USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA
FACULTADE DE MATEMÁTICAS

Traballo Fin de Grao

Resolución numérica de problemas de fluxo de sangue nun conducto

Brais Romero García

Curso 2023/2024

UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

GRAO DE MATEMÁTICAS

Traballo Fin de Grao

RESOLUCIÓN NUMÉRICA DE PROBLEMAS DE FLUXO DE SANGUE NUN CONDUCTO

Brais Romero García

Xullo, 2024

UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Traballo proposto

Área de Coñecemento: Matemática Aplicada
Título: Resolución numérica de problemas de fluxo de sangue nun conducto
Breve descrición do contido
O obxectivo do TFG é coñecer en primeiro lugar modelos de fluxo en sangue nun conducto, que pode ser unha vea ou unha arteria, e posteriormente, estudar métodos numéricos que permitan a súa resolución. O traballo conterá asemade a validación dos mesmos con solucións analíticas coñecidas e datos experimentais, facendo as correspondentes análises de converxencia e estabilidade. Empregaranse métodos de volumes finitos e farase a conexión dos mesmos con métodos para a resolución de EDP's estudados no Grao en Matemáticas. As titoras facilitarán códigos en MATLAB que serán a base para analizar os métodos numéricos cos que se resolverán os modelos.

Índice xeral

Resumo

Propónse un método de volumes finitos completamente implícito para a simulación de fluxo de sangue 1D. Comezamos introducindo o concepto de sistemas de leis de conservación e os métodos de volumes finitos como unha ferramenta para achar solucións numéricas para este tipo de EDPs. Logo, presentamos o modelo de fluxo de sangue 1D que, para a súa posterior discretización, é dividido en tres subsistemas: un para os termos convectivos, un das variables difusivas e un para a presión. Estúdase un método semi-implícito de volumes finitos que resolve de maneira implícita as dúas últimas etapas e que discretiza explícitamente a etapa convectiva. Despois, propoñemos un novo esquema que toma un enfoque implícito para os termos convectivos usando un método de Newton inexacto combinado cun algoritmo BiCGSTAB. Para a discretización dos termos de fluxo, usamos as funcións de fluxo numérico de Rusanov ou Ducros. Finalmente, validamos o novo método mediante comparacións co esquema semi-implícito e solucións exactas a través serie de problemas de Riemann no contexto de simulación do fluxo de sangue.

Abstract

A fully-implicit finite volume method for the simulation of one-dimensional blood flow is proposed. We first introduce the concept of hyperbolic systems of conservation laws, and the finite volume methods as a mean to find numerical solutions for this type of PDEs. Then, the 1D blood flow model is presented and split into three subsystems: the first one containing convective terms, the second one containing diffusive terms and the third one for the pressure variable. We then study a semi-implicit finite volume method that solves implicitly the last two stages and that discretizes explicitly the convective stage. Afterwards, we propose a novel scheme that discretizes the convective terms using an inexact Newton method combined with a BiCGSTAB algorithm. For the discretization of the flux term, we employ Rusanov or Ducros numerical flux functions. Finally, the new method is validated by comparing it with the semi-implicit scheme and exact solutions in a set of Riemann Problems in the context of blood flow simulation.

Introdución

O aparello cardiovascular, encargado de transportar nutrientes e osíxeno a través do corpo, é un dos sistemas máis importantes nos seres humanos, e o seu bo funcionamento é primordial para garantir a integridade e saúde do noso organismo. Sen embargo, nas sociedades occidentais e, concretamente, en España, as afeccións relacionadas co aparello circulatorio seguen sendo a principal causa de morte por enfermidade. No ano 2022 [1], se tomamos conxuntamente as insuficiencias cardíacas, as enfermidades isquémicas do corazón, as enfermidades cerebrovasculares e as doenzas relacionadas con problemas de hipertensión, éstas representan a causa dun 19.2 % das mortes en España, e se reducimos o grupo poblacional ás persoas maiores de 60 anos, o porcentaxe é aínda maior: dun 26.13 % [2].

Polo tanto, o interese por entender o funcionamento do sistema circulatorio cada vez é maior. Cos avances tecnolóxicos das últimas décadas, empeza a ser viable simular modelos orientados a intentar predicir o comportamento do sangue en calquera tipo de situacións, tanto nas que pode estar en risco a vida do paciente (por exemplo, unha obstrución ou unha rotura), como noutras nas que sexa relevante estudar certas patoloxías con maior profundidade.

Os avances informáticos das últimas décadas están facilitando o estudo do sistema cardiovascular e permitindo a creación de novos métodos e modelos, tanto matemáticos como numéricos [3, 4]. A forma máis precisa de definir o fluxo de sangue nun sistema de veas ou arterias é a través de métodos tridimensionais que resolven as ecuacións de Navier-Stokes. Este sistema de ecuacións en derivadas parciais é o modelo mais extendido no estudo do comportamentos dos medios fluidos xa que permite reproducir o seu comportamento con gran exactitude capturando complexos fenómenos tridimensionais. Por iso, numerosos investigadores, como [5, 6], fixeron un esforzo para entender e resolver as ecuacións de Navier-Stokes 3D e a súa aplicación ao estudo do fluxo sanguíneo

Sen embargo, debido á complexidade do sistema cardiovascular, estudar a interacción entre o sangue e as paredes das veas e arterias é complexo, tanto a nivel teórico como computacional. Sempre que é posible, faise algunha simplificación, sendo unha das máis habituais a reducción do número de dimensións espaciais a unha. Os modelos reducidos 1D son moi utilizadas nas matemáticas cardiovasculares e na enxeñería biomédica, debido ao bo acordo que amosan os

resultados obtidos en 3D [7]. Este tipo de modelos poden ser aplicados a todo tipo de problemas, tanto para vasos sanguíneos illados [8, 9, 10] como para redes deles [11, 12, 13], con resultados favorables. Incluso son habituais os modelos que mesturan varias dimensións [14, 15].

Esta redución de variables espaciais trae como vantaxe unha simplificación teórica e de implementación, á parte dunha evidente mellora na velocidade computacional; por este motivo, son modelos moi atractivos para seren utilizados en aplicacións reais, como a planificación de intervencións vasculares, a modelización da estenosis das arterias renais [9] ou a simulación do fluxo en redes do sistema circulatorio [16, 17].

Dentro da bibliografía, atopamos métodos que teñen unha resolución semi-implícita [10, 11, 12]; nestes enfoques, as ecuacións do sangue son modeladas de maneira que poida existir unha resolución por etapas, onde algunhas variables son achadas explícitamente e outras cunha discretización implícita. Existen, tamén, exemplos de problemas resoltos con métodos puramente implícitos [18] e puramente explícitos [19], sendo estes últimos os máis habituais.

Vexamos algúns exemplos con máis detalle: en Hasan [12] móstranse tres algoritmos diferentes, todos eles baseados nunha discretización do espazo baseada no método LCG (*locally conservative Galerkin*) [20]. O primeiro deles integra un método cunha expansión de Taylor, onde todos os elementos da ecuación do sangue menos os termos de fluxo nas fronteiras son resoltos de maneira implícita. Os outros dous son moi similares: no segundo, introdúcese unha modificación do termo convectivo utilizando o método *streamline upwind Petrov–Galerkin* (SUPG) [21], para reducir oscilacións; no terceiro, intégrase o método *forward in time and central in space* (FTCS) para as ecuacións convectivas, buscando así que sexa incondicionalmente estable.

Con respecto a Sherwin [11], enfróntanse ao problema do fluxo sanguíneo con dous sistemas de ecuación equivalentes: un determinado polas variables de área e a velocidade do fluxo na sección, e outro rexido pola área e o fluxo de masa. Para a primeira delas aplícase un método de Galerkin discontinuo, mentres que a segunda soluciónase cun método de Taylor-Galerkin. Os dous esquemas son validados cun esquema modelo de solución coñecida.

Noutro deles, en Casulli [13], trabállase con ecuacións do sangue en dúas dimensións, onde unha representa a lonxitude axial e outra a radial. O procedemento que seguen é modelizar a presión da ecuación do momento e a velocidade na ecuación de fronteira móvil co *θ-method* [22]. Por outra parte, para introducir maior estabilidade, os termos viscosos son discretizados implícitamente, e os advectivos son modelizados cunha metodoloxía de Euler-Lagrange. Este procedemento lévanos a un método que pode ser aplicado tamén a 1D.

Os enfoques semi-implícitos poden ser moi diversos, e a principal razón da súa popularidade é que, como xa puidemos ver, temos liberdade para escoller que método numérico queremos utilizar para resolver cada subsistema, e así combinar varios esquemas. Normalmente, existen unha ou varias etapas que poden ser solucionadas explícitamente sen sacrificar demasiado a estabili-

dade; ademais, as resolucións explícitas soen ser máis rápidas e máis fáciles de implementar en comparación coas semi-implícitas e as puramente implícitas. Con todo, estes métodos introdúcennos limitacións no paso de tempo para garantir a converxencia e é por isto que neste traballo estudaremos un método semi-implícito para a resolución do fluxo de sangue en conductos, e, posteriormente, propoñeremos unha modificación do mesmo para obter un esquema puramente implícito que nos elimine a condición de estabilidade do esquema e, con iso, a restricción do paso temporal.

Neste TFG, introduciremos os sistemas de leis de conservación hiperbólicas e os métodos de volumes finitos no Capítulo 1. A continuación, no Capítulo 2, estudaremos a metodoloxía proposta en Lucca et al. [10]. A partires do seu método semi-implícito de volumes finitos, resolveremos no Capítulo 3 a variable convectiva de maneira implícita, rematando así cun método completamente implícito. Finalmente, compararemos no Capítulo 4 os dous algoritmos e analizaremos se obtemos unha mellora en termos de eficiencia computacional, estabilidade e converxencia.

Capítulo 1

Métodos de volumes finitos para a resolución de sistemas de leis de conservación

Os sistemas de leis de conservación hiperbólicas son fundamentais na física debido a que permiten estudar mecanismos onde se conserva algunha magnitude. Estes sistemas modelizan fenómenos complexos como o fluxo de sangue, a propagación de ondas acústicas, o fluxo de tráfico ou o fluxo de auga en canais. Neste capítulo introducimos as nocións básicas destes modelos físico-matemáticos, ademais dunha introdución aos métodos de volumes finitos e como poden ser aplicados para resolver este tipo de ecuacións.

1.1. Leis de conservación hiperbólicas

Un **sistema de leis de conservación hiperbólicas** é un sistema de ecuacións en derivadas parciais (EDPs) que se rixe pola lei física na que a taxa de cambio nunha rexión dunha certa cantidade coincide co fluxo desta a través do contorno da rexión. A forma xeral para sistemas unidimensionais deste tipo vén definida en [23]:

Definición 1.1. Sexa Ω un subconxunto de $\mathbb{R}^d$ e sexa f unha función $\mathcal{C}^\infty$ de Ω en $\mathbb{R}^d$:

$$f : \Omega \longrightarrow \mathbb{R}^d. \tag{1.1}$$

A forma xeral dun sistema de leis de conservación é

$$q_t + f_x(q) = 0,\ x \in \mathbb{R},\ t > 0, \tag{1.2}$$

onde $q = (q_1, \dots, q_d)$, o vector de variables conservativas, é a función vectorial:

$$\begin{aligned} q : \mathbb{R} \times [0, \infty] &\longrightarrow \Omega, \\ (x, t) &\longrightarrow (q_1, \dots, q_d). \end{aligned} \tag{1.3}$$

O conxunto Ω chamase conxunto de estados e a función $f = (f1, \dots, f_d)$ é o fluxo físico ($f_i : \Omega \longrightarrow \mathbb{R}$).

Se definimos o fluxo positivo de esquerda a dereita podemos tomar un intervalo $I \subset \mathbb{R}$ e integrar (1.2) ao longo del, obtendo:

$$\frac{d}{dt} \int_I q(x,t)\ dx = f(q(a,t)) - f(q(b,t)), \tag{1.4}$$

que representa a lei conservativa en forma integral.

Traballaremos cos problemas de Cauchy para sistemas de leis de conservación hiperbólicas:

Definición 1.2. (Problema de Cauchy)
Atopar unha función $q : (x,t) \in \mathbb{R}^d \times [0,\infty] \longrightarrow q(x,t) \in \Omega$, solución de (1.2), que verifique a condición inicial

$$q(x,0) = q_0(x),\ x \in \mathbb{R}, \tag{1.5}$$

onde $q_0 : x \in \mathbb{R} \longrightarrow q_0(x) \in \Omega$ é unha función coñecida.

E, máis concretamente, centrarémosnos no Capítulo 4 en solucións de problemas de Riemann, que non son máis que un problema de Cauchy onde a condición inicial vén dada por

$$q_0(x) = \begin{cases} q_L, \text{ se } x < 0, \\ q_R, \text{ se } x > 0. \end{cases} \tag{1.6}$$

Ecuación de transporte

O exemplo máis sinxelo dunha ecuación unidimensional de leis de conservación é a **ecuación de transporte**, na que $d = 1$ e o fluxo é proporcional á variable conservativa:

$$f(q) = \lambda q, \tag{1.7}$$

e onde λ é unha constante que pode ser interpretada como a velocidade de propagación. A lei conservativa asociada a este problema é:

$$q_t(x,t) + \lambda q_x(x,t) = 0, \tag{1.8}$$

e, dada a condición inicial (1.5), a solución da ecuación (1.8) vén dada por

$$q(x,t) = q_0(x - \lambda t),\ -\infty < x < \infty,\ t \in [0,\infty). \tag{1.9}$$

Para obter esta solución, introducimos as curvas características:

$$t \longrightarrow X(t),\ \frac{dX}{dt}(t) = \lambda, \tag{1.10}$$

e tomamos a derivada temporal de q:

$$\frac{dq}{dt}(X(t),t) = q_x(X(t),t)\frac{dX}{dt}(t) + q_t(X(t),t) = q_x(X(t),t)\lambda + q_t(X(t),t) = 0, \tag{1.11}$$

que vemos que é 0, polo que a solución é constante ao longo delas.

Se calculamos a solución nun punto arbitrario (x^*, t^*) e chamamos $X^*(t)$ á curva característica que pasa polo punto e que satisfai (1.5):

$$\frac{dX^*}{dt}(t) = \lambda,\ X^*(t^*) = x^*, \tag{1.12}$$

a súa expresión será:

$$X^*(t) = X^*(t^*) + \lambda(t - t^*) = x^* + \lambda(t - t^*), \tag{1.13}$$

e, polo tanto, para $t = 0$, é

$$X^*(0) = x^* - \lambda t^*. \tag{1.14}$$

Tendo en conta (1.11), podemos obter a solución para os puntos da curva:

$$q(x^*, t^*) = q(X^*(0), 0) = q_0(X^*(0)) = q_0(x^* - \lambda t^*), \tag{1.15}$$

e, como (x^*, t^*) son arbitrarios, a solución xeral virá dada por:

$$q(x,t) = q_0(x - \lambda t). \tag{1.16}$$

No caso dun problema de Riemann (1.6), a solución será:

$$q(x,t) = q_0(x - \lambda t) = \begin{cases} q_L,\ \text{se } x - \lambda t < 0, \\ q_R,\ \text{se } x - \lambda t > 0, \end{cases} \tag{1.17}$$

onde $q_L, q_R \in \mathbb{R}$.

Modelo do fluxo de tráfico

Consideremos o fluxo de tráfico nunha autopista. Podemos chamar ρ á densidade de coches, expresada en coches por kilómeto, e u á velocidade. Tomaremos o dominio de rho como $[0, \rho_m]$, con rho_m o valor para o que o fluxo colapsa e se forma un atasco. Como o número de coches ten que ser constante, a densidade e a velocidade seguen a seguinte relación:

$$\rho_t + (\rho u)_x = 0. \tag{1.18}$$

Para atopar unha lei de conservación para ρ, intentamos expresar u en función de ρ. Esta relación ten sentido se interpretamos que, a pesar de que nunha autopista podemos alcanzar unha velocidade máxima de u_m, en situacións de moito tráfico teremos que reducila. O modelo máis simple consiste na relación lineal:

$$u(\rho) = u_m \left(1 - \frac{\rho}{\rho_m}\right). \tag{1.19}$$

En casos de densidade nula, o coche pode alcanzar unha velocidade de u_m, pero achégase a cero cando ρ está cerca de ρ_m.

