Ball State Undergraduate Mathematics Exchange, 18(1), Fall 2024, 2-25
https://openjournals.bsu.edu/mathexchange
DOI: 10.33043/28xy39zc

Pages 2 – 25

A Comparison of First- and Second Order-in-Time Finite Difference Methods Applied to Nonlinear Reactive Transport

Anastasia Wilson , Carson Morris, Kayli Hendricks, Karen Lawrence*

Karen Lawrence graduated from Augusta University in 2022 with a Bachelors Degree of Science in Mathematics, having worked on this paper from her Junior to Senior year. Karen currently works full-time in the legal field and plans to attend law school in 2025.

Carson Morris graduated with a bachelors degree in mathematics at Augusta University in 2022, and he is scheduled to receive his masters degree in mathematics in May of 2024. He has a wide variety of research interests, including but not limited to probability, statistics, numerical analysis, and differential equations.

Kayli Hendricks contributed to this paper as an undergraduate at Augusta University where, in 2023, she graduated with a bachelors degree in mathematics and a Minor in Accounting. Since graduating, she has been promoted to Office Manager at a local tax and accounting firm in Augusta, Georgia.

***Corresponding author**: anawilson@augusta.edu

Anastasia Wilson is an Associate Professor of Mathematics at Augusta University with a Ph.D. in Computational Mathematics from Clemson University and an M.S. in Applied Mathematics from NC State. Formerly at NASA Langley Research Center, her research focuses on mathematical modeling, computational methods, numerical analysis, and optimization, particularly in interdisciplinary.

Abstract

In this paper, we consider solution methods for the nonlinear reactive transport equation used to model the protein adsorption process. Efficient methods for simulating this process are necessary to aid in the development of novel adsorptive chromatography media to ensure high-volume production of purified product for the purposes of biotherapeutics. Using MATLAB®, we compare four finite difference schemes used to solve the nonlinear reactive transport equation, focusing on the differences of efficacy between implicit and explicit methods. As such, two of the methods are semi-implicit and two are explicit with one of each kind using a first-order temporal scheme and one of each using a second-order temporal scheme. The semi-implicit methods evaluate almost all terms implicitly while lagging the nonlinear coefficient function in time to linearize the equations. We include numerical results that indicate optimal convergence of the schemes, and we compare the effectiveness of the schemes in matching experimental data using two different boundary conditions.

1 Introduction

Recent decades have seen a large increase in demand for next-generation biologics, that is, therapies derived from biological sources [1, 2, 3]. In particular, the use of protein therapeutics such as monoclonal antibodies for treatment of COVID-19 and other viruses has led to a need for fast production of such biotherapeutics. Meeting these needs requires the development of novel adsorptive chromatography media to ensure high-volume throughput of purified product.

Resin-bead chromatography, which is used for many bioseparation neads because of its reliability, is relatively slow with low efficiency and, as a result, is considered a “must be addressed" factor to meet the rising demand [4]. Membrane chromatography, which uses a porous, adsorptive membrane instead of small resin beads, is being studied as an alternative to resin bead chromatography (see for example [5, 6, 7, 8, 9, 10, 11] and references therein). Membranes enable higher productivity because they maintain high protein binding capacities at higher flow rates [9]. Efficient methods for simulating the chromatography process using membranes could aid in the development of these media to reduce cost and production time.

Separation processes using adsorptive membranes can be modeled using the

reactive transport equation [12]

$$\omega \frac{\partial C}{\partial t} + \rho_s (1-\omega) \frac{\partial q}{\partial t} + \nabla \cdot (\mathbf{u}C) - \nabla \cdot (\mathbf{D}\nabla C) = f, \ \mathbf{x} \in \Omega, \ t > 0 \tag{1}$$

where ω is the porosity of the media or solid phase, ρ_s is the density of the solid phase, C is the liquid phase concentration, q is the solid phase concentration (or adsorbent), $\mathbf{u}$ is the Darcy velocity, and $\mathbf{D}$ is the hydrodynamic dispersion tensor. The function f represents sources and/or sinks within the domain Ω.

Some novel chromatography membranes being studied use multiple modes of interaction with the product to recover it selectively from impurities in the solution, leading to mathematically complex, nonlinear models to describe the adsorption relationship between C and q [10, 11, 13, 14, 15]. Since the nonlinearity is connected to a time-derivative term, handling this term is non-trivial particularly considering the fact that stability issues commonly arise with explicit numerical solutions of the transport equation.

Due to the nonlinearity arising from the adsorption model, fully explicit methods are easier to implement and are faster to solve in each time iteration. However, fully explicit methods introduce a time step restriction that is not always practical when simulating experimental data. In that case, implicit methods may be faster in the long run since larger time steps are allowed with implicit methods. Still, implicit methods are more complex to implement and care needs to be taken when solving the nonlinearity arising from the adsorption model which is associated with a time-derivative term. While this paper focuses on solution methods with an explicitly-defined adsorption isotherm, having a thorough understanding of methods applied to explicitly-defined adsorption models will provide a basis for future work considering an implicitly-defined isotherm as in [13] which more accurately models multi-modal adsorption.

In general, solution algorithms for these types of transport problems coupled with adsorption models have been well documented. In many cases, finite elements are used to spatially discretize the equation [16, 17, 18, 19, 20, 33, 22, 23, 24, 25]. Because of the potential nonlinearity arising from the adsorption model, the use of finite elements leads to a complex standard form causing higher-order temporal integration methods to be hard to implement. Using finite differences [19, 26, 27] for the spatial discretization greatly simplifies the standard form, and hence higher-order temporal schemes are easier to use with finite differences.

Although finite differences have been used to solve similar transport problems with adsorption, the research team involving the authors has focused on finite element methods in the past [23, 24, 25]. Recent work by our group [25] has compared different higher order temporal discretization schemes using finite elements applied to these types of problems in the case of membrane adsorption and nonlinear adsorption models. Hence the work presented in this paper is meant as a comparison of the same higher order temporal discretization algorithms using finite difference methods applied to the same problem. This paper focuses on using finite differences to solve the one-dimensional reactive transport equation (1) and compares implicit and explicit finite difference schemes using both a first- and second-order temporal integration scheme for

each case of finite difference scheme. MATLAB® is used to implement the finite difference schemes. In this paper, we use an explicit function to describe the relationship between the solid phase concentration q and the liquid phase concentration C, i.e., $q = q(C)$, as is the case for the Langmuir and Freundlich isotherms. These explicit models represent separation processes defined by a single interaction mode, e.g., ion exchange or hydrophobic interaction [28]. We hope to extend the solution methods studied in this paper to account for multiple modes of adsorption by using an implicitly defined isotherm as in [13].

2 Model Equations

In this paper, we restrict our attention to the one-dimensional problem so that the domain Ω is a bounded domain in $\mathbb{R}^1$, ie. $\Omega = [a,b]$ with $|a| < \infty, |b| < \infty$. The inflow boundary of Ω corresponds to $x = a$ and the outflow boundary corresponds to $x = b$.

The transport and adsorption of product at a concentration C in a porous medium can be modeled in one dimension using the reactive transport (advection-diffusion-reaction) equation [12, 33, 26]

$$\omega \frac{\partial C}{\partial t} + \rho_s (1-\omega) \frac{\partial q}{\partial t} + u \frac{\partial C}{\partial x} - D \frac{\partial^2 C}{\partial x^2} = f, \qquad x \in [a,b],\ t \geq 0, \tag{2}$$

$$q = g(q,C), \qquad x \in [a,b], t \geq 0, \tag{3}$$

with the initial and boundary conditions

$$C(x,0) = C_0, \qquad x \in [a,b], \tag{4}$$

$$q(x,0) = q_0, \qquad x \in [a,b], \tag{5}$$

$$C(a,t) = C_{in}, \qquad t \geq 0, \tag{6}$$

$$\left.\frac{\partial C}{\partial x}\right|_{x=b} = 0, \qquad t \geq 0. \tag{7}$$

Here C is the liquid phase concentration, q is the solid phase concentration (or the adsorbed concentration), ω and ρ_s are the porosity and density of the membrane respectively, u is the (Darcy) velocity through the membrane, D is the hydrodynamic dispersion tensor, and $f = f(x,t)$ is a forcing function normally taken to be 0 in the case of protein chromatography. We note that (6) uses a Dirichlet boundary condition on the inflow since the inflow concentration is known, and (7) uses a homogeneous Neumann condition to model continuity of concentration at the outflow. An alternate inflow condition using a Robin-type condition, specifically

$$uC - D \frac{\partial C}{\partial x} = uC_{in}, \qquad x = a, t > 0, \tag{8}$$

can be used to model continuity of flux at the inflow. The two conditions (7) and (8) are the so-called Danckwerts boundary conditions [27, 29]. These are considered in the simulations as a point of comparison.

