UMEÅ UNIVERSITY
Department of Physics
Master's Thesis

June 7, 2024

Nodal Discontinuous Galerkin Spectral Element Method for Advection-Diffusion Equations in Chromatography

Per Sehlstedt
sehlstedtper@gmail.com

Nodal Discontinuous Galerkin Spectral Element Method for Advection-Diffusion Equations in Chromatography

Per Sehlstedt, sehlstedtper@gmail.com

Supervisors:	Olivier Cloarec	Sartorius Stedim Data Analytics AB
	David Andersson	Sartorius Stedim Data Analytics AB
	Philip Semrén	Department of Physics, Umeå University
Examiner:	Karl Larsson	Deptartment of Mathematics and Mathematical Statistics, Umeå University

Master of Science Thesis in Engineering Physics, 30 ECTS

Department of Physics

Umeå University

SE-901 87 Umeå, Sweden

Abstract

In this thesis, we mainly investigate the application of a nodal discontinuous Galerkin spectral element method (DGSEM) for simulating processes in column liquid chromatography. Additionally, we investigate the effectiveness of a total variation diminishing in the mean (TVDM) limiter in controlling spurious oscillations related to the Gibbs phenomenon. With an order-of-accuracy test, we demonstrated that our nodal DGSEM achieved and, in multiple instances, even exceeded theoretical convergence rates, especially with an increased number of elements, validating the use of high-order basis functions for achieving high-order accuracy. We also demonstrated how setup parameters could affect process outcomes, which suggests that numerical simulations can help guide the development of experimental methods since they can explore the solution space of an optimization problem much faster than experimental procedures by leveraging computational speed. Finally, we showed that the TVDM limiter successfully eliminated severe oscillations and negative concentrations near shock regions but introduced significant smearing of the shocks. These findings validate the nodal DGSEM as a highly accurate and reliable tool for detailed modeling of column liquid chromatography, which is essential for improving efficiency, yield, and product quality in biopharmaceutical manufacturing.

Acknowledgements

First, I want to thank my supervisors, Olivier and David, at Sartorius, for allowing me to work on this problem, helping me learn about the applications of column liquid chromatography, and all the support they gave me during the project. Furthermore, I want to thank my academic supervisor, Philip, for providing great feedback and different viewpoints while writing this thesis. Finally, I want to thank my examiner, Karl, for providing help regarding the methods used when needed.

Contents

1 Introduction

Chromatography is an important biophysical technique for qualitative and quantitative analysis [1]. It is a complex process involving various effects, such as fluid dynamics, mass transfer, and equilibrium thermodynamics. However, their relative importance depends on the experimental conditions [2].

There exist many types of chromatography [1]. Column liquid chromatography is a widely employed separation technique in the biopharmaceutical industry's downstream processing [3]. It is crucial in identifying, extracting, and purifying chemical components within a mixture [1]. The technique works by injecting a liquid sample into a mobile phase that passes through a stationary phase, where distinct components interact at different rates, leading to their separation [1, 4].

Since experimental procedures can be costly and time-consuming, computer simulations are crucial in enhancing the efficiency and reliability of liquid chromatography processes. They enable the prediction and optimization of chromatographic outcomes by simulating the complex interactions between the sample components, the mobile phase, and the stationary phase. Even if capturing all the complexities accurately is difficult, the predictions can help develop new procedures, leading to increased productivity and better product quality through improved separation [5].

There are many different models that, to various degrees, describe the movement and interactions of the particles as they move through the column. One of the most prominent and comprehensive is the general rate model (GRM) [2, 5, 6]. However, the practical application of the GRM can often be hindered by computational complexity and unknown parameter values [2, 7]. Therefore, a simpler family of models generally called the lumped kinetic models (LKM), which can be derived from the GRM, are also among the most widely used [2, 6].

Although analytical solutions exist to the most basic configurations of these models (see, e.g., [2, 8]), most applications require numerical methods since no closed-form solutions are known [5]. A natural choice for a numerical method is the discontinuous Galerkin spectral element method (DGSEM), allowing complex geometries, arbitrary high-order (spectral) convergence within elements, mass conservation, and mesh refinement with adaptive element sizes and polynomial orders (hp-adaptivity) [3, 9]. It was first introduced in 1973 by Reed and Hill in [10] in the context of neutron transport [9, 11–13], and can be considered a high-order generalization of the more common finite volume method [14].

This thesis aims to implement a partial differential equation (PDE) solver in Python based on a spatial high-order nodal discontinuous Galerkin spectral element method that can simulate processes relevant to hardware produced by Sartorius in the context of biopharmaceutical separation, particularly column liquid chromatography.

2 Theory

2.1 Column liquid chromatography

In a typical liquid chromatography setup, a liquid mobile phase passes through a bed of porous particles, carrying a mixture of components that interact differently with the stationary phase. These components diffuse in and out of the particles, undergo molecular interactions, or form transient bonds with the stationary phase, and are eventually swept out of the column [2]. In Figure 1 is a simple schematic of a setup.

2.2 The lumped kinetic models

We will use a lumped kinetic model to model the movement and interactions of the particles as they move through the column, which combines a mass balance equation with a kinetic equation. We will summarize the different aspects here, but see, e.g., Chapter 2 in [2] for a more thorough analysis and discussion.

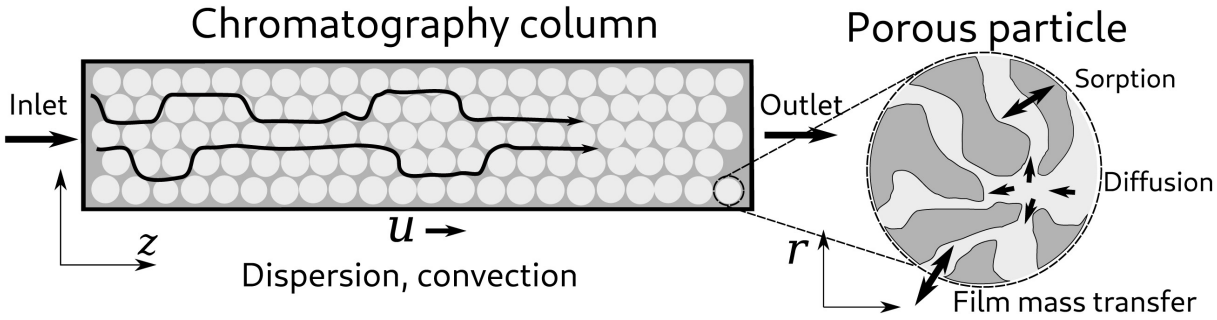

Figure 1: Simple schematic of a liquid chromatography column [15].

A lumped kinetic model assumes the column to be unidimensional, so the mass balance equation only has two independent variables: the time t and the distance z along the column. The compressibility of the mobile phase is neglected since it is almost always negligible in preparative chromatography. Additionally, the partial molar volumes of the sample components are assumed to be the same in both phases since, in liquid chromatography, the differences between these volumes do not exceed a few percent. It is also assumed that there are no thermal effects, and the influence of the heat of adsorption is neglected. Finally, it is assumed that the column is operated under constant conditions, e.g., constant temperature, pressure, and mobile phase flow rate, so that all the physicochemical parameters remain constant. Under these assumptions, the mass balance equation has the form

$$\frac{\partial C_i}{\partial t} + F\frac{\partial C_{s,i}}{\partial t} + v\frac{\partial C_i}{\partial z} = D_L\frac{\partial^2 C_i}{\partial z^2} \quad \text{in } \Omega \times \mathrm{T}, \tag{1}$$

where $C_i = C_{m,i}(t, z)$ and $C_{s,i} = C_{s,i}(t, z)$ are component i's concentrations in the mobile and stationary phases, respectively, $F = (1-\varepsilon)/\varepsilon$ is the phase ratio, with ε as total porosity of the column packing, v is the mobile phase velocity, D_L is the axial dispersion in the mobile phase (due to molecular and eddy diffusion), and $\Omega = [0, L]$ and $\mathrm{T} = (0, \tau]$ are the spatial and temporal domains, respectively, with L being the length of the column and τ being the final time.

Since eq. (1) involves C_i and $C_{s,i}$, a second equation is required for its solution. A practical choice is to have an equation that describes the relationship between the concentrations in the stationary and mobile phases, sometimes called a binding model. In a lumped kinetic model, mass transfer kinetics are slow; thus, a kinetic equation with a relationship like

$$\frac{\partial C_{s,i}}{\partial t} = g_i(C_1, C_2, \ldots, C_i, \ldots, C_n, C_{s,1}, C_{s,2}, \ldots, C_{s,i}, \ldots, C_{s,n}) \quad \text{in } \Omega \times \mathrm{T}, \tag{2}$$

is used, where g_i is some arbitrary function. However, if the mass transfer kinetics occur such that equilibrium is reached quickly, then an equilibrium isotherm with a relationship like

$$C_{s,i} = f_i(C_1, C_2, \ldots, C_i, \ldots, C_n) \quad \text{in } \Omega \times \mathrm{T}, \tag{3}$$

should be used, with f_i being some arbitrary function, which then would lead to what is known as an equilibrium dispersive model.

