16

Global and Dynamic Optimization using the Artificial Chemical Process Paradigm and Fast Monte Carlo Methods for the Solution of Population Balance Models

Roberto Irizarry
DuPont Electronic Technologies. 14 T.W. Alexander Drive P.O. Box 13999.
Research Triangle Park, NC 27709-3999.
USA

1. Introduction

Global and dynamic optimization of engineering problems usually involves complex physico-chemical models as constraints. These models are in general highly non-linear, resulting in multimodal optimization problems. The model may have discontinuous behavior and/or include a very large set of variables. As the complexity of the systems increases, equation-free modeling is becoming more common (Kevrekidis, Gear and Hummer, 2004). For example, in particle dynamics, population balance models are sometimes more effectively solved by the Monte Carlo method.

Stochastic global optimization methods are very important algorithms for the solution of these types of problems. They have been successfully applied to solve challenging problems that cannot be solved using gradient based methods. Stochastic optimization methods have also been used in many algorithms, in which solution of optimization problems is part of the algorithm. Global stochastic optimization strategies have been utilized in learning phase of pattern recognition algorithms using fuzzy logic (Irizarry, 2005b) and neuro-fuzzy systems (Lin, 2008). These methods have been used for the optimization of complex engineering designs involving computational fluid mechanics such as aerodynamics applications (Duvigneau and Visonneau, 2004). Other applications include the determination of molecular structures, including protein structure prediction and protein-small molecule interactions among others (Sahinis, 2009). Batch scheduling problems are another type of problem were stochastic optimization can be very efficient (Liu et al., 2010).

In particular, the solution of dynamic optimization problems is also of great industrial importance for process development and process optimization, since most processes are dynamic. In this type of problem an optimal profile function is sought (vs. an optimal value for a set of variables). For example, in a fed-batch fermenter, the feed-rate schedule is optimized to maximize production of antibiotics, vitamins, enzymes, and other products (Banga et al., 2003). Another example is the determination of optimal temperature profiles in crystallization processes to control crystal size distribution (Ma, Tafti and Braatz, 2002). Dynamic optimization is also of central importance to the application of process control

using model predictive control (Banga, Irizarry-Rivera and Seider, 1998; Pistikopoulos, 2009). Model predictive control provides a sequence of control actions over a future time horizon by solving a dynamic optimization problem that covers past and future behavior of the system.

The genetic algorithm (GA) (Holland, 1975; Goldberg, 1989) and simulated annealing (SA) (Kirkpatrick, Gelatt and Vecchi, 1983; Ingber, 1993) are classical stochastic optimization methods used in many applications and new algorithm developments. GA is based on emulating Darwinian evolution in populations. Evolutionary strategies also focus on real decision variables problems using Darwinian evolution concepts (Schwefel, 1995). In these population-based methods, a large set of configurations forms a population, with new generations created by selection, crossover and mutation operators acting on the current population. This evolution process will increase the fitness of the population to a near optimal value. These algorithms strongly depend on the parameters and types of selection, crossover and mutation mechanisms selected. These operators are continuously being improved and redefined for specific applications and problems (as one of many examples see Tang, Sun and Yang, 2010). Other algorithms like the ant colony (Dorigo and Stutzle, 2004)) and particle swarm optimization (Kennedy and Eberhart, 2001) are inspired by cooperative phenomena of animal behavior or agents.

Simulated annealing was designed for combinatorial optimization problems using concepts from statistical physics. In this case, a very low-energy configuration may be achieved by starting at a high temperature and then gradually lowering the temperature using a cooling schedule. The performance of these algorithms depends strongly on the selection of the cooling schedule, which in general needs to be tuned for specific problems. Furthermore, SA does not consider how to select a step change for the next trial solution, which is critical to the success of the algorithm. This needs to be defined by the user for the problem at hand.

This chapter discussed an alternative for global optimization methodology based on a different paradigm known as the artificial chemical process (Irizarry, 2004). The paradigm has been used to design robust dynamic optimization algorithms (Irizarry, 2005a; Irizarry, 2006) and fuzzy logic algorithms (Irizarry, 2005b). Fast MC algorithms of population balance models are also reviewed (Irizarry, 2007a; Irizarry, 2007b). These coarse graining algorithms accelerate simulation speed by an order of magnitude without loss of accuracy, making optimization of these systems feasible in real time. Unlike other lumping or coarse graining strategies, in this strategy the particle integrity is not lost in the coarsening process. This increase in speed allows the efficient solution of parameter identification problems (Irizarry, 2010) and dynamic optimization problems. The LARES algorithm is described in Section 2. A general purpose algorithm to solve dynamic optimization problems is described in Section 3. Section 4 considers fast MC simulation algorithms for the simulation of population balance models. Fast MC strategies are discussed in Section 5. Section 6 discusses how to combines the algorithms into a hybrid strategy to solve problems involving multiple time scales and inherently stochastic variables.

2. Global optimization using the artificial chemical process paradigm

In this section an optimization algorithm called LARES is reviewed. This algorithm is based on an artificial chemical optimization paradigm introduced in 2004 (Irizarry, 2004). To apply the algorithm, the first step is to encode all decision variables, θ, into a set of integer variables with a very small range of possible values (i.e., 2–10). The integer variables are

called **molecules**, and their respective values are called **states**. As a motivating example consider a case were θ is a vector of real variables. The real decision variables can be encoded using a binary representation (similar to GA). Unlike GA, in this encoding each bit of the string is associated with a molecule variable with two possible values (0 and 1), which represents the value of a specific bit in the binary encoding (see Figure 1). In general, any type of decision variables (including real, integer, logical, and combinatorial) can be encoded into a set of molecules, making the algorithm very flexible.

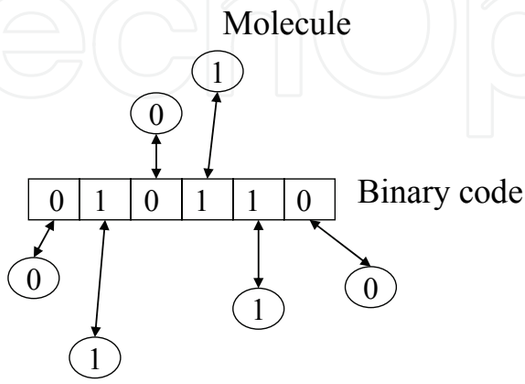


Fig. 1. The concept of molecules for binary encoding of a real variable

Given the decision vector represented as a set of molecules, the LARES algorithm operates on these molecules to create new trial states. At each iteration of the algorithm, a subset of molecules will change state (value) to generate a new trial vector. The artificial chemical plant concept is based on the fact that chemical reactors convert a low-quality material into a high-value product by a series of reactions, feedback loops, and separation steps. The following algorithm is based on an abstraction of those steps.

2.1 The LARES artificial chemical plant paradigm

The LARES algorithm is an iterative improvement methodology, which considers one solution at a time. Given the decision variables encoded into molecules, the algorithm generates a movement of some molecules between four compartments or sets (called L, AR, E, and S). The four compartments are shown in Figure 2 (panel 1). The algorithm starts with all molecules assigned to a set L with an initial state. At each iteration, a set of rules determines the event to be triggered next. Each event is a stochastic subprocess whereby a subset of molecules is moved from one compartment to another. When molecules reach one specific compartment (AR), their state is changed (similar to a reactor in a real chemical plant). The set of rules for the next event selection are based on the previous values of the objective function and the objective function of the best value found so far.

Before describing the algorithm in detail, the different types of possible triggered events are described. Let x_j^g be the state of the molecule j for the best value found so far (or initial trial), and $x^g = (x_1^g, ..., x_V^g)$ the vector of molecular states for the best value found. Let F be the objective function to be minimized. Figure 2 panel 2 shows an example of the state of six molecules for the best value found so far. One possible event consists of a set of molecules being transferred from the Load tank (L set) to the Activation Reactor (AR set), in which the molecules change state to a new random state, $x_j^a \neq x_j^g \ \forall j \in AR$. The state of molecules will

not change while they are inside AR. An example of this type of event is shown in Figure 2 panel 2, where molecules 1 and 4 were moved from L to AR and their states changed from 0 to 1 and 1 to 0, respectively (compare panel 1 and panel 2). This event generates a new trial vector as shown in Figure 2 panel 3, the performance of which is evaluated. In another type of event in a different iteration, some reacted molecules can be sent to the Extraction unit (E set) where they are deactivated back into their previous state upon entering the reactor ($x_j^t = x_j^g \ \forall j \in E$). These extracted molecules could be sent to the Separation unit (S set) or recycled back into the Activation Reactor, where the molecules are reactivated to a new state $x_j^a \neq x_j^g \ \forall j \in E$. At each iteration, a trial vector consists of the activated molecules in AR and the deactivated molecules in the other three sets: $x_j^t = x_j^a \neq x_j^g \ \forall j \in AR$, $x_j^t = x_j^g \ \forall j \notin AR$. If, after any event in the current iteration, a "good batch" is accomplished (i.e., a better objective function is found, $F(\theta_t) \leq F(\theta_b)$, the activation reactor can be emptied into the separator unit, S. In this case all molecules conserve the new state ($x_j^g = x_j^a \ \forall j \in AR$), and a new "batch" is then started. The algorithm is described in detail in the next section.

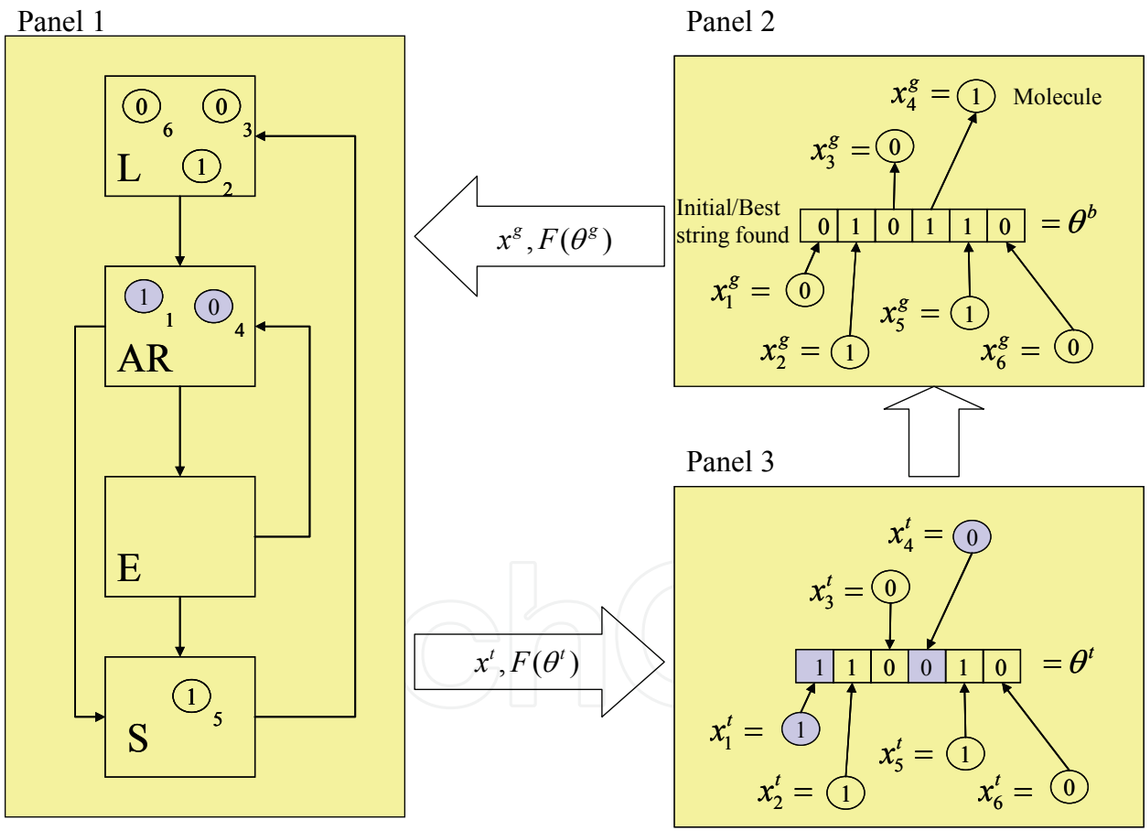


Fig. 2. The artificial chemical process in panel 1 changes the state of the molecules in panel 2 to a new trial state in panel 3 by transferring some molecules from L to AR and changing the states of the molecules in AR.

