD} - \text{EESD}}{\text{MESD}} \tag{14}$$

where

$$\begin{aligned} \text{ME} &= \mathbf{x}^m - \mathbf{x}^e; & \text{MESD} &= \frac{1}{N-1} \sqrt{\sum (\text{ME} - \overline{\text{ME}})^2} \\ \text{EE} &= \mathbf{x}^r - \mathbf{x}^e; & \text{EESD} &= \frac{1}{N-1} \sqrt{\sum (\text{EE} - \overline{\text{EE}})^2} \end{aligned}$$

In the hypothetical case when all  $\mathbf{x}^r = \mathbf{x}^e$ , then  $\text{EE} = 0$ ,  $\text{EESD} = 0$ , and  $\% \text{ SDR} = 100\%$ . On the other hand, the worst case happens when  $\text{EE} > \text{ME}$ , which means that after the application of NDDR, the estimates are farther from the true values than the raw measurements. In that case, % SDR is negative.

#### 5.1. No Gross Error

The average, worst, and best performance of the proposed NDDR algorithm are presented in Figures 7, 8, and 9, respectively. In Table 2, the % SDR of the best performance is compared between 2 other works, at different chosen horizon lengths.

**Table 2.** % SDR of best-performance NDDR (H = 3) compared to other works

	Current Work (H = 3)			Liebman et al. (1992) (H = 10)			Tian et al. (2006) (H = 10)		
	MESD	EESD	% SDR	MESD	EESD	% SDR	MESD	EESD	% SDR
A	0.0817	0.0043	94.53%	0.0559	0.0068	87.8%	0.0578	0.0084	83.58%
T	0.0829	0.0069	91.72%	0.0546	0.0125	77.1%	0.0509	0.0121	76.67%
A <sub>0</sub>	0.0940	0.1168	-	0.0484	0.1757	-	0.0512	0.1833	-
T <sub>0</sub>	0.0842	0.1137	-	0.0536	0.0184	65.7%	0.0521	0.0142	72.11%

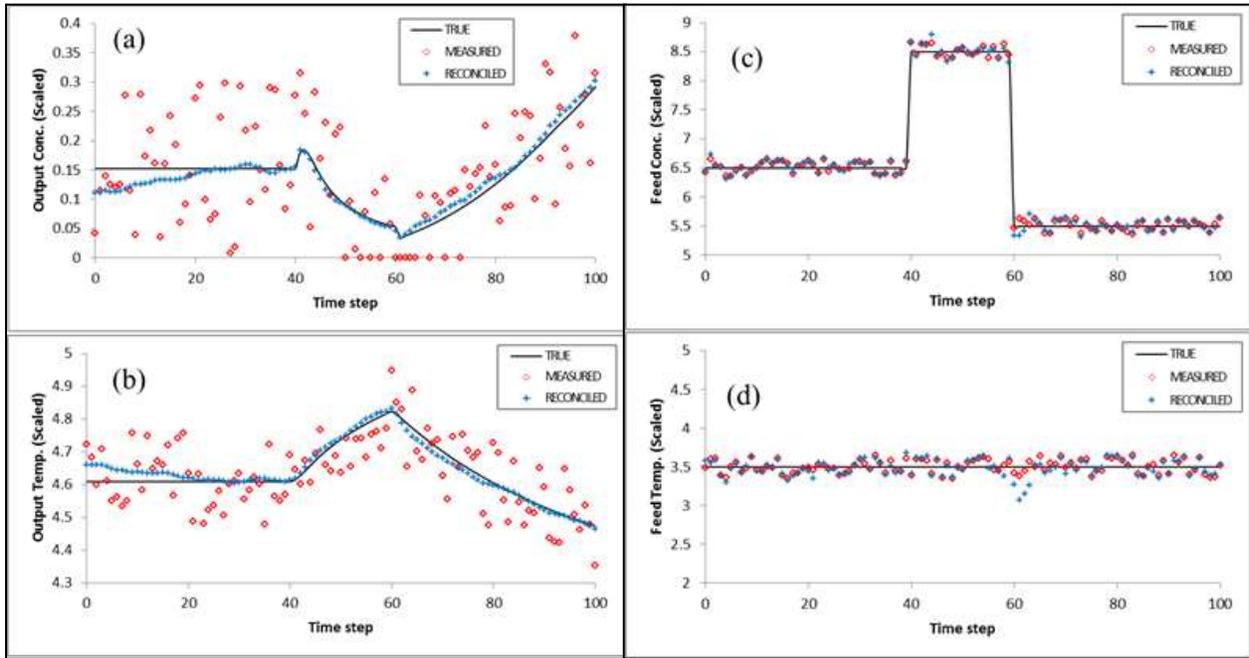


Figure 7. Average performance of NDDR ( $H = 3$ ): (a)  $A$ , (b)  $T$ , (c)  $A_0$ , (d)  $T_0$

