Adsorption 4, 337–344 (1998)
© 1998 Kluwer Academic Publishers. Manufactured in The Netherlands.

Comparison of Finite Difference Techniques for Simulating Pressure Swing Adsorption

YUJUN LIU, JAVIER DELGADO AND JAMES A. RITTER
Department of Chemical Engineering, Swearingen Engineering Center, University of South Carolina, Columbia, SC 29208
Ritter@sun.che.sc.edu

Abstract. Three different finite-difference routines were compared for solving the nonlinear, coupled, partial differential and algebraic equations that describe pressure swing adsorption processes. A successive substitution method (SS), a block LU decomposition procedure (BLUD), and the method of lines approach with adaptive time stepping (DASSL) were used to simulate and compare the computation times required to reach the periodic state for two different PSA systems: PSA-air drying and PSA-solvent vapor recovery. For both systems, the results showed that DASSL was nearly twice as fast as BLUD, whereas SS was nearly an order of magnitude slower than BLUD. DASSL and BLUD were also very robust and accurate, as nearly identical bed profiles were obtained from both methods under both transient and periodic state conditions.

Keywords: pressure swing adsorption, mathematical models, numerical simulation, finite difference, adaptive time stepping

Introduction

A rigorous pressure swing adsorption (PSA) model consists of a system of coupled, partial differential and algebraic equations, which necessarily require numerical solution (Ruthven et al., 1994). At present, finite difference is still the most widely used method in solving the PSA model equations because of its simplicity. In using finite differences, spatial and time derivatives are discretized, resulting in a set of algebraic equations. In much of the early PSA literature, the algebraic equations were solved by a successive substitution method. More recently, other solution methods have been developed, e.g., orthogonal collocation (Ruthven et al., 1994); however, studies comparing one technique to another have rarely been performed, with one exception (Hassan et al., 1987).

The simulation of some PSA systems also requires a great deal of computation time. Examples include simulations of PSA-air purification (AP) (Ritter and Yang, 1991) and PSA-solvent vapor recovery (SVR) (Liu and Ritter, 1996). For very strongly adsorbed species, e.g., the DMMP-activated carbon system used in the PSA-AP study (Ritter and Yang, 1991), it may take tens of thousands of cycles to reach the periodic state, and for typical solvent vapors (e.g., benzene), it may take thousands of cycles (Liu and Ritter, 1996). The large number of cycles required by these systems are in contrast to typical PSA processes for air drying, hydrogen purification or air separation, which generally require tens of cycles to reach the periodic state (Ruthven et al., 1994). Therefore, more efficient methods are required to solve the models that are used to simulate PSA processes.

One way of improving the efficiency of rigorous PSA simulations is to accelerate the convergence to the periodic state (Smith and Westerberg, 1992; Croft and LeVan, 1994). These acceleration techniques are quite suitable if the periodic state is the only information desired. However, these techniques necessarily give rise to a fictitious transient path to the periodic state. In other words, the number of cycles and the corresponding dependent variable profiles are artifacts of the acceleration technique and have no physical significance until the periodic state is reached. Yet, the transient behavior of a PSA process may be significantly different from its periodic state behavior, and process designers utilizing PSA models may necessarily need to know this information. Clearly, more efficient solution techniques that model both the transient and periodic states of the PSA process in "real time" are still highly desirable.

Therefore, the objective of this work is to introduce two very efficient numerical techniques that reduce the computation time and/or storage requirement associated with modeling computationally intensive PSA systems (e.g., PSA-AP and PSA-SVR). Three finite difference methods are compared: a traditional successive substitution (SS) method (Finlayson, 1980), a relatively new block LU decomposition (BLUD) procedure based on Newman's algorithm (Newman, 1991), and a relatively new method of lines approach with an adaptive time stepping code called DASSL (Davies, 1984; Schiesser, 1991; Brennan et al., 1996).

Mathematical Model

To investigate the efficiencies of the three methods, two PSA systems are selected: PSA-AD by activated alumina (Chihara and Suzuki, 1983) and PSA-SVR (benzene) by activated carbon (Liu and Ritter, 1996). The mathematical models for these systems are given in Table 1. Note that in these models the heat transfer between the bed and the environment is accounted for by an overall heat transfer coefficient, h.