2.2 LARES algorithm

Initialization: The algorithm starts by initializing x^g randomly and placing all molecule variables in L.

Outer loop: Perturbation to form AR.

1. Select the number of molecules, N, to be extracted from L and added to the AR set ($N = rF_1$, where and r is a random number uniformly distributed in (0,1)).
2. Select N random molecules from L and add them to the AR set. For each selected molecule j, select a new state $x_j^a \neq x_j^g$ randomly.
3. Form a new trial vector using

$$x_j^t = \begin{cases} x_j^g & if\ x_j \notin AR \\ x_j^a & if\ x_j \in AR \end{cases} \quad (1)$$

4. If performance is improved, accept the trial state as the new best solution (ground state). $x^g \leftarrow x^t$. Send all AR molecules to the S set. Go to Step 1.
5. Set parameters: $RP = F(x^t)$, $|AR|_0 = |AR|$.
6. Inner loop: Iterative improvement of AR
   - 6.1 Select the number of molecules, M, to be extracted from AR to form E ($M = rF_2$, where F_2 is an algorithm parameter, and *r* is a random number.)
   - 6.2 Select M random molecules from AR and transfer them to the E set. Return the state of all molecules *j* in E to the state of the best solution found, $x_j^t = x_j^g$, and build the trial vector as in Step 3 using Eq. (1).
   - 6.3 If the performance is improved, $x^g \leftarrow x^t$, and go to step 1.
   - 6.4 *Improvement criterion for AR:*
     - 6.4.1 If $F(x^t) \leq RP$, add all molecules in E to S and update $RP = F(x^t)$.
     - 6.4.2 If $F(x^t) > RP$, generate a new activated state for all elements in E ($x_j = x_j^a \neq x_j^g, \forall j \in E$) and transfer all molecules in E to AR ($E = \varnothing$)
   - 6.5 Exit the inner loop if AR is too small or if the ratio of the number of iterations relative to the initial size of AR exceeds a given parameter, RRT. Otherwise go to Step 6.1.
7. If the size of L is less than a parameter LT, transfer all S molecules to L.

In Step 1, $F_1 = V \times c_0$; in Step 6.1 $F_2 = |AR|_0 \times c_i$. In both cases, if the number of molecules selected is larger than the set, all molecules in the set are selected. The parameters used for this algorithm are: RRT = 1.0, c_o = 0.3, c_i = 0.25, LT = V/2.

The algorithm was shown to be fast and robust when tested with problems of different degrees of multi-modality, discontinuity and flatness. The molecular representation allows the solution of a large class of problems. This structure is general in purpose but also has the flexibility to add problem-specific features. For example, the "locality" of these operators allows the inclusion of bias in the sub-set formation (Steps 3 and 11) or the transformation rule (Step 5).

2.3 Algorithm performance

This algorithm has been tested and extensively utilized to solve many optimization problems. Its performance in some of the test problems is reviewed in this section. The multi-modal random problem generator of Spears (Spears, 1998) was utilized to test LARES over various degrees of modality for binary representation. The problem generator

generates a set of P random V-bit strings representing the location of the P peaks in space. To evaluate the performance of an arbitrary string, the nearest peak is located (in Hamming space). Then the fitness of the bit string c is calculated as the number of bits the string has in common with that nearest peak, divided by V. The optimum fitness for an individual is 1.0.

$$f(c) = \frac{1}{V} \max_{i=1}^{P} \{V - Ham \min g(c, Peak_i)\} \tag{2}$$

The objective function used in LARES was $F(c) = 1 - f(c)$, while $-F(c)$ was used for the fitness function in GA simulations.

Table 1 shows the results for four study cases. For each set of parameters V and P, 20 random problems were generated in each case. Each algorithm was run on each problem generated. LARES found the global maximum in all cases ($f(c^*) = 0$), while GA failed to find the global maximum for cases 3 and 4, and μGA failed to find the global maximum in three out of four cases. For the first case, LARES converged to a global optimum in 78 function evaluations on average, while GA converged in 900 function evaluations and μGA converged in less than 350 function evaluations. For the second case, LARES found the global maximum in 647 evaluations on average, while GA converged in approximately 3,700 evaluations and μGA failed to find the global maximum in 20,000 function evaluations (see Figure 3). For the third and fourth study cases, LARES was the only algorithm that converged to the global maximum in nearly 30,000 function evaluations. This behavior was explored systematically by De Jong et al. (1997). In their analysis, the authors found that for V = 20, the simple GA will converge in less than 5,000 function evaluations. For V = 100, many trials failed to find the global optimum after 20,000 evaluations.

The algorithm has also being tested with Boolean satisfiability problems (SAT), which refers to the task of finding a truth assignment that makes a Boolean expression true. The Boolean satisfiability problem generator of Mitchel et al. (1992) was used to test the performance of LARES in solving random problems with different levels of epistasis. The model assumes a conjunctive normal form of the Boolean expression with C clauses. All clauses are also assumed to consist of the same number of literals, L. The vector of variables V is represented as a binary string.

A random problem is generated to create C random clauses. Each clause is generated by randomly selecting L variables, and then each variable is negated with probability 0.5. Once a random L-SAT problem is defined, the fitness function, f, is given by the fraction of clauses that are satisfied by the assignment. Note that the main goal of this section is to study LARES with different levels of epistasis. For practical solution of this type of problem, methods such as GSAT (Selman and Kautz, 1993) and WSAT (Gottlieb et al., 2002) have been specially developed.

V	P	Number of Iterations	GA	μGA	LARES
20	20	20,000	0	0	0
100	20	20,000	0	0.03	0
1000	20	30,000	0.16	0.29	0
1000	200	30,000	0.16	0.29	0

Table 1. Comparison of LARES performance with GA for the multi-modal random problem generator.

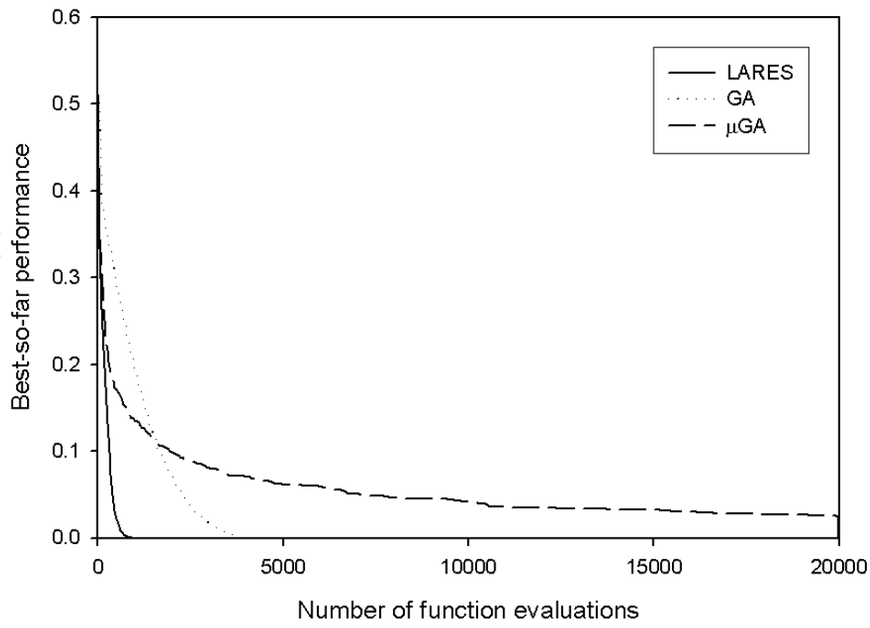


Fig. 3. Average best-so-far curves for LARES, GA and μGA using a multi-modal problem generator with V = 100 and P = 20.

Table 2 shows the solution of a series of L-SAT random problems using LARES and GA. Each test consists of an average of over 20 randomly generated problems. In all simulations, the length of clauses, L, had a fixed value of 3. The number of variables, *V*, was also fixed at a value of 100. The number of clauses was used as a parameter in the simulation, ranging from 200 to 2400. LARES was faster than GA in all cases, but in the last two cases GA found a slightly better solution while μGA found a slightly worse solution to the L-SAT problem (see Figure 9). These results indicate that LARES also behaves very well with problems involving different levels of epistasis.

The LARES algorithm was also applied to a very challenging test bed used by many authors to test real function optimization algorithms. Binary encoding was used to represent real variables. The algorithm was compared with GA using the same test bed, starting with the same initial guesses, and performing the same number of iterations. Comparisons are also made with other methods specifically designed for real-function optimization reported in the literature. Although literature in this field is extensive, few studies involve methods that are efficient for real function optimization. Reported algorithms include Differential Evolution (DE) (Storn and Price, 1997), the Breeder Genetic Algorithm (BGA) (12 Mühlenbein and Schlierkamp-Voosen, 1993), Evolutionary Algorithm with Soft Genetic operators (EASY) (Voigt, 1995), the Line-up Competition Algorithm (LCA) (Yan and Ma, 2001), Continuous Ant Colony Optimization (CACO) (Mathur et al., 2000), Adaptive Simulated Annealing (ASA) (Ingber and Rosen, 1992), Very Fast Simulated Annealing (VFSA) (Ingber, 1993), Guided Evolutionary Simulated Annealing (GESA) (Yip and Pao, 1995) and Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen, Muller and Koumoutsakos, 2003). In summary, LARES had better performance than GA in most instances, and in many cases the speed of LARES is comparable to that of methods specially designed to operate with real-value optimization problems.

Number of classes, C	Number of iterations	GA	μGA	LARES
200	30,000	0	0	0.0003
1200	30,000	0.0433	0.050	0.0469
2400	30,000	0.0651	0.071	0.0675

Table 2. Comparison of LARES and GA performance with the LSAT random problem generator

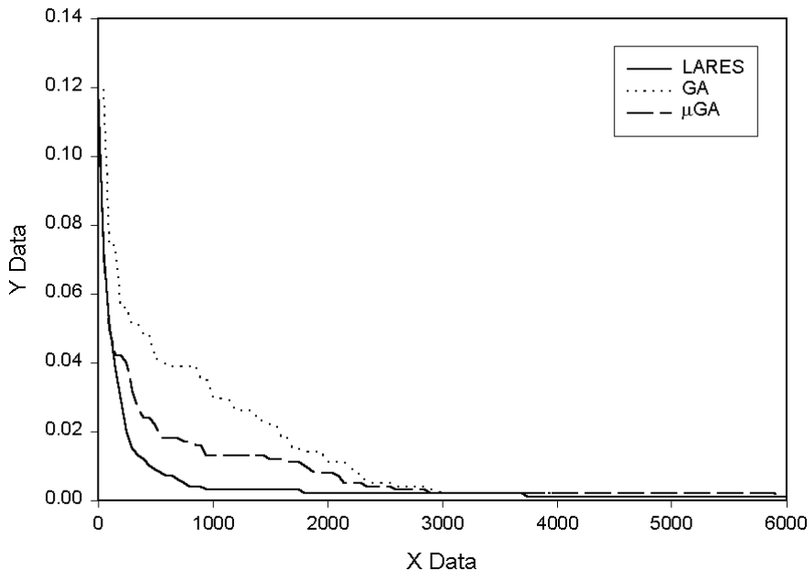


Fig. 4. Average best-so-far curves for LARES, GA, and μGA using a L-SAT problem with C = 200, L = 3, and V = 100.