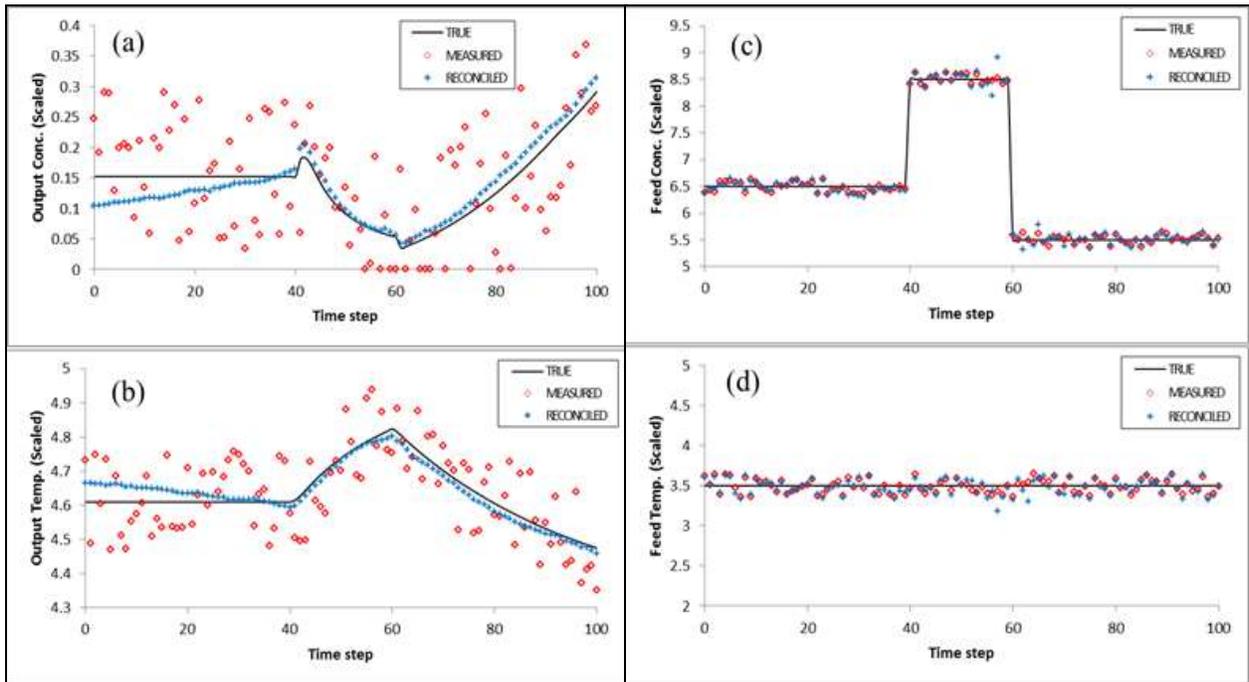
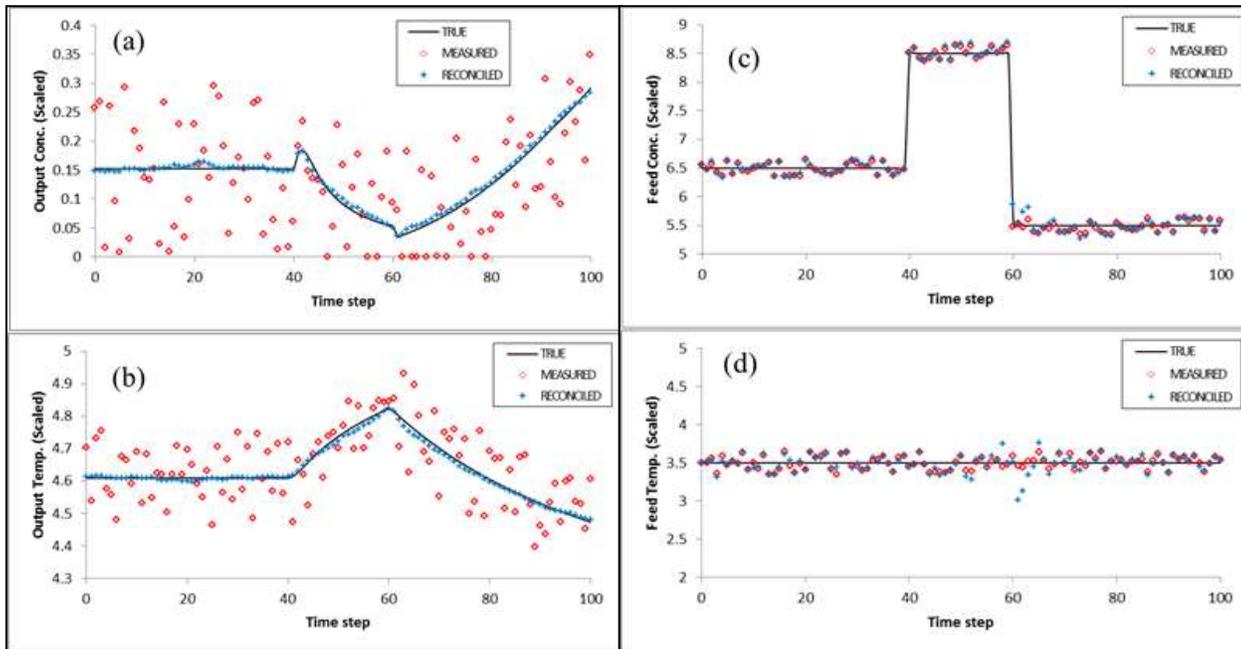