A four-step PSA cycle (cocurrent pressurization, cocurrent high pressure feed, countercurrent blowdown and countercurrent product purge) is used to simulate the PSA-AD and PSA-SVR operations. Each process begins at the second step with clean beds; and the initial conditions for each step correspond to the end profiles of the previous step. Also, the purge gas of the third step comes directly from the step-two product gas of the other bed; therefore, the inlet purge gas has the same composition and temperature as the product gas from the other bed, at all times. Table 2 lists the parameters used in simulating these processes.

Solution of the Model Equations

The first step in solving the N model equations is to discretize the spatial coordinate into NJ nodal points and substitute the spatial derivatives using a finite difference approximation. For SS and BLUD, the time

Table 1. Model equations for the PSA-AD and PSA-SVR system.*

Total mass balance:	$\frac{\partial u}{dz} - \frac{1}{T}\frac{\partial T}{\partial t} + \frac{1}{P}\frac{\partial P}{\partial t} - \frac{u}{T}\frac{\partial T}{\partial z} + \frac{1-\varepsilon}{\varepsilon}\frac{RT}{P}\rho_s\frac{\partial q}{\partial t} = 0$
Component mass balance:	$\frac{\partial y}{dt} + u\frac{\partial y}{\partial z} + (1-y)\frac{1-\varepsilon}{\varepsilon}\frac{RT}{P}\rho_s\frac{\partial q}{\partial t} = 0$
Energy balance:	$(\rho_g C_{pg} + \frac{1-\varepsilon}{\varepsilon}\rho_s C_{ps})\frac{\partial T}{\partial t} + \rho_g C_{pg}\frac{\partial uT}{\partial z} + \Delta H\frac{1-\varepsilon}{\varepsilon}\rho_s\frac{\partial q}{\partial t} + \frac{2h}{\varepsilon r_b}(T - T_0) = 0$
LDF approximation:	$\frac{\partial q}{\partial t} = k_a(q^* - q)$

Adsorption isotherm PSA-AD:

$$q^* = q_s b_0 \frac{Py}{RT}\exp\left(\frac{-\Delta H}{R}\left(\frac{1}{T} - \frac{1}{303}\right)\right)$$

Adsorption isotherm PSA-SVR:

$$q^* = \frac{q_s bPy}{1 + bPy}, \quad b = \frac{b_0}{R\sqrt{T}}\exp\left(\frac{-\Delta H}{RT}\right)$$

Initial and boundary conditions:

Cycle step I (pressurization):		Cycle step II (feed):	
$t = 0$:	$y = y_{IV}\quad q = q_{IV}\quad T = T_{IV}$	$y = y_I\quad q = q_I\quad T = T_I$	for $0 \le z \le L$
$z = 0$:	$y = y_F\quad T = T_F$	$y = y_F\quad T = T_F\quad u = u_F$	for $t \ge 0$
$z = L$:	$u = 0$		for $t \ge 0$
Cycle step III (depressurization):		Cycle step IV (purge):	
$t = 0$:	$y = y_{II}\quad q = q_{II}\quad T = T_{II}$	$y = y_{III}\quad q = q_{III}\quad T = T_{III}$	for $0 \le z \le L$
$z = L$:	$u = 0$	$y(t) = y_{II}(t)\quad T(t) = T_{II}(t)\quad u = u_p$	for $t \ge 0$

*P = pressure (kPa); q = amount adsorbed (mol/kg); q^* = equilibrium amount adsorbed (mol/kg); R = gas constant (m^3/kPa)/(mol K); T = temperature (K), t = time (s); u = velocity (m/s); y = mole fraction in gas phase; z = axial position (m). Other variables are defined in the text or in Table 2.

Table 2. Bed characteristics, process conditions, and physical properties.

	PSA-AD	PSA-SVR
Radius, r_b (m)	0.1	0.027
Length, L (m)	1.0	0.29
Void fraction, ε	0.4	0.43
Density, ρ_s (kg/m^3)	1200	480
Heat capacity of adsorbent, C_{ps} (kJ/(kg K))	1.26	1.05
Feed flow rate, V_F (m^3 STP/s)	1.57×10^{-2}	4.16×10^{-5}
Feed mole fraction, y_F	0.00393	0.005
Feed temperature, T_F (K)	303	293
Feed pressure, P_H (kPa)	507	152
Purge pressure, P_L (kPa)	101	4.56
Purge to feed ratio, γ	2.0	1.5
Total cycle time, t_c (s)	1200	40
Time for each step, t_s (s)		
Steps 1 and 3:	60	10
Steps 2 and 4:	540	10
Ambient temperature, T_0 (K)	303	293
Gas phase density, ρ_g (kg/m^3)	1.2	1.308
Gas phase heat capacity, C_{pg} (kJ/(kg K))	1.0	1.006
Heat of adsorption, ΔH (kJ/mole)	−51.9	−43.5
Mass transfer coefficient, k_a (1/s)		
P_H:	2.78×10^{-4}	0.086
P_L:	1.39×10^{-3}	0.086
Heat transfer coefficient, h (kJ/(m^2 s K))	0.04	0.0314
q_s (mole/kg)	1.0	4.4
b_0 (m^3/mole)	7.57	3.88×10^{-8}

derivatives are also approximated using finite differences, using a fixed time interval. Implicit time stepping is usually used in discretizing the time derivatives. The resulting system of nonlinear algebraic equations is then solved. For DASSL, only the spatial derivatives need to be discretized.

Successive Substitution Method. In this method, the ith discretized governing equation at the jth spatial node point and at time t is written as

$$f_{ij}(v_{km}) = 0, \quad k = 1, \ldots, N; m = j-1, j, j+1 \tag{1}$$

where v_{km} represents the values of the kth unknown at the mth spatial nodal point. Note that the spatial derivatives are approximated using three nodal points and that Eq. (1) contains the values of the time-dependent variables at the previous time step, which are constant at time t. In SS, Eq. (1) is first manipulated in the form

$$v_{ij} = F_{ij}(v_{km}) \tag{2}$$

which yields the iteration formula

$$v_{ij}^{n+1} = F_{ij}^{n}(v_{km}). \tag{3}$$

When $|v_{ij}^{n+1} - v_{ij}^{n}| < e$, where e is a selected convergence criteria, the solutions are found. Then, the calculation can march on to the next time step. SS is convenient to program because no derivatives need to be calculated and no matrices need to be inverted. For a large system of equations, typical of PSA models, this method is preferred; however, it converges linearly and may suffer from convergence problems (Finlayson, 1980).

For complex PSA models, SS does not converge in most cases if Eq. (3) is used directly. Also, the direct use of Eq. (3) is restricted by the range of the initial guess values that can be used to approximate the solution. Finding the limits of the initial guess values may not be a trivial task, especially with limited process information. However, by considering the convergence theorem associated with SS, these problems can be avoided by selecting a proper converging factor β_{ij} to force the scheme to meet the convergence theorem criteria. For model equation i at nodal point j, the criteria that must be met is given by

$$\sum_{i=1}^{N}\sum_{m=j-1}^{j+1}\left|\beta_{ij}\frac{\partial F_{ij}}{\partial v_{ij}}\right| < 1, \quad i = 1, \ldots, N. \quad (4)$$

By making β_{ij} sufficiently small, the same N β_{ij}'s can be used at all of the nodal points.

The converging factor technique is relatively easy to apply, because the converging factor can be incorporated into the computer program, and the β_{ij}'s can be evaluated at the beginning of the simulation. Of course, occasionally, several trials may be needed to find the proper β_{ij}'s to make the scheme convergent throughout the simulation. The cost of this is an increase in the number of iterations required to converge, which results in an increase in the computation time. The extent of the increase in the computation time depends on the PSA process being modeled. For the processes studied here, the time increase makes SS much less efficient compared to the other two methods described.