3. Solution of dynamic optimization problems using LARES

As discussed in the introduction, dynamic optimization is a very important type of optimization problem in operation research and engineering, since many systems of interest are dynamic. In particular, most processes in the chemical industry are batch processes in which optimal reactant addition (and/or temperature) profiles determine product quality. Dynamic optimization is also used in model predictive control systems. These types of problems are more effectively solved using stochastic optimization methods that can escape from local minima and are not affected by singularities. In Banga, Irizarry-Rivera, and Seider (1998), an efficient and robust algorithm was developed to solve dynamic optimization problems using a flexible parametrization of the control law, consisting of a piecewise variable-length linear function. This method resulted in a big improvement over the more traditional piecewise constant approximations (Roubos et al., 1999; Luus, 2000). A generalized algorithm that uses LARES with a very flexible control law representation has also been considered (Irizarry, 2005a). This algorithm is reviewed in this section.

Unlike standard optimization to determine the optimal value of a set of real variables, in dynamic optimization, we seek an optimal function $u(t)$ or a set of functions $u_i(t)$, i= 1,M. The dynamic optimization problem for a single control law can be formulated as:

Find u(t) over $t \in [t_o, t_f]$ such that

$$\min_{u(t)} F(x(t_f)) \tag{3a}$$

subject to:

$$\frac{dx}{dt} = \Psi(x(t), u(t), t) \tag{3b}$$

$$x(t_o) = x_o \tag{3c}$$

$$h(x(t), u(t)) = 0 \tag{3d}$$

$$c(x(t), u(t)) \leq 0 \tag{3e}$$

$$u^L \leq u(t) \leq u^U \tag{3f}$$

where F is the performance index, u is the control law, and x the vector of state variables. The set of constraints consists of the dynamic model (Eqn. 3b), the initial conditions of the state variables (Eqn. 3c), the equality constraints (Eqn. 3d), the inequality constraints (Eqn. 3e), and bounds on the control variables (Eqn. 3f). Different methods to solve this type of problem have been reviewed recently by Banga et al. (2003).

3.1 LARES-PR algorithm

The previously introduced algorithm (Irizarry, 2005; Irizarry, 2006) consists of interfacing the LARES algorithm with a generalized representation of the control law. This procedure decodes the LARES decision variables (molecules) into a flexible representation of the control law based on three key elements: (a) variable-length segments, (b) the use of finite element trial functions to represent the control function in each segment (Zienkiewics, 1977), and (c) switching between different representations to model each segment with different functions. Figure 5 shows an example in which the possible profile is represented by three segments of different lengths. In the first segment, the control function is modeled with a quadratic finite element. The second segment is modeled with a constant function (step function). The third segment is modeled with a linear finite element. In this representation, the segment sizes, the type of function representing each segment, and the adjustable parameters of the selected function for each segment are all decision variables of the optimization problem to be solved. This representation spans a large functional space in which smooth regions, drastic changes in functionality, singularities, and discontinuities of the control function can be found as part of the solution for the optimization problem with a reduced number of decision variables.

The unknown control profile is encoded according to the following procedure. For the segments represented with *finite elements,* the node values of the finite element are part of the decision variables. Molecules are assigned to encode each of these variables using binary encoding. The function selection for each segment is then performed as follows. The data structure starts with all segments represented by finite-element function using the highest order of elements to be considered in the analysis. Then, a logical variable is defined for each segment, which is used as a switching mechanism. The logical variable can replace the

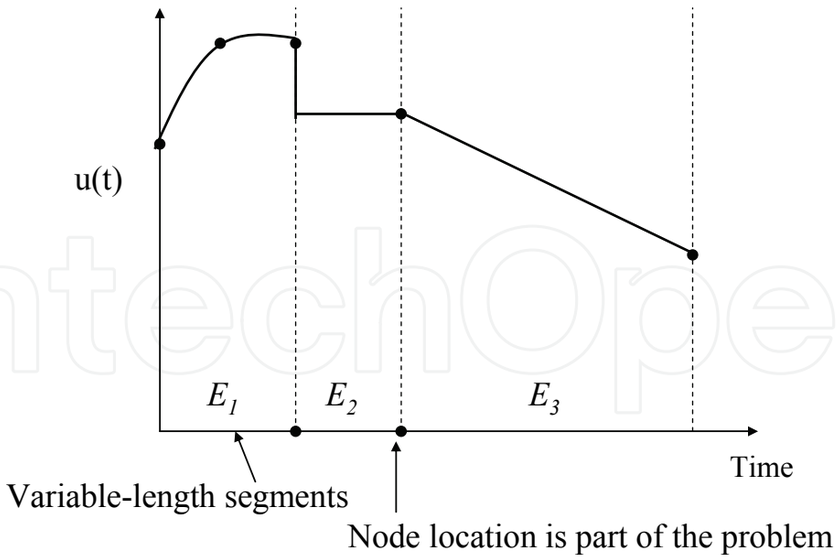


Fig. 5. Profile representation: generalized structure.

element with a lower-order element or with user-defined functions over the same segment interval. Figure 6 illustrates the hybrid formulation. This example consists of quadratic finite elements with three nodes representing each element. The logical variable with state $\delta = (1,1,2,3,0)$ replaces the first two elements with lower-order linear elements (eliminating the middle node as a variable), the third and fourth elements are replaced with user-specified functions, and element 5 is represented with a quadratic element. The combinatorial variables for each finite element, δ_i, is an integer number whose range equals the number of possible functions to be used. One molecule is selected for each combinatorial variable. The number of states for the molecule equals the number of possible functions that can represent a segment.

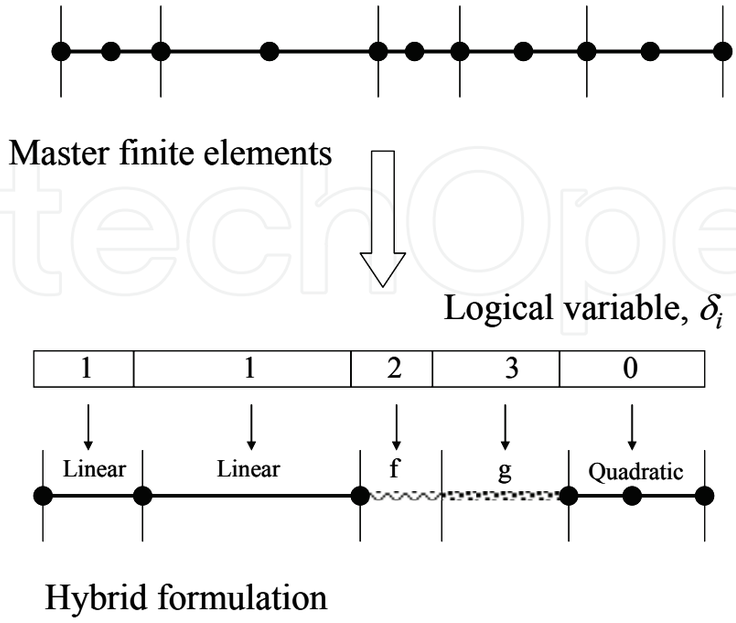


Fig. 6. Encoding a hybrid formulation.

The encoding of the segment size is the most difficult aspect of this formulation. The size of each segment is a component of the decision variables of the optimization problem, represented by a partition $\tau_1....\tau_N$. For each trial solution, a new partition is generated (τ_i^t , where the sequence is in increasing order, $\tau_i^t \le \tau_{i+1}^t$). These variables cannot be encoded directly into LARES, and if standard binary encoding of these parameters in the range $\tau_i \in [0, t_f]$ is utilized, this constraint will be violated frequently during LARES iterations. This problem is avoided by the following two-step procedure, called *Moving Partition (MP) transformation* (Irizarry, 2005a). First, computational variables are chosen for each partition variable ($s_i \in [0,1]$ to represent each τ_i). Mapping from this computational domain to the physical domain is made with the help of the disjoint segments, each one around the partition of the best solution found. Let τ_i^b be the sequence for the best solution found. Then the following boundaries are calculated:

$$\tau_i^{L,b} \equiv \tau_i^b - \beta \cdot \left(\tau_i^b - \tau_{i-1}^b \right) / 2 \tag{4}$$

$$\tau_i^{U,b} \equiv \tau_i^b + \beta \cdot \left(\tau_{i+1}^b - \tau_i^b \right) / 2 \tag{5}$$

where the parameter β is used to control the gap between the disjoint segments. With this boundary for each partition node and the trial vector, s_i^t , the actual trial partition is calculated from the following MP mapping, $T_i : [0,1] \to [\tau_i^{L,b}, \tau_i^{U,b}]$, defined as follows:

$$\tau_i^t = \tau_i^{L,b} + s_i^t \cdot \left(\tau_i^{U,b} - \tau_i^{L,b} \right) \tag{6}$$

The MP is shown schematically in Figure 7. As shown in this figure, the trial variables $s_1,...s_N$ are not in ascending order ($s_2 < s_1$), but the trial partition values, $\tau_1,...\tau_N$ are in ascending order ($\tau_1 < \tau_2$).

With this description of the control law, the LARES-PR algorithm can be described as follows:

LARES-PR algorithm. The overall algorithm is discussed in Irizarry (2005). It consists of interfacing this profile representation with LARES. After a new trial molecular state from LARES, the procedure described in this section consists of (1) decoding, (2) applying MP transformation, (3) building the control law determined by element type, size, and parameters, (4) integrating the model, and (5) feedback to LARES regarding the performance of the control

3.2 Simulation results

LARES-PR performance has been studied with a set of benchmark problems with low sensitivity of the objective function, bang-bang behaviors, singular arc, and discontinuities in the optimal profile. In all cases, the algorithm has proven to be efficient and robust. Figure 8 shows the solution of four optimization problems used by several authors as benchmark problems. The Van der Pol oscillator problem has been studied by Vassiliadis (1993), Tanartkit and Biegler (1995), Banga, Irizarry-Rivera and Seider (1998), and Vassiliadis et al. (1999). This problem was solved using the generalized control function, where each element can be represented by either linear or quadratic Lagrange polynomials. The optimal profile is shown in Figure 8a, with an improved performance index over other methods using only four elements and a smoother profile compared to previous results reported in the literature.