Figure 8. Worst performance of NDDR ( $H = 3$ ): (a)  $A$ , (b)  $T$ , (c)  $A_0$ , (d)  $T_0$



**Figure 9.** Best performance of NDDR ( $H = 3$ ): (a)  $A$ , (b)  $T$ , (c)  $A_0$ , (d)  $T_0$

The omitted % SDR data in Table 2 indicates that % SDR is negative, i.e. the estimate errors are worse than the measurement errors. For the state variables,  $A$  and  $T$ , however, the estimates improved greatly from the raw measurements, even greater than the improvement from literature. Recall that Liebman et al. used SQP, instead of PSO, to solve the NLP. They also used OCFE for discretization. Meanwhile, Tian et al. (2006) used a transfer function matrix model instead of the OCFE [31].

Take note that the proposed NDDR-GED algorithm achieved the better % SDR even at a lower horizon length ( $H = 3$ ) than those used in literature. However, for the input variables,  $A_0$  and  $T_0$ , longer horizon lengths are needed to achieve better results.  $A$  and  $T$  measurements can be reconciled more readily than  $A_0$  and  $T_0$  because they have more *spatial redundancy*, i.e. the way they appear in the balances makes their values more constrained. The values of  $A_0$  and  $T_0$  have more freedom to adjust, and so they need more *temporal redundancy* (longer horizon length) to keep them straight.

Aside from the % SDR, the average accuracy of all estimates with respect to satisfying the process model was also computed. Model accuracy in Figure 7 was found to be  $2.48 \times 10^{-7}$ . Even though the estimates satisfy constraints and minimize Eq. (1), they are only binding within a limited time-window of the model solution.

There are 2 things to note, qualitatively, from Figures 7 to 9: (a) at the onset, estimates are far from the true response, but eventually pick up accuracy. This is because the algorithm takes time before it can capture reality with enough measurement history; (b) estimates are sometimes found to be persistently inaccurate within a period of time (e.g. at time steps 63 to 80 in both  $A$  and  $T$ ). This means that the algorithm failed to locate the global minimum in one time step, but caused the swarm to do the same in the next time steps thanks to the *learning mechanism* (Section 4.5). Almost always, the global minimum fails to be located during rapid transient behavior. Nevertheless, the inaccuracy is smoothly removed in time, discouraging abrupt changes in control action to the process. This is advantageous because control

hardware is saved from wear and tear. Thus, the *learning mechanism* is found to be a double-edged sword to NDDR.

### 5.2. Single Gross Error

The final objective is to test the proposed NDDR-GED algorithm for the adiabatic CSTR test case with a single gross error. From experience, a horizon length of 3 is enough to produce desired results. A bias of magnitude  $+10\sigma$  is doped on each variable A and T. Results are shown in Figures 10 and 11, respectively.

