Sesión 3. Esquemas numéricos en OpenFOAM

E. Martín[1], M. Meis[1,2] y F. Varas[1]

[1]Univ. de Vigo y [2]Vicus Desarrollos Tecnológicos

Dinámica de fluidos computacional con OpenFOAM
18–20 de Junio de 2014

Proyecto CloudPYME

El proyecto CloudPYME (ID 0682_CLOUDPYME2_1_E) está cofinanciado por la Comisión Europea a través del Fondo Europeo de Desarrollo Regional (FEDER), dentro de la tercera convocatoria de proyectos del Programa Operativo de Cooperación Transfronteriza España–Portugal 2007–2013 (POCTEP).

Esquemas numéricos en OpenFOAM

Discretización numérica

- discretización espacial y temporal
  - discretización espacial
  - tratamiento de términos convectivos
  - discretización temporal
- algoritmos globales de resolución
  - SIMPLE
  - PISO
- resolución de sistemas lineales

Outline

Plan

Formulación de modelos

Ecuación de conservación genérica (forma diferencial)

$$\underbrace{\frac{\partial}{\partial t}(\rho\Phi)}_{\text{acumulación}} + \underbrace{div(\rho\vec{U}\Phi)}_{\text{convección}} - \underbrace{div(\rho\Gamma_\Phi\vec{\nabla}\Phi)}_{\text{difusión}} = \underbrace{S(\Phi)}_{\text{fuente}}$$

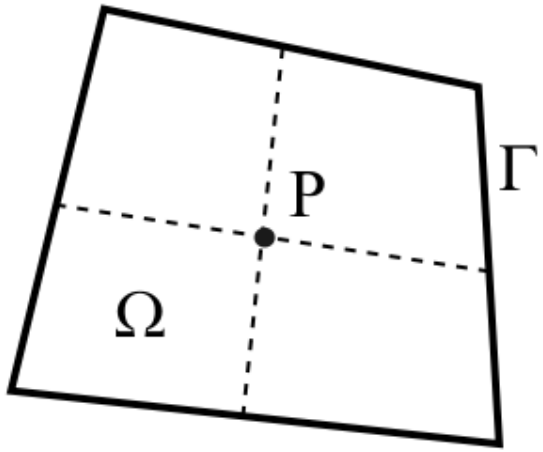
Formulación integral: balance sobre volumen de control V

$$\frac{d}{dt}\int_V \rho\Phi dV + \int_{\partial V} \rho\vec{U}\Phi\cdot d\vec{S} - \int_{\partial V} \rho\Gamma_\Phi\vec{\nabla}\Phi\cdot d\vec{S} = \int_V S(\Phi)dV$$

Discretización mediante volúmenes finitos

Formulación de balance sobre celda Ω

$$\frac{d}{dt}\int_{\Omega}\rho\Phi dV+\int_{\Gamma}\rho\vec{U}\Phi\cdot d\vec{S}-\int_{\Gamma}\rho\Gamma_{\Phi}\vec{\nabla}\Phi\cdot d\vec{S}=\int_{\Omega}S(\Phi)dV$$

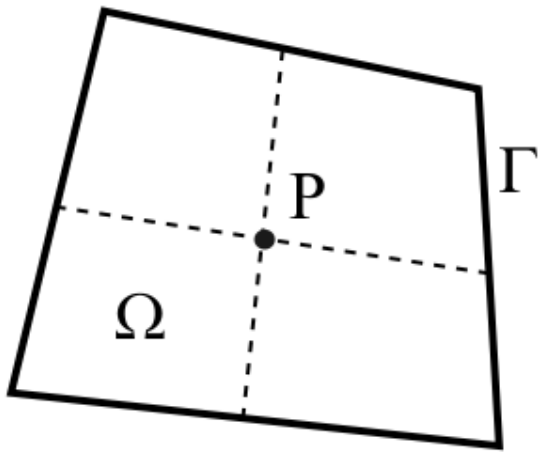


Discretización mediante volúmenes finitos

Aproximación mediante volúmenes finitos

A partir de aproximaciones de Φ en P se debe obtener:

- aproximación de $\int_\Omega F(\Phi) dV$
- aproximación de $\int_\Gamma \vec{G}(\Phi) \cdot d\vec{S}$

