Pontificia Universidad Javeriana Cali
Facultad de Ingeniería y ciencias
Ingeniería de Sistemas y Computación
Proyecto de Grado

FOAMFLOW-2D: Herramienta computacional para la simulación bidimensional de la inyección de espuma en medios porosos estratificados

Marco Antonio Riascos Salguero

Director PhD.Andres Julian Castrillón Vásquez

Mayo de 2026

Abstract

The transport of complex fluids through porous media represents one of the most challenging problems in reservoir physics and contemporary process engineering. In this context, foam flow has emerged as an alternative for improving volumetric sweep efficiency in enhanced oil recovery projects and in the remediation of degraded aquifers.

This work presents FOAMFLOW-2D, a computational tool for the two-dimensional simulation of foam injection in stratified porous media. The application enables interactive simulations and real-time numerical analysis of foam flow by incorporating physicomathematical models that represent the transport of gas and liquid phases, as well as the evolution of foam texture.

The solution was implemented as a web-based platform that executes the computational core directly in the browser using modern architectures such as Web Workers, together with a graphical user interface developed with React. This approach enables the visualization and analysis of phenomena related to concentration profiles, injection pressures, and flow behavior in porous media without relying on complex local infrastructures.

FOAMFLOW-2D aims to facilitate access to advanced fluid modeling tools and serve as an experimentation and learning environment for students, researchers, and professionals interested in the numerical simulation of complex physical phenomena.

Keywords: numerical simulation, foam flow, porous media, enhanced oil recovery, computational simulation.

Resumen

El transporte de fluidos complejos en medios porosos representa uno de los desafíos más exigentes dentro de la física de yacimientos y la ingeniería de procesos contemporánea. En este contexto, el flujo de espuma ha emergido como una alternativa para mejorar la eficiencia de barrido volumétrico en proyectos de recuperación mejorada de hidrocarburos y en la remediación de acuíferos degradados.

Este trabajo presenta FOAMFLOW-2D, una herramienta computacional para la simulación bidimensional de la inyección de espuma en medios porosos estratificados. La aplicación permite realizar simulaciones interactivas y análisis numéricos del flujo de espuma en tiempo real, incorporando modelos físico-matemáticos que representan el transporte de las fases gaseosa y líquida, así como la evolución de la textura de la espuma.

La solución fue implementada como una plataforma web que ejecuta el núcleo computacional directamente en el navegador mediante arquitecturas modernas como Web Workers, junto con una interfaz gráfica desarrollada con React. Este enfoque permite visualizar y analizar fenómenos relacionados con perfiles de concentración, presiones de inyección y comportamiento del flujo en medios porosos, sin depender de infraestructuras locales complejas.

FOAMFLOW-2D busca facilitar el acceso a herramientas avanzadas de modelado de fluidos y servir como un entorno de experimentación y aprendizaje para estudiantes, investigadores y profesionales interesados en la simulación numérica de fenómenos físicos complejos.

Palabras clave: simulación numérica, flujo de espuma, medios porosos, recuperación mejorada de hidrocarburos, simulación computacional.

Índice general

Índice de cuadros

Índice de figuras

CAPÍTULO 1

Introducción

El transporte de fluidos complejos en medios porosos representa uno de los desafíos más exigentes dentro de la física de yacimientos y la ingeniería de procesos contemporánea. Entre estos sistemas, el flujo de espuma ha emergido como una alternativa para optimizar la eficiencia de barrido volumétrico [2] en proyectos de recuperación mejorada de hidrocarburos (EOR) y en la remediación ambiental de acuíferos degradados. En términos prácticos, la inyección directa de gas suele presentar problemas de canalización debido a su baja viscosidad y densidad; no obstante, al dispersarse en una fase líquida continua con surfactantes (sustancias químicas que disminuyen la tensión superficial del agua), la espuma resultante incrementa la viscosidad efectiva del fluido. Este fenómeno frena el avance descontrolado del gas y reconfigura las fuerzas capilares a escala microscópica, forzando un desplazamiento más homogéneo y eficiente a través de los canales del medio poroso.

A pesar de sus evidentes ventajas operacionales, la simulación numérica y el modelado predictivo de la espuma son tareas de una valida sofisticación matemática [4]. A diferencia de los fluidos convencionales, la espuma es un sistema dinámico cuya estructura interna cambia de manera continua, las burbujas nacen mediante mecanismos mecánicos como división laminar [5, 6], se mueven a través de gargantas de poro sumamente estrechas, y se destruyen por coalescencia (proceso físico en el que partículas, gotas, burbujas o materiales se unen al entrar en contacto) inducida por la presión capilar desestabilizante. Capturar esta microfísica requiere formular sistemas de ecuaciones diferenciales altamente no lineales que acoplan el balance de masa con la evolución de la textura del fluido[7], lo que tradicionalmente ha relegado el uso de estas herramientas a costosos e intuitivos entornos de software de escritorio con curvas de aprendizaje sumamente pronunciadas para investigadores y estudiantes.

Con el propósito de mitigar esta brecha tecnológica, este trabajo de grado detalla el diseño, la fundamentación físico-matemática y la implementación de la aplicación `https://foamflowsimulation2d.vercel.app/`, una plataforma web bidimensional para la simulación interactiva y el análisis numérico del flujo de espuma en tiempo real. Al trasladar el núcleo de cómputo intensivo directamente al navegador mediante arquitecturas modernas como los *Web Workers* y acoplarlo a una interfaz gráfica intuitiva basada en React, esta herramienta democratiza el acceso al modelado avanzado de fluidos[8]. La aplicación no solo funciona como un banco de pruebas virtual para evaluar perfiles de concentración y presiones

de inyección sin depender de infraestructuras locales complejas, sino que también se consolida como un entorno didáctico y científico que acerca la física de interfaces a la comunidad académica y profesional.

CAPÍTULO 2

Descripción del Problema

2.1. Formulación

A partir de la problemática expuesta, se identifica una desconexión entre la robustez de los modelos matemáticos actuales y la accesibilidad de las herramientas para su implementación práctica por profesionales del sector. En consecuencia, surge la necesidad de resolver la siguiente interrogante: ¿Es posible desarrollar una aplicación computacional basada en esquemas de elementos finitos que permita simular de manera precisa y amigable el comportamiento de frentes de onda viajero en medios porosos estratificados, integrando parámetros reológicos complejos como los descritos por [7] y [9], tales como la viscosidad efectiva, la movilidad de la espuma, la presión capilar, la concentración de surfactante, la permeabilidad relativa y la evolución dinámica de la textura de la espuma?

2.2. Sistematización

Con el fin de dar respuesta a la pregunta de investigación y cumplir con el desarrollo del proyecto, se plantean las siguientes sub-preguntas que guiarán la metodología:

- ¿Cuáles son las ecuaciones de transporte y los modelos de balance de población que mejor representan la dinámica de la espuma en medios porosos heterogéneos según la literatura técnica [10]?
- ¿Cómo se pueden implementar esquemas numéricos de diferencias finitas o elementos finitos para modelar la transferencia de masa entre capas con distintas propiedades petrofísicas?
- ¿Qué requerimientos de interfaz de usuario y visualización de datos son necesarios para que la herramienta sea considerada funcional y accesible para profesionales no especializados en programación [11]?
- ¿De qué manera se puede validar la precisión de la aplicación mediante la comparación de la velocidad del frente de onda simulado frente a las soluciones analíticas de la teoría de ondas viajeras [7]?

2.3. Objetivos

2.3.1. Objetivo General

Desarrollar una herramienta computacional para la resolución numérica de un sistema acoplado de ecuaciones diferenciales parciales que modela la inyección de espuma en medios porosos estratificados, basada en el modelo de frente viajero de Castrillón Vásquez et al. (2022), implementando esquemas de discretización espacial y temporal que permitan evaluar estabilidad, convergencia y validación de la velocidad teórica del frente

2.3.2. Objetivos Específicos

1. Desarrollar una primera versión de la aplicación para simular la inyección de espuma en un medio poroso homogéneo, utilizando un esquema numérico basado en diferencias finitas o elementod finitos.

2. Adaptar el modelo computacional para representar un medio poroso de dos capas con diferente permeabilidad, incorporando términos de transferencia de masa entre capas y efectos difusivos en la simulación.

3. Implementar en el código la expresión teórica de la velocidad del frente viajero propuesta en el artículo de referencia y comparar los resultados numéricos obtenidos con dicha formulación.

2.4. Justificación

El desarrollo de esta aplicación se justifica por su capacidad para facilitar el trabajo de profesionales de la salud e investigadores, permitiendo que la Pontificia Universidad Javeriana Cali contribuya significativamente a la resolución de problemas complejos con un enfoque humano y técnico. En primer lugar, el proyecto impacta en el ODS 3: Salud y bienestar, al simplificar modelos matemáticos aplicables en tratamientos como la escleroterapia, donde predecir el comportamiento de fluidos es vital para la seguridad del paciente [7]. Al ofrecer una interfaz intuitiva, la Universidad aporta una herramienta que permite a los especialistas médicos optimizar procedimientos clínicos sin necesidad de ser expertos en programación científica [11].

En segundo lugar, este trabajo fortalece la contribución de la institución al ODS 6: Agua limpia y saneamiento, proporcionando a los investigadores ambientales una solución accesible para modelar la limpieza de acuíferos y suelos contaminados [2]. Al reducir las barreras técnicas de los softwares tradicionales, la Universidad democratiza el acceso a simulaciones de alta fidelidad [7], permitiendo que científicos de diversas disciplinas validen sus teorías de forma eficiente. De esta manera, la Javeriana Cali reafirma su compromiso con la excelencia académica y la responsabilidad social, impulsando el progreso en la protección de recursos hídricos y la salud pública mediante la innovación tecnológica [12, 13].

CAPÍTULO 3

Componente Teórico

3.1. Marco Teórico

El flujo multifásico requiere la comprensión de la interacción entre la arquitectura del medio y la naturaleza de los fluidos inyectados.

3.1.1. El Medio Poroso

La porosidad ϕ se define como la relación entre el volumen de vacíos y el volumen total del medio, mientras que la permeabilidad k caracteriza la capacidad del material para permitir el flujo a través de su matriz sólida. En medios estratificados, estas propiedades pueden variar entre capas, generando fenómenos de transferencia de masa y efectos difusivos entre ellas. La descripción matemática de estos procesos se fundamenta en modelos de flujo en medios porosos, tales como la ley de Darcy y modelos extendidos para flujo de espuma [9].

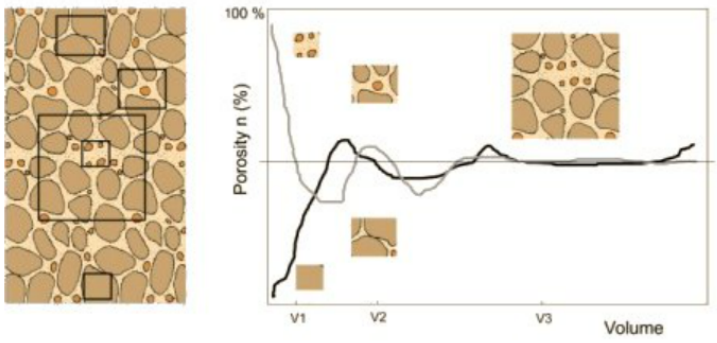


Figura 3.1: Volumen elemental representativo sobre una superficie porosa. Imagen tomada de [2].

Así como se muestra en la Figura 2.1, al momento de medir la porosidad del medio, el Volumen Elemental Representativo (VER) juega un papel importante [2]. De acuerdo con la definición clásica, el VER corresponde al volumen mínimo que, aun siendo seleccionado de manera aleatoria, conserva las características de porosidad del volumen total del medio. A

partir de este volumen representativo, al incrementar el tamaño de la muestra, la porosidad debe mantenerse aproximadamente constante.

Como se menciona en [2], los volúmenes muy pequeños, como V_1 y V_2, no logran caracterizar adecuadamente el medio, ya que no representan su comportamiento global. A partir de un volumen mayor, como V_3, la porosidad tiende a estabilizarse y puede considerarse representativa del material[2].

Este concepto es relevante cuando se analiza la heterogeneidad entre capas, e incluso dentro de una misma capa en diferentes secciones. Si el porcentaje de porosidad ϕ de cada capa tiende a mantenerse relativamente constante dentro de su volumen representativo, la pérdida de solución y el desplazamiento entre capas pueden modelarse de manera más controlada en el proceso de inyección de espuma. Sin embargo, cuando el volumen considerado es demasiado grande, vuelve a evidenciarse la heterogeneidad natural del medio, lo que puede afectar la uniformidad del frente de inyección. Para entender el flujo multifásico en medios porosos, es necesario comprender la interacción entre la arquitectura del medio y la naturaleza de los fluidos inyectados. Factores como la porosidad ϕ, la permeabilidad k de cada capa, la saturación S_α de las fases presentes y las propiedades reológicas de la espuma influyen directamente en el comportamiento del sistema. En particular, la heterogeneidad del medio y la posible estratificación generan diferencias en la distribución de presiones y velocidades, lo que afecta la eficiencia de barrido del fluido inyectado.

A partir de estas propiedades, es posible analizar distintos fenómenos físicos que ocurren dentro del medio poroso, no solo desde el punto de vista del flujo, sino también desde su comportamiento dinámico frente a perturbaciones.

La propagación de ondas en medios porosos saturados puede describirse mediante la teoría de Biot, la cual considera el acoplamiento entre la matriz sólida y la fase fluida, permitiendo la existencia de onda de corte [12]. Aunque este proyecto se enfoca en inyección de espuma, dichos fundamentos aportan al entendimiento general del comportamiento mecánico de un medio poroso en general.

Desde el punto de vista numérico, la resolución del problema requiere la discretización de las ecuaciones diferenciales gobernantes. El método de elementos finitos (FEM) permite obtener una formulación variacional del problema y aproximar la solución en dominios bidimensionales mediante la partición del dominio en elementos discretos [14]. Alternativamente, el método de diferencias finitas (FDM) puede emplearse para la implementación computacional del modelo en configuraciones estructuradas.