Although the derivation of the mass balance equation, eq. (1), relies on a series of assumptions, they do not significantly limit the range of validity of the conclusions drawn from considering the system. The choice of the kinetic equation plays a far more significant role in that sense [2].

2.3 The Langmuir kinetic model

The Langmuir kinetic model is one of the simplest models in nonlinear chromatography. It is based on the principles of equilibrium between the mobile and stationary phases. The model assumes that the stationary phase has a finite number of specific binding sites, with each site capable of holding only one

solute molecule. The adsorption rate is proportional to the solute concentration in the mobile phase. In contrast, the desorption rate is proportional to the solute concentration in the stationary phase. It is particularly useful for predicting and optimizing chromatographic separations, as it accurately reflects the nonlinear adsorption behavior often observed in practice [2].

The competitive version, where all components compete for the same binding sites, is written as

$$\frac{\partial C_{s,i}}{\partial t} = k_{a,i} q_{s,i} \left(1 - \sum_{j=1}^{n} \frac{C_{s,j}}{q_{s,j}} \right) C_i - k_{d,i} C_{s,i} \quad \text{in } \Omega \times \mathrm{T}, \tag{4}$$

where $q_{s,i}$ is the column saturation capacity, and $k_{a,i}$ and $k_{d,i}$ are the rate constants of adsorption and desorption, respectively, of component i [2, 6]. The competitiveness emerges from the summation in the first term, but it can easily be adjusted to encompass other relations (see, e.g., [6]).

2.4 The Langmuir MPM model

By introducing a modulator component (sometimes referred to as a strong solvent or additive), it is possible to modify retention times. One possibility is for the modulator component to change the solubility of the components in the mobile phase. For example, proteins' solubility in water depends on the solution's ionic strength. Therefore, the retention time can be modified by changing the salt concentration of the aqueous solution ("salting out" of proteins) [2]. Hence, a commonly used modulator component is salt.

The Langmuir mobile phase modulator (MPM) model slightly modifies the Langmuir kinetic model. It can be obtained by changing the adsorption and desorption coefficients in eq. (4) to

$$k_{a,i} \leftarrow k_{a,i} \mathrm{e}^{\gamma_i C_0} \quad \text{and} \tag{5a}$$

$$k_{d,i} \leftarrow k_{d,i} {C_0}^{\beta_i}, \tag{5b}$$

where C_0 is the salt concentration in the mobile phase, γ_i describes the hydrophobicity, and β_i the ion-exchange characteristics. Finally, salt is considered intert and therefore $\partial C_{s,0}/\partial t$ is zero [16, 17].

2.5 Boundary conditions

To complement chromatography problems, it is common to use the Danckwerts boundary condition,

$$v C_{\text{inj},i} = v C_i - D_L \frac{\partial C_i}{\partial z} \quad \text{on } \{0\} \times \mathrm{T}, \tag{6a}$$

$$\frac{\partial C_i}{\partial x} = 0 \quad \text{on } \{L\} \times \mathrm{T}, \tag{6b}$$

where $C_{\text{inj},i} = C_{\text{inj},i}(t)$ is the concentration injection profile of component i. The essence is that the mass flux caused by the injection at the column inlet is the same as that achieved in a pipe with the same diameter as the column [2].

2.6 Gibbs phenomenon and limiters

While linear high-order (≥ 2) numerical methods bring many advantages when solving PDEs, such as high-order accuracy, the cost comes when discontinuities and sharp shocks develop in the solution. These events can cause what is known as the Gibbs phenomenon, which results in several effects: a reduction to first-order pointwise accuracy away from the point of discontinuity, a loss of pointwise convergence at the point of discontinuity, and the introduction of artificial and persistent oscillations around the point of discontinuity. Fortunately, this phenomenon is well understood, and a multitude of methods have been developed to address it. Each method offers a unique approach, showing different strengths and weaknesses [9].

Applying simple filtering techniques may seem like a straightforward approach to reduce oscillations. However, these techniques often lead to severe smearing of shocks and violation of conservation, making them inadequate in many cases [9]. Instead, many methods will use a two-step procedure, where the first step is to identify elements that might need a limiting procedure to reduce oscillations. The second step is then to replace the solution polynomials in the troubled elements with reconstructed polynomials, often based on the original element average, to not violate conservation (see, e.g., [9, 12, 13, 18]).

A simple limiting procedure is to use a total variation diminishing in the mean (TVDM) limiter, which replaces the original solution in the troubled element with a first-order reconstruction polynomial. While effective in reducing oscillations, it tends to over-smooth and degrades the high-order accuracy in the element [9].

While it is usually challenging to design limiters that achieve a non-oscillatory property and high-order accuracy, essentially non-oscillatory (ENO) and weighted ENO (WENO) schemes are common approaches, which are well established in finite volumes and finite difference methods [12, 18]. However, many such schemes can be rather complicated, adding a lot of overhead, and will oftentimes require information of neighbors' neighbors to maintain high order, which can destroy the locality and applicability to complex geometries that discontinuous Galerkin (DG) methods often are known for [13].

A newly developed approach, termed oscillation-eliminating DG (OEDG) method involves an alternate progression between the conventional DG scheme and a damping equation. It retains many essential attributes of the conventional DG approach, such as conservation, locality, and high-order convergence rates, while eliminating undesirable oscillations [19].

3 Method

The method developed in this section is mainly based on the book *Nodal Discontinuous Galerkin Methods. Algorithms, Analysis, and Applications* by Jan S. Hesthaven and Tim Warburton, the references therein, and the MATLAB scripts that accompany the text (see [9]).

3.1 Weak forms

Since no closed-form solutions are known to most configurations of our model [5], we will seek approximate solutions that are easy to represent in a computer and which we find by discretizing time and space separately. However, to be able to discretize the problem, we first derive weak forms of the model equations.

We begin with the mass balance equation, eq. (1), and since it will be the same for all components, we will temporarily drop the component index for readability. To deal with a second-order equation, we first reformulate it as a first-order system to retain stability in our upcoming scheme [20]. Since we are dealing with linear convection, and we assumed v and D_L to be constant, we use the simple reformulation

$$\frac{\partial C}{\partial t} = -F\frac{\partial C_s}{\partial t} - vp + D_L\frac{\partial p}{\partial z} \quad \text{in } \Omega \times \mathrm{T}, \tag{7a}$$

$$p = \frac{\partial C}{\partial z} \quad \text{in } \Omega \times \mathrm{T}, \tag{7b}$$

however, there are, of course, other ways it could be done (see, e.g., [3, 5, 8, 14] for a more commonly used reformulation that is a bit more involved since it is originally based on more general convection-diffusion equations, possibly containing nonlinear convection terms and variable coefficients).

We then let $\mathrm{V} = H^1(\Omega)$ be a Hilbert space and $(\cdot, \cdot)$ denote the $L^2(\Omega)$ inner product on Ω, and

formulate our weak form: find $\{C, C_s, p\}$ such that, for every fixed $t \in \mathrm{T}$, $\{C, C_s, p\} \subset \mathrm{V}$ and

$$\left(\frac{\partial C}{\partial t}, \phi\right) = \left(-F\frac{\partial C_s}{\partial t} - vp + D_L\frac{\partial p}{\partial z}, \phi\right), \tag{8a}$$

$$(p, \phi) = \left(\frac{\partial C}{\partial z}, \phi\right), \tag{8b}$$

for any arbitrarily smooth test function ϕ. Important to note is that a solution to eq. (1) is a solution to eq. (8), but the inverse is not necessarily true. However, the equations are equivalent if the solutions are sufficiently smooth.

Similarly, for the kinetic equation, taking eq. (4) as an example, we formulate the weak form: find $\{C_i, C_{s,i}\}$ such that, for every fixed $t \in \mathrm{T}$, $\{C_i, C_{s,i}\} \subset \mathrm{V}$ and

$$\left(\frac{\partial C_{s,i}}{\partial t}, \phi\right) = \left(k_{a,i}q_{s,i}\left[1 - \sum_{j=1}^{n}\frac{C_{s,j}}{q_{s,j}}\right]C_i - k_{d,i}C_{s,i}, \phi\right), \tag{9}$$

for any arbitrarily smooth test function ϕ.

3.2 Spatial discretization

We discretize the spatial domain Ω into K nonoverlapping elements D^k with boundaries $\partial\mathrm{D}^k$, such that

$$\Omega = \bigcup_{k=1}^{K} \mathrm{D}^k. \tag{10}$$

Since it is common for spatially one-dimensional chromatography models, like the one we are using, to develop gradient fronts with varying degrees of steepness that move through the entire domain [5], we use an equidistant grid, $\mathrm{D}^k = [z^k, z^{k+1}]$ with

$$z^1 = 0, \quad z^{K+1} = L, \quad z^{k+1} - z^k = \Delta z > 0 \quad \forall k \in \{n \in \mathbb{N} : 1 \leq n \leq K\},$$

considering that underresolution of sharp gradients can cause spurious oscillations when using DG methods.

