DESIGN OF AN EFFECTIVE NUMERICAL METHOD FOR A REACTION-DIFFUSION SYSTEM WITH INTERNAL AND TRANSIENT LAYERS

By

Ana Maria Soane

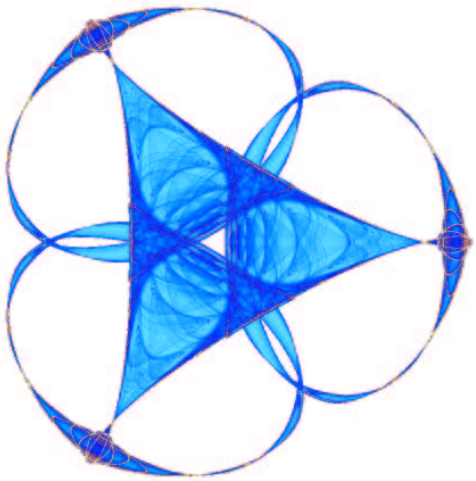
Matthias K. Gobbert

and

Thomas I. Seidman

IMA Preprint Series # 2006

(November 2004)



INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS

UNIVERSITY OF MINNESOTA

514 Vincent Hall

206 Church Street S.E.

Minneapolis, Minnesota 55455–0436

Phone: 612/624-6066 Fax: 612/626-7370

URL: http://www.ima.umn.edu

Design of an Effective Numerical Method for a Reaction-Diffusion System with Internal and Transient Layers[1]

Ana Maria Soane, Matthias K. Gobbert, and
Thomas I. Seidman

Department of Mathematics and Statistics, University of Maryland, Baltimore County,
1000 Hilltop Circle, Baltimore, MD 21250, U.S.A.,
Email addresses: {asoane,gobbert,seidman}@math.umbc.edu

Abstract
A reaction pathway for a classical two-species reaction is considered with one reaction that is several orders of magnitudes faster than the other. To sustain the fast reaction, the transport and reaction effects must balance in such a way as to give an internal layer in space. For the steady-state problem, existing singular perturbation analysis rigorously proves the correct scaling of the internal layer. This work reports the results of exploratory numerical simulations that are designed to provide guidance for the analysis to be performed for the transient problem. The full model is comprised of a system of time-dependent reaction-diffusion equations coupled through the non-linear reaction terms with mixed Dirichlet and Neumann boundary conditions. In addition to internal layers in space, the time-dependent problem possesses an initial transient layer in time. To resolve both types of layers as accurately as possible, we design a finite element method with analytic evaluation of all integrals. This avoids all errors associated with the evaluation of the non-linearities and allows us to provide an analytic Jacobian matrix to the implicit time stepping method in the software package MATLAB. The simulation results show that the method resolves the localized sharp gradients accurately and can predict the scaling of the internal layers for the time-dependent problem. A comparison between our code and the established finite element package FEMLAB confirms the accuracy of our code. It also illustrates that our specialized implementation solves the problem significantly faster and requires substantially less memory.

Keywords: Reaction-diffusion equation, internal layer, transient layer, method of lines, finite element method.

MSC (2000): 35K57, 65M50, 65M60, 76N20.

[1]The second author wishes to thank the Institute for Mathematics and its Applications (IMA) at the University of Minnesota for its hospitality during Fall 2004. The IMA is supported by funds provided by the U.S. National Science Foundation. We also gratefully acknowledge the financial support from the University of Maryland, Baltimore County through a summer faculty fellowship and research assistant support.

1 Introduction

We consider a classical chemical reaction between two species A and B to form a product, generically denoted by $(*)$, according to the reaction mechanism $2\mathrm{A}+\mathrm{B} \to (*)$. This reaction takes place within a thin membrane between 'tanks' with abundant supplies of A to the left and of B to the right of the membrane. We model the transport inside the membrane as diffusive, thus the model will be given by a system of reaction-diffusion equations that are coupled through the non-linear reaction terms.

This problem is intriguing mathematically, if one considers a more detailed model of the reaction pathway involving an intermediate species C that is generated by a 'fast' reaction wherever A and B coexist and depleted comparably 'slowly' by reacting with A to form the product:

$$\begin{aligned} \mathrm{A} + \mathrm{B} &\xrightarrow{\lambda} \mathrm{C}, \\ \mathrm{A} + \mathrm{C} &\xrightarrow{\mu} (*). \end{aligned}$$

Here, λ, μ are the rate coefficients with the units scaled so that $\lambda \gg \mu = 1$, making the first reaction 'fast' relative to the second one. This model implies that wherever A and B coexist the fast reaction depletes them both until only one is left. After this rapid transient period, the continued reaction between A and B relies on diffusion to supply the reactants and will take place only at interface between regions of the two reactants. For the generation of the product to continue at a steady-state, a particular balance between the reactions and diffusion is thus necessary. We intend to attack this problem from both analytic and numerical angles. The rigorous singular perturbation *analysis* for the *steady-state problem* was provided by Seidman and Kalachev [5, 12]. This paper presents results of the *numerical* approach for the *time-dependent problem* that allow the exploration of conjectures about the system's behavior before attempting the rigorous analysis.

1.1 The model

Assuming equal diffusion coefficients for the three species, it is possible to choose units to simultaneously scale the thickness of the membrane, the slower reaction rate, and the diffusion coefficients to 1. The balance of reactions and diffusion over time is then described by the coupled system of non-linear reaction-diffusion equations

$$\left.\begin{aligned} u_t &= u_{xx} - \lambda uv - uw, \\ v_t &= v_{xx} - \lambda uv, \\ w_t &= w_{xx} + \lambda uv - uw \end{aligned}\right\} \quad \text{in } \Omega = (0,1), \tag{1}$$

where $u(x,t)$, $v(x,t)$, $w(x,t)$ denote the concentrations of the chemical species A, B, C, respectively. We assume that the species A is supplied with a given fixed concentration $\alpha > 0$ at $x = 0$, and species B with $\beta > 0$ at $x = 1$. No species flows through any other part of the boundary. This results in the mixed Dirichlet and Neumann boundary conditions

$$\begin{aligned} &u = \alpha, \quad v_x = 0, \quad w_x = 0 \quad \text{at } x = 0, \\ &u_x = 0, \quad v = \beta, \quad w_x = 0 \quad \text{at } x = 1. \end{aligned} \tag{2}$$

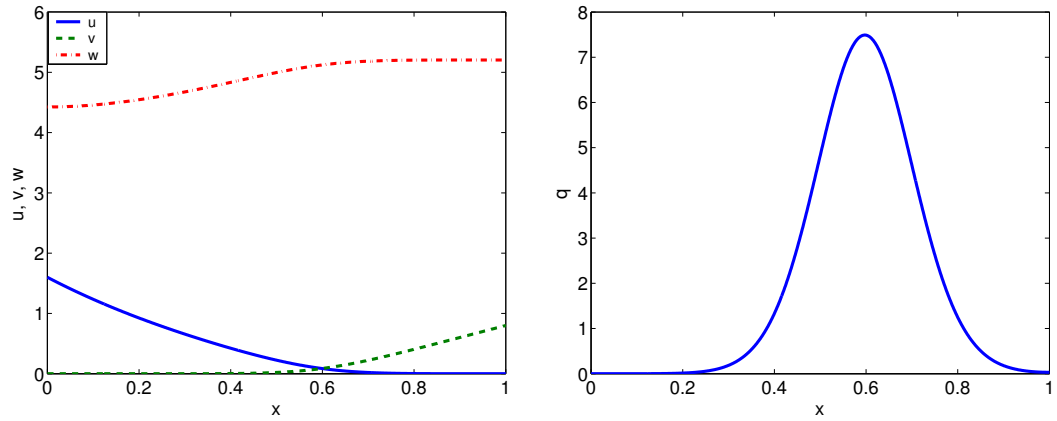
We assume that the non-negative initial concentrations are given

$$u = u_{ini}(x), \quad v = v_{ini}(x), \quad w = w_{ini}(x) \quad \text{at } t = 0. \tag{3}$$

The product is not explicitly tracked in the differential equations. We assume that the boundary and initial data are posed consistently, that is, $u_{ini}(0) = \alpha$ and $v_{ini}(1) = \beta$.

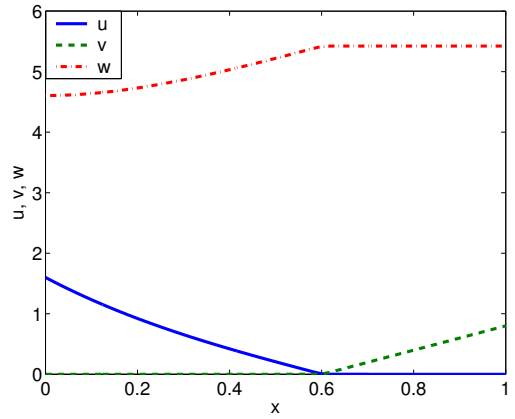
Due to the appearance of the large factor $\lambda \gg 1$ in one of the terms in each reaction-diffusion equation in (1), the equations have features of singularly perturbed problems. A standard form for such problems features a small factor as diffusion coefficient in front of the u_{xx} term. For a general introduction to singularly perturbed convection-diffusion and reaction-diffusion problems and their numerical methods, we refer to [10]. A more recent paper [7] presents a coupled system of stationary singularly perturbed reaction-diffusion equations and shows convergence of a finite difference discretization on a non-uniform mesh of Shishkin type for this problem uniformly in the perturbation parameter. Our problem (1) is distinguished from those by its non-standard formulation, in which the spatial and time derivatives are of the same order of magnitude and a reaction-term is large, which results in the appearance of internal layers that move in time. In addition, this problem is 'more' singular in the sense that its reduced problem is just the algebraic condition $uv = 0$ [5, 12].

We review now in more detail what is known about the stationary problem given by reaction-diffusion equations (1) accompanied by the boundary conditions (2), but with time-derivatives removed. The existence of a steady-state solution, and the convergence as $\lambda \to \infty$ to the unique solution of the associated limit problem, was proved in [12]. As a general reference on the analysis techniques, see [16]. In [5], a formal asymptotic expansion of the steady-state solution is constructed, using the methods of singular perturbation theory, and the rate of convergence for the results in [12] is established. In [12], it is also shown that, at steady-state, for each λ large enough there exist some $x^* = x^*(\lambda) \in (0, 1)$ such that $u(x) \geq v(x)$ for $0 \leq x < x^*$ and $u(x) \leq v(x)$ for $x^* < x \leq 1$. At this location x^*, an internal layer occurs in the reaction rate $q := \lambda uv$ of the fast reaction with width $\mathcal{O}(\varepsilon)$ and height $\mathcal{O}(1/\varepsilon)$, where the scaling is given by $\varepsilon = \lambda^{-1/3}$. This behavior is observable in Figure 1 for simulations with $\alpha = 1.6$ and $\beta = 0.8$. Figures 1 (a), (c), and (e) in the left column show the solution components $u(x)$, $v(x)$, $w(x)$ as functions of x for $\lambda = 10^3$, 10^6, and 10^9, respectively. The interface of the region of $u(x) \geq v(x)$ with $u(x) \leq v(x)$ is in this case located at $x^* \approx 0.6$. In the right column, Figures 1 (b), (d), and (f) plot the reaction rate $q(x) = \lambda u(x)v(x)$ of the fast reaction against x for $\lambda = 10^3$, 10^6, and 10^9, respectively. At the interface point x^*, the fast reaction rate $q(x)$ has a spike. The numerical results show qualitatively that the width of the spike decreases and its height increases with $\lambda \to \infty$, as predicted by the theory. To check that the numerical results reproduce the analytically known scaling quantitatively predicted, Figure 2 plots the scaled rate $\tilde{q}(\xi) := \varepsilon\, q(x)$ against the shifted and scaled coordinate $\xi := (x - x^*)/\varepsilon$ with $\varepsilon = \lambda^{-1/3}$. Since the curves for the three values of λ agree very well, the numerical results confirm the scaling obtained analytically in [5].