Así mismo se plantea integrar todos los conceptos y herramientas mencionadas en el desarrollo de una aplicación orientada a simulaciones bidimensionales de inyección de espuma en medios porosos estratificados. En esta etapa introductoria no se profundiza en la lógica interna del modelo, sino en el hecho de que su implementación se apoyará en herramientas de desarrollo web y cómputo numérico que permitan integrar el procesamiento matemático basado en los módelos dados en [15, 16, 17] con una interfaz accesible. La arquitectura considerará un entorno de programación adecuado para la resolución numérica y su despliegue en un servidor, garantizando portabilidad y disponibilidad del aplicativo.

En este contexto, resulta necesario profundizar en el comportamiento específico de la espuma como agente de desplazamiento, particularmente cuando el medio presenta estratificación.

3.1.2. Flujo de espuma en dos capas

En el caso específico del flujo de espuma en dos capas, se ha demostrado la posibilidad de formación de un frente tipo *traveling wave*, cuya velocidad puede aproximarse como un promedio ponderado de las velocidades individuales de cada capa bajo ciertas hipótesis simplificadas [7]. Este tipo de modelado permite estimar tiempos característicos de estabilización y analizar la interacción entre capas con diferente permeabilidad.

Para comprender la estrategia numérica empleada en la discretización y simulación del frente de espuma, es necesario presentar el modelo físico–matemático que lo sustenta. La formulación parte de una simplificación del modelo cinético completo, adoptando una representación unidimensional bajo la hipótesis de equilibrio local para la textura de la espuma, e incorporando el acoplamiento entre capas mediante términos de intercambio de masa.

Se considera un dominio conformado por dos capas paralelas separadas una distancia d como se puede ver en la Figura 3.1, donde la capa superior posee permeabilidad k_1 y la inferior k_2. El modelo conduce a un sistema de ecuaciones diferenciales parciales acopladas que describen la dinámica del flujo, siguiendo la evolución de la saturación de agua S_{w_i} y la textura de la espuma n_{D_i} en cada capa $i \in \{1, 2\}$. Esta formulación resulta fundamental para estudiar cómo la heterogeneidad del medio y las interacciones no lineales entre fases influyen en la eficiencia de barrido del proceso.

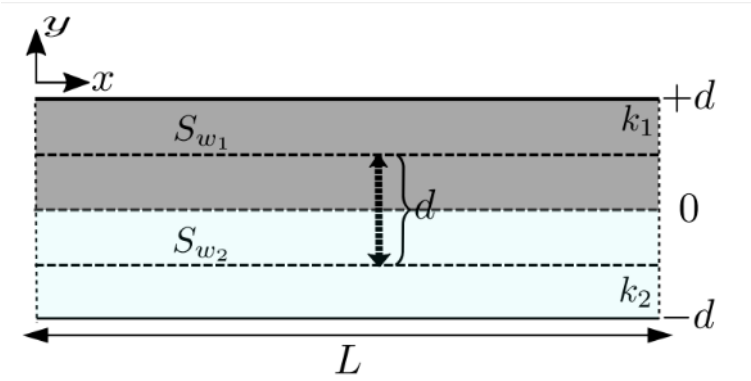


Figura 3.2: Dominio numérico tomada de [3].

Con base en los conceptos anteriores, es pertinente revisar algunos trabajos que han abordado el problema desde diferentes enfoques, tanto teóricos como numéricos.

3.1.3. Trabajos Relacionados

El estudio del flujo de espuma en medios porosos ha evolucionado desde descripciones empíricas hasta modelos mecanísticos que buscan predecir la dinámica del frente de desplazamiento. Un pilar fundamental en esta área es el modelo de *balance de población*, introducido por Kovscek et al. [9], el cual describe la evolución de la textura de la espuma mediante procesos de generación y coalescencia de laminillas en medios como la arenisca Boise [9]. Este enfoque ha permitido diferenciar los comportamientos de la espuma en estados transitorios y estacionarios, sentando las bases para las predicciones mecanísticas en multidimensiones [9, 18].

La literatura científica distingue principalmente dos aproximaciones para modelar la textura: los modelos de equilibrio local (LE) y los modelos cinéticos. Los modelos LE asumen que la textura de la espuma se ajusta instantáneamente a las condiciones del entorno, lo que facilita el estudio de soluciones tipo *traveling wave* y la estabilidad del frente en medios estratificados [7, 19]. Por otro lado, investigaciones como las de Cedro et al. [20] exploran la complejidad de los modelos cinéticos cuando la tasa de generación de burbujas no es inmediata [20].

En cuanto a la heterogeneidad del medio, el impacto de la estratificación es un factor crítico para la eficiencia del proceso de recobro. Zitha et al. [17] utilizaron tomografía computarizada para demostrar cómo la arquitectura de las capas y la velocidad de flujo influyen en

la movilidad de la espuma y su capacidad de actuar como agente divergente en formaciones estratificadas [17, 21]. Recientemente, Castrillón et al. [16, 22] derivaron soluciones analíticas para el frente de onda viajero en medios de dos capas, estableciendo que la velocidad global del frente puede modelarse como un comportamiento acoplado entre los estratos [16, 1]. Estos estudios se han extendido incluso a medios fracturados, donde se analiza el frente de onda considerando la fractura como una capa delgada de alta permeabilidad [22].

Desde la perspectiva del cómputo numérico, la resolución de estos sistemas requiere esquemas robustos debido a la fuerte no linealidad de las ecuaciones de transporte y los efectos de flujo cruzado (*crossflow*) [23]. Se han propuesto algoritmos específicos para resolver el flujo bifásico utilizando métodos de elementos finitos híbridos mixtos y volúmenes finitos [24, 14]. Además, trabajos contemporáneos han integrado técnicas de cuantificación de incertidumbre y análisis de sensibilidad para mejorar la estimación de parámetros a partir de experimentos de laboratorio, garantizando un escalado más fiable hacia aplicaciones de campo [25, 26].

CAPÍTULO 4

Modelamiento Matemático

En este capítulo se presenta el modelo matemático que describe la inyección de espuma en un medio poroso estratificado bidimensional. El sistema se construye a partir de los principios fundamentales de conservación de masa [6], la ley empírica de Darcy [16] y un modelo cinético mecanicista para la evolución de la textura de la espuma [15, 27]. Este enfoque continuo de base física permite un análisis analítico riguroso de la dinámica de los frentes de onda en el sistema multifásico.

4.1. Ecuaciones gobernantes

El flujo inmiscible de dos fases (agua y gas) en un medio poroso saturado se rige por la ley de conservación de la masa [20]. Denotando por $S_w(x, z, t)$ y $S_g(x, z, t)$ a las saturaciones de agua y gas, respectivamente, y considerando un medio con porosidad ϕ, la ecuación de continuidad para la fase acuosa está dada por:

$$\phi \frac{\partial S_w}{\partial t} + \nabla \cdot u_w = 0, \tag{4.1}$$

donde u_w representa la velocidad superficial de la fase acuosa. Dado que el sistema se asume completamente saturado, se cumple la restricción topológica $S_w + S_g = 1$.

Para la dinámica de la espuma, modelada como un conjunto de burbujas de gas separadas por lamelas, se emplea una ecuación de balance de población. Definimos la textura de la espuma adimensional $n_D(x, z, t)$ como la concentración normalizada de burbujas. La ley de transporte toma la forma:

$$\phi \frac{\partial (S_g n_D)}{\partial t} + \nabla \cdot (u_g n_D) = \phi S_g \Phi(S_w, n_D), \tag{4.2}$$

donde u_g es la velocidad superficial de la fase gaseosa. El término fuente $\Phi(S_w, n_D)$ representa la tasa neta de generación y coalescencia de burbujas, gobernada por el modelo cinético lineal dado en [15, 1, 16, 22]:

$$\Phi(S_w, n_D) = K_c \left(n_D^{LE}(S_w) - n_D \right), \tag{4.3}$$

siendo K_c el coeficiente cinético de coalescencia y $n_D^{LE}(S_w)$ la textura de la espuma en equilibrio local [7], la cual se activa suavemente a partir de una saturación de agua crítica S_w^*:

$$n_D^{LE}(S_w) = \begin{cases} \tanh\left(A(S_w - S_w^*)\right), & S_w > S_w^*, \\ 0, & S_w \leq S_w^*, \end{cases} \tag{4.4}$$

con A siendo un parámetro de escalamiento del modelo.

El sistema fluidodinámico se acopla a través de la ley de Darcy generalizada. Definimos la velocidad superficial total del sistema como el vector $u = u_w + u_g$. Imponiendo la condición de incompresibilidad, la divergencia de la velocidad total es nula en todo el dominio:

$$\nabla \cdot u = 0. \tag{4.5}$$

Las velocidades de cada fase se expresan utilizando el flujo fraccional de agua f_w, la movilidad del gas λ_g y el gradiente de la presión capilar P_c:

$$u_w = u f_w + k\lambda_g f_w \nabla P_c, \tag{4.6}$$

donde k corresponde al tensor de permeabilidad absoluta del medio. La acción física de la espuma altera la dinámica reduciendo la permeabilidad relativa del gas k_{rg} a través del factor de reducción de movilidad MRF:

$$k_{rg}(S_w, n_D) = \frac{k_{rg}^0(S_w)}{MRF(n_D)}, \quad \text{con} \quad MRF(n_D) = \beta n_{max} n_D + 1, \tag{4.7}$$

estableciendo la relación matemática exacta que cuantifica el control de la movilidad del gas inyectado.

4.2. Supuestos

El modelo continuo formulado descansa sobre los siguientes supuestos estructurales y termodinámicos, basados en [16, 1]:

- **Incompresibilidad e inmiscibilidad:** Las fases acuosa y gaseosa son tratadas como fluidos incompresibles e inmiscibles en condiciones isotérmicas, sin un intercambio directo de masa entre el líquido y el gas.

- **Estratos homogéneos por secciones:** El dominio presenta heterogeneidad global mediante una configuración estratificada. Específicamente, el dominio Ω está constituido por subdominios rectangulares dentro de los cuales la porosidad ϕ y la permeabilidad absoluta k mantienen valores estrictamente constantes.
- **Concentración química invariante:** Se asume que el surfactante disuelto en la fase acuosa posee una concentración constante y superior a la concentración micelar crítica, permitiendo desacoplar y omitir una ecuación diferencial de advección-difusión adicional para su transporte.
- **Presión capilar univaluada:** La función de presión capilar dependiente de la interfase agua-gas carece de efectos de histéresis, comportándose analíticamente como un mapeo continuo y estrictamente decreciente respecto a la saturación de agua.

4.3. Condiciones de frontera

Para configurar un problema de valor inicial y de contorno bien planteado, se define el dominio computacional como un espacio cerrado $\Omega = [0, L] \times [-d, d_{max}]$ basados en [16]. Las fronteras $\partial\Omega$ se analizan en tres regiones topológicas distintas.

1. **Frontera de inyección en la coordenada longitudinal ($x = 0$):** Se establecen condiciones de frontera tipo Dirichlet para la inyección continua de los fluidos. Se prescriben estados iniciales fijos para todo $z \in [-d, d_{max}]$ y tiempo $t > 0$:

$$S_w(0, z, t) = S_w^-, \quad n_D(0, z, t) = n_D^-, \quad u(0, z, t) \cdot \hat{n} = u_{inj}, \tag{4.8}$$

donde $\hat{n}$ denota el vector normal unitario apuntando hacia el exterior del dominio y u_{inj} la velocidad volumétrica de inyección.

2. **Frontera de producción longitudinal ($x = L$):** Bajo el supuesto de que la longitud del medio es lo suficientemente extensa, el flujo alcanza un estado asintóticamente desarrollado. Esto exige imponer una condición de frontera tipo Neumann homogénea:

$$\frac{\partial S_w}{\partial x}(L, z, t) = 0, \quad \frac{\partial n_D}{\partial x}(L, z, t) = 0. \tag{4.9}$$

3. **Fronteras transversales impermeables ($z = -d$ y $z = d_{max}$):** Para representar el confinamiento de las capas geológicas, la frontera superior e inferior imponen restricciones de

flujo nulo. En dichas regiones, se asignan velocidades normales cero y derivadas direccionales nulas (condición de Neumann):

$$\frac{\partial S_w}{\partial z}(x, z, t) = 0, \quad u_w \cdot \hat{n} = 0, \quad u_g \cdot \hat{n} = 0. \tag{4.10}$$

Este conjunto de condiciones garantiza la conservación y permite estabilizar el sistema diferencial hacia el perfil de frente de onda viajero esperado en medios estratificados.

CAPÍTULO 5

Discretización y Desarrollo del Código

5.1. Método numérico

El modelo numérico describe la evolución de la saturación de agua $S_w(x, z, t)$ en un dominio bidimensional, donde x corresponde a la dirección longitudinal y z a la vertical. El sistema considera dos capas superpuestas (capas 1 y 2), cada una con propiedades potencialmente distintas como porosidad ϕ, permeabilidad k y velocidad superficial u.

La discretización se realiza sobre una malla cartesiana uniforme, mientras que la evolución temporal se resuelve mediante un esquema explícito.