**Table 3.** % SDR of NDDR-GED results with gross error ( $H = 3$ )

	Grossed Variable: A			Grossed Variable: T		
	MESD	EESD	% SDR	MESD	EESD	% SDR
A	0.0941	0.0105	88.86%	0.0813	0.0106	86.98%
T	0.0911	0.0205	77.54%	0.0899	0.0104	88.48%
A <sub>0</sub>	0.0799	0.1709	-	0.0836	0.1741	-
T <sub>0</sub>	0.0892	0.1947	-	0.0856	0.1789	-

Both results are shown to have successfully estimated the bias from the initial steady-state NLP, and eliminate accordingly. For NDDR performance, the same qualitative observations are seen in both figures. However, it is worth mentioning that persistent inaccuracies in variables A and T and lack of smoothness in A<sub>0</sub> and T<sub>0</sub> are more severe in the presence of bias.

Table 3 shows a quantitative evaluation under the presence of bias. The acceptable % SDR for variables A and T shows that their response is smooth. But the negative % SDR in variables A<sub>0</sub> and T<sub>0</sub>, again, indicates that H needs to be increased to achieve smoothness. Due to the  $O(n^3)$  time complexity in gradient repairs, higher values of H would entail a greater computational burden. Thus, they will be deferred to future works.

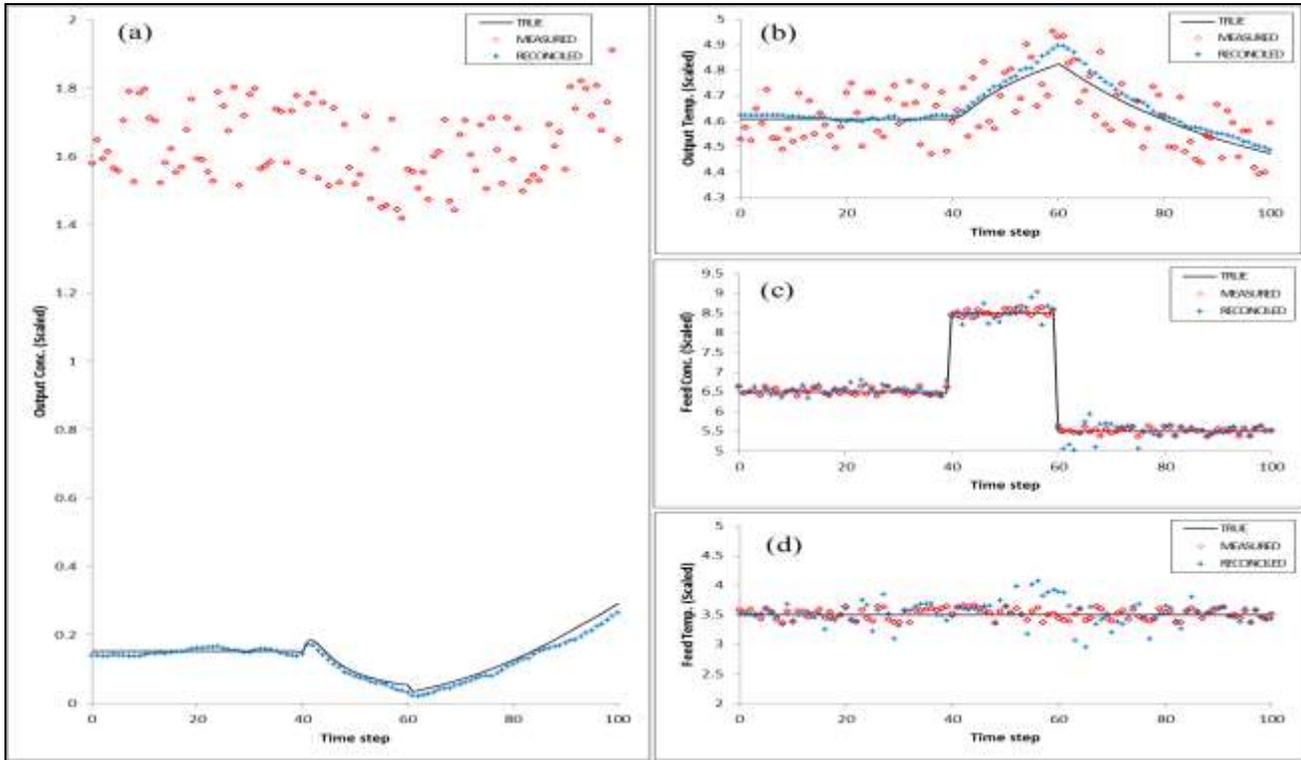


Figure 10. NDDR-GED results ( $H = 3, \mu_A = +10\sigma$ ): (a) A, (b) T, (c)  $A_0$ , (d)  $T_0$