Following the discretization of the spatial domain, we introduce the local space V_h^k, which we define to be the space of all polynomial functions of degree N or less defined on D^k. We also introduce the global space V_h, which we define as

$$\mathrm{V}_h = \bigoplus_{k=1}^{K} \mathrm{V}_h^k, \tag{11}$$

thus a space containing N-th-order piecewise polynomials. Additionally, we equip V_h^k and V_h with the local and global inner products

$$(u, v)_{\mathrm{D}^k} = \int_{\mathrm{D}^k} uv \, \mathrm{d}z \tag{12}$$

and

$$(u, v)_{\Omega} = \sum_{k=1}^{K} (u, v)_{\mathrm{D}^k}, \tag{13}$$

respectively.

Starting with eq. (8), we then assume that the solution $\{C, C_s, p\} \subset \mathrm{V}$ is well approximated by $\{C_h, C_{s,h}, p_h\} \subset \mathrm{V}_h$, which we write as the direct sum of K local polynomial solutions $\left\{C_h^k, C_{s,h}^k, p_h^k\right\} \subset \mathrm{V}_h^k$,

$$\begin{bmatrix} C(z,t) \\ C_s(z,t) \\ p(z,t) \end{bmatrix} \simeq \begin{bmatrix} C_h(z,t) \\ C_{s,h}(z,t) \\ p_h(z,t) \end{bmatrix} = \bigoplus_{k=1}^{K} \begin{bmatrix} C_h^k(z,t) \\ C_{s,h}^k(z,t) \\ p_h^k(z,t) \end{bmatrix}. \tag{14}$$

Subsequently, we can replace the large space V with the much smaller subspace $\mathrm{V}_h \subset \mathrm{V}$ and seek the approximate solutions $\{C_h, C_{s,h}, p_h\}$ instead. Furthermore, we choose our test function to be in the same space as our approximate solution, $\phi = \phi_h \in \mathrm{V}_h$, following a Galerkin method, and require it to be orthogonal to the residual of the approximation to obtain

$$\left(\frac{\partial C_h}{\partial t} + F\frac{\partial C_{s,h}}{\partial t} + vp_h - D_L\frac{\partial p_h}{\partial z}, \phi_h\right)_\Omega = 0, \tag{15a}$$

$$\left(p_h - \frac{\partial C_h}{\partial z}, \phi_h\right)_\Omega = 0. \tag{15b}$$

By introducing $N_p = N + 1$ distinct local grid points, $\left\{z_i^k\right\}_{i=1}^{N_p} \subset \mathrm{D}^k$, we can express V_h^k using a Lagrange basis as

$$\mathrm{V}_h^k = \text{span}\left\{\ell_i^k\right\}_{i=1}^{N_p}, \tag{16}$$

where ℓ_i^k is a Lagrange interpolating polynomial on D^k,

$$z \in \mathrm{D}^k : \ell_i^k(z) = \prod_{\substack{j=1 \\ j\neq i}}^{N_p} \frac{z - z_j^k}{z_i^k - z_j^k}, \tag{17}$$

with the property $\ell_i^k(z_j^k) = \delta_{ij}$.

We can now express ϕ_h as a direct sum of K linear combinations of Lagrange interpolating polynomials and thus transform eq. (15) into a set of local statements using eqs. (10) to (14) and (16), with the result being

$$\left(\frac{\partial C_h^k}{\partial t} + F\frac{\partial C_{s,h}^k}{\partial t} + vp_h^k - D_L\frac{\partial p_h^k}{\partial z}, \ell_j^k\right)_{\mathrm{D}^k} = 0, \tag{18a}$$

$$\left(p_h^k - \frac{\partial C_h^k}{\partial z}, \ell_j^k\right)_{\mathrm{D}^k} = 0, \tag{18b}$$

for all $j \in \{n \in \mathbb{N} : 1 \leq n \leq N_p\}$ on all K elements. However, due to the lack of constraints imposed on the smoothness of the basis functions between elements, we do not have a way to reconstruct meaningful global solutions from the local solutions to eq. (18). Thus, we use integration by parts to obtain

$$\left(\frac{\partial C_h^k}{\partial t}, \ell_j^k\right)_{\mathrm{D}^k} = -F\left(\frac{\partial C_{s,h}^k}{\partial t}, \ell_j^k\right)_{\mathrm{D}^k} - v\left(p_h^k, \ell_j^k\right)_{\mathrm{D}^k} - D_L\left[\left(p, \frac{\partial \ell_j^k}{\partial z}\right)_{\mathrm{D}^k} + \left(\hat{n}\cdot p_h^k, \ell_j^k\right)_{\partial \mathrm{D}^k}\right], \tag{19a}$$

$$\left(p_h^k, \ell_j^k\right)_{\mathrm{D}^k} = -\left(C_h^k, \frac{\partial \ell_j^k}{\partial z}\right)_{\mathrm{D}^k} + \left(\hat{n}\cdot C_h^k, \ell_j^k\right)_{\partial \mathrm{D}^k}, \tag{19b}$$

where $\hat{n}$ is the outward pointing unit normal, which in this one-dimensional case will be a scalar, valued 1 or -1. We can reason that the flux terms in eq. (19) are the pieces that will connect the elements and help us construct meaningful global solutions since both $\left\{C_h^k, C_{s,h}^k, p_h^k\right\}$ and $\left\{C_h^{k+1}, C_{s,h}^{k+1}, p_h^{k+1}\right\}$ depend on the flux evaluation at the point z^{k+1}. Therefore, we introduce the numerical fluxes $\{C^*, p^*\}$ as the unique values to be used at the interface, which we obtain by combining information from both elements.

With the numerical fluxes, we obtain our weak formulation to the mass balance equation using the discontinuous Galerkin spectral element method: find $\left\{C_h^k, C_{s,h}^k, p_h^k\right\}$ such that, for every fixed $t \in \mathrm{T}$,

$\left\{C_h^k, C_{s,h}^k, p_h^k\right\} \subset \mathrm{V}_h^k$ and

$$\left(\frac{\partial C_h^k}{\partial t}, \ell_j^k\right)_{\mathrm{D}^k} = -F\left(\frac{\partial C_{s,h}^k}{\partial t}, \ell_j^k\right)_{\mathrm{D}^k} - v\left(p_h^k, \ell_j^k\right)_{\mathrm{D}^k} - D_L\left[\left(p_h^k, \frac{\partial \ell_j^k}{\partial z}\right)_{\mathrm{D}^k} + \left(\hat{n}\cdot p^*, \ell_j^k\right)_{\partial \mathrm{D}^k}\right], \tag{20a}$$

$$\left(p_h^k, \ell_j^k\right)_{\mathrm{D}^k} = -\left(C_h^k, \frac{\partial \ell_j^k}{\partial z}\right)_{\mathrm{D}^k} + \left(\hat{n}\cdot C^*, \ell_j^k\right)_{\partial \mathrm{D}^k}, \tag{20b}$$

holds $\forall j \in \{n \in \mathbb{N} : 1 \leq n \leq N_p\}$ and $\forall k \in \{n \in \mathbb{N} : 1 \leq n \leq K\}$. It is a spectral element method because it uses high-degree piecewise polynomials as basis functions. Alternatively, we can use integration by parts again to obtain the strong formulation, where no derivatives are related to the test function, and replace eq. (20) with

$$\begin{aligned}\left(\frac{\partial C_h^k}{\partial t}, \ell_j^k\right)_{\mathrm{D}^k} = &-F\left(\frac{\partial C_{s,h}^k}{\partial t}, \ell_j^k\right)_{\mathrm{D}^k} - v\left(p_h^k, \ell_j^k\right)_{\mathrm{D}^k} + D_L\left(\frac{\partial p_h^k}{\partial z}, \ell_j^k\right)_{\mathrm{D}^k} \\ &- D_L\left(\hat{n}\cdot(p_h^k - p^*), \ell_j^k\right)_{\partial \mathrm{D}^k}\end{aligned} \tag{21a}$$

$$\left(p_h^k, \ell_j^k\right)_{\mathrm{D}^k} = \left(\frac{\partial C_h^k}{\partial z}, \ell_j^k\right)_{\mathrm{D}^k} - \left(\hat{n}\cdot(C_h^k - C^*), \ell_j^k\right)_{\partial \mathrm{D}^k}. \tag{21b}$$