En forma continua, la ecuación implementada corresponde a una ley de conservación con un término difusivo capilar:

$$\phi \frac{\partial S_w}{\partial t} \;=\; -\frac{\partial}{\partial x}\big(u\, f_w(S_w, n_D)\big) \;+\; \nabla \cdot \big(D_c(S_w, n_D)\, \nabla S_w\big), \tag{5.1}$$

La textura de espuma n_D se modela como una variable en equilibrio local, calculada algebraicamente a partir de S_w:

$$n_D = \begin{cases} \tanh\big(A(S_w - S_w^*)\big), & S_w > S_w^*, \\ 0, & S_w \le S_w^*. \end{cases} \tag{5.2}$$

La formulación constitutiva se implementa de forma modular en el archivo `constitutive.js`. En particular, el flujo fraccional f_w se obtiene a partir de las movilidades relativas y de una función de reducción de movilidad por espuma (MRF):

Fragmento (flujo fraccional y equilibrio local).

```
export function nD_LE(Sw, params) {
  const { Sw_star, A } = params;

  if (Sw > Sw_star) {
    return Math.tanh(A * (Sw - Sw_star));
  }
  return 0.0;
}
```

```
9
10 export function fw(Sw, nD, k_b, params) {
11   const { mu_w, mu_g } = params;
12
13   const lw = (k_b * krw(Sw, params)) / mu_w;
14   const lg = (k_b * krg0(Sw, params)) / MRF(nD, params) / mu_g;
15
16   return lw / (lw + lg + 1e-30);
17 }
```

La difusión capilar D_c se calcula a partir de la derivada de la presión capilar $P_c(S_w)$, de modo que el término difusivo queda directamente ligado a la física del problema:

Fragmento (difusión capilar).

```
1 export function D_cap(Sw, nD, k_b, phi_b, params) {
2   const { mu_g } = params;
3
4   const lg = (k_b * krg0(Sw, params)) / MRF(nD, params) / mu_g;
5
6   return -lg * fw(Sw, nD, k_b, params)
7     * dPc_dSw(Sw, k_b, phi_b, params);
8 }
```

5.2. Discretización espacial

El dominio se divide en N_x intervalos en $x \in [0, L]$ y N_z intervalos en $z \in [-d, d]$, con:

$$\Delta x = \frac{L}{N_x}, \qquad \Delta z = \frac{2d}{N_z}. \tag{5.3}$$

Las variables se almacenan en arreglos unidimensionales utilizando el mapeo:

$$\texttt{idx} = j(N_x + 1) + i. \tag{5.4}$$

Fragmento (malla e indexación).

```
1 const dx = L / Nx;
2 const dz = (2 * d) / Nz;
```

```
3
4  const numCells = (Nz + 1) * (Nx + 1);
5
6  const Sw     = new Float64Array(numCells);
7  const nD     = new Float64Array(numCells);
8  const phi_2d = new Float64Array(numCells);
9  const k_2d   = new Float64Array(numCells);
10 const u_2d   = new Float64Array(numCells);
11
12 for (let j = 0; j <= Nz; j++) {
13   const z_val = -d + j * dz;
14   const isL1  = z_val > 0.0;
15
16   for (let i = 0; i <= Nx; i++) {
17     const idx = j * (Nx + 1) + i;
18
19     phi_2d[idx] = isL1 ? phi1 : phi2;
20     k_2d[idx]   = isL1 ? k1   : k2;
21     u_2d[idx]   = isL1 ? u1   : u2;
22
23     Sw[idx] = Sw_plus;
24   }
25 }
```

Término advectivo

La derivada $\partial_x(uf_w)$ se aproxima mediante un esquema tipo *upwind*:

$$\frac{(uf_w)_i - (uf_w)_{i-1}}{\Delta x}. \tag{5.5}$$

```
1  export function advectionUpwindX(q, u, Nx, Nz, dx) {
2    const adv = new Float64Array((Nz + 1) * (Nx + 1));
3
4    for (let j = 0; j <= Nz; j++) {
5      for (let i = 1; i < Nx; i++) {
```

```
      const idx = j * (Nx + 1) + i;
      const idx_prev = j * (Nx + 1) + (i - 1);

      adv[idx] = (u[idx] * q[idx]
        - u[idx_prev] * q[idx_prev]) / dx;
    }
  }
  return adv;
}
```

Término difusivo

El término difusivo se discretiza mediante diferencias centradas usando media armónica:

```
const Dx_next = 2.0 * D[idx_next] * D[idx]
  / (D[idx_next] + D[idx] + 1e-30);

const Dx_prev = 2.0 * D[idx] * D[idx_prev]
  / (D[idx] + D[idx_prev] + 1e-30);

const flux_x_next = Dx_next
  * (S[idx_next] - S[idx]) / dx;

const flux_x_prev = Dx_prev
  * (S[idx] - S[idx_prev]) / dx;

div[idx] += (flux_x_next - flux_x_prev) / dx;
```

Condiciones de frontera

Se imponen condiciones Dirichlet en la entrada y Neumann en los demás bordes:

```
// Entrada
for (let j = 0; j <= Nz; j++) {
  const idx0 = j * (Nx + 1);
  Sw[idx0] = Sw_minus;
```

```
  nD[idx0] = nD_inj;
}

// Salida
for (let j = 0; j <= Nz; j++) {
  const idxL  = j * (Nx + 1) + Nx;
  const idxL1 = j * (Nx + 1) + Nx - 1;

  Sw[idxL] = Sw[idxL1];
  nD[idxL] = nD[idxL1];
}
```

5.3. Discretización temporal

La integración en el tiempo se realiza mediante Euler explícito:

$$S_w^{n+1} = S_w^n + \Delta t(\cdots) \tag{5.6}$$

Fragmento (bucle temporal).

```
const adv  = advectionUpwindX(Fw, u_2d, Nx, Nz, dx);
const diff = divDiffusion(Sw, Dc, Nx, Nz, dx, dz, thetaS);

const Sw_new = new Float64Array(numCells);

for (let idx = 0; idx < numCells; idx++) {
  const rhs = (-adv[idx] + diff[idx]) * w_area[idx];

  Sw_new[idx] = Math.max(
    Swc + 1e-6,
    Math.min(
      Sw[idx] + dt * rhs / (M_phi[idx] + 1e-30),
      1 - Sgr - 1e-6
    )
  );
}
```

```
17
18 for (let idx = 0; idx < numCells; idx++) {
19   Sw[idx] = Sw_new[idx];
20   nD[idx] = Math.max(0,
21     Math.min(nD_LE(Sw_new[idx], params), 1));
22 }
```

5.4. Estructura del código

El núcleo numérico se implementa en JavaScript dentro de una arquitectura modular:

- Capa de servicio (API y WebSocket)
- Estado de simulación
- Núcleo numérico

Worker

```
1 worker = new Worker(path.resolve(__dirname,
2   '../simulation/worker.js'), {
3   workerData: { params: state.params }
4 });
5
6 worker.on('message', (msg) => {
7   if (msg.type === 'frame') {
8     broadcast(msg);
9   }
10 });
```

5.5. Pseudocódigo

```
Inicializar malla y propiedades
Inicializar variables

Para cada paso:
        Calcular t rminos f sicos
        Actualizar variables
        Aplicar condiciones
        Registrar resultados
```

CAPÍTULO 6

Construcción de la Aplicación

Este capítulo describe el proceso de construcción del software *Foam Flow Web LE 2D*, desde su arquitectura hasta su uso operativo. El objetivo es documentar las decisiones técnicas que permitieron implementar una herramienta interactiva para simulación numérica bidimensional de flujo de espuma, con visualización científica en tiempo real y exportación de resultados.

6.1. Arquitectura del sistema

La aplicación se construyó con una arquitectura cliente-servidor desacoplada[8]. La capa cliente se implementó como una SPA en React para la interacción de usuario y la visualización de resultados, mientras que la capa servidor se implementó con Node.js y Express para gestionar la lógica de simulación, el estado de corrida y la exposición de servicios REST/WS.

6.1.1. Visión general de la arquitectura

La arquitectura integra cinco bloques técnicos que trabajan de forma coordinada, de la siguiente forma: En el extremo de usuario se ubica el cliente web construido con React y Vite, responsable de la interfaz, la captura de parámetros y la presentación de resultados. Sobre el cliente opera un estado global con Zustand, que sincroniza controles, progreso y datos históricos para las gráficas. Del lado servidor, una API REST en Node.js/Express recibe comandos de ejecución y gestiona la exportación de datos[8]. En paralelo, un canal WebSocket transmite progreso y *frames* de la simulación en tiempo real. Finalmente, el motor numérico concentra la integración temporal, los operadores espaciales y las relaciones constitutivas del modelo.

6.1.2. Diseño de módulos del cliente

En el cliente se adoptaron responsabilidades separadas para reducir acoplamiento y facilitar mantenimiento. El componente **ParameterPanel** concentra la captura y validación inicial de parámetros físicos y numéricos, mientras que **SimControls** administra el ciclo de

vida de la corrida (inicio, pausa, reanudación, detención y exportación). La comunicación con el backend se encapsula en **useSimulation**, que coordina solicitudes REST y suscripción WebSocket. Como soporte transversal, **simStore** mantiene el estado central y el histórico requerido para la visualización[28]. Sobre esta base se construyen los componentes gráficos de mapas 2D, perfiles 1D y métricas dinámicas del frente.

6.1.3. Diseño de módulos del servidor

En el servidor, la organización se divide entre capa de servicio y núcleo de simulación. La capa de rutas expone **routes/simulation** para los comandos de control (`start`, `pause`, `resume`, `stop`) y **routes/export** para la generación de archivos CSV. El estado vivo de la corrida se centraliza en **store/simState**, que conserva parámetros, progreso e históricos. A partir de ese estado, **simulation/worker** ejecuta el bucle temporal de cálculo y se apoya en **simulation/operators** para los términos espaciales y en **simulation/constitutive** para las relaciones físicas del modelo.

6.1.4. Comunicación y flujo de datos

El flujo de datos se desarrolla en cuatro etapas:

1. El usuario define parámetros en interfaz y solicita la ejecución.
2. El cliente envía comandos al backend vía REST.
3. El motor numérico actualiza estado interno y publica eventos por WebSocket.
4. El cliente rehidrata estado local y actualiza visualizaciones de forma incremental.

Este enfoque permite mantener baja latencia de interacción y separa claramente la lógica numérica de la capa de presentación.

6.1.5. Despliegue productivo en Vercel y Render

Para la operación en línea se adoptó un despliegue desacoplado: el frontend se publicó en Vercel como SPA de React/Vite y el backend se ejecutó en Render como servicio Node.js/Express con API REST y canal WebSocket. Esta separación permitió escalar y actualizar cada capa de forma independiente, manteniendo la interfaz disponible aun cuando el servidor de cálculo estuviera en proceso de reinicio o despliegue.

6.1.5.1. Cualidades técnicas de Vercel en el proyecto

En el proyecto, Vercel aportó despliegue continuo desde repositorio, lo que facilitó ciclos rápidos de corrección y validación académica. Su distribución por CDN mejoró la entrega de recursos estáticos y la respuesta inicial de la interfaz. También resultó clave la gestión de variables de entorno para parametrizar la URL del backend sin alterar el código fuente. Además, al operar sobre HTTPS por defecto, la plataforma habilitó el uso de `wss://` para una comunicación WebSocket segura.

6.1.5.2. Cualidades técnicas de Render en el proyecto

Render se utilizó para alojar el motor de cálculo y la API porque permite ejecutar servicios Node.js persistentes y exponer endpoints de control de simulación con estabilidad. Su soporte de WebSocket hizo posible transmitir progreso y *frames* en tiempo real. Asimismo, su modelo de variables de entorno y puerto dinámico es compatible con el uso de `process.env.PORT` implementado en el servidor. Por último, la disponibilidad de logs operativos ayudó a diagnosticar incidencias y a documentar evidencia técnica durante pruebas experimentales.

6.1.5.3. Integración Vercel-Render para ejecución de simulación

La conexión entre ambos servicios se implementó de forma paramétrica. En Vercel se definieron las variables de entorno del frontend: $\texttt{VITE}_{API_{URL}}$ como base REST y, de forma opcional,

6.1.5. Diagramas de arquitectura y análisis

6.1.5.4. Diagrama de contexto del sistema

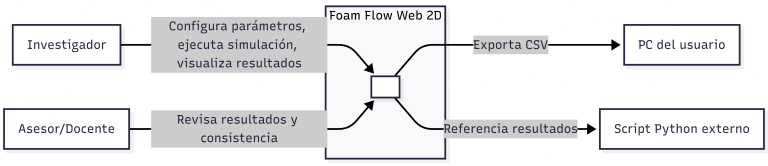


Figura 6.1: Contexto del sistema Foam Flow Web 2D.

Este diagrama delimita frontera de sistema, actores principales y sistemas externos involucrados (usuario, asesor, herramienta de referencia y entorno de ejecución).

6.1.5.5. Diagrama de despliegue

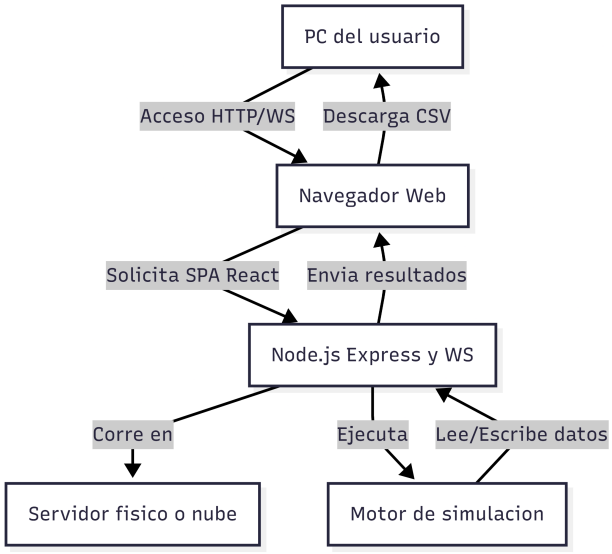


Figura 6.2: Despliegue fisico/logico de cliente y servidor..

Se documenta la distribución de componentes en nodos de ejecución y los protocolos de comunicación (HTTP y WebSocket).

6.1.5.6. Diagrama de casos de uso

Permite justificar el alcance funcional implementado y su trazabilidad con los requisitos del software.

6.1.5.7. Diagrama C4 (nivel contexto)

Resume la posición del sistema en su ecosistema y aclara interacciones de alto nivel con actores y sistemas externos.

6.1.5.8. Diagrama C4 (nivel contenedores)

Evidencia separación de responsabilidades entre frontend, backend y procesamiento numérico.