Reemplazando o valor de u en (1.18), temos:

$$\rho_t + f(\rho)_x = 0, \tag{1.20}$$

onde a función de fluxo vén dada por

$$f(\rho) = \rho u_m \left(1 - \frac{\rho}{\rho_m}\right). \tag{1.21}$$

Outra modelización, que probou ser efectiva en problemas reais [24], é a dada pola función de fluxo

$$f(\rho) = \rho u_m \ln\left(\frac{\rho_m}{\rho}\right). \tag{1.22}$$

Ecuación de Burgers

A ecuación escalar parabólica:

$$\frac{\partial q}{\partial t} + \frac{\partial}{\partial x}\left(\frac{q^2}{2}\right) = 0, \tag{1.23}$$

foi introducida por Burgers como o modelo en derivadas parciais máis sinxelo para o fluxo dun fluido.

Máis exemplos de ecuacións de leis de conservación hiperbólicas poden ser consultados en [25] ou [26].

1.2. Métodos de volumes finitos aplicados a sistemas de leis de conservación

Os **métodos de volumes finitos unidimensionais** consisten en dividir o dominio espacial en varios intervalos, chamados volumes finitos ou celas, e calcular unha aproximación do promedio

da variable conservativa en cada un deles. O dominio $[a, b]$ discretízase cunha malla $\mathcal{M}_{\Delta x}$, onde $x_i, i = 1, \dots, M$ son os nodos e Δx é a norma de $\mathcal{M}_{\Delta x}$:

$$\Delta x = \sup_{x_i \in \mathcal{M}_{\Delta x}} |x_i - x_{i-1}|. \tag{1.24}$$

Por simplicidade, imos traballar cunha distancia entre nodos constante, pero os resultados poden adaptarse de maneira sinxela a distancias variables. Polo tanto, o volume finito ou cela C_i defínese como:

$$C_i = \left(x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}\right) = \left(x_i - \frac{\Delta x}{2}, x_i + \frac{\Delta x}{2}\right). \tag{1.25}$$

A integral da lei conservativa (1.2) en cada cela C_i lese

$$\frac{d}{dt}\int_{C_i} q(x,t)\ dx = f(q(x_{i-\frac{1}{2}}, t)) - f(q(x_{i+\frac{1}{2}}, t)), \tag{1.26}$$

e, integrando esta expresión dende $t = t^n$ ata $t = t^{n+1}$ obtemos

$$\int_{C_i} q(x, t^{n+1})\ dx - \int_{C_i} q(x, t^n)\ dx = \int_{t^n}^{t^{n+1}} f(q(x_{i-\frac{1}{2}}, t))\ dt - \int_{t^n}^{t^{n+1}} f(q(x_{i+\frac{1}{2}}, t))\ dt, \tag{1.27}$$

que, reordeando e dividindo entre Δx convértese en

$$\begin{aligned} \frac{1}{\Delta x}\int_{C_i} q(x, t^{n+1})\ dx = \frac{1}{\Delta x}\int_{C_i} q(x, t^n)\ dx - \\ -\frac{1}{\Delta x}\left[\int_{t^n}^{t^{n+1}} f(q(x_{i-\frac{1}{2}}, t))\ dt - \int_{t^n}^{t^{n+1}} f(q(x_{i+\frac{1}{2}}, t))\ dt\right]. \end{aligned} \tag{1.28}$$

Podemos definir o valor q_i^n coma unha aproximación do valor medio de q na cela C_i e no tempo t^n:

$$q_i^n \approx \frac{1}{\Delta x}\int_{C_i} q(x, t^n)\ dx, \tag{1.29}$$

de tal forma que exista unha función $q^n : [a, b] \longrightarrow \mathbb{R}$ para cada tempo t^n definida como

$$q^n(x) = q_i^n, \text{ para o } i \text{ tal que } x \in C_i. \tag{1.30}$$

Dado q_i^n, queremos aproximar q_i^{n+1}, que sería o valor medio na cela C_i e no tempo t^{n+1}, despois dun paso de tempo $\Delta t = t^{n+1} - t^n$ (que este si pode non ser constante). Normalmente non seremos capaces de obter a solución exacta das integrais de (1.28), polo que temos que aplicar métodos numéricos da forma:

$$q_i^{n+1} = q_i^n - \frac{\Delta t}{\Delta x}\left(f_{i+\frac{1}{2}}^n - f_{i-\frac{1}{2}}^n\right), \tag{1.31}$$

onde $f_{i-\frac{1}{2}}^n$, $f_{i+\frac{1}{2}}^n$ son aproximacións do fluxo no tempo $t = t^n$ e nas posicións $x = x_{i-\frac{1}{2}}$, $x = x_{i+\frac{1}{2}}$, respectivamente.

Podemos aproximar $f^n_{i-\frac{1}{2}}$ baseándonos só nos valores de q^n_{i-1} e q^n_i a través dunha función de fluxo numérico, ϕ, de tal maneira que

$$f^n_{i-\frac{1}{2}} = \phi(q^n_{i-1}, q^n_i), \tag{1.32}$$

sendo o resultado análogo para $f_{i+\frac{1}{2}}$. Así, o método numérico pasa a ser:

$$q^{n+1}_i = q^n_i - \frac{\Delta t}{\Delta x}\left[\phi(q^n_i, q^n_{i+1}) - \phi(q^n_{i-1}, q^n_i)\right], \tag{1.33}$$

onde o esquema numérico que obteremos dependerá só da expresión de ϕ.

Fixémonos que tamén poderíamos considerar tomar os fluxos avaliados no tempo novo, $f(q^{n+1})$, e realizando un procedemento similar chegamos a un esquema da forma:

$$q^{n+1}_i = q^n_i - \frac{\Delta t}{\Delta x}\left[\phi(q^{n+1}_i, q^{n+1}_{i+1}) - \phi(q^{n+1}_{i-1}, q^{n+1}_i)\right], \tag{1.34}$$

que habería que resolver de forma implícita utilizando outro método numérico, coma un método de punto fixo ou un método de Newton.

1.2.1. Consistencia e estabilidade

O fluxo numérico debe aproximar as integrais de (1.28) e, en particular, se a variable conservativa é constante en x ($q(x,t) = \widetilde{q}$), entón a solución non cambiará co tempo, e as integrais virán dadas por $f(\widetilde{q})$. Polo tanto, precisamos que se cumpra a relación

$$\phi(\widetilde{q}, \widetilde{q}) = f(\widetilde{q}), \tag{1.35}$$

para calquera valor $\widetilde{q}$.

Por outra parte, o análise de estabilidade de von Neumann dános un estudo desta propiedade para métodos en diferencias finitas para ecuacións en derivadas parciais, e unha descripción do procedemento pode ser atopada en [27]. Todos os métodos explícitos de volumes finitos deben verificar a condición CFL (Courant, Friedrichs and Lewy), descrita en [28] e definida como

$$\frac{|\lambda|\Delta t}{\Delta x} \leq 1, \tag{1.36}$$

onde λ é o máximo dos autovalores do jacobiano da función de fluxo.

1.2.2. Exemplos de métodos de volumes finitos

Forwards in Time Centered in Space

Tomemos os seguintes cocientes como aproximacións das ecuacións en derivadas parciais:

$$q_t \approx \frac{q^{n+1}_i - q^n_i}{\Delta t},\ q_x \approx \frac{q^n_{i+1} - q^n_{i-1}}{2\Delta x}. \tag{1.37}$$

Se recordamos, por exemplo, a ecuación de transporte (1.8) e sustituimos, obtemos:

$$\frac{q_i^{n+1} - q_i^n}{\Delta t} + \lambda \frac{q_{i+1}^n - q_{i-1}^n}{2\Delta x} = 0, \tag{1.38}$$

e se despexamos q_i^{n+1}, teremos

$$q_i^{n+1} = q_i^n - \frac{\lambda \Delta t}{2\Delta x}(q_{i+1}^n - q_{i-1}^n). \tag{1.39}$$

Podemos definir a función de fluxo numérico e escribir o método en forma conservativa se observamos que

$$f_{i-\frac{1}{2}}^n = \phi(q_{i-1}^n, q_i^n) = \frac{\lambda}{2}\left[q_{i-1}^n + q_i^n\right], \tag{1.40}$$

sendo análoga a definición de $f_{i+\frac{1}{2}}$. Sen embargo, este método é incondicionalmente inestable, polo que non ten aplicacións fóra do campo teórico.

Esquema clásico de Lax-Friedrichs

O método de Lax-Friedrichs para un problema de leis de conservación hiperbólicas unidimensionais, que pode ser consultado en [29], formúlase como:

$$q_i^{n+1} = \frac{1}{2}(q_{i-1}^n + q_{i+1}^n) - \frac{\Delta t}{2\Delta x}\left(f(q_{i+1}^n) - f(q_{i-1}^n)\right). \tag{1.41}$$

Aínda que non parece conservativo a primeira vista, operando chegamos á función de fluxo numérico

$$\phi^{LF}(q_i^n, q_{i+1}^n) = \frac{f(q_i^n) + f(q_{i+1}^n)}{2} - \frac{\alpha^{LF}}{2}(q_{i+1}^n - q_i^n), \tag{1.42}$$

onde $\alpha^{LF} = \frac{\Delta t}{\Delta x}$. Aínda que é moi similar ao visto en 1.39, a sustitución de q_i^n por $\frac{1}{2}(q_{i-1}^n + q_{i+1}^n)$ na aproximación da derivada temporal fai que o método pase a ser consistente e condicionalmente estable no caso lineal.

Fluxo numérico de Rusanov

O método de Lax-Friedrichs pode resultar difusivo cerca de choques, e isto pode ser debido á elección do termo α^{LF}. Unha mellor elección *local* deste termo é a dada por:

$$\alpha_{i+\frac{1}{2}}^R = \text{máx}(|f'(q_i^n)|, |f'(q_{i+1}^n)|). \tag{1.43}$$

O fluxo resultante, chamado fluxo de Rusanov, é:

$$f_{i+\frac{1}{2}} = \phi^R(q_i^n, q_{i+1}^n) = \frac{1}{2}(f(q_i^n) + f(q_{i+1}^n)) - \frac{\alpha_{i+\frac{1}{2}}^R}{2}(q_{i+1}^n - q_i^n). \tag{1.44}$$

Fluxo numérico de Ducros

Outro fluxo numérico que toma o termo α de maneira local é o de Ducros, que toma o fluxo da suma en vez da suma dos fluxos, da seguinte maneira:

$$f_{i+\frac{1}{2}} = \phi^D(q_i^n, q_{i+1}^n) = f\left(\frac{1}{2}(q_i^n + q_{i+1}^n)\right) - \frac{\alpha_{i+\frac{1}{2}}^D}{2}(q_{i+1}^n - q_i^n), \tag{1.45}$$

con $\alpha_{i+\frac{1}{2}}^D = \text{máx}(|f'(q_i^n)|, |f'(q_{i+1}^n)|)$.

Outros métodos numéricos de volumes finitos inclúen os esquemas descentrados ou o método de Godunov. Para a resolución da ecuación do sangue traballaremos cos métodos de Rusanov e de Ducros.

Capítulo 2

Esquema semi-implícito de volumes finitos para o fluxo de sangue

Neste capítulo introducimos o modelo para o fluxo de sangue en vasos sanguíneos e presentamos un método numérico baseado na partición do sistema de EDPs en tres subsistemas máis sinxelos de solucionar. Este enfoque, presentado en [10], permítenos obter un método semi-implícito que resolve cada un destas etapas con métodos numéricos adecuados.

2.1. Modelo para fluxos fisiolóxicos en conductos flexibles

Para o modelo do fluxo da sangue imos interpretar un vaso sanguíneo coma un cilindro axisimétrico deformable con certas magnitudes en cada punto, que serán: a área do tubo, medida en m^2; a presión do sangue contra as paredes do conducto, medida en Pa; e a velocidade do fluido, medida en m/s. O valor do cálculo numérico destas magnitudes para cada punto x indican unha aproximación da magnitude na sección no momento t.

O **fluxo do sangue como un fluido hidrostático e incompresible en vasos sanguíneos** pode ser descrito por un sistema de dúas ecuacións en derivadas parciais, representando a conservación da masa e do momento:

$$\frac{\partial A}{\partial t} + \frac{\partial Au}{\partial x} = 0, \tag{2.1a}$$

$$\frac{\partial Au}{\partial t} + \frac{\partial Au^2}{\partial x} + \frac{A}{\rho}\frac{\partial p}{\partial x} = -su, \tag{2.1b}$$

onde $x \in [x_L, x_R] \subset \mathbb{R}$ é a coordeada espacial, t é a coordeada temporal, $A(x,t)$ é a área da sección e $u(x,t)$, $p(x,t)$ son a velocidade media e a presión media do fluido na sección, respectivamente. Por outra parte, ρ representa a densidade do fluido e s é a forza de fricción que xera o conducto sobre o sangue.

Como temos tres variables descoñecidas, A, Au e p, pero só dúas ecuacións, requerimos dunha relación de peche adicional. Polo tanto, imos asociar a área da sección do vaso sanguíneo coa presión a través da chamada lei tubular, que describe a súa parede como un anel viscoelástico independente, dando lugar á seguinte relación:

$$p(x,t) = p_e + \psi(A, A_0, K) + \varphi(A, A_0)\frac{\partial A}{\partial t}, \tag{2.2}$$

onde p_e é a presión externa, ψ representa a compoñente elástica da presión, que ten expresión

$$\psi(A, A_0, K) = K(x)\theta(A, A_0), \text{ con } \phi(A, A_0) = \left(\frac{A}{A_0}\right)^m - \left(\frac{A}{A_0}\right)^n, \tag{2.3}$$

e

$$\varphi(A, A_0) = \frac{\Gamma}{A_0\sqrt{A}}, \tag{2.4}$$

que modeliza o comportamento viscoelástico da parede do conducto. Nas ecuacións anteriores, A_0 denota a área da sección de referencia, $K(x)$ é o coeficiente de rixidez da parede do vaso e Γ correspóndese co parámetro viscoelástico. Os parámetros m e n están relacionados co comportamento dos vasos sanguíneos, e cambian entre veas e arterias.

A partires da ecuación de conservación de masa, (2.1a), e seguindo o procedemento de Cauchy-Kovalewskaya [26], a derivada parcial temporal da área pode ser expresada como una derivada temporal pola relación

$$\frac{\partial A}{\partial t} = -\frac{\partial Au}{\partial x}. \tag{2.5}$$

Aplicando (2.5) na lei tubular (2.2), o gradiente de presión en (2.1b) pódese calcular como

$$\frac{\partial p}{\partial x} = \frac{\partial \hat{p}}{\partial x} - \frac{\partial}{\partial x}\left(\varphi\frac{\partial Au}{\partial x}\right), \tag{2.6}$$

onde $\hat{p} = p_e + \psi(A, A_0, K)$ describe a deformación elástica da parede do conduto cando hai variacións de presión.

Introducimos agora a notación $q = Au$ para a variable conservativa do fluxo de masa, $f(q) = \frac{q^2}{A}$ para o fluxo convectivo e, tendo en conta (2.6) podemos reescribir o sistema (2.1) como

$$\frac{\partial A}{\partial t} + \frac{\partial q}{\partial x} = 0, \tag{2.7a}$$

$$\frac{\partial q}{\partial t} + \frac{\partial f(q)}{\partial x} + \frac{A}{\rho}\frac{\partial \hat{p}}{\partial x} - \frac{A}{\rho}\frac{\partial}{\partial x}\left(\varphi\frac{\partial q}{\partial x}\right) = -s\frac{q}{A}. \tag{2.7b}$$

Sumando e restando na ecuación do momento a derivada espacial da presión multiplicada pola área e dividida pola densidade, $\frac{A}{\rho}\frac{\partial p}{\partial x}$ o sistema anterior convértese en

$$\frac{\partial A}{\partial t} + \frac{\partial q}{\partial x} = 0, \tag{2.8a}$$

$$\frac{\partial q}{\partial t} + \frac{\partial f(q)}{\partial x} + \frac{A}{\rho}\frac{\partial p}{\partial x} - \frac{A}{\rho}\frac{\partial}{\partial x}\left(\varphi\frac{\partial q}{\partial x}\right) - \frac{A}{\rho}\frac{\partial(p-\hat{p})}{\partial x} = -s\frac{q}{A}. \tag{2.8b}$$

Para a súa discretización numérica, o sistema (2.8) será dividido en tres subsistemas seguindo o procedemento mostrado en [30]: un convectivo, un relacionado coa presión e outro relacionado ca viscoelasticidade. Estas dúas últimas fases son resoltas de maneira implícita, mentres que a etapa convectiva é resolta cunha discretización explícita do espazo, o que fai que o sistema sexa semi-implícito. Vexámolo en máis detalle:

Etapa convectiva

O subsistema convectivo, que no caso 1D redúcese a unha ecuación escalar, é

$$\frac{\partial q}{\partial t} + \frac{\partial f(q)}{\partial x} = 0. \tag{2.9}$$

Ao ser o único subsistema resolto de maneira explícita, é o único responsable da restrición de estabilidade CFL no paso do tempo. Se nos fixamos, o autovalor do sistema (2.9) é $\lambda = 2u$, e, polo tanto, a condición CFL depende soamente da velocidade do fluxo. Como consecuencia, as ondas de presión e a viscoelasticidade non limitan o paso temporal, dando lugar a un método altamente eficiente para o estudo do fluxo de sangue [10].

