A numerical solution to enzyme emulsion liquid membrane reactor model for sequential bienzymatic reaction

Parimal Pal\textsuperscript{a}, Siddhartha Datta\textsuperscript{b} & Pinaki Bhattacharyya\textsuperscript{*}

\textsuperscript{a} Department of Chemical Engineering, Regional Engineering College, Durgapur, India
\textsuperscript{b} Department of Chemical Engineering, Jadavpur University, Calcutta 700 032, India

Received 18 April 2000; accepted 19 February 2001

Dynamic mathematical models of liquid membrane-immobilized multienzyme reaction systems involve a large number of algebraic, ordinary and nonlinear partial differential equations with different types of boundary conditions. Solutions of such model equations are often difficult and stand in the way of realistic modeling efforts in this field. In the present study, attempt has been made to numerically solve such model equations for a liquid membrane-immobilized sequential bienzymatic reaction system through development of a software (MEMBSOL). In the solution process, partial differential equations have been converted into ordinary differential equations by using a finite difference technique in which the spatial derivatives have been discretized while leaving the temporal derivatives unconverted. Through mathematical manipulations, use of fictitious points at the boundaries has been eliminated thereby making the solution-approach a general-purpose one. To integrate the differential equations, Runge-Kutta-Fehlberg method has been applied. An automatic step-size adjustment mechanism has been incorporated in the integration process to produce results with a reasonably desired level of accuracy. In the present computations, numerical solutions with 0.1 percent relative error have been obtained.

Use of enzymes is no longer limited to production of biochemicals and pharmaceuticals only, these biocatalysts have now moved into new niches - from wastewater treatment to biomedical analysis. When such enzymes are immobilized in emulsion liquid membranes, they constitute what is known as enzyme emulsion liquid membrane reactors (EELM) and these EELM reactors offer a well-reproducible and excellent technique for enzyme-based manufacture of biochemicals, pharmaceuticals etc. at a cost lower than that involved in their conventional enzyme-based manufacture. This becomes possible due to the fact that liquid membrane-immobilized expensive enzymes can be used repeatedly and continuously while maintaining their activity for a long time under the protective barrier of the membrane. Liquid membranes offer immunity to the entrapped enzymes from the adverse effects of pH, high substrate concentration and other chemicals.

For scale-up and design of these liquid membrane-based bioreactors, mathematical modeling is very essential. But only in few recent research works\textsuperscript{1,2}, such modeling has been taken up. However, these models deal with single enzyme systems only, while it is understood that multienzyme systems are very common in biochemical industries. The authors\textsuperscript{1} have recently developed a dynamic mathematical model to describe the transient behavior of a liquid membrane-immobilized sequential bienzymatic reaction system. One problem which seriously stands in the way of dynamic modeling of these bioreactors is the mathematical solution of the model equations, particularly those involved in multienzyme systems. Liquid membrane-immobilized sequential multienzyme reaction systems involve a number of physico-chemical phenomena like peri-emulsion mass transfer, carrier-mediated or carrier-independent diffusion of substrate and product through the membrane, interfacial complexation and decomplexation reactions, internal phase enzymatic reactions following modified Michaelis-Menten kinetics, interfacial phase and chemical equilibrium, interrelations between sequential enzymatic reactions etc. As a result, a realistic mathematical model attempting to represent such a system takes complicated form involving quite a number of algebraic, ordinary differential and nonlinear partial differential equations with different types of boundary conditions (Neumann, Dirichlet etc.). Analytical solutions of liquid membrane-immobilized multienzyme reaction systems are considered very difficult, particularly when the model equations involve both type-
I and type-II transport mechanisms. For liquid membrane-immobilized reaction systems, perturbation solutions have been obtained by some researchers\textsuperscript{4,5}. But in these model equations, only the simplest type (type-I) transport mechanism has been considered. Only in recent few papers\textsuperscript{1,2}, liquid membrane-immobilized enzymatic reaction systems have been modeled considering both type-I and type-II transport mechanisms. In some of these papers\textsuperscript{1}, the model equations have been solved numerically by a combination of orthogonal collocation technique\textsuperscript{6,7} and the Gear's algorithm in the subroutine DINVAG of IMSL library, whereas in the other solution attempt\textsuperscript{2}, numerical solutions of the model equations have been obtained through conversion of partial differential equations into ordinary differential equations by the technique of Method of Lines. However, in both these cases, model equations are for single enzyme systems only and there is no scope for adjustment in the integration step size so as to produce results with a desired level of accuracy. For multienzyme systems, the model equations present far more complicated pictures so far as their solutions are concerned.

In the available standard techniques of solution of partial differential equations, there is no general approach in eliminating the values of the dependent variables at the fictitious points at the boundaries. These techniques are, therefore, problem-specific and a new formulation is required whenever a new problem is encountered. No general solution technique for such systems has yet been developed. In the present work, an attempt has been made to develop a software for numerical solution of the model equations for liquid membrane-immobilized sequential bienzymatic reaction system. In this developed technique, approach of solution has been made general-purpose through adoption of mathematical manipulations which eliminate use of fictitious points at the boundaries. The software (MEMBSOL), with the incorporation of an automatic step-size adjustment mechanism in the integration process, is capable of producing results with a reasonably desired level of accuracy.

The Dynamic mathematical model developed\textsuperscript{3} for the liquid membrane-immobilized sequential bienzymatic reaction system,

\[ \text{n maltose} \xrightarrow{\alpha-\text{glucosidase}} \text{2n glucose} \]
\[ \text{glucose} \xrightarrow{\text{oxidase}} \text{2n gluconic acid}, \]

has been considered for developing the present software for numerical solution. The model equations for the above bienzymatic reaction system immobilized in emulsion liquid membrane take the forms as described in the subsequent section.

**Model Equations**

(A) Mass balance of substrate in the external phase (\( S_e \)):

\[ -V_e \frac{dS_e}{dt} = \frac{3}{R}(V_i + V_m +)k_{eq}(S_e - S_{e*}) \]  \( \ldots (1) \)

(B) Mass balance of substrate in the membrane portion of the globules (\( S_m \)):

\[ \phi_m \frac{\partial S_m}{\partial t} = (D_{es}/r^2)(r^2 \frac{\partial S_m}{\partial r}) - \phi_s \left( \frac{\partial S_i}{\partial r} + \frac{V_{mi}S}{k_{mi} + S} \right) \]  \( \ldots (2) \)

where \( \phi_s = V_i/(V_i + V_m), \phi_m = V_{mi}/(V_i + V_m) \)

(C) Mass balance of the 1\textsuperscript{st} enzymatic reaction product glucose (designated as G) in the internal phase of the emulsion liquid membranes:

\[ \frac{dG}{dt} = \frac{V_{mi}S}{k_{mi} + S} \cdot \frac{V_{n2}G}{k_{n2}(1 + P/Ki) + G} \]  \( \ldots (3) \)

(D) Mass balance of the 2\textsuperscript{nd} enzymatic reaction product, gluconic acid (designated as P) in the internal phase of the emulsion liquid membranes:

\[ \phi_i \frac{\partial P}{\partial t} = \phi_m \frac{\partial C}{\partial t} = (D_{es}/r^2)(r^2 \frac{\partial C}{\partial r}) + \frac{V_{n2}G}{k_{n2}(1 + P/Ki) + G} \]  \( \ldots (4) \)

(E) Mass balance of carrier (B) and carrier-product complex (C) in emulsion phase:

\[ \phi_m \left( \frac{\partial B}{\partial t} + \frac{\partial C}{\partial t} \right) = (1/r^2)(D_{cb}r^2 \frac{\partial B}{\partial r}) + D_{ec}r^2 \frac{\partial C}{\partial r} \]  \( \ldots (5) \)

(F) Mass balance of final, external product calcium gluconate (\( P_e \)):

\[ V_e \frac{dP_e}{dt} = \frac{3}{R}(V_i + V_m)k_{eq}(P_e - P_{e*}) \]  \( \ldots (6) \)
(G) Relation between external phase stripping reagent concentration \((R_s)\) and final external phase product \((P_e)\) i.e., calcium gluconate :

\[ R_s = R_{s0} - P_e \] ... (7)

(H) Relation between carrier \((B)\) and product-carrier complex \((C)\) in the emulsion :

\[ B = B_0 - C \] ... (8)

(I) Distribution of substrate between membrane and aqueous phases :

\[ S_m = \alpha \cdot S_{\text{aqueous}} \] ... (9)

Equilibrium Relations :

\[
\begin{bmatrix} C \\ P \\ B \end{bmatrix} = K_{ei} 
\] ...

\[
\begin{bmatrix} C \\ R_x \\ B \end{bmatrix} = K_{ee} \] ...

Boundary conditions :

\[
\frac{\partial S_m}{\partial r} = 0 \quad \text{at} \quad r = 0 \quad \text{for} \quad t > 0 \quad ... (12)
\]

\[
\frac{\partial B}{\partial r} = \frac{\partial C}{\partial r} = 0 \quad \text{at} \quad r = 0 \quad \text{for} \quad t > 0 \quad ... (13)
\]

\[
K_{ei}(P_e^+ - P_e) = -D_x \frac{\partial C}{\partial r} \quad \text{at} \quad r = R \quad \text{for} \quad t > 0
\]

\[
K_{ee}(S_e - S_{e^+}) = D_x \frac{\partial S_m}{\partial r} \quad \text{at} \quad r = R \quad \text{for} \quad t > 0
\]

\[ S_m = \alpha \cdot S' \quad \text{at} \quad r = R \quad \text{for} \quad t > 0 \quad ... (15)\]

Initial Conditions :

\[ S_e = S_{e0} ; \quad R_s = R_{s0} ; \quad P_e = 0 \quad \text{at} \quad t = 0 \quad \text{for} \quad 0 \leq r \leq R \quad ... (17)\]

\[ S = S_m = 0 \quad \text{at} \quad t = 0 \quad \text{for} \quad 0 \leq r \leq R \quad ... (18)\]

\[ B = B_0 ; \quad C = C_0 \quad \text{at} \quad t = 0 \quad \text{for} \quad 0 \leq r \leq R \quad ... (19)\]

\[ G = 0 ; \quad P = 0 \quad \text{at} \quad t = 0 \quad \text{for} \quad 0 \leq r \leq R \quad ... (20)\]

**Developing the software**

The present dynamic mathematical model which describes the transient behaviour of the liquid membrane-encapsulated sequential biocatalytic reaction system involves a number of nonlinear partial differential equations. The first task, therefore, is to convert these partial differential equations into ordinary differential equations. In doing so, finite difference technique is applied in which the spatial derivatives are discretized while the temporal derivatives are left unconverted.

For numerical solution of the partial differential equations, a spatial grid is used to provide for the variation of the dependent variable (i.e., substrate or product concentration designated as \(y(r,t)\)) with respect to spatial variable \((r)\). A grid may be defined as indicated in Fig. 1.

The position along the grid is defined in terms of grid index, \(i\), the total number of grid points, \(N\), and the length of the system in each spatial dimension, in this case \(R\). Each grid point is separated by grid spacing \(\Delta r = R/(N-1)\). So, any grid position along the grid can be specified in terms of the index as \(r = (i-1)\Delta r\) where \(i = 1, 2, 3, \ldots, N-1, N\). When \(i = 1, r = 0\) and when \(i = N, r = (N-1)\Delta r = R\). In the present scheme, \(i = 1\) and \(N\) will correspond to the center of the spherical emulsion globule and the external surface of the globule respectively.

In development of approximations for the spatial derivatives, Taylor series is used as the basis. To derive an algebraic formula for the approximation of a first order derivative of the form, \(dy(dr) = y(n)\), using the grid values \(y(n+1), y(n), y(n-1)\), the following Taylor series is used:

\[ y(n+1) = y(n) + (dy(dr))(n+1 - n) + (1/2!) \]

\[ (d^2 y(dr)^2)((n + 1 - n)^2) + \ldots \]

\[ = y(n) + (dy(dr))(n+1) + (1/2!) \quad (d^2 y(dy)^2)(\Delta r)^2 + \ldots \] ... (21)

where \(\Delta r = r_{i+1} - r_i\)
\[ y(n+1) = y(n) + \frac{dy(y)}{dr}(\Delta r) + \frac{1}{2!}(d^2y(n)/dr^2)(\Delta r)^2 \]  
\[ \frac{dy(n)}{dr^2} = \frac{y(n+1) - y(n)}{2\Delta r} + O(\Delta r^2) \]  
\[ y(n) = y(n) + \frac{dy(n)}{dr}(\Delta r)^2 + \frac{1}{2!}(d^2y(n)/dr^2)(\Delta r)^2 + \frac{1}{2!}(d^3y(n)/dr^3)(\Delta r)^3 + \cdots \]  
\[ dy(n)/dr = (-3y(n) + 4y(n) - y(n))/2\Delta r + O(\Delta r^2) \]  
\[ y(n) = y(n) + \frac{dy(n)}{dr}(\Delta r)^2 + \frac{1}{2!}(d^2y(n)/dr^2)(\Delta r)^2 + \frac{1}{3!}(d^3y(n)/dr^3)(\Delta r)^3 + \cdots \]  
\[ y(n+1) = y(n) + \frac{3k_1 - 2k_2 + k_3}{6} + \cdots \]  
Error Monitoring

The estimate of the local truncation error in Runge-Kutta-Fehlberg method is obtained using Eq.(35).

\[ E = \frac{k_1 - 128k_3}{360} - \frac{2197k_4}{4275} - \frac{k_5}{75240} + \frac{2k_6}{5} - \frac{55}{5} \]  
Thus approximations for all the grid points are derived and these are now programmed taking special care for the boundary points where the denominator in the computation of a derivative often involves a zero valued term. In such a situation, l'Hospital's rule (differentiation of both the numerator as well as the denominator with respect to the spatial variable \( r \)) is applied.

After all the partial differential equations are converted into ordinary differential equations, the next step obviously becomes integration of the differential equations. The Runge-Kutta-Fehlberg method is used to integrate the differential equations.

The integration algorithm used are:

\[ k_1 = hf(y_i, t_i) \]  
\[ k_2 = hf(y_i + \frac{1}{4}k_1, \frac{t_i}{4} + \frac{h}{4}) \]  
\[ k_3 = hf(y_i + \frac{3}{32}k_1 + \frac{9}{32}k_2, \frac{t_i}{2} + \frac{3}{8}h) \]  
\[ k_4 = hf(y_i + h + \frac{1932}{2197}k_1 - \frac{7200}{2197}k_2 + \frac{7296}{2197}k_3, \frac{t_i}{12} + h) \]  
\[ k_5 = hf(y_i + \frac{439}{216}k_1 + \frac{448}{216}k_2, \frac{845}{1041}k_1 + \frac{4104}{k_1} + \frac{11k_6}{40}, \frac{t_i}{2}) \]  
\[ y_{i+1} = y_i + \frac{1}{2}(k_1 + 2k_2 + k_3 - k_4 + k_5 + k_6) - \frac{1}{5}k_5 \]  
\[ y_{i+1} = y_i + \frac{25k_1}{216} + \frac{1408k_2}{2565} + \frac{2197k_3}{4104} + \frac{2k_4}{5} + \frac{k_5}{55} \]  
Truncation error for each dependent variable is estimated and compared with the specified error tolerance. In the computer coding, the integration step size is made to be dependent on the error tolerance so that if error tolerance exceeds, the integration step size is reduced and integration is repeated from the current base point. If all the dependent variables pass the
error test then integration is allowed to continue after applying the error correction to improve the solution. When error criterion is not violated, it is checked whether step size can be increased before taking a step along the solution. Thus change in step size depends on the logical relation between the user-specified error criterion and the actual error that results during integration. This is how automatic step size adjustment mechanism works in the integration process.

Basic Integration Operations

The integration process is completed through the following basic operations:

(i) evaluation of the derivatives \( \frac{dy}{dt} \);
(ii) stepping from base point at \( i \) to the advanced point at \( i+1 \) using Runge-Kutta-Fehlberg integration algorithm (Eq. 34);
(iii) evaluation of the error at the advanced point by Eq. (35);
(iv) comparison of the truncation error with the user-specified error criterion to relate this to integration step size;
(v) adjustment of the integration step-size to meet user-specified error criterion.

Computation Procedure

The total computation procedure as shown in Fig. 2 is completed through the following steps:

(i) a subroutine INIT AL is written to define the initial condition vector \( y_0 \);
(ii) the initial condition vector is sent (through COMMON) to the integrator to start the numerical integration at the base point \( i = 0 \);
(iii) the integrator calls a user-written subroutine DERV to evaluate the derivatives at the point \( i(=0) \) for the first call to DERV and the integrator then takes a step along the solution using the stepping formula of the Runge-Kutta-Fehlberg integration algorithm and if necessary using the derivatives at several points between \( i \) and \( i+1 \) for the purpose of producing the solution vector;
(iv) the integrator then calls DERV to evaluate the derivative vector at \( i+1 \) using the previously computed solution vector at \( i+1 \) and in the next step the integration error vector is computed using Eq. (35);
(v) if the estimated error vector exceeds the user-specified error criterion, the integration step size is reduced and integration step is repeated from the current base point by returning to step (iii);
(vi) when estimated error vector remains within the user-specified limit, error vector is added to the solution vector at \( i+1 \) to improve the solution.
Table 3—Numerical output from the software showing variation of normalized complex concentration $CN(r,t)$ with radial position $r$ (in meter) and time $t$ (in minutes)

<table>
<thead>
<tr>
<th>Time, min</th>
<th>$CN(0,t)$</th>
<th>$CN(0.0000,t)$</th>
<th>$CN(0.0001,t)$</th>
<th>$CN(0.00015,t)$</th>
<th>$CN(0.0002,t)$</th>
<th>$CN(0.00025,t)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>00.00</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>15.00</td>
<td>0.0275</td>
<td>0.0275</td>
<td>0.0275</td>
<td>0.0274</td>
<td>0.0270</td>
<td>0.0200</td>
</tr>
<tr>
<td>30.00</td>
<td>0.0475</td>
<td>0.0475</td>
<td>0.0450</td>
<td>0.0440</td>
<td>0.0425</td>
<td>0.0400</td>
</tr>
<tr>
<td>45.00</td>
<td>0.0550</td>
<td>0.0550</td>
<td>0.0540</td>
<td>0.0530</td>
<td>0.0525</td>
<td>0.0520</td>
</tr>
<tr>
<td>60.00</td>
<td>0.0600</td>
<td>0.0600</td>
<td>0.0600</td>
<td>0.0575</td>
<td>0.0565</td>
<td>0.0560</td>
</tr>
</tbody>
</table>

Fig. 3—Numerical output from the software showing variation of integrated variables $S_e, P_e$ (normalized to $S_eN, P_eN$ on division by initial value of $S_e$) with time.

In Fig. 2, depicting the computation procedure, /Y/ is a common area defined for the dependent variable vector or solution vector. /T/ is a common area defined for the independent vector time ($t$) and /F/ represents a common area defined for the derivative function vector ($\frac{dy}{dt}$).

Results and Discussion

The software (MEMBSOL) developed has successfully solved the model equations for the liquid membrane-immobilized sequential bi-enzymatic reaction system. A typical set of model parameters and initial conditions have been presented in Table 1 and Table 2 respectively. The numerical output in terms of some dependent variables (integrated), such as $S_e$ and $P_e$ (normalized to variables $S_eN$ and $P_eN$ on division by initial external phase substrate concentration) against time have been displayed in Fig. 3. Variation of the space variable $C(r,t)$ with respect to radial position ($r$) and time on normalisation to $CN(r,t)$ in the same way has been shown in Table 3. In obtaining the numerical results from the software, relative error has been fixed at 0.001 (i.e., 0.1%) which is well within acceptable limit for the present system as well as for the similar situations. Advantage of this relative error term is that it accommodates all variables regardless of their magnitudes.

While developing the present software, $4^{th}$ order Runge-Kutta method was initially used to integrate the differential equations for the sake of simplicity. However, subsequently, $4^{th}$ order Runge-Kutta method has been replaced by Runge-Kutta-Fehlberg method as it is considered computationally more efficient than the former. In the classical $4^{th}$ order Runge-Kutta method, a total number of 11 ($3\times 4 - 1$) function evaluations are required at each stage of the numerical solution whereas Runge-Kutta-Fehlberg algorithm involves just six functional evaluations for moving from a base point to an advanced point. The local and global error terms in Runge-Kutta-Fehlberg integration algorithm, are sixth and fifth order respectively whereas these error terms in Runge-Kutta algorithm, are fifth and fourth order respectively. An important aspect of the software is that the user need not bother about integration step size. The integrator automatically selects the integration step size to achieve the desired accuracy.