Finally, we express the local solutions using nodal expansions,

$$z \in \mathrm{D}^k : \begin{bmatrix} C_h^k(z,t) \\ C_{s,h}^k(z,t) \\ p_h^k(z,t) \end{bmatrix} = \sum_{i=1}^{N_p} \begin{bmatrix} C_h^k(z_i^k,t) \\ C_{s,h}^k(z_i^k,t) \\ p_h^k(z_i^k,t) \end{bmatrix} \ell_i^k(z), \tag{22}$$

which for the strong formulation results in the linear systems

$$\mathcal{M}^k \frac{\partial \boldsymbol{C}_h^k}{\partial t} = -\mathcal{M}^k F \frac{\partial \boldsymbol{C}_{\boldsymbol{s},h}^k}{\partial t} - \mathcal{M}^k u \boldsymbol{p}_h^k + \mathcal{S}^k D_L \boldsymbol{p}_h^k - D_L\left[(p_h^k - p^*)\boldsymbol{\ell}^{\boldsymbol{k}}\right]_{z^k}^{z^{k+1}}, \tag{23a}$$

$$\mathcal{M}^k \boldsymbol{p}_h^k = \mathcal{S}^k \boldsymbol{C}_h^k - \left[(C_h^k - C^*)\boldsymbol{\ell}^{\boldsymbol{k}}\right]_{z^k}^{z^{k+1}}, \tag{23b}$$

where we have defined $\mathcal{M}^k, \mathcal{S}^k \in \mathbb{R}^{N_p \times N_p}$ as

$$\mathcal{M}_{ij}^k = \left(\ell_i^k, \ell_j^k\right)_{\mathrm{D}^k}, \quad \mathcal{S}_{ij}^k = \left(\ell_i^k, \frac{\partial \ell_j^k}{\partial z}\right)_{\mathrm{D}^k}, \tag{24}$$

and $\boldsymbol{C}_h^k, \boldsymbol{C}_{\boldsymbol{s},h}^k, \boldsymbol{p}_h^k, \boldsymbol{\ell}^k \in \mathbb{R}^{N_p}$ as

$$\boldsymbol{C}_{h,i}^k = C_h^k(z_i^k,t), \quad \boldsymbol{C}_{\boldsymbol{s},h,i}^k = C_{s,h}^k(z_i^k,t), \quad \boldsymbol{p}_{h,i}^k = p_h^k(z_i^k,t). \quad \boldsymbol{\ell}_i^k = \ell_i^k(z).$$

Following the same type of reasoning for eq. (9), we can easily see that the only special treatment needed is for the nonlinear terms,

$$f(z,t) \propto C_{s,j}(z,t) C_i(z,t),$$

since there are no spatial derivatives. To deal with these, we use the approximation

$$z \in \mathrm{D}^k : f(z,t) \simeq \sum_{i=1}^{N_p} f(z_i^k,t)\ell_i^k(z),$$

which can introduce aliasing errors and, in turn, cause instability if f is not sufficiently smooth. However, this approach has significant computational advantages due to its simplicity. So from eq. (9), we obtain the linear system

$$\mathcal{M}^k \frac{\partial \boldsymbol{C}_{\boldsymbol{s},\boldsymbol{i},h}^k}{\partial t} = \mathcal{M}^k k_{a,i} q_{s,i}\left(1 - \sum_{j=1}^{n} \frac{\boldsymbol{C}_{\boldsymbol{s},\boldsymbol{j},h}^k}{q_{s,j}}\right) \boldsymbol{C}_{\boldsymbol{i},h}^k - \mathcal{M}^k k_{d,i} \boldsymbol{C}_{\boldsymbol{s},\boldsymbol{i},h}^k, \tag{25}$$

where we have had to reintroduce the component indexes.

3.3 Elementwise operations

Storing all K sets of $\left\{\mathcal{M}^k, \mathcal{S}^k\right\}$ can be memory intensive since they are unique to each element. Therefore, we map each element to a reference element I, where we then can use a single set $\{\mathcal{M}, \mathcal{S}\}$ for all elements. Another benefit of the mapping is that it can help with numerical conditioning.

In our one-dimensional case with an equidistant grid, we use the affine mapping

$$z \in \mathrm{D}^k : z(r) = z^k + \frac{1+r}{2}\Delta z, \tag{26}$$

with the reference variable $r \in \mathrm{I} = [-1, 1]$. Furthermore, we introduce the inner product on I as

$$(u, v)_{\mathrm{I}} = \int_{\mathrm{I}} uv \, \mathrm{d}r.$$

Applying the mapping to our local solutions, we obtain

$$z \in \mathrm{D}^k : u_h(r,t) = \sum_{i=1}^{N_p} u_h(z(r_i),t)\ell_i(r) \quad \forall u_h \in \left\{C_h^k, C_{s,h}^k, p_h^k\right\},$$

where ℓ_i is a Lagrange interpolating polynomial on I, and $\{r_i\}_{i=1}^{N_p}$ are the points in I that will give us back our physical grid points $\left\{z_i^k\right\}_{i=1}^{N_p}$ in D^k. Similarly, applying the mapping to $\left\{\mathcal{M}^k, \mathcal{S}^k\right\}$ from eq. (24), we obtain

$$\mathcal{M}_{ij}^k = \left(\ell_i^k, \ell_j^k\right)_{\mathrm{D}^k} = \frac{\Delta z}{2}\left(\ell_i, \ell_j\right)_{\mathrm{I}} = \frac{\Delta z}{2}\mathcal{M}_{ij} \tag{27}$$

and

$$\mathcal{S}_{ij}^k = \left(\ell_i^k, \frac{\partial \ell_j^k}{\partial z}\right)_{\mathrm{D}^k} = \left(\ell_i, \frac{\partial \ell_j}{\partial r}\right)_{\mathrm{I}} = \mathcal{S}_{ij}. \tag{28}$$

So far, we have only considered nodal expansions of our local solutions; however, mathematically equivalent but computationally different representations that can help determine $\mathcal{M}$ and $\mathcal{S}$ are modal expansions of the form

$$u_h(r,t) = \sum_{n=1}^{N_p} \hat{u}_n(t)\widetilde{P}_{n-1}(r),$$

where $\hat{u}_n(t)$ are modal coefficients and $\widetilde{P}_n(r)$ are normalized Legendre polynomials with the subscript denoting the order, such that

$$\left(\widetilde{P}_m, \widetilde{P}_n\right)_{\mathrm{I}} = \delta_{mn}. \tag{29}$$

By choosing the modal coefficients such that the expansion is interpolatory at the local grid points the same way our nodal expansion is,

$$u_h(r_i,t) = \sum_{n=1}^{N_p} \hat{u}_n(t)\widetilde{P}_{n-1}(r_i),$$

we can directly transform between the nodal and modal representations via the relation

$$\mathcal{V}\hat{\boldsymbol{u}} = \boldsymbol{u_h},$$

where $\mathcal{V} \in \mathbb{R}^{N_p \times N_p}$ is recognized as a generalized Vandermonde matrix and

$$\mathcal{V}_{ij} = \widetilde{P}_{j-1}(r_i), \quad \hat{\boldsymbol{u}}_i = \hat{u}_i(t), \quad \boldsymbol{u_h}_{,i} = u_h(z(r_i),t).$$

Subsequently, due to the uniqueness of polynomial interpolation, we have

$$\ell_i(r) = \sum_{n=1}^{N_p} \left(\mathcal{V}^T\right)_{in}^{-1} \widetilde{P}_{n-1}(r),$$

which can be used to show that

$$\mathcal{M} = \left(\mathcal{V}\mathcal{V}^T\right)^{-1}, \quad \mathcal{S} = \mathcal{M}\mathcal{V}_r\mathcal{V}^{-1}, \quad \mathcal{V}_{r,(i,j)} = \left.\frac{\mathrm{d}\widetilde{P}_j}{\mathrm{d}r}\right|_{r_i},$$

using the orthonormality of $\widetilde{P}_n$, eq. (29).

The conditioning of $\mathcal{V}$ is determined solely by the local grid points $\{r_i\}_{i=1}^{N_p}$, so to ensure $\mathcal{V}$ is non-singular, we choose them to be Legendre-Gauss-Lobatto (LGL) points. The great thing with LGL points is that they maintain desirable properties regarding numerical quadrature and polynomial interpolation while including the endpoints of the element. There are sets with more optimal behavior, but the inclusion of the endpoints greatly simplifies the evaluation of the flux terms. Looking at eq. (23a) for example, if we use LGL points, we can rewrite the flux term as

$$z \in \mathrm{D}^k : \left[(p_h^k - p^*)\boldsymbol{\ell}^{\boldsymbol{k}}\right]_{z^k}^{z^{k+1}} = \mathcal{E}\begin{bmatrix} \hat{n}\cdot(p_h^k - p^*)\big|_{z(r_1)} \\ \hat{n}\cdot(p_h^k - p^*)\big|_{z(r_{N_p})} \end{bmatrix}, \tag{30}$$

where we define $\mathcal{E} \in \mathbb{R}^{N_p \times 2}$ as $\mathcal{E}_{i1} = \delta_{i1}$ and $\mathcal{E}_{i2} = \delta_{iN_p}$.

Since we ensured $\mathcal{V}$ to be well conditioned and invertible, $\mathcal{M}$ will be as well. Therefore, it will be beneficial to introduce two new operators, which we define as

$$\mathcal{D}_r = \mathcal{M}^{-1}\mathcal{S} \quad \text{and} \quad \mathcal{L} = \mathcal{M}^{-1}\mathcal{E}, \tag{31}$$

where $\mathcal{D}_r$ is generally referred to as a differentiation matrix, transforming point values to derivatives at these same points, and $\mathcal{L}$ is the lift matrix, lifting facial features to become volume features.