Etapa difusiva

O subsistema difusivo, que de novo está formado por unha soa ecuación, é

$$\frac{\partial q}{\partial t} - \frac{A}{\rho}\frac{\partial}{\partial x}\left(\varphi\frac{\partial q}{\partial x}\right) = -(1-\beta)s\frac{q}{A}, \tag{2.10}$$

onde o parámetro β permítenos decidir se consideramos o termo fonte no subsistema difusivo ou no de presións. Será discretizado implícitamente no tempo.

Etapa de presións

Finalmente o subsistema de presións acaba sendo

$$\frac{\partial A}{\partial t} + \frac{\partial q}{\partial x} = 0, \tag{2.11a}$$

$$\frac{\partial q}{\partial t} + \frac{A}{\rho}\frac{\partial p}{\partial x} - \frac{A}{\rho}\frac{\partial(p-\hat{p})}{\partial x} = -\beta s\frac{q}{A}, \tag{2.11b}$$

que será discretizado utilizando un método implícito de volumes finitos nunha malla encaixada.

Etapa de correcciónS

A presión calculada en (2.11) é utilizada para correxir a velocidade intermedia computada nos pasos convectivo e difusivo, obtendo así un valor máis preciso no novo paso temporal.

2.2. Esquema semi-implícito

Para discretizar o sistema (2.8), primeiro temos que realizar unha semi-discretización na variable temporal que nos leve á división nos tres subsistemas definidos anteriormente. Se discretizamos o sistema (2.8) con respecto á variable temporal chegamos ao seguinte esquema:

$$A(p^{n+1}) = A(p^n) - \Delta t \frac{\partial q^{n+1}}{\partial x} \tag{2.12a}$$

$$q^{n+1} = q^n - \Delta t \frac{\partial f(q^n)}{\partial x} + \Delta t \frac{A(p^n)}{\rho} \frac{\partial}{\partial x}\left(\varphi^n \frac{\partial q^{**}}{\partial x}\right) - \Delta t \frac{A(p^n)}{\rho} \frac{\partial p^{n+1}}{\partial x} + \Delta t \frac{A(p^n)}{\rho} \frac{\partial (p-\hat{p})^n}{\partial x} - \beta \Delta t \frac{s}{A(p^n)} q^{**} - (1-\beta)\Delta t \frac{s}{A(p^n)} q^{n+1} \tag{2.12b}$$

onde p^n, q^n son as solucións coñecidas no tempo t^n, mentres que q^{n+1} e p^{n+1} son as aproximacións das solucións no tempo novo. Utilizamos, ademais, a notación $A(p)$ para denotar que a área pode ser calculada a partires da presión coa lei tubular (2.2).

Denotamos

$$q^* = q^n - \Delta t \frac{\partial f(q^n)}{\partial x} \tag{2.13}$$

como unha primeira aproximación de q, que só ten os termos convectivos. Por outra parte, utilizamos a notación

$$q^{**} = q^* + \Delta t \frac{A(p^n)}{\rho} \frac{\partial}{\partial x}\left(\varphi^n \frac{\partial q^{**}}{\partial x}\right) - (1-\beta)\Delta t \frac{s}{A(p^n)} q^{**} \tag{2.14}$$

para a segunda aproximación de q, que inclúe tamén os termos difusivos.

Con estas notacións, podemos transformar a ecuación (2.12b) nunha relación máis sinxela se as introducimos. Primeiro, reordeamos e introducimos q^*, facendo tamén a substitución $\gamma^n = \frac{s}{A(p^n)}$

$$q^{n+1} = q^* + \Delta t \frac{A(p^n)}{\rho} \frac{\partial}{\partial x}\left(\varphi^n \frac{\partial q^{**}}{\partial x}\right) - (1-\beta)\Delta t \gamma^n q^{**} - \Delta t \frac{A(p^n)}{\rho} \frac{\partial p^{n+1}}{\partial x} + \Delta t \frac{A(p^n)}{\rho} \frac{\partial (p-\hat{p})^n}{\partial x} - \beta \Delta t \gamma^n q^{n+1}. \tag{2.15}$$

Agora, substituímos a variable q^{**}

$$q^{n+1} = q^{**} - \beta \Delta t \gamma^n q^{n+1} - \Delta t \frac{A(p^n)}{\rho} \frac{\partial p^{n+1}}{\partial x} + \Delta t \frac{A(p^n)}{\rho} \frac{\partial (p-\hat{p})^n}{\partial x}, \tag{2.16}$$

e agrupamos os valores de q^{n+1} á esquerda da ecuación

$$q^{n+1}(1+\beta\Delta t\gamma^n) = q^{**} - \Delta t\frac{A(p^n)}{\rho}\frac{\partial p^{n+1}}{\partial x} + \Delta t\frac{A(p^n)}{\rho}\frac{\partial (p-\hat{p})^n}{\partial x}. \tag{2.17}$$

Finalmente, despexamos a variable q^{n+1} e chegamos a un sistema algo máis compacto en comparación con (2.12):

$$A(p^{n+1}) = A(p^n) - \Delta t\frac{\partial q^{n+1}}{\partial x}, \tag{2.18a}$$

$$q^{n+1} = \frac{1}{1+\beta\Delta t\gamma^n}\left(q^{**} - \Delta t\frac{A(p^n)}{\rho}\frac{\partial p^{n+1}}{\partial x} + \Delta t\frac{A(p^n)}{\rho}\frac{\partial (p-\hat{p})^n}{\partial x}\right). \tag{2.18b}$$

Mallas encaixadas

Neste método, na discretización do espazo traballaremos con dúas mallas encaixadas. Para a primeira delas, o vaso sanguíneo xenérico de lonxitude L é dividido en N_C intervalos de lonxitude Δx, que neste caso será uniforme (aínda que se podería adaptar a un caso de non uniformidade). Cada cela C_i ten dous extremos, un á esquerda, con índice $l(i) = i - \frac{1}{2}$, e outro á dereita, con índice $r(i) = i + \frac{1}{2}$.

Pola outra parte, as celas da malla dual teñen como extremos os puntos medios das celas da malla orixinal. Máis explícitamente, os baricentros x_i, x_{i+1} de dúas celas veciñas C_i, C_{i+1} da malla primitiva están asociados cunha cela da malla dual de intervalo $D_{i+\frac{1}{2}} = [x_i, x_{i+1}]$, e a súa lonxitude coincide con Δx. As celas do borde son distintas, xa que van dende o extremo do dominio ata o punto máis próximo, polo que a súa lonxitude sería $\frac{\Delta x}{2}$. Os centros das celas $D_{i+\frac{1}{2}}$ serán denotados como $x_{i+\frac{1}{2}}$. Polo tanto, a malla dual ten $N_D = N_C + 1$ celas. Con respecto ás variables conservativas, tanto a variable de presión p como a da área da sección A serán computadas na malla primitiva, mentres que as variables da velocidade u e do momento q están asociadas á malla dual. Na Figura 2.1 móstrase unha representación gráfica deste encaixamento, onde pode apreciarse con maior claridade.

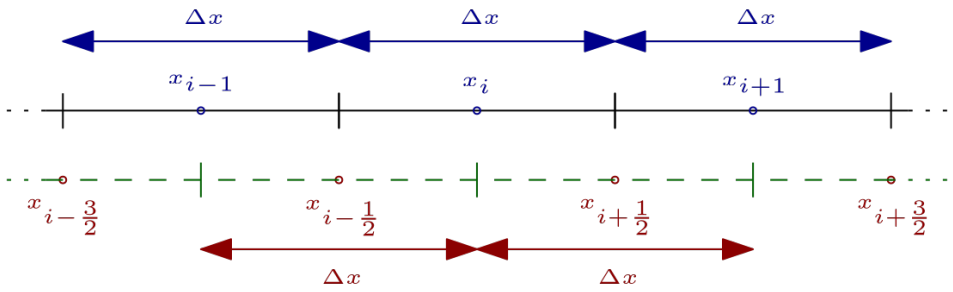

Figura 2.1: Visualización das mallas encaixadas para as celas primal C_i, as duais asociadas $D_{i-\frac{1}{2}}$, $D_{i+\frac{1}{2}}$ e as contiguas

Etapa convectiva

A etapa convectiva é a única que é explícita no esquema semi-implícito e é a que nos introduce a limitación CFL do paso do tempo. Se integramos (2.13) ao longo dunha cela dual, temos

$$q^*_{i+\frac{1}{2}} = q^n_{i+\frac{1}{2}} - \frac{\Delta t}{\Delta x}(f^n_{i+1} - f^n_i). \tag{2.19}$$

Os termos dos fluxos nos bordos das celas, f^n_{i+1} e f^n_i aproxímanse usando unha función de fluxo numérico de Ducros, ϕ^D,

$$\begin{aligned} f^n_i = \phi^D(q^n_{i-\frac{1}{2}}, q^n_{i+\frac{1}{2}}, u^n_{i-\frac{1}{2}}, u^n_{i+\frac{1}{2}}) = \frac{1}{2}(u^n_{i+\frac{1}{2}} + u^n_{i-\frac{1}{2}})\frac{1}{2}(q^n_{i+\frac{1}{2}} + q^n_{i-\frac{1}{2}}) - \frac{1}{2}\alpha^n_i(q^n_{i+\frac{1}{2}} - q^n_{i-\frac{1}{2}}), \\ \alpha^n_i = 2\,\text{máx}(|u^n_{i-\frac{1}{2}})|, |u^n_{i+\frac{1}{2}}|). \end{aligned} \tag{2.20}$$

O esquema anterior é de primeira orde en espazo e usando o análise de estabilidade de von Neumann pódese mostrar que é condicionalmente estable baixo a condición CFL,

$$\Delta t \leq \text{CFL} \min_{j\in\{1,\ldots,N_D\}} \left(\frac{\Delta x}{2|u^n_{j-\frac{1}{2}}|} \right), \tag{2.21}$$

con CFL ≤ 1. Para aumentar a precisión, pódese introducir una metodoloxía tipo Kolgan. Este enfoque baséase na idea de substituir as variables conservativas por interpolacións feitas con polinomios. Neste suposto, tense que verificar que CFL ≤ 0.5.

Etapa difusiva

No momento en que aproximamos q^* na etapa convectiva, resolvemos a etapa difusiva usando un método implícito de volumes finitos. Neste caso, faise unha discretización do espazo baseada no esquema *backwards in time centered in space* (BTCS).

Aplicando este método á variable q^{**} introducida en (2.14) obtemos a seguinte ecuación, en notación matriz-vector:

$$\mathbf{D}\mathbf{q}^{**} = \mathbf{v}, \tag{2.22}$$

onde $\mathbf{D}$ é unha matriz tridigonal con valores:

$$\begin{aligned} a_i &= -\frac{\Delta t}{\rho}\frac{\varphi^n_-}{\Delta x}, \\ b_i &= \frac{\Delta x}{A^n_{i+\frac{1}{2}}}(1 + (1-\beta)\Delta t\gamma^n_{i+\frac{1}{2}}) + \frac{\Delta t}{\rho}\left(\frac{\varphi^n_+}{\Delta x} + \frac{\varphi^n_-}{\Delta x}\right), \\ c_i &= -\frac{\Delta t}{\rho}\frac{\varphi^n_+}{\Delta x}, \end{aligned}$$

onde

$$A^n_{i+\frac{1}{2}} = \frac{A(p^n_i) + A(p^n_{i+1})}{2},$$

e

$$\varphi_-^n = \frac{\varphi(A^n_{i+\frac{1}{2}}) + \varphi(A^n_{i-\frac{1}{2}})}{2}, \quad \varphi_+^n = \frac{\varphi(A^n_{i+\frac{3}{2}}) + \varphi(A^n_{i+\frac{1}{2}})}{2}.$$

$\mathbf{q}^*$∗ é o vector que contén a segunda aproximación da variable q en cada cela dual e no tempo t^{n+1}; e v é o vector que contén a primeira aproximación q^*, xa sabida. En resumo, para cada aproximación $q^{**}_{i+\frac{1}{2}}$, estamos tendo en conta o valor de q^{**} na cela $D_{i+\frac{1}{2}}$ e nas celas da esquerda e da dereita, ademáis do valor de $q^*_{i+\frac{1}{2}}$, que coñecemos.

Como a matriz D é simétrica definida positiva, o sistema (2.22) é resolto usando o método do gradiente conxugado. Polo tanto, o esquema para o subsistema difusivo é incondicionalmente estable, de primeira orde no tempo e de segunda orde no espazo

Etapa de presións

Despois de facer unha aproximación de q^{**}, podemos facer a resolución do subsistema de presións. Se o discretizamos implícitamente no tempo, chegamos a un esquema que podemos escribir de maneira compacta con notación matriz-vector:

$$\mathbf{M}(\mathbf{p}^{n+1}) + \mathbf{T}\mathbf{p}^{n+1} = \mathbf{b}^n, \tag{2.23}$$

onde $\mathbf{M} = \text{diag}(M(p_i^{n+1}))$ é unha función vectorial que contén os termos non lineais, sendo $M(p_i) = A(p_i)\Delta x$. $\mathbf{T}$ é unha matriz tridiagonal simétrica definida positiva, con entradas

$$\begin{aligned}
a_i &= -\frac{\Delta t^2}{\Delta x}\frac{A^n_{i-\frac{1}{2}}}{\rho(1+\beta\Delta t\gamma^n_{i-\frac{1}{2}})},\\
b_i &= \frac{\Delta t^2}{\Delta x}\frac{A^n_{i+\frac{1}{2}}}{\rho(1+\beta\Delta t\gamma^n_{i+\frac{1}{2}})} + \frac{\Delta t^2}{\Delta x}\frac{A^n_{i-\frac{1}{2}}}{\rho(1+\beta\Delta t\gamma^n_{i-\frac{1}{2}})},\\
c_i &= -\frac{\Delta t^2}{\Delta x}\frac{A^n_{i+\frac{1}{2}}}{\rho(1+\beta\Delta t\gamma^n_{i+\frac{1}{2}})}.
\end{aligned}$$

$\mathbf{p}^{n+1}$ é o vector da nova presión na malla primitiva e $\mathbf{b^n}$ é o vector de valores coñecidos, onde cada elemento é

$$b_i^n = M(p_i^n) - \Delta t\left(\frac{q^{**}_{i+\frac{1}{2}}}{1+\beta\Delta t\gamma^n_{i+\frac{1}{2}}} - \frac{q^{**}_{i-\frac{1}{2}}}{1+\beta\Delta t\gamma^n_{i-\frac{1}{2}}}\right) -$$

$$-\Delta t^2\left[\frac{1}{\Delta x}\frac{A^n_{i+\frac{1}{2}}}{\rho(1+\beta\Delta t\gamma^n_{i+\frac{1}{2}})}((p-\hat{p})^n_{i+1} - (p-\hat{p})^n_i) - \frac{1}{\Delta x}\ \frac{A^n_{i-\frac{1}{2}}}{\rho(1+\beta\Delta t\gamma^n_{i-\frac{1}{2}})}((p-\hat{p})^n_i - (p-\hat{p})^n_{i-1})\right].$$

O tipo de non linealidade de $\mathbf{M}(\mathbf{p})$ non permite resolver o sistema (2.23) a través dun método de Newton típico. Sen embargo, podemos utilizar un algoritmo anidado tipo Newton, proposto

en [31], que combinado co método do gradiente conxugado demostra ser eficiente para resolver o sistema non lineal. A metodoloxía deste esquema consiste en separar os termos non lineais para despois linealizalos en secuencia e xerar un método iterativo anidado.