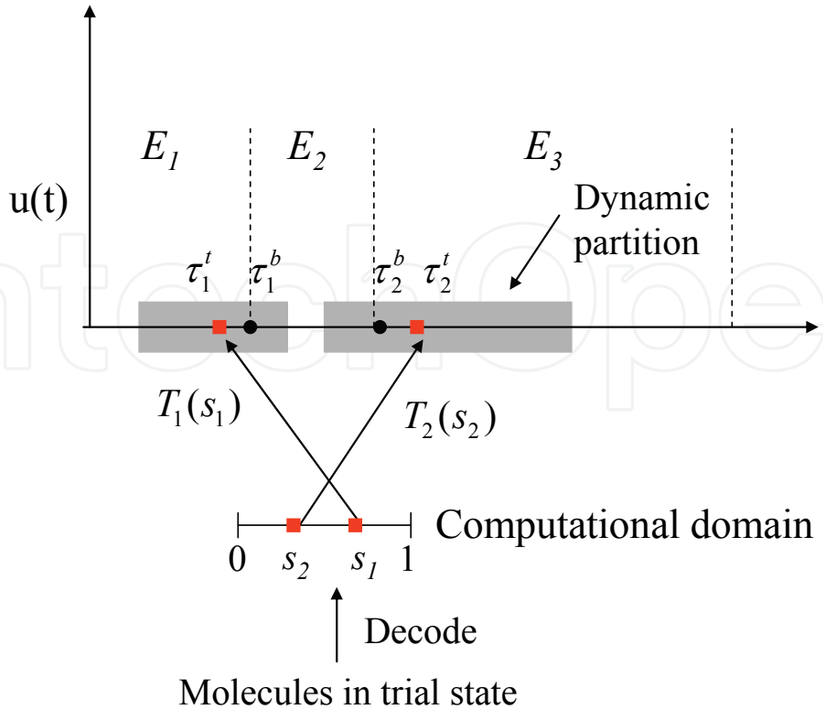


Fig. 7. Moving partition transformation. Each variable in a computational domain is mapped into a corresponding space/time subdomain.

The second case shown in Figure 8b is a plug-flow reactor with singular arc. In this problem, a plug-flow reactor packed with a mixture of catalysts is used to perform the reaction $A \Leftrightarrow B \Leftrightarrow C$. The fraction of catalyst is adjusted throughout the reactor to maximize the product C. This problem was solved using the generalized control law approximation with three possible functions for the element E_i: δ_i = {1, 2, 3} = {u(t) = constant finite element, u(t) = u_{max}, u(t) = u_{min}}. The optimal control law is shown in Figure 8b. Figure 8c shows the optimal production of secreted protein in a fed-batch reactor. This problem consists of a bioreactor operated in fed-batch mode studied by Park and Ramirez (1988), Luus (1992), Banga, Irizarry-Rivera and Seider (1988), Vassiliadis et al. (1999), and Sarkar and Modak (2003). This problem shows very low performance index sensitivity of the control profile, often leading to computational difficulties particularly when gradient-based algorithms are used. The fed-batch reactor problem was also solved using the generalized control law approximation, with three possible functions for the element E_i: δ_i = {1, 2, 3} = {u(t) = quadratic finite element, u(t) = u_{max}, u(t) = u_{min}}with 10 elements. The control law gives a global optimum for this problem.

Figure 8d shows the optimal profile for the bang-bang control problem (see Irizarry 2005a for a detailed description of the problem). To solve this problem, eight elements were used, and the function approximation of each element is either a linear interpolation function or the bang-bang constant functions δ_i={1, 2, 3}={u(t)=linear trial function, u(t)= u_{max}, u(t)= u_{min}}. LARES-PR found the correct bang-bang feature as part of the solution, that is, δ^*= {4,3,4,3,4,3,4}. This is an important element of the proposed method, which can be used with general-purpose approximation functions or with problem-specific functionalities for which the proposed algorithm will identify problem features in addition to the solution.

In most cases, the near optimum value (less than 0.5% of global optimum) was found in less than 1,000–10,000 function evaluations. The algorithm continues to refine the solution at a slower rate, resulting in very accurate solutions. In most cases, a very accurate solution can be found in 50,000–100,000 iterations. The results demonstrate that LARES-PR is robust and has fast convergence properties when compared with other stochastic optimization methods.

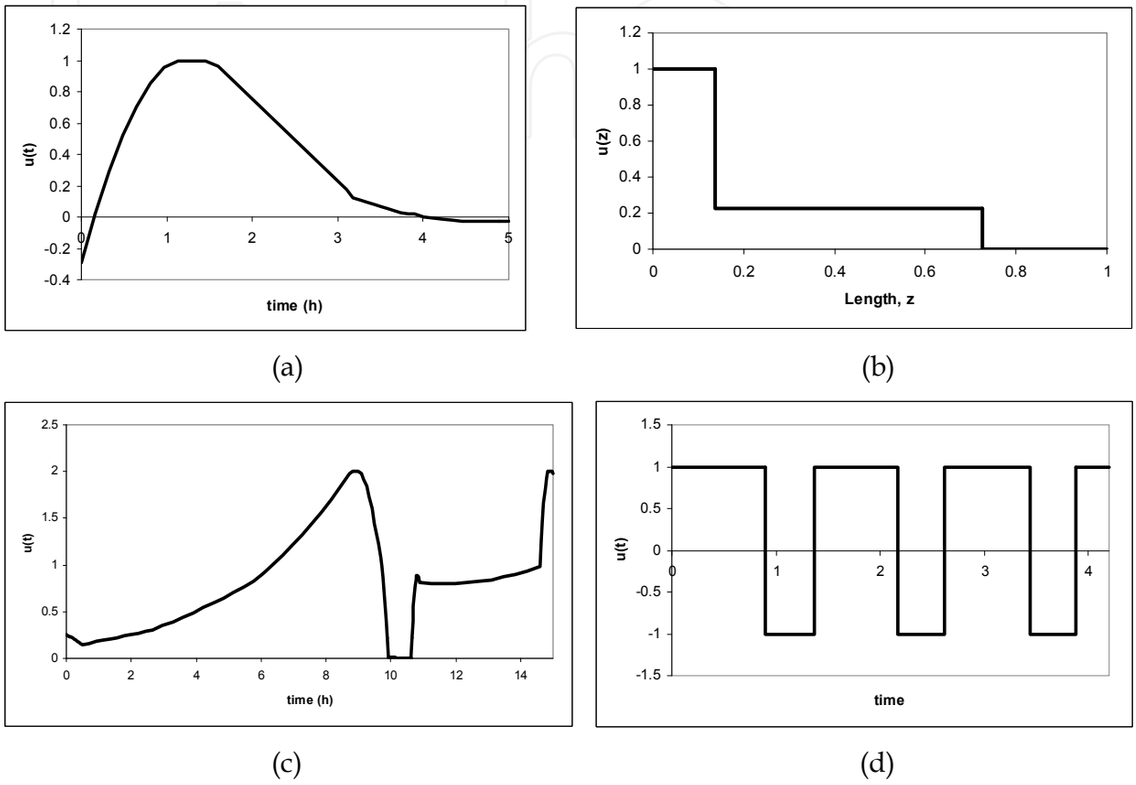


Fig. 8. Optimal profiles: (a) Van der Pol oscillator problem, (b) plug-flow reactor with singular arc, (c) fed-batch bioreactor, (d) bang-bang control problem.

LARES-PR has also being applied to large-scale optimal control problems with discrete-time dynamics and multiple control laws. The dynamic integrated climate-economy (DICE) model for global warming (Nordhaus, 1994) is a model of a very important problem, which posed several challenges in finding the optimal profile. This model consists of maximizing the discounted sum of per capita utilities consumption subject to the dynamics of emissions, economic impact, and economic cost of policies to control global warming. Moles, Banga and Keller (2004) made an extensive study of optimal policy with a modified version of this model using different global optimization algorithms (ICRS, LJ, DE, SRES, GLOBAL, GCSOLVE). As discussed in Moles, Banga and Keller (2004), the numerical solution of this multimodal NLP is very challenging, due to the non-convexities and discontinuous nature of the dynamics.

As the time horizon is discrete, the dynamic optimization problem can be formulated as a standard NLP problem with the value of the control laws at each discrete time as a decision variable. Using this approach, the number of decision variables increases as the time horizon, N_t, increases (number of decision variables = N_t*N_u), resulting in a large-scale

nonlinear optimization problem. Alternatively, the profile representation of LARES-PR can be utilized to represent the control law with a very small set of decision variables.
Figure 9 shows the performance of LARES, DE, CRS, and LARES-PR in solving the original DICE model for a time horizon of 50 decades. As shown in Figure 9, LARES and LARES-PR are faster than DE and CRS in converging to a near global optimum. In particular, LARES-PR was much faster than all methods with a high-quality solution: The best value found for each algorithm was: $W^* = 966.91767$ (DE), 966.91632 (LARES-PR), 966.91353 (LARES), and 966.69733 (ICRS). When the number of finite elements was increased from five to eight, the solution was improved to almost identical to the DE results in fewer iterations, with $W^* = 966.91711$ (LARES-PR).
LARES-PR effectively solved this problem, which consisted of finding the optimal functionality of two simultaneous control laws. To implement multiple control laws, a representation is defined for each unknown profile (PR_1 and PR_2).

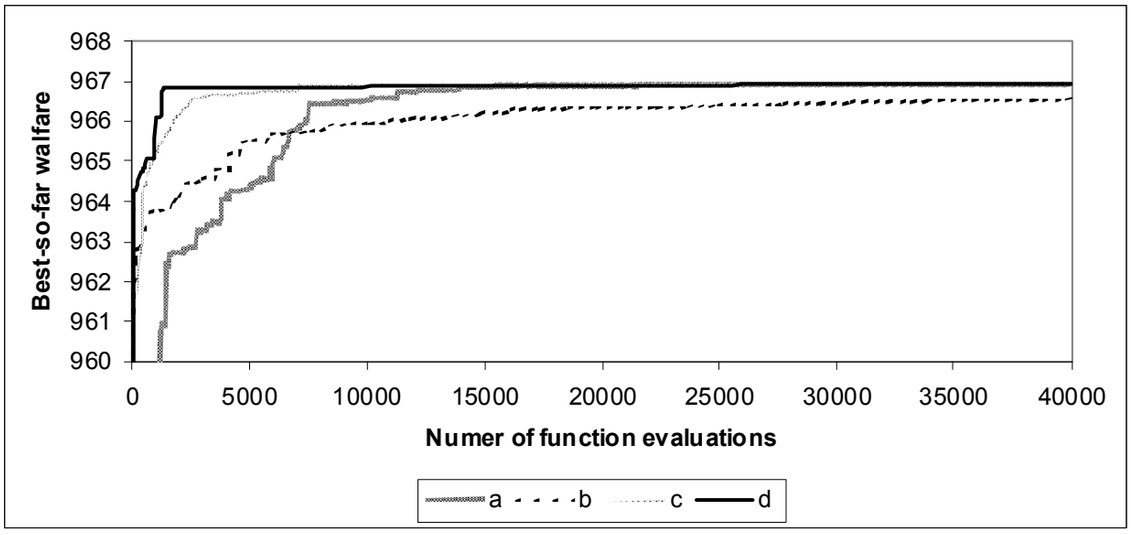


Fig. 9. Performance for the DICE climate-economy discrete time model: (a) DE, (b) CRS, (c) LARES, (d) LARES-PR.

4. Monte Carlo methods for population balance problems

A great majority of products are composed of finely divided solids or contain finely divided particles as part of their composition. One example is metallic microparticles and nanoparticles used in electronic ink compositions. Another example is the dispersion of pigments in paints. Most pharmaceutical products include a crystallization of organic powders. The particle size distribution, shape, and composition of these finely divided particles control the properties of the final product. Therefore, the understanding of these particles and how they form is of great importance (Irizarry, 2010a). The macroscopic modeling of particle formation consists of formulating population balance equations for the problem at hand. When optimization of these systems is pursued, the population balance equations appear as part of the constraints (Irizarry, 2005; Irizarry, 2006).