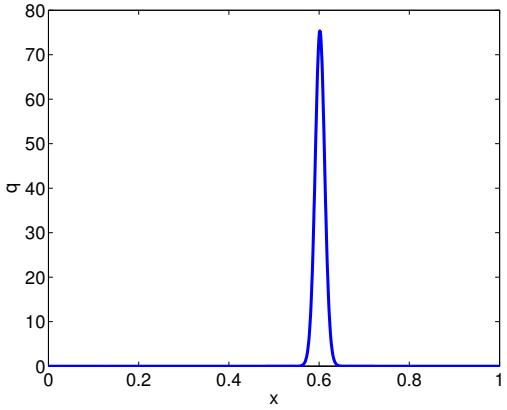


(a) $\lambda = 10^3$

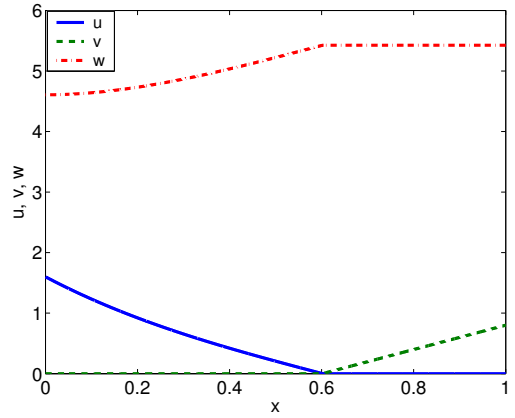
(b) $\lambda = 10^3$



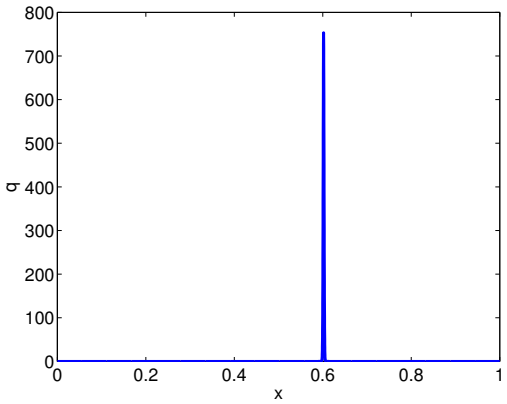
(c) $\lambda = 10^6$



(d) $\lambda = 10^6$



(e) $\lambda = 10^9$



(f) $\lambda = 10^9$

Figure 1: (a), (c), (e) Solutions of the stationary problem u, v, and w and (b), (d), (f) reaction rate $q = \lambda uv$ for (a) and (b) $\lambda = 10^3$, (c) and (d) $\lambda = 10^6$, and (e) and (f) $\lambda = 10^9$. Notice the different scales of the vertical axes in plots (b), (d), and (f).

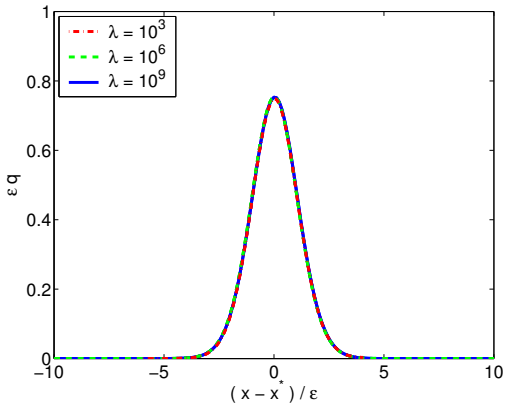


Figure 2: Plot of the scaled fast reaction rate εq vs. the scaled and shifted position $(x-x^*)/\varepsilon$ for $\lambda = 10^3, 10^6, 10^9$.

Using the knowledge about the behavior of the stationary problem, we now formulate some conjectures about the behavior of the time-dependent problem:

1. Analogously to the behavior of the system at steady-state, we expect that the solution of the time-dependent system will also exhibit one or more internal layers.

2. If the species A and B coexist initially, we expect them to react locally during a rapid transient layer (of duration $\mathcal{O}(1/\lambda)$). After this initial layer, the domain is partitioned into subintervals with either $u \approx 0$ or $v \approx 0$, separated by interfaces.

3. During a secondary initial phase (of duration $\mathcal{O}(1)$ with respect to λ), we expect these interfaces to move smoothly and to coalesce (in pairs) over time until only one interface is left. Based on chemical interpretation of the model, we do not expect that any new interfaces will be created over time.

4. Once only one interface is left (or if the initial condition only allowed for one interface), we expect it to move over time to the steady-state position x^* for the value of λ used. That is, we expect the transient solution to converge to the known steady-state solution. The exact motion of the position(s) of the interface(s) over time is not clear at present.

Notice that there are only partial analytical results for the time-dependent problem (1)–(3) available at this point. For instance, existence can be established (for any fixed λ), a lower bound $0 \leq u(x,t), v(x,t), w(x,t)$ can be derived from a maximum principle, and upper bounds can be established for $u(x,t)$ and $v(x,t)$, though not for $w(x,t)$ (at present). The numerical results are intended to provide significant insight into the behavior as a guide to analytic studies yet to be performed. For instance, it is of interest to confirm numerically the correct scaling of the internal layers in the transient problem [4].

1.2 The numerical method

The problem (1)–(3) is a coupled system of non-linear reaction-diffusion equation. On the one hand, it is challenging numerically, because we expect to see both steep gradients in

space and rapid transients in time. On the other hand, these are localized in space and in time, respectively. Moreover, the problem is posed on a one-dimensional domain and the non-linearity is only of quadratic type (products uv and uw) and only in the reaction terms (referring to the form that (1) is written in). Since the problem arose originally in the context of a student consulting project in a second-year graduate class, for which only half a semester time was available to deliver a complete solution to the client, one crucial consideration was to leverage existing reliable and powerful software as much as possible to speed up the real-time solution process. Therefore, we use a method of lines approach, in which we derive a spatial discretization only and thereby obtain a coupled system of non-linear ordinary differential equations, which in turn is integrated in time using an existing reliable and efficient software.

For the spatial discretization, we use the finite element method with piecewise linear basis functions. For simplicity, a uniform mesh is used. We accept at this point that a very fine mesh will be necessary to capture the behavior of the system inside the internal layer(s) for large values of λ. However, this has the advantage that we can capture the layer(s) with the same accuracy, no matter where each might be located at a certain point in time. Given this advantage of simplicity and reliability, a uniform mesh is appropriate. The necessary fine resolution can be easily achieved on computers available today for all λ values of interest. Using the method of lines, we reduce our system to an ODE system in time. To solve this stiff system, we use the `ode15s` solver [13] in MATLAB [8] with automatic step-size and order control, that is suitable to capture the initial transient layer both accurately and efficiently. Additionally, we avoid the introduction of any additional numerical error by evaluating all integrals arising in the finite element method (including those arising from the non-linearity of the reaction terms on the right-hand side of (1)) analytically. This allows us to supply the mass matrix, stiffness matrix, and the Jacobian of the ODE system analytically to the ODE solver `ode15s`. In summary, the method is both very accurate and leverages readily available software effectively, solving the problem (1)–(3) in fewer than 250 lines of MATLAB code. Section 2 presents additional considerations for the choices made and provides the details of the numerical method.

It should be mentioned that another approach would have been to use a commercial software package such as FEMLAB 2.3 [2] to solve the problem (1)–(3) directly. In fact, one reason for the choice of the finite element method was the ability to check the solution against this established software package. Therefore, we restrict our code to a standard finite element formulation and to MATLAB's `ode15s`, equivalent to available components in FEMLAB 2.3, so we are able to check our code against FEMLAB. We find that, for instance, the number of time steps taken are equal, giving confidence that the codes give equivalent solutions.

One downside resulting from following FEMLAB's approach is that neither it nor our method can guarantee the non-negativity of solutions. We have tracked the minimum value of the numerical solutions and have observed small negative values of the same magnitude for both codes for large values of λ. Computationally, tests with a decreasing tolerance of the ODE method lead to decreasing magnitudes of negative values, hence we are able to control those numbers. Analytically, we know that our spatial discretization with a lumped mass matrix together with the implicit Euler method in time does guarantee the non-negativity of the solution [3, 11]. But it turns out that even when restricting MATLAB's `ode15s` to

the implicit Euler method, small negative values in the solution persist. This is the result of the non-linear solver, a simplified Newton method, inside `ode15s` that is not especially designed to preserve non-negativity; this behavior agrees with the prediction in [11]. We note also that restricting `ode15s` to method order $k = 1$ is costly computationally. Since even this cannot guarantee non-negativity, we stick with the higher efficiency of a variable order method, in agreement with FEMLAB, for now, in order to retain the ability to compare our method directly to FEMLAB. This is an interesting line of research for the future, for which we will be able to take advantage of our own code and design a different formulation that guarantees the non-negativity of the solution.

Additionally, for problems such as these with internal layers with rapidly varying quantities, it would clearly be desirable to use adaptive mesh refinement and coarsening so as to use a fine mesh exactly in those regions of Ω around the internal layers. For instance, singularly perturbed stationary convection-diffusion equation in standard form are considered in [6, 14], that propose moving mesh algorithms (with fixed number of nodes) and present rigorous convergence analyses and numerical examples of the methods uniformly in the perturbation parameter. An adaptive method for a time-dependent heat equation in one spatial dimension is presented, for instance, in [1]. The results in these references cannot be applied directly to our problem, as it is a posed in a different, non-standard form. (Notice that due to the unknown and moving location of the internal layers, it is not possible to design a specialized, but fixed in time, mesh of Shishkin type a priori.) For the stationary problem, the results in Figure 1 were checked against FEMLAB 2.3 with adaptivity; the adaptive method demonstrated its effectiveness by using more than an order of magnitude fewer nodes than the uniform mesh. But FEMLAB 2.3 does not have an adaptive solver for the time-dependent problems at present, so this was not an option available at this time. The use of any other software or development thereof was not possible in the time-frame of a consulting class project.

Hence, this work uses a uniform over-discretization of the domain and does not propose a method that converges uniformly in λ. The results on this fine mesh will be useful in the future to assess the accuracy of alternative, in particular, adaptive methods. The desire to generalize our code to non-uniform meshes in the future is another reason for our insistence on hand coding the spatial discretization for our numerical method. Moreover, higher-dimensional formulations of the problem (1)–(3) are of interest in applications, and our code will control the use of memory better than a black-box package.

1.3 The simulation results

We consider the problem (1)–(3) on $\Omega = (0, 1)$ for times $0 \leq t \leq 10$. This final time for the time-dependent simulations proves sufficient to approximate steady-state. We perform simulations for three values of the fast rate coefficient $\lambda = 10^3, 10^6, 10^9$ in an effort to analyze the limiting behavior $\lambda \to \infty$. Section 3 presents the detailed results of the simulations; this section can be read independently of Section 2. The results obtained allow us to describe the behavior of the system as follows:

- Section 3.1 considers three types of initial conditions with increasing degree of complexity. If species A and B coexist initially at a point in the domain, the fast reaction

depletes both reactants in a rapid initial transient layer, until only one of the species remains at this point. This confirms the conjecture about the smooth behavior of the interfaces and their monotone coalescence. If the initial condition had several, say, four regions where alternately $u > v$ and $u < v$, then three interfaces between regions with $v \approx 0$ and $u \approx 0$ develop during this transition. The fast reaction rate $q = \lambda uv$ is zero wherever either $u \approx 0$ or $v \approx 0$ and has large values inside the interfaces, where non-vanishing concentrations of A and B meet. Outside of this initial transient layer, the *width* of the internal layers is on the same order of magnitude as at steady-state. In a second phase of the system evolution, all but one of the interfaces coalesce in pairs. After this phase, the remaining interface has already the same *width* and *height* of the interface at steady-state, but still moves over time to its steady-state location. The results validate that the transient solution tends to the steady-state. We note that this result is not obvious as no proof exists for the fact that $w(x, t)$ stays bounded over time. One additional—and unexpected—observation is that the remaining interface, after all but one have coalesced, might overshoot the location of the steady-state interface initially.

- When combining in Section 3.2 the results for all values of λ studied, we can confirm that after the coalescing phase the remaining internal layer has the *width* $\mathcal{O}(\varepsilon)$ and *height* $\mathcal{O}(1/\varepsilon)$ with the same scaling $\varepsilon = \lambda^{-1/3}$ as the steady-state problem. This result has not yet been shown analytically. In this paper, we have not analyzed the durations of the initial transient and coalescence phases, yet. Neither have simulations for additional initial conditions or for different values of α and β been performed that might lead to different motions of the interfaces; but we do believe that the results presented here are typical and show all qualitative features to be expected.

- In Section 3.3, some numerical convergence and performance studies are summarized. The convergence results confirm the theoretically expected order of convergence of the numerical method. Further, we document how the NDFk method in MATLAB's ODE solver `ode15s` automatically selects the step size Δt and the method order k. Since the time steps need to be much smaller in the initial transient layer than for larger times, this illustrates that is advantageous to use a sophisticated solver for problems with transient layers. Finally, we present a comparison of our code using MATLAB with the software package FEMLAB 2.3. We demonstrate that our code is about 3 times faster and uses about 40% less memory than FEMLAB in each case considered.

2 The numerical method

To discretize the reaction-diffusion equations (1) in space, we use the finite element method with linear basis functions, although reaction-diffusion equations are also often discretized in space by finite difference or finite volume methods [10]. In view of the steady-state results in Figure 1, we made our choice to try to require as little regularity for the solutions as possible. For linear basis functions, we expect the numerical solution to show second-order convergence in the L^2-norm, provided the mesh can resolve the internal layer, that is, the mesh spacing h satisfies $h < \varepsilon = \lambda^{-1/3}$; at this point, we do not seek a method that is convergent uniformly in λ. The choice of linear basis functions, as opposed to higher-order, more complicated elements, was made to ease the analytic computation of all terms and their Jacobian for the ODE solver; see below. At the same time, linear finite elements on a fine mesh are also a reasonable choice to capture behavior as seen in Figure 1.

The finite element discretization involves integrals of the non-linear reaction terms on the right-hand side of (1). Usually, these integrals would require a numerical approximation, introducing additional error in the neighborhood of the anticipated steep gradients. Instead, we take advantage of the special form of the non-linearity and obtain the integrals analytically, avoiding this source of error. Additionally, knowing all terms analytically allows us to supply an analytic Jacobian to the ODE method for best accuracy and best efficiency of its integrated non-linear solver, a simplified Newton method. Finally, using an ODE solver with variable time-stepping and variable order is shown to handle the rapid initial transient layer efficiently.

