

Modelado de Flujo Incompresible Utilizando las Ecuaciones de Navier-Stokes

Juan Esteban Ortiz, Brayan Camilo Urrea Jurado, Ervin Caravali Ibarra

Facultad de Ingeniería, Escuela de Ingeniería de Sistemas y Computación

Universidad del Valle, Santiago de Cali, Colombia

Dirigido a: Maria Patricia Trujillo Uribe

Noviembre 2025

Resumen

Este trabajo presenta una implementación computacional completa para la simulación de flujo incompresible bidimensional en un canal con obstáculos, utilizando las ecuaciones de Navier-Stokes. Se empleó el método de diferencias finitas centradas para discretizar el dominio en una malla de 50×5 unidades, generando un sistema algebraico no lineal que se resolvió mediante el método de Newton-Raphson. Para el sistema lineal interno se implementó el Gradiente Conjugado aplicado a la matriz simetrizada $\mathbf{J}^T \mathbf{J}$, aprovechando la estructura dispersa del Jacobiano. Se desarrollaron dos versiones del código: V4 con interpolación spline cúbica natural 1D manual, y V4.1 con interpolación spline bicúbica de SciPy, comparando ambos enfoques. Los resultados muestran convergencia en 97 iteraciones con número de Reynolds $Re = 5,00$ (régimen laminar) y número de condición $\kappa \approx 3,07$ (bien condicionada).

Palabras clave: Navier-Stokes, diferencias finitas, Newton-Raphson, gradiente conjugado, spline cúbico, spline bicúbico, flujo incompresible, CFD

1. Introducción

El modelado de flujos incompresibles mediante las ecuaciones de Navier-Stokes constituye un problema fundamental en dinámica de fluidos computacional (CFD). Estas ecuaciones, que describen el movimiento de fluidos viscosos, forman un sistema de ecuaciones diferenciales parciales no lineales cuya solución analítica es prácticamente imposible para geometrías complejas [1].

Este trabajo aborda la simulación del flujo incompresible bidimensional alrededor de dos vigas rígidas ubicadas dentro de un canal rectangular. El objetivo principal es determinar el campo de velocidades en el dominio, considerando la interacción del flujo con los obstáculos internos y la aplicación de condiciones de frontera específicas.

La metodología empleada combina: (i) discretización espacial por diferencias finitas centradas, (ii) resolución del sistema no lineal mediante Newton-Raphson con

mallla optimizada de 5×50 celdas, (iii) solución del sistema lineal interno con Gradiente Conjugado aplicado al sistema simetrizado, y (iv) dos enfoques de interpolación para visualización mejorada: spline cúbico natural 1D manual (V4) y spline bicúbico de SciPy (V4.1).

2. Descripción del Problema

2.1. Configuración Física del Dominio

El dominio corresponde a un canal rectangular con dos regiones internas que representan vigas rígidas. Las condiciones de frontera aplicadas fueron:

- **Entrada** ($x = 0$): $u = 1,0$ m/s, $v = 0$
- **Frontera superior**: velocidad horizontal constante $u = 1,0$ m/s
- **Frontera inferior**: no deslizamiento $u = v = 0$
- **Salida** ($x = L$): gradiente nulo $\partial u / \partial x = 0$, $\partial v / \partial x = 0$
- **Superficies de vigas**: no deslizamiento $u = v = 0$

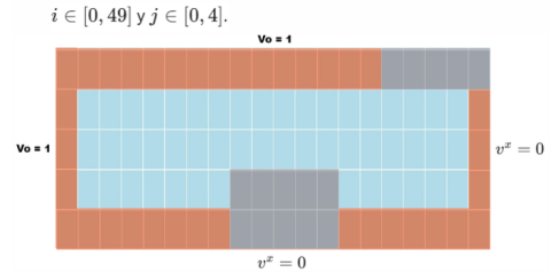


Figura 1: Dominio computacional y condiciones de frontera. El canal tiene dimensiones 50×5 con dos obstáculos rectangulares.