O θ-*method*

Para aumentar a precisión temporal do método semi-implícito pódese utilizar o θ-*method* [22], e así contrarrestar a principal desventaxa que supón separar a ecuación do sangue en etapas.

Propiedade C

É importante destacar que o método ten a propiedade de estar ben equilibrado, [32].

Proposición 2.1. *(Propiedade C) O modelo anteriormente descrito está ben equilibrado; isto é, dados* $q^n = 0$ *e* $p^n = cte.$*, entón* $p^{n+1} = p^n$ *e* $q^{n+1} = q^n = 0$.

Demostración. A proba pode ser achada en [10]. □

Condicións de contorno

Nos resultados numéricos para o estudo do fluxo de sangue, presentados no Capítulo 4, empregaremos condicións de contorno de tipo Dirichlet fortes na variable q do fluxo de masa, o que quere dicir que imporemos nos extremos da malla dual o valor correspondente en cada caso, e na malla primal os valores de A.

Capítulo 3

Método implícito para a resolución do fluxo de sangue nun conducto

A nova metodoloxía que propoñemos neste capítulo encargase de transformar o método anterior nun método completamente implícito. Tal e como foi exposto, a única parte explícita da metodoloxía anterior é a que está relacionada coa etapa convectiva da ecuación do sangue. O procedemento consistía en resolver a ecuación (2.9) a través dun método explícito de volumes finitos baseado na función de fluxo de Ducros. Máis concretamente, estabamos definindo o valor de q^* mediante (2.13) e resolvendo para esa variable.

Agora, seguindo o feito para as ecuacións de Navier-Stokes en 3D en [33], imos realizar a resolución desta etapa do método a través dunha discretización implícita do espazo, e utilizando un método Newton inexacto en combinación cun método de subespazos de Krylov (neste caso, o BiCGSTAB). De forma máis específica, o que imos facer é definir q^* implícitamente como

$$q^* = q^n - \Delta t \frac{\partial f(q^*)}{\partial x}, \tag{3.1}$$

e utilizar o novo método para achar esta variable.

Queremos resolver a ecuación (3.1), un problema que podemos reinterpretar como crear unha función h definida como

$$h(q^*) = q^* - q^n + \Delta t \; \partial_x f(q^*) \tag{3.2}$$

e intentar dar co valor de q^* que faga $h(q^*) = 0$. Imos formular un **método de Newton inexacto** no que, comezando con $q_0^* = q^n$ como estimación inicial, resolvemos a ecuación

$$J(q_k^*)\Delta q_k^* = -h(q_k^*) \tag{3.3}$$

para Δq_k^*, que representa o paso de Newton, e despois actualizamos a nosa estimación con

$$\hat{q}_{k+1}^* = q_k^* + \delta_k \Delta q_k^*. \tag{3.4}$$

$J(q_k^*) = \partial_{q_k^*} h(q_k^*)$ é a derivada parcial da nosa función h con respecto á variable q_k^* e, inicialmente, $\delta_k = 1$. A parte inexacta deste Newton xorde da existencia deste parámetro, que modificaremos segundo o residuo da función h en relación á iteración anterior.

Facemos a comprobación

$$\|h(\hat{q}_{k+1}^*)\| \leq \|h(q_k^*)\|; \tag{3.5}$$

se non se cumpre, tomamos $\widetilde{\delta}_k = \frac{\delta_k}{2}$ e volvemos a calcular $\hat{q}_{k+1}^*$ con

$$\hat{q}_{k+1}^* = q_k^* + \widetilde{\delta}_k \Delta q_k^* \tag{3.6}$$

e repetimos dende o test 3.5. Cando achemos un $\widetilde{\delta}_k$ que o verifique, tomamos $q_{k+1}^* := \hat{q}_{k+1}^*$.

Se ε_N é a tolerancia que pedimos ao Newton, seguimos iterando ata atopar un k' que verifique $\|h(q_{k'}^*)\| < \varepsilon_N$. Nese momento, paramos o método e tomamos $q^* := q_{k'}^*$.

Este procedemento, que parece sinxelo, complícase cando temos en conta que precisamos resolver a ecuación (3.3) de maneira implícita para achar o valor de Δq_k^*. Sen embargo, podemos realizar unha linealización do noso problema e interpretar (3.3) como un sistema lineal do tipo $Ax = b$, que podemos resolver con numerosos algoritmos. Neste traballo, solucionaremos ese sistema lineal empregando o **método do gradiente biconxugado estabilizado** (a partires de agora, BiCGSTAB, polas súas siglas en inglés).

3.1. Cálculo do residuo

O lado dereito da ecuación (3.3) non é máis que calcular o residuo dado pola nosa función h para un q_k^*,

$$h(q_k^*) = q_k^* - q^n + \Delta t \; \partial_x f(q_k^*). \tag{3.7}$$

Para simplificar a presentación do método, despreciaremos o índice k describindo o procedemento para un q^* xeral.

Imos facer a identificación:

$$\delta q^{**} := \Delta t \partial_x f(q^*). \tag{3.8}$$

Se integramos este valor ao longo dunha cela dual $D_{i+\frac{1}{2}}$ e aplicamos Gauss, temos

$$\delta q_{i+\frac{1}{2}}^{**} = \frac{\Delta t}{\Delta x} \left(f_{i+1}^* - f_i^* \right), \tag{3.9}$$

como xa vimos en (1.31), so que neste caso temos un cambio de índices, xa que calculamos a variable conservativa q na cela dual.

Cada un destes fluxos pode ser computado, por exemplo, empregando unha función de fluxo numérico de Rusanov [25], ϕ^R, da forma

$$\begin{aligned} f_i^* &= \phi^R(q^*_{i-\frac{1}{2}}, q^*_{i+\frac{1}{2}}, u^n_{i-\frac{1}{2}}, u^n_{i+\frac{1}{2}}) = \\ &= \frac{1}{2}\left(f(q^*_{i-\frac{1}{2}}) + f(q^*_{i+\frac{1}{2}})\right) - \frac{1}{2}\alpha^R_{ij}(q^*_{i+\frac{1}{2}} - q^*_{i-\frac{1}{2}}), \\ u^n_{ij} &= \frac{1}{2}(u^n_{i-\frac{1}{2}} + u^n_{i+\frac{1}{2}}), \quad \alpha^R_{ij} = 2|u^n_{ij}|. \end{aligned} \tag{3.10}$$

Recordemos que a función do fluxo do sangue é $f(q) = \frac{q^2}{A}$, polo que a súa derivada con respecto ao momento é $\frac{\partial f(q)}{\partial q} = 2\frac{q}{A} = 2u$. A función de fluxo de Rusanov toma como α o máximo dos valores absolutos das derivadas nas celas, pero nós neste caso calculamos a media ponderada das derivadas e ese é o valor co que calculamos o α. Facemos esta diferenciación porque consegue que teñamos menos viscosidade numérica, aínda que sacrifiquemos un pouco de precisión.

Podemos, tamén, utilizar outras funcións de fluxo numérico, como a de Ducros [34]:

$$\begin{aligned} f_i^* &= \phi^D(q^*_{i-\frac{1}{2}}, q^*_{i+\frac{1}{2}}, u^n_{i-\frac{1}{2}}, u^n_{i+\frac{1}{2}}) = \\ &= \frac{1}{2}(q^*_{i-\frac{1}{2}} + q^*_{i+\frac{1}{2}})u^n_{ij} - \frac{1}{2}\alpha^D_{ij}(q^*_{i+\frac{1}{2}} - q^*_{i-\frac{1}{2}}), \\ u^n_{ij} &= \frac{1}{2}(u^n_i + u^n_j), \quad \alpha^D_{ij} = 2|u^n_{ij}|, \end{aligned} \tag{3.11}$$

que é lineal en q^*, polo que os coste computacional redúcese.

3.2. Cálculo do produto matriz-vector

O lado esquerdo da ecuación (3.3), que nos serve de base para calcular o paso de Newton, consiste nun produto matriz-vector onde $J(q^*) = \frac{\partial h(q^*)}{\partial q^*}$ é a matriz jacobiana da función h con respecto a q^* e onde Δq^* é o vector do paso de Newton. Como queremos resolver o sistema a través dun método de resolución de sistemas lineais non-simétricos (BiCGSTAB), temos que realizar unha linearización deste produto matriz-vector. Nótese que en ningún momento construimos a Jacobiana directamente, senon que utilizamos unha modificación do algoritmo que traballa sen a necesidade dunha matriz A explícita.

A función de fluxo de Ducros xa é lineal, polo que o produto matriz-vector simplemente é

$$\begin{aligned} &\frac{\partial \phi^D(\Delta q^*_{i-\frac{1}{2}}, \Delta q^*_{i+\frac{1}{2}}, u^n_{i-\frac{1}{2}}, u^n_{i+\frac{1}{2}})}{\partial(q^*_{i-\frac{1}{2}}, q^*_{i+\frac{1}{2}})} \cdot (\Delta q^*_{i-\frac{1}{2}}, \Delta q^*_{i+\frac{1}{2}}) = \\ &= \frac{1}{2}u^n_{ij}(\Delta q^*_{i-\frac{1}{2}} + \Delta q^*_{i+\frac{1}{2}}) - \frac{1}{2}\alpha^D_{ij}(\Delta q^*_{i+\frac{1}{2}} - \Delta q^*_{i-\frac{1}{2}}). \end{aligned} \tag{3.12}$$

Por outro lado, a linearización da función de fluxo de Rusanov na dirección normal, que é

non lineal en q^*, sería

$$\begin{aligned} &\frac{\partial \phi^R(\Delta q^*_{i-\frac{1}{2}}, \Delta q^*_{i+\frac{1}{2}}, u^n_{i-\frac{1}{2}}, u^n_{i+\frac{1}{2}})}{\partial (q^*_{i-\frac{1}{2}}, q^*_{i+\frac{1}{2}})} \cdot (\Delta q^*_{i-\frac{1}{2}}, \Delta q^*_{i+\frac{1}{2}}) = \\ &= (q^*_{i-\frac{1}{2}} \Delta q^*_{i-\frac{1}{2}} + q^*_{i+\frac{1}{2}} \Delta q^*_{i+\frac{1}{2}}) - \frac{1}{2} \alpha^R_{ij} (\Delta q^*_{i+\frac{1}{2}} - \Delta q^*_{i-\frac{1}{2}}), \end{aligned} \tag{3.13}$$

e onde $q^*_{i-\frac{1}{2}}$ e $q^*_{i+\frac{1}{2}}$ denotan o momento nas celas $D_{i-\frac{1}{2}}$ e $D_{i+\frac{1}{2}}$ da iteración actual de Newton e $\Delta q^*_{i-\frac{1}{2}}$, $\Delta q^*_{i+\frac{1}{2}}$ os pasos de Newton asociados á ecuación (3.3).

Con estes dous cálculos, o do residuo e o do produto matriz-vector, temos todo o que precisabamos para calcular o paso dentro do Newton inexacto, polo que xa podemos aplicar o método completo a un problema de fluxo real.

3.2.1. BiCGSTAB

O método de subespazos de Krylov máis sinxelo é o método do gradiente conxugado (CG, polas siglas en inglés), desenvolvido en [35]. É un método de proxección ortogonal que só é consistente en sistemas con matrices simétricas definidas positivas ou Hermitianas definidas positivas. Se comezamos cun sistema $\mathbf{Ax} = \mathbf{b}$, onde a solución vén dada por $\mathbf{x}^*$, é interesante observar que $\mathbf{x}^*$ tamén minimiza a función cuadrática

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{Ax} - \mathbf{x}^T\mathbf{b}, \tag{3.14}$$

xa que, por imposición, a súa matriz Hessiana é simétrica definida positiva

$$\mathbf{H}(f(\mathbf{x})) = \mathbf{A}, \tag{3.15}$$

e a solución $\mathbf{x}^*$ tamén é raíz do seu gradiente,

$$\nabla f(\mathbf{x}^*) = \mathbf{Ax} - \mathbf{b}. \tag{3.16}$$

O CG utiliza estes feitos para mover a solución en dirección do gradiente negativo, achegándose cada vez máis á solución do gradiente da función e, polo tanto, á solución do sistema.

Resolver sistemas lineales non simétricos con métodos de subespazos de Krylov é considerablemente máis difícil e costoso a nivel computacional, pero cos anos e cos avances computacionais foron xurdindo métodos capaces de enfrontarse a estes problemas, como o CGS (*conjugate gradient squared*) [36]. Tamén foi creado o BiCG (*biconjugate gradient*), pero é un método numéricamente inestable. Van Der Vorst [37] propuxo o BiCGSTAB como unha variante do anterior, que ten unha converxencia máis rápida e consistente e non require de transpoñer a matriz do sistema. Añadimos o pseudocódigo do método no Algoritmo 1, presente en [37].

Algoritmo 1 BiCGSTAB sen precondicionamento

x_0 é a estimación inicial; $r_0 = b - Ax_0$;
$\hat{r}_0$ é un vector arbitrario tal que $(\hat{r}_0, r_0) \neq 0$, por exemplo, $\hat{r}_0 = r_0$;
$\rho_0 = \alpha = \omega_0 = 1$;
$v_0 = p_0 = 0$;
for $i = 1, 2, 3, \ldots$, **do**
  $\rho_i = (\hat{r}_0, r_{i-1})$;$\beta = (\rho_i/\rho_{i-1})(\alpha/\omega_{i-1})$;
  $p_i = r_{i-1} + \beta(p_{i-1} - \omega_{i-1}v_{i-1}$;
  $v_i = Ap_i$;
  $\alpha = \rho_i/(\hat{r}_o, v_i)$;
  $s = r_{i-1} - \alpha v_i$;
  $t = As$;
  $\omega_i = (t, s)/(t, t)$;
  $x_i = x_{i-1} + \alpha p_i + \omega_i s$;
  if $x_i < \varepsilon_B$ **then**
    return x_i
  end if
  $r_i = s - \omega_i t$
end for

Como na ecuación do cálculo do paso (3.3) non temos a matriz A (que neste caso sería $J(q_k^*)$), facemos unha modificación do algoritmo onde creamos unha función que, dado un vector v, realiza o produto matriz-vector mostrado en (3.12) ou (3.13). Os códigos de MatLab destes dous algoritmos están presentes no Apéndice I.

Capítulo 4

Resultados numéricos

Aplicamos agora o método implícito á ecuación de transporte (1.8) e á ecuación de Burgers (1.23), a través das cales faremos un estudo numérico da orde do método. Despois, introduciremos os resultados de aplicar o método implícito de volumes finitos ao problema do fluxo de sangue, e compararemos co método semi-implícito para ver se acadamos os obxectivos buscados: eliminar a condición CFL e conseguir maior eficiencia computacional sen sacrificar a precisión.

4.1. Aplicación do método implícito de volumes finitos á ecuación de transporte

É interesante observar que na sección anterior o que fixemos foi definir o método para unha ecuación calquera, non necesariamente a do sange. De feito, podemos ter unha ecuación máis sinxela, como a de ecuación de transporte lineal escalar:

$$\frac{\partial q}{\partial t} + \frac{\partial f(q)}{\partial x} = \frac{\partial q}{\partial t} + \lambda \frac{\partial q}{\partial x} = 0. \tag{4.1}$$

O procedemento de resolución a través do algoritmo numérico cunha función de fluxo diferente á presentada no caso de fluxos de sangue é prácticamente análogo, exceptuando que neste caso, en vez de traballar con q^*, calculamos directamente o valor de q^{n+1}. É dicir, a ecuación que resolvemos implícitamente é

$$q^{n+1} = q^n - \lambda \Delta t \frac{\partial q^{n+1}}{\partial x}. \tag{4.2}$$

Observemos, ademais, que xa non precisamos dunha discretización dual do espazo, polo que simplificamos a malla a unha formada por N_C celas C_i da mesma lonxitude Δx.