Putting it all together, eqs. (23), (25) to (28), (30) and (31), we obtain

$$\frac{\partial \boldsymbol{C}_{\boldsymbol{i},h}^k}{\partial t} = -F\frac{\partial \boldsymbol{C}_{\boldsymbol{s},\boldsymbol{i},h}^k}{\partial t} - v\boldsymbol{p}_{\boldsymbol{i},h}^k + D_L\frac{2}{\Delta z}\mathcal{D}_r\boldsymbol{p}_{\boldsymbol{i},h}^k - D_L\mathcal{L}\left(\frac{2}{\Delta z}\begin{bmatrix} \hat{n}\cdot(p_{i,h}^k - p_i^*)\big|_{z_1^k} \\ \hat{n}\cdot(p_{i,h}^k - p_i^*)\big|_{z_{N_p}^k} \end{bmatrix}\right), \tag{32a}$$

$$\boldsymbol{p}_{\boldsymbol{i},h}^k = \frac{2}{\Delta z}\mathcal{D}_r\boldsymbol{C}_{\boldsymbol{i},h}^k - \mathcal{L}\left(\frac{2}{\Delta z}\begin{bmatrix} \hat{n}\cdot(C_{i,h}^k - C_i^*)\big|_{z_1^k} \\ \hat{n}\cdot(C_{i,h}^k - C_i^*)\big|_{z_{N_p}^k} \end{bmatrix}\right), \tag{32b}$$

$$\frac{\partial \boldsymbol{C}_{\boldsymbol{s},\boldsymbol{i},h}^k}{\partial t} = k_{a,i}q_{s,i}\left(1 - \sum_{j=1}^{n}\frac{\boldsymbol{C}_{\boldsymbol{s},\boldsymbol{j},h}^k}{q_{s,j}}\right)\boldsymbol{C}_{\boldsymbol{i},h}^k - k_{d,i}\boldsymbol{C}_{\boldsymbol{s},\boldsymbol{i},h}^k, \tag{32c}$$

which is our finalized scheme for spatial discretization, where we again have reintroduced the component indexes.

3.4 Numerical fluxes

The choice of the numerical flux is a central part of the discontinuous Galerkin scheme and is where we can introduce knowledge of the dynamics of the problem. We will refer to the interior information at the element boundary by a superscript "$-$" and to the exterior information by a superscript "$+$". Using this notation, we define the average as

$$\{\!\{u\}\!\} = \frac{u^- + u^+}{2},$$

and the jumps along a normal, $\hat{n}$, as

$$[\![u]\!] = \hat{n}^- u^- + \hat{n}^+ u^+ = \hat{n}^-(u^- - u^+).$$

In our scheme, eq. (32), we first restrict the numerical flux as

$$C^* = C^*(C_h^+, C_h^-),$$
$$p^* = p^*(C_h^+, C_h^-, p_h^+, p_h^-),$$

such that the auxiliary function p_h^k becomes truly local and is resolved in each element. Otherwise, the two first-order equations, eqs. (32a) and (32b), would be tightly coupled through the numerical flux and, hence, must be solved simultaneously as a globally coupled system.

For the specific choice of the flux, we see that we have one term that relates to linear convection, which has a preferred direction of propagation, and another that relates to diffusion, which does not. We therefore choose

$$C^* = \{\!\{C_h\}\!\} + 0.5[\![C_h]\!],$$
$$p^* = \{\!\{p_h\}\!\} - 0.5[\![C_h]\!],$$

which resemble upwind fluxes but in opposite directions. Actually, due to v being assumed constant, we obtain precisely an upwind flux for the linear convective term. Similarly, since D_L is constant, the combination of C^* and p^* effectively results in upwinding in opposite directions, which captures the dynamic of spreading not having a preferred direction for the diffusive term. This choice for the fluxes is inspired by internal penalty fluxes and methods that are often known as local discontinuous Galerkin (LDG) methods (see, e.g., [9, 21]), where they effectively utilize upwinding, even if it is counter-intuitive to do so.

3.5 Boundary conditions

Following our reformulation of the mass balance, eq. (1) to eq. (7), we can similarly rewrite the boundary conditions, eq. (6), as

$$p = \frac{v}{D_L}\left[C - C_{\text{inj}}\right] \quad \text{on } \{0\} \times \mathrm{T}, \tag{33a}$$
$$p = 0 \quad \text{on } \{L\} \times \mathrm{T}. \tag{33b}$$

We will weakly impose the boundary conditions through the numerical fluxes by defining exterior ghost states (C_h^+, p_h^+) at the domain boundary. Since eq. (33) reflect two Neumann conditions, we define the ghost states to be

$$C_h^+ = C_h^-,\ p_h^+ = -p_h^- + 2f(t) \ \Rightarrow \ \begin{cases} \{\!\{C_h\}\!\} = C_h^-, & [\![C_h]\!] = 0, \\ \{\!\{p_h\}\!\} = f(t), & [\![p_h]\!] = 2\hat{n}^-(p_h^- - f(t)), \end{cases}$$

where $f(t)$ is the right hand side of either eq. (33a) or eq. (33b) depending on the boundary.

3.6 Temporal discretization

Due to our choice of having a kinetic equation as complement to the mass balance equation and our restriction on the numerical fluxes, resolving eq. (32b) locally, the spatial discretization leads to an explicit system of first-order ordinary differential equations (ODEs), eqs. (32a) and (32c), to be integrated in time. Hence, we can solve the system as an initial value problem.

If we would have used an equilibrium isotherm or unrestricted numerical fluxes, we could have ended up with a differential-algebraic system of equations (DAE) instead, which is generally more complex to solve compared to ODEs.

Since we implement our solver in Python, we use `solve_ivp` from `SciPy` [22], which has a wide selection of numerical methods with adaptive time stepping implemented. We will mostly use the BDF-method, which is an implicit multi-step variable-order (1 to 5) method based on a backward differentiation formula for the derivative approximation.

For the initial values, we set all concentrations to be zero inside the column,

$$C_h = C_{s,h} = p_h = 0 \quad \text{in } \Omega \times \{0\}.$$

3.7 TVDM limiter

The limiting procedure to enforce the TVDM property for our implementation is applied after each timestep to every element, and will be based on the use of the minmod function m, defined as

$$m(a_1, \ldots, a_n) = \begin{cases} s \min\limits_{1 \leq i \leq n} |a_i|, & |s| = 1, \\ 0, & \text{otherwise}, \end{cases} \qquad s = \frac{1}{n} \sum_{i=1}^{n} \operatorname{sign}(a_i).$$

The first step is to compute the limited edge values, y_l^k and y_r^k, using

$$y_l^k = \bar{u}_h^k - m\left(\bar{u}_h^k - y_l^k, \bar{u}_h^k - \bar{u}_h^{k-1}, \bar{u}_h^{k+1} - \bar{u}_h^k\right),$$
$$y_r^k = \bar{u}_h^k - m\left(y_r^k - \bar{u}_h^k, \bar{u}_h^k - \bar{u}_h^{k-1}, \bar{u}_h^{k+1} - \bar{u}_h^k\right),$$

where $\bar{u}_h^k$ is the element average. The second step is to compare the limited edge values to the current edge values. If $\left|u_{N_p}^k - y_r^k\right| < \epsilon$ and $\left|u_1^k - y_l^k\right| < \epsilon$, for some arbitrarily small ϵ, there is no need for limiting and the local solution is not altered. However, if there is a need for limiting, the local solution is replaced with

$$\Pi^1 u_h^k(z) = \bar{u}_h^k + \left(z - z_1^k\right) m\left(\frac{\partial u_h^k}{\partial z}, \frac{\bar{u}_h^{k+1} - \bar{u}_h^k}{\Delta z/2}, \frac{\bar{u}_h^k - \bar{u}_h^{k-1}}{\Delta z/2}\right),$$

which will retain the same element average to preserve mass conservation.

3.8 Software verification

Chromatography Analysis and Design Toolkit (CADET) is an open-source modeling and simulation framework for column liquid chromatography written in C++ [4] and will be used to validate some of the results of this work. More specifically, we will use DG CADET presented in [5].

We will perform order-of-accuracy tests to see if we asymptotically approach the theoretical rate of $N + 1$ for DG schemes. We compute the experimental order of convergence (EOC) as

$$EOC = \frac{\log\left(\frac{e_k}{e_{k-1}}\right)}{\log\left(\frac{n_{k-1}}{n_k}\right)}, \tag{34}$$

with $e_k \in \mathbb{R}^+$ being some error norm of the discretization error and $n_k \in \mathbb{N}$ being the number of degrees of freedom (DoFs). The error norms we will consider are the standard L^1 and L^2 norms, and we will measure them on the outlet profiles of the simulations.

With `solve_ivp` being able to perform adaptive time stepping, we set the relative and absolute time integration tolerances as 10^{-10} and 10^{-12}, respectively, since we are mostly interested in the spatial discretization error and want to ensure that that error dominates for the considered ranges.