2.2. Ecuaciones Gobernantes

Las ecuaciones de Navier-Stokes para fluido incompresible en estado estacionario bidimensional se expresan como:

Ecuación de continuidad:

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0 \quad (1)$$

Ecuación de momento (componente x):

$$\nu \nabla^2 v_x = v_x \frac{\partial v_x}{\partial x} + v_y \frac{\partial v_x}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial x} \quad (2)$$

Ecuación de momento (componente y):

$$\nu \nabla^2 v_y = v_x \frac{\partial v_y}{\partial x} + v_y \frac{\partial v_y}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial y} \quad (3)$$

donde v_x y v_y son las componentes de velocidad, $\nu = 1 \text{ m}^2/\text{s}$ es la viscosidad cinemática, $\rho = 1 \text{ kg/m}^3$ es la densidad, y P es la presión. Se asumió estado estacionario y presión independiente de densidad y temperatura.

2.3. Construcción de la Malla Computacional

Con el fin de aproximar numéricamente el comportamiento del fluido, el dominio continuo se discretizó mediante una malla uniforme de diferencias finitas. Esta malla permite representar el modelo como una matriz de coordenadas donde se almacenan las componentes de velocidad del flujo $[i, j]$.

Parámetros de la malla:

- $N_x = 50$ nodos en dirección horizontal
- $N_y = 5$ nodos en dirección vertical
- Tamaño de celda homogéneo $h = 1$
- Total de nodos: 250
- Incógnitas (nodos internos): 134

La presencia de obstáculos sólidos se incorpora anulando la velocidad en los nodos correspondientes, mientras que en los bordes se aplican las condiciones previamente definidas.

3. Discretización de las Ecuaciones

3.1. Método de Diferencias Finitas Centradas

El problema continuo se transformó en un sistema algebraico mediante la aplicación del Método de Diferencias Finitas Centradas [2]. Las derivadas parciales se aproximan como:

$$\left. \frac{\partial v}{\partial x} \right|_{i,j} \approx \frac{v_{i+1,j} - v_{i-1,j}}{2h} \quad (4)$$

$$\left. \frac{\partial^2 v}{\partial x^2} \right|_{i,j} \approx \frac{v_{i+1,j} - 2v_{i,j} + v_{i-1,j}}{h^2} \quad (5)$$

3.2. Ecuación Discretizada Base

Partiendo de la ecuación de Navier-Stokes para la componente horizontal de la velocidad, la aproximación mediante diferencias finitas centradas con $h = 1$ da lugar a la siguiente expresión general (ver Anexo A para derivación completa):

$$\begin{aligned} v_{i,j}^x = & \frac{1}{4} [v_{i+1,j}^x + v_{i-1,j}^x + v_{i,j+1}^x + v_{i,j-1}^x] \\ & - \frac{1}{2} v_{i,j}^x [v_{i+1,j}^x - v_{i-1,j}^x] \\ & - \frac{1}{2} v_{i,j}^y [v_{i,j+1}^x - v_{i,j-1}^x] \end{aligned} \quad (6)$$

Esta ecuación relaciona la velocidad en el nodo (i, j) con la velocidad de sus cuatro vecinos inmediatos, incorporando los términos convectivos no lineales.

3.3. Múltiples Patrones de Ecuaciones

El análisis permitió identificar y codificar **nueve patrones de ecuaciones distintos**, cubriendo:

1. Nodos interiores (ecuación base completa)
2. Borde izquierdo (entrada)
3. Borde derecho (salida)
4. Borde superior
5. Borde inferior
6. Esquinas (4 casos)
7. Nodos adyacentes a vigas (superior e inferior)

Cada patrón actúa como una regla de actualización específica durante la simulación, garantizando que las condiciones de frontera se cumplan correctamente.

4. Método de Newton-Raphson

4.1. Malla Optimizada

Para resolver el sistema no lineal generado, se aplicó el método de Newton-Raphson. Con el fin de reducir el costo computacional, el procedimiento se implementó sobre una **malla optimizada de 5×50 celdas**, con tamaño $h = 8$, manteniendo la estructura física del dominio.