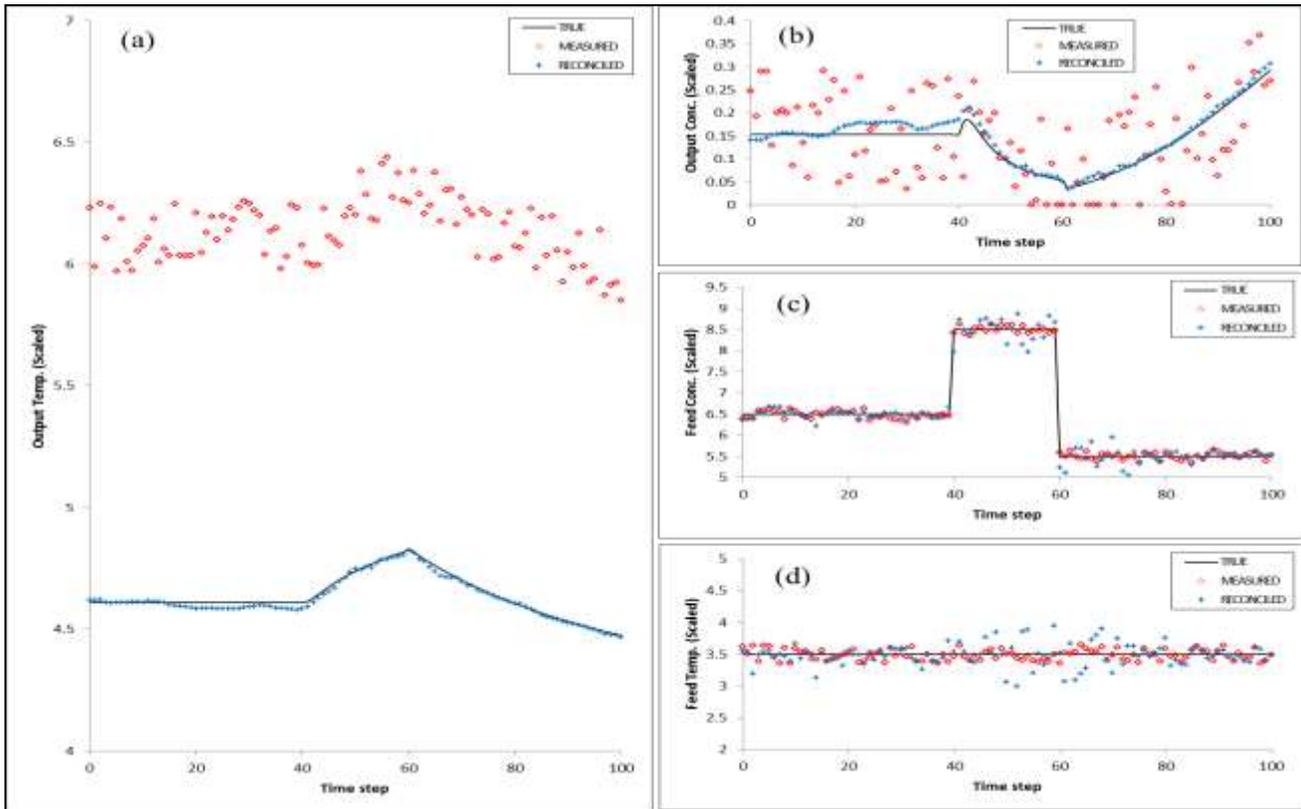


Figure 11. NDDR-GED results ( $H = 3, \mu_T = +10\sigma$ ): (a) T, (b) A, (c)  $A_0$ , (d)  $T_0$

## 6. CONCLUSION AND RECOMMENDATIONS

An algorithm for NDDR-GED was developed using the moving horizon approach. It was assumed that gross errors are persistent over time and that no additional gross errors are to be detected from other sensors mid-simulation. The features of the algorithm include the hybrid Nelder-Mead Simplex and Particle Swarm Optimization (NM-PSO) method, a novel learning mechanism, and orthogonal collocation on finite elements (OCFE) for discretizing the process model DAE. The learning mechanism is warranted by the use of a population-based metaheuristic optimization algorithm in solving the NDDR NLP.