Os fluxos numéricos, neste caso, serán

$$\begin{aligned} \phi^R(q_i^{n+1}, q_j^{n+1}) &= \frac{1}{2}(f(q_i^{n+1}) + f(q_j^{n+1})) - \frac{1}{2}\alpha_{ij}^R(q_j^{n+1} - q_i^{n+1}), \\ \alpha_{ij}^R &= |\lambda|, \end{aligned} \tag{4.3}$$

e

$$\begin{aligned} \phi^D(q_i^{n+1}, q_j^{n+1}) &= \lambda\frac{1}{2}(q_i^{n+1} + q_j^{n+1}) - \frac{1}{2}\alpha_{ij}^D(q_j^{n+1} - q_i^{n+1}), \\ \alpha_{ij}^D &= |\lambda|; \end{aligned} \tag{4.4}$$

fixémonos que tanto Rusanov como Ducros coinciden, e como estamos ca ecuación de transporte (4.1) tamén son lineais. Polo tanto, o cálculo do produto matriz-vector será trivial neste caso:

$$\frac{\partial\phi(\Delta q_i^{n+1}, \Delta q_j^{n+1})}{\partial(q_i^{n+1}, q_j^{n+1})} \cdot (\Delta q_i^{n+1}, \Delta q_j^{n+1}) = \lambda\frac{1}{2}(\Delta q_i^{n+1} + \Delta q_j^{n+1}) - \frac{1}{2}\alpha_{ij}(\Delta q_j^{n+1} - \Delta q_i^{n+1}). \tag{4.5}$$

4.1.1. Estudo de converxencia do método implícito aplicado á ecuación de transporte

Nesta sección imos utilizar a mencionada ecuación de transporte para verificar que o método é de primeira orde de converxencia na variable espacial. Unha forma de facer isto é comparar a solución calculada numéricamente a través do algoritmo con calquera test para o que coñezamos a solución exacta.

Imos marcar unha condición inicial suave, neste caso $q_0(x) = \cos(\pi x)$ no intervalo $[-1, 1]$, con $\lambda = 0.5$ e condicións de contorno periódicas. A solución exacta para este test é $q(x,t) = \cos(\pi(x - \lambda t))$. Compararemos esta solución coa obtida polo noso método de Newton inexacto con cálculo do paso a través dun BiCGSTAB modificado.

A orde determínase mirando o comportamento do erro $\varepsilon_{\Delta x}$ nas normas $\|\cdot\|_1$ e $\|\cdot\|_2$. O erro é calculado no tempo $t_f = 1$s nunha malla espaciada uniformemente e con $\Delta x = \frac{2}{N_C}$. Os erros para cada norma son definidos como

$$\varepsilon_{\Delta x}^1(t_f) = \Delta x \sum_{i=1}^{N_C} \| \; q(x_i, t_f) - q_i^n \; \|, \tag{4.6}$$

e

$$\varepsilon_{\Delta x}^2(t_f) = \sqrt{\Delta x \sum_{i=1}^{N_C} (\| \; q(x_i, t_f) - q_i^n \; \|)^2}. \tag{4.7}$$

Dicimos que un método é de orde p se existe una constante k que cumprindo

$$\varepsilon_{\Delta x_k}(t_f) \leq k(\Delta x_k)^p \tag{4.8}$$

verifique, despois de refinar a malla, que

$$\varepsilon_{\Delta x_{k+1}}(t_f) \leq k(\Delta x_{k+1})^p, \tag{4.9}$$

para calquera malla $\mathcal{M}_{\Delta x_k}$.

Os mallados escollidos (como se pode ver no cadro 4.1) verifican que $\frac{\Delta x_k}{\Delta x_{k+1}} = 2$. Polo tanto, un esquema de orde p satisfai a ecuación

$$\Upsilon_{k+1} := \frac{\varepsilon_{\Delta x_k}(t_f)}{\varepsilon_{\Delta x_{k+1}}(t_f)} \leq \frac{(\Delta x_k)^p}{(\Delta x_{k+1})^p} = 2^p \tag{4.10}$$

que, aplicando logaritmos, danos a relación

$$\log(\Upsilon_{k+1}) \leq p \log(2). \tag{4.11}$$

Entón, en termos de Υ_k, para dúas mallas $\mathcal{M}_{\Delta x_k}$, $\mathcal{M}_{\Delta x_{k+1}}$, a orde pode ser estimada como

$$p_{k+1} := \log_2(\Upsilon_{k+1}). \tag{4.12}$$

O cadro 4.1 mostra os erros do método con este problema para diferentes mallas e a estimación da orde do método para cada unha delas. Como podemos ver, este procedemento danos como resultado unha converxencia de primeira orde. Por outra banda, a figura 4.1 mostra a estimación numérica deste problema, a solución exacta e o erro entre as dúas para $N_C = 100$. Os códigos cos que se obtiveron estes resultados inclúense no Apéndice I.1.

N_C	Δx_k	Erro en $\|\cdot\|_1$	p_k	Erro en $\|\cdot\|_2$	p_k
50	0,04	$1.4738 \cdot 10^{-1}$	———	$1.1588 \cdot 10^{-1}$	———
100	0,02	$7.6111 \cdot 10^{-2}$	0,95	$5.9796 \cdot 10^{-2}$	0,96
200	0,01	$3.8663 \cdot 10^{-2}$	0,98	$3.0368 \cdot 10^{-2}$	0,98
400	0,005	$1.9483 \cdot 10^{-2}$	0,99	$1.5303 \cdot 10^{-2}$	0,99
800	0,0025	$9.7796 \cdot 10^{-3}$	0,99	$7.6809 \cdot 10^{-3}$	0,99

Cadro 4.1: Erros e orde do método implícito de volumes finitos para a ecuación de transporte con condición inicial $q_0(x) = \cos(\pi x)$.

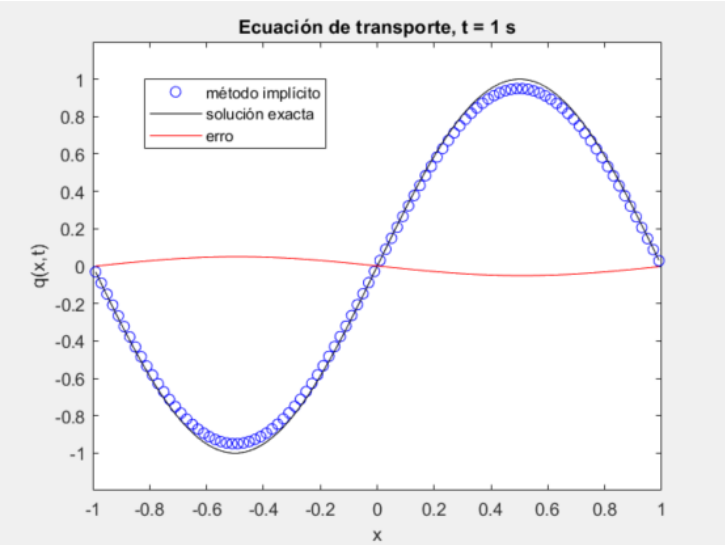

Figura 4.1: Solución numérica obtida con método de volumes finitos implícito para a solución da ecuación de transporte lineal escalar con condición inicial $q_0(x)\cos(\pi x)$ para $t = 1$s. Solución exacta e erro.

4.2. Aplicación do método implícito á ecuación de Burgers

Da mesma maneira que fixemos na sección anterior, aplicamos o método á ecuación de Burgers, que recordemos que é:

$$\frac{\partial q}{\partial t} + \frac{\partial f(q)}{\partial x} = \frac{\partial q}{\partial t} + \frac{\partial}{\partial x}\left(\frac{q^2}{2}\right) = 0; \tag{4.13}$$

polo tanto, neste caso temos que discretizar de maneira implícita a ecuación:

$$q^{n+1} = q^n - \Delta t \frac{\partial}{\partial x}\left(\frac{(q^{n+1})^2}{2}\right). \tag{4.14}$$

Os fluxos numéricos neste caso serán

$$\begin{gathered}\phi^R(q_i^{n+1}, q_j^{n+1}) = \frac{1}{2}(f(q_i^{n+1}) + f(q_j^{n+1})) - \frac{1}{2}\alpha_{ij}^R(q_j^{n+1} - q_i^{n+1}), \\ \alpha_{ij}^R = \text{máx}(|q_i^n|, |q_j^n|),\end{gathered} \tag{4.15}$$

e

$$\begin{gathered}\phi^D(q_i^{n+1}, q_j^{n+1}) = \frac{1}{4}(q_i^{n+1} + q_j^{n+1})q_{ij}^n - \frac{1}{2}\alpha_{ij}^D(q_j^{n+1} - q_i^{n+1}), \\ q_{ij}^n = \frac{1}{2}(q_i^n + q_j^n), \quad \alpha_{ij}^D = \text{máx}(|q_i^n|, |q_j^n|).\end{gathered} \tag{4.16}$$

No que respecta ao producto matriz-vector, a linealización do fluxo de Ducros sería:

$$\frac{\partial \phi^D(\Delta q_i^{n+1}, \Delta q_j^{n+1})}{\partial (q_i^{n+1}, q_j^{n+1})} \cdot (\Delta q_i^{n+1}, \Delta q_j^{n+1}) = \frac{1}{4} q_{ij}^n (\Delta q_i^{n+1} + \Delta q_j^{n+1}) - \frac{1}{2} \alpha_{ij}^D (\Delta q_j^{n+1} - \Delta q_i^{n+1}); \quad (4.17)$$

mentres que a de Rusanov é:

$$\frac{\partial \phi^R(\Delta q_i^{n+1}, \Delta q_j^{n+1})}{\partial (q_i^{n+1}, q_j^{n+1})} \cdot (\Delta q_i^{n+1}, \Delta q_j^{n+1}) = \frac{1}{2} (q_i^{n+1} \Delta q_i^{n+1} + q_j^{n+1} \Delta q_j^{n+1}) - \frac{1}{2} \alpha_{ij}^R (\Delta q_j^{n+1} - \Delta q_i^{n+1}), \quad (4.18)$$

onde q_i^{n+1}, q_j^{n+1} son os valores da variable conservativa nas celas C_i e C_j da iteración actual de Newton e Δq_i^{n+1}, Δq_j^{n+1} os pasos de Newton asociados á ecuación (3.3).

Fixémonos que estas funcións numéricas e as súas linealizacións recordan ás que tiñamos para a fase convectiva da ecuación do sangue ((3.10), (3.11), (3.13) e (3.12)). Isto é debido a que as dúas funcións de fluxo son cuadráticas e comparten similitudes á hora de ser modelizadas.

4.2.1. Estudo de converxencia do método implícito aplicado á ecuación de Burgers

Realicemos agora unha segunda verificación de que o método co que estamos a traballar é de primeira orde de converxencia na variable espacial. Neste caso, a condición inicial suave será $q_0(x) = e^{-(10x)^2}$ no intervalo $[-0.8, 0.8]$ e con condicións periódicas de contorno. A solución será calculada cun método explícito de volumes finitos, e este resultado numérico servirá de base para a comparación coa solución obtida a través do novo método implícito.

A orde calcúlase mirando o comportamento dos erros $\varepsilon_{\Delta x}$ (4.6), (4.7), da mesma maneira que na Sección 4.1.1. Os erros son calculados no tempo $t_f = 0.5s$ nunha malla espaciada uniformemente e con $\Delta x = \frac{1.6}{N_C}$. De novo, as mallas escollidas no cadro 4.2 verifican $\frac{\Delta x_k}{\Delta x_{k+1}} = 2$.

O cadro 4.2 amosa os erros do método para este problema con diferentes mallados e a estimación da orde do método para cada un deles. De novo, temos como resultado unha converxencia de primeira orde. Tamén engadimos a figura 4.2, que mostra o cálculo numérico da variable conservativa polo método explícito e polo método implícito, para $N_C = 320$. Os algoritmos para obter estes resultados pódense atopar no Apéndice I.2.

N_C	Δx_k	Erro en $\|\cdot\|_1$	p_k	Erro en $\|\cdot\|_2$	p_k
80	0,02	$1.3641 \cdot 10^{-2}$	———	$4.5090 \cdot 10^{-3}$	———
160	0,01	$8.2969 \cdot 10^{-3}$	0,72	$2.5023 \cdot 10^{-3}$	0,85
320	0,005	$4.7430 \cdot 10^{-3}$	0,81	$1.3567 \cdot 10^{-3}$	0,88
640	0,0025	$2.5142 \cdot 10^{-3}$	0,92	$7.1652 \cdot 10^{-4}$	0,92
1280	0,00125	$1.3002 \cdot 10^{-3}$	0,95	$3.6484 \cdot 10^{-4}$	0,97

Cadro 4.2: Erros e orde do método implícito de volumes finitos para a ecuación de Burgers con condición inicial $q_0(x) = e^{-(10x)^2}$.

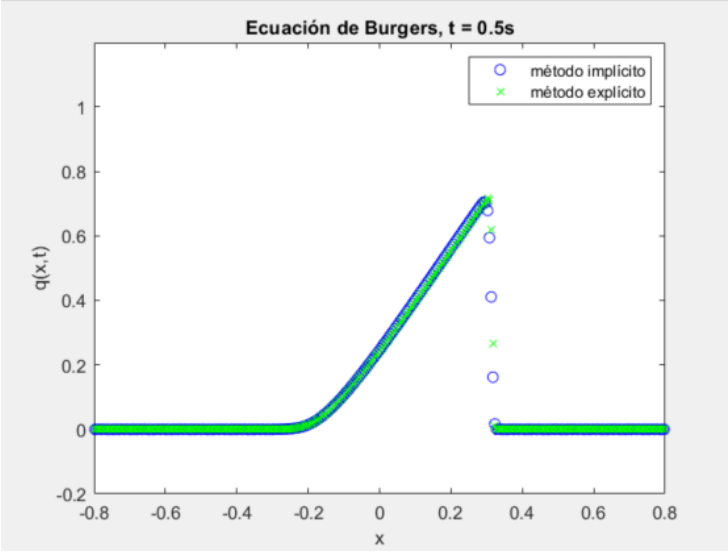

Figura 4.2: Solución numérica obtida empregando o método implícito para a resolución da ecuación de Burgers con condición inicial $q_0(x) = e^{-(10x)^2}$ para $t = 0.5$s. Comparación coa solución obtida cun método explícito.

4.3. Comparación entre os métodos de resolución do fluxo de sangue

O método de volumes finitos completamente implícito para a resolución de problemas de fluxo de sangue en vasos sanguíneos foi desenvolvido para intentar superar as limitacións que tiña o método semi-implícito, e intentar dar cun esquema máis estable e eficiente. Nesta sección imos analizar se estos obxectivos foron alcanzados e se as prediccióǹs que tiñamos son visibles a nivel computacional.

Antes de nada, recordamos o feito de que método semi-implícito presentaba unha limitación do paso temporal, como xa foi exposto no Capítulo 2. Esta restricción CFL, (2.21), impedíanos tomar pasos de tempo demasiado grandes, xa que se poñía en risco a estabilidade do esquema. No código de Matlab, a variable `dtmax` designa a cota superior do paso temporal, á que identificaremos coma Δt_m, e en cada iteración do esquema semi-implícito o paso de tempo será calculado como:

$$\Delta t = \text{mín}(\Delta t_m, \Delta t_{\text{CFL}}), \tag{4.19}$$

onde Δt_{CFL} é o paso temporal límite que marca a condición CFL. Sen embargo, a modificación da etapa convectiva no método implícito elimina definitivamente este obstáculo e o noso paso temporal sempre será $\Delta t = \Delta t_m$. Polo tanto, esperamos un comportamento estable incluso para pasos de tempo superiores aos que nos marcaría a condición CFL na discretización explícita do subsistema convectivo.

Problemas de Riemann

Introducimos agora os catro problemas de Riemann cos que imos comprobar o método implícito e realizar a comparación co método semi-implícito. Os detalles dos parámetros para cada un deles poden ser atopados no Cadro 4.3. As solucións exactas son calculadas usando un *exact Riemann solver*, presentado en [38]. Nos tests a densidade é $\rho = 1050$, consideramos un tubo de lonxitude $L = 0.3$ e todos eles presentan un salto inicial no punto medio do conduto, menos o RP4 que o ten a $\frac{1}{3}$ da lonxitude total. Nas seguintes seccións comparamos entre o método semi-implícito e o implícito a través destes problemas. Ademais, se non se indica o contrario, os tests serán realizados con $N_C = 400$, $\varepsilon_N = 10^{-3}$ e $\varepsilon_B = 10^{-7}$.

Test	K_{ref}	m	n	$A_{0,\text{ref}}$	$p_{e,\text{ref}}$	A_L	A_R	u_L	u_R	t_f
RP1	$2 \cdot 10^4$	0.5	0	$3.14 \cdot 10^{-4}$	0	$3.5 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	0	0	0.02
RP2	5	10	-1.5	10^{-4}	0	$0.99A_{0,\text{ref}}$	$2.08A_{0,\text{ref}}$	0	0	0.007
RP3	33.333	10	-1.5	$2.877 \cdot 10^{-5}$	0	$3.2 \cdot 10^{-5}$	$3.2 \cdot 10^{-5}$	0.1	0.2	0.01
RP4	58725	0.5	0	$3.14 \cdot 10^{-4}$	0	$A_{0,\text{ref}}$	$A_{0,\text{ref}}$	1	-1	0.008

Cadro 4.3: Parámetros e condicións de contorno usados para os problemas de Riemann: K_{ref} a rixidez de referencia; m e n os exponentes da parte elástica; $A_{0,\text{ref}}$ a área da sección de referencia; $p_{e,\text{ref}}$ a presión externa de referencia. A_L, A_R, u_L e u_R son as seccións e as velocidades nos contornos esquerdo e dereito, respectivamente. Os valores están en unidades do Sistema Internacional.

4.3.1. Validación do método numérico

Buscaremos valores de Δt_m que non superen a condición CFL, para que os dous algoritmos computen as solucións cun mesmo paso de tempo constante. Nos Cadros 4.4 e 4.5 móstranse os erros L^2 para cada un dos tests con valores adecuados de Δt para $N_C = 400$. Como podemos ver, os erros son moi pequenos e os métodos dan residuos moi similares entre eles en todos os tests, validando así o novo método completamente implícito. As figuras 4.3, 4.4, 4.5 e 4.6 mostran esta concordancia entre os métodos de forma gráfica.

	RP1		RP2	
Δt	10^{-4}		10^{-4}	
	SI	I	SI	I
p	37.877	37.842	102.21	102.94
A	$8.5522 \cdot 10^{-6}$	$8.5525 \cdot 10^{-6}$	$1.0747 \cdot 10^{-5}$	$1.0746 \cdot 10^{-5}$
u	$8.1952 \cdot 10^{-2}$	$8.1954 \cdot 10^{-2}$	$2.0111 \cdot 10^{-1}$	$2.0065 \cdot 10^{-1}$

Cadro 4.4: Erros L^2 para os problemas de Riemann RP1 e RP2 nas variables de presión p, área A e velocidade u co mesmo paso de tempo Δt nos métodos semi-implícito (SI) e implícito (I). As unidades son do Sistema Internacional.

	RP3		RP4	
Δt	10^{-4}		10^{-4}	
	SI	I	SI	I
p	100.76	100.68	270.19	269.58
A	$1.3848 \cdot 10^{-6}$	$1.3844 \cdot 10^{-6}$	$1.7121 \cdot 10^{-5}$	$1.7124 \cdot 10^{-5}$
u	$1.6721 \cdot 10^{-1}$	$1.6723 \cdot 10^{-1}$	$2.6933 \cdot 10^{-1}$	$2.6936 \cdot 10^{-1}$

Cadro 4.5: Erros L^2 para os problemas de Riemann RP3 e RP4 nas variables de presión p, área A e velocidade u co mesmo paso de tempo Δt nos métodos semi-implícito (SI) e implícito (I). As unidades son do Sistema Internacional.

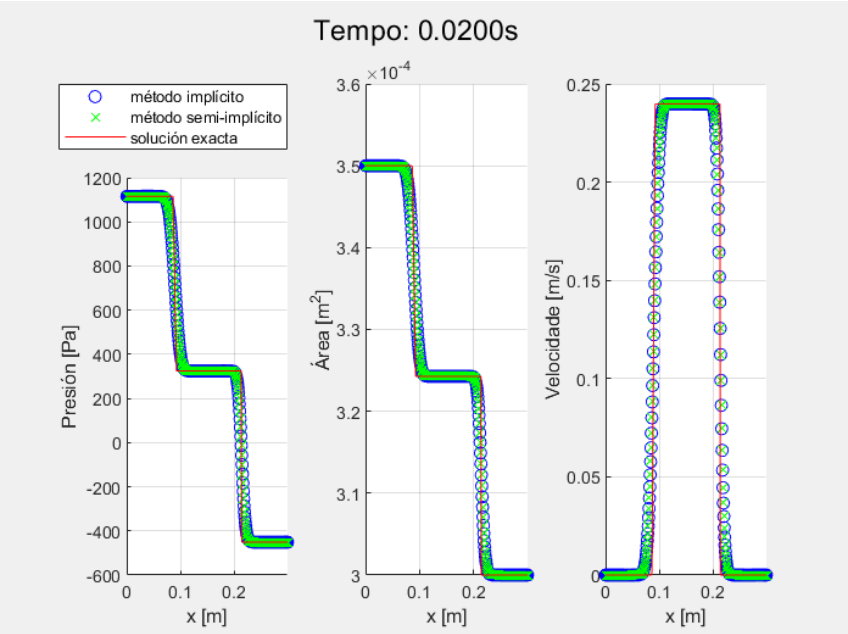

Figura 4.3: Comparación dos resultados numéricos da presión, da área e da velocidade da sección dados polo método semi-implícito e polo método implícito cos resultados da solución exacta con $\Delta t_m = 10^{-4}$ para o RP1.

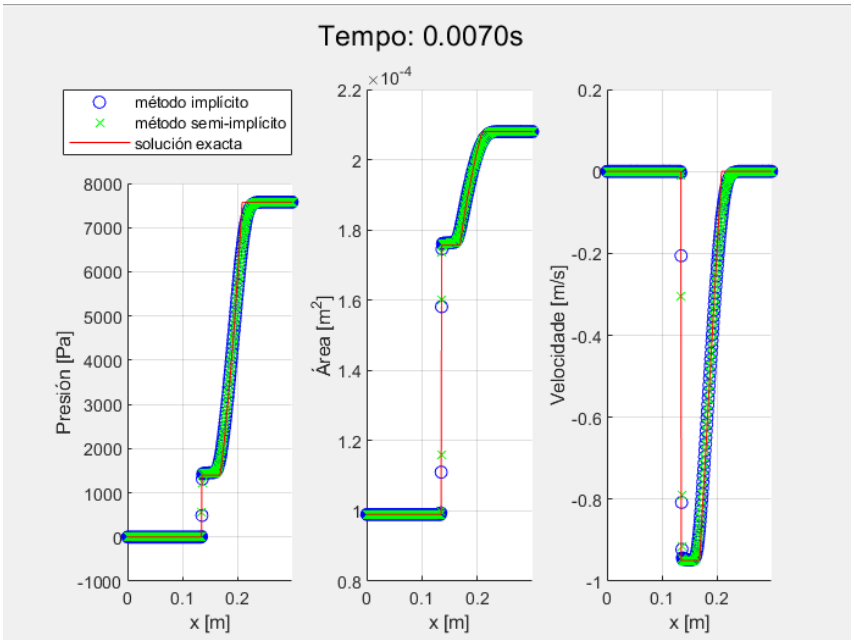

Figura 4.4: Comparación dos resultados numéricos da presión, da área e da velocidade da sección dados polo método semi-implícito e polo método implícito cos resultados da solución exacta con $\Delta t_m = 10^{-4}$ para o RP2.

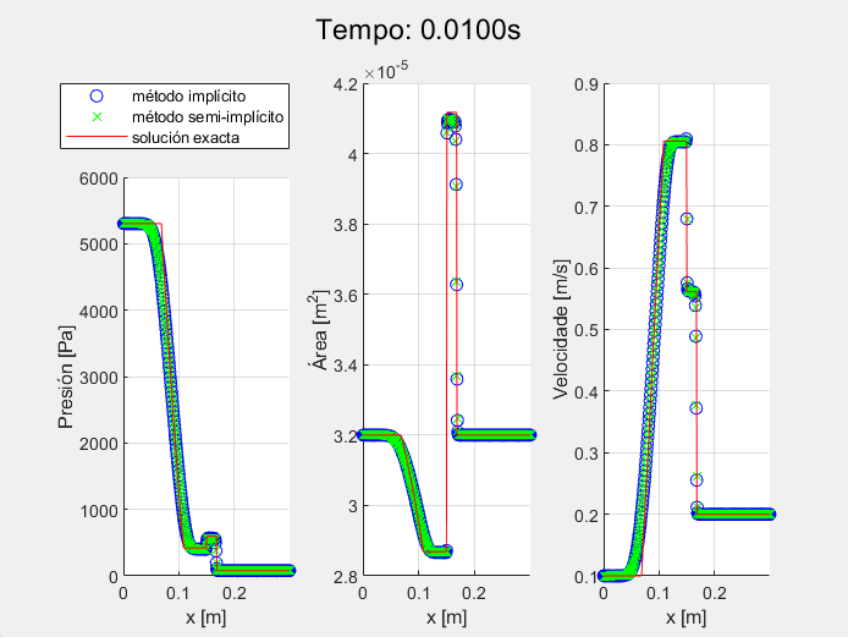

Figura 4.5: Comparación dos resultados numéricos da presión, da área e da velocidade da sección dados polo método semi-implícito e polo método implícito cos resultados da solución exacta con $\Delta t_m = 10^{-4}$ para o RP3.

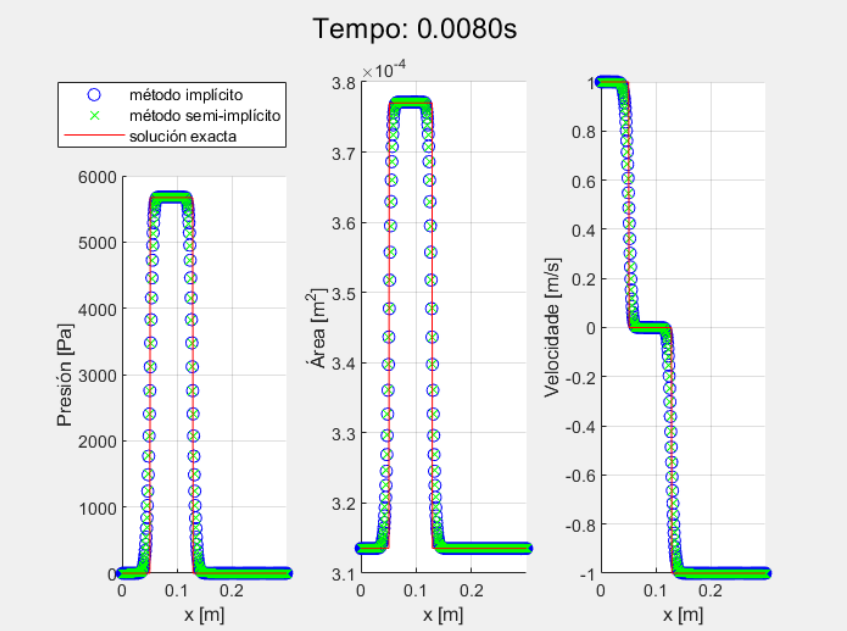

Figura 4.6: Comparación dos resultados numéricos da presión, da área e da velocidade da sección dados polo método semi-implícito e polo método implícito cos resultados da solución exacta con $\Delta t_m = 10^{-4}$ para o RP4.

4.3.2. Estudo das propiedades de estabilidade

Se executamos o RP2 para o método semi-implícito cun $\Delta t_m = 5 \cdot 10^{-4}$, vemos que o paso temporal nunca chega a tomar ese valor, e que non alcanza valores superiores a $3.9 \cdot 10^{-4}$, debido á limitación (2.21). De feito, se despexamos o valor do CFL na anterior ecuación, temos que:

$$\text{CFL} = \frac{\Delta t}{\Delta x} \max_{j \in \{1,\dots,N_D\}} (2|u^n_{j-\frac{1}{2}}|), \tag{4.20}$$

e, desta maneira, podemos calcular canto valería esta variable no caso de tomar $\Delta t = \Delta t_m$. Para o test RP2 con $\Delta t = 5 \cdot 10^{-4}$ alcánzanse valores de CFL $\approx 1.23 > 1$, polo que non podemos garantir a estabilidade para este problema de Riemann co método semi-implícito. Sen embargo, se facemos o mesmo test co esquema implícito observamos que non existe limitación do paso temporal e que aínda así hai consistencia. De feito, vemos boa concordancia entre os resultados, a pesar de que o implícito toma menos pasos para chegar á solución (ver Figura 4.7).

Outro exemplo podemos velo se executamos para o semi-implícito o RP3 con $\Delta t_m = 5 \cdot 10^{-4}$, onde vemos que o paso temporal non toma ese valor e que non supera $2.26 \cdot 10^{-4}$. Neste exemplo, se tomamos $\Delta t = \Delta t_m$ na fórmula (4.20), alcánzanse valores de CFL $\approx 1.07 > 1$; de novo, non se pode garantir a consistencia do semi-implícito. Tomar este paso temporal co método implícito fainos chegar ás mesmas conclusións que no test anterior, coma se pode observar na Figura 4.8.

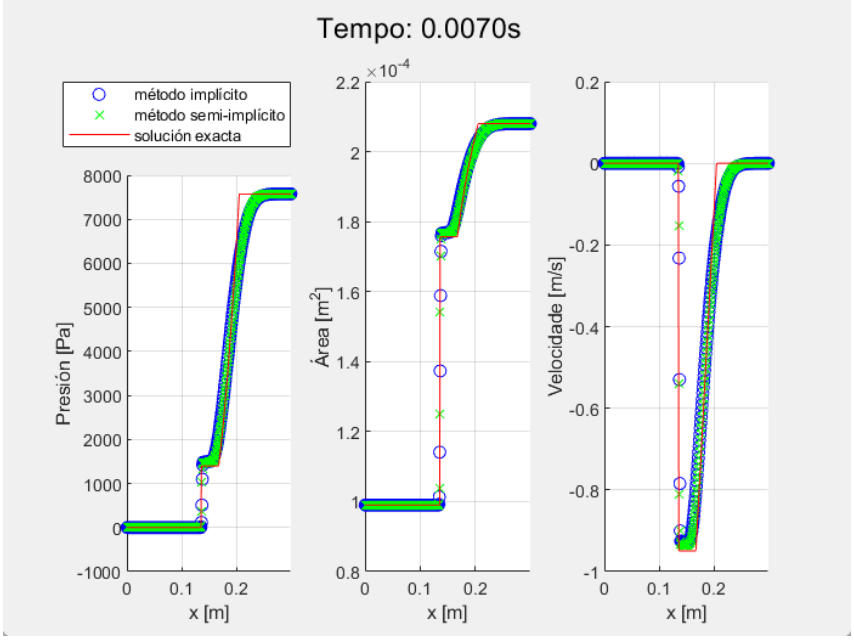

Figura 4.7: Comparación dos resultados numéricos da presión, da área e da velocidade da sección dados polo método semi-implícito e polo método implícito cos resultados da solución exacta con $\Delta t_m = 5 \cdot 10^{-4}$ para o RP2 (Δt variable).

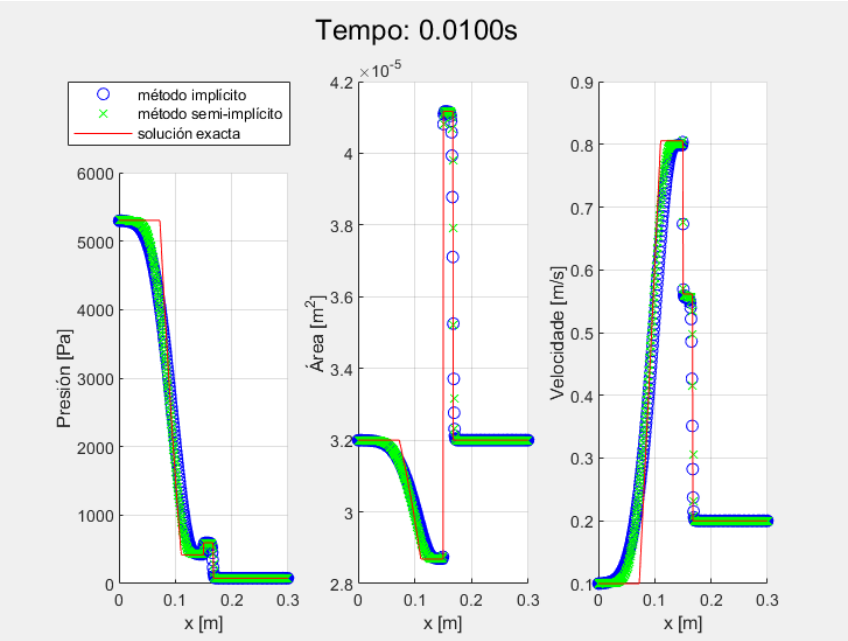

Figura 4.8: Comparación dos resultados numéricos da presión, da área e da velocidade da sección dados polo método semi-implícito e polo método implícito cos resultados da solución exacta con $\Delta t_m = 5 \cdot 10^{-4}$ para o RP3 (Δt variable).