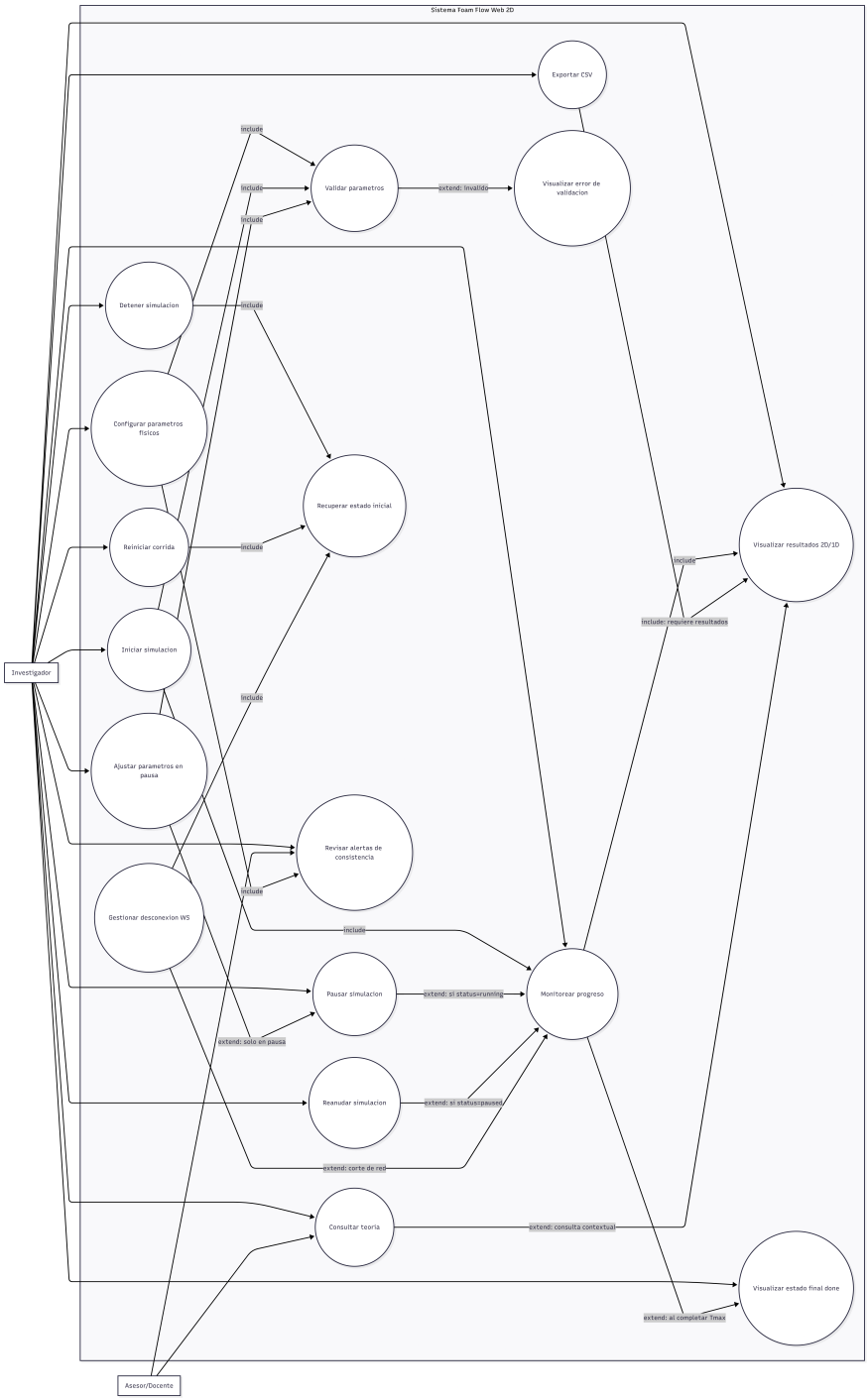


Figura 6.3: Casos de uso del sistema y relaciones include/extend.

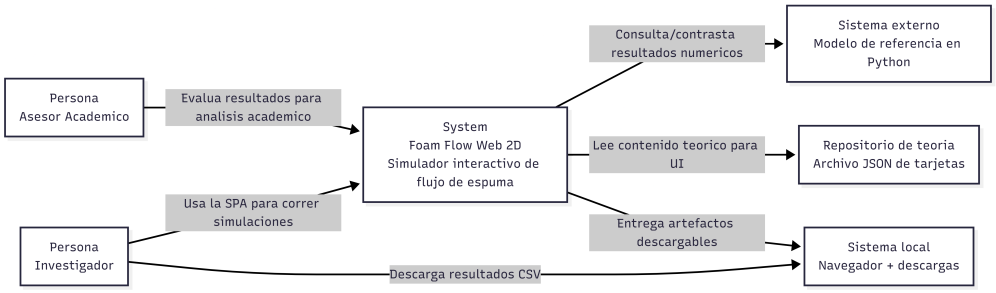


Figura 6.4: Diagrama C4 (nivel contexto)

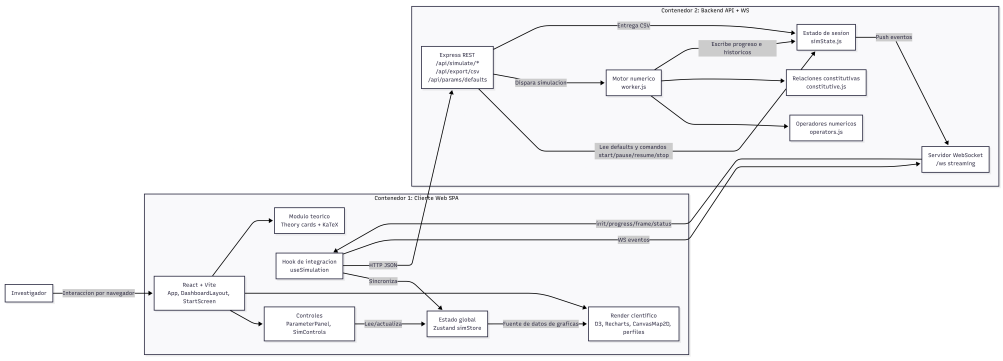


Figura 6.5: C4 nivel contenedores: cliente SPA, API/WS y motor numerico.

6.1.5.9. Diagrama de secuencia de ejecución

Muestra orden temporal de mensajes para inicio de simulación, streaming, pausa/reanudación, detención y exportación.

6.1.6. Especificación de requisitos previa a diagramas

Antes de presentar los diagramas de requisitos, se define la especificación textual para asegurar trazabilidad y calidad documental. Esta especificación se redactó siguiendo nueve criterios de levantamiento de requisitos, aplicados tanto a RF como a RNF.

6.1.6.1. Requisitos funcionales (RF)

La Tabla 6.1 presenta los requisitos funcionales en formato de especificacion tipo IEEE.

6.1.6.2. Requisitos no funcionales (RNF)

La Tabla 6.2 presenta los requisitos no funcionales en formato tipo IEEE con metricas verificables.

6.1.6.3. Restricciones del sistema

La Tabla 6.3 resume las restricciones que condicionan implementacion y alcance.

6.1.6.4. Diagrama de requisitos funcionales

Presenta encadenamiento entre captura de parámetros, ejecución, visualización y exportación de resultados.

6.1.6.5. Diagrama de requisitos no funcionales

Relaciona rendimiento, confiabilidad, mantenibilidad, seguridad y observabilidad con decisiones de arquitectura.

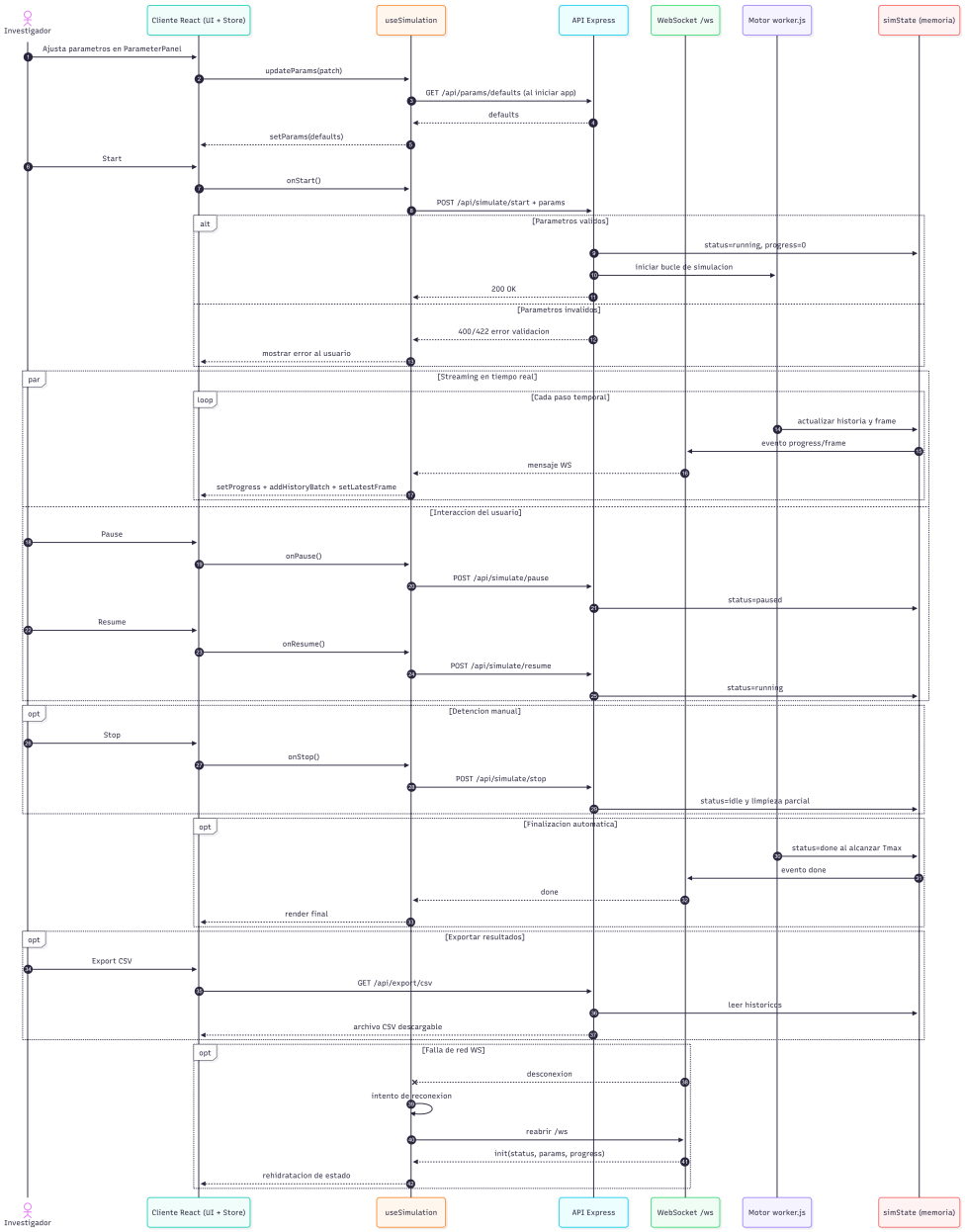


Figura 6.6: Secuencia de interaccion entre usuario, cliente, API, WebSocket y motor numerico.

Cuadro 6.1: Requisitos funcionales (formato tipo IEEE)

ID	Descripcion	Prioridad	Criterio de aceptacion	Verificacion	Trazabilidad
RF-01	Configurar parametros fisicos y numericos antes de la corrida.	Alta	Parametros validos quedan persistidos en estado global y se envian al backend al ejecutar `start`.	Prueba funcional del formulario y traza HTTP.	ParameterPanel, simStore, API `/simulate/start`.
RF-02	Validar rangos, tipos y obligatoriedad de entradas.	Alta	Si existe valor invalido, la corrida no inicia y se informa el error puntual.	Casos positivos y negativos de validacion.	ParameterPanel, validadores cliente/servidor.
RF-03	Iniciar simulacion bidimensional en backend.	Alta	Tras `start`, el estado pasa a `running` y se emiten eventos de progreso/frame por WS.	Prueba de integracion API + WS.	routes/simulation, worker, WebSocket.
RF-04	Gestionar ciclo de vida con `pause`, `resume` y `stop`.	Alta	Cada comando actualiza estado de corrida sin corromper historicos.	Prueba de transiciones de estado.	SimControls, simState, endpoints de control.
RF-05	Visualizar mapas 2D de variables de proceso.	Alta	Con cada frame recibido, el mapa 2D se actualiza sin recarga de pagina.	Prueba funcional de render incremental.	CanvasMap2D, SaturationMap, CrossflowMap.
RF-06	Visualizar perfiles 1D y series temporales.	Media-Alta	Las graficas reflejan historicos acumulados y son coherentes con el estado de simulacion.	Comparacion entre datos del store y datos graficados.	Profiles1D, PressureProfile, FrontDistance.
RF-07	Exportar resultados de simulacion en CSV.	Media-Alta	Al solicitar exportacion, se descarga archivo no vacio con encabezados y registros.	Prueba de endpoint y validacion del CSV.	routes/export, estado historico en servidor.
RF-08	Recuperar estado basico tras reconexion WS.	Media	Al reconectar, se restablece `status/progress/params` sin reinicio manual de la UI.	Prueba de resiliencia con desconexion/reconexion.	useSimulation, WebSocket, simStore.

Cuadro 6.2: Requisitos no funcionales (formato tipo IEEE)

ID	Atributo de calidad	Prioridad	Especificacion	Verificacion
RNF-01	Rendimiento de interfaz	Alta	Tiempo de respuesta de acciones UI menor a 500 ms en condiciones normales de laboratorio.	Medicion con herramientas de rendimiento del navegador.
RNF-02	Latencia de propagacion	Alta	Retardo servidor-cliente para frame visualizado menor o igual a 1 s en red local estable.	Prueba temporal con marcas de tiempo en WS.
RNF-03	Confiabilidad operativa	Alta	Continuidad de sesion ante fallos transitorios mediante reconexion automatica del canal WS.	Ensayos de desconexion controlada y validacion de recuperacion.
RNF-04	Seguridad de entrada	Alta	Toda entrada de API debe validarse en servidor para prevenir ejecucion con datos invalidos.	Pruebas con payloads invalidos y respuestas HTTP 4xx esperadas.
RNF-05	Usabilidad academica	Media-Alta	Usuario objetivo completa el flujo principal tras induccion corta, sin asistencia tecnica permanente.	Prueba guiada con checklist de tareas completadas.
RNF-06	Mantenibilidad	Media-Alta	Cambios en visualizacion o nucleo numerico se realizan sin refactorizacion global.	Evidencia de separacion modular y cambios localizados.
RNF-07	Portabilidad	Media	Compatibilidad con navegadores modernos ES6+ y entorno Node.js LTS.	Matriz de pruebas por navegador y version de runtime.
RNF-08	Observabilidad	Media	Disponibilidad de estado, progreso y eventos para diagnostico durante ejecucion.	Inspeccion de logs/eventos y validacion en interfaz.