Population balance models are continuity equations of a particle population evolving by different mechanisms (such as aggregation, breakage, nucleation, and growth). The continuous population balance equation, PBE, is a deterministic integro-differential equation that describes the dynamics of a particle density function as a function of continuous particle properties (i.e. volume, particle radius, surface area, etc.). As the dimensionality of the PBE increases, the direct numerical solution of these equations becomes more difficult. For a multidimensional population balance equation, the Monte Carlo (MC) solution is an attractive alternative (and in many cases the only option). In these methods, the system evolution is modeled by a simple stochastic game, which is robust and easy to implement (Gillespie, 1975; Garcia et al., 1987). For systems close to the thermodynamic limit, both the MC solution and the direct numerical solution of the PBE converge to the same results. In many situations of practical interest, the MC solution may become very slow. Several optimization approaches have been developed to increase the MC simulation speed. The point ensemble Monte Carlo (PEMC) algorithm and the τ–PEMC algorithm developed in 2007 (Irizarry, 2007a; Irizarry, 2007b) are approximated MC methods that increase simulation speed by orders of magnitude when compared with existing MC methods.

Population balance models can be formulated as discrete events Markov processes (also known as a jump Markov process). The standard exact simulation method (exact MC) for jump Markov processes consists of selecting the time for the next event and the type of event sequentially until a final time is reached (one trajectory). Many trajectories are calculated to generate the probability distribution function of the Markov process variables. This simulation method uses the propensity functions of each event, E_s , defined as follows:

$R(E_s)\,\mathrm{d}t \equiv$ the probability that the event E_s occurs in the time interval (t, t + dt).

The time for the next event, τ, is sampled from an exponential distribution:

$$P(\tau) = R_\Sigma \exp[R_\Sigma \tau]\,. \tag{7}$$

where R_Σ is the total propensity of all possible events ($R_\Sigma = \sum_i R(E_i)$). The probability of the event, E_i , occurring next (i.e. that particles will aggregate) is proportional to the event propensity, $R(E_i)$:

$$P(E_i) = R(E_i) / R_\Sigma\,. \tag{8}$$

In the inverse method, this distribution is sampled using a uniform random variable $r \in (0,1)$ and then solving the following equation:

$$\sum_{i=1}^{f} R(E_i) < r\, R_\Sigma \le \sum_{i=1}^{f+1} R(E_i) \tag{9}$$

where $E_i, i = 1,T$ is the indexed list of all possible events, and T is the total number of events. The solution of this equation, E_f , is the next event to be executed at time $t+\tau$. Alternatively, the acceptance-rejection method can be used to sample the next event [5].

This simulation procedure has been used to develop the stochastic simulation algorithm, SSA, for chemical kinetics (Gillespie, 1976; Gillespie, 1977). In this method the firing of a chemical reaction represents a discrete chemical kinetic event of the Markov process. This algorithm is also known as kinetic Monte Carlo (KMC). The exact MC has also been used for the MC solution of population balance models. In many situations of practical interest, the MC solution may become very slow, especially when the number of particles in the simulation box is increased and the total number of events becomes very large or when the computational cost of calculating all the rates, R_Σ is large. In these cases the calculation of Eqs. (1) and (3) becomes very computationally expensive, slowing down the generation of trajectories.

To further accelerate the MC simulation of population balance models, a new approach was introduced in 2007 (Irizarry, 2007a; Irizarry, 2007b). These algorithms are based on the construction of a jump Markov process called PERP, which approximates the actual jump Markov model. These algorithms are shown to reduce CPU time by orders of magnitude without sacrificing simulation accuracy, when compared with optimized exact MC methods. Unlike other coarse graining (or lumping) strategies in which information and identity is lost, in these algorithms, the history of each particle is retained while a coarse view of the process is taken. These two algorithms are summarized in the next section.

5. Fast Monte Carlo algorithms

The PEMC and τ-PEMC algorithms are based on the simulation of an approximated jump Markov process called PERP (Irizarry, 2007a; Irizarry, 2007b). This approximated Markov process is based on three ideas. First, the total population is "discretized" into subpopulations of particles with sizes of specified intervals. Each subpopulation is viewed as a "chemical species" with the number of particles in the subpopulation representing the number of molecules of that species in the simulation volume. Second, the inter-particle interactions (i.e. aggregation, nucleation, breakage) are viewed as a set of special types of reaction, in which the reaction products are allocated stochastically to the existing species using probability functions that are mass conserving on average. Third, the original set of subpopulations is coupled with the system of "chemical species". The PERP Markov process is described next.

5.1 PERP Markov process and PEMC algorithm

The first step in this approximated Markov process is the partition of the particles in the simulation volume into a set of sub-ensembles, Φ_i , called point ensembles. This partition is made using a set of M grid points of representative sizes $v_1,\ldots v_M$. All simulation particles in an interval around the grid point v_i are allocated in point ensemble Φ_i. Let N_i be the number of particles in point ensemble Φ_i. The state vector is then defined as $(\boldsymbol{N},\boldsymbol{\Phi})$ were $\boldsymbol{N}=(N_1,N_2,...,N_M)^T$ and $\Phi=(\Phi_1,\Phi_2,...,\Phi_M)^T$. Here, each grid point is viewed as a chemical pseudo-specie, S_i, with N_i molecules in the simulation volume. For example, Figure 10 shows a size-dependent partition of 17 particles (each with a set of different properties) into five point ensembles. The five pseudo-species $(S_1,S_2,...,S_5)$ defined by this partition have (3, 4, 6, 3, 1) molecules in the simulation volume.

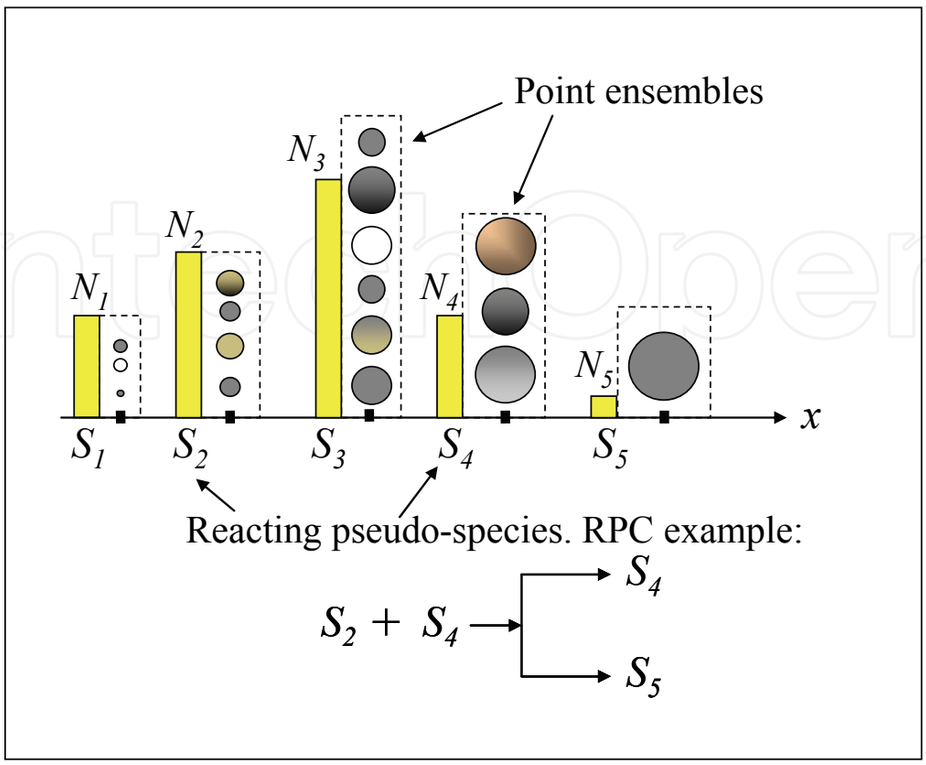


Fig. 10. Schematic of PERP jump Markov model

In this approximated process, the discrete events consist of a set of "reaction channels" where the reaction part involves the pseudo-species to mimic the actual particle event (i.e. aggregation between two particles). Unlike standard reaction channels of chemical kinetics, the product component consists of several steps, some of them also stochastic processes.

In the exact Markov process, an event, *E*, is defined in terms of the possible interactions between the simulation particles. For example, an aggregation between two particles i and j is an event that creates a new particle in the simulation volume ($E: x_{new} = x_i + x_j$) and eliminates the mother particles from the simulation volume. The reactant component of the RPC mimics the same event, but between the pseudo-species instead of actual particles. For example, the events representing the aggregation mechanism in the original Markov process are replaced by an "aggregation reaction" of pseudo-species ($E_s : S_i + S_j$) in the PERP Markov process. In the case of aggregation, the propensity function for these "reaction channels" is given by:

$$R(E_s) = \left(1 - \frac{1}{2}\delta_{ij}\right) N_i N_j q(v_i, v_j) / V_s \tag{10}$$

The propensity functions for other mechanisms have also been described (Irizarry, 2007a). Notice that the propensity of each event in the PERP Markov process is given only in terms of ***N***. For the aggregation case discussed here, the PERP event is summarized in the following algorithm:

Algorithm M1: Fire an aggregation RPC event: $E_f = S_i + S_j$ with $(v_i + v_j) \in [v_k, v_{k+1}]$.

1. Reduce N_i and N_j by one.
2. Select random particles, *n and m*, from point ensembles *i* and *j* ($x_n \in \Phi_i$ and $x_m \in \Phi_j$).
3. Form a new particle from the mother particles, $x_{new} = x_n + x_m$.
4. Eliminate the particles *n* and *m* from their ensembles.
5. Find the product sub-specie S_p : *p* = *k* with probably P_f or *p* = *k* + *1* otherwise.
6. Allocate the new particle to the product point ensemble $x_{new} \in \Phi_p$, increase N_p by one.

The product pseudo-specie S_{new} in the current event is determined by letting the landing interval $[v_k, v_{k+1}]$ be defined as the interval such that $v_{new} = v_i + v_j \in [v_k, v_{k+1}]$. The product pseudo-specie is S_k ($S_{new} = S_k$) with probability P_s. In this case, the number of molecules N_k is increased by one, and x_{new} is allocated to Φ_k. Otherwise $S_{new} = S_{k+1}$, N_{k+1} is increased by one, and x_{new} is allocated to Φ_{k+1}. The product probability parameter, P_s, is calculated using the mass conservation equation for this landing interval: $v_k P_s + v_{k+1}(1 - P_s) = v_{new}$. All these events in the product component of the RPC are simply called PERP events. RPCs can be defined for any mechanism (Irizarry, 2007a)
The PEMC algorithm is the exact MC simulation of the approximated PERP Markov process. A detailed description of these steps has been published (Irizarry, 2007b). Let $E_i, i = 1,..,T$ be the list of all possible RPCs describing the population balance model at hand. The PEMC method is summarized in the following algorithm:

Algorithm M2: PEMC (one iteration).

1. Find the time to fire the next RPC using Eq. (7). Update the time $t = t + \tau$.
2. Find the RPC to be fired next solving Eq. (9).
3. Fire the selected PERP event (Algorithm M1)

These steps are repeated until the final time is reached. This algorithm is very fast because the set of RPCs is more compact than the set of all possible events between particles, making the calculations of Eq. (7) and (9) very fast.