At this point, we take advantage of the fact that the problem is posed only on a one-dimensional domain, so it is easily possible on today's computers to resolve $\Omega = (0, 1)$ with a sufficiently fine uniform mesh. For instance, for the largest value $\lambda = 10^9$ of interest, we anticipate the interface to have width of order $\varepsilon = \lambda^{-1/3} = 10^{-3}$. Using the rule of thumb that we wish to place at least 8 points inside this layer, we need about 8,000 points to cover Ω. One option is to apply the parabolic solver in the software FEMLAB 2.3 [2] directly to the problem (1)–(3). We chose, however, to develop the finite element discretization ourselves, for reasons discussed in the Introduction, and then call on the stiff ODE solver `ode15s` in the standard software MATLAB [8] to handle the semi-discrete problem. In this way, we are able to control the order of variables in the ODE system better, and our method turns out to be about a factor 3 faster and to use 40% less memory than FEMLAB 2.3.

We now present briefly the details of how the finite element discretization is used in the method of lines to obtain the system of ordinary differential equations; we use the equation for $w(x, t)$ in (1) as example. Define the spatial mesh with N nodes with uniform spacing $h = 1/(N - 1)$ so we get the mesh $0 = x_1 < x_2 < \cdots < x_N = 1$. Let $\varphi_i : \Omega \to \mathbb{R}$, $i = 1, \ldots, N$, denote the piecewise linear nodal basis functions on this mesh; that is, $\varphi_i(x)$ is a linear function on each mesh element $[x_i, x_{i+1}]$ and satisfies $\varphi_i(x_i) = 1$ for all i and $\varphi_i(x_j) = 0$ for all $i \neq j$. We wish to obtain approximations as expansions in these basis functions

$$w(x, t) \approx w_h(x, t) := \sum_{j=1}^{N} \mathbf{w}_j(t)\, \varphi_j(x) \tag{4}$$

and analogously for $u(x, t)$ and $v(x, t)$.

For convenience, we recollect the equation for $w(x,t)$ from (1) along with its boundary conditions and initial conditions

$$\begin{aligned} &w_t - w_{xx} = f^{(w)} && \text{in } \Omega = (0,1), \\ &w_x = 0 && \text{on } \partial\Omega, \\ &w = w_{ini}(x) && \text{at } t = 0, \end{aligned} \tag{5}$$

where $f^{(w)} = \lambda uv - uw$ for short. (The term $-uw$ is formally taken as part of the right-hand side here. For analysis purposes, it might be more appropriate to move it as uw to the left-hand side. However, since we will derive analytical expressions for all terms in the semidiscretization, this does not make any difference to the numerical method.) Multiply the partial differential equation in (5) by a test function $\varphi : \Omega \to \mathbb{R}$ and integrate over Ω to obtain

$$\int_0^1 \varphi\, w_t \, dx - \int_0^1 \varphi\, w_{xx} \, dx = \int_0^1 \varphi\, f^{(w)} \, dx.$$

We use integration by parts and the no-flux boundary conditions to obtain the weak formulation of (5) on which the finite element method is based: Find $w : \Omega \times [0, t_{fin}] \to \mathbb{R}$ with $w(\cdot,t) \in \mathcal{A}^{(w)}$ and $w_t(\cdot,t) \in \mathcal{A}^{(w)}$ for all t such that

$$\int_0^1 \varphi\, w_t \, dx + \int_0^1 \varphi_x\, w_x \, dx = \int_0^1 \varphi\, f^{(w)} \, dx \quad \forall \varphi \in \Phi^{(w)}. \tag{6}$$

Here, both trial function space $\mathcal{A}^{(w)}$ and test function space $\Phi^{(w)}$ are chosen as the Sobolev space $H^1(\Omega)$. Substituting (4) into (6) and choosing $\varphi = \varphi_i$ for $i = 1, \ldots, N$, we obtain the system of N ordinary differential equations

$$\sum_{j=1}^{N} \int_0^1 \varphi_i \varphi_j \, dx \, \frac{d\mathbf{w}_j}{dt} + \sum_{j=1}^{N} \int_0^1 (\varphi_i)_x (\varphi_j)_x \, dx \; \mathbf{w}_j = \int_0^1 \varphi_i \, f^{(w)} \, dx, \quad i = 1, \ldots, N, \tag{7}$$

for the solution coefficients $\mathbf{w}_j(t)$, $j = 1, \ldots, N$. We introduce the standard definitions of the mass matrix $M^{(w)} \in \mathbb{R}^{N \times N}$, whose elements are defined as $M^{(w)}_{ij} = \int_0^1 \varphi_i \varphi_j \, dx$, and the stiffness matrix $K^{(w)} \in \mathbb{R}^{N \times N}$ with elements defined by $K^{(w)}_{ij} = \int_0^1 (\varphi_i)_x (\varphi_j)_x \, dx$. We also define the right-hand side vector $F^{(w)} \in \mathbb{R}^N$ with component functions $F^{(w)}_i = \int_0^1 \varphi_i f^{(w)} \, dx$. Using this notation, the semi-discrete system of ordinary differential equation can be written in vector form as

$$M^{(w)} \frac{d\mathbf{w}}{dt} = -K^{(w)} \mathbf{w} + F^{(w)} \tag{8}$$

for the vector of coefficient functions $\mathbf{w} = (\mathbf{w}_j(t))_{j=1,\ldots,N}$. Here, $M^{(w)}$ and $K^{(w)}$ are constant matrices and can be computed analytically as

$$M^{(w)} = \frac{h}{6} \begin{bmatrix} 2 & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 2 \end{bmatrix}, \qquad K^{(w)} = \frac{1}{h} \begin{bmatrix} 1 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 1 \end{bmatrix}. \tag{9}$$

In the derivation of the corresponding semi-discretization for the approximations to $u(x,t)$ and $v(x,t)$, the weak formulations require different trial and test function spaces instead of $H^1(\Omega)$ in (6) to account for the Dirichlet boundary condition at one of the endpoints of Ω. For instance, the trial space for $u(\cdot,t)$ is $\mathcal{A}^{(u)} := \{u \in H^1(\Omega),\, u(0) = \alpha\}$ and the test function space for φ is $\Phi^{(u)} := \{\varphi \in H^1(\Omega),\, \varphi(0) = 0\}$. Hence, the approximating equation corresponding to (7) is applicable only for test functions φ_i, $i = 2, \ldots, N$, excluding φ_1 due to the Dirichlet boundary condition at $x = 0$. The boundary condition $u(0,t) = \alpha$ can be cast in the form of an ODE

$$\frac{d\mathbf{u}_1}{dt} = 0, \quad \mathbf{u}_1(0) = \alpha \tag{10}$$

and used as equation for the component $\mathbf{u}_1(t)$. Then, the system of coupled ordinary differential equations for the vector of coefficient functions $\mathbf{u} = (\mathbf{u}_j(t))_{j=1,\ldots,N}$, reads

$$M^{(u)}\,\frac{d\mathbf{u}}{dt} = -K^{(u)}\,\mathbf{u} + F^{(u)}, \tag{11}$$

with mass matrix $M^{(u)} \in \mathbb{R}^{N\times N}$ and stiffness matrix $K^{(u)} \in \mathbb{R}^{N\times N}$ identical to $M^{(w)}$ and $K^{(w)}$ in (9), respectively, except that their first rows are modified to implement the ODE (10) for the first coefficient function $\mathbf{u}_1(t)$. Finally, an analogous system for the vector of coefficient functions $\mathbf{v} = (\mathbf{v}_j(t))_{j=1,\ldots,N}$ is derived as

$$M^{(v)}\,\frac{d\mathbf{v}}{dt} = -K^{(v)}\,\mathbf{v} + F^{(v)} \tag{12}$$

with $M^{(v)}, K^{(v)} \in \mathbb{R}^{N\times N}$, where the last row is modified from (8) to represent

$$\frac{d\mathbf{v}_N}{dt} = 0, \quad \mathbf{v}_N(0) = \beta. \tag{13}$$

We still need to apply the approximations $u \approx u_h$, $v \approx v_h$, and $w \approx w_h$ to, for instance, the right-hand side $F_i^{(w)} = \int_0^1 \varphi_i f^{(w)}\,dx$ of (8) with $f^{(w)} = \lambda uv - uw$. We note at this point that all right-hand side functions $F^{(u)}$, $F^{(v)}$, and $F^{(w)}$ in (11), (12), and (8) depend on *all* approximations u_h, v_h, and w_h; therefore, while we derived each ODE system of order N separately for clarity of presentation, it is clear that they must be solved simultaneously as one larger system of $3N$ ordinary differential equations. In the approximation $F^{(w)}$, we take advantage of the known particular form of the right-hand side function $f^{(w)}$ and the simple form of the piecewise linear basis functions to compute the vector $F^{(w)}$ analytically as

$$F^{(w)} = \frac{h}{12}\begin{bmatrix} \lambda\,(3\mathbf{u}_1\mathbf{v}_1 + \mathbf{u}_1\mathbf{v}_2 + \mathbf{u}_2\mathbf{v}_1 + \mathbf{u}_2\mathbf{v}_2) \\ -\,(3\mathbf{u}_1\mathbf{w}_1 + \mathbf{u}_1\mathbf{w}_2 + \mathbf{u}_2\mathbf{w}_1 + \mathbf{u}_2\mathbf{w}_2) \\ \vdots \\ \lambda\,(\mathbf{u}_{i-1}\mathbf{v}_{i-1} + \mathbf{u}_{i-1}\mathbf{v}_i + \mathbf{u}_i\mathbf{v}_{i-1} + 6\mathbf{u}_i\mathbf{v}_i \\ \quad + \mathbf{u}_i\mathbf{v}_{i+1} + \mathbf{u}_{i+1}\mathbf{v}_i + \mathbf{u}_{i+1}\mathbf{v}_{i+1}) \\ -\,(\mathbf{u}_{i-1}\mathbf{w}_{i-1} + \mathbf{u}_{i-1}\mathbf{w}_i + \mathbf{u}_i\mathbf{w}_{i-1} + 6\mathbf{u}_i\mathbf{w}_i \\ \quad + \mathbf{u}_i\mathbf{w}_{i+1} + \mathbf{u}_{i+1}\mathbf{w}_i + \mathbf{u}_{i+1}\mathbf{w}_{i+1}) \\ \vdots \\ \lambda\,(3\mathbf{u}_{N-1}\mathbf{v}_{N-1} + \mathbf{u}_{N-1}\mathbf{v}_N + \mathbf{u}_N\mathbf{v}_{N-1} + \mathbf{u}_N\mathbf{v}_N) \\ -\,(3\mathbf{u}_{N-1}\mathbf{w}_{N-1} + \mathbf{u}_{N-1}\mathbf{w}_N + \mathbf{u}_N\mathbf{w}_{N-1} + \mathbf{u}_N\mathbf{w}_N) \end{bmatrix}.$$

An analogous result is obtained for $F^{(u)}$, noting that the first component $F_1^{(u)} = 0$ in order for the first equation of (11) to implement (10), and similarly for $F^{(v)}$ with $F_N^{(v)} = 0$.

The combined system of $3N$ ordinary differential equations

$$\begin{aligned} &M^{(ode)} \frac{d\mathbf{y}}{dt} = f^{(ode)}(t, \mathbf{y}), \qquad 0 < t \le t_{fin}, \\ &\mathbf{y}(0) = \mathbf{y}_0, \end{aligned} \tag{14}$$

with

$$f^{(ode)}(t, \mathbf{y}) := -K^{(ode)}\, \mathbf{y} + F^{(ode)}(\mathbf{y}), \tag{15}$$

that will be fed to the ODE solver, is fully specified once we specify the order of the component functions of $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{w}$ in the unknown vector $\mathbf{y}$. We choose to interleave their components as

$$\mathbf{y} := [\mathbf{u}_1, \mathbf{v}_1, \mathbf{w}_1, \mathbf{u}_2, \mathbf{v}_2, \mathbf{w}_2, \ldots, \mathbf{u}_N, \mathbf{v}_N, \mathbf{w}_N]^T \in \mathbb{R}^{3N}. \tag{16}$$

The right-hand side vector $F^{(ode)}(\mathbf{y})$ is interleaved analogously. The constant matrix $M^{(ode)} \in \mathbb{R}^{3N \times 3N}$ is then given by interleaving the row and column entries of $M^{(u)}$, $M^{(v)}$, and $M^{(w)}$, and analogously the constant matrix $K^{(ode)} \in \mathbb{R}^{3N \times 3N}$. Both matrices are band matrices with bandwidth 7 and efficiently stored in MATLAB's sparse storage format.

Several choices in our approach are not standard. For instance, it is customary to use Gaussian quadrature to approximate the integral in $F^{(ode)}$. Compared to that, our approach of evaluating all integrals analytically has the advantages: We obviously avoid introducing any error associated with the numerical quadrature used; this is a minor point, because one can use sufficiently accurate Gaussian quadrature to obtain exact numerical quadrature rules, whenever possible. Additionally, explicit formulas for $F^{(ode)}$ can be evaluated more rapidly than any quadrature rule.