The algorithm was, then, tested to an adiabatic CSTR test case with a first-order exothermic reaction. Quantitatively, the performance of the algorithm is satisfactory, having a % SDR of up to 94%, better than those from literature even at a lower horizon length. The reconciled estimates were accurate to the model within  $10^{-7}$ . Also, biased variables are accounted for effectively. However, the main problem is the lack of smoothness for the variables  $A_0$  and  $T_0$ , as well as inaccuracies in variables  $A$  and  $T$  during rapid transient states. This is attributed to: (a) inadequate redundancy by using a small horizon length; and, (b) failure of the NM-PSO to locate the global minimum (or the correct one if there are multiple global minima); and, (c) the learning mechanism misleading the swarm to an erroneous global minimum for a period of time.

In this work, the choice of  $H = 3$  suffices to demonstrate the effectiveness of the algorithm. It was not chosen by a tuning method, but chosen based from experience only. Thus, the recommendation is to test increasing horizon lengths in the MHE. However, a faster and just-as-reliable constraint handling technique should be used. If gradient repair is still desired, a parallel computing package can be used for improving speed, such as in MATLAB<sup>®</sup>.

## 7. REFERENCES

- [1] Bai, S. and Thibault, J., Dynamic Data Reconciliation: Theory and Practice. USA: Verlag Dr. Muller Aktiengesellschaft & Co. KG, 2010.
- [2] Romagnoli, J. A. and Sanchez, M. C., Data Processing and Reconciliation for Chemical Process Operations. San Diego, CA: Academic Press, 2000.
- [3] Kuehn, D. R., and Davidson, H., "Computer Control II-Mathematics of control," Chemical Engineering Progress, 57 (6): 44-47 (1961).
- [4] Narasimhan, S. and Jordache, C., Data Reconciliation and Gross Error Detection: An Intelligent Use of Process Data. Houston, TX: Gulf Publishing Company, 2000.
- [5] Seborg, D. E., Edgar, T. F., and Mellichamp, D. A., Process Dynamics and Control. 2nd Ed. John Wiley & Sons Inc.: 2004.
- [6] Yamamura, K., Nakajima, M., and Matsuyama, H., "Detection of gross errors in process data using mass and energy balances," International Chemical Engineering, 28 (1): 91-97, 1988.
- [7] Gelb, A. (Ed.), Applied Optimal Estimation. MIT Press: Cambridge, MA, 1974.
- [8] Stanley, G. M. and Mah, R. S. H., "Estimation of flows and temperatures in process networks," AIChE Journal, 23 (5): 642-650 (1977).