5.2 τ-PEMC

The τ–PEMC algorithm is a τ-leap solution of the approximated PERP Markov process. The τ-leap method is an approximated stochastic simulation, were many events are fired at once over the time interval. Consider a coarser time interval, τ, such that many events occur in this interval, but small enough that the propensity functions will not change appreciably during τ. When this condition is satisfied, all reaction channels can be considered as independent events (Gillespie, 2001), and the number of firings for each reaction, E_j, is a Poisson random variable with distribution $P_{PD}\left(k_j; R(E_j), \tau\right)$, where

$$P_{PD}\left(k; a, \tau\right) = \frac{e^{-a\tau}}{k!}\left(a\tau\right)^k \tag{11}$$

The accuracy and speed of the method depends on the selection of time τ during the simulation (Gillespie and Petzold, 2003). Since the Poisson distribution is not bounded, it could generate negative values for the concentration.

Another improvement to the method is to replace the Poisson distribution with a binomial distribution (Tian and Burrage, 2004; Chatterjee, Vlachos, and Katsoulakis, 2005):

$$P_{BD}\left(k;p,k_{\max}\right)=\frac{k_{\max}}{k!\left(k_{\max}-k\right)!}p^{k}\left(1-p\right)^{k_{\max}-k} \tag{12}$$

In this case the number of firings for each reaction, k_j, is sampled from a binomial random variable with distribution $P_{PD}\left(k_j;p_j,k_{\max}^{j}\right)$, where $k_{\max}^{j}$ is the maximum number of times reaction j can be fired (after consuming the limiting component). The firing probability for E_j is $p_j = R(E_j)\tau / k_{\max}^{j}$. The binomial distribution eliminates the problem of negative concentrations and is more robust with respect to larger τ values.
The PERP process can be simulated using the τ-leap method, as previously described (Irizarry, 2007b). A pseudo-code for this algorithm is described as follows:

Algorithm M3: τ-PEMC (one iteration)

1. Select the time parameter, τ.
2. For each RPC, E_s, take a sample k_s from a binomial distribution (Eq. (12)) using the parameter $p = R(E_s)\tau / k_{\max}^{s}$.
3. Fire each PERP event, E_s, k_s times (execute algorithm M1 k_s consecutive times).

Continue steps 1-3 until the final time is reached.

5.3 Performance of the PEMC algorithm with complex kernels

The numerical accuracy of these algorithms has been studied with complex coagulation kernels of physical relevance. Numerical results are compared with the generalized approximation method (GA) developed by Piskunov and Golubev (2002) and Piskunov et al. (2002). The values for the second moment by the GA method are considered the most accurate existing numerical results in the literature. They are used here as benchmark values. The following kernels are considered:

i. The Brownian kernel,

$$q_B(u,v)=\left(u^{1/3}+v^{1/3}\right)\left(u^{-1/3}+v^{-1/3}\right) \tag{13}$$

ii. The coagulation kernel simulating the process of migration and coalescence of particles on a heated substrate,

$$q_+(u,v)=u^{2/3}+v^{2/3} \tag{14}$$

iii. The gravitational coagulation of particles in the Stokes regime,

$$q_C(u,v)=\left(u^{1/3}+v^{1/3}\right)^{2}\left|u^{2/3}-v^{2/3}\right| \tag{15}$$

For the gravitational kernel, some moments diverge after a critical point that depends on initial conditions. The critical point for the initial conditions used here is in the range of 0.5–0.8 (Piskunov et al., 2002).

The initial conditions used in the simulation for q_B and q_+ kernels are a mono-disperse solution with unit particle volume and unit total particle concentration. For the case of q_C, the following initial conditions are utilized: $n(v,0) = 0.5\delta(v-1) + 0.25\delta(v-2)$. In Table 3, the CPU time and final second moments of the simulations for different methods are compared as a function of kernel type and number of particles. For the τ-PEMC, two representative coarse-graining factors (f = 100 and f = 1,000) were examined. All calculations were performed using Visual Fortran 6.6 on an Intel Pentium M 1.6 GHz machine with 504 MB RAM. Table 3 shows that τ-PEMC can be 14–44 times faster than PEMC.

For kernel q_B, the τ-PEMC method gives accurate results in all cases with a coarse-graining factor f = 100. For the case f = 1,000, the number of particles had to be increased to 50,000 to generate accurate second moments (see Table 3). The q_+ kernel is more computationally challenging than the Brownian kernel, resulting in a wider distribution. Numerical results presented previously (Irizarry, 2007a) show that PEMC also quickly converges to the GA values. Table 3 shows the solution convergence to the exact value as a function of increasing the number of particles. For the factor f = 1000, large deviations of the second moments were observed even when a large number of particles was utilized. Good results were obtained for $f = 100$.

For the q_C kernel, some moments diverge at a critical value (gelling point). For the initial conditions used here, this critical value is believed to be between 0.5 and 0.8 (Piskunov et al., 2002). Unlike most numerical methods, the PEMC gives accurate results for all times, especially at time 0.6, where all discretization methods diverge (see Irizarry, 2007a). The τ-PEMC method can converge to the exact solution for times far from the critical point (time ≤ 0.4). When the time reaches the critical point, the errors for the second moment were quite large in all cases.

		PEMC		τ-PEMC f=10^2		τ-PEMC f=10^3		AR		GA
Kernel	N_p	*CPU*	$m_2(t_f)$	*CPU*	$m_2(t_f)$	*CPU*	$m_2(t_f)$	*CPU*	$m_2(t_f)$	$m_2(t_f)$
q_B	10,000	3.68	412	0.22	413	0.06	364	0.18	413	416
	20,000	7.24	415	0.45	414	0.12	381	0.35	420	
	50,000	18.12	416	1.11	415	0.51	400	0.89	414	
q_+	10,000	5.75	2.15E+5	0.52	1.90E+5	0.13	7.45E+4	331.1	2.10E+5	2.29E+5
	20,000	11.52	2.20E+5	1.01	2.03E+5	0.24	1.14E+5	662.7	2.24E+5	
	50,000	28.74	2.37E+5	2.49	2.29E+5	0.57	1.74E+5	1642.2	2.39E+5	
q_C	10,000	0.36	33.00	0.02	9.02	0.006	0.0	70.2	6E+3	23.27
	20,000	0.73	28.50	0.06	12.2	0.02	4.36	115.6	5E+3	
	50,000	1.76	23.30	0.18	15.7	0.05	7.16	290.0	5E+3	

Table 3. Comparison of second moments at final time t_f and *CPU* times between *τ-PEMC* and *PEMC* for kernels q_B, q_+, and q_C. Numerical results are compared with the generalized approximation method (*GA*) and the acceptance-rejection *MC* method (*AR*).

These results show that the τ-PEMC method generates accurate results if τ is selected to be small enough and the number of particles large enough. For the cases studied here,

N_p = 20,000 and $f = 100$, give very accurate results for all kernels (away from gelling points). For simulations near gelling points, the performance of the τ-PEMC deteriorates, and the PEMC should be used in this case.

Table 3 also shows results using the acceptance rejection (AR) method of Garcia et. al. (1987) for comparison. For the coagulation case, this method is very attractive because it is simple to implement and avoids calculation of all interactions between particles, making the method very computationally efficient. As noticed by other authors, the CPU time and accuracy of this method depends drastically on the problem (kernel and initial conditions). As shown in Table 1, for some cases the AR method can be very efficient (q_B) while in other cases the CPU time is very high (q_+). Additionally, the AR method diverges for the q_C kernel, also shown in Table 3.

6. Hybrid Monte Carlo algorithm for population balance models with stochastic and deterministic variables

Most MC implementations of population balance models have focused on the solution of the PBE to approximate macroscopic variables. Less attention has been focused on the solution of population balance models where some species are far from the thermodynamic limit (very dilute or finite) and other species can be considered deterministic (high concentration). In this case the MC is more accurate than direct numerical solution, which ignores the inherent fluctuations of the system. This type of problem often results in a stochastic system that contains both stochastic and deterministic variables with multiple timescales for the different mechanisms. In this case, the direct MC simulation will be accurate but very ineffective in terms of CPU time. Furthermore, most of the computational time is spent sampling the fast events. This type of situation has been considered in the case of chemical kinetics and biological systems for which efficient hybrid algorithms have been developed to solve multi-scale problems (Salis and Kaznessis, 2005; Kaznessis, 2006; Haseltine and Rawlings, 2002; Haseltine and Rawlings, 2005). In these algorithms, the fast processes are approximated by continuous models, and the slow processes are solved by the exact MC method in a hybrid algorithm.

Disparate scales in population balance models may arise because some species are concentrated while others are very dilute. For accurate simulation of the dilute species, a large number of simulation particles are needed. In this case, the exact MC methods become very slow. A recently introduced hybrid strategy (Irizarry, 2010b) is reviewed. In this strategy, the τ-PEMC is used for the parts of the system than can be considered large, and the PEMC is used for the stochastic events.

6.1 Hybrid algorithms in chemical kinetics

The hybrid strategy for chemical kinetic problems (Salis and Kaznessis, 2005; Kaznessis, 2006; Haseltine and Rawlings, 2002) is based on partitioning between fast and slow reactions. To split the system between slow and fast events, the following criteria for fast events are utilized:

1. The fast events occur many times in a small time interval.
2. The effect of these events on the number of particles and the propensity functions is small relative to the total propensity function and the total number of particles.

The slow processes are simulated using SSA, while the fast processes are integrated using the Langevin equations. To make this hybrid simulation self-consistent, the coupling

between both processes must be considered. If we look at the interval between the previous slow event at time t_o and the time for the next slow event ($t_o + \tau$), the following equation is satisfied:

$$\int_{t_0}^{t+\tau} R_{\Sigma}^{slow} dt + \ln(r) = 0 \tag{16}$$

where r is a random number from the uniform distribution (*0, 1*). This equation replaces Eq. (7) for the time of the next slow event. In the interval $[t_o, t_o + \tau]$, the dynamics of the fast system can be integrated in a seamless way since by definition no slow events are present in this interval. Thus, Eq. (16) becomes a constraint in the hybrid strategy that needs to be monitored while integrating the fast process.

As previously discussed (Salis and Kaznessis, 2005), one way to implement this constraint is to notice that the integral term is monotonically increases, and the second term is a negative term. Therefore, the time for the next slow event can by found by monitoring the zero crossing of the residual equation:

$$RES(t) \equiv \int_{t_0}^{t_0+t} R_{\Sigma}^{slow} dt + \ln(r) \tag{17}$$

6.2 Hybrid algorithm in multi-scale MC simulation of particulate processes

The hybrid strategy described in Section 6.1 is utilized with the slow mechanisms simulated using the PEMC algorithm. Instead of approximating the fast mechanism with a continuous model, as in the case of chemical kinetics, the τ-PEMC algorithm is utilized to model the fast mechanisms. The τ-PEMC method allows for coarse simulation in time while maintaining individual particle properties, in contrast to continuous models such as Langevin equations in which particle integrity is lost. The hybrid algorithm consists of the τ-leap integration of the PERP Markov process for fast events while monitoring the residual Eq. (17) for the firing of the next slow event of the PERP Markov process. This process is shown schematically in Figure 11. The detailed description of the algorithm is given in Irizarry (2010b).

6.3 Simulation results

Consider a system with two type of particles, A and B. B particles are much smaller than A particles. A particles can grow by an aggregation mechanism. B particles are stable from aggregation with other B particles but can condense on the surface of growing A particles. Furthermore, it is assumed that B particles are very dilute compared to A particles. These conditions make A-B condensation events a stochastic process, while A-A aggregation events can be approximated as continuous events.