4 Result

4.1 Order-of-accuracy test

In table 1, we present the results of an order-of-accuracy test conducted on a four-component LKM simulation with Langmuir MPM binding. The chosen parameters are listed in table 2. The injection profiles were given by

$$C_{\text{inj},0}(t) = \begin{cases} 0.0, & t \leq 1.0, \\ t - 1.0, & t > 1.0, \end{cases}$$
$$C_{\text{inj},i}(t) = \begin{cases} 1.0, & t < 0.5, \\ 0.0, & t \geq 0.5, \end{cases} \quad i \in \{1, 2, 3\},$$

and in Figure 2 is the outlet profile obtained from a high-resolution reference simulation, which was obtained using DG CADET with a 10th-order polynomial and 512 elements.

When using a small number of elements, the EOC is lower than the theoretical values for all tested polynomial orders in both the $L^1(\{L\} \times \mathrm{T})$ and $L^2(\{L\} \times \mathrm{T})$ norms, with values falling between 1 and 3. However, as the number of elements increases, the EOC rises and, most times, even exceeds the theoretical values, showing super-convergence. For polynomial orders 4 and 5, we observe values rise to between 7 and 10 for the middle range of elements before dropping again for the largest number of elements. For polynomial order 3, we observe a more steady EOC with most values around 6.5.

Table 1: Order-of-accuracy test results from a four-component LKM with Langmuir MPM binding. The chosen parameters are listed in table 2. The outlet profile of the reference solution can be seen in Figure 2.

N	K	L^1 error	L^1 EOC	L^2 error	L^2 EOC
3	5	3.407×10^{-2}	-	8.068×10^{-2}	-
3	15	5.135×10^{-3}	1.72	1.769×10^{-2}	1.38
3	30	5.357×10^{-4}	3.26	2.267×10^{-3}	2.96
3	60	1.113×10^{-5}	5.59	5.751×10^{-5}	5.30
3	90	7.882×10^{-7}	6.53	4.158×10^{-6}	6.48
3	120	1.187×10^{-7}	6.58	6.230×10^{-7}	6.60
3	150	2.923×10^{-8}	6.28	1.472×10^{-7}	6.46
4	5	2.256×10^{-2}	-	5.705×10^{-2}	-
4	15	1.993×10^{-3}	2.21	7.068×10^{-3}	1.90
4	30	6.130×10^{-5}	5.02	2.785×10^{-4}	4.67
4	60	2.455×10^{-7}	7.96	1.215×10^{-6}	7.84
4	90	1.012×10^{-8}	7.87	3.647×10^{-8}	8.65
4	120	3.475×10^{-9}	3.71	9.202×10^{-9}	4.79
4	150	2.081×10^{-9}	2.30	5.121×10^{-9}	2.63
5	5	1.583×10^{-2}	-	4.132×10^{-2}	-
5	15	6.779×10^{-4}	2.87	2.463×10^{-3}	2.57
5	30	4.719×10^{-6}	7.17	2.277×10^{-5}	6.76
5	60	9.303×10^{-9}	8.99	2.545×10^{-8}	9.81
5	90	2.649×10^{-9}	3.10	6.021×10^{-9}	3.56
5	120	1.544×10^{-9}	1.88	3.544×10^{-9}	1.84

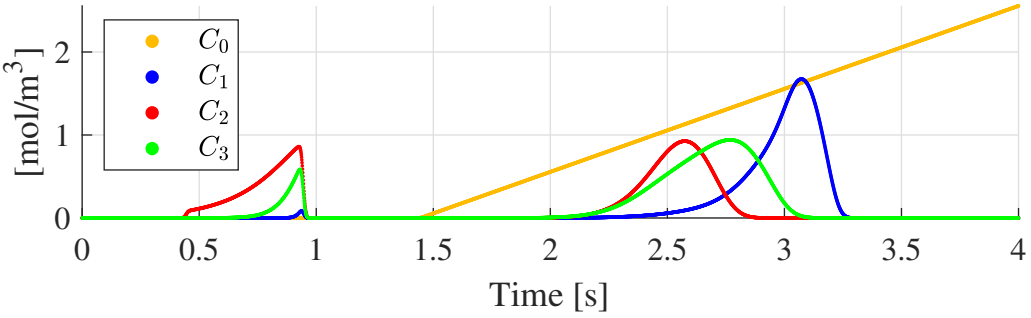

Figure 2: Outlet profile of high-resolution reference simulation computed using DG CADET with a 10th-order polynomial and 512 elements for order-of-accuracy test.

Table 2: LKM and Langmuir MPM parameters for order-of-accuracy test.

Parameters	Symbol	Unit	Value
Components	n	-	4
Total porosity	ε	-	0.6
Column length	L	m	2.0
Mobile phase velocity	v	$\mathrm{m\,s^{-1}}$	4.5
Axial dispersion	D_L	$\mathrm{m^2\,s^{-1}}$	1.4×10^{-3}
Final time	τ	s	4.0
Column saturation capacity	$q_{s,i}$	$\mathrm{mol\,m^{-3}}$	[-, 4.70, 5.29, 3.70]
Adsorption rate	$k_{a,i}$	$\mathrm{m^3\,mol^{-1}\,s^{-1}}$	[-, 35.5, 1.59, 8.70]
Desorption rate	$k_{d,i}$	$\mathrm{m^{3\beta_i}mol^{-\beta_i}s^{-1}}$	[-, 1.18, 3.06, 2.00]
Hydrophobicity	γ_i	$\mathrm{m^3\,mol^{-1}}$	[-, 0.0, 0.0, 0.0]
Ion-exchange characteristics	β_i	-	[-, 10, 9, 8]

4.2 Different injection profiles

In Figure 3, we show a comparison between two different salt injection profiles for a four-component LKM simulation with Langmuir MPM binding. The chosen parameters are listed in table 3. In the first simulation, Figure 3 a), a simple linear injection was used,

$$C_{\text{inj},0}(t) = \begin{cases} 0.0, & t \leq 1.0, \\ t - 1.0, & t > 1.0. \end{cases}$$

In the second simulation, Figure 3 b), a multi-step injection was used,

$$C_{\text{inj},0}(t) = \begin{cases} 0.0, & t < 1.0, \\ 5(t - 1.0), & 1.0 \leq t < 1.15, \\ 0.75, & 1.15 \leq t < 1.5, \\ 2.5(t - 1.5) + 0.75, & 1.5 \leq t < 1.6, \\ 1.0, & 1.6 \leq t < 2.1, \\ (t - 2.1) + 1.0, & 2.1 \leq t, \end{cases}$$

In both simulations, the regular components used a step input for the injection,

$$C_{\text{inj},i}(t) = \begin{cases} 1.0, & t < 0.5, \\ 0.0, & t \geq 0.5, \end{cases} \quad i \in \{1, 2, 3\}\,.$$

In the outlet profiles, we observe initial peaks between $t = 0.5$ and $t = 1.0$ for components C_1, C_2, and C_3 in both simulations. These peaks suggest that the column is fully saturated before the salt injection. Following the salt injection, there is a significant overlap between the peaks of C_1 and C_3 for the simple linear injection profile. Additionally, the peak for C_2 is quite wide. In contrast, with the multi-step injection, there is a clear separation between the peaks of the three components, and the peak for C_2 is much sharper.

4.3 Oscillations

In Figure 4, we have replicated an example shown in [3, 5, 23] to evaluate the performance of the TVDM limiter compared to the original scheme when shocks appear in the solution. It is a two-component LKM simulation with competitive Langmuir binding. The chosen parameters are listed in table 4. We