Conclusions

Enzymes which are conventionally used for production of foods, pharmaceuticals and biomedicals are now being proved as promising tools even in biomedical analysis and wastewater treatment. But the main problem associated with an enzyme-based process is the high cost of the enzymes themselves. So, if by some means, these enzymes can be used repeatedly then the enzyme-based process may be made cost-effective and economically viable. Immobilization of
enzymes is considered as one of the effective means by which enzyme-activity can be retained for a prolonged period permitting repeated use. When enzymes are immobilized in emulsion liquid membranes, they constitute what is known as enzyme emulsion liquid membrane reactor (EELM) and these EELM reactors offer a well-reproducible and excellent technique for enzyme-based manufacture of biochemicals, pharmaceuticals etc. at a cost lower than that involved in conventional enzyme-based manufacturing techniques, as the liquid membrane-immobilized enzymes can be used repeatedly and continuously under the protective barrier of the liquid membrane. For scale-up and design of these noble bioreactors, mathematical modeling is very essential. But only in very few recent research works, mathematical modeling of EELM reactors have been taken up seriously. One problem which very often stands in the way of dynamic modeling of these bioreactors is the difficulty in mathematical solution of the concerned model equations. Though in some cases, modeling efforts have covered single enzyme systems, no such work has yet been reported for the multienzyme systems which are considered to present much more complicated picture in terms of mathematical expressions while being very useful for industrial production of biochemicals and pharmaceuticals compared to single enzyme systems.

Liquid membrane-immobilized multienzyme systems involve a number of algebraic, ordinary differential and nonlinear partial differential equations with different types of boundary conditions. Model equations take complicated forms because of presence of a number of physico-chemical phenomena like, per-emulsion mass transfer, diffusion through the membrane, different transport mechanisms of the reactants and the products etc. One such dynamic mathematical model has been developed (and published) for a sequential bienzymatic reaction system immobilized in emulsion liquid membrane by the present authors. A software (MEMBSOL) has been developed for numerical solution of this liquid membrane-immobilized and enzyme-based bioreactor model in the present work. The computation technique is based on a three point central difference scheme. Use of fictitious points at the boundaries could be avoided through mathematical manipulations. The software can produce results with a reasonably high accuracy as it incorporates an automatic step-size adjustment mechanism in the integration process. For instance, in the present computations, results have been obtained on fixing the relative error at 0.1 percent. This numerical solution technique is, thus expected to facilitate realistic modeling work for similar multienzyme-based processes and bioreactors under immobilized conditions of the enzymes (in emulsion liquid membranes) and will, thus pave the way for scale-up and design of industrial bioreactors (agitated batch or continuous types) using sequential multienzymatic reactions for the final productions.

**Nomenclature**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>B or ([B])</td>
<td>carrier concentration, kmol/m³</td>
</tr>
<tr>
<td>C or ([C])</td>
<td>concentration of carrier-product complex, kmol/m³</td>
</tr>
<tr>
<td>(D_e)</td>
<td>effective diffusivity, m²/s</td>
</tr>
<tr>
<td>(K_e)</td>
<td>equilibrium constant at internal interface of membrane (kmol/m³)⁻¹</td>
</tr>
<tr>
<td>(K_{ee})</td>
<td>equilibrium constant at external interface of membrane, dimensionless</td>
</tr>
<tr>
<td>(K_i)</td>
<td>product inhibition constant, kmol/m³</td>
</tr>
<tr>
<td>(k_eP)</td>
<td>external phase mass transfer coefficient for substrate, m/s</td>
</tr>
<tr>
<td>(k_m)</td>
<td>Michaelis-Menten constant, kmol/m³</td>
</tr>
<tr>
<td>(P) or ([P])</td>
<td>concentration of product, kmol/m³</td>
</tr>
<tr>
<td>(r)</td>
<td>emulsion globule radius, m</td>
</tr>
<tr>
<td>(S) or ([S])</td>
<td>concentration of substrate, kmol/m³</td>
</tr>
<tr>
<td>(V_e, V_i, V_m)</td>
<td>emulsion phase volumes for external, internal and membrane phases, m³</td>
</tr>
<tr>
<td>(V_{max})</td>
<td>Michaelis-Menten maximum reaction velocity, kmol/m³s</td>
</tr>
</tbody>
</table>

**Greek Letters**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\alpha)</td>
<td>phase partition coefficient</td>
</tr>
<tr>
<td>*</td>
<td>interface</td>
</tr>
</tbody>
</table>

**Superscripts**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>interface</td>
</tr>
</tbody>
</table>

**Subscripts**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>carrier ; c = complex ; e = external phase ; i = internal phase ; m = membrane phase ; p = product ; s = substrate ; 0 = initial condition</td>
</tr>
</tbody>
</table>

**References**