4.3.3. Eficiencia computacional

A eficiencia computacional depende do Δt_m escollido. Se tomamos un valor superior ao marcado no método semi-implícito pola condición CFL, o esquema implícito tomará pasos de tempo máis grandes en comparación, mantendo os bos resultados obtidos. No Cadro 4.6 mostramos un caso para cada problema de Riemann onde a variable Δt_m é o suficientemente grande como para que o método implícito realice un menor número de iteracións. Por esta razón, a execución tarda menos tempo computacional, que era o obxectivo principal polo que desenvolvemos este método implícito. Os programas cos que se fixeron todas as simulacións veñen presentados no Apéndice I.3.

	RP1		RP2		RP3		RP4	
Δt_m	$2 \cdot 10^{-3}$		$5 \cdot 10^{-4}$		$5 \cdot 10^{-4}$		$5 \cdot 10^{-4}$	
	SI	I	SI	I	SI	I	SI	I
Tempo	4.92	3.22	130.09	93.84	166.66	55.65	8.04	6.90
Nº pasos	14	10	19	14	52	20	24	16

Cadro 4.6: Tempo computacional en segundos e número de pasos dos métodos semi-implícito (SI) e implícito (I) para a resolución dos problemas de Riemann cun Δt_m maior que o dado pola condición CFL.

Capítulo 5

Conclusión

Os modelos de fluxo sanguíneo están no seu momento de maior investigación ata o momento, e coa chegada de novos avances tecnolóxicos parece que a tendencia tende á alza. Na Introdución, fixemos un percorrido polas diferentes variantes de resolución deste problema, e centrámonos en modificar un esquema semi-implícito para o modelo 1D desenvolvido en [10]. Este modelo usa unha técnica de división da ecuación do sangue baseada en [30] que a parte en tres subsistemas: un subsistema convectivo, que é discretizado explícitamente a través dun método de volumes finitos cunha función de fluxo numérico de Ducros; un subsistema difusivo, só necesario se é considerada a viscoelasticidade do conducto e que é resolto implícitamente usando un BTCS; e un subsistema de presións, tamén resolto de maneira implícita con volumes finitos. Este sistema ten unha condición CFL que depende só da velocidade do fluxo, e non da velocidade da onda ou dos termos viscoelásticos. Ademais, impleméntase un θ-*method* para correccións na variable temporal.

No Capítulo 3, modificamos a etapa convectiva do modelo anterior e resolvémola cun método de Newton inexacto, onde a variable do paso é calculada a través do BiCGSTAB, un método de subespazos de Krylov. Así, eliminamos de maneira permanente a restricción CFL da parte convectiva e obtemos un método que pasa a ser completamente implícito, e, polo tanto, incondicionalmente estable.

Os resultados vistos no Capítulo 4 mostran unha concordancia moi forte entre os dous métodos, validando os resultados tanto entre eles como ante as solucións exactas. Sen embargo, o método completamente implícito mostra unha maior estabilidade cando pasos de tempo máis grandes son considerados. Nestes casos tamén se observa unha maior eficiencia computacional, chegando a solucións similares ás do método semi-implícito cun menor número de pasos de tempo así como nun menor tempo computacional.

O método implícito de volumes finitos para sistemas de leis de conservación hiperbólicas

foi verificado numéricamente, pero non realizamos unha análise teórica da súa converxencia, e tampouco verificamos que o método aplicado á ecuación do sangue mantén o comportamento de primeira orde. A diferenza de como fan en [10], nós non comprobamos se o método pode ser extendido a unións de condutos, ou incluso a redes deles. Penso que sería interesante facer estas validacións e comparar cos resultados obtidos para o método semi-implícito. Poderíase, tamén, adaptar a discretización implícita do subsistema convectivo para intentar acadar unha converxencia de maior orde, ou incluso intentar aumentar o número de dimensións a dúas. Todas estas modificacións quedan propostas para futuros investigadores que queiran traballar no campo da simulación do fluxo sanguíneo.

Neste traballo apliquei diversas técnicas e competencias aprendidas na carreira, que me serviron para abordar os conceptos e as metodoloxías con maior facilidade. En *Informática* familiariceime coa programación no contexto matemático, e foi a primeira ocasión na que traballei con Matlab. No curso *Cálculo Numérico nunha Variable* fun introducido ás técnicas de resolución de ecuacións mediante métodos numéricos, e estudei o método de Newton-Rhapson, que é modificado e empregado para achar unha solución á ecuación (3.1); ademais, coñecín os primeiros métodos para o cálculo de derivadas e apliquei os conceptos en entornos de programación.

Por outra banda, as materias *Introdución ás Ecuacións Diferenciais Ordinarias*, *Ecuacións Diferenciais Ordinarias* e *Series de Fourier e Introdución ás Ecuacións en Derivadas Parciais* foron clave para obter unha base teórica e de interpretación das EDOs e, máis adiante, das EDPs. Tamén, en *Análise Numérico Matricial* descubrín métodos numéricos para a resolución de sistemas de ecuacións, tanto directos como iterativos, que foron fundamentais para a mellor comprensión do problema do cálculo do paso de Newton en (3.3); en clases prácticas, plasmei estas nocións en plataformas de codificación.

Logo, no curso de *Métodos Numéricos en Optimización e Ecuacións Diferenciais* vin métodos numéricos para a resolución de EDOs, como os métodos de Euler explícito e Euler implícito ou os de Runge-Kutta, e tamén profundicei nos conceptos de consistencia, estabilidade, converxencia e orde. Por outra parte, tamén adquirín coñecementos sobre métodos numéricos para sistemas de ecuacións lineais, entre os que se encontran o método do gradiente e o método do gradiente conxugado, que serven de base teórica para o desenvolvemento do BiCGSTAB. Ademáis, continuei a miña formación en Matlab aplicando todos estes coñecementos. Por último, na materia *Modelización Matemática* descubrín diferentes modelos para a mecánica de fluidos, que foron esenciais para facilitar o meu entendemento do modelo do fluxo de sangue nun conduto.

Apéndice I

Códigos MatLab

Este apéndice serve de repositorio para os códigos principais utilizados para obter os resultados e as gráficas do Capítulo 4. Supón a parte máis importante do TFG, e sen estes códigos o método implícito de volumes finitos para a modelización do fluxo sanguíneo en 1D non existiría fóra do campo teórico. Todos os programas aquí presentados foron escritos íntegramente por min, e mellorados co paso do tempo ata conseguir os resultados que esperabamos. A través deste proceso, fun capaz de profundizar nos métodos numéricos e de aplicar de maneira efectiva os conceptos teóricos aprendidos. En total, representan máis de 500 liñas de código.

Os códigos están comentados en inglés e a maior parte das variables teñen nomes neste idioma. Esta decisión é consciente, xa que considero que fai os algoritmos máis accesibles, ademais de que o inglés é o idioma estandarizado para a maioría de linguaxes de programación.

I.1. Método implícito de volumes finitos aplicado á ecuación de transporte

Nesta primeira parte introducimos o código que resolve a ecuación de transporte co método implícito de volumes finitos, como foi introducido na Sección 4.1. Depois, engadimos tamén a subrutina encargada de calcular o δq^{**} da ecuación (3.8), o algoritmo BiCGSTAB que resolve a ecuación do paso de Newton (3.3) e o procedemento que calcula o produto matriz-vector do lado esquerdo de (3.3), ademáis do código do fluxo numérico.

transport_eq

```
% Generate the grid
xi = -1;
```

```
xf = 1;
ngrid = 100;
dx = (xf-xi)/ngrid;
X = colon(xi+dx/2,dx,xf);

%Time step and total elapsed time
dt = 0.001;
time = 1;
time_steps = time/dt;

%q0 initial condition
N = 1;
q0 = zeros(N, ngrid);
for i = 1:ngrid
    q0(i) = cos(pi*X(i));
end

%lambda and alpha initialization
lambda = 0.5;
alpha = lambda;

qn = q0;

%First plot and plot limits
yl = min(q0) - 0.2;
yr = max(q0) + 0.2;
plot(X, qn, "bo")
ylim([yl yr])
title("0 s")

eps = 10^-10;
for step = 1:time_steps
    %Number of max steps to calculate the q^{n+1}
    maxsteps = 1000;
    qk = qn;
    Dqk = zeros(1,ngrid);
    for k = 1:maxsteps
        %To calculate deltaq, we use the previous q^n
```

```
        deltaq = calc_deltaq(dt, dx, qk, alpha, ngrid);
        fq = qk - qn + deltaq;
        nfq = norm(fq);

        % Use of the bicgstab method to calculate the newton step
        b_eps = 10^-12;
        b_its = 1000;
        newt_step = mod_bicgstab2(dt, dx, b_eps, -fq, Dqk, b_its, lambda);

        % Inexact Newton
        delta = 1;
        qk1 = qk + delta*newt_step;
        fq1 = qk1 - qn + deltaq;
        while norm(fq1) > nfq
            delta = delta/2;
            qk1 = qk + delta*newt_step;
            fq1 = qk1 - qn + deltaq;
        end
        nfq1 = norm(fq1);
        qk = qk1;
        if nfq1 < eps
            qk = double(qk);
            qn = qk;
            break
        end

        % Step update
        Dqk = newt_step;
        if k == maxsteps
            fprintf('WARNING\n')
        end
    end
    % Plot
    plot(X, qn, 'bo')
    ylim([yl yr])
    title([num2str(dt*step),' s'])
    drawnow
end
```

```
%Draw exact solution and compare
exact = zeros(N, ngrid);
for i = 1:ngrid
    exact(i) = cos(pi*(X(i)-lambda*time));
end
hold on
plot(X, exact, "k-")
ylim([yl yr])

diff = qn - exact;
plot(X, diff, "r-")
title('Ecuaci n de transporte, t = 1 s')
legend('m todo impl cito','soluci n exacta','erro')
ylabel('q(x,t)')
xlabel('x')

norm1 = dx*sum(abs(diff));
norm2 = sqrt(dx*sum(abs(diff).^2));

fprintf("The norm 1 is %e\n", norm1)
fprintf("The norm 2 is %e\n", norm2)
fprintf("dx is %f and dt is %f\n",dx, dt)
```

calc_deltaq

```
function qdelta = calc_deltaq(dt, dx, q, alpha, N)
    %Calculates the RHS of the Newton step equation
    qdelta = zeros(1,N);
    for i = 1:N
        for nij = [-1, 1]
            j = mod(i+nij - 1, N) + 1;
            qdelta(i) = qdelta(i) + Flux(q, i, j, nij, alpha);
        end
        qdelta(i) = qdelta(i)*dt/dx;
end
```

mod_bicgstab2

```
function x = mod_bicgstab2(dt, dx, eps, fq, Dqk, nits, lambda)
    %BiCGSTAB algorithm that solves implicitly the Newton step equation
    Aqk = JDelta(dt, dx, Dqk, lambda);
    rk = fq - Aqk;
    rt0 = rk;
    pk = rk;
    for k = 0:nits
        Apk = JDelta(dt, dx, pk, lambda);
        alphak = dot(rk,rt0)/dot(Apk,rt0);
        sk = rk - alphak*Apk;
        Ask = JDelta(dt, dx, sk, lambda);
        wk = dot(Ask,sk)/dot(Ask,Ask);
        Dqk = Dqk + alphak*pk + wk*sk;
        rk_old = rk;
        rk = sk - wk*Ask;

        if norm(rk) < eps
            x = Dqk;
            return
        end

        betak = (alphak/wk)*(dot(rk,rt0)/dot(rk_old,rt0));
        pk = rk + betak*(pk - wk*Apk);
    end
    fprintf("Not enough BiCGSTAB iterations\n")
    fprintf("Norm(rk) = %f\n\n",norm(rk))
    x = Dqk;
end
```

JDelta

```
function Aq = JDelta(dt, dx, Deltaq, lambda)
    %Finds the matrix-vector product given Deltaq
    N = size(Deltaq, 2);
    Aq = Deltaq;
    for i = 1:N
```

```
        for nij = [-1, 1]
            j = mod(i+nij - 1, N) + 1;
            Aq(i) = Aq(i) + dt/dx*(1/2*lambda*(Deltaq(i) + Deltaq(j))*nij - 1
        end
end
```

Flux

```
function flux = Flux(q, i, j, nij, alpha)
    flux = 1/2*(F(q(i)) + F(q(j)))*nij - 1/2*alpha*(q(j) - q(i));
end
```

I.2. Método implícito de volumes finitos aplicado á ecuación de Burgers

Engádense agora as funcións análogas ás da ecuación de transporte para a ecuación de Burgers. Hai unha gran similitude entre os programas, que é natural considerando que estamos aplicando o mesmo método. Engadimos, a parte, a función que computa a linealización do fluxo numérico para o produto matriz-vector.

burgers_eq

```
%Generate the grid
xi = -0.8;
xf = 0.8;
ngrid = 320;
dx = (xf-xi)/ngrid;
X = colon(xi+dx/2,dx,xf);

%Time step and total elapsed time
dtref = 0.01;
dt = dtref;
tend = 0.5;
time_steps = tend/dt;

%q0 initial condition
```

```
N = 1;
q0 = zeros(N, ngrid);
for i = 1:ngrid
    q0(i) = exp(-X(i)^2/0.01);
end

%Number of variables
qn = q0;

%First plot and plot limits
yl = min(q0) - 0.2;
yr = max(q0) + 0.2;
plot(X, qn, "bo")
ylim([yl yr])
title("0 s")

eps = 10^-8;
for step = 1:time_steps
    %Number of max steps to calculate the q^{n+1}
    maxsteps = 1000;
    qk = qn;
    Dqk = zeros(1,ngrid);
    for k = 1:maxsteps
        %To calculate deltaq, we use the previous q^n
        deltaq = calc_deltaq(dt, dx, qk, ngrid);
        fq = qk - qn + deltaq;
        nfq = norm(fq);

        %Use of the bicgstab method to calculate the newton step
        b_eps = 10^-8;
        b_its = 1000;
        newt_step = mod_bicgstab2(dt, dx, b_eps, -fq, Dqk, qn, qk, b_its);

        %Inexact Newton
        delta = 1;
        qk1 = qk + delta*newt_step;
        fq1 = qk1 - qn + deltaq;
        while norm(fq1) > nfq
```

```
            delta = delta/2;
            qk1 = qk + delta*newt_step;
            fq1 = qk1 - qn + deltaq;
        end
        nfq1 = norm(fq1);
        qk = qk1;
        if nfq1 < eps
            qk = double(qk);
            qn = qk;
            break
        end
        Dqk = newt_step;
        if k == maxsteps
            fprintf("WARNING\n")
        end
    end
    plot(X, qn, "bo")
    ylim([yl yr])
    title([num2str(dt*step),' s'])
    drawnow
end
```

calc_deltaq

```
function qdelta = calc_deltaq(dt, dx, q, N)
    % Calculates the RHS of the Newton step equation
    qdelta = zeros(1,N);
    for i = 1:N
        for nij = [-1, 1]
            j = mod(i+nij - 1, N) + 1;
            qdelta(i) = qdelta(i) + Flux(q, i, j, nij);
        end
        qdelta(i) = qdelta(i)*dt/dx;
end
```

mod_bicgstab2

```
function x = mod_bicgstab2(dt, dx, eps, fq, Dqk, qn, qk, nits)
    %BiCGSTAB algorithm that solves implicitly the Newton step equation
    Aqk = JDelta(dt, dx, Dqk, qn, qk);
    rk = fq - Aqk;
    rt0 = rk;
    pk = rk;
    for k = 0:nits
        Apk = JDelta(dt, dx, pk, qn, qk);
        alphak = dot(rk,rt0)/dot(Apk,rt0);
        sk = rk - alphak*Apk;
        Ask = JDelta(dt, dx, sk, qn, qk);
        wk = dot(Ask,sk)/dot(Ask,Ask);
        Dqk = Dqk + alphak*pk + wk*sk;
        rk_old = rk;
        rk = sk - wk*Ask;

        if norm(rk) < eps
            x = Dqk;
            return
        end

        betak = (alphak/wk)*(dot(rk,rt0)/dot(rk_old,rt0));
        pk = rk + betak*(pk - wk*Apk);
    end
    fprintf("Not enough BiCGSTAB iterations\n")
    fprintf("Norm(rk) = %f\n\n",norm(rk))
    x = Dqk;
end
```

JDelta

```
function Aq = JDelta(dt, dx, Dq, qn, qk)
    %Finds the matrix-vector product given Deltaq, qn and qk
    N = size(Dq, 2);
    Aq = Dq;
    for i = 1:N
        for nij = [-1, 1]
            j = mod(i+nij - 1, N) + 1;
```

```
            Aq(i) = Aq(i) + dt/dx*LinFlux(Dq, qn, qk, i, j, nij);
        end
    end
```

Flux

```
function flux = Flux(q, i, j, nij)
    alpha = max(q(i), q(j));
    flux = 1/2*(F(q(i)) + F(q(j)))*nij - 1/2*alpha*(q(j) - q(i));
end
```

LinFlux

```
function flux = LinFlux(Dq, qn, qk, i, j, nij)
    alpha = max(qn(i), qn(j));
    fi = Dq(i)*qk(i);
    fj = Dq(j)*qk(j);
    flux = 1/2*(fi + fj)*nij - 1/2*alpha*(Dq(j) - Dq(i));
end
```

I.3. Método implícito de volumes finitos aplicado á ecuación do sangue

Nesta sección achégase, en primeiro lugar, un esquema do programa *SIBlood_ mod*, que inclúe tanto o método semi-implícito para a resolución numérica do modelo do sangue presentado no Capítulo 2, como o método implícito desenvolto no Capítulo 3. Este programa foi aportado polas titoras e eu centreime en entendelo e en modificar a resolución do subsistema convectivo para que pasase a ter unha discretización implícita, seguindo o procedemento teórico do Capítulo 3. Ademais, engadín ferramentas para facilitar a comparación entre os dous métodos e permitir obter o tempo computacional e os erros se comparamos ca solución exacta.