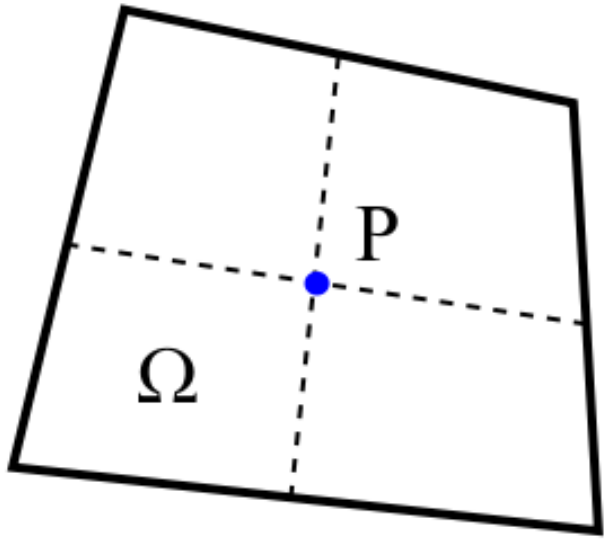


Integración sobre celdas

Aproximación de integral sobre celda Ω

$$F(\vec{x}) = F_P + (\vec{\nabla}F)_P \cdot (\vec{x} - \vec{x}_P) + O(||\vec{x} - \vec{x}_P||^2)$$

$$\int_\Omega F d\Omega \simeq \int_\Omega (F_P + (\vec{\nabla}F)_P \cdot (\vec{x} - \vec{x}_P)) d\Omega = F_P V_P$$

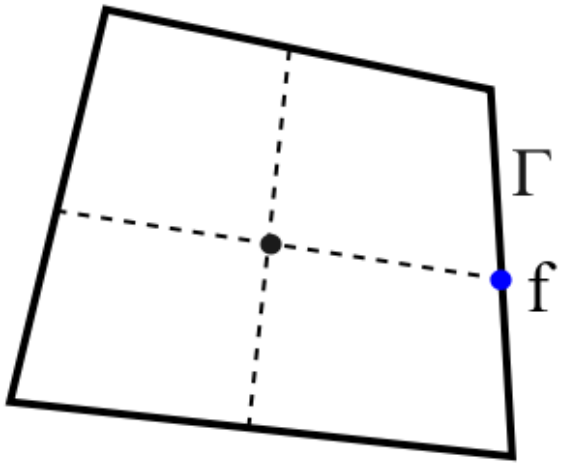


Integración sobre caras

Aproximación de integral sobre cara Γ

$$\vec{G}(\vec{x}) = \vec{G}_f + (\vec{\nabla}\vec{G})_f(\vec{x} - \vec{x}_f) + O(||\vec{x} - \vec{x}_f||^2)$$

$$\int_\Gamma \vec{G} \cdot d\vec{S} \simeq \int_\Gamma (\vec{G}_f + (\vec{\nabla}\vec{G})_f(\vec{x} - \vec{x}_f)) \cdot d\vec{S} = \vec{G}_f \cdot \vec{S}_f$$

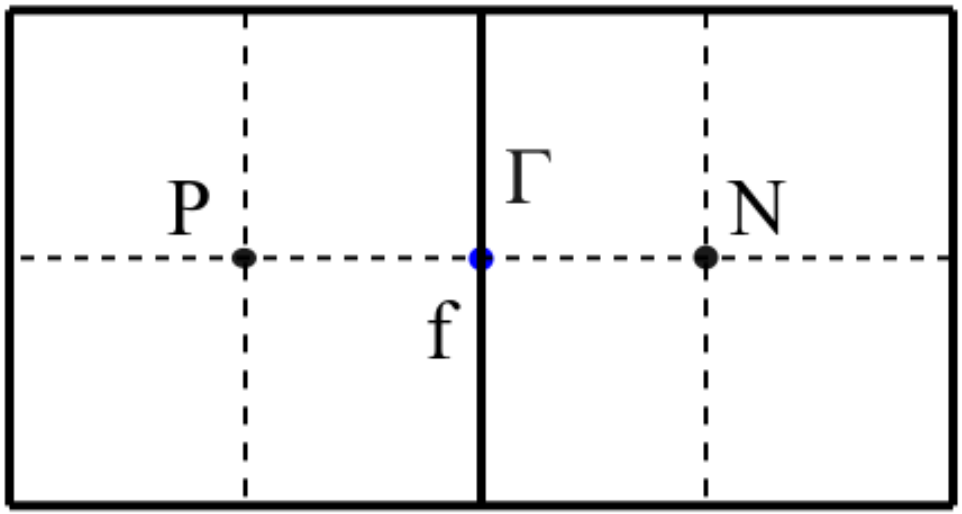


Aproximación de términos difusivos

Cálculo de flujo difusivo

$$\int_{\Gamma} \rho \Gamma_{\Phi} \vec{\nabla} \Phi \cdot d\vec{S} \simeq (\rho \Gamma_{\Phi})_f (\vec{\nabla} \Phi)_f \cdot \vec{S}_f$$

$$(\vec{\nabla} \Phi)_f \cdot \vec{S}_f \simeq |\vec{S}_f| \frac{\Phi_N - \Phi_P}{|\vec{d}|}$$