But the key reason for our approach relates to our goal to feed (14) to a commodity ODE solver: Since the system (14) results from a method of lines discretization of a partial differential equation, it is necessarily a stiff ODE system. This requires the use of implicit time-stepping methods such as `ode15s` in MATLAB that implement the numerical differentiation formulas NDFk with order $k = 1, \ldots, 5$ with automatic time-step and order selection [13]; we select this method mostly because of its automatic time-step selection, expecting that this will deal with the expected initial transient layer efficiently and reliably.

Moreover, the system (14) consists of non-linear ordinary differential equations, and the time-stepping will involve the solution of a system of non-linear equations for the coefficients of $\mathbf{y}$ at every time step. A simplified Newton method is used in `ode15s` for this purpose that requires the Jacobian matrix $J^{(ode)} = \partial f^{(ode)}/\partial \mathbf{y} \in \mathbb{R}^{3N \times 3N}$ of the vector function on the right-hand side $f^{(ode)}(t, \mathbf{y})$ of the ODE system (14). The method `ode15s` can use either an internally computed numerical approximation to the Jacobian or a user-supplied Jacobian. Since this problem involves steep gradients in the internal layers, we want to provide the most accurate Jacobian possible. Therefore, we compute the Jacobian of the right-hand side of (14) analytically and supply it in this form to the function `ode15s`.

Finally, to explain our choice of interleaving the components of $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{w}$ in $\mathbf{y}$, we point out that the Newton method inside `ode15s` involves the solution of a linear system of equations with system matrix of the form $A := M^{(ode)} - \tau\, J^{(ode)}$ with a scalar τ, whose value depends on the time step Δt and on the order k of the NDFk method

used in the current time step [13]. MATLAB employs Gaussian elimination for this linear solve which necessarily introduces fill-in within the bandwidth of the sparse band matrix A. Therefore, the efficiency is determined by choosing an ordering of the equations that minimizes the bandwidth. With our interleaved ordering in (16), the matrix $A \in \mathbb{R}^{3N\times 3N}$ has a bandwidth of 11, constant independent of N. This results in a complexity for Gaussian elimination that is linear in N. This is in contrast to the 'natural' ordering in $\mathbf{y} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N, \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N]^T \in \mathbb{R}^{3N}$, which would result in a bandwidth of $2N + 1$ and in a complexity of N^2 for Gaussian elimination. In fact, FEMLAB 2.3 uses this ordering, and we observed results for the simulations to be a factor 3 slower and to use 40% more memory. However, FEMLAB 2.3 apparently performs a re-ordering of the equations, as well, and this re-ordering is so effective that its timings are also linear in N.

For completeness, we explain how the stationary results in Figures 1 and 2 were computed. The problem is the same as (1) without time derivatives together with boundary conditions (2). The spatial discretization uses the same finite element method with linear basis functions as the time-dependent problem. This gives a non-linear algebraic system of $3N$ equations, for which we use our own implementation of the Newton method with an analytically supplied Jacobian. Various values of N and tolerances were used to ensure the reliability of the results. The location of the interface $x^* = x^*(\lambda)$ in Figure 2 is numerically determined as the mesh point associated with the maximum value of $q = \lambda uv$.

3 Simulation results

This section reports the details of the numerical results for the time-dependent problem (1)–(3). In all cases, we use the values $\alpha = 1.6$ and $\beta = 0.8$, and compute over the time interval $0 \leq t \leq 10$. This final time for the time-dependent simulations proves sufficient to approximate steady-state. Section 3.1 reports detailed results of various computed quantities for the value of $\lambda = 10^6$. This value is chosen because, based on the steady-state results, we expect internal layers with width on the order of $\varepsilon = \lambda^{-1/3} = 0.01$ and this width is still large enough to be discernible in plots on the scale of the domain $\Omega = (0, 1)$. For this fixed value of λ, three different types of initial conditions of increasing complexity are considered. Section 3.2 combines results for three different values of $\lambda = 10^3, 10^6, 10^9$ to analyze the behavior as $\lambda \to \infty$. To this end, the crucial quantity $q = \lambda uv$ is considered. Finally, Section 3.3 summarizes both numerical convergence and performance results to give some insight into the behavior of the numerical method.

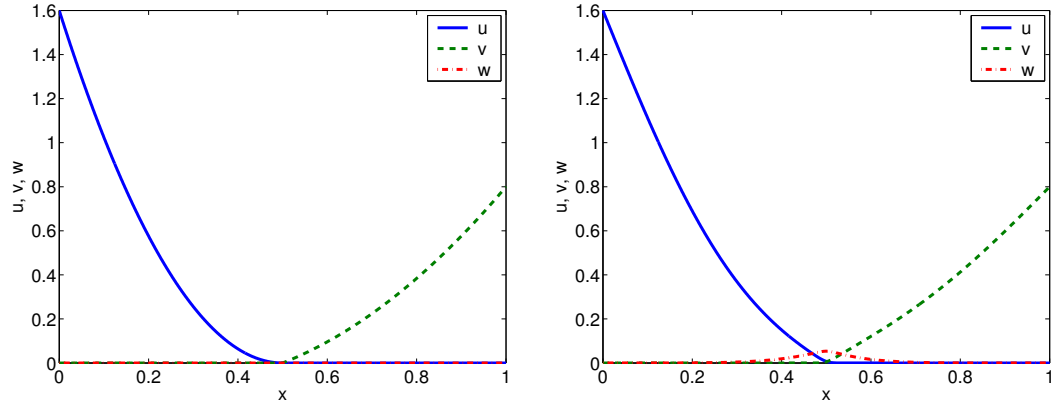
In each case, the number of nodes N of the uniform numerical mesh is chosen to guarantee at least 8 mesh points within the width of the internal layers, anticipated as $\varepsilon = \lambda^{-1/3}$. That is, we use (at least) $N = 129$ for $\lambda = 10^3$, $N = 1025$ for $\lambda = 10^6$, and $N = 8193$ for $\lambda = 10^9$. Notice that N is chosen so that the mesh size $h = 1/(N - 1)$ is a computer number to avoid unnecessary round-off in calculations with h. The choice of time step Δt and method order k of the NDFk method is performed automatically by MATLAB's `ode15s` function [8, 13]. We control the accuracy of the method by selecting relative and absolute tolerances on the estimated truncation error of the ODE method; we typically used tolerances similar to MATLAB's default choice, say, `RelTol` $= 10^{-3}$, `AbsTol` $= 10^{-6}$. In fact, simulations were performed for a range of values for N (up to 16385) and the ODE tolerances (down to 10^{-14}) and also some simulations compared to results from FEMLAB 2.3 [2]. Therefore, we feel comfortable that the simulation results are reliable. Particularly, we also monitored the minimum value of the concentrations $\min_x\{u(x,t)\}$, $\min_x\{v(x,t)\}$, and $\min_x\{w(x,t)\}$ at all times t. Small negative values were observed for the largest value of λ considered; we were able to control the magnitude of these values by tightening the tolerances of the ODE method, hence we are confident that our solutions are reliable.

3.1 Time-dependent studies for fixed $\lambda = 10^6$

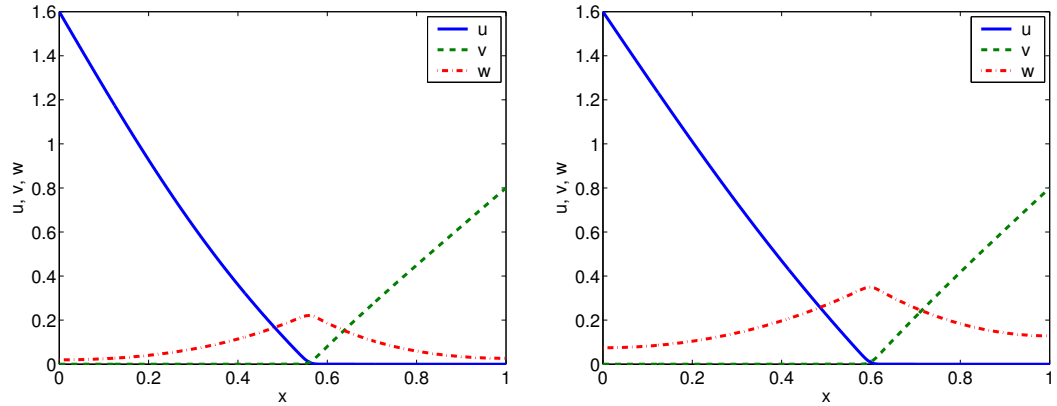
The simulations for the time-dependent problem with $\lambda = 10^6$ were performed for three types of initial conditions of increasing complexity. In all cases, the functions u_{ini}, v_{ini}, and w_{ini} are chosen to satisfy the boundary conditions. Also, since we are interested in internal layers at present, $u_{ini}(1)$ and $v_{ini}(0)$ are chosen zero to avoid additional transient layers near the boundary in the approach to steady-state. For all three types of initial conditions we assume that the species C is not present in the domain initially, i.e., we choose $w_{ini} \equiv 0$.

3.1.1 Type 1 initial condition: two non-overlapping regions

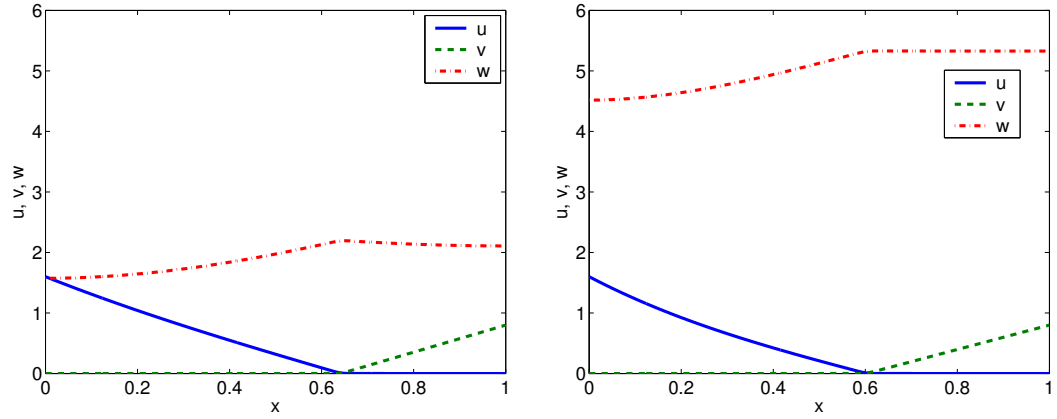
The simplest type of initial condition is inspired by the steady-state results in Figure 1, in which the species A and B do not coexist at steady-state, that is, they occupy non-overlapping geometric regions and with only a single interface. So, we choose also in initial



(a) $t = 0$ (b) $t = 0.009$

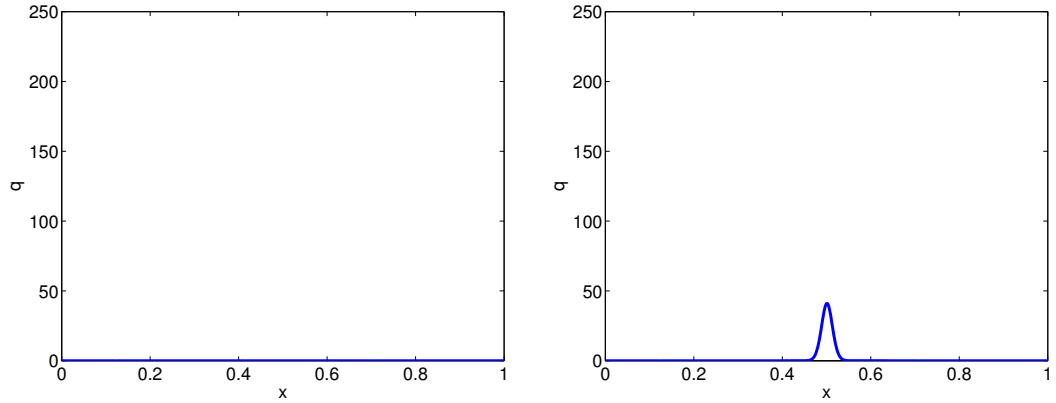


(c) $t = 0.052$ (d) $t = 0.098$



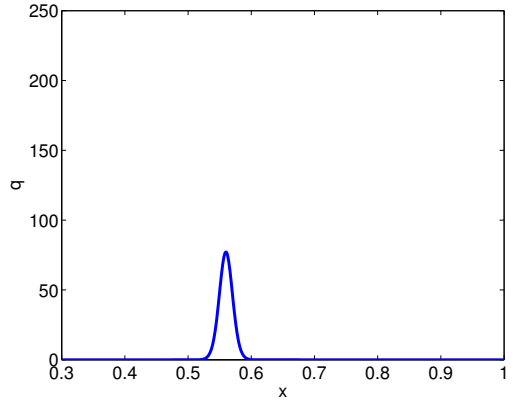
(e) $t = 1.053$ (f) $t = 10$

Figure 3: Plot of the solutions u, v, and w for type 1 initial conditions and $\lambda = 10^6$ at different time steps: (a) $t = 0$, (b) $t = 0.009$, (c) $t = 0.052$, (d) $t = 0.098$, (e) $t = 1.053$, (f) $t = 10$. Notice the different scales on the vertical axes.