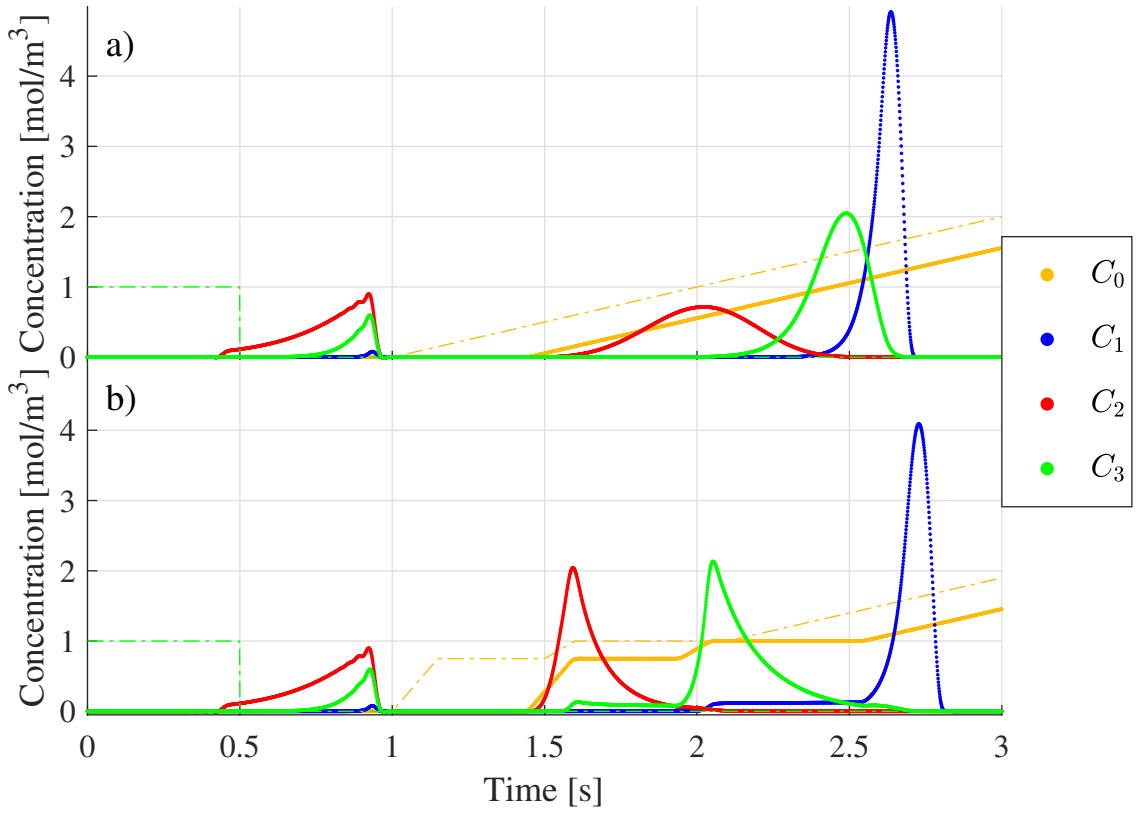

Figure 3: Outlet concentration profiles from four-component LKM simulation with Langmuir MPM binding using different salt injection profiles. The chosen parameters are listed in table 3. The dashed-dotted lines show the injection profiles. The injections for C_1 and C_2 are not seen since they are the same as for C_3.

Table 3: LKM and Langmuir MPM parameters used when comparing different injection profiles for the salt component.

Parameters	Symbol	Unit	Value
Components	n	-	4
Total porosity	ε	-	0.6
Column length	L	m	2.0
Mobile phase velocity	v	$\mathrm{m\,s^{-1}}$	4.5
Axial dispersion	D_L	$\mathrm{m^2\,s^{-1}}$	1.4×10^{-3}
Final time	τ	s	3.0
Column saturation capacity	$q_{s,i}$	$\mathrm{mol\,m^{-3}}$	[-, 4.70, 5.29, 3.70]
Adsorption rate	$k_{a,i}$	$\mathrm{m^3\,mol^{-1}\,s^{-1}}$	[-, 35.5, 1.59, 8.70]
Desorption rate	$k_{d,i}$	$\mathrm{m^{3\beta_i}mol^{-\beta_i}s^{-1}}$	[-, 10.18, 30.06, 20.00]
Hydrophobicity	γ_i	$\mathrm{m^3\,mol^{-1}}$	[-, 0.0, 0.0, 0.0]
Ion-exchange characteristics	β_i	-	[-, 21, 3, 12]

only present the results when using a 3rd-order polynomial. However, we also ran simulations with a 4th-order polynomial, and the results were nearly identical.

When using 50 elements, we observe that the original scheme shows severe oscillations and negative concentrations near the shocks. The TVDM limiter removes these oscillations and negative concentrations with the same settings. However, the cost is severe smearing of the shocks. Moving up to 100 elements, we observe a significant reduction of the oscillations and negative concentrations for the orig-

inal scheme. Similarly, we see a decrease in smearing when using the TVDM limiter, but it is still significant. With 200 elements, the oscillations have almost been eliminated from the original scheme, with only tiny amounts of negative concentrations showing. The smearing is still present when using the TVDM limiter but is significantly less severe than before.

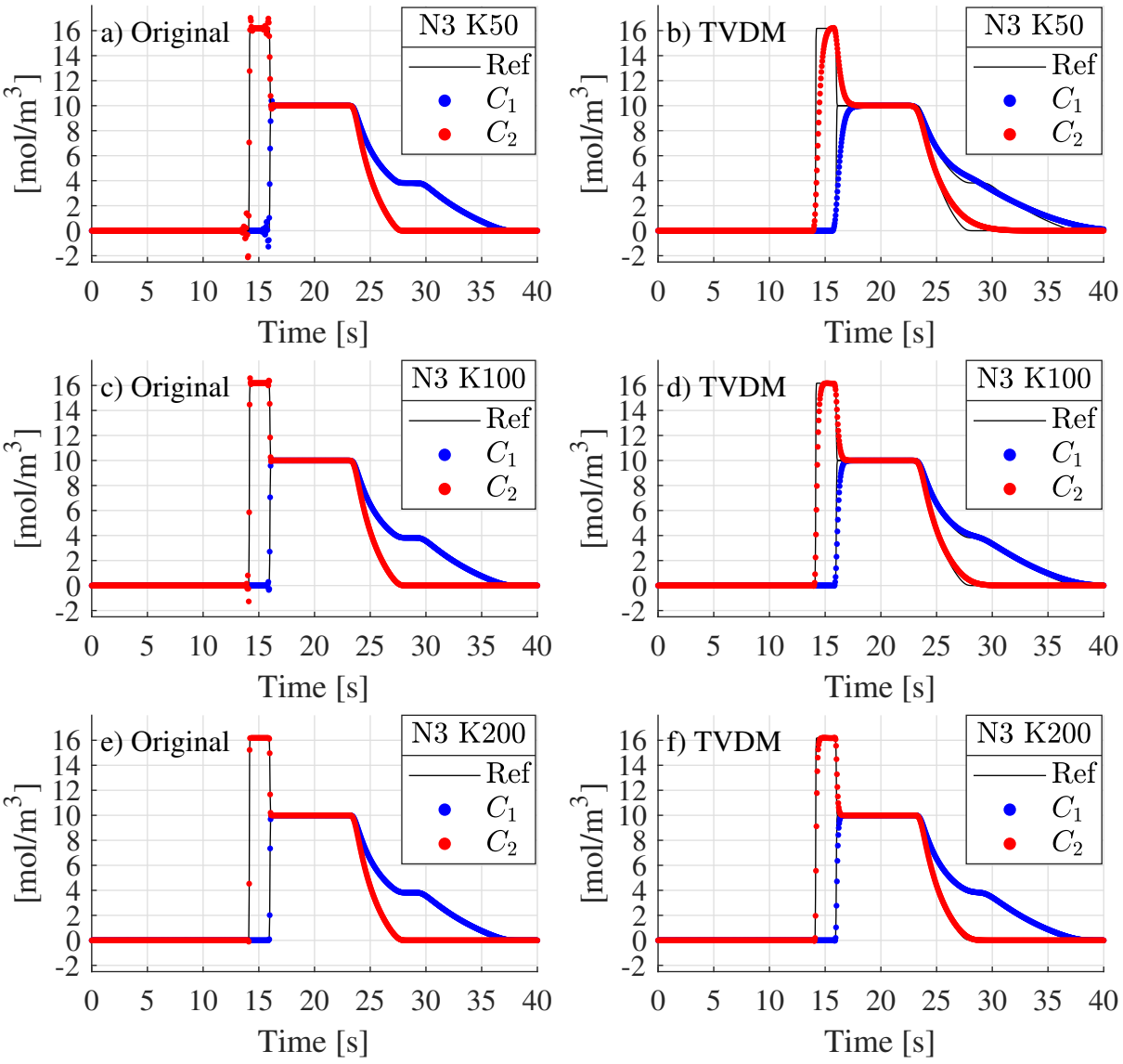

Figure 4: Comparison between original and TVDM schemes for different numbers of elements using a 3rd-order polynomial. The outlet concentration profiles are from two-component LKM simulations with competitive Langmuir binding using very low axial dispersion. The chosen parameters are listed in table 4. The high-resolution reference solution was obtained using DG CADET with a 10th-order polynomial and 512 elements.

5 Discussion

The results presented in this thesis provide insights into applying a nodal discontinuous Galerkin spectral element method for simulating processes in column liquid chromatography.

Table 4: Parameters for two-component LKM with competitive Langmuir binding. The adsorption and desorption coefficients have been multiplied by a large number, in this case 10^3, to artificially simulate the kinetic equation on a faster time scale to replicate the original examples shown in [3, 5, 23] where mass transfer kinetics are assumed to be fast.

Parameters	Symbol	Unit	Value
Components	n	-	2
Total porosity	ε	-	0.4
Column length	L	m	1.0
Mobile phase velocity	v	$\mathrm{m\,s^{-1}}$	0.1
Axial dispersion	D_L	$\mathrm{m^2\,s^{-1}}$	1.0×10^{-5}
Final time	τ	s	40.0
Column saturation capacity	$q_{s,i}$	$\mathrm{mol\,m^{-3}}$	$[10.0, 10.0]$
Adsorption rate	$k_{a,i}$	$\mathrm{m^3\,mol^{-1}\,s^{-1}}$	$[0.10, 0.05] \times 10^3$
Desorption rate	$k_{d,i}$	$\mathrm{s^{-1}}$	$[1.0, 1.0] \times 10^3$

5.1 Order-of-accuracy

Despite using an unusual reformulation in the weak forms, specifically for the mass balance equation, the order-of-accuracy tests demonstrate that our nodal DGSEM, coupled with a well-informed choice for the numerical fluxes, not only achieves but surpasses the theoretical convergence rates of $N+1$ for the upper-end of the considered ranges of elements. The drop in EOC for polynomial orders 4 and 5 for the largest number of elements is likely due to the temporal discretization error starting to dominate, considering the tolerances we set. However, it's important to note that the theoretical rates are based on an asymptotic behavior as the number of elements goes to infinity. Analysis using arbitrary precision arithmetic could be explored to understand the method's convergence fully. At the same time, this demonstrates that precise results can be achieved with few DoFs using our method.

5.2 Implications for chromatographic modeling

The test showing how different injection profiles of the mobile phase modulator affect the outcome clearly shows that accurate and reliable simulations can be essential for optimizing chromatographic processes, which are critical for the purification of biopharmaceuticals. The ability to model these processes with high fidelity can lead to better understanding and improvements in efficiency, yield, and product quality.

While our study focuses on the variation of the injection profile, it demonstrates the possibilities for optimizing other setup parameters, such as column length, mobile phase velocity, and porosity, to suit different needs. The power of numerical simulations like ours lies in their ability to expedite the exploration of solution spaces for such optimization problems, far outpacing experimental procedures by leveraging computational speed. This underscores the potential of data-driven simulations, where real-time experimental data informs the simulations, enhancing our understanding of the ongoing experimental procedure.

5.3 Oscillations and limiters

Our study also evaluated the TVDM limiter's performance in controlling oscillations near shock regions. The results showed that when the original scheme exhibited severe oscillations and negative concentrations near shocks, the TVDM limiter's application effectively eliminated these issues. However, this improvement came at the cost of significant smearing of the shocks, which in many cases will not suffice as a solution to the problem. For both approaches, the best solution was to increase the resolution of

the spatial discretization, which minimized their respective downside. However, increasing the resolution increases the size of the problem, which adds computational cost. Therefore, this suggests further investigations of limiters that retain high-order accuracy while preserving the advantageous properties of the original DG scheme to handle challenging setups containing shocks. Attractive candidates could be WENO and OEDG methods. Alternatively, one could investigate how common these situations are in actual experiments and determine if such efforts are necessary or if the basic scheme suffices.

5.4 Future work

Implementing the method in a high-level language like Python enhances its comprehensibility and provides a robust foundation for future development. Python's intuitive syntax and readability enable developers to grasp the logic and flow of the method quickly, fostering quicker onboarding and collaboration within a team. This clarity not only aids in understanding the current implementation but also simplifies debugging and maintenance. Moreover, Python's extensive libraries and community support provide a wealth of resources and tools that can be leveraged to extend functionality. By establishing a clear, well-documented, and easily adaptable codebase in Python, future enhancements and iterations become more manageable, promoting sustainable and scalable development practices.

While we are yet to delve into the simulations' execution times, the potential of moving from an interpreted language like Python to a compiled one is promising for some practical application. The transition can lead to substantial performance gains, a crucial factor for data-driven simulations. It would empower us to simulate much larger systems within a reasonable time, instilling confidence in the software's scalability and efficiency.

However, to be clear, a low-level implementation does not have to replace a high-level implementation. Having complementary implementations in both a high-level language and a low-level language offers distinct advantages that cater to different stages of development. A high-level language like Python is invaluable for rapid prototyping due to its simplicity, readability, and vast ecosystem of libraries. This allows developers to quickly test ideas, iterate on designs, and validate concepts without getting bogged down by complex syntax or low-level details. Once the prototype proves successful, transitioning to a low-level language becomes beneficial since it offers greater control over system resources, optimized performance, and finer management of hardware-specific features. This combination enables a smooth workflow where the high-level implementation accelerates development and validation while the low-level implementation ensures the final product's efficiency, performance, and scalability.

Future work could expand the implementation to handle more types of models and binding equations. A clear limitation of the current implementation is that it can only handle systems of ODEs during temporal discretization. An implementation that can handle DAE systems would help establish its broader applicability and allow it to simulate equilibrium dispersive models with equilibrium isotherms, for example.

6 Conclusion

This thesis has demonstrated the potential of a nodal DGSEM in simulating chromatographic processes with high accuracy. The findings underscore the method's suitability for detailed and reliable modeling, which is essential for optimizing biopharmaceutical separations. Future research directions outlined provide a roadmap for enhancing the method's capabilities and extending its application, paving the way for further advancements.

References

[1]O. Coskun, "Separation techniques: chromatography", North Clin Istanb **3**, 156–160 (2016).

[2]G. Guiochon, D. G. Shirazi, and A. Felinger, *Fundamentals of preparative and nonlinear chromatography* (Academic Press, 2006).

[3]K. Meyer, J. K. Huusom, and J. Abildskov, "High-order approximation of chromatographic models using a nodal discontinuous galerkin approach", Computers & Chemical Engineering **109**, 68–76 (2018).

[4]S. Leweke and E. von Lieres, "Chromatography analysis and design toolkit (cadet)", Computers & Chemical Engineering **113**, 274–294 (2018).

[5]J. M. Breuer, S. Leweke, J. Schmölder, G. Gassner, and E. von Lieres, "Spatial discontinuous galerkin spectral element method for a family of chromatography models in cadet", Computers & Chemical Engineering **177**, 108340 (2023).

[6]D. Andersson, R. Sjögren, and B. Corbett, "Numerical simulation of the general rate model of chromatography using orthogonal collocation", Computers & Chemical Engineering **170**, 108068 (2023).

[7]E. von Lieres and J. Andersson, "A fast and accurate solver for the general rate model of column liquid chromatography", Computers & Chemical Engineering **34**, 1180–1191 (2010).

[8]S. Javeed, S. Qamar, W. Ashraf, G. Warnecke, and A. Seidel-Morgenstern, "Analysis and numerical investigation of two dynamic models for liquid chromatography", Chemical Engineering Science **90**, 17–31 (2013).

[9]J. S. Hesthaven and T. Warburton, *Nodal discontinuous Galerkin methods. Algorithms, analysis, and applications*, Vol. 54, Texts Appl. Math. (New York, NY: Springer, 2008).

[10]W. H. Reed and T. R. Hill, *Triangular mesh methods for the neutron transport equation*, (Oct. 1973) https://www.osti.gov/biblio/4491151.

[11]B. Cockburn, G. E. Karniadakis, and C.-W. Shu, "The development of discontinuous galerkin methods", in Discontinuous galerkin methods, edited by B. Cockburn, G. E. Karniadakis, and C.-W. Shu (2000), pp. 3–50.

[12]J. Qiu and C.-W. Shu, "Runge–kutta discontinuous galerkin method using weno limiters", SIAM J. Scientific Computing **26**, 907–929 (2005).

[13]X. Zhong and C.-W. Shu, "A simple weighted essentially nonoscillatory limiter for runge–kutta discontinuous galerkin methods", Journal of Computational Physics **232**, 397–415 (2013).

[14]K. Meyer, S. Leweke, E. von Lieres, J. K. Huusom, and J. Abildskov, "Chromatech: a discontinuous galerkin spectral element simulator for preparative liquid chromatography", Computers & Chemical Engineering **141**, 107012 (2020).

[15]P. Söderström, "Physics-Informed Neural Networks for Liquid Chromatography", https://urn.kb.se/resolve?urn=urn:nbn:se:umu:diva-196783, Dissertation (Umeå University, 2022).

[16]D. Karlsson, N. Jakobsson, A. Axelsson, and B. Nilsson, "Model-based optimization of a preparative ion-exchange step for antibody purification", Journal of Chromatography A **1055**, 29–39 (2004).

[17]W. R. Melander, Z. El Rassi, and C. Horváth, "Interplay of hydrophobic and electrostatic interactions in biopolymer chromatography: effect of salts on the retention of proteins", Journal of Chromatography A **469**, 3–27 (1989).

[18]Y.-T. Zhang and C.-W. Shu, "Chapter 5 - ENO and WENO Schemes", in *Handbook of numerical methods for hyperbolic problems*, Vol. 17, edited by R. Abgrall and C.-W. Shu, Handbook of Numerical Analysis (Elsevier, 2016), pp. 103–122.

[19]M. Peng, Z. Sun, and K. Wu, *OEDG: Oscillation-eliminating discontinuous Galerkin method for hyperbolic conservation laws*, 2023.

[20]F. Bassi and S. Rebay, "A high-order accurate discontinuous finite element method for the numerical solution of the compressible navier–stokes equations", Journal of Computational Physics **131**, 267–279 (1997).

[21]B. Cockburn and C.-W. Shu, "The local discontinuous galerkin method for time-dependent convection-diffusion systems", SIAM Journal on Numerical Analysis **35**, 2440–2463 (1998).

[22]P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python", Nature Methods **17**, 261–272 (2020).

[23]O. Shipilova, T. Sainio, and H. Haario, "Particle transport method for simulation of multicomponent chromatography problems", Journal of Chromatography A **1204**, 62–71 (2008).