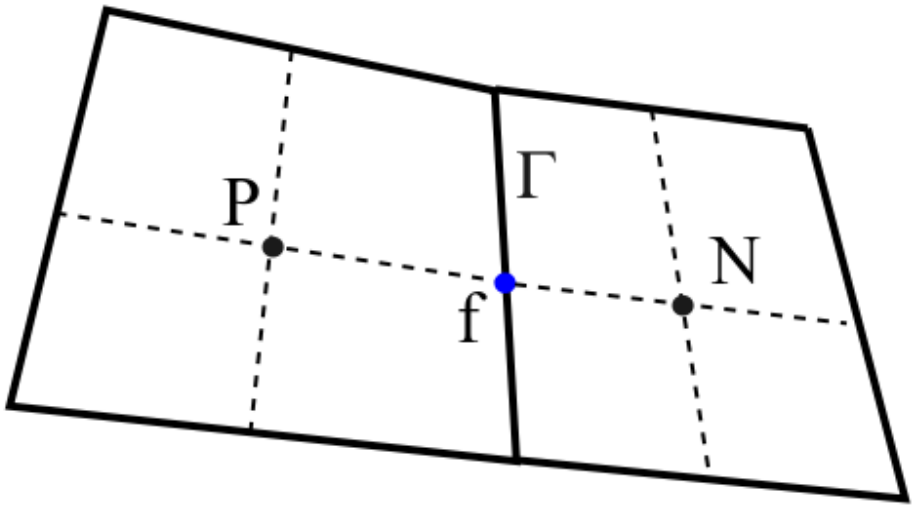


Aproximación de términos difusivos

Cálculo de flujo difusivo (malla general)

$$\int_\Gamma \rho\Gamma_\Phi \vec{\nabla}\Phi \cdot d\vec{S} \simeq (\rho\Gamma_\Phi)_f (\vec{\nabla}\Phi)_f \cdot \vec{S}_f$$

$$(\vec{\nabla}\Phi)_f = f_x(\vec{\nabla}\Phi)_P + (1-f_x)(\vec{\nabla}\Phi)_N \quad \text{con} \quad (\vec{\nabla}\Phi)_P \simeq \frac{1}{V_P}\sum_f \Phi_f \vec{S}_f$$

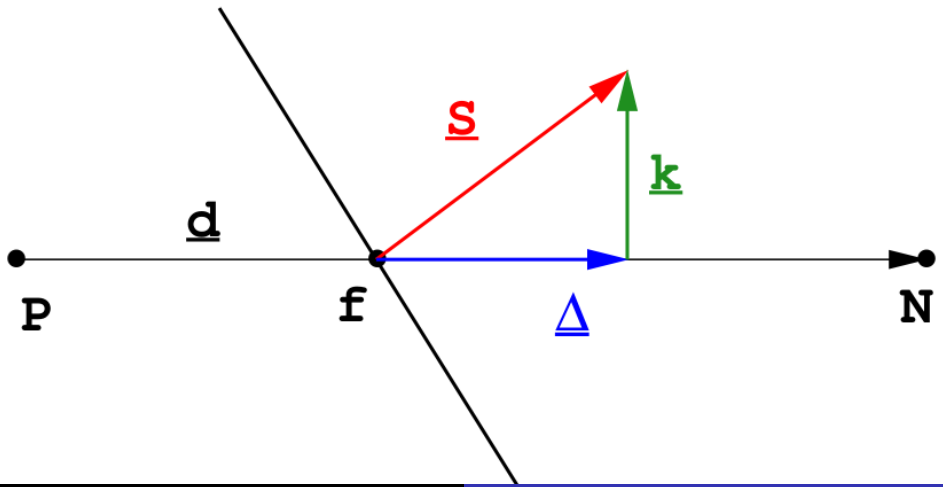


Aproximación de términos difusivos

Implementación práctica

- Esquema anterior produce *stencil* grande
  (matrices menos huecas y mayor consumo de memoria)
- En práctica se introducen correcciones

$$(\vec{\nabla}\Phi)_f \cdot \vec{S}_f = \underbrace{\vec{\Delta} \cdot (\vec{\nabla}\Phi)_f} + \underbrace{\vec{k} \cdot (\vec{\nabla}\Phi)_f}$$