Equation (2) describes the flow of the carrying fluid through the membrane,

where we assume instantaneous adsorption of the desired product from solution. The system is closed by the isotherm relationship, equation (3), which gives the isotherm relationship which describes the ability of the membrane to bind the protein. This relationship can be explicit linear or nonlinear functions of concentration [12], or it can be defined as a nonlinear, implicitly function of q and C [13, 30].

The solution methods in this paper are applied to the modified system in which the chain rule is used to rewrite the reactive transport equation (2). Rearranging (3) and differentiating in time, we have

$$\frac{\partial q}{\partial t} - \frac{\partial g(q,C)}{\partial t} = 0 \tag{9}$$

$$\Rightarrow \frac{\partial q}{\partial t} - \frac{\partial g}{\partial q}\frac{\partial q}{\partial t} - \frac{\partial g}{\partial C}\frac{\partial C}{\partial t} = 0 \tag{10}$$

$$\Rightarrow \frac{\partial q}{\partial t} = \frac{\partial g/\partial C}{1-\partial g/\partial q}\frac{\partial C}{\partial t} = \tilde{g}(q,C)\frac{\partial C}{\partial t} \tag{11}$$

where

$$\tilde{g}(q,C) = \frac{\partial g/\partial C}{1-\partial g/\partial q}. \tag{12}$$

Substituting (11) in (2) leads to the modified modeling equations:

$$(\omega + (1-\omega)\rho_s \tilde{g}(q,C))\frac{\partial C}{\partial t} + \mathbf{u}\cdot\nabla C - \nabla\cdot(\mathbf{D}\nabla C) = f, \tag{13}$$

$$q = g(q,C). \tag{14}$$

Note that in the case of an explicit isotherm equation, $\tilde{g}$ simplifies to $\tilde{g}(C) = \partial g/\partial C = q'(C)$, and hence the system reduces to one equation with one unknown:

$$(\omega + (1-\omega)\rho_s \tilde{g}(C))\frac{\partial C}{\partial t} + \mathbf{u}\cdot\nabla C - \nabla\cdot(\mathbf{D}\nabla C) = f. \tag{15}$$

3 Numerical Solution Schemes

For the numerical solution schemes, we assume an explicit isotherm model and apply finite difference methods to spatially discretize the one-dimensional reactive transport equation given in (15). We consider both explicit and semi-implicit temporal solution methods.

The semi-implicit methods evaluate almost all terms implicitly while lagging the nonlinear coefficient function $\tilde{g}(C)$ in time to linearize the equations as detailed in Section 3.1.1. For both the semi-implicit case and the explicit case, we apply a first-order and a second-order discretization method. Respectively for semi-implicit methods, we use lagged Backward Euler and lagged Implicit Trapezoid methods; and for explicit methods, we use Forward Euler and Improved Euler (Explicit Trapezoid).

The fully discrete problem partitions the time interval $[t_0, T]$ and spatial domain $[x_0, x_S]$ as

$$0 = t_0 < t_1 < \cdots < t_N = T \quad \text{and} \quad a = x_0 < x_1 < \cdots < x_S = b$$

respectively where $\Delta t = t_{n+1} - t_n$ with $t_n = n\Delta t$ and $\Delta x = x_{i+1} - x_i$ with $x_i = i\Delta x$. Defining $f_i^n := f(t_n, x_i)$ we have $C^n = C(t_n, x)$ and $q^n = q(t_n, x)$ for the continuous solutions and $C_i^n = C(t_n, x_i)$ and $q_i^n = q(t_n, x_i)$ for the discrete solutions. We will denote the discrete solutions in their entirety by $\hat{C}$ and $\hat{q}$.

3.1 Semi-Implicit Methods

In this section, we derive the systems of equations needed to solve the reactive transport equation (15) beginning with two different implicit methods: Backward Euler and Implicit Trapezoid. With both methods, we lag the coefficient function in equation (15) in order to linearize the equation and avoid an otherwise necessary root-finding method, such as Newton's method. This lagging changes our methods to semi-implicit.

3.1.1 First-Order in Time: Backward Euler

From equation (15), we define $g(C) = \omega + (1-\omega)\rho_s q'(C)$ and apply finite difference approximations to $\frac{\partial C}{\partial x}$ and $\frac{\partial^2 C}{\partial x^2}$ and a Backward Euler temporal discretization to $\frac{\partial C}{\partial t}$ to get

$$g(C_i^{n-1})\left(\frac{C_i^n - C_i^{n-1}}{\Delta t}\right) + \mathbf{u}_i^n\left(\frac{C_{i+1}^n - C_i^n}{\Delta x}\right) - \mathbf{D}_i^n\left(\frac{C_{i+1}^n - 2C_i^n + C_{i-1}^n}{(\Delta x)^2}\right) = f_i^n.$$

where $g(C_i^{n-1})$ is lagged in time to linearize the equation. Rearranging we find

$$\left(-\frac{\mathbf{D}_i^n \Delta t}{g(C_i^{n-1})(\Delta x)^2}\right)C_{i-1}^n + \left(1 - \frac{\mathbf{u}_i^n \Delta t}{g(C_i^{n-1})\Delta x} + \frac{2\mathbf{D}_i^n \Delta t}{g(C_i^{n-1})(\Delta x)^2}\right)C_i^n + \left(\frac{\mathbf{u}_i^n \Delta t}{g(C_i^{n-1})\Delta x} - \frac{\mathbf{D}_i^n \Delta t}{g(C_i^{n-1})(\Delta x)^2}\right)C_{i+1}^n = C_i^{n-1} + \left(\frac{\Delta t}{g(C_i^{n-1})}\right)f_i^n. \quad (16)$$

Denoting the coefficients of C_{i-1}^n, C_i^n, and C_{i+1}^n by X_i^n, Y_i^n, and Z_i^n respectively, and renaming the right hand side of (16) as R_i^{n-1}, we have simply

$$X_i^n C_{i-1}^n + Y_i^n C_i^n + Z_i^n C_{i+1}^n = R_i^{n-1}. \quad (17)$$

A system of equations is then formed using (17) and the boundary conditions described in Section 2.

To enforce the nonhomogeneous Dirichlet boundary condition at the inflow given by (6) and the homogeneous Neumann boundary condition at the outflow given by (7), we respectively set

$$C_0^n = C_{in} \quad \text{and} \quad C_S^n = C_{S-1}^n. \quad (18)$$

3.1.2 Second-Order in Time: Implicit Trapezoid

For the Implicit Trapezoid method we isolate $\frac{\partial C}{\partial t}$ to get

$$\frac{\partial C}{\partial t} = \frac{1}{g(C)}\left[-\mathbf{u}\frac{\partial C}{\partial x} + \mathbf{D}\frac{\partial^2 C}{\partial x^2} + f\right] = F(t, C). \quad (19)$$

Applying finite difference methods to $\frac{\partial C}{\partial x}$ and $\frac{\partial^2 C}{\partial x^2}$, we now arrive at

$$F(t,C_i) = \frac{1}{g(C_i)}\left[-\mathbf{u}_i\left(\frac{C_{i+1}-C_i}{\Delta x}\right) + \mathbf{D}_i\left(\frac{C_{i+1}-2C_i+C_{i-1}}{(\Delta x)^2}\right) + f_i\right]. \tag{20}$$

Note that we still use a first order approximation for the first derivative even though Implicit Trapezoid is a second-order method in time. Second order, centered-difference approximations for the first derivative term are known to be less accurate near a concentration front caused by an inflow flux as we are considering in this case. In the case of protein chromatography, accuracy near the concentration front is of vital importance. Forward or backward difference approximations are more accurate in this case. However, second-order, one-sided approximations cannot be used near the boundaries. Initial experimentation using a different approximation (either a centered-difference or first-order) near the boundary showed a reduction in the accuracy of the approximation so that using a second-order, one-sided approximation did not pay off. Hence we chose to use the simpler first-order approximation.

Evaluating at the nth time step gives

$$F(t^n,C_i^n) = \frac{1}{g(C_i^{n-1})}\left[-\mathbf{u}_i^n\left(\frac{C_{i+1}^n-C_i^n}{\Delta x}\right) + \mathbf{D}_i^n\left(\frac{C_{i+1}^n-2C_i^n+C_{i-1}^n}{(\Delta x)^2}\right) + f_i^n\right],$$

where we lag $g(C)$ in order to avoid root-finding methods. $F(C_i^{n-1},t^{n-1})$ is obtained similarly by evaluating (20) at the $n-1$st time step. Then applying an Implicit Trapezoid approximation, we obtain

$$\begin{aligned}\frac{C_i^n-C_i^{n-1}}{\Delta t} = \frac{1}{2g(C_i^{n-1})}\Bigg[&-\mathbf{u}_i^n\left(\frac{C_{i+1}^n-C_i^n}{\Delta x}+\frac{C_{i+1}^{n-1}-C_i^{n-1}}{\Delta x}\right)+\\ &\mathbf{D}_i^n\left(\frac{C_{i+1}^n-2C_i^n+C_{i-1}^n}{(\Delta x)^2}+\frac{C_{i+1}^{n-1}-2C_i^{n-1}+C_{i-1}^{n-1}}{(\Delta x)^2}\right)+(f_i^n+f_i^{n-1})\Bigg].\end{aligned}$$

Multiplying by $(\Delta x)^2\Delta t$ on both sides, expanding, and gathering like terms, we obtain

$$\begin{aligned}&\left(\frac{\mathbf{D}_i^n\Delta t}{2g(C_i^{n-1})}\right)C_{i-1}^n+\left(\frac{\mathbf{u}_i^n\Delta t\Delta x}{2g(C_i^{n-1})}-\frac{\mathbf{D}_i^n\Delta t}{g(C_i^{n-1})}-(\Delta x)^2\right)C_i^n\\ &\qquad+\left(-\frac{\mathbf{u}_i^n\Delta t\Delta x}{2g(C_i^{n-1})}+\frac{\mathbf{D}_i^n\Delta t}{2g(C_i^{n-1})}\right)C_{i+1}^n\\ =&\left(-\frac{\mathbf{D}_i^n\Delta t}{2g(C_i^{n-1})}\right)C_{i-1}^{n-1}+\left(-\frac{\mathbf{u}_i^n\Delta t\Delta x}{2g(C_i^{n-1})}+\frac{\mathbf{D}_i^n\Delta t}{g(C_i^{n-1})}-(\Delta x)^2\right)C_i^{n-1}\\ &+\left(\frac{\mathbf{u}_i^n\Delta t\Delta x}{2g(C_i^{n-1})}-\frac{\mathbf{D}_i^n\Delta t}{2g(C_i^{n-1})}\right)C_{i+1}^{n-1}+\left(-\frac{(\Delta x)^2\Delta t}{2g(C_i^{n-1})}\right)(f_i^n+f_i^{n-1}).\end{aligned} \tag{21}$$

Denoting the coefficients of C_{i-1}^n, C_i^n, and C_{i+1}^n for the Implicit Trapezoid approximation by $\hat{X}_i^n$, $\hat{Y}_i^n$, and $\hat{Z}_i^n$ respectively, and renaming the right hand side

of (21) as $\hat{R}_i^{n-1}$, we get

$$\hat{X}_i^n C_{i-1}^n + \hat{Y}_i^n C_i^n + \hat{Z}_i^n C_{i+1}^n = \hat{R}_i^{n-1}.$$

We note that although the system simplifies to a tridiagonal matrix as with Backward Euler, $\hat{X}_i^n$, $\hat{Y}_i^n$, $\hat{Z}_i^n$, and $\hat{R}_i^{n-1}$ are much more complicated so that more care must be taken when programming the equations. The process for calculating the concentration at any given time is now identical to that of Backward Euler.

3.2 Fully Explicit Methods

In this section, we derive the systems of equations needed to solve the reactive transport equation (15) using two different fully explicit methods: Forward Euler and Improved Euler (Explicit Trapezoid). Since explicit methods have known stability issues, a right-sided approximation is used to approximate $\frac{\partial C}{\partial x}$ to increase the stability of the explicit method solutions.

3.2.1 First-Order in Time: Forward Euler

As with the implicit methods, we use finite difference approximations for the spatial derivatives in (15), but now apply a Forward Euler approximation for $\frac{\partial C}{\partial t}$ to get

$$g(C_i^{n-1})\left(\frac{C_i^n - C_i^{n-1}}{\Delta t}\right) + \mathbf{u}_i^{n-1}\left(\frac{C_{i+1}^{n-1} - C_i^{n-1}}{\Delta x}\right)$$

$$-\mathbf{D}_i^{n-1}\left(\frac{C_{i+1}^{n-1} - 2C_i^{n-1} + C_{i-1}^{n-1}}{(\Delta x)^2}\right) = f_i^{n-1}$$

where again $g(C) = \omega + (1-\omega)\rho_s q'(C)$. We then solve for C_i^n getting

$$C_i^n = \left(1 + \frac{\mathbf{u}_i^{n-1}\Delta t}{g(C_i^{n-1})\Delta x} - \frac{2\mathbf{D}_i^{n-1}\Delta t}{g(C_i^{n-1})(\Delta x)^2}\right)C_i^{n-1}$$
$$+\left(\frac{-\mathbf{u}_i^{n-1}\Delta t}{g(C_i^{n-1})\Delta x} + \frac{\mathbf{D}_i^{n-1}\Delta t}{g(C_i^{n-1})(\Delta x)^2}\right)C_{i+1}^{n-1} + \left(\frac{\mathbf{D}_i^{n-1}\Delta t}{g(C_i^{n-1})(\Delta x)^2}\right)C_{i-1}^{n-1} + \frac{f_i^{n-1}\Delta t}{g(C_i^{n-1})}. \quad (22)$$

For simplicity, we let X_i^{n-1}, Y_i^{n-1}, Z_i^{n-1} denote the coefficient functions of C_{i-1}^{n-1}, C_i^{n-1}, C_{i+1}^{n-1} respectively and F_i^{n-1} denote the last term in (22). Then we have simply

$$C_i^n = X_i^{n-1}C_{i-1}^{n-1} + Y_i^{n-1}C_i^{n-1} + Z_i^{n-1}C_{i+1}^{n-1} + F_i^{n-1}$$

for all interior nodes ($i = 1, ..., S-1$). We enforce the boundary conditions on the exterior nodes as described in Section 3.1.1.

3.2.2 Second-Order in Time: Improved Euler (Explicit Trapezoid)

To obtain the Improved Euler formulation, we use an explicit approximation of the Implicit Trapezoid rule:

$$\frac{C_i^n - C_i^{n-1}}{\Delta t} = \frac{1}{2}\left[F(t^n, \bar{C}_i^n) + F(t^{n-1}, C_i^{n-1})\right], \quad (23)$$

where $\bar{C}_i^n = C_i^{n-1} + \Delta t F(t^{n-1}, C_i^{n-1})$ is the Forward Euler approximation of C and $F(t, C_i)$ is given by (20). Our derivation for $\bar{C}$ is the same as our Forward Euler derivation above, but we store it with different variables; namely, we label our coefficients as FX_i^{n-1}, FY_i^{n-1}, and FZ_i^{n-1} and the final term as FF_i^{n-1}:

$$\bar{C}_i^n = FX_i^{n-1} C_{i-1}^{n-1} + FY_i^{n-1} C_i^{n-1} + FZ_i^{n-1} C_{i+1}^{n-1} + FF_i^{n-1}.$$

The F in each term stands for *Forward* as these are the coefficients for the Forward Euler solution.

Next we use $\bar{C}$ to calculate the first term on the right side of (23)

$$F(t^n, \bar{C}_i^n) = \frac{1}{g(\bar{C}_i^n)} \left[f_i^n - \mathbf{u}_i^n \left(\frac{\bar{C}_{i+1}^n - \bar{C}_i^n}{\Delta x} \right) + \mathbf{D}_i^n \left(\frac{\bar{C}_{i+1}^n - 2\bar{C}_i^n + \bar{C}_{i-1}^n}{(\Delta x)^2} \right) \right],$$

which we will denote by PF, with P standing for *Primary*. Rearranging, we get

$$\begin{aligned} PF(t^n, \bar{C}_i^n) = &\left(\frac{\mathbf{D}_i}{g(\bar{C}_i^n)(\Delta x)^2} - \frac{\mathbf{u}_i^n}{g(\bar{C}_i^n)\Delta x} \right) \bar{C}_{i+1}^n \\ &+ \left(\frac{\mathbf{u}_i^n}{g(\bar{C}_i^n)\Delta x} - \frac{2\mathbf{D}_i^n}{g(\bar{C}_i^n)(\Delta x)^2} \right) \bar{C}_i^n \\ &+ \left(\frac{\mathbf{D}_i^n}{g(\bar{C}_i^n)(\Delta x)^2} \right) \bar{C}_{i-1}^n + \frac{f_i^n}{g(\bar{C}_i^n)}, \end{aligned} \quad (24)$$

and for simplicity we rewrite this as

$$PF(t^n, \bar{C}_i^n) = PX_i^{n-1} \bar{C}_{i-1}^n + PY_i^{n-1} \bar{C}_i^n + PZ_i^{n-1} \bar{C}_{i+1}^n + PpF_i^{n-1}.$$

by renaming the coefficients and last term of (24) as PX_i^{n-1}, PY_i^{n-1}, PZ_i^{n-1}, and PpF_i^{n-1} respectively.

The calculations for $F(t^{n-1}, C_i^{n-1})$, the second F in our Improved Euler formula, are similar to the *Primary* calculations above with the only difference being the shift to the $n-1$ time step. We use the label SF, standing for *Secondary F* to distinguish this term. The Improved Euler Formula is then given by

$$C_i^n = C_i^{n-1} + \frac{\Delta t}{2} \left[PF(t^n, \bar{C}_i^n) + SF(t^{n-1}, C_i^{n-1}) \right].$$

We use this equation for all interior nodes ($i = 1, ..., S-1$). Again, we enforce the boundary conditions on the exterior nodes as described in Section 3.1.1.

4 Numerical Results

In this section, we present results from implementations of the finite difference schemes described in this paper using MATLAB®. First, we give numerical convergence rates for the numerical approximation to the 1D nonlinear reactive transport equation using the four finite difference methods described in Section 3. Second, we compare numerical simulation data to experimental data in an effort to determine better ways to accurately predict breakthrough. We conclude this section with a comparison of the methods.

4.1 Computation of Convergence Rates

To illustrate the solution methods were performing correctly, we computed the numerical error convergence rates for each of the four methods described above. The error was calculated by computing the discrete $L^2(L^2)$ and $L^\infty(L^2)$ time-space norms of the difference between the theoretical and numerical solutions for each method [31].

For the computation of convergence rates, we take the domain to be $[t_0,T]\times[x_0,x_S]=[0,1]\times[0,1]$. The right hand side function f, the boundary conditions, and the initial condition were determined by the chosen true solution

$$C(t,x)=t\sin(\pi x).$$

Note that this function has homogeneous Dirichlet boundary conditions and an initial function that is zero everywhere. Additionally for simplicity, parameters are chosen to be the following

$$\mathbf{D}=1,\mathbf{u}=1,\rho_s=1,K_{eq}=1,\text{ and }q_{max}=1,\omega=.5$$

where K_{eq} and q_{max} are the parameters in the Langmuir adsorption model

$$q=\frac{K_{eq}q_{max}C}{1+K_{eq}C}. \tag{25}$$

As mentioned above, we make use of the discrete $L^2(L^2)$ and $L^\infty(L^2)$ time-space norms [31] which we denote for simplicity as $\|\cdot\|_{L^2(0,T;L^2(\Omega))}$ and $\|\cdot\|_{L^\infty(0,T;L^2(\Omega))}$ respectively. Specifically, for the exact solution $C(t,x)$ and the approximate solution $\hat{C}$,

$$\|C-\hat{C}\|_{L^2(L^2)}=\left(\sum_{n=0}^{N}\|C(t^n,\cdot)-\hat{C}^n\|^2_{L^2(\Omega)}dt\right)^{1/2}$$

and

$$\|C-\hat{C}\|_{L^\infty(L^2)}=\max_{0\le n\le N}\|C(t^n,\cdot)-\hat{C}^n\|_{L^2(\Omega)}$$

where

$$\|C(t^n,\cdot)-\hat{C}^n\|_{L^2(\Omega)}=\left(\sum_{i=0}^{S}\left(C(t^n,x_i)-\hat{C}_i^n\right)^2\right)^{1/2}.$$

To obtain the order of convergence for the error for both the Forward Euler and Backward Euler solutions, Δx and Δt are each cut in half as the errors are calculated for each pair of discretization values $((\Delta t)_j,(\Delta x)_j)$. The order of convergence is then calculated by

$$\text{Order}_j=\frac{\ln\left|\frac{\text{Err}_j}{\text{Err}_{j-1}}\right|}{\ln\left|\frac{\Delta t_j}{\Delta t_{j-1}}\right|}. \tag{26}$$

where Err_j indicates either the L^2 time-space norm $\|\cdot\|_{L^2(0,T;L^2(\Omega))}$ or the L^∞ time-space norm $\|\cdot\|_{L^\infty(0,T;L^2(\Omega))}$. Since Forward Euler and Backward Euler are both first-order time discretization methods, we expect the order to approach 1 as Δt and Δx are decreased.

For the second order methods (Implicit Trapezoid and Improved Euler), the discretization parameters were treated separately. Specifically, Δt was again cut in half each time while Δx was cut by a fourth; this was to account for the fact that the finite difference approximation used for the spatial derivative $\frac{\partial C}{\partial x}$ is only first order. The order was calculated the same as in (26). We then expect the order to approach 2 as Δt and Δx are decreased indicating that the methods are second order in time.

4.1.1 Fully Implicit Methods

First-Order in Time: Backward Euler Backward Euler, being a first order method, is $\mathscr{O}(\Delta t)$, so in combination with the finite difference formulas this leads to an overall error that is linear in both space and time as in

$$||C-\hat{C}||_{L^2(L^2)} \sim \mathscr{O}(\Delta t+\Delta x)$$

and

$$\|C-\hat{C}\|_{L^\infty(L^2)} \sim \mathscr{O}(\Delta t+\Delta x)$$

where C is the exact solution and $\hat{C}$ is the approximate solution. Therefore, we expect the rate of convergence to converge to 1 as Δt and Δx decrease which is shown by the results in Table 1.

$(\Delta t, \Delta x) \rightarrow$	$\left(\frac{1}{16}, \frac{1}{8}\right)$	$\left(\frac{1}{32}, \frac{1}{16}\right)$	$\left(\frac{1}{64}, \frac{1}{32}\right)$	$\left(\frac{1}{128}, \frac{1}{64}\right)$	$\left(\frac{1}{256}, \frac{1}{128}\right)$
$\|C-\hat{C}\|_{L^\infty(L^2)}$	5.221E-02	2.308E-02	1.082E-02	5.237E-03	2.575E-03
Rate	–	1.1776	1.0926	1.0474	1.0240
$\|C-\hat{C}\|_{L^2(L^2)}$	2.973E-02	1.282E-02	5.936E-03	2.853E-03	1.399-03
Rate	–	1.2131	1.1110	1.0568	1.0288

Table 1: Approximation errors and experimental convergence rates for the Backward Euler approximation. As Δt and Δx are cut in half, the L^2 error is reduced the same amount which is consistent with the theoretical convergence rates.

Second-Order in Time: Implicit Trapezoid Implicit Trapezoid is a second order method in time, $\mathscr{O}(\Delta t^2)$. In combination with the finite difference formulas this leads to an overall error that is linear in space yet quadratic in time as in

$$||C-\hat{C}||_{L^2(L^2)} \sim \mathscr{O}((\Delta t)^2+\Delta x)$$

and

$$\|C-\hat{C}\|_{L^\infty(L^2)} \sim \mathscr{O}((\Delta t)^2+\Delta x)$$

where C is the exact solution and $\hat{C}$ is the approximate solution. As expected, we see the convergence rate approaching 2 with both the L^∞ and L^2 errors as shown in Table 2.

$(\Delta t, \Delta x) \rightarrow$	$\left(\frac{1}{16}, \frac{1}{4}\right)$	$\left(\frac{1}{32}, \frac{1}{16}\right)$	$\left(\frac{1}{64}, \frac{1}{64}\right)$	$\left(\frac{1}{128}, \frac{1}{256}\right)$
$\|C-\hat{C}\|_{L^\infty(L^2)}$	1.334E-01	2.327E-02	5.237E-03	1.256E-03
Rate	–	2.5193	2.1515	2.0596
$\|C-\hat{C}\|_{L^2(L^2)}$	7.687E-02	1.308E-02	2.873E-03	6.688E-04
Rate	–	2.5553	2.1867	2.1029

Table 2: Approximation errors and experimental convergence rates for the Implicit Trapezoid approximation. As Δt is cut in half and Δx is cut into fourths, the L^2 error is reduced by a fourth which indicates quadratic convergence in time. Note that only four columns of data are given because of the small size of Δt and Δx.

4.1.2 Fully Explicit Methods

First-Order in Time: Forward Euler As with Backward Euler, Forwad Euler is a first-order time integration method. Therefore, we expect the rate of convergence to approach to 1 as Δt and Δx decrease which is shown by the results in Table 3. As Forward Euler is a fully explicit method, the stability of the solution depends on Δt being small enough with respect to Δx. Although a full theoretical stability analysis is ongoing, we expect from initial numerical investigations to have a second order stability requirement, i.e. $\Delta t \sim \mathscr{O}((\Delta x)^2)$. Furthermore, this requirement is consistent with what is expected for numerical stability in non-reactive diffusion problems [32]. Therefore to ensure stability of the error results shown in Table 3, we set $\Delta t = \frac{(\Delta x)^2}{50}$ and reduce Δx by half each time.

Δt	$\Delta x \rightarrow$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$
$4*\frac{(\Delta x)^2}{50}$	$\|C-\hat{C}\|_{L^\infty(L^2)}$	4.18E-01	1.34E-01	5.32E-02	2.35E-02	1.10E-02
	Rate	–	1.64	1.33	1.17	1.09
	$\|C-\hat{C}\|_{L^2(L^2)}$	2.32E-01	7.53E-02	2.98E-02	1.32E-02	6.20E-03
	Rate	–	1.62	1.33	1.17	1.09

Table 3: Approximation errors and experimental convergence rates for the Forward Euler approximation. As Δx and Δt decrease, the convergence rate approaches 1 indicating a linear rate.

Second-Order in Time: Improved Euler (Explicit Trapezoid) As with the Implicit Trapezoid method, Improved Euler is a second-order time integration method resulting in an overall error that is linear in space yet quadratic in time. To account for the slower spatial convergence, we modify the error computations similarly to what was done with Implicit Trapezoid: Δx is reduced by a fourth while Δt is reduced by only a half. We then expect the convergence rate to approach 2 as is shown in Table 4 with both the L^∞ and L^2 errors for Improved Euler.

As with Forward Euler, Improved Euler is a fully explicit method therefore

requiring a stability condition on Δt. Again from initial numerical investigations, we expect $\Delta t \sim \mathscr{O}(\Delta x)^2$. To maintain stability of the solution for the error computations with Δt and Δx being reduced by half independently in each iteration, we set Δt very small to start, specifically $\Delta t = \frac{1}{3000}(\Delta x_0)^2$ where $\Delta x_0 = 0.5$ is the initial Δx value.

$(\Delta t, \Delta x) \rightarrow$	$\left(\frac{1}{3000}\left(\frac{1}{2}\right)^2, \frac{1}{2}\right)$	$\left(\frac{1}{6000}\left(\frac{1}{2}\right)^2, \frac{1}{8}\right)$	$\left(\frac{1}{12000}\left(\frac{1}{2}\right)^2, \frac{1}{32}\right)$	$\left(\frac{1}{24000}\left(\frac{1}{2}\right)^2, \frac{1}{128}\right)$
$\|C-\hat{C}\|_{L^\infty(L^2)}$	4.18E-01	5.32E-02	1.10E-02	2.62E-03
Rate	–	2.974	2.270	2.072
$\|C-\hat{C}\|_{L^2(L^2)}$	2.31E-01	2.99E-02	6.20E-03	1.47E-03
Rate	–	2.952	2.268	2.072

Table 4: Approximation errors and experimental convergence rates for the Improved Euler approximation. As Δx and Δt decrease, the convergence rate approaches 2 indicating a quadratic rate. Note that, as with the Implicit Trapezoid results, only four columns of data are given because of the small size of Δt and Δx.

4.2 Experimental Comparison

To determine the efficacy of our simulation methods, we compare the simulated solution to experimental data. For the comparison with experimental data, we choose parameter values to mimic the associated laboratory conditions. The experimental data was obtained by Juan Wang in the Department of Chemical and Biomolecular Engineering at Clemson University as part of Scott Husson's Bioseparations and Advanced Separation Materials research group.

The multimodal membrane (MMM) was prepared by modification of a commercial regenerated cellulose membrane from Whatman, Inc., which has an average effective pore size of 1 μm and a thickness of 0.7 mm. Details of membrane synthesis were described previously [10]. A stack of 10 MMMs was placed in a Mustang Coin® module (Pall Corporation, Port Washington, NY) with one piece of 25 μm nominal pore diameter filter paper (Whatman 5) placed on each side of the stack. The resulting bed height of the module is approximately 0.7 cm. We orient our one-dimensional model in the vertical direction; hence, the interval for x is $[a,b] = [0,0.7]$. The effective membrane diameter within the module is 1.6 cm, but this does not affect the one-dimensional model. The module was installed in an AKTA purifier. The membrane bed porosity was measured to be $\omega = 0.84$. The interval for t was chosen to simulate the entire adsorption portion of the chromatography process which was found to be $[0,T] = [0,80]$ for the laboratory experiment. An immunoglobulin G (IgG) protein feed solution was pumped vertically through the membrane module using a constant pressure differential providing a constant flow rate of 0.1 mL/min.

For the dispersion, we chose a model correlated with low Peclet numbers [12] to account for the spreading effect due to both diffusion and convective flow. In the case of one-dimension, we consider only the longitudinal dispersivity D_L which is modeled in a low Peclét flow regime as

$$D = D_L = \omega d_0 + \alpha_L |\mathbf{u}|,$$

where ω is the porosity, d_0 is the diffusion coefficient, $|\mathbf{u}|$ is the mean microscopic velocity, and α_L and α_T are the longitudinal and transverse dispersivities, respectively. For an understanding of dispersion models in higher dimensions, one can see for example [12]. As stated above, the flow is purely vertical for the physical experiment being considered so we orient the longitudinal direction of the dispersion in the vertical direction.

Values for the diffusion coefficient of protein molecules in porous membranes range from 6×10^{-4} cm^2/min (for small proteins such as glycine) to 2.28×10^{-5} cm^2/min (for large proteins such as immunoglobulin G) [33]. We chose a diffusion coefficient value of 2.28×10^{-5} cm^2/min as IgG was used in the experiment.

The value for the mean microscopic velocity $|\mathbf{u}|$ was approximated from the constant flow rate. As the experimental flow rate is 0.1 mL/min:

$$|\mathbf{u}| \approx \frac{\text{flowrate}}{\pi r^2} = 0.0497 \text{ cm/min}. \tag{27}$$

The value given in (27) was also used for the coefficient velocity function for the advection term in the reactive transport equation (15) when using a Dirichlet inflow boundary condition. When considering a Danckwerts inflow boundary condition, $\mathbf{u}$ in both the reactive transport equation (15) and the inflow boundary condition (8) was taken to be as large as the flow rate (0.1) in an attempt to increase the accuracy of the simulations.

Values for the longitudinal dispersivity, α_L, are known to vary depending on the physical properties of the membrane. Experimental values for α_L on a small scale range from around a few millimeters to nearly a hundred centimeters depending on the porosity [12, 34], and α_L decreases as the porosity increases [35]. As our porosity is closer to 1, α_L should be on the lower end of this range, i.e., on the order of a few millimeters. The specific value of α_L was chosen to obtain a better match between the experimental and simulated data while still staying in the acceptable range. The specific values used are given with the results below.

For the numerical simulation results shown below, we use Langmuir's adsorption model (equation (25)) and obtained q_{max} and K_{eq} by fitting the model to measured adsorption isotherm data:

$$q_{max} = 150 \text{ mg/mL} \quad \text{and} \quad K_{eq} = 2.06 \text{ mL/mg}.$$

We set Δx and Δt to be small enough for each method to ensure the error in the numerical methods did not affect the results significantly. More specifics are given with the results below.

Productivity is an important metric in evaluating bind-and-elute chromatography processes. Previously, Husson and coworkers [9] defined productivity as the mass of protein that can bind per volume of membrane per time, and provided a convenient expression to calculate it:

$$\text{Productivity} = \frac{\text{B}_{\text{dynamic}}}{\text{t}_{\text{break}}}, \tag{28}$$

where B_{dynamic} is the dynamic binding capacity and t_{break} is the time it takes to reach the point of 10% breakthrough, i.e., the time at which the column effluent concentration reaches 10% of the feed concentration. Focusing on a 10% breakthrough is commonly used in practice as the time at which the protein column would be moved to the next stage in a continuous chromatography process [36, 37]. From a productivity standpoint, differences in dynamic binding capacity do not impact productivity nearly as significantly as differences in residence times [9]. Therefore, high productivities can be achieved when the time to breakthrough is short; that is, the membrane quickly reaches its capacity to bind proteins. Hence accurately predicting the time to 10% breakthrough is of great importance when simulating the chromatography process.

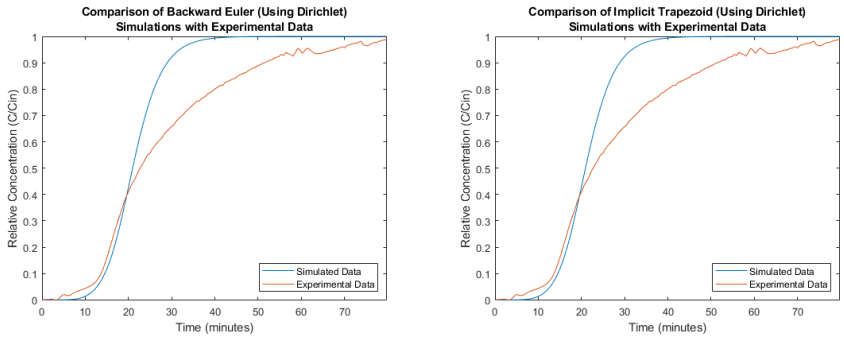
The information on protein breakthrough is visualized using a breakthrough curve, a plot of the relative concentration (C/C_{in}) of the effluent (outflow) as a function of time. In other words, Figures 1 and 2 show $C/C_{in} = C(b)/C_{in}$ as a function of time. We assessed the validity of our approach by comparing our simulated breakthrough curves with the experimental curves generated from laboratory data.

4.2.1 Fully Implicit Methods

In this section, we compare the numerical solutions obtained from Backward Euler and Implicit Trapezoid with the experimental data described above. Figure 1 shows comparisons of the numerical solutions to the experimental data. The discretization parameters were set to $(\Delta t, \Delta x) = (0.1, 0.1)$ to generate the numerical solutions. Both Backward Euler and Implicit Trapezoid using a Dirichlet inflow condition produced results that matched the experimental data very well up through approximately 40% breakthrough (where the curves intersect).

Although the accuracy of the simulations decreases after 40% breakthrough, these methods still provide a high amount of accuracy since predicting the experimental results close to column saturation is known to be difficult due to complex end behavior in the adsorption phase of the chromotographic process. The reasons for this complexity are still unclear but are thought to be cause by protein-protein interactions, non-specific binding, and multilayer adsorption effects [36]. Dimartino, Boi, and Sarti even point out that using the entire range of the concentration during the adsorption process would in fact make the model less accurate initially where it is more important to have higher accuracy [36]. Additionally, the adsorption mechanisms in effect for the experimental data under consideration in this paper are due to multiple modes of adsorption thus making the data more complex and likely more difficult to predict than data obtained with one mode of adsorption (e.g. affinity adsorption in [36]). Consequently, the fact that both solution methods provide accurate results not only for t_{break}, the time to 10% breakthrough, but also up through 40% breakthrough is strong support for the effectiveness of these methods as simulation tools.

Danckwerts Boundary Conditions The Danckwerts boundary conditions, specifically a Robin type boundary condition on the inflow and a Neumann boundary condition on the outflow, were also considered since these boundary conditions are theoretically more realistic. However, for our chosen model and solution methods, the results were much less accurate than those produced by the Dirichlet inflow condition, and hence we chose not to provide a reference figure. Using $(\Delta t, \Delta x) = (0.1, 0.1)$ as with the Dirichlet inflow condition, the data generated by Backward Euler and Implicit Trapezoid using Danckwerts the accuracy of the data was lower for earlier time although the shape of the numerical curve more closely resembles the shape of the experimental data for later t-values. Considering the metrics applied above to the results with Dirichlet inflow condition, accuracy earlier on is more important, and hence this inflow condition does not provide an effective simulation tool for the situation under consideration.



(a) Backward Euler (Dirichlet) (b) Implicit Trapezoid (Dirichlet)

Figure 1: Comparison of Backward Euler and Implicit Trapezoid using Dirichlet inflow boundary condition. Both methods produce accurate results up through 40% breakthrough. Accurately predicting the end behavior is a known issue with simulating adsorption due to complex end behavior of the adsorption phase.

The cause of the inaccuracy with the Danckwerts conditions is yet to be determined. Initial investigations using a higher-order approximation for the derivative term in the inflow condition showed no improvement. The decreased accuracy may be due to the simplifying assumption of a constant velocity affects since the velocity is taken into account in the inflow Danckwerts boundary condition. Further research is necessary to understand why these alternate boundary conditions resulted in less accurate results.

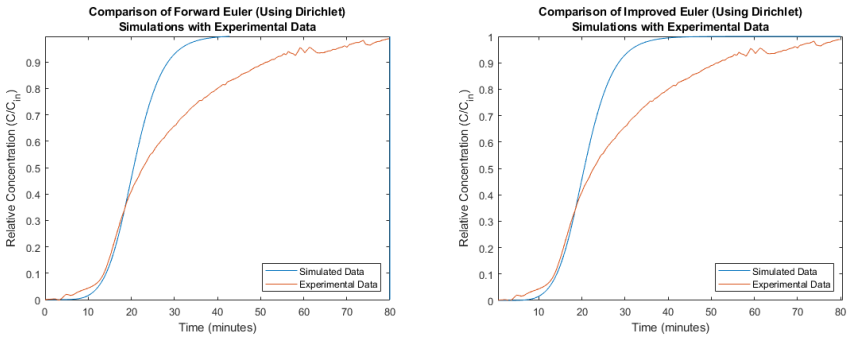
4.2.2 Fully Explicit Methods

In this section, we compare the numerical solutions obtained from Forward Euler and Improved Euler with the experimental data described above. As we can see from Figure 2, both the Forward Euler and the Improved Euler solutions accurately predict the experimental data up through 40% breakthrough similar to the semi-implicit methods, and therefore they both provide effective

simulation tools.

For both simulation results shown in Figure 2, Δx is set to 0.1 and Δt is set to $\frac{1}{50}(\Delta x)^2$. We found that making Δx any bigger (0.125) or smaller (0.0625) resulted in a lower level of accuracy. Changes in the coefficient of our Δt value from 1 to 1/200 seemed to have no noticeable effect; values outside of this range were not investigated. It is not surprising that changes in the Forward Euler code affecting accuracy had the same effect in the Improved Euler code as the Forward Euler method is used to calculate the Improved Euler results. We used this relationship to inform what discretization values were considered for producing the simulation results with Improved Euler.

Danckwerts Boundary Conditions Implementing the Danckwerts inflow boundary condition for Forward Euler and Improved Euler exacerbated the instability inherent in these explicit methods. Through numerical investigations, we found that the Danckwerts inflow condition could be used only if the magnitude of the derivative $\frac{\partial C}{\partial x}$ at the particular time step was less than 17 for Forward Euler and 2.1 for Improved Euler. Consequently, the Danckwerts inflow condition was implemented on a conditional basis: Danckwerts was used as the inflow condition for a particular time step if the magnitude of the derivative at that timestep was less than the given number (17 for Forward Euler, 2.1 for Improved Euler) and the Dirichlet inflow condition was used otherwise. With this implementation, the Forward Euler solution produced simulated results that were visually indistinguishable from the results using a Dirichlet inflow condition as shown in Figure 2a.



(a) Forward Euler (Dirichlet) (b) Improved Euler (Dirichlet)

Figure 2: Comparison of Forward Euler and Improved Euler. With $\Delta x = 0.1$ and $\Delta t = \frac{1}{50}(\Delta x)^2$ both graphs produce similar and accurate results particularly up through 40% breakthrough.

The results from Improved Euler with the Danckwerts inflow boundary conditions contain a notable difference from the graph for Improved Euler with the Dirichlet inflow boundary condition. Using the conditional implementation of the Danckwerts inflow condition described in the previous paragraph, the Improved Euler solution simulated losing some of the mass of the protein

effluent so that in the end the simulated outflow never reached the magnitude of the inflow concentration. In this way, the simulation is not physically accurate, so Improved Euler with a Danckwerts inflow condition cannot be used to predict experimental results. As with the semi-implicit methods, using a higher-order approximation for the derivative term in the inflow condition showed no improvement for either fully explicit method.

4.3 Comparison of Numerical Schemes

4.3.1 Semi-Implicit Methods: Backward Euler vs. Implicit Trapezoid

Although different size reductions for $\Delta t, \Delta x$ in the error analysis for Backward Euler and Implicit Trapezoid make it difficult to judge their effectiveness, we can make use of the common $(\Delta t, \Delta x)$ between the two methods, namely $(\frac{1}{32}, \frac{1}{16})$. The errors shown for both methods in Tables 1 and 2 are on the same order (10^{-2}) although the Backward Euler error is marginally smaller; however, reducing Δt while holding Δx constant would result in a larger reduction in the error with Implicit Trapezoid since it is of higher order.

For both Backward Euler and Implicit Trapezoid, simulations were done using a Neumann outflow while changing the inflow boundary condition from Dirichlet to Danckwerts. Using the Dirichlet boundary condition, both graphs shown in Figure 1 appear to be nearly identical. However, it is possible that with higher dimensions, the difference between the two results would be more noticeable. In the case of higher dimensions, the higher-order Implicit Trapezoid method may prove more useful in allowing larger step sizes while preserving accuracy, but further research is necessary to determine if this is the case.

We also compared computation time in regards to the simulations. The computation times were calculated with MATLAB's `tic` and `toc` commands. With $\Delta t = \Delta x = 0.1$ using the Dirichlet inflow boundary condition, Backward Euler has a computation time in the range of 0.083 to 0.097 seconds, while Implicit Trapezoid has a computation time in the range of 0.091 to 0.120 seconds. Considering the Danckwerts inflow boundary condition, Backward Euler has a computation time in the range of 0.078 to 0.096 seconds and Implicit Trapezoid has a computation time in the range of 0.109 to 0.131 seconds. Considering the similarity in the simulated results for the chosen discretization parameters, Backward Euler was more efficient than Implicit Trapezoid in producing the comparative data shown in this paper.

4.3.2 Explicit Methods: Forward Euler vs. Improved Euler

In order to compare the error results for Improved Euler with the error results for Forward Euler, we computed the errors associated with the Forward Euler solution for the same Δt and Δx values used for the Improved Euler error results given in Table 4; these new error calculations are given in Table 5. The L^∞ and L^2 errors for both methods are very similar. The Forward Euler error is occasionally slightly smaller than the Improved Euler error although the largest

difference is on the order of 10^{-3} and this difference shrinks as Δt and Δx shrink. Additionally, we would expect reducing Δt while holding Δx constant to result in a larger reduction in the error with Improved Euler since it is of higher order.

$(\Delta t, \Delta x) \rightarrow$	$\left(\frac{1}{3000}\left(\frac{1}{2}\right)^2, \frac{1}{2}\right)$	$\left(\frac{1}{6000}\left(\frac{1}{2}\right)^2, \frac{1}{8}\right)$	$\left(\frac{1}{12000}\left(\frac{1}{2}\right)^2, \frac{1}{32}\right)$	$\left(\frac{1}{24000}\left(\frac{1}{2}\right)^2, \frac{1}{128}\right)$
$\|C-\hat{C}\|_{L^\infty(L^2)}$	4.18E-01	5.30E-02	1.10E-02	2.62E-03
$\|C-\hat{C}\|_{L^2(L^2)}$	2.27E-01	2.94E-02	6.10E-03	1.45E-03

Table 5: Approximation errors and convergence rates for Forward Euler to compare to Table 5.

With a Dirichlet inflow boundary condition both codes produced accurate simulation results as shown in Figure 2. However, a Danckwerts inflow boundary condition could only be implemented conditionally depending on the magnitude of the first derivative which produced results that were unrealistic with Improved Euler and indistinguishable from the Dirichlet inflow condition with Forward Euler. Consequently, neither of the explicit methods were capable of fully utilizing a Danckwerts inflow boundary condition for simulations.

For our time analysis, we ran both codes with $\Delta x = \Delta t = 0.1$ and used our Dirichlet and Homogeneous Neumann boundary conditions; although this choice of Δt does not follow the stability requirements discussed previously, we found that the time step restriction was not necessary in practice with the chosen parameter values. We did not do a time analysis with Danckwerts due to the unrealistic output discussed in Section 4.2.2. The Forward Euler run time ranged from 0.008 to 0.032 seconds, and the Improved Euler run time ranged from 0.017 to 0.053 seconds. Hence, in general Forward Euler produced results faster than Improved Euler. However as with Implicit Trapezoid, the higher-order Improved Euler method may prove more useful in allowing larger step sizes in the case of higher dimension. It remains to be seen whether the time step restriction will negate the potential for larger time steps with Improved Euler in higher dimensions. Thus for one dimension, Forward Euler was the more efficient explicit simulation method.

4.3.3 Implicit vs. Explicit Methods

In comparing the implicit and explicit methods, the differences between errors were negligible so we will focus on two topics: efficacy with different boundary conditions and computation time.

With the implicit methods, both types of inflow boundary conditions that were considered (Dirichlet and Danckwerts) were easy to implement and produced reasonably accurate results in comparison to the experimental data. In comparison of the simulation results with the two boundary conditions, the Dirichlet condition was better for one dimensional simulations, but the Danckwerts condition may provide more accurate results in higher dimensions. For the explicit methods, the Danckwerts inflow condition exacerbated the instability, so a conditional implementation was necessary. Consequently, it did not noticeably improve the Forward Euler results, and it made the Improved Euler results less

accurate (in fact, unrealistic). Overall, the implicit methods were much better when considering more complex inflow boundary conditions.

For this comparison, we use the computation times described in Sections 4.3.1 and 4.3.2. Between the two implicit methods, Backward Euler had a slightly shorter computation time as expected since there are more function evaluations for Implicit Trapezoid. The explicit methods were much faster (as much as an order of magnitude faster) when considering the same discretization parameters. Between the two explicit methods, Forward Euler had faster computation times as expected since it has fewer function evaluations. However, the faster computation time can only be achieved if stability does not require the timestep to be extremely small. Therefore, the explicit methods are conditionally better in terms of computation time.

5 Conclusion

In this paper, we developed and compared four finite difference schemes, two semi-implicit and two explicit, for solving the nonlinear reactive transport equation. A summary of some of our findings is provided in Table 6. No strategy was a clear winner over the others, but the implicit methods held a slight advantage because they had no time step restriction. Between the two implicit methods, the Backward Euler solution scheme was moderately better in one-dimension because of the shorter computation times and simpler derivation and computation. However, the explicit methods had extremely short computation times when a larger timestep could be taken as was the case with the simulation comparison to experimental data. Between the two explicit methods, Forward Euler was better in one dimension as it ran more quickly and was more versatile than Improved Euler. Thus, while Backward Euler is best overall for the purpose in this paper, Forward Euler is the better of the two Explicit Methods. In general, which strategy is best for a given situation depends on the needs of the end user as is illustrated in Table 6.

If the user is looking for...	**Then use...**
Less complex to implement	Forward Euler
Less complex to calculate converging error	Backward Euler
Fastest computation time	Forward Euler* or Backward Euler
Increased accuracy with faster computation time	Improved Euler* or Implicit Trapezoid
Increased accuracy with larger time steps	Implicit Trapezoid
More robust with boundary conditions	Backward Euler or Implicit Trapezoid

Table 6: A comparison of the benefits of each method. The asterisk (*) indicates the benefit holds only if stability does not require a very small timestep.

Our next steps include investigating more complex kinematics, variable velocity, higher dimensions, adaptive time-stepping schemes and other linearization techniques in an effort to produce more accurate and yet efficient simulation methods. Incorporating more complex kinematic equations such as non-instantaneous adsorption and an isotherm that incorporates multiple modes of adsorption may

result in more accurate predictions of the end behavior of the experimental data. Additionally, a variable velocity may increase the accuracy of the simulation predictions, particularly with a Danckwerts inflow condition.

With higher dimensions, the higher-order schemes may prove more useful in allowing larger time steps while preserving accuracy. Incorporating an adaptive time-stepping scheme with Improved Euler could also help the simulations achieve larger time steps while still maintaining accuracy. Given our use of semi-implicit methods in this paper, we intend to explore fully implicit methods in the future considering both built-in MATLAB® nonlinear solvers and custom programmed linearization techniques such as Newton iteration or a fixed point with Anderson Acceleration. It remains to be seen whether the increased computational cost of the more complex solution schemes will cancel out the benefit of the increased accuracy. As Dimartino, Boi, and Sarti [36] point out, an effective simulation tool for chromatography must have a model that is accurate while being as simple as possible.

Initial numerical investigations indicate that the explicit methods are stable only for small enough Δt whereas the semi-implicit methods have no requirements on Δt for stability. A theoretical stability analysis is ongoing and will be included in a subsequent paper.

6 Acknowledgements

This research was supported by the National Science Foundation under Grant No. NSF-DMS-2011911. The authors are grateful to Professor Scott Husson and Dr. Juan Wang for their work in producing the experimental data.

Bibliography

[1] Davies, N. The future of biologics. *Thepharmaletter*. (2017)

[2] Research, T. Global Biological Drugs Market to be Worth US 287,139.7 Million by 2020. (http://globenewswire.com/news-release/2014/10/20/674317/10103285/en/Global-Biological-Drugs-Market-to-be-Worth-US-287-139-7-Million-by-2020-Transparency -Market-Research.html,2014)

[3] Hiller, A. Fast Growth Foreseen for Protein Therapeutics. *Genet. Eng. Biotechn.*. **29** pp. 153-155 (2009), http://www.genengnews.com/gen-articles/fast-growth-foreseen-for-protein-therapeutics/2722/

[4] Langer, E. Focus on Efficiency: Single-use, analytical methods and downstream processing at the forefront. *Pharm. Manuf.*. **March** pp. 3-11 (2013)

[5] Bhut, B., Wickramasinghe, S. & Husson, S. Preparation of high-capacity, weak anion-exchange membranes for protein separations using surface-initiated atom transfer radical polymerization. *J. Membr. Sci.*. **325** pp. 176-183 (2008)

[6] Bhut, B. & Husson, S. Dramatic performance improvement of weak anion-exchange membranes for chromatographic bioseparations. *J. Membr. Sci.*. **337** pp. 215-233 (2009)

[7] Bhut, B., Christensen, K. & Husson, S. Membrane chromatography: Protein purification from E.colilysate using newly designed and commercial anion-exchange stationary phases. *J. Chromatogr. A*. **1217**, 4946-4957 (2010)

[8] Bhut, B., Christensen, K. & Husson, S. Membrane chromatography: Protein purification from E. Colilysate using newly designed and commercial anion-exchange stationary phases. *J. Chromatogr. A*. **1217**, 4946-3957 (2010)

[9] Chenette, H., Robinson, J., Hobley, E. & Husson, S. Development of high-productivity, strong cation-exchange adsorbers for protein capture by graft polymerization from membranes with different pore sizes. *J. Membrane Sci.*. **423424** pp. 43-52 (2012)

[10] Wang, J., Sproul, R., Anderson, L. & Husson, S. Development of multimodal membrane adsorbers for antibody purification using atom transfer radical polymerization. *Polymer*. **55**, 1404-1411 (2014)

[11] Wang, J., Wilson, A., Robinson, J., Jenkins, E. & Husson, S. A new multimodal membrane adsorber for monoclonal antibody purifications. *J. Membr. Sci.*. **492** pp. 137-146 (2015)

[12] Marsily, G. Quantitative Hydrogeology: Groundwater Hydrology for Engineers. (Academic Press,1986)

[13] Nfor, B., Noverraz, M., Chilamkurthi, S., Verhaert, P., Wielen, L. & Ottens, M. High-throughput Isotherm Determination and Thermodynamic Modeling of Protein Adsorption on Mixed Mode Adsorbents. *J. Chromatogr. A*. **1217** pp. 6829-6850 (2010)

[14] Singh, N., Husson, S., Zdyrko, B. & Luzinov, I. Surface modification of microporous PVDF membranes by ATRP. *J. Membrane Sci.*. **262** pp. 81-90 (2005)

[15] Singh, N., Wang, J., Ulbrict, M., Wickramasinghe, S. & Husson, S. Surface-initiated atom transfer radical polymerization: A new method for the preparation of polymeric membrane adsorbers. *J. Membrane Sci.*. **309** pp. 64-72 (2008)

[16] Boyer, T., Miller, C. & Singer, P. Modeling the Removal of Dissolved Organic Carbon by Ion Exchange in a Completely Mixed Flow Reactor. *Water Res.*. **42** pp. 1897-1906 (2008)

[17] Farthing, M., Kees, C., Russell, T. & Miller, C. An ELLAM Approximation for Advective-Dispersive Transport with Nonlinear Sorption. *Adv. Water Resour.*. **29** pp. 657-675 (2006)

[18] Kaur, J., Malengier, B. & Remesíková, M. Convergence of an operator splitting method on a bounded domain for a convection-diffusion-reaction system. *J. Math. Anal. Appl.*. **348** pp. 894-914 (2008)

[19] Poulain, C. & Finlayson, B. A Comparison of Numerical Methods Applied to Non-linear Adsorption Columns. *Int. J. Numer. Meth. Fl.*. **17** pp. 839-859 (1993)

[20] Remesíková, M. Solution of Convection-Diffusion Problems with Nonequilibrium Adsorption. *J. Comput. Appl. Math.*. **169** pp. 101-116 (2004)

[21] Suen, S. & Etzel, M. A mathematical analysis of affinity membrane bioseparations. *Chem. Eng. Sci.*. **47**, 1355-1364 (1992)

[22] Arbogast, T., Wheeler, M. & Zhang, N. A nonlinear mixed finite element method for a degenerate parabolic equation arising in flow in porous media. *SIAM J. Numer. Anal.*. **33** pp. 1669-1687 (1996)

[23] Wilson, A. & Jenkins, E. Numerical Simulation of Solid Phase Adsorption Models Using Time-Integrated, Up-winded Finite Element Strategies. *Comput. Sci. Eng.*. (2019)

[24] Wilson, A. & Jenkins, E. Analysis of a fully implicit SUPG scheme for a filtration and separation model. *Comp. Appl. Math.*. (2020)

[25] Wilson, A. & Jenkins, E. Towards Higher Order Methods for Nonlinear Adsorption Problems.

[26] Tarafder, A. Modeling and Multi-Objective Optimization of a Chromatographic System. *Multi-Objective Optimization In Chemical Engineering: Developments And Applications*. (2013)

[27] Yang, H., Bitzer, M. & Etzel, M. Analysis of Protein Purification Using Ion-Exchange Membranes. *Ind. Eng. Chem. Res.*. **38** pp. 4044-4050 (1999)

[28] Scopes, R. Protein Purification: Principles and Practice. (Springer-Verlag,1994)

[29] Mott, H. & Green, Z. On Danckwerts' Boundary Conditions for the Plug-Flow with Disperson/Reaction Model. *Chem. Eng. Comm.*. **202**, 739-745 (2015)

[30] Mollerup, J. A review of the thermodynamics of protein association to ligands, protein adsorption and adsorption isotherm. *Chem. Eng. Technol.*. **31**, 864-874 (2008)

[31] Strikwerda, J. Finite Difference Schemes and Partial Differential Equations. (SIAM,2004)

[32] Cheney, E. & Kincaid, D. Numerical Mathematics and Computing. (Brooks Cole,2007)

[33] Suen, S. & Etzel, M. A mathematical analysis of affinity membrane bioseparations. *Chem. Eng. Sci.*. **47**, 1355-1364 (1992)

[34] Agarwal, N., Semmens, M., Novak, P. & Hozalski, R. Zone of influence of a gas permeable membrane system for delivery of gases to groundwater. *Water Resour. Res.*. **41** (2005)

[35] Xu, M. & Eckstein, Y. Statistical analysis of the relationships between dispersivity and other physical properties of porous media. *Hydrogeol. J.*. **5**, 4-20 (1997)

[36] Dimartino, S., Boi, C. & Sarti, G. A validated model for the simulation of protein purification through affinity membrane chromatography. *J Chrom A.* **1218** pp. 1677-1690 (2022)

[37] Riske, F. & Ransohoff, T. Development of Continuous Capture Steps in Bioprocess Applications. *Preparative Chromatography For Separation Of Proteins.* (2017)