Block LU Decomposition Method. In this approach, the solution to the PSA model equations is obtained by using a technique developed by Newman, which is tailored for block-tridiagonal systems (Newman, 1991). According to Newman's algorithm, the set of nonlinear equations resulting from discretization using finite differences in the spatial and temporal coordinates are first expanded in a Taylor series and then solved by the LU decomposition method using a Newton-Raphson procedure. Expanding each of the functions shown in Eq. (1) in a Taylor series about the trial values of each variable, followed by rearrangement, yields at the spatial nodal point j and the present time t

$$f_{ij}^{t} = \sum_{k=1}^{N}\left(A_{ik}v_{kj-1}^{t} + B_{ik}v_{kj}^{t} + D_{ik}v_{kj+1}^{t}\right) - G_{ij}^{t} = 0, \qquad i = 1, \ldots, N \quad (5)$$

where

$$A_{ik} = \left(\frac{\partial f_{ij}^{t}}{\partial v_{kj-1}^{t}}\right)^{0}, \quad B_{ik} = \left(\frac{\partial f_{ij}^{t}}{\partial v_{kj}^{t}}\right)^{0}, \quad D_{ik} = \left(\frac{\partial f_{ij}^{t}}{\partial v_{kj+1}^{t}}\right)^{0}, \quad (6)$$
$$G_{ij}^{t} = \left(-f_{ij}^{t} + \sum_{k=1}^{N}\sum_{m=j-1}^{j+1}\left(\frac{\partial f_{ij}^{t}}{\partial v_{km}^{t}}\right)v_{km}^{t}\right)^{0}.$$

The superscript 0 denotes that the expression is evaluated at previous trial values. If the functions in Eq. (1) (i.e., discretized model equations) are nonlinear, A, B and D become functions of v_{km}, in which k is one or more of the N variables, and m takes on one or more of the $j-1$, j, and $j+1$ values. For linear equations, A, B and D are constants or functions of only the independent variables such as position and time. At the boundaries, the equations are written similarly to Eq. (5), but with B, D and X replacing A, B and D, and corresponding to v_{kj}, v_{kj+1} and v_{kj+2} for $j = 1$. For $j = NJ$, Y, A and B are used corresponding to v_{kNJ-2}, v_{kNJ-1} and v_{kNJ}. Note that three-point forward and backward finite difference algorithms are used at $j = 1$ and $j = NJ$, respectively. In the present study, backward finite difference expressions are used to approximate the derivatives because of the stiff spatial gradients, i.e., only the values at node $j-1$ and j are used for approximating the derivatives in node j; therefore, $D_{ik} = 0$.

In Newman's algorithm, Eq. (5) and the boundary equations are written in the following block matrix form

$$\begin{bmatrix} B(1) & D(1) & X(1) & & & & \\ A(2) & B(2) & D(2) & & & & \\ & & \ddots & & & & \\ & & & A(j) & B(j) & D(j) & \\ & & & & & \ddots & \\ & & & & Y(NJ) & A(NJ) & B(NJ) \end{bmatrix} \times \begin{bmatrix} v(1) \\ v(2) \\ \vdots \\ v(j) \\ \vdots \\ v(NJ) \end{bmatrix} = \begin{bmatrix} G(1) \\ G(2) \\ \vdots \\ G(j) \\ \vdots \\ G(NJ) \end{bmatrix} \quad (7)$$

where $v(j) = [v_{1j}\ v_{2j}\ \cdots\ v_{Nj}]^T$ and $G(j) = [G_{1j}\ G_{2j}\ \cdots\ G_{Nj}]^T$. In Eq. (7), if $E(j)$ is used to represent any of the $Y(1)$, $A(j)$, $B(j)$, $D(j)$ and $X(NJ)$, then it becomes

$$E(j) = \begin{bmatrix} E_{11} & E_{12} & \cdots & E_{1i} & \cdots & E_{1N} \\ \vdots & \vdots & & \vdots & & \vdots \\ E_{j1} & E_{j2} & \cdots & E_{ji} & \cdots & E_{jN} \\ \vdots & \vdots & & \vdots & & \vdots \\ E_{N1} & E_{N2} & \cdots & E_{Ni} & \cdots & E_{NN} \end{bmatrix}_j . \qquad (8)$$

Equation (7) is solved by the LU decomposition method, which is more efficient than Gauss elimination (Barbosa Mota et al., 1997). Information on the LU decomposition can be found in most textbooks (e.g., Golub and Van Loan, 1989); details on block LU decomposition have been described by Fan and White (1991). Brief details are given below.

The procedure consists of two phases. In the first phase, let $\mathbf{M}$ represent the $NJ \times NJ$ block coefficient matrix containing $X(1), \ldots, A(j), B(j), D(j)$ and $Y(NJ)$; and $\mathbf{V}$ and $\mathbf{G}$, respectively, represent the NJ dimensional vectors containing the elements $v(j)$ and $G(j)$ in Eq. (7). Then, $\mathbf{M}$ is decomposed into a lower triangular block matrix $\mathbf{L}$ and a unit upper triangular matrix $\mathbf{U}(\mathbf{M} = \mathbf{LU})$, which replace the original matrix $\mathbf{M}$ in the computer program. The solution of this linear system takes place in the second phase, during which the triangular system $\mathbf{L}\delta = \mathbf{G}$ is solved for the intermediate vector δ by forward substitution, and $\mathbf{UV} = \delta$ is solved for $\mathbf{V}$ by backward substitution.

For a large number of spatial nodes, the coefficient matrix becomes very large and usually sparse. Also, if all the elements in the coefficient matrices in Eq. (7) are retained, the storage requirements become extremely large. By using Newman's algorithm, only the nonzero elements in the coefficient block matrix are utilized. They are used only once at a particular nodal point, and then replaced at the next point. Therefore, there is no need to store values for the matrix at every point. A significant savings on the computer storage is realized when the number of the unknowns and number of nodal points are large. Also, this approach converges quadratically, which is a great improvement over SS, which converges linearly. Furthermore, a subroutine called BAND(J) has been written by Newman (1991) for performing the LU decomposition, and backward and forward calculations. Users only need to provide the discretized model equations, and the corresponding coefficients to use BAND(J) without major modifications.

Method of Lines with DASSL. This method utilizes finite differences to discretize the derivatives in the spatial direction, which transforms the original set of partial differential equations into a system of mixed DAEs (Schiesser, 1991) of the form

$$\mathbf{f}(\mathbf{v}, \mathbf{v}', t) = \mathbf{0} \qquad (9)$$

with the initial conditions given by

$$\mathbf{f}(\mathbf{v}(0), \mathbf{v}'(0), 0) = \mathbf{0} \qquad (10)$$

Because of the stiff spatial gradients and moving fronts associated with PSA processes, a backward finite difference approximation scheme for the first-order spatial derivatives is used. The solution of the system of equations denoted by Eq. (9) is computed in two stages.

The first stage uses a fixed time step, finite difference scheme to compute a consistent set of initial conditions (variables and their time derivatives). For problems like in this study, in which the initial values of v_{ij} are given, DASSL takes a small implicit Euler step for its first step and uses a damped Newton iteration to solve the nonlinear system of equations to obtain the initial time derivatives. The second stage in DASSL uses this set of initial conditions as the starting point for the integration (Brennan et al., 1996).

DASSL is a sophisticated DAE solver that implements backward differentiation formulas of varying order for the time derivative v' in the DAE system. Basically, the algorithms used in DASSL are an extension of the numerical ODE methods originated by Gear (1971). A predictor formula is used to obtain the predicted solution, and to obtain the final solution a corrector formula is solved using some variant of Newton's method. However, instead of always using the first-order formula, DASSL approximates the time derivatives using a varying order (1 to 5) backward difference formula. On each step, DASSL chooses a new order and the time step size, based on the behavior of the solution. A large time step is used over regions where the dependent variables change slowly with time. Likewise, the time step size is reduced in regions where the variables change rapidly. In addition, DASSL has error control which permits the determination of accurate solutions. These features of the software result in significant time savings when computing the solution without compromising its accuracy. The algorithm and strategies used in DASSL have been detailed by Brennan et al. (1996) and Schiesser (1994). The FORTRAN source code for DASSL is available through the National Energy Software Center (Brennan et al., 1996). It is easy to use, and requires only a user subroutine (which defines the transformed model equations) and the convergence criteria as inputs to the main program.

Simulation Results and Discussion

The model equations for the two PSA systems were solved using the three finite difference methods described above. All of the simulations were carried out on a SunSparc Station 10/40. Great care was taken to ensure that input/output statements in the computer code were minimized and essentially the same for all three methods. This ensured that the computation times reported below were indeed a result of solving the system of equations and that comparisons were made on a fair basis.

Table 3 presents the CPU times that were obtained from simulating the two different PSA processes, using all three methods. Note that because SS took so much time, the simulations were not allowed to progress to the periodic state and only five cycles were run. From this data, a five-cycle-based average CPU time per cycle was determined for SS. This was justified since it was found that the CPU time used for each cycle did not change for the fixed time step methods (SS and BLUD). For comparison of SS with BLUD and DASSL, the average time per cycle is also given for both BLUD and DASSL; however, it was calculated from the total time it took to reach the periodic state divided by the number of cycles required.

Simulation of PSA-Air Drying. To simulate the PSA-AD system, 61 spatial nodes were used, along with a 0.25 s time step (SS and BLUD, only). The time step varied from 10^{-8} to 50 s for DASSL. For this case, it took each method 40 cycles to reach the periodic state. The magnitude of the constant time step, and the relatively few cycles to reach the periodic state were both characteristic of this rather conventional PSA system (Ruthven et al., 1994).

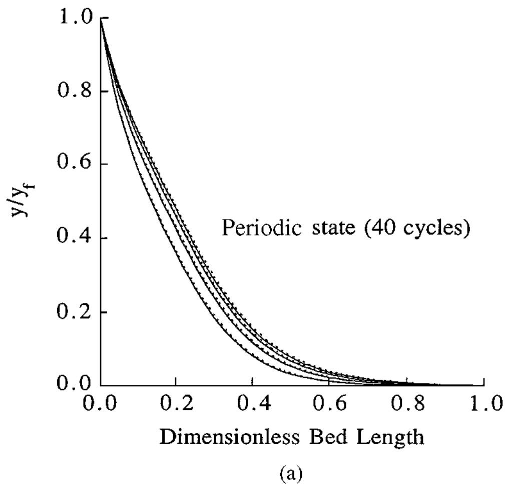
Table 3 shows that SS was two orders of magnitude slower compared to either BLUD or DASSL. This was due to the use of converging factors that were necessary to ensure convergence with SS (Finlayson, 1980). In addition, SS converged linearly, whereas both BLUD and DASSL (for the most part) converged quadratically. DASSL was found to be the fastest method, being nearly twice as fast as BLUD for the PSA-AD system. This was due to the variable time-stepping routine built into DASSL. It was interesting that the time step size varied by nearly ten orders of magnitude, indicating that this system was quite stiff at times. Clearly, this large variation in the time step was responsible for the significantly reduced CPU time.

The PSA-AD transient and periodic state bed profiles for the gas phase concentration and temperature are shown in Fig. 1 for both BLUD and DASSL. These profiles all correspond to a snapshot in time at the end of the adsorption step (i.e., step II of the PSA-AD cycle). Remarkably, all of the bed profiles obtained from BLUD and DASSL were nearly identical in every respect, whether at transient or periodic state conditions. Clearly, both BLUD and DASSL have the

Table 3. Comparison of the CPU time using the three finite difference methods for simulating PSA.

	PSA-AD			PSA-SVR		
Number of nodes		61			31	
Time step size[a] (s)		0.25[b]			0.01[c]	
Cycles to steady-state		40			3552	
Method	SS	BLUD	DASSL	SS	BLUD	DASSL
CPU time (min)	—	45	26	—	960	482
CPU time/cycle (s/cycle)[d]	4774	67	38	249	16	8

[a]Time step-sizes for SS and BLUD only.
[b]Time step-size for DASSL varied from 10^{-8} to 50 s.
[c]Time step size for DASSL varied from 10^{-6} to 1.5 s.
[d]CPU time for SS was based on the first five cycles.



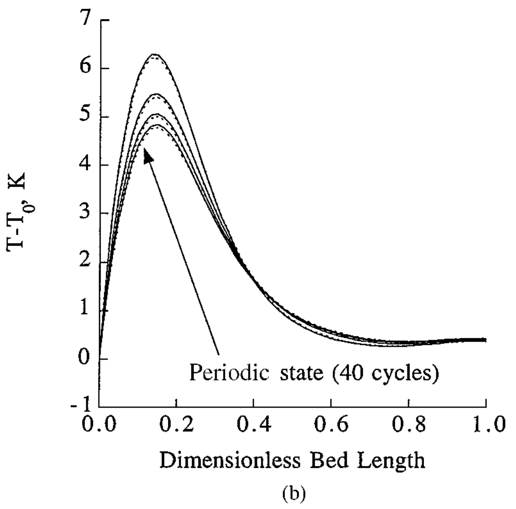
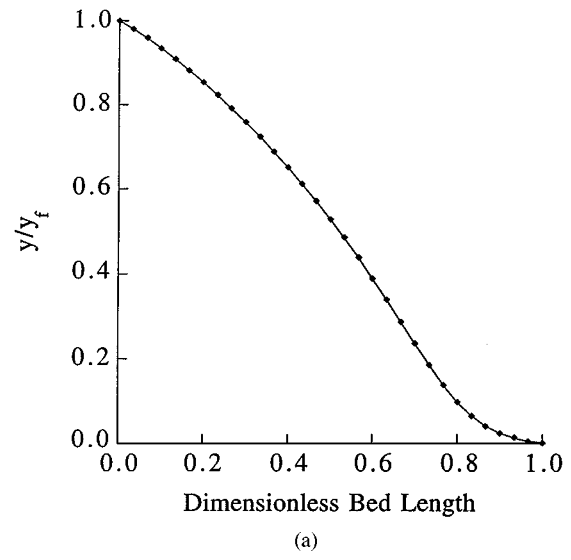


Figure 1. Transient and periodic state (a) concentration and (b) temperature profiles at the end of the adsorption step for the PSA-AD system. Solid and dotted lines depict results for DASSL and BLUD, respectively. Curves are for 10, 20, 30, and 40 cycles.



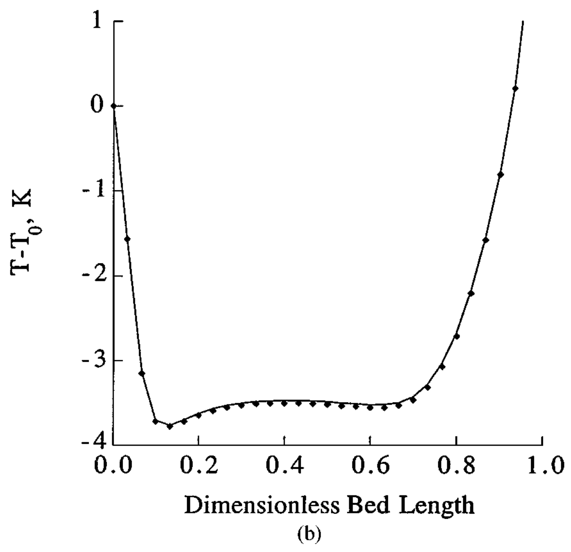


Figure 2. Periodic state (a) concentration and (b) temperature profiles at the end of the adsorption step for the PSA-SVR system. Solid lines and markers depict results for DASSL and BLUD, respectively.

capability to capture the nonlinear essence of the solution without sacrificing accuracy. The temperature profile shown in Fig. 1(b), being highly nonlinear, illustrates this point well. The fact that DASSL was twice as fast as BLUD makes it a very powerful method for simulating conventional PSA systems like PSA-AD. This remarkable agreement between the two methods also suggests that the numerical errors are either the same, which is highly unlikely because each method employs vastly different temporal step sizes, or small, which is more probable and indicates that both techniques are fairly robust and accurate.

Simulation of PSA-Solvent Vapor Recovery. To simulate the PSA-SVR system, 31 nodes were used with a 0.01 s time step (SS and BLUD, only). For this system, the DASSL time step varied from 10^{-6} to 1.5 s, and the approach to the periodic state was very slow, taking 3,552 cycles to reach it. The small magnitude of the constant time step and the slow approach to the periodic state were both very characteristic of PSA-SVR processes, mainly because of the tremendous adsorption capacity associated with these systems (Liu and Ritter, 1996).

Table 3 shows that BLUD and DASSL were each an order of magnitude faster than SS. Again, this was due to SS requiring the use of converging factors, and the fact that SS converges linearly. Similarly to the PSA-AD system results, DASSL was twice as fast as BLUD, because of the use of a variable time step. For this system, the time step size varied by six orders of magnitude, thereby allowing for efficient use of the computer, manifested as a significantly reduced CPU time to reach the periodic state. This range also indicated some degree of stiffness in the PSA-SVR system of equations, but apparently less than that exhibited by the PSA-AD system.