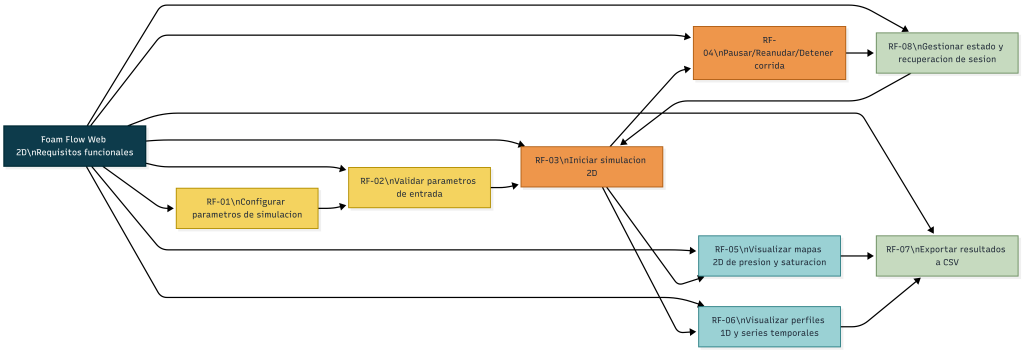


Figura 6.7: Trazabilidad y dependencias entre requisitos funcionales (RF).

Cuadro 6.3: Restricciones del sistema (formato tipo IEEE)

ID	Tipo	Restriccion
RST-01	Tecnologica	El frontend debe conservar compatibilidad con React/-Vite y el backend con Node.js/Express, en coherencia con la base instalada del proyecto.
RST-02	Operativa	La simulacion se ejecuta en recursos de computo de proposito general; no se asume GPU dedicada ni cluster HPC en la version de pregrado.
RST-03	Alcance	La exportacion de los datos oficial de la version actual es CSV; formatos adicionales se consideran trabajo futuro.

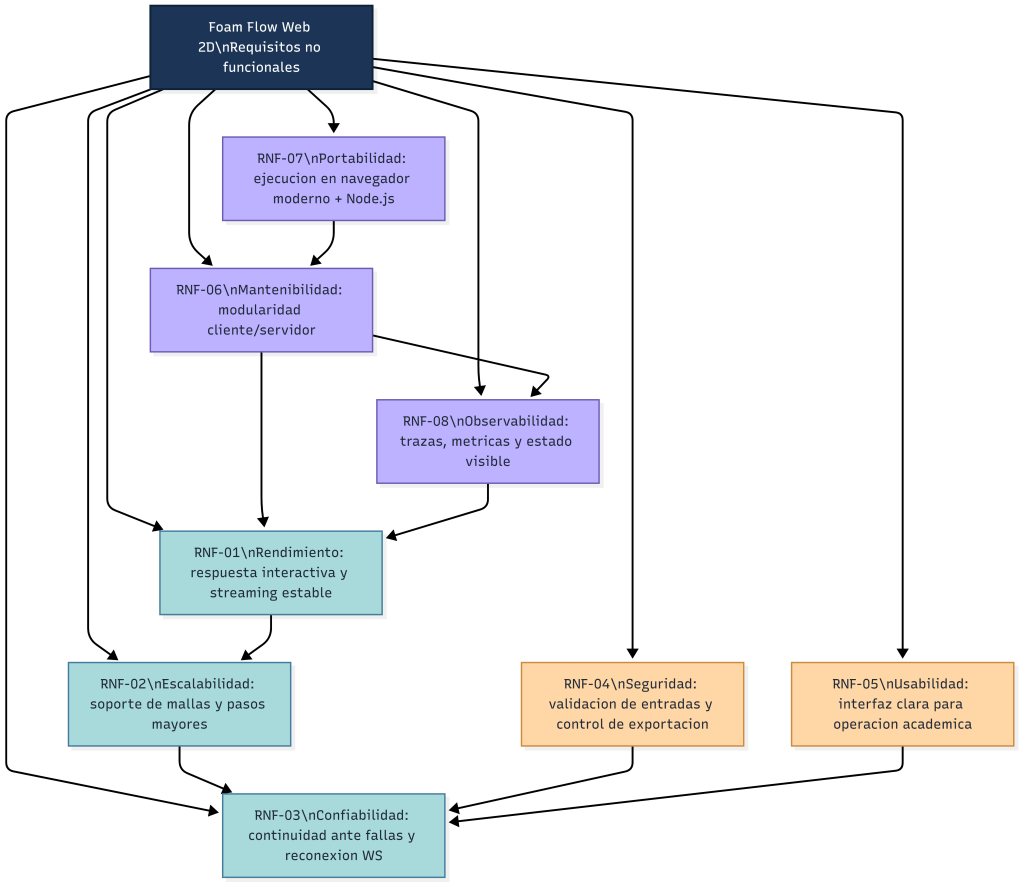


Figura 6.8: Relaciones entre atributos de calidad del sistema (RNF).

6.2. Diseño de la interfaz

El diseño de interfaz se abordó como una capa de operación científica, priorizando legibilidad de variables, control del experimento numérico y retroalimentación continua al usuario.

6.2.1. Principios de diseño adoptados

El diseño de interfaz se apoyó en cuatro principios articulados. Primero, la claridad visual, lograda mediante separación explícita entre panel de parámetros, controles de simulación y área de resultados. Segundo, la coherencia operacional, manteniendo un flujo estable para iniciar, pausar, reanudar y detener corridas sin ambigüedad para el usuario. Tercero, la retroalimentación inmediata, expresada en el estado de corrida, barras de progreso y actualización continua de gráficas. Finalmente, se incorporó soporte académico con tarjetas teóricas que contextualizan variables y fenómenos relevantes.

6.2.2. Mockups de la interfaz

Figura 6.9: Vista principal de la aplicación.

Estas evidencias visuales respaldan decisiones de usabilidad y permiten demostrar alineación entre diseño propuesto y comportamiento implementado.

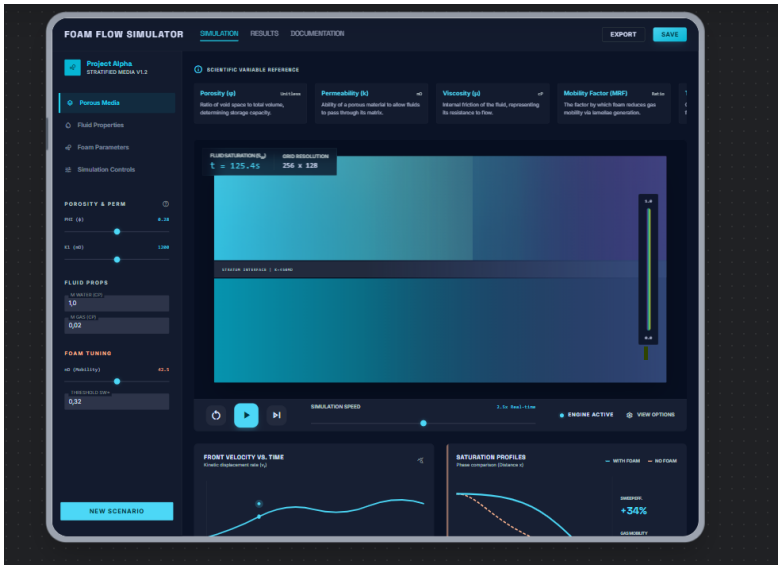


Figura 6.10: Panel de parametros y controles de ejecucion.

6.3. Funcionalidades principales

Las funcionalidades nucleares implementadas se describen a continuación.

6.3.1. Gestión de simulación numérica

La gestión de simulación contempla configuración de parámetros físicos y numéricos, validación previa al inicio y control de ejecución por estados `idle`, `running`, `paused` y `done`. Esta estructura permite que el usuario opere la corrida de manera predecible, evitando transiciones inconsistentes y facilitando la trazabilidad del experimento.

6.3.2. Visualización científica de resultados

En la capa de análisis, la aplicación ofrece mapas 2D para observar la distribución espacial de variables del proceso, perfiles 1D para inspección puntual y series temporales para seguir la dinámica del frente y la evolución de la presión. Esta combinación permite lectura complementaria del fenómeno tanto en espacio como en tiempo.

6.3.3. Persistencia ligera y exportación

La solución incorpora persistencia ligera del estado de simulación en servidor durante la sesión activa, lo que permite continuidad operativa. Ante reconexión de WebSocket se recupera un estado básico para rehidratar la interfaz, y al finalizar una corrida los resultados pueden exportarse en CSV para su postproceso en herramientas externas de análisis.

6.3.4. Manejo de errores y resiliencia

En términos de resiliencia, la aplicación detecta y reporta parámetros inválidos antes del cálculo, maneja desconexión y reconexión del canal WebSocket y mantiene continuidad operativa mediante rehidratación de estado en cliente. Con ello se reduce la pérdida de contexto durante incidentes transitorios de red.

6.4. Manual de usuario

El manual completo se encuentra en el siguiente enlace: https://foamflowsimulation2d.vercel.app/manual_usuario.html

Adicionalmente, se resume el flujo operativo principal para consulta rápida:

1. Ingresar parámetros físicos y numéricos en el panel de configuración.
2. Ejecutar validación y corregir entradas en caso de advertencias.
3. Iniciar simulación y monitorear indicadores de progreso.
4. Usar pausa/reanudación para inspección intermedia de resultados.
5. Finalizar corrida y exportar datos en CSV para análisis externo.

6.4.1. Recomendaciones de uso académico

Para fortalecer el valor académico de los resultados, se recomienda registrar en cada experimento el conjunto exacto de parámetros utilizados, comparar corridas variando un solo parámetro por iteración y adjuntar en anexos las capturas de resultados clave junto con las tablas exportadas. Esta práctica favorece interpretación rigurosa y escalabilidad del análisis.

6.4.2. Consideraciones de reproducibilidad

Para facilitar la reproducibilidad experimental, se incluye el enlace al repositorio con el código fuente de la aplicación: https://github.com/marco2712/Foam-flow-simulation2d.git. Este enlace también se encuentra disponible en el pie de página de la aplicación.

CAPÍTULO 7

Validación del Modelo

7.1. Validación 1D

Se presenta la comparación entre los perfiles 1D extraídos del documento original [1] y los generados por el software desarrollado (app). Ya que las gráficas de [1] fueron cortadas para ajustarse visualmente, el error ha sido medido parametrizando y normalizando las curvas extraídas iterativamente desde un dominio espacial para el eje x entre 0 y 0.25 m. Además, se tuvo en consideración que los Casos 1 y 2 fueron evaluados a $t = 2000$ segundos, mientras que el Caso 3 se evaluó a $t = 4000$ segundos.

Se analiza el error (RMSE) interpolado de las curvas en los tres casos de estudio, junto con los efectos del *viscous crossflow*.

7.1.1. Caso 1: Tendencia de la capa 1 hacia la capa 2 a $t = 2000$ s

En el Caso 1, evaluado a los 2000 segundos, se observa una fuerte tendencia de flujo cruzado viscoso desde la capa 1 hacia la capa 2, lo cual modifica y arrastra el avance del frente.

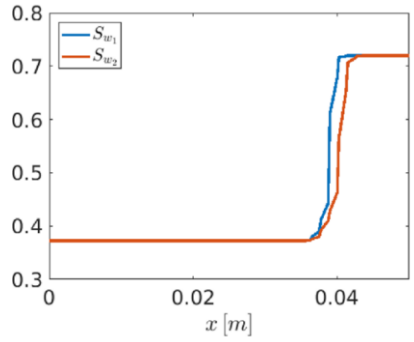


Figura 7.1: Perfil 1D Caso 1 [1]

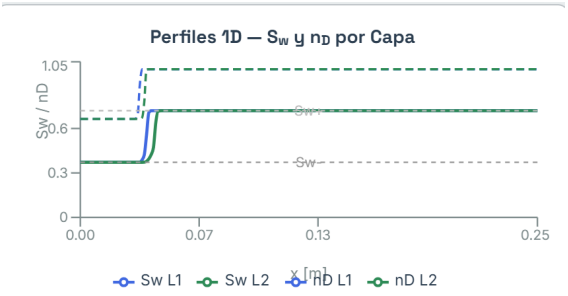


Figura 7.2: Perfil 1D Caso 1 (App)

Resultado del cálculo de error: El Error Cuadrático Medio de la Raíz (RMSE) interpolado directamente sobre la forma de las curvas en escala física ($x \in [0, 0.25]$) es **0.1786**.

A pesar de que las imágenes del paper sufrieron recortes, la normalización de la coordenada espacial en su rango original permite verificar que la simulación captura con precisión

la tendencia fundamental, esto es, la migración de masa y transferencia debido al arrastre viscoso desde la capa de mayor movilidad (capa 1) a la de menor (capa 2).

7.1.2. Caso 2: Sin tendencia (No viscous crossflow) a $t = 2000$ s

En el Caso 2, tabulado igualmente a los 2000 segundos, las viscosidades y parámetros se escogen de tal firma que se suprime el efecto del empuje viscoso vertical predominante entre capas.

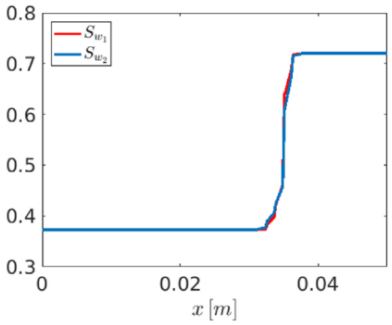


Figura 7.3: Perfil 1D Caso 2 [1]

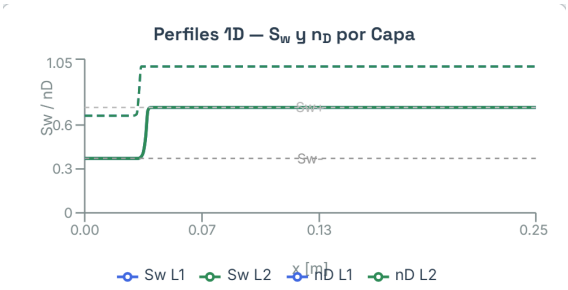


Figura 7.4: Perfil 1D Caso 2 (App)

Resultado del cálculo de error: El Error RMSE interpolado es **0.2168**.