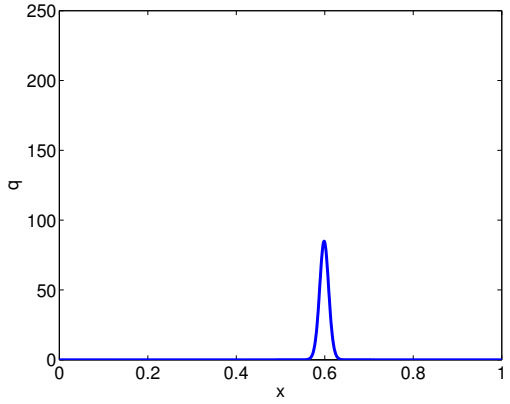


(a) $t = 0$

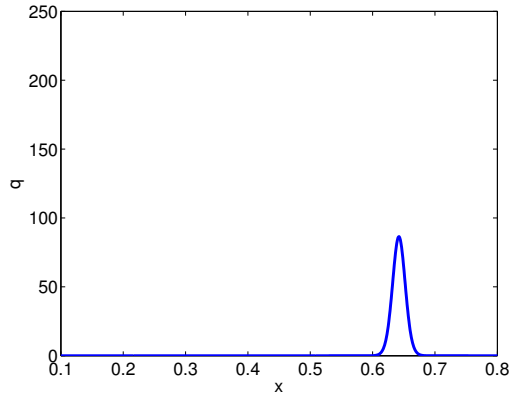
(b) $t = 0.009$



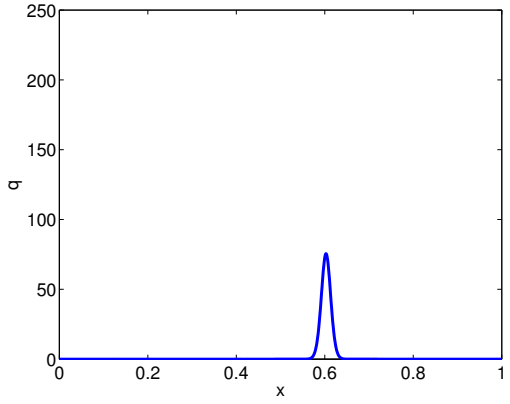
(c) $t = 0.052$



(d) $t = 0.098$

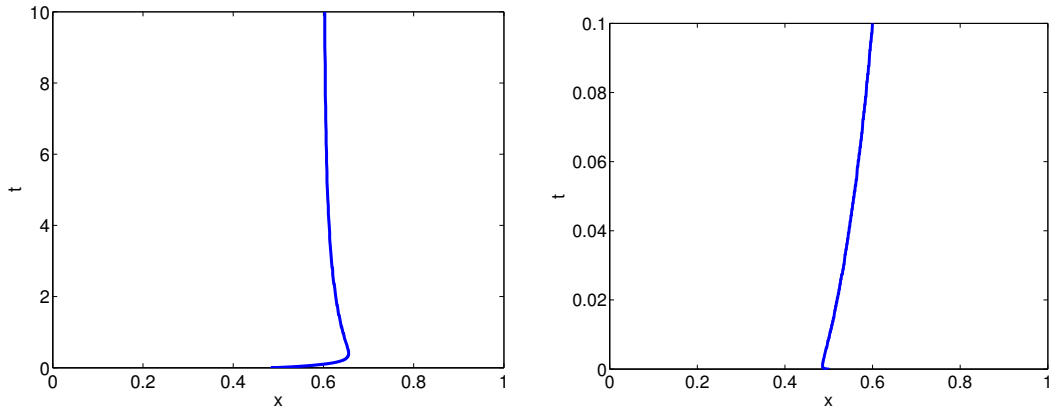


(e) $t = 1.053$



(f) $t = 10$

Figure 4: Plot of the reaction rate $q = \lambda uv$ for type 1 initial condition and $\lambda = 10^6$ at different time steps: (a) $t = 0$, (b) $t = 0.009$, (c) $t = 0.052$, (d) $t = 0.098$, (e) $t = 1.053$, (f) $t = 10$.



(a) (b)

Figure 5: Motion of internal layer for type 1 initial conditions and $\lambda = 10^6$ for times (a) $0 \leq t \leq 10$ and (b) $0 \leq t \leq 0.1$.

condition comprised of two non-overlapping regions with either $u > 0$ and $v = 0$ or $u = 0$ and $v > 0$. However, we purposefully locate the interface between the two regions at $x = 0.5$, which is significantly different from the steady-state interface at about 0.6. Hence, we expect this interface to move to the right over time. Specifically, we choose

$$\begin{aligned} u_{ini}(x) &= \begin{cases} 6.4(x-0.5)^2 & \text{if } 0 \leq x < 0.5, \\ 0 & \text{if } 0.5 \leq x \leq 1, \end{cases} \\ v_{ini}(x) &= \begin{cases} 0 & \text{if } 0 \leq x < 0.5, \\ 1.6x(x-0.5) & \text{if } 0.5 \leq x \leq 1, \end{cases} \\ w_{ini}(x) &= 0, \end{aligned}$$

which is plotted in Figure 3 (a). Since the regions of non-zero concentrations for u and v are chosen non-overlapping, we have $q = \lambda uv \equiv 0$ at $t = 0$, which is confirmed in Figure 4 (a).

Figure 3 shows the solutions u, v, and w at six different time steps, while Figure 4 shows the reaction rate $q = \lambda uv$ at the same time steps. After $t = 0$, species A diffuses to the region occupied by B and vice versa. Once the species A and B coexist in the same region, they react rapidly and form C. This behavior can be observed in Figure 3 (b), where we notice an increase in the concentration of C, $w(x)$, around the point 0.5. At the same location, we note the appearance of an internal layer in the reaction rate q, as shown in Figure 4 (b). The *width* of the layer appears already comparable to the one in the steady-state result in Figure 1 (b), but its height still grows over time to reach the steady-state value. (Notice the different scales on the vertical axes in Figures 4 and 1 (b).) As t increases, C continues to be produced in the region where A and B coexist and the internal layer in the reaction rate q moves to the right. Meanwhile, the concentration of C increases in the rest of the domain due to diffusion. We can see in Figure 3 (c) that by $t = 0.052$, we have $w(x) > 0$ everywhere in the domain. By $t = 1.053$, the concentration profiles of A and B are close to those from steady-state and at $t = 10$ the steady-state is reached, as can be observed in Figures 3 (e) and (f). Notice in Figure 4 (e) at time $t = 1.053$ that the location of the interface is clearly to the right of the eventual steady-state value.

To visualize the motion of the interface between regions dominated by species A or B for all times $0 \le t \le 10$, Figure 5 (a) plots the interface over the entire (x,t)-plane. This plot is determined numerically by computing $z(x,t) := u(x,t) - v(x,t)$ and determining the contour level $z = 0$ using MATLAB's `contour` function. With t increasing on the vertical axis, Figure 5 (a) shows the motion of the interface from $x = 0.5$ at $t = 0$ to a maximum value of about $x \approx 0.65$ at $t \approx 0.5$, before converging more slowly to the steady-state value of about 0.6. Figure 5 (b) zooms in on the transient behavior for $0 \le t \le 0.1$.

3.1.2 Type 2 initial condition: four non-overlapping regions

The second type of initial condition is designed so that there will be four non-overlapping regions of species concentrations with either A or B, that is, we prescribe either $u = 0$ or $v = 0$ at every point in the domain at $t = 0$. Due to diffusion, A and B will contact each other, and the reaction rate $q = \lambda uv$ will become non-zero at the three interfaces between the regions. Since the steady-state solution only admits one interface, we expect two of the three interfaces to coalesce within some initial phase. Concretely, we choose the initial condition

$$u_{ini}(x) = \begin{cases} 6.4(0.25 - x) & \text{if } 0 \le x < 0.25, \\ 0 & \text{if } 0.25 \le x < 0.5, \\ 16(0.75 - x)(x - 1) & \text{if } 0.5 \le x < 0.75, \\ 0 & \text{if } 0.75 \le x \le 1, \end{cases}$$

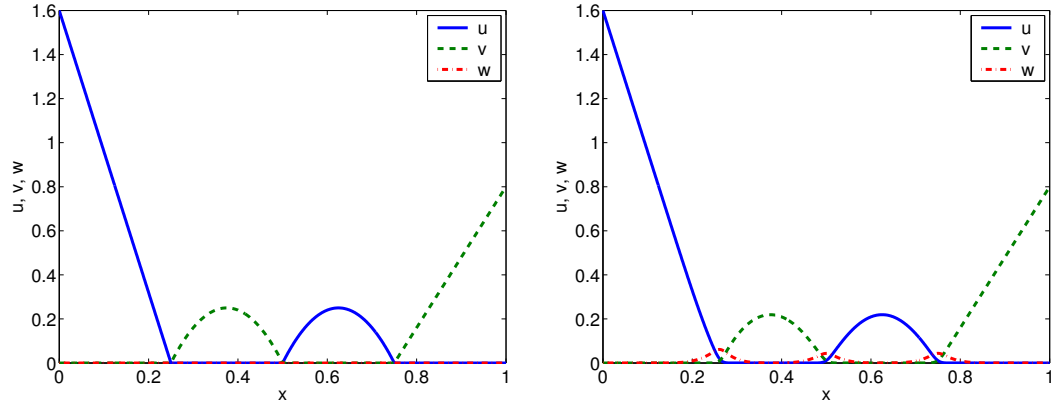
$$v_{ini}(x) = \begin{cases} 0 & \text{if } 0 \le x < 0.25, \\ 16(1 - x)(x - 0.25) & \text{if } 0.25 \le x < 0.5, \\ 0 & \text{if } 0.5 \le x < 0.75, \\ 3.2(x - 0.75) & \text{if } 0.75 \le x \le 1. \end{cases}$$

$$w_{ini}(x) = 0.$$

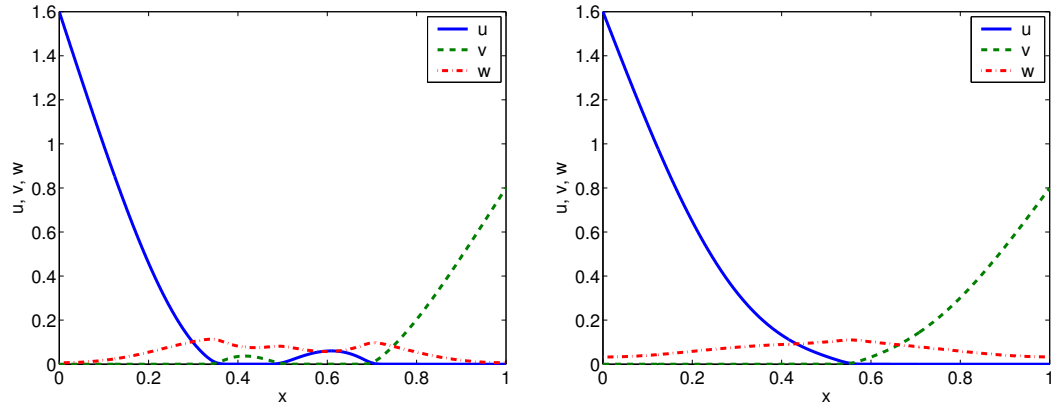
This condition is plotted in Figure 6 (a), and the plot of q in Figure 7 (a) confirms that $q = \lambda uv \equiv 0$ initially.

Figure 6 shows the solutions u, v, and w at six different time steps and Figure 7 plots the fast reaction rate $q = \lambda uv$ at the same time steps. We observe the rapid appearance of three interfaces when the reaction between species A and B starts. These layers appear to have comparable *width* to the one in the steady-state result in Figure 2 (d). (Notice the different scales on the vertical axes in Figures 4 and 1 (d).) The layers move and within quite a short period of time, two of the layers coalesce; indeed, by time $t = 0.021$ there is only one internal layer, as can be seen in Figure 7 (d). By $t = 1.228$ in Figure 7 (e), also its *height* has reached the steady-state value, while it continues to move to the steady-state location at about 0.6, which is reached by $t = 10$ in Figure 7 (f).

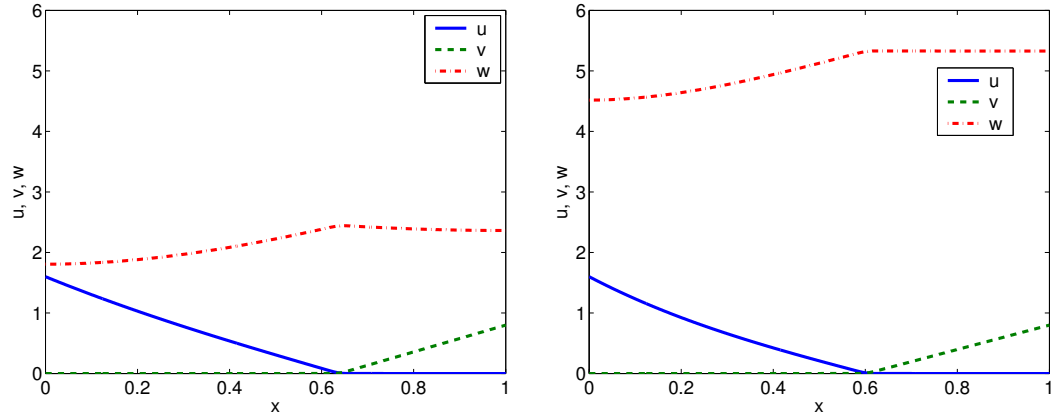
We again analyze the interface motion in detail. Figure 8 (a) plots the interface over the entire (x,t)-plane for all $0 \le t \le 10$. We notice again that the sole remaining interface after the initial transient moves to the right, before approaching the steady-state value from the right. To see the coalescing behavior more clearly, Figure 8 (b) zooms in on the time interval $0 \le t \le 0.1$. We notice that by time $t \approx 0.01$, the first and second interface have coalesced. It is interesting to notice in Figure 8 (a) that the third interface originally moved to the left towards the other interfaces, before moving to the right, as seen in Figure 8 (a).