The PSA-SVR periodic state bed profiles for the gas phase concentration and temperature are shown in Fig. 2 for both BLUD and DASSL. As before, these profiles all correspond to a snapshot in time at the end of the adsorption step (i.e., step II of the PSA-SVR cycle). In all cases, excellent agreement between the two finite difference methods (BLUD and DASSL) was found. Note the steepness and nonlinearity of the temperature profile; yet, both methods were capable of capturing this highly nonlinear behavior associated with this unique and rather unconventional PSA system. Since DASSL was twice as fast as BLUD, it represents a very powerful method for simulating PSA systems that slowly approach the periodic state, like PSA-AP (Ritter and Yang, 1991) and PSA-SVR (Liu and Ritter, 1996). Also, this excellent agreement between the two methods suggests that the numerical errors are probably small (based on the same reason that is given for the PSA-AD study), which again indicates that both techniques are fairly robust and accurate.

Acknowledgments

The authors gratefully acknowledge financial support from the National Science Foundation under Grants CTS-9410630 and OSR-9108 772-004, and from the Westvaco Charleston Research Center.

References

Barbosa Mota, J.P., E. Saatdjian, D. Tondeur, and A.E. Rodrigues, "On the Numerical Solution of Partial Differential Equations with Two Spatial Scales," *Computers Chem. Engng.*, **21**, 387–397 (1997).

Brennan, K.E., S.L. Campbell, and L.R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, SIAM, Philadelphia, 1996.

Chihara, K. and M. Suzuki, "Simulation of Non-Isothermal Pressure Swing Adsorption," *J. Chem. Eng. Japan*, **16**, 53–60 (1983).

Croft, D.T. and M.D. LeVan, "Periodic States of Adsorption Cycles I. Direct Determination and Stability," *Chem. Eng. Sci.*, **49**, 1821–1829 (1994).

Davies, M.E., *Numerical Methods and Modeling for Chemical Engineers*, John Wiley and Sons, New York, 1984.

Fan, D. and R.E. White, "Modification of Newman's BAND(J) Subroutine to Multiregion Systems Containing Interior Boundaries: MBAND," *J. Electrochem. Soc.*, **138**, 1688–1691 (1991).

Finlayson, B.A., *Nonlinear Analysis in Chemical Engineering*, McGraw-Hill, New York, 1980.

Gear, C.W., "The Simultaneous Numerical Solution of Differential Algebraic Equations," *IEEE Trans. Circuit Theory*, CT-**18**, 89–95 (1971).

Golub, G.H. and C.F. Van Loan, *Matrix Computations*, 2nd edition, Johns Hopkins University Press, Baltimore, 1989.

Hassan, M.M., N.S. Raghavan, and D.M. Ruthven, "Numerical Simulation of a Pressure Swing Adsorption System: A Comparative Study of Finite Difference and Collocation Methods," *Can. J. Chem. Eng.*, **65**, 512–516 (1987).

Liu, Y. and J.A. Ritter, "Pressure Swing Adsorption-Solvent Vapor Recovery: Process Dynamics and Parametric Study," *Ind. Eng. Chem. Res.*, **35**, 2299–2312 (1996).

Newman, J.S., *Electrochemical Systems*, Prentice Hall, New York, 1991.

Ritter, J.A. and R.T. Yang, "Pressure Swing Adsorption: Experimental and Theoretical Study on Air Purification and Vapor Recovery," *Ind. Eng. Chem. Res.*, **30**, 1023–1032 (1991).

Ruthven, D.M., S. Farooq, and K.S. Knaebel, *Pressure Swing Adsorption*, VCH Publishers, Inc., New York, 1994.

Schiesser, W.E., *The Numerical Method of Lines. Integration of Partial Differential Equations*, Academic Press, New York, 1991.

Schiesser, W.E., *Computational Mathematics in Engineering and Applied Science: ODEs, DAEs, and PDEs*, CRC Press, Boca Raton, 1994.

Smith, O.J. and A.W. Westerberg, "Acceleration of Cyclic Steady State Convergence for Pressure Swing Adsorption Models," *Ind. Chem. Eng. Res.*, **31**, 1569–1573 (1992).