Figure 12 shows five instances of the test problem for the case W = 100 (See Irizarry, 2010b for details). As shown in Figures 12a and 12b, the condensation of B monomers (measured with parameters x_1 defined in Irizarry, 2010b) is a stochastic process with considerable variability between trajectories, while the aggregation of A particles can be approximated as deterministic. In this case all A-particle size distribution trajectories of the second moment are almost identical. The PEMC is used as a benchmark for the accurate stochastic simulation. As there are disparate rates between condensation and aggregation, the hybrid

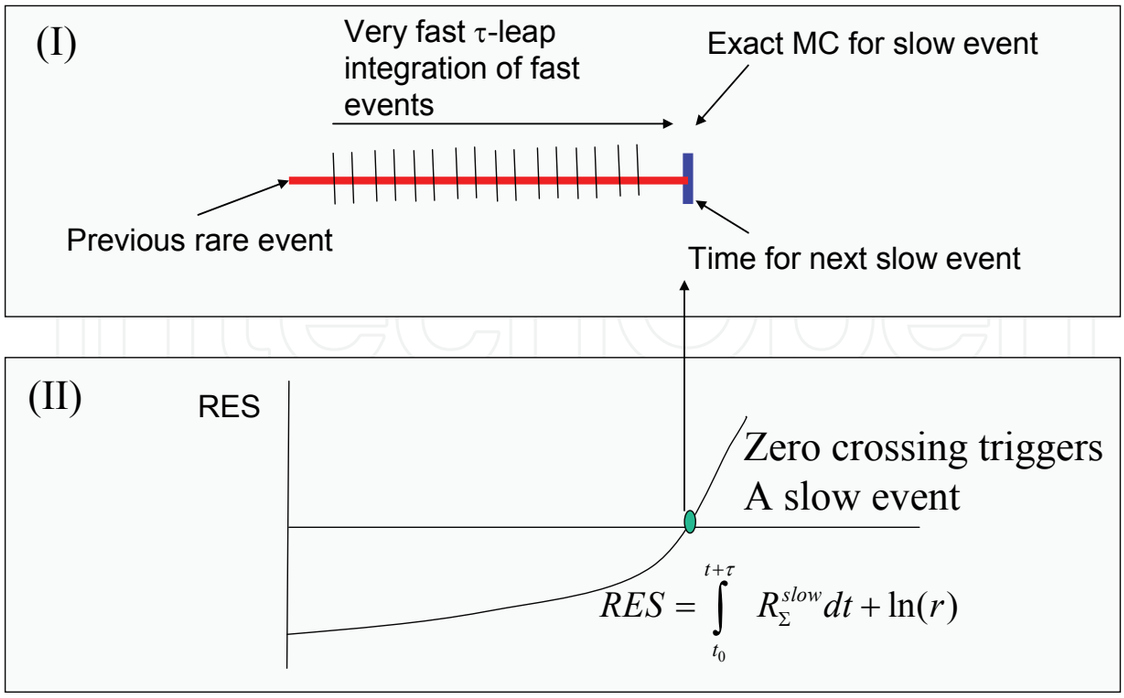
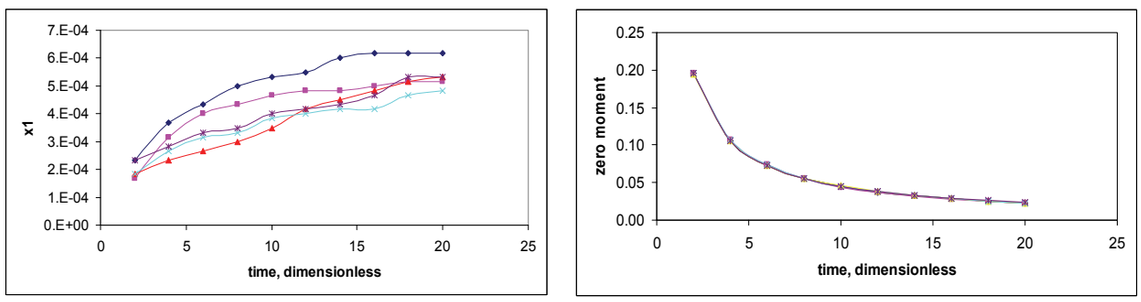


Fig. 11. Schematic of the hybrid algorithm. Integrate fast processes (panel I) while monitoring for the next slow event (panel II). At a zero crossing, a PEMC iteration is executed.



(a) (b)

Fig. 12. Five PEMC trajectories of the test problem with W = 100. (a) The fraction of B in A-B particles is stochastic A-B (parameter x_1). (b) Zero moment of the The size distribution of the main population (A) is deterministic.

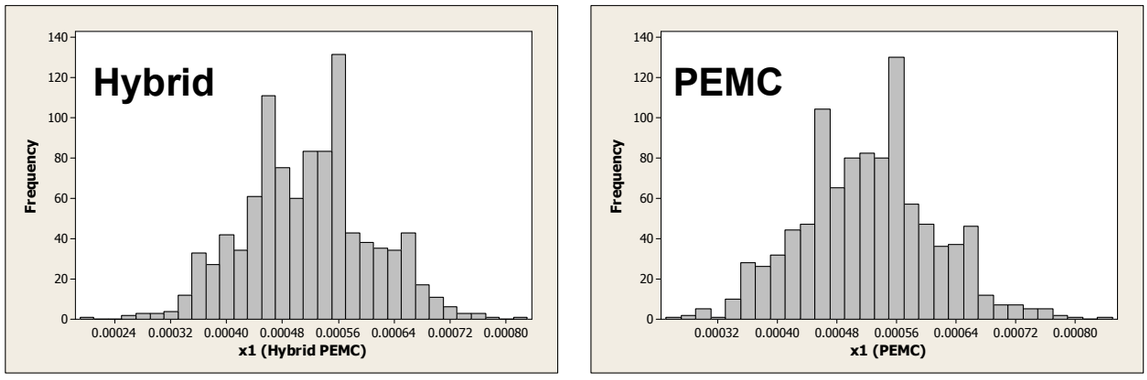
algorithm can be utilized. As the rate of aggregation is reduced with time, the hybrid algorithm correctly switches to a PEMC simulation of the entire system. The hybrid algorithm can simulate stochastic variables (A-B) at speeds approaching τ-PEMC.

The statistics of the condensation of B monomers for the case *W = 100* is summarized in Table 4 for 1,000 simulations. The histogram of the parameter x_1 (which measures the concentration of B particles on A particles) for the hybrid-PEMC and PEMC solutions is shown in Figure 13. The hybrid and PEMC solutions are in excellent agreement. This result is remarkable considering that condensation events are very rare (~30 condensation events vs. ~60,000 aggregation events). The box plot for these parameters is shown in Figure 14. An analysis of variance was performed to compare the population generated by both methods.

For the case of x_1, both simulations are statistically equivalent. The hybrid algorithm is 17 times faster than the PEMC algorithm per trajectory. This increase in speed is very important, since many trajectories (~10^3) are needed to generate good statistics for the slow process. If the model is used for optimization, many design instances are needed (~10^4), each consisting of many trajectories, resulting in a very large number of simulations (~10^7). In this case any increase in the simulation speed of a trajectory will have a tremendous impact on the total simulation speed.

	PEMC	Hybrid-PEMC
Average value of x_1	5.20×10^{-4}	5.13×10^{-4}
Standard deviation of x_1	8.9×10^{-5}	9.0×10^{-5}
Average number of slow aggregation events	58,623	58,629
Average number of condensation events	31.2	30.8

Table 4. A comparison of hybrid-PEMC and PEMC for the test problem with *W = 100* (Average of 1,000 trajectories).



(a) (b)

Fig. 13. Histogram of the x_1 parameter for the test problem with W=100 (1,000 trajectories). (a) Hybrid-PEMC and (b) PEMC.

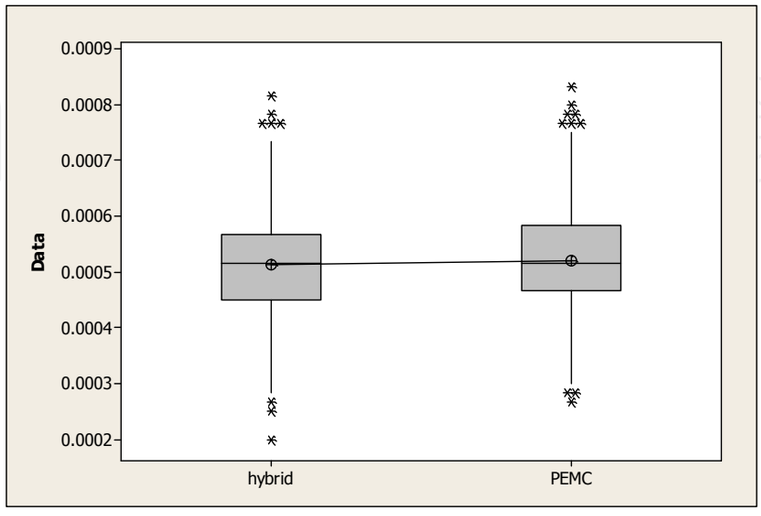


Fig. 14. A box-plot comparison of the hybrid-PEMC and PEMC solutions (W=100).

Hybrid and MC simulations produce statistically equivalent results, but the hybrid's increase in computational speed allows optimization problems involving these types of models to be solved very efficiently.

7. Conclusions and discussion

This chapter reviewed the artificial chemical process paradigm for global optimization. The LARES algorithm is very robust and efficient, converging to near-global optimal solutions when solving different classes of problems with different degrees of multi-modality, epistasis, flatness, and discontinuities. Future research will consider the use of the ACP paradigm in the development of new problem-specific algorithms. The algorithm was utilized to develop dynamic optimization strategies, LARES-PR and hybrid LARES-PR. The power of the algorithm lies in its utilization of a generalized approximation of the control function, composed of variable-length segments of finite elements of different orders or using specified functions. This generalized representation of the trial control law is possible due to the two-step encoding of the decision variables and the capability of LARES for multiple encoding. Multiple encoding allows the inclusion of different types of problem-specific finite elements (constant, linear, quadratic, etc.) and/or specialized functions to approximate the control law without any tailoring of the optimization algorithm. This approach is particularly effective for the solution of problems in which manipulated variables experience transition from smooth variations over time to discrete changes. Numerical experiments demonstrate that this algorithm is robust in finding global optimums for the different types of problems and definitions of the generalized control law introduced in this work.

To accelerate optimization of systems that use MC simulations as part of their constraints, a new general-purpose MC algorithm for the dynamics of the particulate process was proposed, PEMC. The method has been shown to reduce CPU time without sacrificing simulation accuracy. While a coarse view of the process is taken, particle history is retained. The method was extended with the τ-PEMC method, proposed to further improve CPU time with negligible simulation errors. As with the original PEMC, internal coordinates can be handled effectively. The CPU times reported here show that accurate results can be achieved with simulation times less than a second using a low-end PC. These results demonstrate the feasibility of stochastic optimization using PEMC and τ-PEMC. A new hybrid strategy was developed to solve stochastic population balance models with multiple time scales. This self-consistent hybrid method combines the PEMC and τ-PEMC algorithms to accelerate simulation time while capturing the stochastic nature of the slow process. The simulation speed and accuracy of the hybrid strategy depends on the selection of the τ parameter, the criteria for the partition between slow and fast events, and the grid quality of the point ensembles.

8. References

Banga J. R., Balsa-Canto E., Moles C. G., and Alonso A. A. (2003), Dynamic optimization of bioreactors: a review, *Proceedings of the Indian Academy of Science* Part A, 69, 257-266.

Banga J. R., Irizarry-Rivera R., and Seider W. D. (1998), Stochastic optimization for optimal and model-predictive control, *Computers & Chemical Engineering*, 22, 603-612.

Banga J.R., Irizarry R., and, Seider W.D., (1998) Stochastic optimization for optimal and model-predictive control, *Computers Chem. Engng*. 22: 603-612.