Despois, engádense os códigos íntegros do Newton inexacto para o cálculo implícito do subsistema convectivo, a subrutina que calcula o δq^{**} da ecuación (3.8), o proceso que calcula as funcións de fluxo numérico de Rusanov (3.10) e de Ducros (3.11), o algoritmo BiCGSTAB que resolve a ecuación do paso de Newton (3.3), o procedemento que calcula o produto matriz-vector do lado esquerdo de (3.3) e, finalmente, o bloque de código encargado do cálculo das linealizacións

dos fluxos numéricos de Rusanov (3.13) e de Ducros (3.12). Estes códigos foron programados dende cero e de maneira íntegra por min, partindo das ecuacións mostradas no Capítulo 3.

Esquema de fluxo do código para a modelización do sangue

- Definición e inicialización de variables.
- Configuración e elección dos problemas de Riemann.
- Inicialización das variables do bucle temporal e da solución exacta.
- Inicio do bucle na variable temporal.
  1. Cálculo de variables auxiliares.
  2. Cómputo do paso de tempo.
  3. Axuste das condicións de contorno.
  4. Subsistema viscoso.
  5. Subsistema convectivo.
  6. Subsistema de presións.
  7. Etapa de correccións.
  8. Graficación da solución numérica e da exacta.
- Mostra dos erros, do tempo computacional e do gráfico comparativo.

Subsistema convectivo

```
qk = q;
Dqk = q;
for iInexNewton = 1:inex_its
    % Calculation of the RHS of the equation and the norm of h
    deltaq = calc_deltaq(qk, u);
    fq = qk - q + deltaq;
    nfq = norm(fq);

    % Use of the bicgstab method to calculate the newton step
    newt_step = mod_bicgstab2(bicgstab_eps, -fq, Dqk, u, qk, bicgstab_its);

    % Inexact Newton
    delta = 1;
```

```
        qk1 = qk + delta*newt_step;
        fq1 = qk1 - q + deltaq;
        nfq1 = norm(fq1);
        while nfq1 >= nfq
            delta = delta/2;
            qk1 = qk + delta*newt_step;
            fq1 = qk1 - q + deltaq;
            nfq1 = norm(fq1);
        end
        qk = qk1;

        % Convergence condition
        if nfq1 < inex_eps
            deltaq = double(deltaq);
            break
        end

        % Deltaq for the next newton step
        Dqk = newt_step;

        % In case there isn't convergence, the program throws a warning
        if iInexNewton == inex_its
            fprintf('WARNING, Inexact Newton didn't converge\n')
        end
    end

    % Left boundary
    Fq(:,1) = qBL;
    % Interior
    for i=2:IMAX
        Fq(:, i) = Fq(:,i) - deltaq(:,i);
    end
    % Right boundary
    Fq(:,IMAX+1) = qBR;
```

calc_deltaq

```
function deltaq = calc_deltaq(q, u)
```

```
    % Code to estimate the deltaq of the RHS of the Newton equation
    global dt dx IMAX

    deltaq = zeros(1, IMAX+1);
    Fluxval = zeros(1, IMAX);

    for i=1:IMAX
        Fluxval(i) = Flux(q, u, i, i+1);
    end
    for i=2:IMAX
        deltaq(i) = dt/dx*(-Fluxval(i-1)+Fluxval(i));
    end
end
```

Flux

```
function flux = Flux(q, u, i, j)
    % Calculates the flux between q_i and q_j
    [ui, uj] = deal(u(i), u(j));
    [qi, qj] = deal(q(i), q(j));
    unij = 1/2*(ui + uj);
    alpha = 2*abs(unij);

    % Ducros
    fluxd = 1/2*(qi + qj)*unij - 1/2*alpha*(qj - qi);

    % Rusanov
    fluxr = 1/2*(F(qi,ui) + F(qj,uj)) - 1/2*alpha*(qj - qi);

    flux = fluxr;
end
```

mod_bicgstab2

```
function x = mod_bicgstab2(eps, hq, Dqk, qk, un, nits)
    % Modified BiCGSTAB algorithm for the calculation of the Newton step
    Aqk = JDelta(Dqk, qk, un);
```

```
        rk = hq - Aqk;
        rt0 = rk;
        pk = rk;
        for k = 0:nits
            Apk = JDelta(pk, qk, un);
            alphak = dot(rk,rt0)/dot(Apk,rt0);
            sk = rk - alphak*Apk;
            Ask = JDelta(sk, qk, un);
            wk = dot(Ask,sk)/dot(Ask,Ask);
            Dqk = Dqk + alphak*pk + wk*sk;
            rk_old = rk;
            rk = sk - wk*Ask;
            betak = (alphak/wk)*(dot(rk,rt0)/dot(rk_old,rt0));
            pk = rk + betak*(pk - wk*Apk);
        end
        %In case there isn't convergence, the program prints a message
        fprintf("Not enough BiCGSTAB iterations\n")
        fprintf("Norm(rk) = %f\n\n",norm(rk))
        x = Dqk;
    end
```

JDelta

```
    function Aq = BJDelta(Dq, qk, u)
        %Program to find the matrix-vector product of Dq and qk
        global dt dx IMAX
        Aq = Dq;  %IMAX+1
        LinFluxval = zeros(1, IMAX);

        for i = 1:IMAX
            LinFluxval(i) = LinFlux(Dq, qk, u, i, i+1, 1);
        end
        for i = 2:IMAX
            Aq(i) = Aq(i) + dt/dx*(-LinFluxval(i-1)+LinFluxval(i));
        end
    end
```

LinFlux

```
function flux = LinFlux(Dq, qk, un, i, j, nij)
    % Routine to compute the linealization of the numerical flux
    [Dqi, Dqj] = deal(Dq(i), Dq(j));
    [uni, unj] = deal(un(i), un(j));
    unij = 1/2*(uni+unj)*nij;
    alpha = 2*abs(unij);

    % Ducros
    fluxd = 1/2*unij*(Dqi+Dqj) - 1/2*alpha*(Dqj-Dqi);

    % Rusanov
    [uix, ujx] = deal(qk(i)*nij,qk(j)*nij);
    fluxr = (uix*Dqi+ujx*Dqj) - 1/2*alpha*(Dqj-Dqi);

    flux = fluxr;
end
```

Bibliografía

[1] INE. Las 15 causas de muerte más frecuentes en españa. [Online]. Available: https://public.tableau.com/views/CAUSAS_DE_MUERTE/Dashboard1?:showVizHome=no&:embed=true

[2] INE. Defunciones por causas (lista reducida) por sexo y grupos de edad. [Online]. Available: https://www.ine.es/jaxiT3/Tabla.htm?t=7947

[3] A. Quarteroni, L. Formaggia, and A. Veneziani, "Cardiovascular mathematics," in *Proceedings of the International Congress of Mathematicians*, vol. 1. European Mathematical Society Madrid, Spain, 2006, pp. 479–512.

[4] D. Boffi, L. F. Pavarino, G. Rozza, S. Scacchi, and C. Vergara, *Mathematical and Numerical Modeling of the Cardiovascular System and Applications*. Springer, 2018, vol. 16.

[5] A. Quarteroni, *Modelling the cardiovascular system - A mathematical adventure: Part I.* SIAM News, 2001, vol. 34.

[6] A. Quarteroni, *Modelling the cardiovascular system - A mathematical adventure: Part II.* SIAM News, 2001, vol. 34.

[7] L. Grinberg, E. Cheever, T. Anor, J. R. Madsen, and G. Karniadakis, "Modeling blood flow circulation in intracranial arterial networks: a comparative 3d/1d simulation study," *Annals of biomedical engineering*, vol. 39, pp. 297–309, 2011.

[8] L. Formaggia, D. Lamponi, and A. Quarteroni, "One-dimensional models for blood flow in arteries," *Journal of engineering mathematics*, vol. 47, pp. 251–276, 2003.

[9] B. Steele, "Using one-dimensional finite element analysis to estimate differential pressure of renal artery stenoses," in *2007 Computers in Cardiology*. IEEE, 2007, pp. 391–394.

[10] A. Lucca, S. Busto, L. Müller, E. Toro, and M. Dumbser, "A semi-implicit finite volume scheme for blood flow in elastic and viscoelastic vessels," *Journal of Computational Physics*, vol. 495, p. 112530, 2023.

[11] S. J. Sherwin, L. Formaggia, J. Peiro, and V. Franke, "Computational modelling of 1d blood flow with variable mechanical properties and its application to the simulation of wave propagation in the human arterial system," *International journal for numerical methods in fluids*, vol. 43, no. 6-7, pp. 673–700, 2003.

[12] H. M. Hasan, A. Coccarelli, and P.Ñithiarasu, "Novel semi-implicit, locally conservative galerkin (silcg) methods: application to blood flow in a systemic circulation," *Computer Methods in Applied Mechanics and Engineering*, vol. 332, pp. 217–233, 2018.

[13] V. Casulli, M. Dumbser, and E. F. Toro, "Semi-implicit numerical modeling of axially symmetric flows in compliant arterial systems," *International journal for numerical methods in biomedical engineering*, vol. 28, no. 2, pp. 257–272, 2012.

[14] L. Formaggia, F.Ñobile, A. Quarteroni, and A. Veneziani, "Multiscale modelling of the circulatory system: a preliminary analysis," *Computing and visualization in science*, vol. 2, pp. 75–83, 1999.

[15] T. Köppl, E. Vidotto, and B. Wohlmuth, "A 3d-1d coupled blood flow and oxygen transport model to generate microvascular networks," *International journal for numerical methods in biomedical engineering*, vol. 36, no. 10, p. e3386, 2020.

[16] L. O. Müller and E. F. Toro, "A global multiscale mathematical model for the human circulation with emphasis on the venous system," *International journal for numerical methods in biomedical engineering*, vol. 30, no. 7, pp. 681–725, 2014.

[17] P. J. Blanco, S. M. Watanabe, M. A. R. Passos, P. A. Lemos, and R. A. Feijóo, "An anatomically detailed arterial network model for one-dimensional computational hemodynamics," *IEEE Transactions on biomedical engineering*, vol. 62, no. 2, pp. 736–753, 2014.

[18] A. C. I. Malossi, P. J. Blanco, P. Crosetto, S. Deparis, and A. Quarteroni, "Implicit coupling of one-dimensional and three-dimensional blood flow models with compliant vessels," *Multiscale Modeling & Simulation*, vol. 11, no. 2, pp. 474–506, 2013.

[19] L. O. Müller, C. Parés, and E. F. Toro, "Well-balanced high-order numerical schemes for one-dimensional blood flow in vessels with varying mechanical properties," *Journal of computational physics*, vol. 242, pp. 53–85, 2013.

[20] T. J. Hughes, G. Engel, L. Mazzei, and M. G. Larson, "The continuous galerkin method is locally conservative," *Journal of Computational Physics*, vol. 163, no. 2, pp. 467–488, 2000.

[21] A.Ñ. Brooks and T. J. Hughes, "Streamline upwind/petrov-galerkin formulations for convection dominated flows with particular emphasis on the incompressible navier-stokes equations," *Computer methods in applied mechanics and engineering*, vol. 32, no. 1-3, pp. 199–259, 1982.

[22] V. Assimakopoulos and K.Ñikolopoulos, "The theta model: a decomposition approach to forecasting," *International journal of forecasting*, vol. 16, no. 4, pp. 521–530, 2000.

[23] R. J. LeVeque, *Finite volume methods for hyperbolic problems*. Cambridge university press, 2002, vol. 31.

[24] G. B. Whitham, *Linear and nonlinear waves*. John Wiley & Sons, 2011.

[25] M. E. Vázquez-Cendón, *Solving hyperbolic equations with finite volume methods*. Springer, 2015, vol. 90.

[26] E. F. Toro, *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*. Springer Science & Business Media, 2013.

[27] J. C. Strikwerda, *Finite difference schemes and partial differential equations*. SIAM, 2004.

[28] R. Courant, K. Friedrichs, and H. Lewy, "On the partial difference equations of mathematical physics," *IBM journal of Research and Development*, vol. 11, no. 2, pp. 215–234, 1967.

[29] D. R. Durran, *Numerical methods for fluid dynamics: With applications to geophysics*. Springer Science & Business Media, 2010, vol. 32.

[30] E. F. Toro and M. Vázquez-Cendón, "Flux splitting schemes for the euler equations," *Computers & Fluids*, vol. 70, pp. 1–12, 2012.

[31] V. Casulli and P. Zanolli, "A nested newton-type algorithm for finite volume methods solving richards' equation in mixed form," *SIAM Journal on Scientific Computing*, vol. 32, no. 4, pp. 2255–2273, 2010.

[32] A. Bermudez and M. E. Vazquez, "Upwind methods for hyperbolic conservation laws with source terms," *Computers & Fluids*, vol. 23, no. 8, pp. 1049–1071, 1994.

[33] A. Lucca, S. Busto, and M. Dumbser, "An implicit staggered hybrid finite volume/finite element solver for the incompressible navier-stokes equations," *arXiv preprint arXiv:2302.05688*, 2023.

[34] F. Ducros, V. Ferrand, F.Ñicoud, C. Weber, D. Darracq, C. Gacherieu, and T. Poinsot, "Large-eddy simulation of the shock/turbulence interaction," *Journal of Computational Physics*, vol. 152, no. 2, pp. 517–549, 1999.

[35] M. R. Hestenes, E. Stiefel *et al.*, *Methods of conjugate gradients for solving linear systems*. NBS Washington, DC, 1952, vol. 49, no. 1.

[36] P. Sonneveld, "Cgs, a fast lanczos-type solver for nonsymmetric linear systems," *SIAM journal on scientific and statistical computing*, vol. 10, no. 1, pp. 36–52, 1989.

[37] H. A. van der Vorst, "Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, pp. 631–644, 1992.

[38] E. F. Toro and A. Siviglia, "Flow in collapsible tubes with discontinuous mechanical properties: mathematical model and exact solutions," *Communications in Computational Physics*, vol. 13, no. 2, pp. 361–385, 2013.