En este escenario, la capa 1 y la capa 2 no experimentan una fuerte transferencia cruzada, avanzando juntas casi uniformemente. La reconstrucción sobre el rango espacial normalizado $[0, 0.25]$ detalla un acoplamiento satisfactorio, lo cual valida que, cuando las fuerzas transientes son equilibradas, el sistema numérico propuesto obedece de manera coherente al comportamiento ideal sin inestabilidades cruzadas.

7.1.3. Caso 3: Tendencia de la capa 2 hacia la capa 1 a $t = 4000$ s

Para visualizar mejor el efecto prolongado de un escenario inverso, el Caso 3 es analizado para un tiempo mayor, $t = 4000$ segundos. Aquí, la tendencia muestra el flujo cruzado viscoso desde la capa 2 hacia la capa 1.

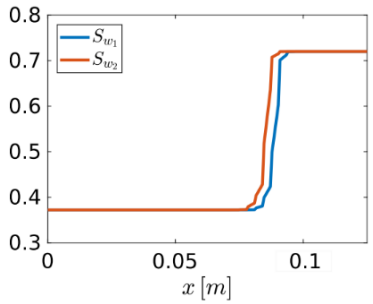


Figura 7.5: Perfil 1D Caso 3 [1]

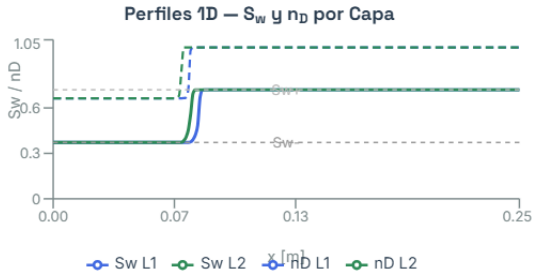


Figura 7.6: Perfil 1D Caso 3 (App)

Resultado del cálculo de error: El Error RMSE de reconstrucción espacial de las curvas es **0.1928**.

La dinámica mostrada por la transferencia ascensional desde la capa 2, la cual tiene ahora influencia en retrasar la invasión en su interfaz, es replicada correctamente frente al esquema analítico con un ajuste de 0.1928. El hecho de que la comparación requiera doblar el tiempo (4000 s) frente a los otros casos confirma que la magnitud del estrangulamiento de los bordes y el acoplamiento inferior rigen los principios físicos correctos dentro de la ventana de dominio 0 a 0.25 de manera robusta.

7.2. Validación 2D

7.2.1. Validación por comparación de frentes Caso Cross Flow [1]

Se comparan frentes extraídos de imágenes ([1] vs Aplicación). Dado que son capturas (fotos) y no necesariamente corresponden exactamente al mismo instante, el indicador principal se basa en **similitud de forma**: se calcula el frente como $x(y/d)$ y el error $e(y) = x_{\text{app}}(y) - x_{\text{paper}}(y)$. Para comparar forma, se alinea con un corrimiento constante Δx (promedio de e) y se analiza el error alineado $e_{\text{forma}}(y) = e(y) - \Delta x$.

Escalas usadas: $x \in [0, 0.25]\,\text{m}$ y $y \in [-d, d]$ con $d = 0.005\,\text{m}$ (por lo tanto $y/d \in [-1, 1]$).

7.2.2. Análisis de velocidades del frente

Se dispone de 6 imágenes en total (3 del Paper y 3 de la Aplicación), una por caso: Figura 7.7 Las velocidades del frente (v) para cada caso se toman de los valores reportados (Paper) y los obtenidos en la Aplicación. Se calcula el error absoluto $|\Delta v| = |v_{app} - v_{paper}|$ y el error relativo $\varepsilon_v = |\Delta v|/v_{paper}$.

Caso	v_{paper} (m/s)	v_{app} (m/s)	$\lvert\Delta v\rvert$ (m/s)	ε_v (%)
1	$2{,}31 \times 10^{-5}$	$2{,}39 \times 10^{-5}$	$8{,}00 \times 10^{-7}$	3.46
2	$2{,}04 \times 10^{-5}$	$2{,}14 \times 10^{-5}$	$1{,}00 \times 10^{-6}$	4.90
3	$2{,}02 \times 10^{-5}$	$2{,}14 \times 10^{-5}$	$1{,}20 \times 10^{-6}$	5.94

Conclusión (velocidades): La discrepancia relativa de velocidad entre Aplicación y Paper se encuentra entre 3.46 % y 5.94 % para los casos analizados.

7.2.2.1. Caso 1

Tiempos considerados: 1000 s, 2000 s, 3000 s.

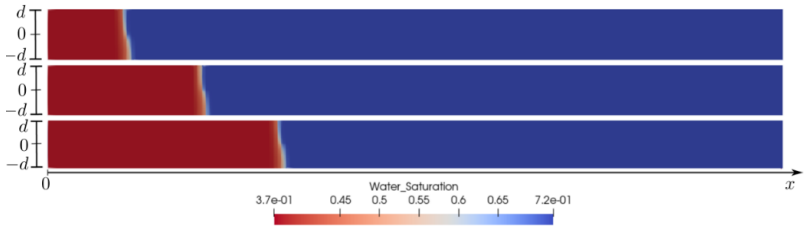
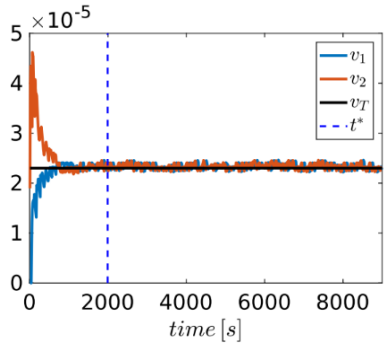
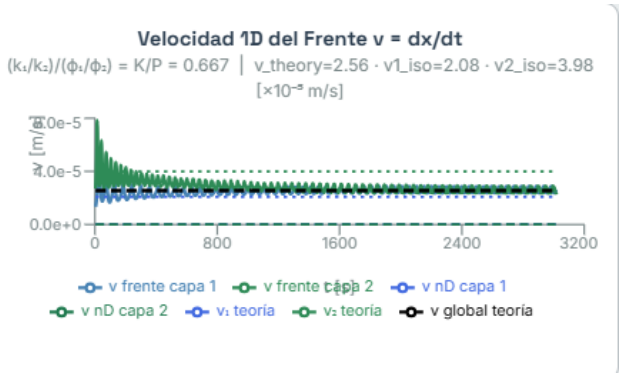


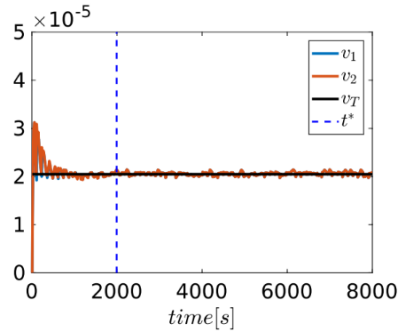
Figura 7.8: Caso 1: simulación tomada de [1]



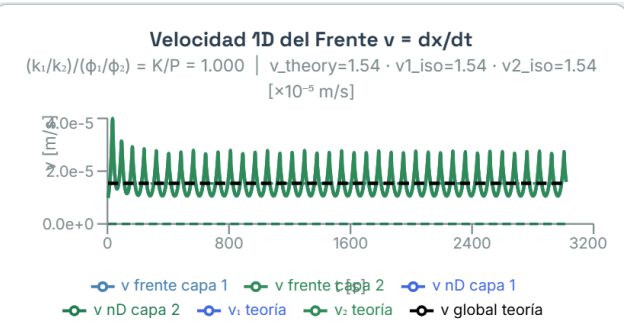
(a) Gráfica de velocidad [1]: "Velocidad del frente reportada en el Paper para el Caso 1"



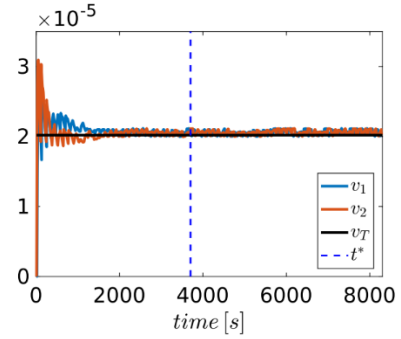
(b) Gráfica de velocidad (Aplicación): "Velocidad del frente obtenida en la Aplicación para el Caso 1"



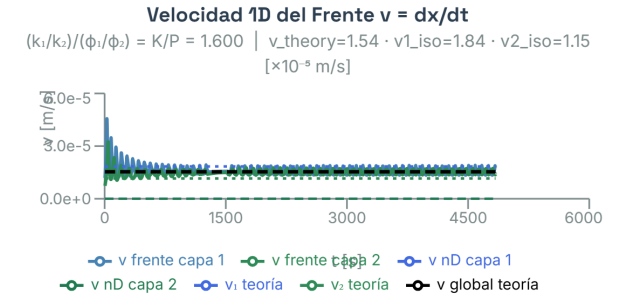
(c) Gráfica de velocidad [1]: "Velocidad del frente reportada en el Paper para el Caso 2."



(d) Gráfica de velocidad (Aplicación): "Velocidad del frente obtenida en la Aplicación para el Caso 2".



(e) Gráfica de velocidad [1]: "Velocidad del frente reportada en el Paper para el Caso 3."



(f) Gráfica de velocidad (Aplicación): "Velocidad del frente obtenida en la Aplicación para el Caso 3".

Figura 7.7: Velocidades del frente tomadas de [1] vs la aplicación

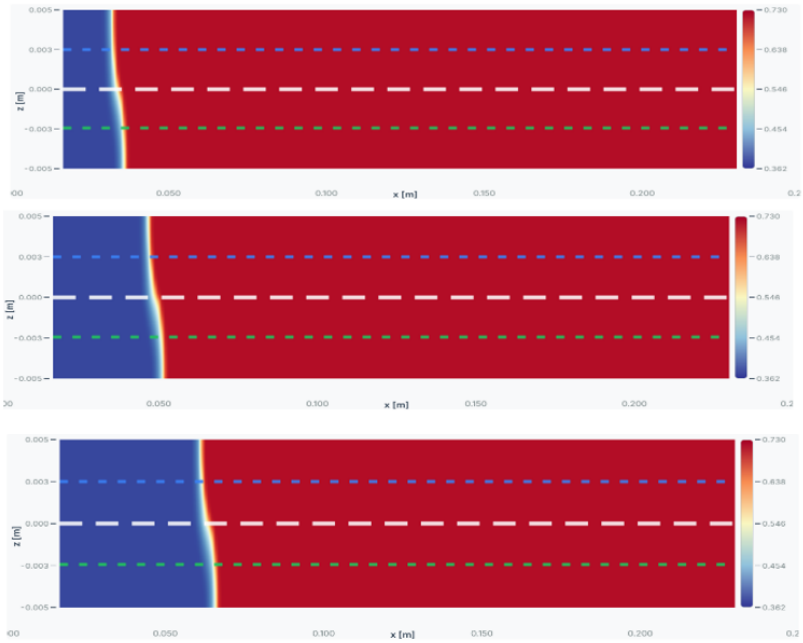


Figura 7.9: Caso 1: simulación realizada en la aplicación bidimensional

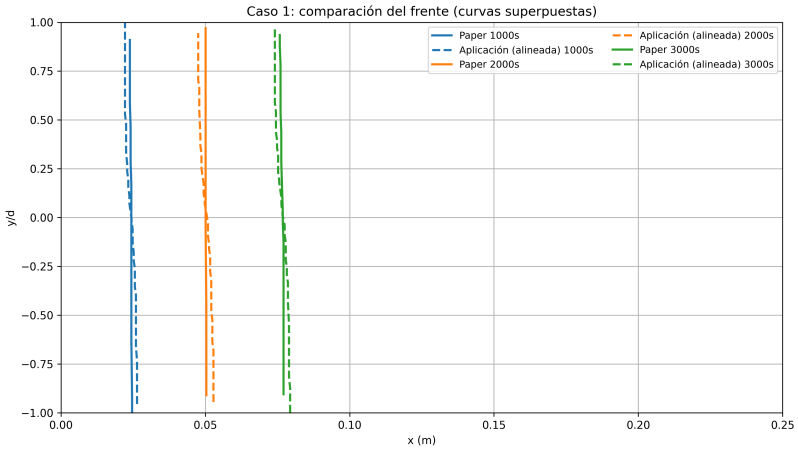


Figura 7.10: Caso 1: frentes detectadas. Se muestran puntos extraídos y la curva interpolada $x(y/d)$ para [1] y Aplicación en cada tiempo.

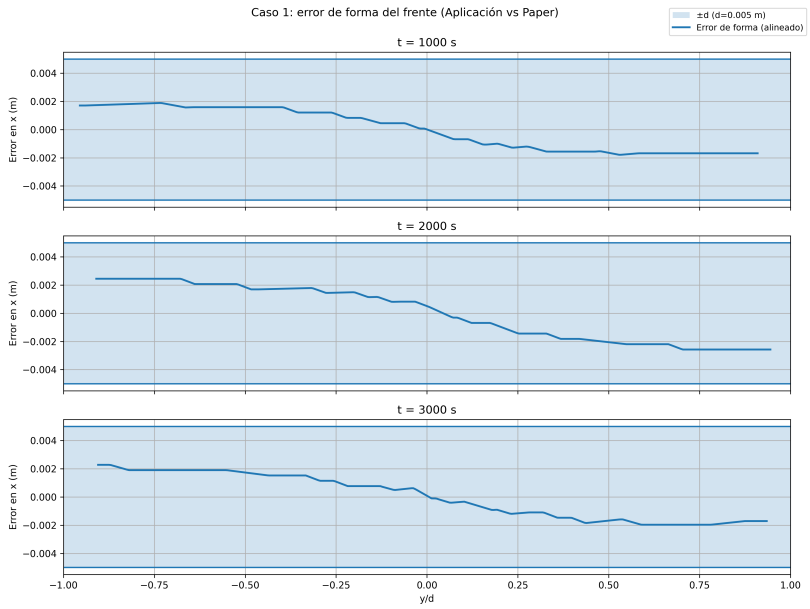


Figura 7.11: Caso 1: error de forma del frente $e_{\text{forma}}(y) = x_{\text{app}}(y) - x_{\text{paper}}(y) - \Delta x$ con banda de tolerancia $\pm d$ (d=0.005 m).