Chatterjee, A., Vlachos, D.G. and Katsoulakis M.A. (2005) Binomial distribution based τ-leap accelerated stochastic simulation. *J. Chem. Phys.*, 122 024112.

Cuthrell J.E. and L.T. Biegler (1989), Simultaneous optimization and solution methods for batch reactor control profiles, *Comput. Chem. Engng.*, 13, 49-62.

De Jong, K.A., M.A. Potter, and W.M. Spears (1997). Using Problem Generators to Explore the Effects of Epistasis. Proceedings of the Int'l Conference on Genetic Algorithms.

Dorigo, M. and Stutzle T. (2004) Ant colony optimization. Bambridge, MA; MIT Press.

Duvigneau R. and Visonneau M. (2004) Hybrid genetic algorithms and artificial neural networks for complex design optimization in CFD, *Int. J. Numer. Meth. Fluids*, 44, 1257-1278.

Garcia, A.L., van den Broek, C., Aertsens, A., and Serneels, (1987) A Monte Carlo method of coagulation. *Physica* (1987) 143A, 535-546.

Gillespie, D. T. (1975), An Exact Method for Numerically Simulating the Stochastic Coalescence Process in a Cloud. *J. Atmos. Sci*. 32 1977-.

Gillespie, D.T. (1077) Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.* 81 2340–2361.

Gillespie, D.T. (1976) A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics* 22 403–434.

Gillespie, D.T. (2001) Approximate accelerated simulation of chemically reacting systems. *J. Chem. Phys.* 115, 1716-1733.

Gillespie, D.T. and Petzold (2003) Improved leap-size selection for accelerated stochastic simulation. L.R., *J. Chem. Phys.* 119 (2003) 8229-8234.

Goldberg, D.E. (1989), Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA:Addison-Wesley.

Gottlieb J., E. Marchiori and C. Rossi (2002), Evolutionary Algorithms for the Satisfiability Problem, *Evolutionary Computation* 10(1), 25-49.

H. Salis and Y. Kaznessis (2005) Accurate hybrid stochastic simulation of a system of coupled chemical or biochemical reactions *J. Chem. Phys.* 122 54103-1-13.

Hansen N., S.D. Muller and P. Koumoutsakos (2003), Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES), *Evolutionary Computation*, 11(1), 1-18.

Haseltine E.L. and Rawlings J.B., *J. Chem. Phys.* 117 (2002) 6959-6969.

Haseltine E.L. and Rawlings J.B., *J. Chem. Phys.* 123 (2005) 164115 -1–15.

Holland, J. H. (1975). Adaptation in natural and artificial systems. Ann Arbor: University of Michigan Press.

Ingber, L. (1993), Simulated Annealing: Practice versus Theory, *J. Mathl. Comput. Modelling* 18(11), 29-57.

Ingber, L. and B. Rosen (1992), Genetic Algorithms and Very Fast Simulated Reannealing: A Comparison, *J. of Mathematical and Computer Modeling* 16(11), 87-100.

Irizarry R. (2004) LARES: An Artificial Chemical Process Approach for Optimization, *Evolutionary Computation Journal*, 12 (4), 435-460.

Irizarry R. (2005a) A Generalized Framework for Solving Dynamic Optimization Problems using the Artificial Chemical Process Paradigm: Applications to Particulate

Processes and Discrete Dynamic Systems, *Chemical Engineering Science*, 60, 5663-5681.

Irizarry R. (2005b) Fuzzy classification with an artificial chemical process, *Chemical Engineering Science*, 60, 399-412.

Irizarry R. (2006) Hybrid Dynamic Optimization using Artificial Chemical Process: Extended LARES-PR, *Industrial & Engineering Chemistry Research*, 45, 8400-8412.

Irizarry R. (2007a) Fast Monte Carlo Methodology for Multivariate Particulate Systems-I: Point Ensemble Monte Carlo. *Chemical Engineering Science*, 63, 7649 – 7664.

Irizarry R. (2007b)Fast Monte Carlo Methodology for Multivariate Particulate Systems-II: τ-PEMC., *Chemical Engineering Science* 63, 7665 – 7675.

Irizarry R. (2010a) Simulated dynamic optical response strategy for model identification of metal colloid synthesis. *Ind. Eng. Chem. Res.*, 49, 5588–5602.

Irizarry R. (2010b) Multi-time scale point ensemble Monte Carlo method for the solution of stochastic population balance models. *Proc. 4th International Conference on Population Balance Modeling* September 15-17 2010, Berlin, Germany, 771-788.

Kaznessis Y. N. (2006) Multi-scale models for gene network engineering, *Chem. Eng. Sci.* 61 940 – 953.

Kennedy, J. and Eberhart, R.C. (2001) Swarm intelligence. Silicon Valley, LA.H.: Morgan Kaufmann Publishers.

Kevrekidis I. G., Gear C.W., and Hummer G. (2004), Equation-Free: The Computer-Aided Analysis of Complex Multiscale Systems, *AIChE J.*, 50, 1346-1355.

Kirkpatrick, S., C.D. Gelatt, Jr. and M.P. Vecchi (1983), Optimization by Simulated Annealing, *Science* 220(4598), 671-680.

Liu B., Wanga L., Liud Y., Qiane B., Jin Y. (2010) An effective hybrid particle swarm optimization for batch scheduling of polypropylene processes, *Computers and Chemical Engineering*, 34 (2010) 518–528.

Luus, R. (1992), On the application of iterative dynamic programming to singular optimal control problems, *IEEE Transac. Autom. Control* 37(11), 1802-1806.

Luus, R. (2000), Iterative dynamic programming. London, UK; Chapman and Hall/CRC.

Ma D. L., Tafti D. K., and Braatz R. D. (2002), Optimal control and simulation of multidimensional cystallization process, *Comput. Chem. Engng.* 26, 1103-1116.

Mathur M., S.B. Karale, S. Priye, V.K. Jayaraman and Kulkarni B.D. (2000), Ant Colony Approach for Continuous Optimization, *Ind. Eng. Chem. Res.* 39, 3814-3822.

Mitchell, D., B. Selman, and H. Levesque (1992). Hard and easy distributions of SAT problems. In Proceedings of the Tenth National Conference in Artificial Intelligence, 459-465. AAAI Press/The MIT Press.

Moles C.G., Banga J.R., Keller K. (2004), Solving nonconvex climate control problems: pitfalls and algorithm performances, *Applied Soft Computing*, *in press*.

Mühlenbein, H. and D. Schlierkamp-Voosen (1993), Predictive Models for the Breeder Genetic Algorithm, I. Continuous Parameter Optimization, *Evolutionary Computation* 1(1), 25-49.

Nordhaus, W.D. (1994), Managing the Global Commons: The Economics of climate change. MIT press, Cambridge, Massachusetts.

Park S. and Ramirez W. F. (1988), Optimal production of secreted protein in fed-batch reactors, *A.I.Ch.E. J.*, 34, 1550-1558.

Pistikopoulos E.N. (2009) Perspectives in multiparametric programming and explicit model predictive control, *AIChE J*., 55, 1918.

Roubos J. A., van Straten G., and van Boxtel A. J. B. (1999), An evolutionary strategy for fed-batch bioreactor optimization; concepts and performance, *J. of Biotechnology,* 67(2/3), 173-178.

Sahinidis N.V. (2009) optimization techniques in molecular structure and function elucidation, *Computers and Chemical Engineering*, 33, 2055-2062.

Sarkar D. and Modak J. M. (2003a), Optimization of fed-batch bioreactors using genetics algorithms, *Chemical Engineering Science*, 58, 2283-2296.

Schwefel, H.P. (1995), Evolution and Optimum Seeking, John Wiley.

Selman, B. and H. Kautz (1993). An Empirical Study of Greedy Local Search for Satisfiability Testing. Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93), Washington, D.C.

Spears, W.M. (1998). The Role of Mutation and Recombination in Evolutionary Algorithms. Ph.D. Dissertation, George Mason University, Fairfax, Virginia.

Storn, R. and K. Price (1997), Differential Evolution- A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, *Journal of Global Optimization* 11, 341-359.

Tanartkit P. and Biegler L. T. (1995), Stable decomposition for dynamic optimization, *I &EC Res.*, 34, 1253-1266.

Tanartkit P. and L.T. Biegler (1996), A nested, simultaneous approach for dynamic optimization problems-I, *Comput. Chem. Engng*., 20, 735-741.

Tang K., Sun T., Tang J. (2010) An improved genetic algorithm based on a novel selection strategy for nonlinear programming problems. *Computers and chemical engineering* (article in press).

Tian, T. and Burrage, K., *J. Chem. Phys.* 121, (2004) 10356-10356.

Tomshine J. and Kaznessis Y.N. (2006) Optimization of a Stochastically Simulated Gene Network Model via Simulated Annealing. *Biophysical Journal* 91 (2006) 3196–3205.

Vassiliadis V. (1993). Computational solution of dynamic optimization problems with general differential-algebraic constraints. PhD Thesis, Imperial College, University of London, UK.

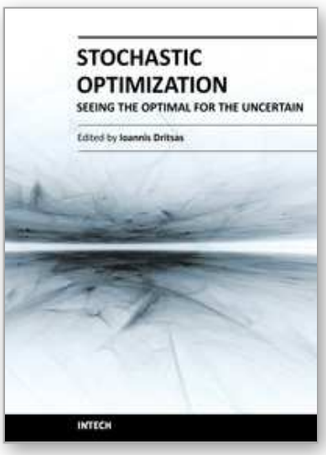
Vassiliadis V. S., Canto E. B., and Banga J. R. (1999), Second-order sensitivities of general dynamic systems with application to optimal control problems, *Chemical Engineering Science*, 54, 3851-3860.

Voigt, H. M. (1995), Soft Genetic Operators in Evolutionary Computation, Evolution and Biology Computation, Lecture Notes in Computer Science 899, Springer, Berlin, pp.123-141.

Yan L. and D. Ma (2001), Global optimization of non-convex nonlinear programs using Line-up Competition Algorithm, *Computers and Chemical Engineering* 25, 1601-1610.

Yip, P. and Y.H. Pao (1995), Combinatorial Optimization with the Use of Guided Evolutionary Simulated Annealing, *IEEE Transactions on Neural Networks*, 6(2), 290-295.

Zienkiewics O. C., The finite Element Method, third ed. McGraw Hill, New York (1977).



Stochastic Optimization - Seeing the Optimal for the Uncertain
Edited by Dr. Ioannis Dritsas

ISBN 978-953-307-829-8
Hard cover, 476 pages
Publisher InTech
Published online 28, February, 2011
Published in print edition February, 2011

Stochastic Optimization Algorithms have become essential tools in solving a wide range of difficult and critical optimization problems. Such methods are able to find the optimum solution of a problem with uncertain elements or to algorithmically incorporate uncertainty to solve a deterministic problem. They even succeed in â€œfighting uncertainty with uncertaintyâ€. This book discusses theoretical aspects of many such algorithms and covers their application in various scientific fields.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Roberto Irizarry (2011). Global and Dynamic Optimization using the Artificial Chemical Process Paradigm and Fast Monte Carlo Methods for the Solution of Population Balance Models, Stochastic Optimization - Seeing the Optimal for the Uncertain, Dr. Ioannis Dritsas (Ed.), ISBN: 978-953-307-829-8, InTech, Available from: http://www.intechopen.com/books/stochastic-optimization-seeing-the-optimal-for-the-uncertain/global-and-dynamic-optimization-using-the-artificial-chemical-process-paradigm-and-fast-monte-carlo-

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen