Nonsmooth Dynamic Simulation with Linear Programming Based Methods

Vipin Gopal and Lorenz T. Biegler

EDRC 06-207-95

Nonsmooth Dynamic Simulation With Linear Programming Based Methods

Vipin Gopal and Lorenz T. Biegler*

Department of Chemical Engineering,
Carnegie Mellon University, Pittsburgh, PA 15213
email: biegler@cmu.edu

Abstract: Process simulation has emerged as a valuable tool for process design, analysis and operation. In this work, we extend the capabilities of iterated linear programming (LP) for dealing with problems encountered in dynamic nonsmooth process simulation. A previously developed LP method is refined with the addition of a new descent strategy which combines line search with a trust region approach. This adds more stability and efficiency to the method. The LP method has the advantage of naturally dealing with profile bounds as well. This is demonstrated to avoid the computational difficulties which arise from the iterates going into physically unrealistic regions. A new method for the treatment of discontinuities occurring in dynamic simulation problems is also presented in this paper. The method ensures that any event which has occurred within the time interval in consideration is detected and if more than one event occurs, the detected one is indeed the earliest one. A specific class of implicitly discontinuous process simulation problems, phase equilibrium calculations is also looked at. A new formulation is introduced to solve multiphase problems.

Keywords: Nonsmooth Simulation, Iterated Linear Programming, Line Search, Trust Region, Profile Bounds, Discontinuity.

1. Introduction

Recent years have seen the emergence of *process simulation* as a valuable tool for plant design, analysis and operation. A variety of factors like increased safety concerns and strict environmental regulations have contributed to this trend. Of increasing interest in the simulation community is the development of efficient *dynamic simulation* tools. Many authors have surveyed the present and future applications of such tools in chemical process industries (Perkins, 1986; Marquardt, 1991; Naess *etal:*, 1992; Pantelides and Barton, 1993).

* to whom all correspondence should be addressed

Equation oriented simulators, as opposed to their sequential modular counterparts, rely on an equation solving engine which must be able to handle poor starting points, inequality constraints, singularities and ill-conditioning among others. Steady state and dynamic chemical process simulation also invariably encounter variables which need to be restricted within certain bounds. Typical examples are the nonnegativity restrictions on physical quantities like mass, volume or temperature. Process models could show abnormal behavior if these variables are allowed to take up values beyond these bounds because of physical or mathematical infeasibility. A systematic treatment of these bounds by the solver can avoid convergence failures which might arise due to ad hoc strategies currently employed.

Another important issue is the treatment of nonsmooth relations characterized by discontinuous derivatives or kinks, which are often enforced as procedures in the sequential modular mode. Bullard (1991) examined some of these problems in the context of steady state simulation. Marquardt (1991) on the other hand, points out that most of the current dynamic simulation packages are developed to solve models of predominantly continuous nature. However, few industrial processes could be considered to operate in a truly continuous manner. Simulation of dynamic chemical processes often involve discrete actions and logical constraints and these can lead to discontinuities in the modeling equations. Process models can thus be nonsmooth at various points in state space, with these points specified through explicit or implicit relations. Typical examples of these include phase and flow transitions, heat and mass transfer correlations in various regimes and startup and shutdown operations in continuous plants. The discontinuity might cause either a switch in the model equations or even a change in the structure and dimensionality of the plant model. Hence an effective and consistent representation mechanism of the model discontinuities and their solution technique is very important in an equation based modeling environment.

A method based on iterated linear programming is presented in this paper for the treatment of variable bounds and nonsmoothness occurring in process simulation problems. In the next section we look at a previously developed iterated linear programming technique for constrained simulation problems modified with an improved descent strategy. The LP based method provides a very natural way of incorporating variable bounds. Other than bounds on variables, physically realistic conditions could also be added through simple inequalities. Section 3 focuses on the need for an efficient treatment of variable bounds and how the LP method provides a framework for the purpose. A new algorithm for the treatment of discontinuities occurring in dynamic simulation problems is presented in

section 4. Section 5 describes the treatment of implicit discontinuities by looking at a specific class of problems: phase equilibrium calculations. Conclusions and future work are outlined in section 6.

2. Improved Iterated LP Method for Constrained Simulation Problems

Bullard and Biegler (1991) introduced a technique based on iterated linear programming for solving a nonlinear system of equations subject to inequality constraints and variable bounds.

$$\begin{aligned} h/x) &= 0 \\ g_k(x) &\leq 0 \\ x^L \leq x \leq x^u \end{aligned} \qquad \text{(CSP)}$$

This problem is termed as the *constrained simulation problem (CSP).* The basic idea of the approach is to convert the equation solving problem to an optimization problem. A sequence of linear programs is then solved to yield search directions that will lead to the solution of the nonlinear problem.

To solve (CSP), a merit function is defined as

$$\mu(x^i) = \sum_j \left|h_j(x^i)\right| + \sum_k g_k(x^i)_+ \qquad (1)$$

where $g_k(x^*)+ = \max[0, £_4(JO]$. Since /i ➢ 0 has a minimum of zero if and only if (CSP) is solved, *JJL* a non differentiate function, is minimized subject to $x^L \leq x \leq x^v$. By adding auxiliary variables, this problem can be written as :

$$\begin{aligned} \min & \sum_j (p_j + n_j) + \sum_k s_k \\ s.t.\ & h_j(x) = p_j - n_j \\ & g_k(x) \leq s_k \\ & x^L \leq x \leq x^u \\ & p_{Jf} n_{Jt} s_k \geq 0 \end{aligned} \qquad (2)$$

Barrodale and Roberts (1978) proved that for $£pj + n_j$ at a minimum, a complementarity condition $pp-_j = 0$ holds. Thus $(p_j + n_j)$ represents $\backslash hj(x)\backslash$ and s_k represents $g_k(x)+$. Linearizing the constraints about x^J leads to the following constrained simulation linear program, CSLP.

$$
\begin{aligned}
&\min \sum_j (p_j + n_j) + \sum_k s_k \\
&\text{s.t. } h_j(x^i) + \nabla h_j(x^i)^T d = p_j - n_j \\
&\quad g_k(x^l) + \nabla g_k(x^l)^T d \le s_k \qquad \text{(CSLP)} \\
&\quad x^L \le x^l + d^l \le x^v \\
&\quad p_{Jf} n_{Jf} s_k \ge 0
\end{aligned}
$$

The solution of (CSLP) generates a search direction *d*. To stabilize the performance of the algorithm, a line search is carried out to find some fraction *a* so that JC*+^= x^i + *ad*. It should be noted that this method reduces to Newton's method in the absence of active inequality constraints and thus has the ability to converge quadratically to the solution. This LP approach is also shown to generate a descent direction for the 1-norm of the constraint violations. The method has global convergence properties for a nonzero solution of the linear program. The algorithm when tested on problems ranging from 2 to 977 equations and compared favorably with the existing methods.

Here a new descent strategy is incorporated into the aforementioned algorithm by combining two major classes of strategies used to ensure descent: line search and trust region methods. Dennis and Schnabel (1983) and Fletcher (1987) provide a detailed study of these methods.

Line Search Methods

The basic concept behind using line search methods is to choose a descent direction d^l from the current point x^1 and select an 'acceptable[1] point x^{1*1} in this direction at which *H(x)* decreases. The implementation for the algorithm by Bullard and Biegler (1991) uses the traditional monotonic, Armijo-type line search (Armijo,1966). The search direction *d!* is obtained from the solution of (CSLP) and a condition (3) that requires a sufficient decrease in the merit function is enforced.

$$
\mu(x^i \cdot \backslash - a d^1) - \; ii(x^l) \le 0.1a \;\; D_d n \qquad (3)
$$

Here $D_d\mu$ is an upper bound on the directional derivative of the merit function, which is given by

$$
D_d\mu = \sum_j (p_j \qquad t \qquad j \qquad k
$$

At the full step *(a=l),* (3) is evaluated. If satisfied, the variables are updated and a new step is calculated. Otherwise a new fractional step (5) is calculated from a quadratic interpolation of the merit function and continued until a suitable *a* is found.

$$a = maxLoia, \frac{-0.5\ \alpha^2 D_d \mu}{\mu(x^i + \alpha d^i) - n(x^i) - \ aDji} \Bigg\} \qquad (5)$$

Since the merit function is nondifferentiable, it is possible for the *Maratos effect* to occur (Fletcher, 1987). Here JC[1] may be arbitrarily close to the solution JC\ but a full step may fail to reduce the merit function. A nonmonotonic line search such as the watchdog technique (Chamberlein *et. ai,* 1982) or a second order correction (Fletcher, 1987) could be used to overcome this effect.

Trust Region Methods

Search directions generated by CSLP are always descent directions *($D_d$fi <0)* for *d* * 0, but they still may be poor if the problem is ill-conditioned. In such cases, the search direction is relatively large and is nearly orthogonal to the steepest descent direction. Trust region methods avoid this problem by restricting the stepsize d^t to an area A^i around the current point x^r. The trust region is frequently adjusted after each step depending on how valid the approximations of the model are. Most of the cunrent implementations increase or decrease the trust region by comparing the *actual* reduction in the merit function and the *predicted* reduction on linearization. The closer their ratio is to unity, the better the agreement. Duff *et. al,* (1987) describe this strategy in the context of using linear programs to solve sparse nonlinear equations.

A Combination

The main difference between line search and trust region methods lie in the sequence in which they use the linear model and enforce the acceptable step length. The line search methods uses a linear model to obtain a search direction d and then chooses a fractional step a. Thus the line search occurs on a 1-dimensional subspace. On the other hand, trust region methods first choose a maximum acceptable step length A and then use the linear model to obtain a search direction d, the length of which cannot exceed A. The search here is not restricted to a lower dimensional subspace. The trust region step may not always be the Newton direction as small A[f]s will cause the step to be in the direction of steepest descent. Nevertheless, both methods exhibit quadratic convergence near the solution.

Relatively little work is necessary for the line search method as only one LP is solved at each iteration. But, when trust region methods are used, finding the ideal trust region

bounds is likely to require solution of more than one LP per iteration. However, trust region methods require few major iterations as they use a Newton step if it lies within the trust region.

In a combination of these two methods, the trust region eliminates the possibility of a large ill-conditioned LP step which result in slow convergence. On the other hand, a line search helps to reduce the number of times the LP should be solved leading to less computational expense. First the trust region constraints are added to the linear model to generate a search direction d\ A line search is done in this direction to get the new point x^M. The predicted and actual reduction in merit function is then calculated on the basis of the new point and the size of the trust region is then updated. The detailed algorithm is given below.

0. i=0.

Set trust region bounds

$\overline{4} = 0.5 \ max(x^\circ - x^L, x^u - x^\circ) \qquad y^\circ = 1$

Initialize the problem (at the solution of the previous time step in the dynamic problem).

1. Evaluate h, Vh, g, Vg (constraints and their gradients).

2. Solve $CSLPfx\hat{}d')$ with trust region constraints.

3. Compute 'predicted' change in exact penalty by linearization

   $\mu_l(x^i, d^i) - \mu(x^i) \approx \Delta\mu_{li}$

   If $\backslash Aji_a \backslash \leq e_n$ return. Else go to 4.

4. Setct=l.

5. JC" = $x^1 + ad^1$. Calculate ^^(exact penalty at the new point x^N)

6. If the actual change in exact penalty $An_t = /i_N - ii(x') \leq 0J \ a \ D_d ii$, go to 7.

   Else set $a = max\{W.Ola, \frac{-o5c f^D}{\quad} \wedge \frac{\quad}{\quad} 1$ ·Go to 5.

   ($D_d fi$ is an upper bound on the directional derivative evaluated from the solution of the CSLP)

7. New point $JC^{i+/} = JC^1 + ad^1$

8. Set the trust region radius for the next iteration

   Calculate the ratio of *actual* to *predicted* change $r_1 = \dfrac{\mu(x^i + \alpha d^i) - \mu(x^i)}{\mu_l(x^i, \alpha d^i) - \mu(x^i)}$

$$\gamma^{i+1} = \begin{cases} \gamma^i / m & \text{if } r_i < \rho_1 \\ m\gamma^1 & \text{if } n > p2 \\ Y & \text{otherwise} \end{cases}$$

The next trust region $4^{<>y} = A \ max(/^{>;}, y')$

9. i = i-h7. go to 2.

Parameters p_y, p_2 and m are fixed at 0.25, 0.75 and 2 respectively similar to the values used by Zhang *et. al.* (1985). y' is set to 10^{-3}. The convergence criteria employed here is $|\wedge/ijj \leq \pounds j$ (step 3). Note that if (3) is not satisfied and the line search fails, $r_i < 0.10$ and y is automatically decreased. If $fi; \leq e_2$, the problem is solved. Otherwise, x^i is a stationary point which we term as a *pseudosolution.* Heuristic strategies for recovering from a pseudosolution (e.g. rescaling the problem) are described by Bullard and Biegler (1991).

Numerical examples

The iterated LP method with the new descent strategy was tested on 14 small problems reported in the literature. All the problems were solved using a Fortran program OPTLP which implements the combined line search / trust region descent strategy. It uses QPSOL (Gill *et. al.*, 1983) for the solution of the LPs and implements a relaxation formulation to handle inconsistent linearizations. The tolerance specified was that the 1-norm of the constraint violations be less than 10^{-7}. The numerical results for the solution of these problems without scaling are listed in Table 1. The individual problem descriptions could be found in Bullard and Biegler (1991) (The numbers in parentheses in column 1 refer to the problem numbers used in that paper).

The question which arises in the context of descent in this framework is when to apply which strategy. In general, line search methods perform better than trust region methods on well-conditioned problems whereas trust region methods show better performance with ill-conditioned ones. They also perform similarly in a vast majority of problems.

For problem 1 a pure trust region method (5 LPs solved) is a better strategy than a pure line search method (18 LPs solved). However, the combined method solved the problem in 5 iterations, showing a performance similar to the better trust region method. On the other hand, line search shows better performance than trust region in problems 2,3,4 and 14. From the table it is evident that the combined method shows a performance similar to the better strategy here, the line search. Hence, for the test problems considered, when there is

a significant difference in the two descent methods, the combined strategy shows a performance similar to the better method.

Problem	Number of Variables	Number of LPs solved		
		Line Search	Line Search + Trust Region	Trust Region
1 (21)	1	18	5	5
2 (10)	2	12	12	21
3 (17)	4	11	11	18
4 (11)	4	15	14	27
5 U«)	2	10	10	14
d[b])	2	10	9	9
U[c])	2	10	10	10
6 (2)	2	6	6	6
7 (4)	2	14	13	13
8 (7)	2	7	7	7
9 (8)	2	5	5	5
10 (12)	2	4	4	4
11 (15)	2	5	5	5
12 (16)	2	6	6	6
13 (9)	3	6	6	6
14 (19»)	7	8	8	pseudo-solution
(19[b])	7	15	15	>100
(19°)	7	9	9	11

Table 1: Comparison of number of LPs solved with pure line search, pure trast region and a combination of line search and trust region methods.

For problems 5-13 there is not much difference between the line search and trust region methods as the number of LPs solved are almost the same. The combined method is as good as either of them. Thus, in problems where there is no significant difference between a line search and a trust region approach, a combination of the two methods does not lead to a decrease in computational efficiency.

These observations are significant because the combined method performs at least as well as either the trust region or the line search methods on all the problems, regardless of whether there is a difference between the individual performance of these methods on a particular problem. Larger numerical examples are currently under consideration. The combined strategy is expected to perform better for larger problems as well.

3. Treatment of profile bounds

Converting the equation solving problem to an optimization framework allows for inequalities and conditionals to be added easily. This allows the user to incorporate physically meaningful conditionals and inequalities limiting the operation of the process under consideration, in a natural fashion. These could be thermodynamic constraints governing the behavior of the system or even simple physical constraints (For eg. *sum(component holdups)* $\leq$ *volume of the vessel)*. Bounds on functions can thus be easily added into this framework.

Chemical process simulation problems often contain variables like temperature, pressure and molefraction, which are restricted to be within a certain region. Process models are often formulated with the assumption that the variables lie within their specified bounds. It is quite likely that process models show egregious behavior if these variables are allowed to venture out of their specified region. Hence it is important to avoid regions of *physical* and *mathematical* infeasibility in process simulation problems. Most of the conventional nonlinear equation solving techniques do not have a systematic and efficient technique for enforcing bounds on variables. The LP based method presents a very natural way of dealing with variable bounds. The LP method limits the iterates within the bounds on variables and avoids the mathematical problems which arise otherwise.

In the following example we show how enforcement of variable bounds in an LP method helps to solve the problem by restricting the iterates from going into regions of mathematical infeasiblilty.

Example

This example is based on a parameter estimation problem originally formulated by the Dow Chemical Company (Blaueffl/., 1981).

$$\wedge \quad = -k_2 y_8 y_2 \tag{6a}$$

$$\frac{dy_2}{dt} = -*/JW2 + {}^{k}syw - \wedge\wedge \tag{6b}$$

$$\frac{dy_3}{dt} = -k_2 y_8 y_2 + k_1 y_6 y_4 - 0.5 \wedge y_9 \tag{6c}$$

$$\frac{dy_4}{dt} = -k_1 y_6 y_4 + 0.5 k_3 y_9 \tag{6d}$$

$$\frac{dy_5}{dt} = -k_1 y_6 y_2 - k_3 y_{10} \tag{6e}$$

$$\frac{dy_6}{dt} = -k_1 (y_6 y_2 + y_6 y_4) + k_3 (y_{10} + 0.5 y_9) \tag{6f}$$

$$y_7 = -K_4 + y_6 + y_8 + y_9 + y_{10} \tag{6g}$$

$$y_8 = \frac{K_2 y_1}{K_2 + y_7} \tag{6h}$$

$$y_9 = \frac{K_3 y_3}{K_3 + y_7} \tag{6i}$$

$$y_{10} = \frac{K_1 y_5}{K_1 + y_7} \tag{6j}$$

$$y_{11} = K_6 \ln(y_7 + \wedge \mathrm{j} \tag{6k}$$

Here, y's represent the state variables and *K's* and ik's are parameters as reported by R.H.Farris (Biegler *et al.*, 1986). Values of the parameters are given by (7).

$k_1 = 1.8192$

$k_2 = 2.8595$

$k_3 = 2926$

$K_1 = 2.575 \times 10^{-16}$

$K_2 = 4.876 \times 10^{-14}$

$K_3 = 1.7884 \times 10^{-11}$ (7)

$K_4 = 1.31 \times 10^{-2}$

$K_5 = 1 \times 10^{-10}$

$K_6 = 1 \times 10^{13}$

The problem was attempted under two conditions: one with the specification of bounds and the other without. In the unbounded case, the solution of the first LP gave a search direction which led to a negative value of y_7. (6k) could not be evaluated at that point and the method failed. However, in the bounded case, the variables y_1 to y_w were bounded below by zero as they represent concentrations in the original problem. The values of y_7 obtained from solving the dynamic problem with the incorporation of bounds are plotted in figure 1. y_7 is very close to zero and with the imposition of bounds, the LP method never allows the iterates to go beyond them, which caused the numerical difficulty in the previous case. Solution of this problem required 1163 LP iterations for 494 time steps.

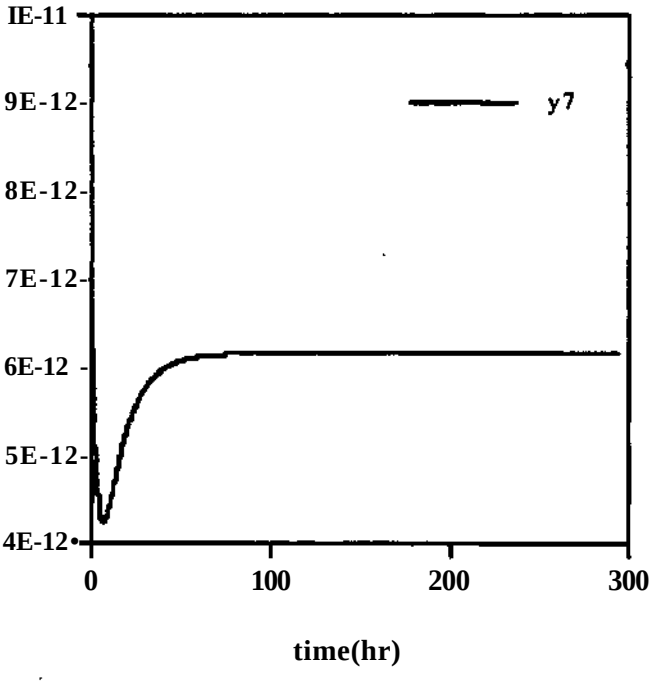


Figure 1: Solution of the problem with the incorporation of bounds

Another important aspect of the treatment of bounds is in step- size control. Although the DAE solution profile lies within the specified bounds, the discretized equations could have a solution which lie outside the variable bounds. The methods which cannot handle the treatment of bounds might accept this solution as the true solution and proceed with the integration. The.LP method will terminate with a pseudo-solution (a solution at the bounds) with the current step-size when there is no solution within the specified bounds and the problem is then resolved with a smaller step size for more accurate results.

4. Explicit discontinuities

Very few industrial processes operate in a 'truly[1] continuous manner. Discontinuities in the modeling equations occur as a result of discrete actions and logical conditions, in the dynamic simulation of these processes. In this section we look at various aspects of this problem.

Discrete changes in the modeling equations are marked by the occurrence of an *event.* The methods employed to solve the dynamic problem should thus be able to determine the time of occurrence of these events. Two types of events can occur depending upon the manner in which the time of occurrence of the event is determined (Barton, 1992): *time* and *state* events. The time of occurrence of *time events* is known in advance making them easier to deal with. However, *state events* occur as a result of the system satisfying certain logical conditions and hence the time of occurrence is not known in advance. Hence the detection of state events calls for condition monitoring while the problem is being solved in a particular time interval and determining the exact time at which the conditions are satisfied.

A related issue associated with these problems is the detection of the correct state event. Consider the scenario in which more than one logical condition is satisfied within shon times. The solution algorithm should be able to determine which of those logical conditions is satisfied first and identify the state event associated with it. This is important since, if the wrong logical condition is detected as having been satisfied first, we might end up with an entirely different set of equations in the next phase.

Dynamic simulation problems within a continuous interval can, in most cases, be described using a differential-algebraic equation (DAE) system. The differential equations describe the time dependent behavior of the system whereas the algebraic equations enforce physically meaningful relations. Once an event has occurred, the system is described by new set of equations (DAEs) and/or variables. A new set of initial values for these variables is required for further solution of the dynamic problem. Typically, they are

determined from the final values of the variables in the previous interval and from the conditions arising from the events themselves. In practice such an exercise is far from trivial for most of the problems.

In short, three important issues are to be considered while dealing with nonsmooth dynamic simulation problems -

1. Detection of (state) events which occur within a specified time interval
2. Exact location of the earliest event within that interval.
3. Reinitialization for restarting the integration following a discontinuity.

Here we see that LP-based methods can aid in event detection and resolution.

ExistingMethods

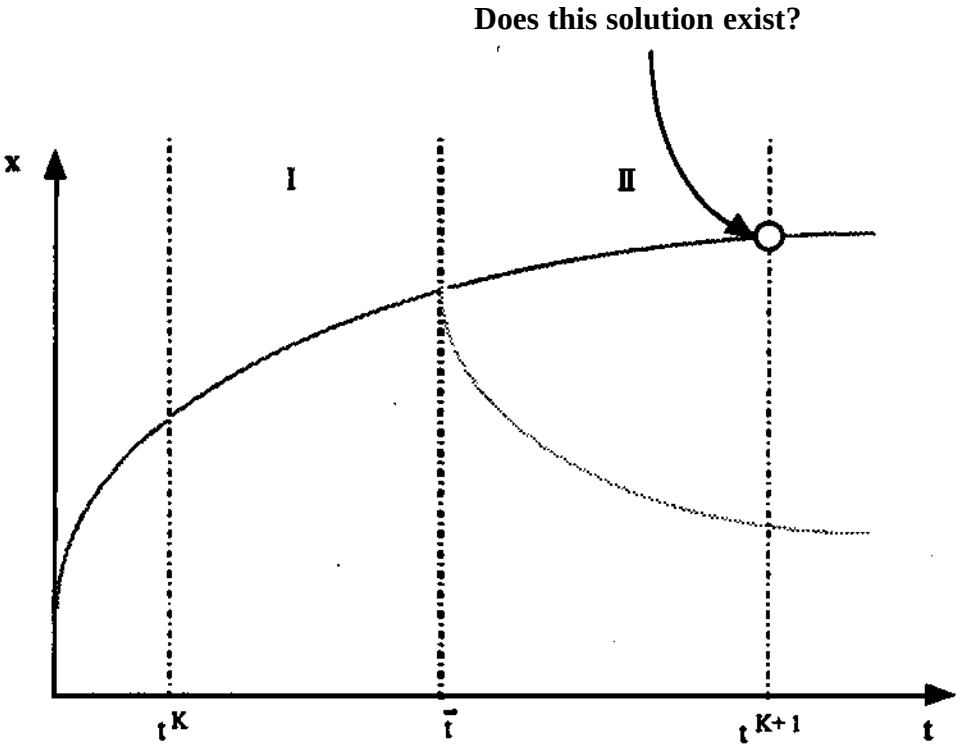
According to Marquardt (1991), the state of the art in dynamic simulation with discontinuities are the so-called *discontinuity locking* and *switchingfunctions* (Joglekar and Reklaitis, 1984; Pantelides, 1988a; Smith and Morton, 1988). When integration is carried out, the model is locked in one continuous case and no discontinuity is allowed. The switching functions are designed in such a fashion that their zeros correspond to the points of discontinuity. The switching functions are designed in such a fashion that they show a sign change when a switch in model takes place during an integration step. Once a discontinuity is detected, the occurrence time is located by an interpolation mechanism. The integration proceeds from that time onwards with the newly activated model. Below, we briefly examine some representative algorithms with respect to various aspects of this problem.

Existence ofSolution

An issue which has often been of concern in techniques using discontinuity locking is the question whether the current set of equations, which describes the region before the discontinuity, can provide a solution in the region beyond the discontinuity also. This is important because the discontinuity locking techniques require the current active set of equations to give a solution across the discontinuity for the detection and even location of the discontinuity (Figure 2). Nonexistence of solution across the discontinuity could cause presently used discontinuity locking methods to fail (Joglekar and Reklaitis, 1984; Pantelides and Barton, 1993)

The LP based approach however has the advantage of having a penalty function in (2) to allow constraint relaxation if solutions do not exist. In this case, the algorithm will

converge to a pseudosolution and we can use this to confirm that we have crossed a discontinuity.



— Solution of the model in I

····· Solution of the model in II

Figure 2: Existence of solution in discontinuity locking methods

Detectionandlocationofdiscontinuities

The first step in the treatment of discontinuities is their 'detection[1]. A broad spectrum of algorithms base the detection of discontinuities on the signs of the switching functions and their derivatives at the boundaries of the interval (e.g. Joglekar and Reklaitis, 1984). Such a procedure can fail to detect multiple zero crossings of the switching function within that interval (Park and Barton, 1993). An alternate method suggested to overcome this difficulty is to construct a polynomial approximation of the switching function and to solve for all the roots of this function within an interval (e.g. Park and Barton, 1993). The earliest of such roots determined is identified as the offending discontinuity. Most of these methods have been implemented in conjunction with the polynomial interpolation formulae generated by the Backward Difference Formula (BDF) methods employed to solve the DAEs themselves. These methods are promising, but there are some issues which need to be considered for polynomial-based techniques. In many problems it is likely that discontinuities are encountered one after the other in very short time intervals. In such a

scenario, the problem is reinitialized over and over again. Nordsieck (1962) points out that if the interval is changed while the strong transient is still present, this interval change itself results in a new shock excitation and the interval control tends to become erratic. Thus there is a certain amount of nonconformity in starting and interval changing, as starting essentially involves eliminating a very large transient and changing the interval during a large transient can lead to erratic interval behavior. Also the interval control itself may contain feedback loops and the polynomial coefficients may exhibit abnormal behavior. In this case, the polynomial approximations of the switching functions can no longer be trusted to determine the exact time of occurence of a discontinuity in such a scenario. Another question which arises in this context is whether one can isolate the correct event as occurring first with absolute certainty by using the polynomial approximations, in the event of more than one logical condition being triggered within short times. For reasons mentioned above and due to frequently occuring jump discontinuities in the simulation variables, it is quite possible that the wrong logical condition might be detected as having satisfied first. This favors the need for an iterative solving procedure rather than relying *only* upon the polynomial approximation for the detection of an event. Below we suggest such a procedure.

Proposed Algorithm

0. £=0 xP known (Initial conditions)
1. Problem at the next time step: $r^{*+/} = t^k + h^k$
   Solve the discretized problem using the LP based algorithm described before.
   Let x^{*+y} be the converged solution at that time.
2. Check for discontinuities (zeros of switching functions) within the time interval.
   If no discontinuity detected, k = k+1, go to 1. Else, continue.
3. Solve for f from $i^* = \pm$ tolerance
   (The sign depends on the direction of crossing of the switching function. If the switching function is negative before the discontinuity, + sign is chosen and if it is positive, - sign is chosen)
4. Resolve the problem at t\
5. (a) If z* does not lie within the tolerance specifications at Tor
   (b) If some other z is found to be crossing zero from the new polynomial constructed from this reduced interval, go to 3 to re-evaluate with the new polynomial. Else
6. Reinitialize the problem at f for the new set of equations describing the system.
   k=k+l, go to 1.

Solving the problem at the point of discontinuity helps to locate it accurately. Note that 5(a) makes sure that the time of occurrence of the discontinuity is determined within a specified tolerance and 5(b) ensures that the detected discontinuity is indeed the earliest one.

Modeling Explicit Discontinuities

The algorithm described in this section satisfies the requirements for dealing efficiently with the three important aspects of nonsmooth dynamic simulation: detection of the earliest state event, location of that event within specified error bounds and reinitialization of the problem for the next interval. The reinitialization step has not been dealt with in detail here, the systematic treatment developed by Pantelides (1988b) could be easily adapted to the reinitialization step in this approach.

Nonsmooth dynamic simulation involves transition from one state of the system to another or in other words, a change from one set of equations describing the process to another. To initialize the problem, smooth inequality constraints can be added to the LP formulation as described in Bullard and Biegler (1991) to determine the set of equations which are initially active. Once this is known, we need to keep track of the logical conditions triggering a change from this model to a different one. This is where the switching functions come into play. We know that an event has been triggered when one or more of these switching functions crosses zero. The new set of equations associated with that particular switching function takes over and the simulation proceeds.

Explicit discontinuities occurring in chemical process simulation could be classified as history independent and dependent discontinuities. History independent discontinuities are the ones in which the current state of the system is independent of the prior states. The transitions between the system states are allowed in any direction and are triggered by the same logical condition in all directions. Examples of these are simple pressure valves (opening and closing triggered by a certain fixed pressure). On the other hand, the current state of the system described by a history dependent discontinuity is dependent on the prior states. They are tougher to solve when compared to their history independent counterparts. Let us consider an example to illustrate the method.

Example : Three Tanks with Hysteresis Valves

Here we consider a problem with 3 tanks interlinked with 4 hysteresis valves as shown in figure 3.

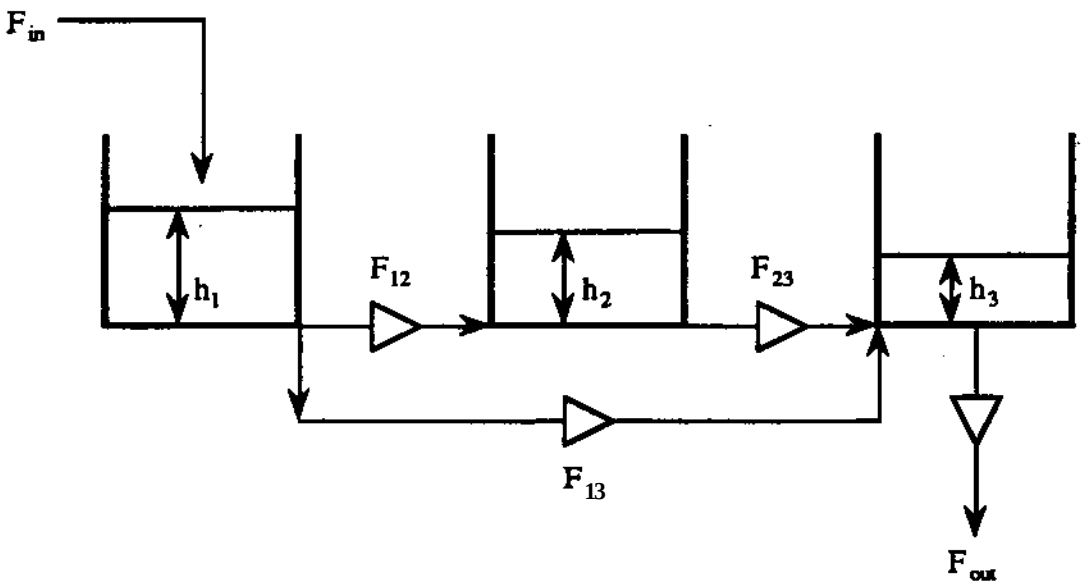


Figure 3: Three tanks problem

Description of a hysteresis valve

Consider the valve *ij* fitted between tanks / and *j*. Let h_i and h_j be the liquid levels in these tanks. The hysteresis valve has two positions: closed (flow=0) and open (flow=function($h_i - h_j$)). The logical condition triggering a change in these two states are

1. Closed to open if $h_i - h_j \geq h^{\wedge}$ (8a)
2. Open to closed if $h_L - h_j < h^{\downarrow}_{y}$ (8b)

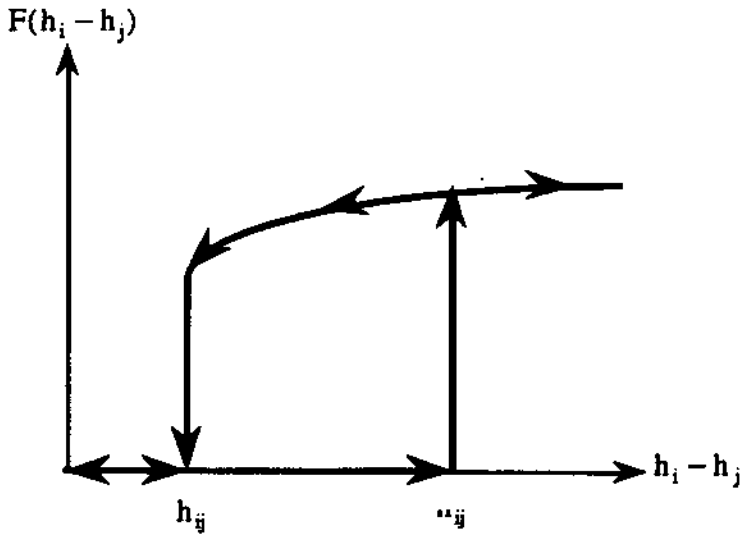


Figure 4: Hysteresis curve showing flow in the pipes

When h_{ij}^{L} and *hfj* are set to different values, graphically, the valve behavior can be represented using a hysteresis curve (Figure 4). Two sets of equations are to be looked at when dealing with each valve: one which is valid when the valve is open and the other when the valve is closed. Let $y_{ij}^{?}$ be a flag for the current position of the valve *ij.*

Consider the case when the valve is closed ($y_{ij}^{?} = 0$). From (8a) when $A_i - h_j \pounds h_{ij}''$, the valve opens. So, the corresponding switching function will be

$$z_{ij}^{closed} = 1 * - \wedge - \wedge \qquad \text{(9a)}$$

The flow through the pipe $Ff''*=0$ (9b)

Similarly, when the valve is open ($y_{ij}^{?} = 7$), the switching function will be

$$z_{ij}^{open} = h_i - h_j - h_{ij}^{L} \qquad \text{(10a)}$$

The relief flow in this case is $F^{\circ}f^{\epsilon n} = a_{tJ}\overline{\wedge - h_j}$ (10b)

We want to incorporate (9) when the valve is closed and (10) when the valve is open. The final set of equations will be

$$\begin{aligned} z_{ij} &= (h_i - h_j - h_{ij}^{U})(1 - y_{ij}^{0}) + (h_i - h_j - h_{ij}^{L})y_{ij}^{0} \\ F_{ij} &= y_{ij}^{0}(\alpha_{ij}\sqrt{h_i - h_j}) \end{aligned} \qquad \text{(11)}$$

Since y_{ij}^{*} represents the current state of the valve (which is known), (11) reduces to (9) when the valve is closed and to (10) when the valve is open. A switch in valve position is detected by a zero crossing of *z* and the value of $y_{ij}^{?}$ is changed when the integration continues from the point of discontinuity.

The hysteresis nature of the valves causes the history of the system to play an important part in determining the position of any of these valves at a given time. Therefore, it is simpler to model this system using the procedure described before. The position of each valve being monitored by (11), the dynamics of the system of tanks and valves in figure 5 can be formulated as in (12).

$$
\begin{aligned}
A_1 \frac{dh_1}{dt} &= F_{01} - F_{12} - F_{13} \\
A_2 \frac{dh_2}{dt} &= F_{12} - F_{23} \\
A_3 \frac{dh_3}{dt} &= F_{13} + F_{23} - F_{out} \\
F_{ij} &= y_{ij}^0 (\alpha_{ij} \sqrt{h_i - h_j}) \\
z_{ij} &= (h_i - h_j - h_{ij}^U)(1 - y_{ij}^0) + (h_i - h_j - h_{ij}^L) y_{ij}^0 \\
(h_4 &= 0)
\end{aligned}
\quad \left. \begin{array}{l} \\ \\ \\ \\ \\ \end{array} \right\} ij = 12,13,23,34 \qquad (12)
$$

The variables /* are the liquid levels in the tanks, *F* the flowrates in the pipes and *a* the valve constants. This system was simulated using our UP based algorithm. The values for the valve constants and the breakpoints are given in Table 2. The initial conditions correspond to all tanks empty and all valves closed. The constants were chosen to demonstrate the opening and closing of all the valves within a shon time. Figure 5 shows the levels of liquid in the three tanks. The driving force, the difference in liquids levels in the tanks is plotted in figure 6. The flowrate in the pipes is shown in figure 7.

ij	hi	ul) hij	aij
12	30	75	12
13	60	100	12
23	30	50	15
3 out	40	50	10

Table 2: Parameters for the three tanks problem

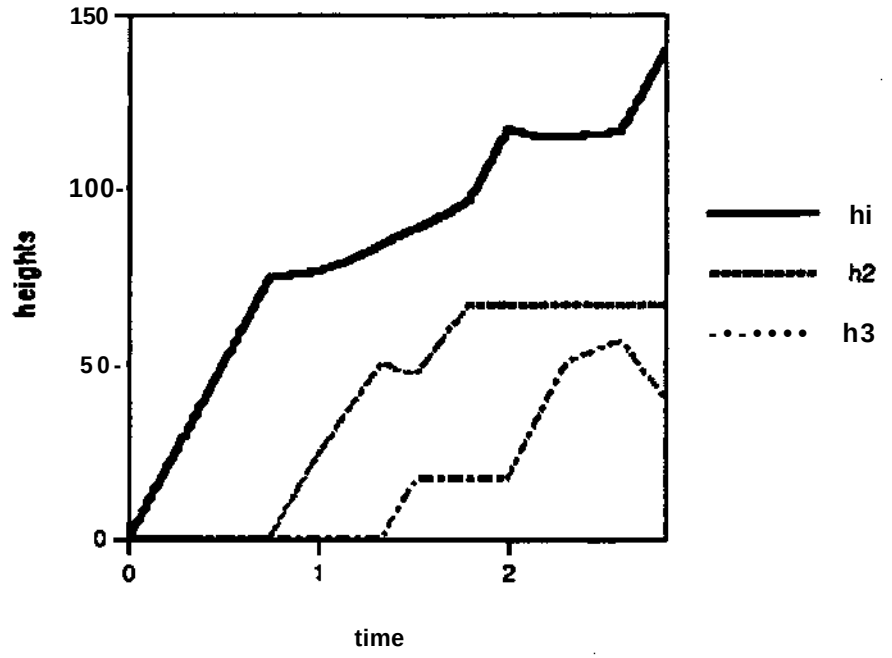


Figure 5: Height of liquid in the tanks

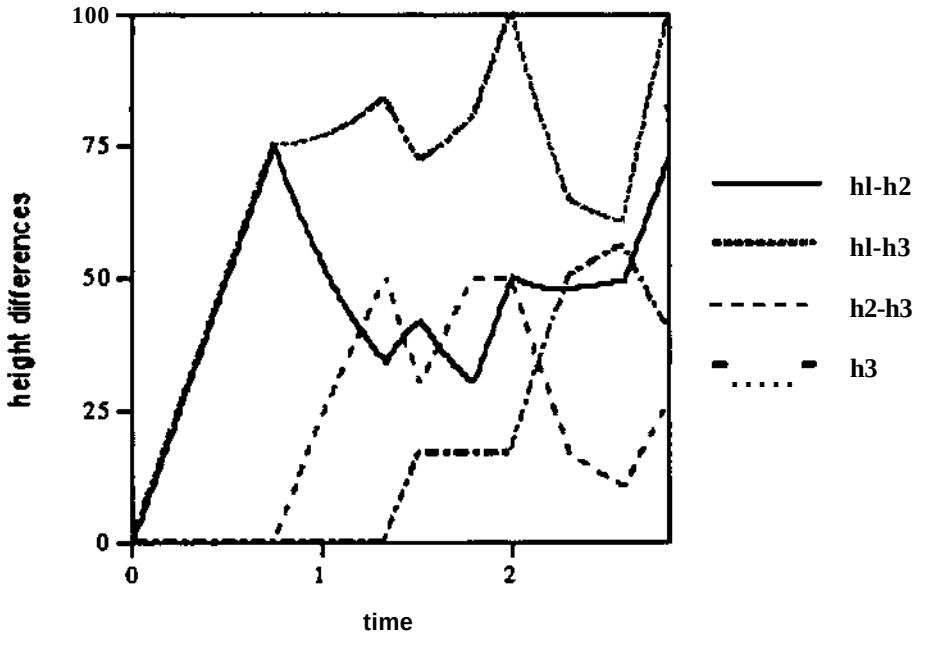


Figure 6: The difference in levels of liquid in the tanks

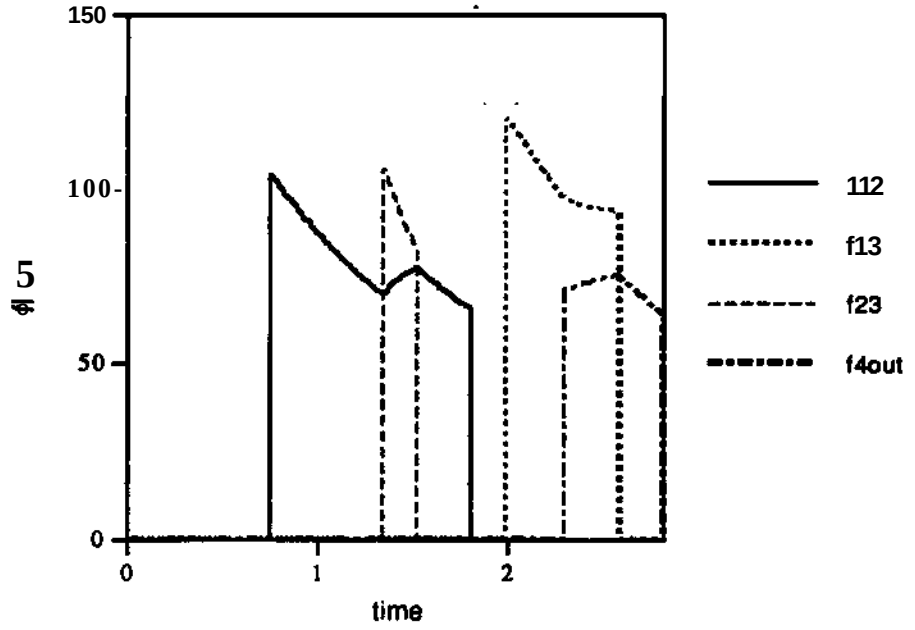


Figure 7: Flowrates in the pipes

5. Implicit Discontinuities

In this section we consider a class of implicitly discontinuous problems encountered often in chemical process simulation: calculation of phase equilibrium. Consider a simple isothermal flash (figure 8) to highlight some of the basic issues which are relevant in this problem.

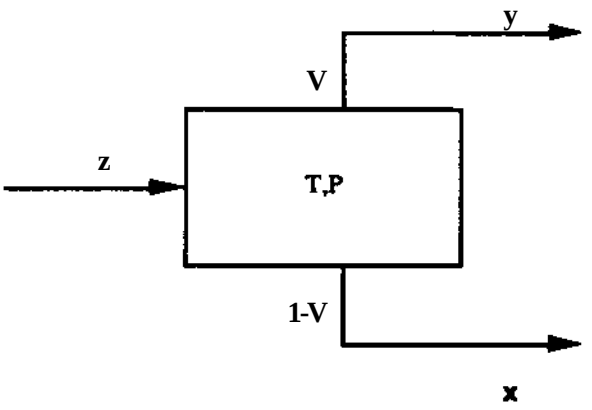


Figure 8: Isothermal flash operation

A feed stream of composition z enters the flash column at a specified temperature and pressure. The products are a vapor phase stream of composition y and a fractional flowrate V and/or a liquid phase stream of composition x. Different sets of equations are valid depending on the number of phases present at equilibrium. For example, the equilibrium equations are generally not valid in either of the single phase regions. The combined set of equations is an implicitly nonsmooth problem with well defined smooth regions. Thus, one of the basic problems associated with phase equilibrium calculations, especially in an equation oriented simulation environment, is that the number of phases is not known *a priori.*

Among the different approaches suggested to tackle this problem, the classical "sequential modular" approach is the most familiar one (Boston and Britt, 1978; Nelson, 1987). In general, these methods calculate the bubble and dew points (in the case of two phase calculations) and then determine the number of phases at equilibrium. The set of equations applicable to that particular phase behavior is then solved. Such a procedure based approach, however, can be difficult to incorporate in an equation oriented simulation environment.

Phase equilibrium problems are an important class of problems which can be addressed using our LP-based method. Bullard and Biegler (1993) applied a penalty based extension of their LP formulation, P-SONATA, to deal with two phase vapor liquid equilibrium (VLE) problems. The basic idea of their formulation was to introduce a 'pseudo* pressure P^and use it to solve the equilibrium expression in all the regions. P_P is allowed to differ from the specified pressure P_s in the phase equilibrium expression if P_s goes above the bubble or below the dew point pressures and the equilibrium expression is relaxed in the one phase regions. A consistent set of equations is thus obtained in both one and two phase regions by adding a penalty term to the objective to account for differences from equilibrium. The pressure relaxation formulation gave promising results for steady state two phase systems including isothermal flash problems with ideal and nonideal phase equilibrium relations and limiting distillation cases. However, it would be difficult to extend such a pressure relaxation formulation to a multiphase system. In the following section we introduce a new formulation to deal with multiphase systems as well.

Formulationformultiphasesystems

The following formulation is based on the idea that if two phases do not coexist at equilibrium, then the corresponding equilibrium expression could be relaxed. This could be done by relaxing the equilibrium constant itself, rather than dealing with pressure as in Bullard and Biegler (1993). NLP (13) is formulated based on such a relaxation. Thus the

new formulation is not limited to VLE systems and could be used for multiphase systems as well.

$$
\begin{aligned}
&\min \sum_p \delta_p \\
&\text{s.t.} \quad F - \sum_p M_p = 0 \\
&\qquad Fz_i - \sum_p M_p x_i^p = 0 \qquad i = 1, n_e \\
&\qquad x_i^{p*} - \gamma_p K_i^p x_i^p = 0 \qquad p = 1, n_p,\ p \neq p^* \\
&\qquad\qquad\qquad\qquad\qquad\qquad i = 1, n_c \\
&\qquad \sum_i x_i^{p*} - \sum_i x_i^p = 0 \qquad p = 1, n_p,\ p \neq p_{ref} \\
&\qquad \delta_p \geq \gamma_p - 1 \qquad p = 1, n_p,\ p \neq p^* \qquad (13) \\
&\qquad \delta_p \geq 1 - \gamma_p \qquad p = 1, n_p,\ p \neq p^* \\
&\qquad 0 \leq x_i^p \leq 1 \\
&\qquad 0 \leq M^p \leq F
\end{aligned}
$$

Mp is the molar flowrate of phase p, xf is the moie fraction of component i in phase p and p_{ref} is a reference phase based on which the equilibrium expressions are defined. $\bar{K}_i^p$ is a 'pseudo' equilibrium constant, which is defined as $\bar{K}_i^p = Y_p K_i^*$, where K_i^* is the equilibrium constant computed at the specified temperature and pressure.

The approach we propose here is related to a Gibbs free energy minimization formulation. In the Appendix we show that the Kuhn-Tucker conditions of the Gibbs minimization problem are equivalent to the solution of (13) for a multiphase system. The relationship of Y_p at the solution to the existence of phases is also given in the Appendix. For a two-phase flash, (13) reduces to (14).

$$
\begin{aligned}
&\min S \\
&\text{s.t. } F\text{-}L\text{-}V = O \\
&\quad z_i F\text{-}x_i L\text{-}y_i V = O \qquad i = l,n \\
&\quad y_i\text{-}yK_i(P,T,x)x_i = O \qquad i = \backslash,n \\
&\quad \sum_i y_i - \sum_i x_i = 0 \\
&\quad 8 \geq y\text{-}\backslash \\
&\quad \delta \geq 1 - \gamma \\
&\quad \delta \geq 0 \\
&\quad 0 \leq x_i, y_i \leq 1 \\
&\quad 0 \leq L, V \leq F
\end{aligned}
\tag{14}
$$

In the two phase region, $y=\backslash$ and the equilibrium expression is satisfied. In the single phase regions, y differs from 1 and the equilibrium expression is relaxed. In the Appendix we show that $y > 1$ in the single phase liquid region and $y < 1$ in the single phase vapor region.

Recently Swaney and Kendlbacher(1994) expressed the complementarity condition equivalently as

$$\sum_i x?\text{-}J + v^P = O \tag{15a}$$

where the slack $y/^p$ and the phase amount M^p (e.g., V or LP) are complementary.

$$M^p \geq 0, \quad \backslash ff^p \geq O_t \quad M^p y/^p = 0 \tag{15b}$$

Such a formulation could also be solved reliably using iterated LP.

Examples

1. Dynamic Simulation of a Two-Phase Flash Tank Covering Three Regimes of Operation
In this example we consider an n-butane, n-pentane, n-hexane system where the ideal flash unit is modeled using the Antoine equation. A constant feed of molefractions 0.3,0.3 and 0.4 for the respective components is supplied to the unit. The flash is carried out at a pressure of 7600 mm Hg. The temperature is varied linearly from r,=385K to $r_/$=420K from time $/_0$=0 to $/_7$=10. It should be noted that the dew and bubble points of the feed lie in this range. The motivation behind choosing this example problem is to see whether the algorithm could capture the transition from a single phase region to a two phase region and vice versa as the temperature was varied.

Mind

$$\frac{dM_i}{dt} = \sum_i f_i - V - L$$

$$H = M,x$$

$$y = yK(T)x$$

$$\sum_i \quad _i = 0 \qquad (16)$$

$$T = T(t)$$

$$-\delta \le \gamma - 1 \le \delta$$

$$0 \le x_i, y_i \le 1$$

The flash unit was modeled as in (16) where L is the liquid flowrate, V is the vapor flowrate, H is the component molar holdup, x and y are the component molfractions in the liquid and vapor phases and M, is the total liquid holdup. All variables in (16) are functions of time. Vapor holdup is neglected in this formulation. The holdups were discretized using implicit Euler and the corresponding set of equations were solved using iterated LP at time steps of 0.1. Figure 9 shows the liquid and vapor stream flowrates with time.

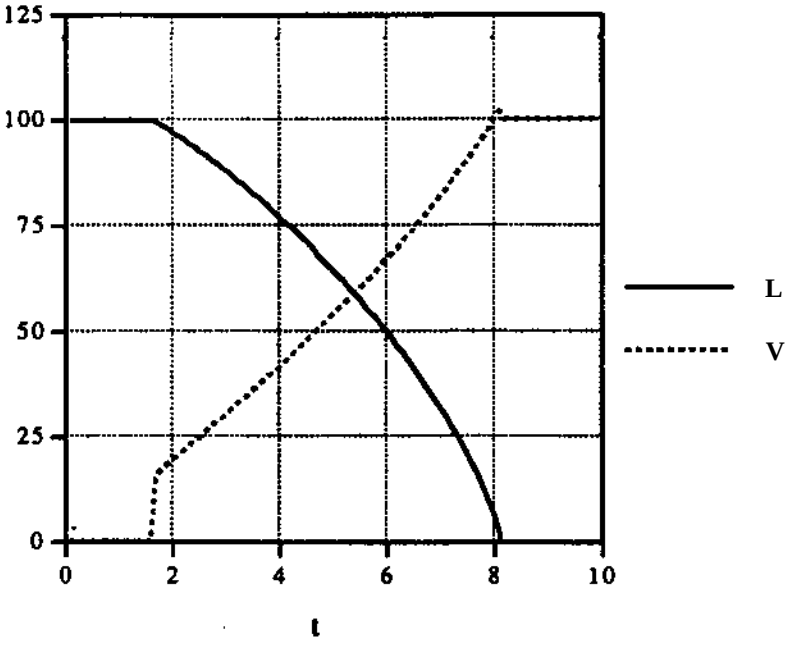


Figure 9: Liquid and vapor flowrates in example 1

The plot indicates that the algorithm handled the transition from the single phase liquid region to the two phase region and then to the single phase vapor region well. The average number of iterations for solving the problems per time step were 4.1. In the single phase regions, solution of the discretized equations on an average took 3 iterations.

2. *Dynamic Simulation of a Nonideal Three Phase Flash*

This example examines a benzene - isopropanol - water system considered in steady state by Pham and Doherty (1990). For different temperature - pressure conditions this system could exist as a LLV (liquid-liquid-vapor) or LL (liquid-liquid) or LV (liquid-vapor) or just as a single phase liquid or single phase vapor. To accommodate this possibility, we posed the formulation (17) for the dynamic simulation of this system.

$$
\begin{aligned}
& Min 8^{J} + 8^{IJ} \\
& s.t. \quad \frac{dH}{dt} = f - Vy - L^{I}x^{I} - L^{II}x^{II} \\
& \quad \frac{d}{dt}(M_i^{I} + M_i^{II}) \qquad f_i - V - L^{I} - L^{II} \\
& \quad H = M\backslash x^{I} + \mathrm{M/'x"} \\
& \quad L^{I} = \alpha^{I}\sqrt{M_i^{I}} \\
& \quad L^{II} = \alpha^{II}\sqrt{M_i^{II}} \\
& \quad y = y^{I}K^{I}x^{I} \\
& \quad y = Y"K"x" \\
& \quad \sum_i y_i - \sum_i x_i^{I} = 0 \\
& \quad \sum_i y_i - \sum_i x_i^{II} = 0 \\
& \quad T = T(t) \qquad\qquad (17) \\
& \quad -\delta^{I} \le \gamma^{I} - 1 \le \delta^{I} \\
& \quad -\delta^{II} \le \gamma^{II} - 1 \le \delta^{II} \\
& \quad 0 \le x_i^{I}, x_i^{II}, y_i \le 1
\end{aligned}
$$

As in example 1, a constant feed of molefractions 0.5, 0.08 and 0.42 of benzene, isopropanol and water respectively were fed into the flash tank. At constant pressure of 1 atm, the temperature is decreased linearly from 70°C to 68°C. The initial conditions correspond to steady state values at 70°C. The activity coefficients for this system were calculated using the regular solution model. Figure 10 plots the flowrates of the two liquid and the vapor streams with time. Liquid phases 1 and 2 correspond to the water rich and

benzene rich phases respectively. As shown in the figure, initial conditions correspond to a liquid-vapor system. A second liquid phase appears later on and the system enters the three phase *ll* - L^{u} - *V* region. Upon further decrease of temperature, the vapor phase vanishes altogether, resulting in a two phase liquid region. The algorithm was thus successful in simulating the three phase LLV and the two phase LV and LL regions with a single formulation. An average of 3.4 iterations were required to solve the descretized equations at each time step.

Both examples show that the penalty based iterative LP strategy could be a strong tool for solving dynamic problems involving phase transitions, without having to compute the dew and bubble points and then determining the number of phases at any time. The transitions between phase combinations were handled efficiently by a single consistent formulation. Recently, a lot of research is being done on developing global algorithms for solving phase equilibrium problems of this type. Beyond initialization, however, applying a global optimization algorithm to dynamic phase equilibrium problems could be expensive and is often unnecessary. The reason here is that the solution at a particular time step gives good starting points for the problem at the next time step. A formulation such as the one presented in this paper has its relevance in this context. It can efficiently tackle the problem of phase change and at the same time is far less expensive than a global optimization approach.

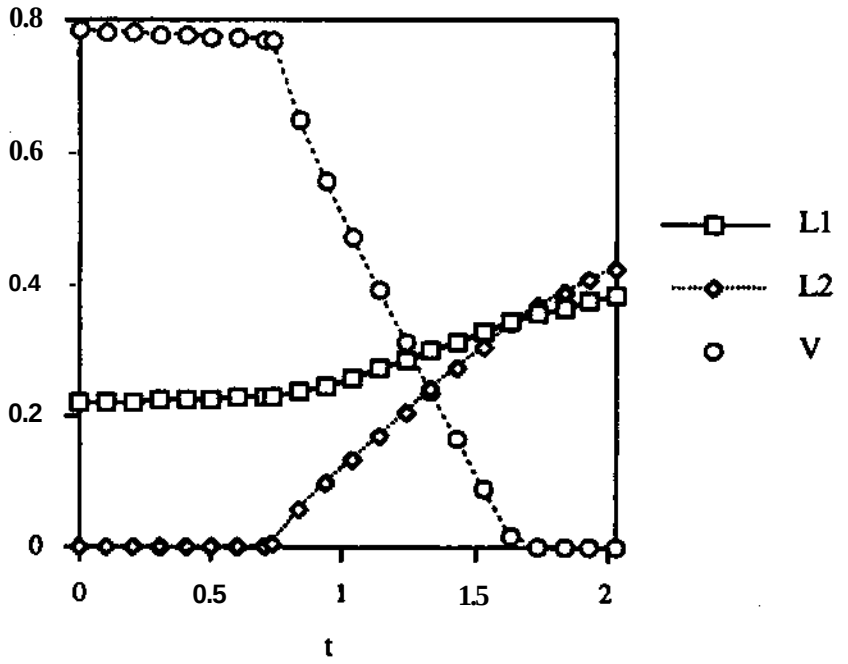


Figure 10: Flowrates of the two liquid and vapor streams in example 2

6. Conclusions

A previously developed LP-based method has been extended to a variety of dynamic simulation problems. The method has been refined with the addition of a new descent strategy which combines line search with a trust region approach. The LP based method has been demonstrated to be an effective way for the imposition of bounds on variables in dynamic simulation problems. An improved method for the treatment of discontinuities occurring in nonsmooth dynamic simulation problems has been developed. The method ensures that any event which has occurred within the time interval in consideration is detected and if more than one event occurs, the detected one is indeed the earliest one. In the case of nonexistence of solution across a discontinuity, a penalty term introduced in our approach makes it less likely to fail when compared to the presently used discontinuity locking methods. A specific class of process simulation problems, phase equilibrium calculations has been looked at as a special case. A new formulation for solving multiphase equilibrium problems has been presented. A penalty term introduced in the objective takes care of the appearance and disappearance of phases. Example problems have been solved to demonstrate the feasibility of the approach.

With the results from small problems being encouraging, future work will focus on solving large scale problems. An LP interface to a sparse DAE solver like SDASSL is expected to be an efficient way of dealing with large nonsmooth dynamic simulation problems. The LP will provide an efficient way for dealing with variable bounds and conditionals whereas the DAE solver will automatically generate the polynomial functions for the detection of discontinuities.

References

Armijo, L., Minimization of Functions Having Lipschitz Continuous First Partial Derivatives, *Pacific J. Math.*, 16, 1 (1966)

Barrodale, I. and F.D.K. Roberts, An Efficient Algorithm for Discrete Li Linear Approximations with Linear Constraints, *S1AM J. Numer. Anal,* 15, 3 (1978)

Barton, P.I., Ph.D. Dissertation, Imperial College of Science, Technology and Medicine, London, UK (1992)

Biegler, L. T., J. J. Damiano and G. E. Blau, Nonlinear Parameter Estimation: A Case Study Comparison, *AIChE J.*, 32(1), 29 (1986)

Blau, G. E., L. Kirby and M. Marks, An Industrial Kinteics Problem for Testing Nonlinear Parameter Estimation Algorithms, Process Math Modeling Department, The Dow Chemical Company (1981)

Boston, J. F. and H. L. Britt, A Radically Different Formulation and Solution of the Single-Stage Flash Problem, *Computers Chem. Engng.* 2, 109 (1978)

Bullard, L. G. and L. T. Biegler, Iterative Linear Programming Strategies for Constrained Simulation, *Computers Chem. Engng.*, 15, 4 (1991)

Billiard, L.G., Ph.D. Dissertation, Carnegie Mellon University, Pittsburgh, PA (1991)

Billiard, L. G. and L. T. Biegler, Iterated Linear Programming Strategies for Nonsmooth Simulation: a Penalty Based Method for Vapor-Liquid Equilibrium Applications. *Computers Chem. Engng.*, 17, 95 (1993)

Chamberlein, R. M., M. J. D. Powell, C. Lemerchal and H. C. H. C. Pedersen, The watchdog method for forcing convergence in algorithms for constrained optimization, *Math. Prog.*, 16,1 (1982)

Dennis, J.E. and R.B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Prentice-Hall, INc, Englewood Cliffs, NJ (1983)

Duff, I.S., J. Nocedal and J.K. Reid, The Use of Linear Programming for the Solution of Sparse Sets of Nonlinear Equations, *SJAM J. Sci. Statist. Compute* 8, 99 (1987)

Fletcher, R., Practical Methods of Optimization, Wiley, New York (1987)

Gill, P. E., W. Murray, M. A. Saunders, and M. H. Wright, User's Guide for SOL/QPSOL: A Fortran Package for Quadratic Programming, Technical Report SOL 83-7, Department of Operations Research, Stanford University, Stanford, CA (1983)

Joglekar, G.S. and G.V. Reklaitis, A Simulator for Batch and Semi-Continuous Processes, *Computers Chem. Engng.*, 8, 315 (1984)

Marquardt, W., Dynamic Process Simulation - Recent Progress and Future Challenges. In: *Chemical Process Control* (Y. Arkun and W.H. Ray, eds.), CACHE-AIChE Publications, 131 (1991)

Naess, L., A. Mjaavatten, and J. Li, Using Dynamic Process Simulation from Conception to Normal Operation of Process Plants, *Computers Chem. Engng.*, 16, SI 19 (1992)

Nelson P. A., Rapid Phase Determination in Multiple-Phase Flash Calculations. *Computers Chem. Engng.* 11,6 (1987)

Nordsieck, A., On the Numerical Integration of Ordinary Differential Equations, *Math. Comp.*, 16, 22 (1962)

Pantelides, C. C, SpeedUp- Recent Advances in Process Simulation, *Computers Chem. Engng.*, 12, 745 (1988a)

Pantelides, C. C, The Consistent Initialization of Differential-Algebraic Systems, *SJAM J. Sci. Stat. Comput.*, 9(2), 213 (1988b)

Pantelides, C. C. and P. I. Barton, Equation-Oriented Dynamic Simulation: Current Status and Future Perspectives, *Computers chem. Engng.* 17, S263 (1993)

Park, T. and P. I. Barton, A New Algorithm for the Accurate and Efficient Location of State Events, *Special Topical Conference on Industrial Chemical Technology, AJChE Annual Meeting,* St. Louis, MO (1993)

Perkins, J.D., Survey of Existing Systems for the Dynamic Simulation of Industrial Processes, *Modeling, Identification and Control,* 7, 71 (1986)

Pham, H. H. and M. F. Doherty, Design and Synthesis of Heterogeneous Azeotropic Distillations - 1. Heterogeneous Phase Diagrams, *Chem. Engng Sci.,* 45, 7 (1990)

Smith, G. J. and W. Morton, Dynamic Simulation Using an Equation-Oriented Flowsheeting Package, *Computers chem. Engng.* 12,469 (1988)

Swaney, R. E. and T. Kendlbacher, Robust Solution of Phase Equilibrium Calculations, *AJChE Annual Meeting,* San Francisco, CA (1994)

Zhang, J., N. Kim and L. Lasdon, An Improved Successive Linear Programming Algorithm, *Mgmt ScL,* 31, 10 (1985)

Appendix : Multiphase formulation

We show that the solution of the formulation (16) for multiphase systems would satisfy the Kuhn-Tucker conditions of a Gibbs free energy minimization formulation.

For an *n* component, *p* phase mixture, the Gibbs minimization formulation would be

$$\begin{aligned} \text{Min} \quad & G = \underset{p}{\text{XX}}\text{«r}(4\text{G}/ + RT\ln f?) \\ & \sum n? = n! \qquad i = l, n_c \\ \text{s.t} \quad & \sum_{i}{}' > {}_i^p \geq 0 \qquad p = 1, n_p \end{aligned} \tag{A.1}$$

where *nf* denotes the moles of component/ in phase *p*. Simplifying (A1),

$$\begin{aligned} \text{Min} \quad & G = \sum_i n_i^T \Delta G_i^f + \sum_p \sum_i n_i^p RT \ln f_i^p \\ & \sum n_i^p = n_i^T \qquad i = 1, n_c \\ \text{s.t} \quad & \\ & 5 > \text{\textasciicircum}0 \qquad p = l, n_p \end{aligned} \tag{A.2}$$

The Kuhn-Tucker conditions of the problem can be simplified to

$$RT\ln f_i^p + RT\sum_p\sum_i n_i^p \frac{\partial \ln f_i^p}{\partial n_i^p} + a_{t;} - p_p = 0 \qquad \text{i} = \text{l,n}_c,\ p = \text{\},n_p \qquad \text{(A.3a)}$$

$$\beta_p \sum_i n_i^p = 0 \qquad p = 1, n_p \qquad \text{(A.3b)}$$

where Oi and /5^ are the multipliers for the equalities and the inequalities respectively.

From the Gibbs-Duhem theorem, $\underset{7T}{]\pounds\pounds} nf \frac{\partial l_{,,} f_i^p}{\partial n_i^p} = 0$

Define $p_p = RT\ln r_p$ (A.4)

(A3a) reduces to $RT\ln\frac{f_i^p}{\Gamma_p}$ = a-, a constant for component *i* in all phases

or $\frac{\text{JLL}}{\Gamma_{\text{A>.}}} = \frac{\text{2i}}{\Gamma_{\text{P.}}}$ Vp_m,p_n (A.5)

If]^nf = A/' = 0, where A/^p is the molar flowrate of phase *p*, (A3b) =* $f\}_P$ > *0* and hence (A4) =» F/>> 1.

On the other hand if M^p > 0, (A3b) =^ jβ/> = 0 and hence (A4). => Tp = 1

ie Fp = 1 =^ phase *p* exists

Jp > 1 =* phase p does not exist (A.6)

Define a reference phase p_{ref}

(A5) can be written as $\frac{\Gamma_{p_{ref}}}{F_P}$ $p = 1, n_p,\ p \neq p_{ref}$ $i = hn_e$ (A.7)

But $/\,; = PxW$ (A.8)

(A7) simplifies to $\text{jcf}'' == \left(\frac{\Gamma_{p_{ref}}}{\Gamma_p}\right)\left(\frac{\phi_i^p}{\phi_i^{p_{ref}}}\right)x_i^p$ $p = 1, n_p,\ p \neq p_{ref}$ $i = 1, n_c$ (A.9)

$$x_i^{p_{ref}} = \gamma_p K_i^p x_i^p \qquad \text{(A.10)}$$

where $y_p = \frac{F_p}{F_{p_{ref}}}$, a constant (A. 11)

and $K_{ip} = \frac{x_{ip}}{x_{ip_{ref}}}$, the equilibrium constant for phases p and p_{ref}

(A10) could be written as $x_{ip} = \text{iff } JC_{ip}$ (A.12)

where $\bar{K}_{ip}$ *is apseudo equilibrium constant.* Note that $\bar{K}_{ip} = \hat{f}_{ip}$ when both phases p and p_{ref} coexist.

For the flash operation, $nf = Af \wedge jcf_f$ so we can express the flash problem using the penalty formulation as in (7), which will reduce to (8) for a two phase system (with the vapor phase as the reference phase).

The following shows the correlation of the values of yp to the existence of phases

(a) $yp < 1 \Rightarrow F_p > F_{p_{ref}} \Rightarrow F_p > 1 \Rightarrow$ phase/? does not exist

(b) $y_{p_2} > y_{p_1} \Rightarrow F_{p_2} > T_{p_1} \Rightarrow F_{p_2} > 1 \Rightarrow$ phase p_n does not exist.

(c) Any $y_p > 1 \Rightarrow F_p \wedge > F_p \Rightarrow F_p \wedge > 1 \Rightarrow$ phase p_{ref} does not exist.

For a two phase system this becomes:

y = 1 => both liquid and vapor phases exist

y> 1 => vapor phase does not exist

Y < 1 =» liquid phase does not exist