- [9] Haseltine, E. and Rawlings, J., "A Critical Evaluation of Extended Kalman Filtering and Moving Horizon Estimation," *Ind. Eng. Chem. Res.*, 44: 2451-2460, 2005.
- [10] Jang, S., Joseph, B., and Mukai, H., "Comparison of two approaches to on-line parameter and state estimation of nonlinear systems," *Ind. Eng. Chem. Proc. Des. Dev.*, 25 (1986): 809-814.
- [11] Liebman, M. J., T. F. Edgar and L. S. Lasdon, "Efficient data reconciliation and estimation for dynamic processes using nonlinear programming techniques," *Comput. Chem. Engng.*, 16 (1992): 963-986.
- [12] Vachhani, P., Rengaswamy, R., Gangwal, V., and Narasimhan, S., "Recursive estimation in constrained nonlinear dynamical systems," *AICHE Journal*, 51 (3):946-959 (2005).
- [13] Zhang, Z. and Chen, J., "Simultaneous data reconciliation and gross error detection for dynamic systems using particle filter and measurement test," *Comput. Chem. Engng.*, 69 (2014): 66-74.
- [14] Zhang, Z. and Chen, J., "Correntropy based data reconciliation and gross error detection and identification for nonlinear dynamic processes," *Comput. Chem. Engng.*, 75 (2015): 120-134.
- [15] Taylor, J. and Laylabadi, M., "A novel adaptive nonlinear dynamic data reconciliation and gross error detection method," *Proceedings of the 2006 IEEE International Conference on Control Applications*, 1783-1788.
- [16] Miao, Y., Su, H., Wang, W., Chu, J., "Simultaneous data reconciliation and joint bias and leak estimation based on support vector regression," *Comput. Chem. Engng.*, 35 (10): 2141-2151 (2011).
- [17] Bai, S., McLean, D., and Thibault, J., "Autoassociative neural networks for robust dynamic data reconciliation," *AICHE Journal*, 53 (2007): 438-448.
- [18] Tjoa, I. and Biegler, L., "Simultaneous strategies for data reconciliation and gross error detection of nonlinear systems," *Comput. Chem. Engng.*, 15 (10): 679-690 (1991).
- [19] Sanchez, M., Bandoni, A., and Romagnoli, J., "PLADAT: A package for process variable classification and plant data reconciliation," *Comput. Chem. Engng.*, 16 (1992): S499-S506.
- [20] Valdetaro, E. D. and Schirru, R., "Simultaneous model selection, robust data reconciliation and outlier detection with swarm intelligence in a thermal reactor power calculation," *Ann. Nucl. Energy*, 38 (2011): 1820-1832.
- [21] Prata, D. M., Schwaab, M., Lima, E. L., and Pinto, J. C., "Simultaneous robust data reconciliation and gross error detection through particle swarm optimization for an industrial polypropylene reactor," *Chem. Eng. Sci.*, 65 (2010): 4943-4954.
- [22] Wongrat, W., Srinophakun, T., and Srinophakun, P., "Modified genetic algorithm for nonlinear data reconciliation," *Comput. Chem. Engng.*, 29 (2005): 1059-1067.
- [23] Prata, D. M., Schwaab, M., Lima, E. L., and Pinto, J. C., "Nonlinear Dynamic Data Reconciliation and Parameter Estimation through Particle Swarm Optimization: Application for an Industrial Polypropylene Reactor," *Chem Eng. Sci.*, 64 (2009): 3953-3967.
- [24] Kennedy J., and Eberhart R., "Particle swarm optimization," *Proceedings of the IEEE Conference on Neural Networks IV, Piscataway, NJ, USA; 1995.*
- [25] Parsopoulos, K.E., Laskari, E.C., Vrahatis, M.N., "Solving l-1 norm errors-in- variables problems using particle swarm optimization," In: *Proceedings of the Conference on Artificial Intelligence and Applications*, pp. 185-190 (2001).
- [26] Zahara, E. and Kao, Y., "Hybrid Nelder-Mead simplex and particle swarm optimization for constrained engineering design problems," *Expert Systems with Applications*, 36 (2009): 3880-3886.
- [27] Carey, G. and Finlayson, B., "Orthogonal collocation on finite elements," *Chem. Engng. Sci.*, 30 (1975): 587-596.

- [28] Dunia, R., Qin, S. J., Edgar, T. F., and McAvoy, T. J., "Identification of faulty sensors using principal component analysis," *AIChE Journal*, 42 (10): 2797-2812 (1996).
- [29] Chootinan, P. and Chen, A., "Constraint handling in genetic algorithms using a gradient-based repair method," *Computers and Operations Research*, 33 (2006): 2263-2281.
- [30] Seinfeld, J., "Optimal stochastic control of nonlinear systems," *AIChE Journal*, 16 (6): 1016-1022 (1970).
- [31] Tian, X., Xia, B., Yu, Z., and Yang, S. H., "Non-linear dynamic data reconciliation for industrial processes," *Proceedings of the 2006 IEEE International Conference on Systems, Man, and Cybernetics*, 5291-5296.

## 8. NOMENCLATURE

Symbol	Description	Units
A	tank concentration	gmol/cm <sup>3</sup>
T	tank temperature	K
A <sub>0</sub>	inlet flow concentration	gmol/cm <sup>3</sup>
T <sub>0</sub>	inlet flow temperature	K
x <sup>m</sup>	value of measured variable	-
u	value of unmeasured variable	-
x <sup>e</sup>	exact/true value	-
ε	random error	-
μ	gross error/bias	-
σ	standard deviation	-
x <sup>r</sup>	reconciled estimate	-
P	number of particles	-
I	number of iterations	-
N	number of decision variables	-
H	horizon length	-
t	discrete sampling time	sec
% SDR	percent std. dev. reduction	-