Tiempo	$\bar{x}_{paper}$ (m)	$\bar{x}_{app}$ (m)	Δx (m)	RMSE forma (m)	Dentro $\pm d$ (%)	Calidad
1000 s	0.024251	0.024802	0.000551	0.001435	100.0	aceptable
2000 s	0.050194	0.041123	-0.009071	0.001897	100.0	aceptable
3000 s	0.076661	0.057325	-0.019336	0.001541	100.0	aceptable

Conclusión del caso 1: En términos de similitud de forma (tras alinear por Δx), la comparación se clasifica como aceptable.

7.2.2.2. Caso 2

Tiempos considerados: 1000 s, 2000 s, 3000 s.

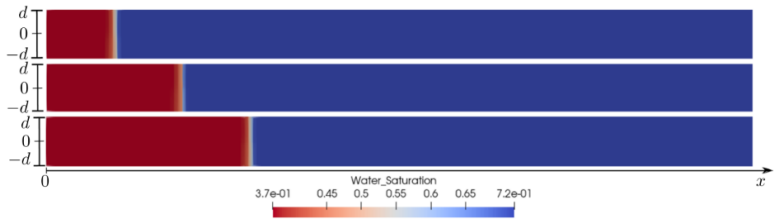


Figura 7.12: Caso 2: simulación tomada [1]

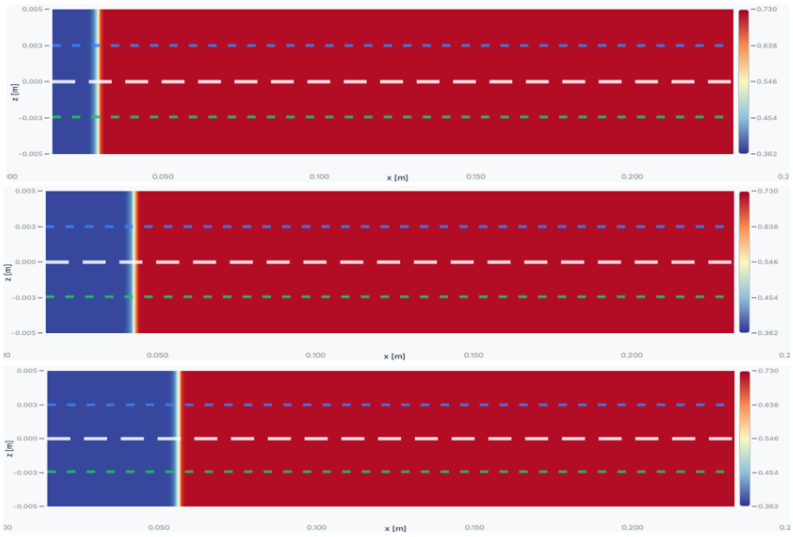


Figura 7.13: Caso 2: simulación realizada en la aplicación bidimensional

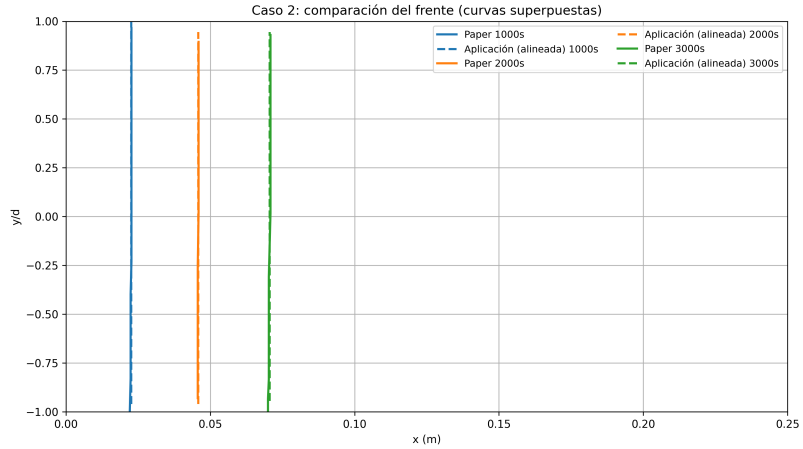


Figura 7.14: Caso 2: frentes detectados. Se muestran puntos extraídos y la curva interpolada $x(y/d)$ para [1] y Aplicación en cada tiempo.

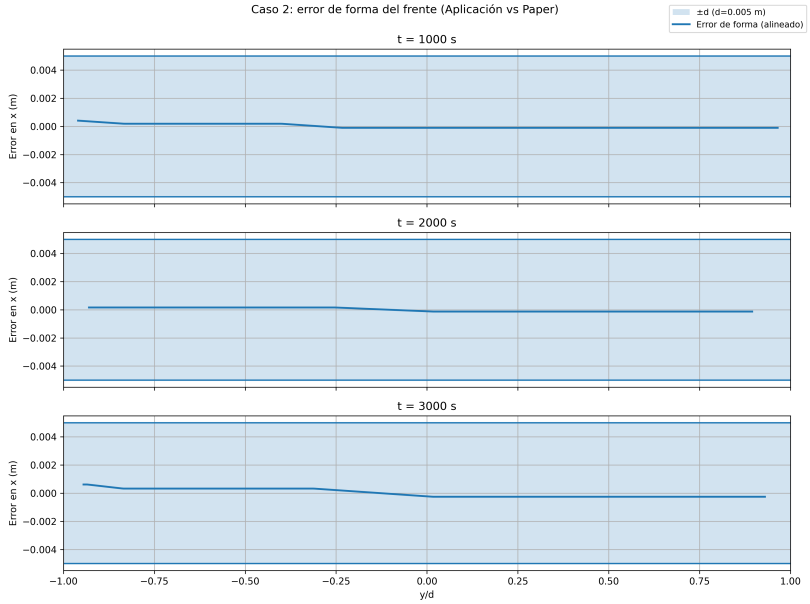


Figura 7.15: Caso 2: error de forma del frente $e_{\text{forma}}(y) = x_{\text{app}}(y) - x_{\text{paper}}(y) - \Delta x$ con banda de tolerancia $\pm d$ (d=0.005 m).

Tiempo	$\bar{x}_{paper}$ (m)	$\bar{x}_{app}$ (m)	Δx (m)	RMSE forma (m)	Dentro $\pm d$ (%)	Calidad
1000 s	0.022570	0.020806	-0.001764	0.000147	100.0	buena
2000 s	0.045801	0.035294	-0.010507	0.000137	100.0	buena
3000 s	0.070591	0.050393	-0.020198	0.000285	100.0	buena

Conclusión del caso 2: En términos de similitud de forma (tras alinear por Δx), la comparación se clasifica como buena.

7.2.2.3. Caso 3

Tiempos considerados: 3000 s, 4000 s, 5000 s.

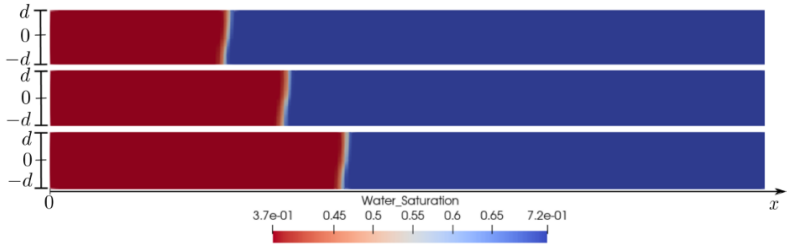


Figura 7.16: Caso 3: simulación de [1]

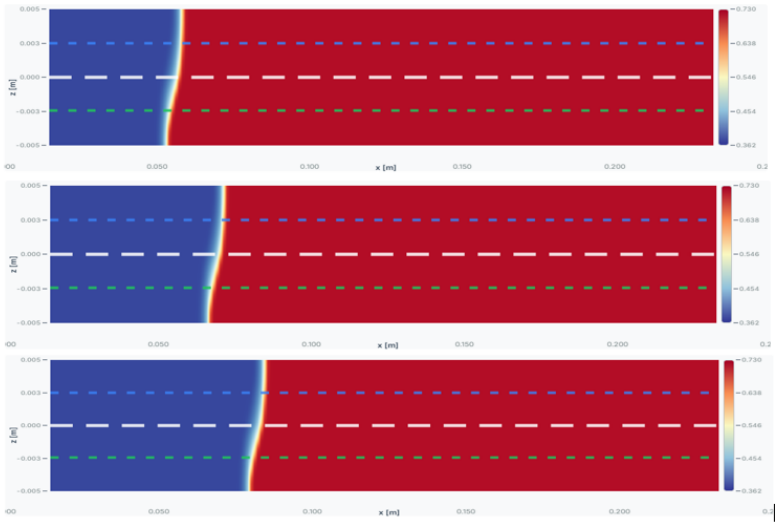


Figura 7.17: Caso 3: simulación realizada en la aplicación bidimensional

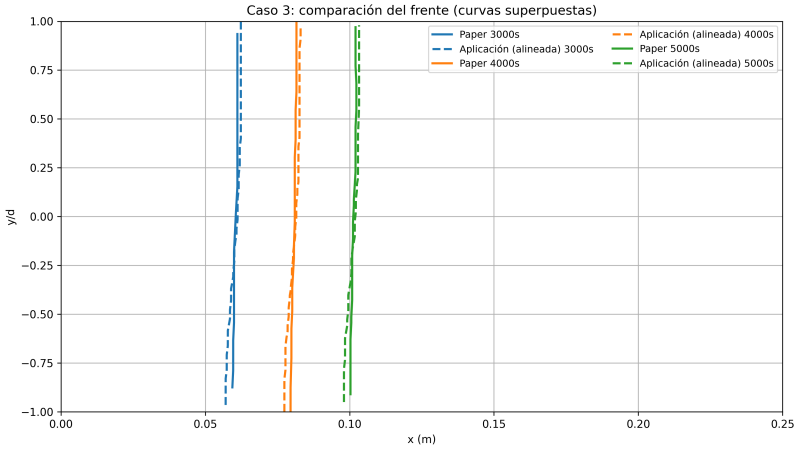


Figura 7.18: Caso 3: frentes detectados. Se muestran puntos extraídos y la curva interpolada $x(y/d)$ para [1] y Aplicación en cada tiempo.

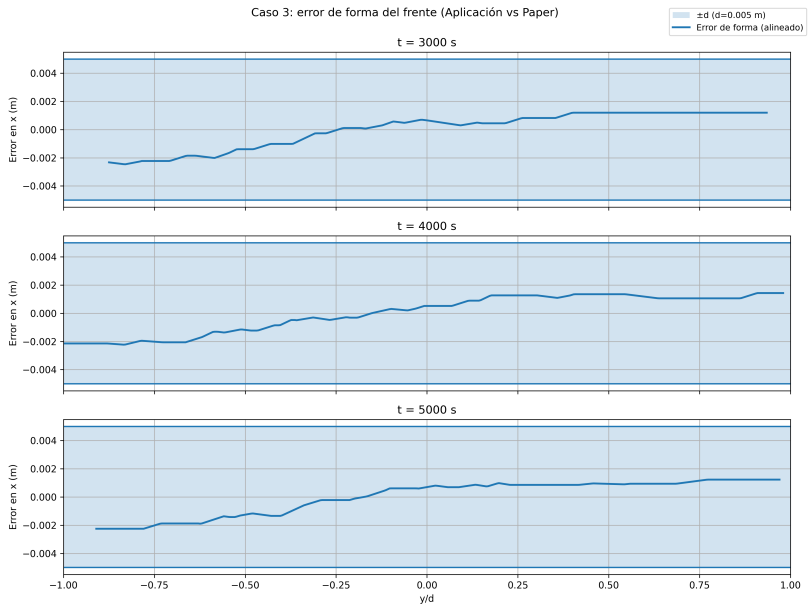


Figura 7.19: Caso 3: error de forma del frente $e_{\text{forma}}(y) = x_{\text{app}}(y) - x_{\text{paper}}(y) - \Delta x$ con banda de tolerancia $\pm d$ (d=0.005 m).

Tiempo	$\bar{x}_{paper}$ (m)	$\bar{x}_{app}$ (m)	Δx (m)	RMSE forma (m)	Dentro $\pm d$ (%)	Calidad
3000 s	0.060497	0.050446	-0.010051	0.001242	100.0	buena
4000 s	0.080701	0.064781	-0.015920	0.001300	100.0	aceptable
5000 s	0.101370	0.078932	-0.022438	0.001208	100.0	buena

Conclusión del caso 3: En términos de similitud de forma (tras alinear por Δx), la comparación se clasifica como aceptable.

7.3. Análisis de resultados

Como complemento al análisis previo, se validaron los frentes de invasión simulados mediante la comparación directa con los datos de la literatura. Para esto, se examinaron tanto las velocidades unidimensionales como la morfología bidimensional de dichos frentes.

Validación 1D (Velocidades y Perfiles)

El cálculo de las velocidades medias de avance del frente muestra una correlación sólida con las referencias bibliográficas. Los errores relativos porcentuales se mantuvieron en niveles bajos en todos los escenarios analizados:

- **Caso 1:** Error relativo de 3,46 %.
- **Caso 2:** Error relativo de 4,90 %.
- **Caso 3:** Error relativo de 5,94 %.

Estos márgenes confirman la precisión del modelo al predecir el avance en la dimensión longitudinal (x).

A partir de esta validación 1D, se logró caracterizar cuantitativamente la morfología de la onda de invasión. Para ello, se calculó la raíz del error cuadrático medio (RMSE) sobre el dominio normalizado, obteniendo valores de 0,1786 para el Caso 1, 0,2168 para el Caso 2 y 0,1928 para el Caso 3. Estos resultados respaldan la precisión del perfil de desplazamiento simulado.