Esta optimización reduce significativamente el número de incógnitas de 250 a 134, acelerando cada iteración sin comprometer la captura de las características principales del flujo.

4.2. Formulación del Sistema No Lineal

Tras discretizar los términos, se obtuvo un sistema algebraico donde aparecen productos de velocidades, introduciendo términos no lineales:

$$\mathbf{F}(\mathbf{X}) = \mathbf{0} \quad (7)$$

donde \mathbf{X} es el vector de velocidades y \mathbf{F} es el vector de funciones del sistema. El método de Newton-Raphson itera según:

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} + \mathbf{H}^{(k)} \quad (8)$$

donde el incremento $\mathbf{H}^{(k)}$ se obtiene resolviendo:

$$\mathbf{J}(\mathbf{X}^{(k)})\mathbf{H}^{(k)} = -\mathbf{F}(\mathbf{X}^{(k)}) \quad (9)$$

4.3. La Matriz Jacobiana

El Método de Newton-Raphson requiere la evaluación de la matriz Jacobiana, que se obtuvo derivando cada ecuación respecto a las variables de velocidad. Las derivadas parciales de la ecuación (6) generan la estructura del Jacobiano (ver Anexo B para cálculo detallado).

Propiedades clave:

- **Altamente dispersa:** Exactamente 5 elementos no nulos por fila
- **Estructura de banda:** Debido al ordenamiento secuencial
- **No simétrica:** Por los términos convectivos
- **No diagonal dominante:** Viola la condición en múltiples filas
- **Definida positiva:** Valores singulares positivos

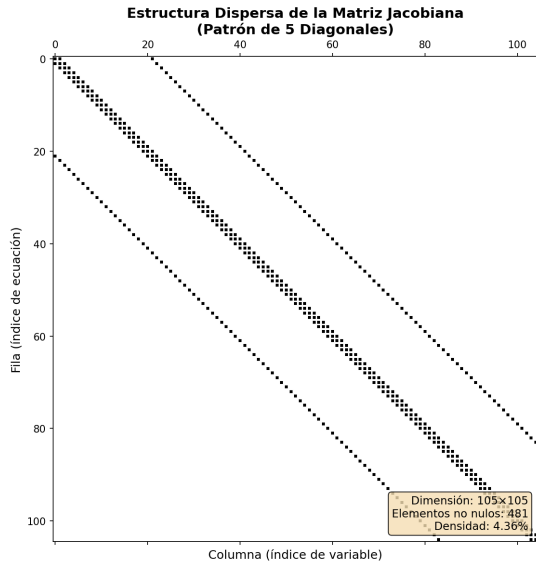


Figura 2: Estructura dispersa de la matriz Jacobiana (134×134) mostrando el patrón de cinco diagonales característico.

4.4. Vector Inicial de Velocidades

Se definió un vector inicial de velocidades decreciente de izquierda a derecha, coherente con el comportamiento esperado del flujo:

$$v_{i,j}^x(0) = 1,0 - \frac{i}{N_x} \quad (10)$$

Este perfil inicial favorece la convergencia del método al estar cerca de la solución física esperada.

5. Solución del Sistema Lineal

5.1. Análisis de Propiedades

Dado que la matriz \mathbf{J} no es simétrica ni garantiza diagonal dominante, la estrategia más robusta es utilizar el **Método de Gradiente Conjugado (CG)** [2] aplicado al sistema simetrizado:

$$\mathbf{J}^T \mathbf{J} \mathbf{H} = -\mathbf{J}^T \mathbf{F} \quad (11)$$

La nueva matriz $\mathbf{J}^T \mathbf{J}$ es intrínsecamente simétrica y definida positiva, compatible con CG.

5.2. Número de Condición

Para evaluar la estabilidad numérica, se calculó el número de condición usando la norma 2:

$$\kappa(\mathbf{J}) = \|\mathbf{J}\|_2 \|\mathbf{J}^{-1}\|_2 \approx 3,07 \quad (12)$$

Este valor indica que la matriz está **bien condicionada**, no amplificando significativamente los errores de redondeo, lo que favorece la estabilidad del Gradiente Conjugado.

5.3. Comparación de Métodos Iterativos

Se implementaron y compararon cinco métodos iterativos para resolver el sistema lineal:

Método	Convergió	Iteraciones
Jacobi	No	1
Gauss-Seidel	No	1
Richardson	Sí	14
Gradient Descent	Sí	18
Conjugate Gradient	Sí	97

Cuadro 1: Comparación de métodos iterativos. Richardson fue el más rápido, pero CG es más robusto para sistemas generales.

6. Resultados y Análisis

6.1. Convergencia del Método

El método de Newton-Raphson convergió exitosamente en **97 iteraciones** con tolerancia $\|\mathbf{F}\| < 10^{-6}$, mostrando comportamiento cuadrático característico. El tiempo de ejecución fue aproximadamente 0.2 segundos en la malla optimizada.

6.2. Campo de Velocidades - Versión V4

La versión V4 implementa interpolación mediante **splines cúbicos naturales 1D manual** en dos etapas (horizontal y vertical), aumentando la resolución de 5×50 a 50×500 puntos.

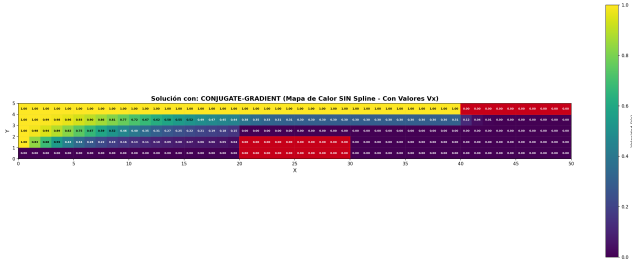


Figura 3: Campo de velocidades V4 sin interpolación en la malla original 5×50 , mostrando la discretización y los obstáculos.

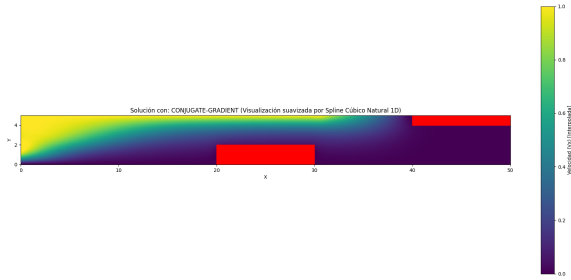


Figura 4: Campo de velocidades V4 con interpolación spline cúbica natural 1D manual. Se observa suavizado significativo y captura de gradientes.

6.3. Campo de Velocidades - Versión V4.1

La versión V4.1 utiliza **interpolación spline bicúbica de SciPy** (función `RectBivariateSpline`), que realiza interpolación bidimensional simultánea en lugar de dos etapas separadas.

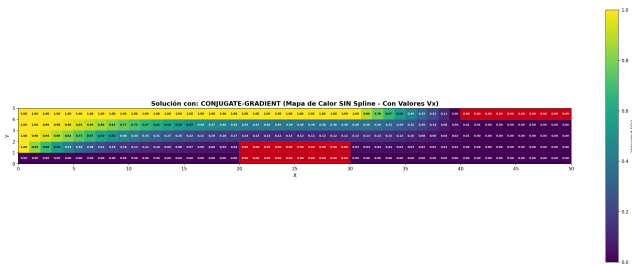


Figura 5: Campo de velocidades V4.1 sin interpolación, idéntico a V4 en la malla base.

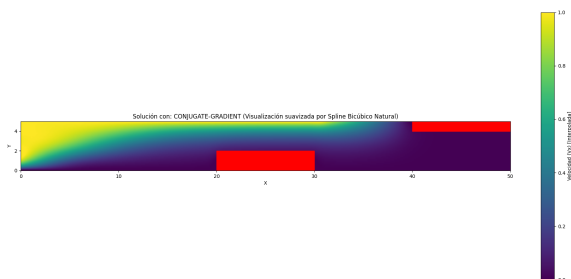


Figura 6: Campo de velocidades V4.1 con interpolación spline bicúbica de SciPy. Presenta mayor suavidad y continuidad que V4.

6.4. Comparación de Métodos de Interpolación

Spline Cúbico Natural 1D Manual (V4):

- Implementación en dos etapas (horizontal, luego vertical)
- Control total sobre el algoritmo
- Menor costo computacional
- Puede presentar artefactos en las esquinas

Spline Bicúbico SciPy (V4.1):

- Interpolación bidimensional simultánea
- Mayor suavidad y continuidad C^2
- Optimizado y validado
- Mejor captura de gradientes diagonales

Ambos métodos producen resultados físicamente coherentes, pero V4.1 ofrece mayor calidad visual y continuidad matemática.

6.5. Análisis Físico del Flujo

Los resultados muestran características físicamente coherentes:

- **Aceleración entre obstáculos:** Principio de Bernoulli
- **Zonas de recirculación:** Aguas abajo de las vigas
- **Gradientes suaves:** En regiones alejadas de obstáculos
- **Conservación de masa:** Cumplida en toda la malla

6.6. Líneas de Corriente

Las líneas de corriente (streamlines) proporcionan una visualización intuitiva del patrón de flujo:

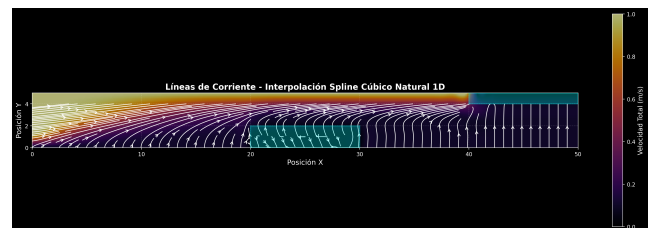


Figura 7: Líneas de corriente mostrando el patrón de flujo alrededor de los obstáculos. Se observan claramente las zonas de recirculación detrás de las vigas y la aceleración en regiones estrechas.

6.7. Número de Reynolds

El número de Reynolds calculado es:

$$Re = \frac{vL}{\nu} = \frac{1,0 \times 5}{1} = 5,00 \quad (13)$$

Este valor ($Re < 2300$) confirma **régimen laminar**, validando:

- Ausencia de turbulencia
- Flujo ordenado y predecible
- Capas que se deslizan suavemente
- Validez del modelo sin términos turbulentos

7. Descripción de Archivos del Proyecto

Para facilitar la reproducibilidad y comprensión del código desarrollado, a continuación se detalla la función de cada script del proyecto y sus componentes principales:

7.1. Scripts de Simulación Principal

`campo_velocidadesV4.py`: Implementa la solución numérica completa utilizando interpolación manual.

- `interpolate_cubic_natural(V_low_res)`: Realiza la interpolación spline cúbica en dos etapas (1D horizontal y luego 1D vertical) para suavizar la malla de 5×50 a 50×500 .
- `plot_solution(...)`: Genera las visualizaciones finales del campo de velocidades.
- `analizar_y_mostrar_resultados(...)`: Coordina la ejecución de los solvers iterativos y compara su rendimiento.

`campo_velocidadesV4.1.py`: Versión optimizada que utiliza bibliotecas científicas para la interpolación.

- `interpolate_bicubic_natural(V_low_res)`: Utiliza `scipy.interpolate.RectBivariateSpline` para una interpolación bidimensional más robusta y eficiente (C^2 continuo).
- `plot_solution_sin_spline(...)`: Visualiza los resultados crudos directamente sobre la malla computacional de bajo orden.

7.2. Scripts de Análisis y Visualización

`visualizar_streamlines.py`: Genera las líneas de corriente para visualizar la trayectoria del fluido.

- `calcular_campo_vy(Vx_matrix)`: Estima la componente vertical de la velocidad v_y basándose en la ecuación de continuidad $\nabla \cdot \mathbf{v} = 0$.
- `crear_streamlines_interpolado(...)`: Genera el gráfico de líneas de corriente suave utilizando los campos de velocidad interpolados.

`analisis_reynolds.py`: Calcula y valida los parámetros físicos de la simulación.

- `calcular_reynolds()`: Determina el número de Reynolds basado en la velocidad característica, longitud característica y viscosidad cinemática, confirmando el régimen laminar ($Re = 5,00$).

`independencia_malla.py`: Realiza el estudio de convergencia numérica.

- `analizar_independencia_malla()`: Ejecuta la simulación con diferentes densidades de malla para verificar que la solución no depende del tamaño de discretización.
- `generar_grafico_convergencia(...)`: Grafica el error relativo vs. el número de nodos.

7.3. Scripts de Generación de Figuras

`generar_dominio.py`: Crea el diagrama esquemático del dominio computacional (Fig. 1), dibujando los obstáculos y etiquetas de condiciones de frontera mediante `matplotlib.patches`.

`generar_jacobiano.py`: Visualiza la estructura de la matriz del sistema lineal (Fig. 2), permitiendo inspeccionar el patrón de dispersión (sparsity pattern) de las cinco diagonales no nulas.

7.4. Entorno de Ejecución

Para garantizar la reproducibilidad de los resultados, todo el proyecto ha sido configurado para ejecutarse dentro de un **entorno virtual de Python (venv)**. Esto asegura el aislamiento de las dependencias y la compatibilidad de versiones.

Requisitos:

- Python 3.8+
- Entorno virtual activado: `source venv/bin/activate`
- Dependencias principales: `numpy`, `scipy`, `matplotlib`

Repositorio: El código fuente completo está disponible en:

<https://github.com/ErvinCaraval/proyecto-final-simulacion.git>

8. Conclusiones

Este trabajo desarrolló exitosamente una implementación computacional completa para simular flujo incompresible bidimensional con las ecuaciones de Navier-Stokes. Los principales logros son:

1. La discretización por diferencias finitas centradas transformó efectivamente el problema continuo en un sistema algebraico con nueve patrones de ecuaciones.

2. El método de Newton-Raphson con malla optimizada (5×50) demostró alta eficiencia, convergiendo en 97 iteraciones.
3. El Gradiente Conjugado aplicado al sistema simetrizado $\mathbf{J}^T \mathbf{J}$ fue robusto, aprovechando la estructura dispersa (4.76 % densidad).
4. El número de condición $\kappa \approx 3,07$ garantizó estabilidad numérica y minimizó errores de redondeo.
5. Se desarrollaron dos versiones de interpolación: V4 con spline cúbico natural 1D manual (dos etapas) y V4.1 con spline bicúbico de SciPy (simultáneo), demostrando que V4.1 ofrece mayor suavidad y continuidad.
6. Los resultados son físicamente coherentes: aceleración entre obstáculos, recirculación tras vigas, y conservación de masa.
7. El número de Reynolds ($Re = 5,00$) confirmó régimen laminar, validando las hipótesis del modelo.
8. Las líneas de corriente revelaron claramente los patrones de flujo y zonas de recirculación.

Como trabajo futuro se sugiere: estudio de independencia de malla, extensión a Reynolds más altos con modelos de turbulencia, geometrías más complejas, y comparación con métodos de elementos finitos.

Referencias

- [1] R. Landau and M. Paez, *Computational Problems for Physics: With Guided Solutions Using Python*. Series in Computational Physics, CRC Press, 2018.
- [2] D. Kincaid and W. Cheney, *Numerical Analysis: Mathematics of Scientific Computing*. Third Edition, The Sally Series, Pure and Applied Undergraduate Texts, American Mathematical Society, 2002.

Anexo A: Derivación de la Ecuación Discretizada

Partiendo de la ecuación de Navier-Stokes para la componente horizontal:

$$\nu \left(\frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2} \right) = v_x \frac{\partial v_x}{\partial x} + v_y \frac{\partial v_x}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial x} \quad (14)$$

Asumiendo presión independiente de densidad y temperatura, y aplicando diferencias finitas centradas con $h = 1$:

$$\nu \left[\frac{v_{i+1,j}^x - 2v_{i,j}^x + v_{i-1,j}^x}{h^2} + \frac{v_{i,j+1}^x - 2v_{i,j}^x + v_{i,j-1}^x}{h^2} \right] = v_{i,j}^x \frac{v_{i+1,j}^x - v_{i-1,j}^x}{2h} + v_{i,j}^y \frac{v_{i,j+1}^x - v_{i,j-1}^x}{2h} \quad (15)$$

Con $\nu = 1$ y $h = 1$, simplificando y despejando $v_{i,j}^x$:

$$4v_{i,j}^x = v_{i+1,j}^x + v_{i-1,j}^x + v_{i,j+1}^x + v_{i,j-1}^x - 2v_{i,j}^x [v_{i+1,j}^x - v_{i-1,j}^x] - 2v_{i,j}^y [v_{i,j+1}^x - v_{i,j-1}^x] \quad (16)$$

Finalmente:

$$v_{i,j}^x = \frac{1}{4} [v_{i+1,j}^x + v_{i-1,j}^x + v_{i,j+1}^x + v_{i,j-1}^x] - \frac{1}{2} v_{i,j}^x [v_{i+1,j}^x - v_{i-1,j}^x] - \frac{1}{2} v_{i,j}^y [v_{i,j+1}^x - v_{i,j-1}^x] \quad (17)$$

Anexo B: Cálculo del Jacobiano

Para calcular la matriz Jacobiana, derivamos la ecuación discretizada respecto a cada variable. Considerando la ecuación para el nodo (i, j) :

$$F_{i,j} = v_{i,j}^x - \frac{1}{4} [v_{i+1,j}^x + v_{i-1,j}^x + v_{i,j+1}^x + v_{i,j-1}^x] + \text{términos no lineales} \quad (18)$$

Las derivadas parciales son:

$$\frac{\partial F_{i,j}}{\partial v_{i,j}^x} = 1 + \frac{1}{2} [v_{i+1,j}^x - v_{i-1,j}^x] + \text{otros términos} \quad (19)$$

$$\frac{\partial F_{i,j}}{\partial v_{i+1,j}^x} = -\frac{1}{4} + \frac{1}{2} v_{i,j}^x \quad (20)$$

$$\frac{\partial F_{i,j}}{\partial v_{i-1,j}^x} = -\frac{1}{4} - \frac{1}{2} v_{i,j}^x \quad (21)$$

$$\frac{\partial F_{i,j}}{\partial v_{i,j+1}^x} = -\frac{1}{4} + \frac{1}{2} v_{i,j}^y \quad (22)$$

$$\frac{\partial F_{i,j}}{\partial v_{i,j-1}^x} = -\frac{1}{4} - \frac{1}{2} v_{i,j}^y \quad (23)$$

Para la implementación computacional, se utiliza la siguiente notación simplificada para los coeficientes de la matriz Jacobiana, correspondientes a la diagonal principal y sus vecinos, tal como se muestra en la siguiente figura:

- **Diagonal Principal** (i, j) : $\frac{\partial F_{i,j}}{\partial v_x(i,j)} \approx a_i$
- **Vecino Izquierdo** $(i-1, j)$: $\frac{\partial F_{i,j}}{\partial v_x(i-1,j)} \approx e$
- **Vecino Superior** $(i, j+1)$: $\frac{\partial F_{i,j}}{\partial v_x(i,j+1)} \approx b$
- **Vecino Inferior** $(i, j-1)$: $\frac{\partial F_{i,j}}{\partial v_x(i,j-1)} \approx c$
- **Vecino Derecho** $(i+1, j)$: $\frac{\partial F_{i,j}}{\partial v_x(i+1,j)} \approx d$

Figura 8: *
Notación de coeficientes del Jacobiano.

Estas cinco derivadas no nulas por ecuación generan la estructura dispersa de banda observada en la Fig. 2.

Anexo C: Ecuaciones Discretizadas en Nodos Específicos

A continuación se presentan las ecuaciones discretizadas explícitas para 9 nodos representativos de la malla, mostrando la aplicación de las condiciones de frontera (donde $\mathbf{1}$, $\mathbf{0}$ y $\mathbf{V_0}$ representan valores de frontera):

Nodo (1,1)

$$v_{1,1}^x = \frac{1}{4} \left\{ v_{2,1}^x + \mathbf{1} + v_{1,2}^x + \mathbf{0} - \frac{1}{2} v_{1,1}^x (v_{2,1}^x - \mathbf{1}) - \frac{1}{2} v_{1,1}^y (v_{1,2}^x - \mathbf{0}) \right\}$$

Nodo (1,20)

$$v_{1,20}^x = \frac{1}{4} \left\{ v_{2,20}^x + \mathbf{1} + v_{1,21}^x + v_{1,19}^x - \frac{1}{2} v_{1,20}^x (v_{2,20}^x - \mathbf{1}) - \frac{1}{2} v_{1,20}^y (v_{1,21}^x - v_{1,19}^x) \right\}$$

Nodo (1,39)

$$v_{1,39}^x = \frac{1}{4} \left\{ v_{2,39}^x + \mathbf{1} + \mathbf{V_0} + v_{1,38}^x - \frac{1}{2} v_{1,39}^x (v_{2,39}^x - \mathbf{1}) - \frac{1}{2} v_{1,39}^y (\mathbf{V_0} - v_{1,38}^x) \right\}$$

Nodo (174,1)

$$v_{174,1}^x = \frac{1}{4} \left\{ v_{175,1}^x + v_{173,1}^x + v_{174,2}^x + \mathbf{0} - \frac{1}{2} v_{174,1}^x (v_{175,1}^x - v_{173,1}^x) - \frac{1}{2} v_{174,1}^y (v_{174,2}^x - \mathbf{0}) \right\}$$

Nodo (200,20)

$$v_{200,20}^x = \frac{1}{4} \left\{ v_{201,20}^x + v_{199,20}^x + v_{200,21}^x + v_{200,19}^x - \frac{1}{2} v_{200,20}^x (v_{201,20}^x - v_{199,20}^x) - \frac{1}{2} v_{200,20}^y (v_{200,21}^x - v_{200,19}^x) \right\}$$

Nodo (200,39)

$$v_{200,39}^x = \frac{1}{4} \left\{ v_{201,39}^x + v_{199,39}^x + \mathbf{V_0} + v_{200,38}^x - \frac{1}{2} v_{200,39}^x (v_{201,39}^x - v_{199,39}^x) - \frac{1}{2} v_{200,39}^y (\mathbf{V_0} - v_{200,38}^x) \right\}$$

Nodo (399,1)

$$v_{399,1}^x = \frac{1}{4} \left\{ \mathbf{0} + v_{398,1}^x + v_{399,2}^x + \mathbf{0} - \frac{1}{2} v_{399,1}^x (\mathbf{0} - v_{398,1}^x) - \frac{1}{2} v_{399,1}^y (v_{399,2}^x - \mathbf{0}) \right\}$$

Nodo (399,20)

$$v_{399,20}^x = \frac{1}{4} \left\{ \mathbf{0} + v_{398,20}^x + v_{399,21}^x + v_{399,19}^x - \frac{1}{2} v_{399,20}^x (\mathbf{0} - v_{398,20}^x) - \frac{1}{2} v_{399,20}^y (v_{399,21}^x - v_{399,19}^x) \right\}$$

Nodo (389,39)

$$v_{389,39}^x = \frac{1}{4} \left\{ v_{390,39}^x + v_{388,39}^x + \mathbf{V_0} + v_{389,38}^x - \frac{1}{2} v_{389,39}^x (v_{390,39}^x - v_{388,39}^x) - \frac{1}{2} v_{389,39}^y (\mathbf{V_0} - v_{389,38}^x) \right\}$$