(a) $t = 0$ (b) $t = 0.0009$

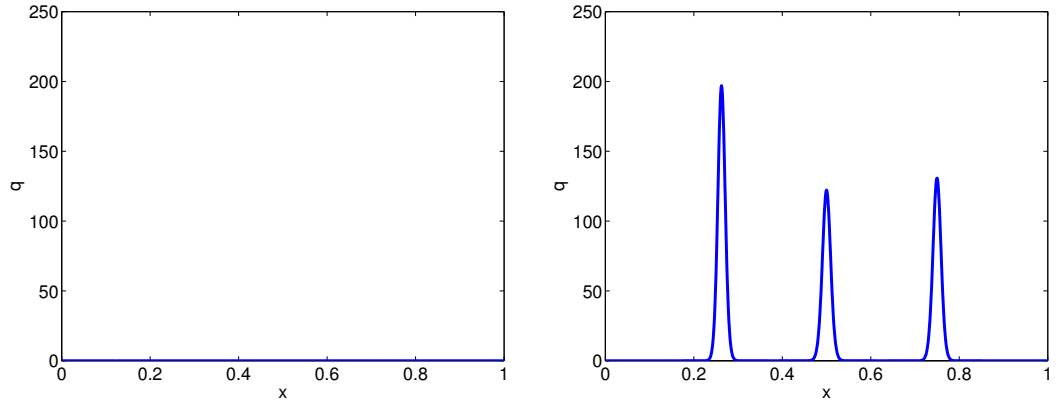


(c) $t = 0.008$ (d) $t = 0.021$



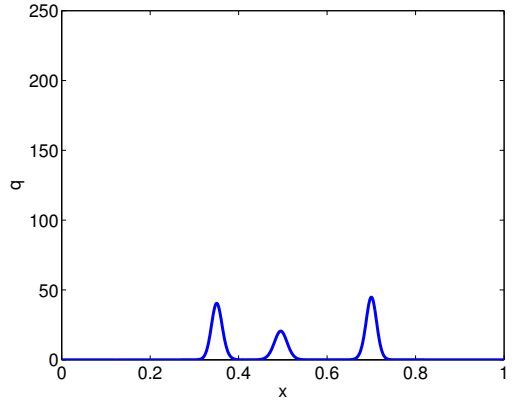
(e) $t = 1.228$ (f) $t = 10$

Figure 6: Plot of the solutions u, v, and w for type 2 initial conditions and $\lambda = 10^6$ at different time steps: (a) $t = 0$, (b) $t = 0.0009$, (c) $t = 0.008$, (d) $t = 0.021$, (e) $t = 1.228$, (f) $t = 10$. Notice the different scales on the vertical axes.

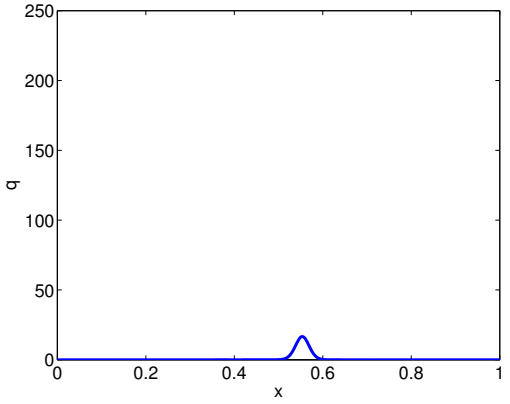


(a) $t = 0$

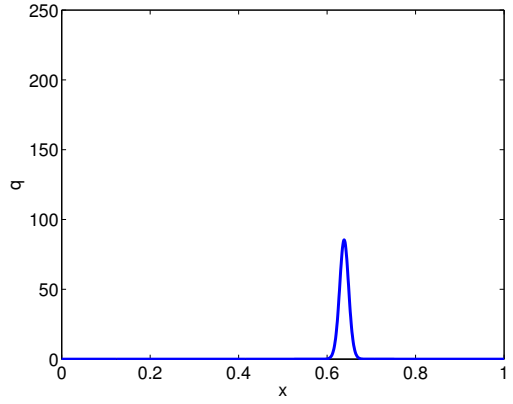
(b) $t = 0.0009$



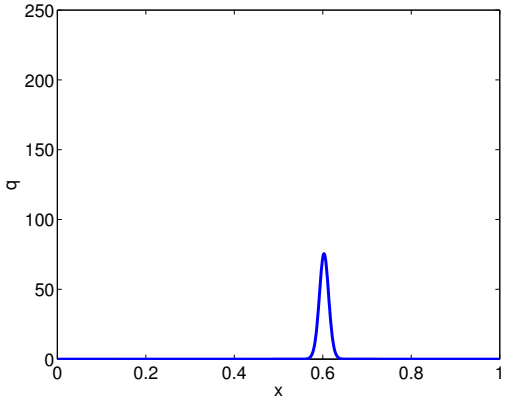
(c) $t = 0.008$



(d) $t = 0.021$

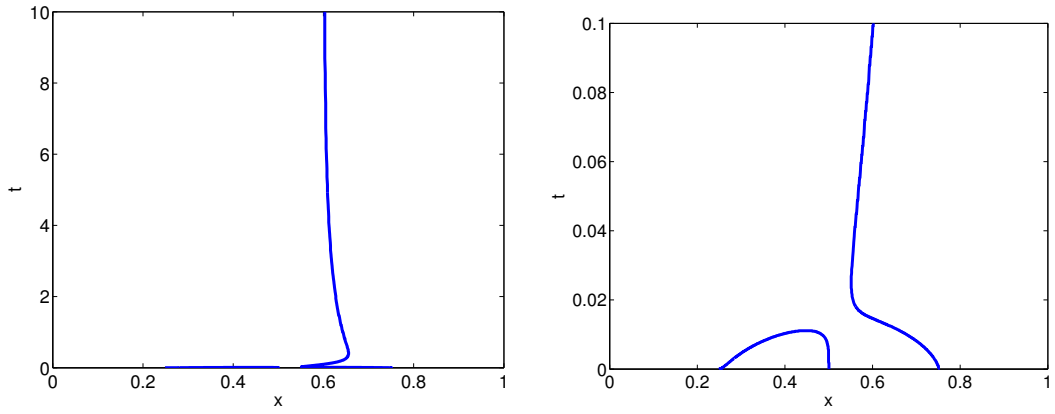


(e) $t = 1.228$



(f) $t = 10$

Figure 7: Plot of the reaction rate $q = \lambda uv$ for type 2 initial condition and $\lambda = 10^6$ at different time steps: (a) $t = 0$, (b) $t = 0.0009$, (c) $t = 0.008$, (d) $t = 0.021$, (e) $t = 1.228$, (f) $t = 10$.



(a) (b)

Figure 8: Motion of internal layers for type 2 initial conditions and $\lambda = 10^6$ for times (a) $0 \leq t \leq 10$ and (b) $0 \leq t \leq 0.1$.

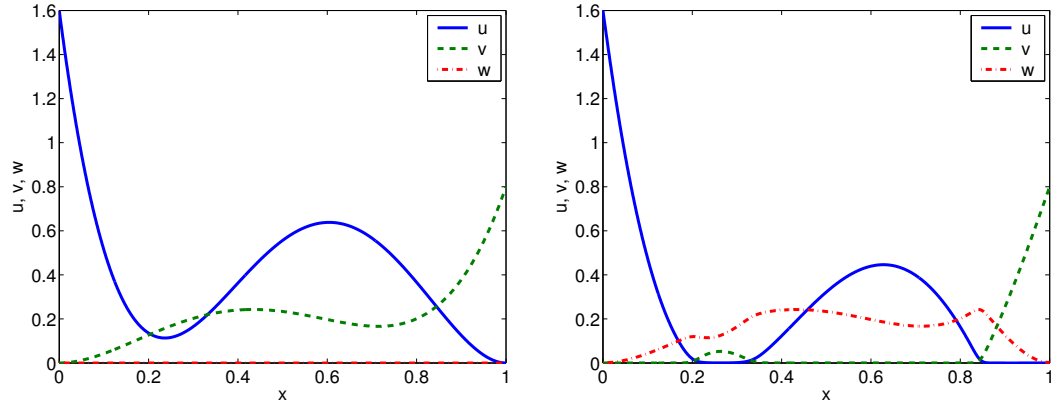
3.1.3 Type 3 initial condition: four overlapping regions

The third type of initial condition implements a more general case in which the species A and B are allowed to coexist everywhere in the domain initially: their regions overlap. Still, we design it so that there are alternately two regions with $u > v$ and two regions with $u < v$; we anticipate that this should result in three interfaces developing. More precisely, we conjecture that there will be two initial transient behaviors: Due to the fast nature of the reaction between A and B, we expect that diffusion and the slower reaction will be comparatively negligible: the intermediate C will be created extremely rapidly initially, until either A or B is depleted at each point in the domain. At this time, there will be three interfaces between the four non-overlapping regions solely occupied by A or B. During a second, slower phase, two of the three interfaces should coalesce and the remaining one tend to the steady-state value, just as in the previous case. Specifically, we choose

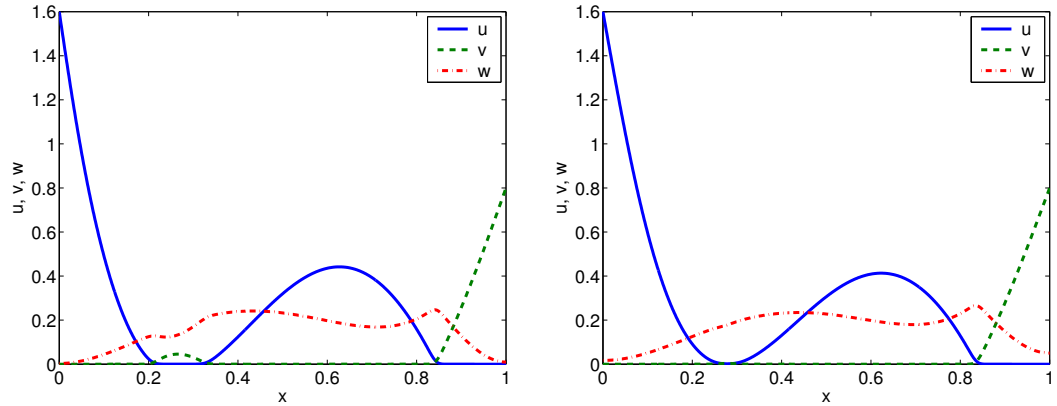
$$\begin{aligned} u_{ini}(x) &= 27.3x^4 - 67x^3 + 53.7x^2 - 15.6x + 1.6, \\ v_{ini}(x) &= 9x^4 - 13.77x^3 + 5.57x^2, \\ w_{ini}(x) &= 0. \end{aligned}$$

This is plotted in Figure 9 (a). Notice that $u > v$ for about $0 \leq x \leq 0.2$ and $0.35 \leq x \leq 0.85$, with $u < v$ in the remaining regions. At the initial time $t = 0$, the reaction rate $q = \lambda uv$ is positive (and large), in contrast to the other two types of initial conditions; this is confirmed by Figure 10 (a).

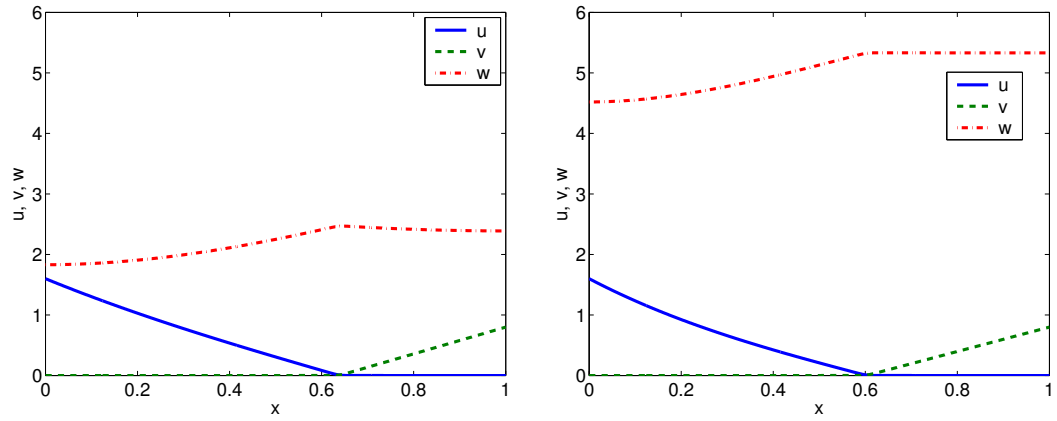
Figure 9 shows the solutions u, v, and w at six different time steps. Figure 10 plots the reaction rate $q = \lambda uv$ at the corresponding time steps. We notice the change in magnitude of the reaction rate in the very short time interval $0 \leq t \leq 0.00007$. Also, once the species A and B start reacting, three internal layers all of roughly the same width occur in the reaction rate q, see Figure 10 (c) at $t = 0.009$. Two of these layers coalesce and the third one moves to the steady-state location by $t = 10$. Again, the layers attain the *width* of the steady-state layer in Figure 2 (f) very quickly, while their *height* continues to adjust over time. (Notice the different scales on the vertical axes in Figures 4 and 1 (f).)



(a) $t = 0$ (b) $t = 0.00007$

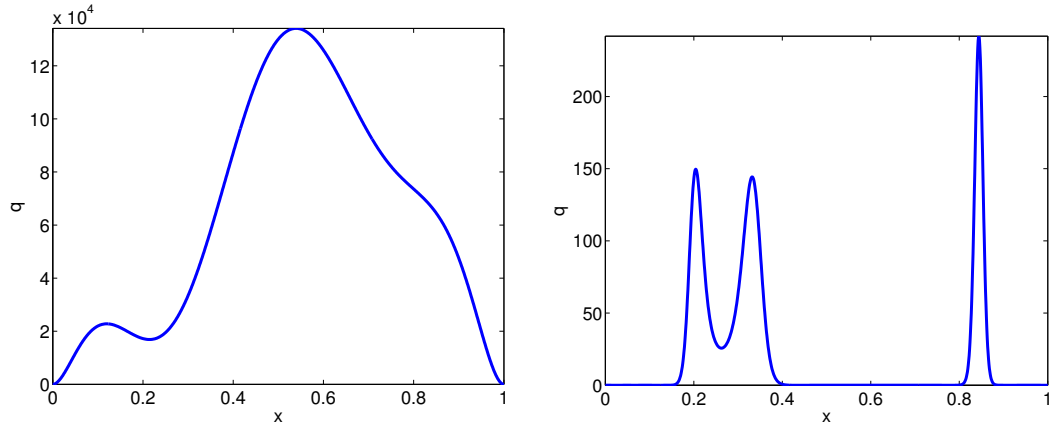


(c) $t = 0.009$ (d) $t = 0.015$

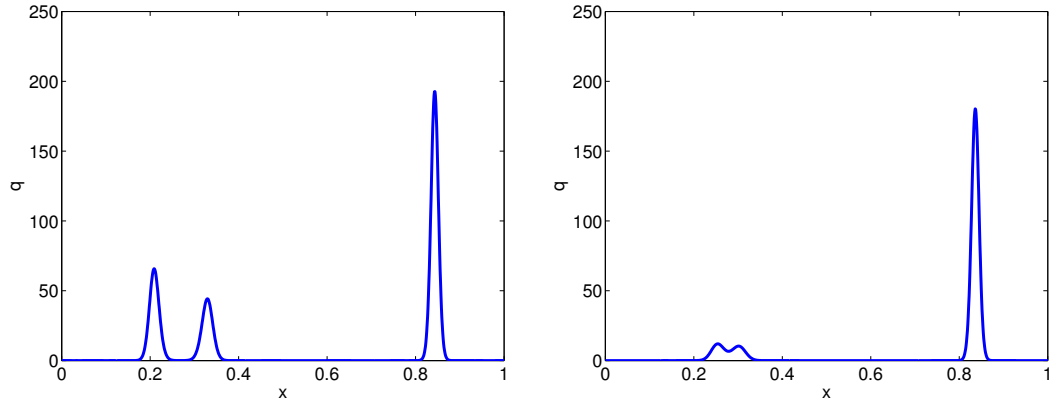


(e) $t = 1.086$ (f) $t = 10$

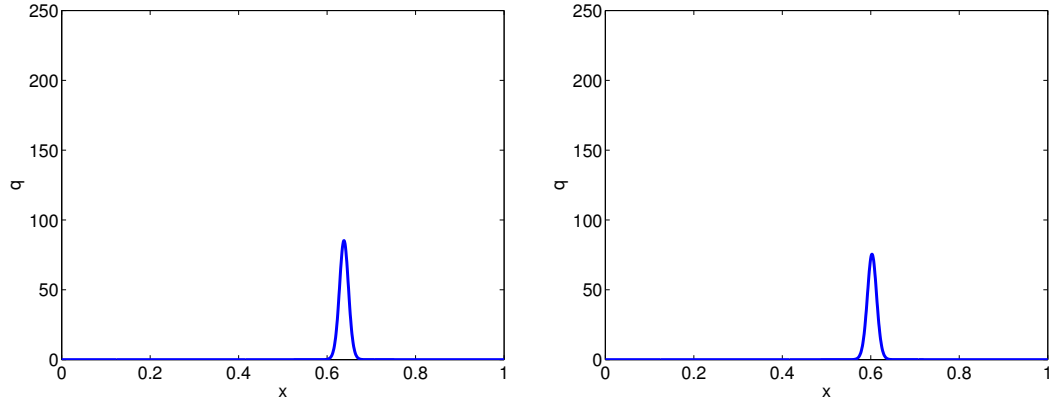
Figure 9: Plot of the solutions u, v, and w for type 3 initial conditions and $\lambda = 10^6$ at different time steps: (a) $t = 0$, (b) $t = 0.00007$, (c) $t = 0.009$, (d) $t = 0.015$, (e) $t = 1.086$, (f) $t = 10$. Notice the different scales on the vertical axes.



(a) $t = 0$ (b) $t = 0.00007$

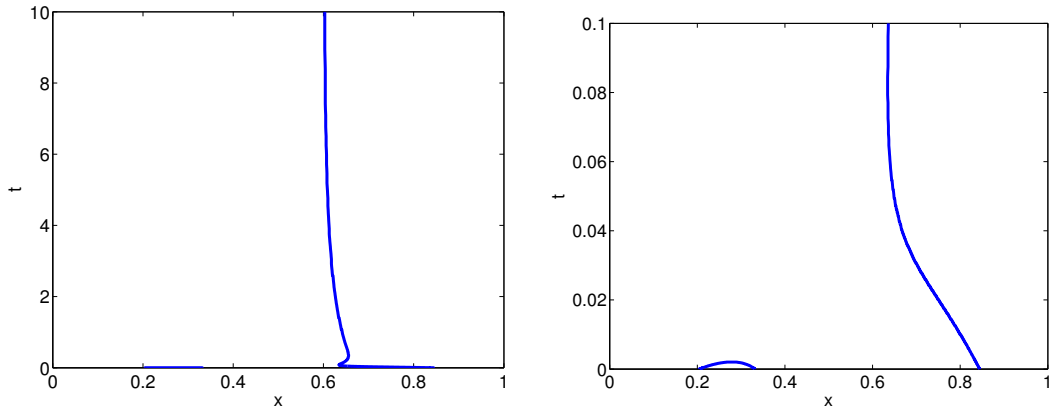


(c) $t = 0.009$ (d) $t = 0.015$



(e) $t = 1.086$ (f) $t = 10$

Figure 10: Plot of the reaction rate $q = \lambda uv$ for type 3 initial condition and $\lambda = 10^6$ at different time steps: (a) $t = 0$, (b) $t = 0.00007$, (c) $t = 0.009$, (d) $t = 0.015$, (e) $t = 1.086$, (f) $t = 10$. Notice the different scales on the vertical axes.



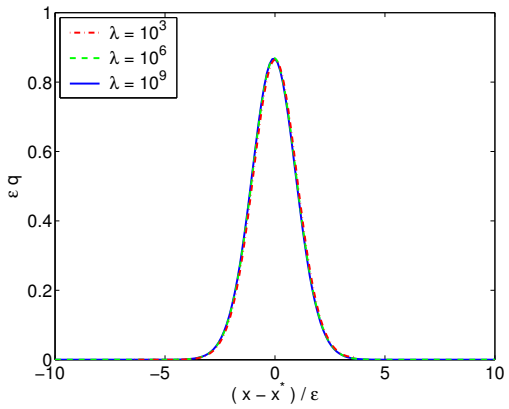
(a) (b)

Figure 11: Motion of internal layers for type 3 initial conditions and $\lambda = 10^6$ for times (a) $0 \le t \le 10$ and (b) $0 \le t \le 0.1$.

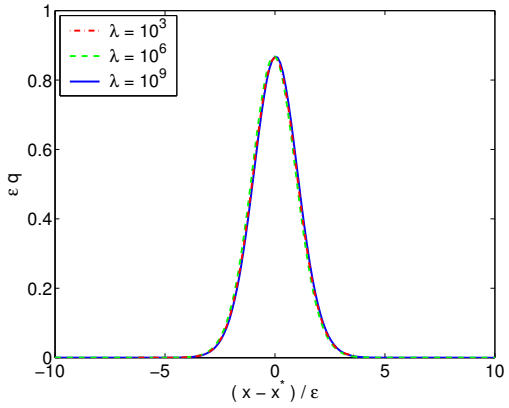
Figure 11 (a) shows the motion of the interfaces in the (x,t)-plane for $0 \le t \le 10$. The overall behavior is similar to, but faster than, the previous case. We see in the zoom in Figure 11 (b) for $0 \le t \le 0.1$ that the first two interfaces coalesce by $t < 0.05$, which is indeed faster than observed in Figure 8 (b).

3.2 Asymptotic results for $\lambda \to \infty$

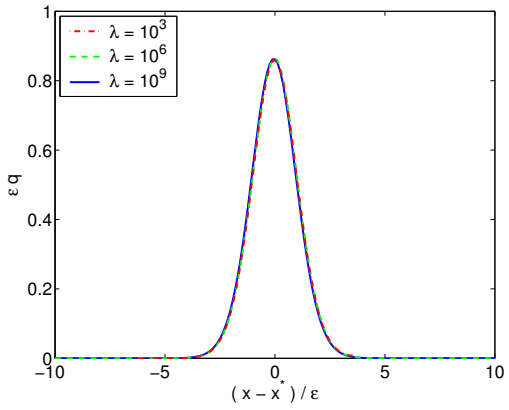
Recall that one major purpose of the numerical simulations was to provide guidance as to the asymptotic behavior of the solutions as $\lambda \to \infty$. To this end, we conduct studies for the three progressively larger values of $\lambda = 10^3, 10^6, 10^9$. Similar qualitative results were obtained as presented for $\lambda = 10^6$ in the previous section; the cases were distinguished by the width and height of $q = \lambda uv$ of each interface: the width decreased and the height increased with growing λ. We conjecture that the scaling is the same as has been proven analytically for the steady-state, namely as $\varepsilon = \lambda^{-1/3}$; no such analytic results are presently available for the time-dependent problem (1)–(3). To check the conjecture, we pick a time after the initial transient phase such that only one internal layer exists and scale the reaction rate q. Figure 12 shows this scaled quantity $\tilde{q} = \varepsilon q$ plotted vs. the scaled and shifted variable $\xi = (x - x^*)/\varepsilon$, where x^* is the numerically determined location of the interface at this time. Figure 12 (a), (b), and (c) show the plots for the type 1, type 2, and type 3 initial conditions, respectively. Figure 12 is at $t = 1$, but similar results were already observed for, e.g., $t = 0.1$. Notice that the convergence with $\lambda \to \infty$ is rather quick, as the scaled rates for all values of λ agree with each other very well. This confirms the expected scaling $\varepsilon = \lambda^{-1/3}$ for the internal layers.



(a)



(b)



(c)

Figure 12: Plot of the scaled fast reaction rate εq vs. the scaled and shifted position $(x-x^*)/\varepsilon$ at time $t = 1$ for $\lambda = 10^3, 10^6, 10^9$, for (a) type 1 initial conditions, (b) type 2 initial conditions, and (c) type 3 initial conditions.

3.3 Numerical convergence and performance studies

3.3.1 Convergence studies

We observe that canonical theory for finite element methods for parabolic reaction-diffusion problems such as (1) applies; see for instance [9, 15]. For linear basis functions this theory predicts second order convergence in the L^2-norm, that is, the error between the true solution u and the numerical approximation in the semidiscretization $u_h(\cdot, t)$ satisfies

$$\|u(\cdot, t) - u_h(\cdot, t)\|_{L^2(\Omega)} \leq C\, h^2, \qquad \text{as } h \to 0,$$

with constant C independent of t for any fixed λ. To check the asymptotic behavior of the error, we compute the observed order of convergence of the finite element method

$$p^{(u)} := \log_2 \left(\frac{\|u_h - u_{h/2}\|_{L^2(\Omega)}}{\|u_{h/2} - u_{h/4}\|_{L^2(\Omega)}} \right), \tag{17}$$

One has analogous definitions for $p^{(v)}$ and $p^{(w)}$ with respect to the approximations to v and w, respectively. This estimated order of convergence is expected to tend toward the value 2 as the spatial mesh becomes finer. Since we actually have two sources of errors, from the spatial discretization and from the time discretization, in order to observe the expected order of convergence, the spatial error should dominate the time error. Therefore, in these convergence studies for the semidiscretization we choose the tolerances in `ode15s` as `RelTol` $= 10^{-12}$, `AbsTol` $= 10^{-14}$. Table 1 shows the computed approximated values for $p^{(u)}$, $p^{(v)}$, and $p^{(w)}$ at $t = 1$ for studies with $\lambda = 10^6$ for the type 1 initial condition; very similar results were observed for the other types of initial conditions. It can be seen that the estimated convergence orders are about 2 for all three species approximations u_h, v_h, w_h for all types of initial conditions. We note that when measured in the L^2-norm, the convergence is already very good for smaller values of N, even though this means that fewer than 8 nodes are placed inside the internal layer of width $\varepsilon = \lambda^{-1/3}$. This reflects the very smooth behavior of the solutions outside of the internal layer combined with the small impact of an error inside a thin layer in the L^2-norm.

Table 1: Convergence order estimates $t = 1$ for type 1 initial condition.

N	$p^{(u)}$	$p^{(v)}$	$p^{(w)}$
129	2.027513	2.028129	2.028091
257	2.006831	2.006986	2.006957
513	2.001704	2.001743	2.001736
1025	2.000425	2.000435	2.000434
2049	2.000106	2.000108	2.000108
4097	2.000048	2.000026	2.000018

3.3.2 Performance studies

To gauge the numerical efficiency of the automatic time step and method order selection implemented in MATLAB's `ode15s`, Figure 13 plots the history of time step Δt vs. time t and the automatically selected method order k vs. time t. Here, we use the problem with type 3 initial condition, $\lambda = 10^6$ and $N = 1025$, using `RelTol` $= 10^{-3}$ and `AbsTol` $= 10^{-6}$. In Figures 13 (a), (c), and (e), we see that the time step is required to be very small initially to handle the initial transient but then grows significantly. This means that initially more time steps are taken and later, when the solution tends to steady-state, it is possible to use fewer, larger time steps. We note that even for the type 1 and type 2 initial conditions, small time steps are necessary initially to resolve the rapid transition from $q \equiv 0$ to $q > 0$. Figures 13 (b), (d), and (f) show the method order k that is chosen by `ode15s`; we see that it is fairly high after the initial start-up period. This allows us to conclude that the ODE problem is fairly smooth and not too stiff for the values of λ considered here. These results explain our insistence on using a sophisticated ODE solver with automatic time step selection, which made simulations for a problem with transient layers much more efficient.

Finally, we return to the comparison of our code using MATLAB to using the software package FEMLAB 2.3 directly. In order to get a fair comparison, we selected linear finite elements, a Jacobian obtained by automatic differentiation, and the same number of nodes in FEMLAB. Moreover, we insisted that FEMLAB 2.3 use its default ODE solver, which is `ode15s`, and chose the same tolerances `RelTol` $= 10^{-3}$ and `AbsTol` $= 10^{-6}$. Both MATLAB and FEMLAB studies are performed under the Linux operating system on a computer with a Pentium 4 CPU running at 3.2 GHz (cache size 1024 kB) with 1 GB of memory.

Table 2 shows the performance results for $\lambda = 10^3$ in the first row and for $\lambda = 10^6$ in the second row. Tables 2 (a) and (c) show that the ODE solver uses the same number of time steps for both cases of λ and for all values of N in each case. The larger value of λ requires about three times as many time steps as the smaller one. While the number of time steps agrees exactly, we have confirmed that not exactly the same solutions are computed by both methods. Based on the fact that the same number of time steps are used, we can fairly compare the observed wall clock times in seconds reported in Tables 2 (b) and (d). Again, the timings for each case of N are three times larger for $\lambda = 10^6$ as for $\lambda = 10^3$. Within each case of λ, the observed times grow linearly with N; this corresponds to the complexity of $\mathcal{O}(N)$ of Gaussian elimination with fixed bandwidth. This is expected for our ordering of variables in the ODE system, as discussed in Section 2; it was not originally expected for FEMLAB's ordering of variables and demonstrates clearly that the professional-grade package performs a re-ordering of the equation that is as good as our custom ordering. Comparing the timings for our code to FEMLAB 2.3, we observe that our code is about a factor 3 faster in every case. This factor suffers slightly as N grows larger.

We also compared the memory used by our code and by FEMLAB 2.3 by monitoring manually the Unix command `top` during the simulations. For the case of $\lambda = 10^6$, we find that our code requires 14 MB for $N = 1025$ and 138 MB for $N = 8193$, while FEMLAB 2.3 needs 25 MB and 230 MB, respectively. For both codes, we notice a memory increas of about a factor 8 from $N = 1025$ to $N = 8193$, as expected. Comparing both codes, we see that our code uses significantly less memory, about 40%, than FEMLAB 2.3. The same behaviors are observed for the case of $\lambda = 10^9$, where we find that our code requires 50 MB for $N = 1025$

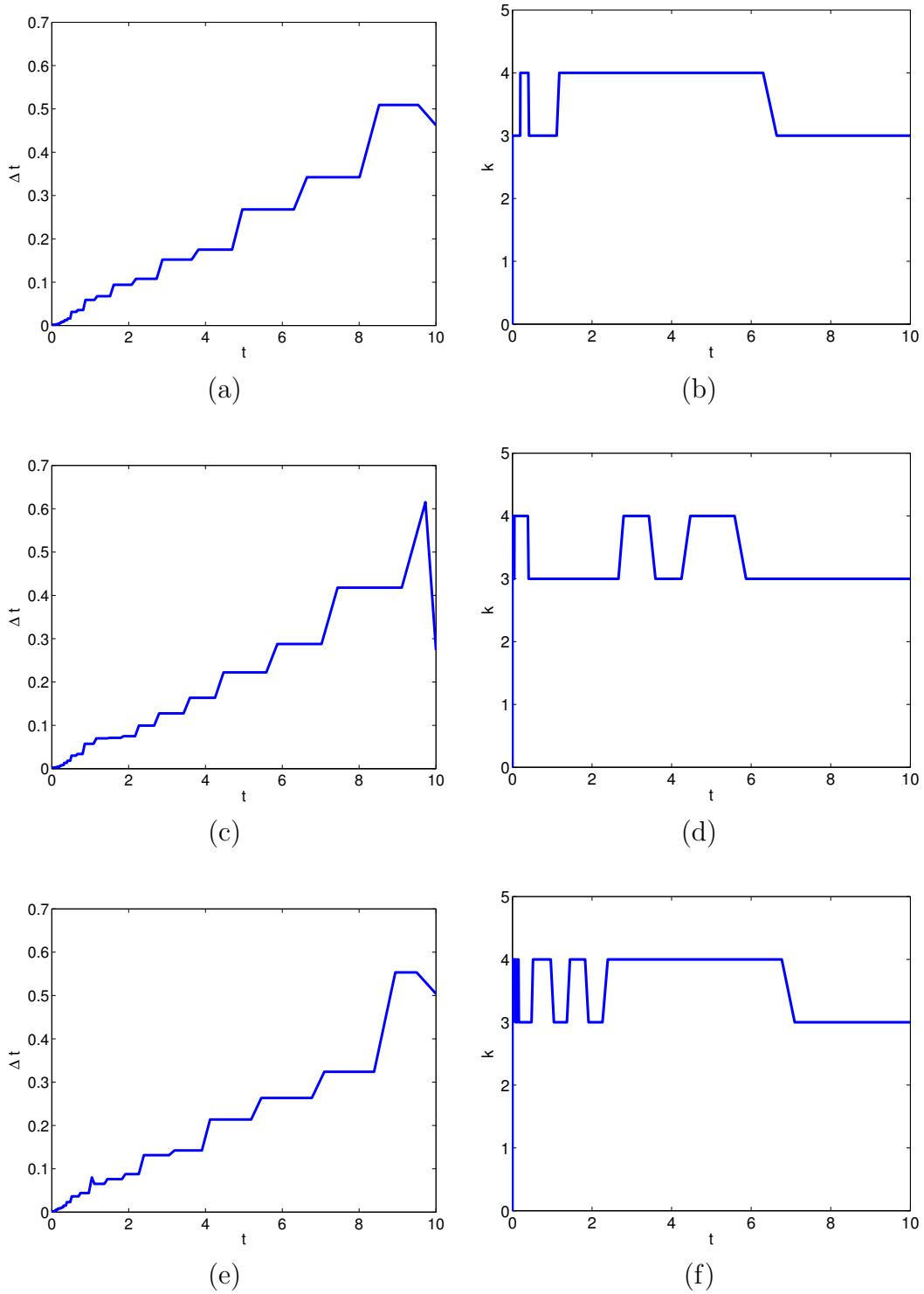


Figure 13: History of (a), (c), (e) the automatically selected time step Δt vs. t and (b), (d), (f) the automatically selected method order k vs. t of the NDFk method for (a), (b) type 1 initial conditions, (c), (d) type 2 initial conditions, and (e), (f) type 3 initial conditions.

and 430 MB for $N = 8193$, while FEMLAB 2.3 needs 93 MB and 770 MB, respectively. We used $N = 1025$ here just to compare the memory usage, even though the approximation may not be reliable, as the layer of width $\varepsilon = \lambda^{1/3}$ is not resolved in this case. In all cases, we point out that we cannot predict the memory usage by the codes, because both of them use MATLAB's `ode15s`, which continues to allocate memory internally to store the solution at all time steps.

Table 2: Performance comparison of our code and FEMLAB 2.3 for simulations for $0 \le t \le 10$. (a) and (b) Simulations for $\lambda = 10^3$. (c) and (d) Simulations for $\lambda = 10^6$.

Number of time steps taken		
N	our code	FEMLAB
1025	144	144
2049	145	145
4097	144	144
8193	144	144

(a)

Observed wall clock time in seconds		
N	our code	FEMLAB
1025	3	24
2049	7	34
4097	17	48
8193	36	94

(b)

Number of time steps taken		
N	our code	FEMLAB
1025	361	361
2049	363	363
4097	363	363
8193	359	359

(c)

Observed wall clock time in seconds		
N	our code	FEMLAB
1025	11	75
2049	26	100
4097	67	167
8193	143	317

(d)

Acknowledgments

This project started as the first author's student consulting project for the third author as client in the graduate mathematics class *Introduction to Interdisciplinary Consulting* taught by the second author in Fall 2003. We gratefully acknowledge support of the Department of Mathematics and Statistics for this exciting instructional initiative. Furthermore, we thank L. V. Kalachev for useful comments and discussions.

References

[1] J. M. Coyle, J. E. Flaherty, and R. Ludwig. On the stability of mesh equidistribution strategies for time-dependent partial differential equations. *J. Comput. Phys.*, 62:26–39, 1986.

[2] FEMLAB Version 2.3. COMSOL AB, `http://www.comsol.com`.

[3] W. Hundsdorfer and J. Verwer. *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*, volume 33 of *Springer Series in Computational Mathematics*. Springer-Verlag, 2003.

[4] L. V. Kalachev. Personal communication.

[5] L. V. Kalachev and T. I. Seidman. Singular perturbation analysis of a stationary diffusion/reaction system whose solution exhibits a corner-type behavior in the interior of the domain. *J. Math. Anal. Appl.*, 288:722–743, 2003.

[6] N. Kopteva and M. Stynes. A robust adaptive method for a quasi-linear one-dimensional convection-diffusion problem. *SIAM J. Numer. Anal.*, 39(4):1446–1467, 2001.

[7] N. Madden and M. Stynes. A uniformly convergent numerical method for a coupled system of two singularly perturbed linear reaction-diffusion problems. *IMA J. Numer. Anal.*, 23(4):627–644, 2003.

[8] MATLAB Release 13 (Version 6.5). The MathWorks, `http://www.mathworks.com`.

[9] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*, volume 23 of *Springer Series in Computational Mathematics*. Springer-Verlag, 1994.

[10] H.-G. Roos, M. Stynes, and L. Tobiska. *Numerical Methods for Singularly Perturbed Differential Equations: Convection-Diffusion and Flow Problems*, volume 24 of *Springer Series in Computational Mathematics*. Springer-Verlag, 1996.

[11] A. Sandu. Positive numerical integration methods for chemical kinetic systems. *J. Comput. Phys.*, 170(2):589–602, 2001.

[12] T. I. Seidman and L. V. Kalachev. A one-dimensional reaction/diffusion system with a fast reaction. *J. Math. Anal. Appl.*, 209:392–414, 1997.

[13] L. F. Shampine and M. W. Reichelt. The MATLAB ODE suite. *SIAM J. Sci. Comput.*, 18(1):1–22, 1997.

[14] M. Stynes. An adaptive uniformly convergent numerical method for a semilinear singular perturbation problem. *SIAM J. Numer. Anal.*, 26(2):442–455, 1989.

[15] V. Thomée. *Galerkin Finite Element Methods for Parabolic Problems*, volume 25 of *Springer Series in Computational Mathematics*. Springer-Verlag, 1997.

[16] A. B. Vasil'eva, V. F. Butuzov, and L. V. Kalachev. *The Boundary Function Method for Singular Perturbation Problems*. SIAM, 1995.