Validación 2D (Forma del frente)

La geometría bidimensional de los frentes hidrodinámicos se evaluó mediante el error transversal de la forma ($e_{\text{forma}}(y)$), aplicando un desplazamiento previo Δx para alinear los perfiles. Tomando como referencia una tolerancia geométrica de $d = 0{,}005$ m, los resultados indican que:

- Todos los contornos extraídos a lo largo del tiempo (entre 1000 s y 5000 s) se situaron dentro de la banda de tolerancia establecida de $\pm d$.
- El error de forma (RMSE) osciló entre un mínimo de $\sim 0{,}00014$ m para el Caso 2 y un máximo de $\sim 0{,}0019$ m durante el transitorio del Caso 1.

El cumplimiento de estos límites de tolerancia demuestra que el código reproduce correctamente los patrones de *viscous crossflow* y las dinámicas de desplazamiento entre capas.

CAPÍTULO 8

Conclusiones

Conclusiones

- Se desarrolló *FOAMFLOW-2D*, una herramienta computacional que integra modelos matemáticos complejos de dinámica de fluidos con tecnologías web modernas. La implementación del núcleo numérico en JavaScript y su ejecución mediante *Web Workers* permitió desacoplar el procesamiento intensivo, garantizando una interfaz reactiva y eficiente durante la resolución del sistema de ecuaciones diferenciales.

- La arquitectura orientada a servicios, soportada en React y Node.js, demostró alta eficacia para el manejo de flujos de datos científicos en tiempo real. La integración de *WebSockets* fue fundamental para minimizar la latencia en la visualización, mientras que el despliegue en la nube (Vercel y Render) validó la portabilidad del sistema, eximiendo al usuario de configuraciones locales complejas.

- La precisión de la herramienta se validó frente a literatura técnica de referencia, registrando una discrepancia relativa en la velocidad del frente de entre **3.46 % y 5.94 %**. Asimismo, los análisis de similitud espacial confirmaron que la aplicación modela con alta fidelidad la física del fenómeno de *crossflow* viscoso en medios estratificados, cumpliendo con el rigor académico exigido.

- El proyecto aporta de manera directa a los Objetivos de Desarrollo Sostenible (ODS 3 y 6) al facilitar el acceso a simulaciones para aplicaciones en salud (como la escleroterapia) y medio ambiente (remediación de acuíferos). Como resultado, se entrega una plataforma que acorta la brecha entre el modelamiento matemático avanzado y su aplicación práctica por parte de investigadores y profesionales.

CAPÍTULO 9

Trabajo Futuro

Una de las vetas de desarrollo más prometedoras para expandir las capacidades de *foam-flow-web* radica en la superación del paradigma del equilibrio local, una suposición común en los simuladores de primera generación que simplifica drásticamente el comportamiento temporal del fluido. En el modelo actual, se asume que los procesos de generación y destrucción de burbujas ocurren de manera instantánea, adaptándose de inmediato a las condiciones macroscópicas de presión y velocidad del entorno. Si bien esta aproximación permite resolver el sistema con un menor costo computacional y ofrece una excelente aproximación inicial, la realidad física en un medio poroso real exhibe transiciones mucho más lentas y complejas, donde la espuma requiere recorrer distancias considerables antes de alcanzar un estado estacionario estable.

Para dotar al simulador de un rigor científico aún mayor, se proyecta la integración de un modelo de balance de población completo y dinámico que opere decididamente fuera del equilibrio local. Este enfoque metodológico exige sustituir las funciones algebraicas instantáneas por ecuaciones diferenciales de transporte adicionales que gobiernen la evolución de la densidad numérica de burbujas a lo largo del tiempo y el espacio. Al introducir tasas finitas de corte, retrasos por histéresis capilar y cinéticas de degradación de película, el motor de cálculo podrá reproducir fenómenos transitorios críticos, como el retraso en la generación de espuma cerca del pozo de inyección o el colapso abrupto del frente ante variaciones bruscas de salinidad. Esta evolución arquitectónica permitirá a la herramienta web ofrecer predicciones de campo sumamente realistas, convirtiéndola en un puente robusto entre la experimentación a escala de laboratorio y el diseño de estrategias de inyección a gran escala.

Por otra parte, y con el objetivo de trascender los límites geométricos actuales, se contempla como una línea de investigación fundamental la extensión del motor numérico hacia un modelo tridimensional (3D) de inyección de espuma. Aunque el dominio bidimensional provisto en esta etapa captura con éxito la dinámica del flujo fraccional en planos específicos, los escenarios reales de explotación en yacimientos o remediación de acuíferos están intrínsecamente condicionados por fenómenos tridimensionales imposibles de replicar en 2D. La inclusión de una tercera dimensión geométrica permitirá modelar de manera fidedigna la segregación gravitacional donde el gas y la fase líquida experimentan una separación vertical debido a sus densidades dispares, el comportamiento de la espuma ante la anisotropía y he-

terogeneidad vertical de las capas geológicas, y los patrones de barrido radiales y divergentes característicos de los esquemas reales de pozos múltiples. Esta migración hacia entornos 3D planteará un reto sustancial en el diseño de software, obligando a optimizar el paralelismo computacional en los *Web Workers* o a explorar tecnologías de cómputo en GPU mediante *WebGPU*, asegurando que la interactividad y la respuesta en tiempo real sigan siendo los pilares identitarios de la plataforma.

Anexo: Declaración sobre el uso de Inteligencia Artificial

En concordancia con las políticas de integridad académica y transparencia en la investigación, se declara que se utilizó la herramienta de Inteligencia Artificial (IA) generativa **Google Gemini** durante el desarrollo de este trabajo.

El uso de esta tecnología se limitó estrictamente a dos fases metodológicas: la asistencia en la búsqueda bibliográfica avanzada y la optimización editorial del texto. En ningún caso se delegó en la IA la creación de contenido original, el análisis de datos ni las conclusiones de la investigación.

Las tareas específicas y la metodología de uso se detallan a continuación:

1. **Diseño de estrategias de búsqueda bibliográfica:** Se utilizó la IA como apoyo para identificar artículos científicos de referencia. Específicamente, se formularon cadenas de búsqueda con operadores booleanos optimizadas para motores de indexación académica, enfocadas en los siguientes ejes temáticos:

   - *Modelo matemático (conservación no lineal):* `("Buckley-Leverett equation.OR "two-phase flow") AND "porous media.AND ("2D numerical simulation.OR "finite difference method") AND after:2020`
   - *Traveling waves en medios porosos:* `("traveling wave solution.OR "shock wave propagation") AND "porous media.AND ("two-phase flow.OR "fractional flow") AND after:2020`
   - *Medios porosos heterogéneos o por capas:* `("layered porous media.OR "heterogeneous porous medium") AND ("two-phase flow.OR "Buckley-Leverett") AND ("numerical simulation.OR "finite volume") AND after:2020`
   - *Modelación computacional 2D:* `("two-dimensional model.OR "2D simulation") AND "porous media flow.AND ("finite difference.OR "finite volume method") AND after:2020`
   - *Término difusivo y capilaridad:* `(çapillary diffusion.OR "parabolic regularization") AND "Buckley-Leverett.AND "porous media.AND after:2020`
   - *Fundamentos matemáticos avanzados:* `("nonlinear conservation law") AND "porous media.AND ("traveling wave.OR "shock formation") AND after:2020`

2. **Optimización de la escritura y redacción académica:** Con el objetivo de mejorar la calidad del manuscrito, se interactuó con la IA solicitando propuestas de mejora sobre borradores propios. Este proceso incluyó:

   - Peticiones explícitas de reescritura para mejorar la fluidez, cohesión y claridad de los párrafos en la introducción y el marco teórico.
   - Corrección gramatical, detección de redundancias y ajuste al tono formal requerido en textos de ingeniería.
   - Sugerencias de vocabulario técnico especializado para la descripción de los esquemas numéricos.

   Nota de autoría: El diseño metodológico, la selección final de las referencias bibliográficas, el desarrollo de los modelos de simulación, la interpretación de los resultados y la redacción definitiva de este documento son responsabilidad exclusiva del autor. El uso de Google Gemini actuó únicamente como un asistente técnico y editorial avanzado.

Bibliografía

[1] A. J. C. Vasquez, P.Z.S.Paz, and G. Chapiro, "On the viscous crossflow during the foam displacement in two-layered porous media," *Transport in Porous Media*, vol. 151, p. 2835–2857, 2024.

[2] É. P. F. de Lausanne, "Ver y escalas de observación. vicaire, módulo 3, capítulo 1," s.f. Recuperado el 16 de febrero de 2026.

[3] J. Sanchez, J. Collazos, O. Vargas, A. J. C. V., and C. Ramirez, "Numerical modeling of traveling foam fronts in stratified porous media using inverse euler, assuming local equilibrium," *Submitted*, vol.–, no.–, pp. 1–16, 2025.

[4] A. J. C. Vasquez and G. Chapiro, "Wavefront velocity for foam flow in three-layer porous media," *XLIII Ibero- Latin-American Congress on Computational Methods in Engineering (CILAMCE)*, 2022.

[5] J. Bear, *Modeling Phenomena of Flow and Transport in Porous Media*. Theory and Applications of Transport in Porous Media, Springer International Publishing, 2018.

[6] J. Bikerman, *Foams*. Springer-Verlag, New York, 1973.

[7] E. Ashoori, D. Marchesin, and W. R. Rossen, "Roles of transient and local equilibrium foam behavior in porous media: Traveling wave," *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, vol. 377, no. 1–3, pp. 228–242, 2011.

[8] C. B. Reynoso, "Introducción a la arquitectura de software," *Universidad de Buenos Aires*, vol. 33, p. 11, 2004.

[9] A. R. Kovscek, T. W. Patzek, and C. J. Radke, "A mechanistic population balance model for transient and steady-state foam flow in boise sandstone," *Chemical Engineering Science*, vol. 50, no. 23, pp. 3783–3799, 1995.

[10] G. J. Hirasaki, "The steam-foam process," *Journal of Petroleum Technology*, vol. 41, no. 5, pp. 449–456, 1989.

[11] N. Sarochar, "Desarrollo de un simulador 3d para propagación de ondas en medios porosos," 2020.

[12] I. P. Tello-Guerrero, H. Gonzáles-Alvarez, and A. E. Calle-Ochoa, "Modelado de propagación de ondas en medios porosos saturados con fluidos," *Mecánica Computacional*, 2009.

[13] T. G. Roberts, S. J. Cox, A. L. Lewis, and S. A. Jones, "Characterisation and optimisation of foams for varicose vein sclerotherapy," *Biorheology*, vol. 57, no. 2-4, pp. 77–85, 2020.

[14] F. F. de Paula, I. Igreja, T. Quinelato, and G. Chapiro, "A numerical investigation into the influence of the surfactant injection technique on the foam flow in heterogeneous porous media," *Advances in Water Resources*, vol. 171, p. 104358, 2023.

[15] E. Ashoori, D. Marchesin, and W. R. Rossen, "Dynamic foam behavior in the entrance region of a porous medium," *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, vol. 377, pp. 217–227, 2011.

[16] A. J. C. Vásquez, L. F. Lozano, W. S. Pereira, J. B. Cedro, and G. Chapiro, "The traveling wavefront for foam flow in two-layer porous media," *Computational Geosciences*, vol. 26, no. 6, pp. 1549–1561, 2022.

[17] P. L. Zitha, Q. P. Nguyen, and P. K. Currie, "Effect of flow velocity and rock layering on foam flow: an x-ray computed tomography study," in *SPE Asia Pacific Oil and Gas Conference and Exhibition*, pp. SPE–80530, SPE, 2003.

[18] A. R. Kovscek, T. W. Patzek, and C. J. Radke, "Mechanistic prediction of foam displacement in multidimensions: A population balance approach," in *SPE/DOE Improved Oil Recovery Symposium*, Society of Petroleum Engineers, 1994.

[19] E. Ashoori, D. Marchesin, and W. R. Rossen, "Stability analysis of uniform equilibrium foam states for EOR processes," *Transport in Porous Media*, vol. 92, no. 3, pp. 573–595, 2012.

[20] J. B. Cedro, R. V. Quispe, M. C. Coaquira, L. F. Lozano, and G. Chapiro, "Estudo de um modelo cinético para escoamento de espuma em meios porosos," in *XL Ibero-Latin-American Congress on Computational Methods in Engineering (CILAMCE)*, (Natal, Brazil), pp. 1–13, 2019.

[21] Q. P. Nguyen, P. K. Currie, and P. L. J. Zitha, "Effect of crossflow on foam-induced diversion in layered formations," *SPE Journal*, vol. 10, pp. 54–65, 03 2005.

[22] A. Castrillón Vásquez, L. Lozano, and G. Chapiro, "The traveling foam wavefront in fractured porous medium," *Journal of Computational Physics*, vol. 519, p. 113437, 2024.

[23] H. J. Bertin, O. G. Apaydin, L. M. Castanier, and A. R. Kovscek, "Foam flow in heterogeneous porous media: Effect of cross flow," *SPE Journal*, vol. 4, pp. 75–82, 06 1999.

[24] F. F. de Paula, T. Quinelato, I. Igreja, and G. Chapiro, "A numerical algorithm to solve the two-phase flow in porous media including foam displacement," *Lecture Notes in Computer Science*, vol. 12143, pp. 18–31, 2020.

[25] A. Valdez, B. Rocha, A. Pérez-Gramatges, J. Façanha, A. de Souza, G. Chapiro, and R. dos Santos, "Foam assisted water-gas flow parameters: from core-flood experiment to uncertainty quantification and sensitivity analysis," *Transport in Porous Media*, 2021.

[26] S. Berg, E. Unsal, and H. Dijk, "Sensitivity and uncertainty analysis for parameterization of multiphase flow models," *Transport in Porous Media*, pp. 1–31, 2021.

[27] L. F. Lozano, *Diffusive effects in Riemann solutions for the three phase flow in porous media*. PhD thesis, Instituto Nacional de Matemática Pura e Aplicada, IMPA, 2018.

[28] D. Skvorc, M. Horvat, and S. Srbljic, "Performance evaluation of websocket protocol for implementation of full-duplex web streams," in *2014 37th international convention on information and communication technology, electronics and microelectronics (MIPRO)*, pp. 1003–1008, IEEE, 2014.