Aproximación de términos fuente

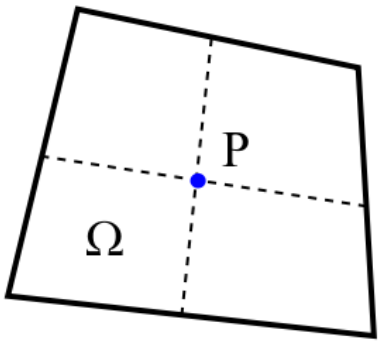
Cálculo de términos fuente

- términos lineales:

$$\int_{\Omega} S(\Phi) dV \simeq (S(\Phi))_P V_P = S_p \Phi_P V_P$$

- términos no lineales (linealización):

$$\int_{\Omega} S(\Phi) dV \simeq (S(\Phi))_P V_P \simeq S_{0P} V_P + S_{1P} \Phi_P V_P$$



Aproximación de términos convectivos

Un ejemplo elemental

Problema estacionario unidimensional sin fuentes

$$\frac{d}{dx}(\rho U \Phi) - \frac{d}{dx}(\rho \Gamma_\Phi \frac{d\Phi}{dx}) = 0$$

$$\Phi(0) = \Phi_0 \qquad \Phi(L) = 0$$

Forma adimensional (ρ, U y Γ_Φ constantes)

$$\frac{d\hat{\Phi}}{d\hat{x}} - \frac{1}{Pe}\frac{d^2\hat{\Phi}}{d\hat{x}^2} = 0$$

$$\hat{\Phi}(0) = 1 \qquad \hat{\Phi}(1) = 0$$

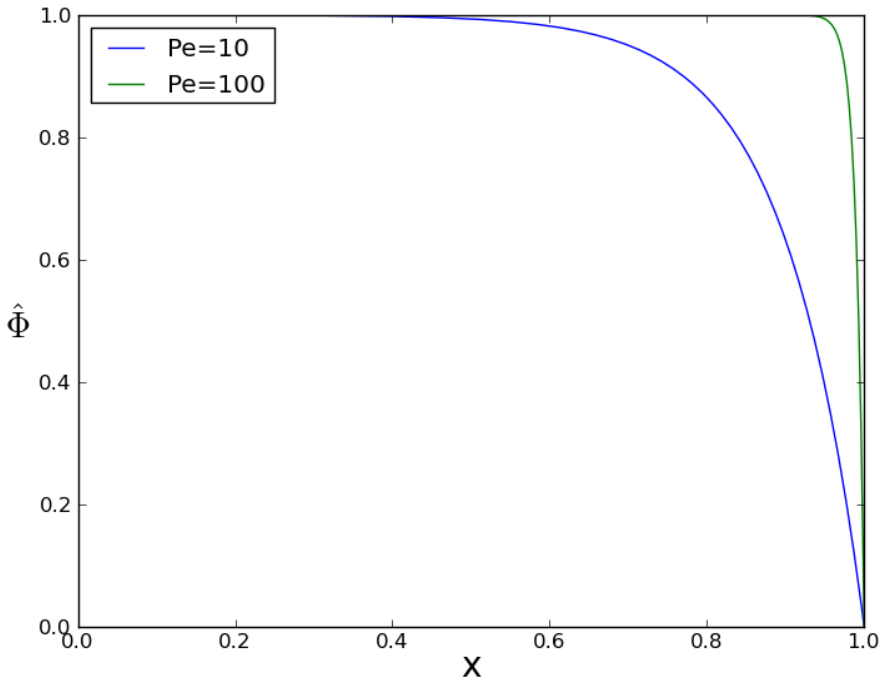
donde $Pe = \frac{LU}{\Gamma_\Phi}$

Aproximación de términos convectivos

Un ejemplo elemental

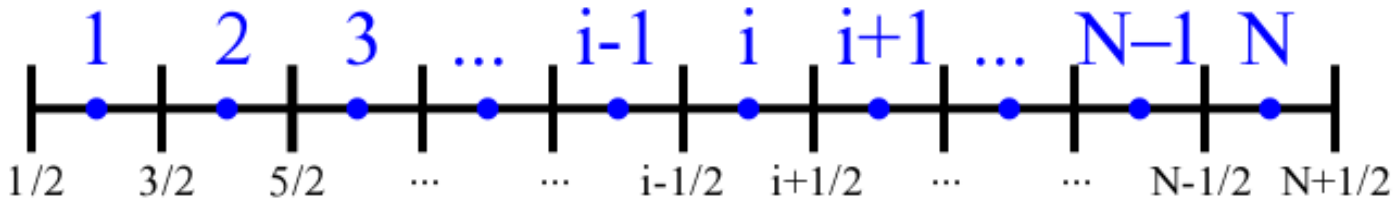
Solución analítica de problema modelo

$$\hat{\Phi}(x) = \frac{exp(Pe\ x) - exp(Pe)}{1 - exp(Pe)}$$



Aproximación de términos convectivos

Un ejemplo elemental

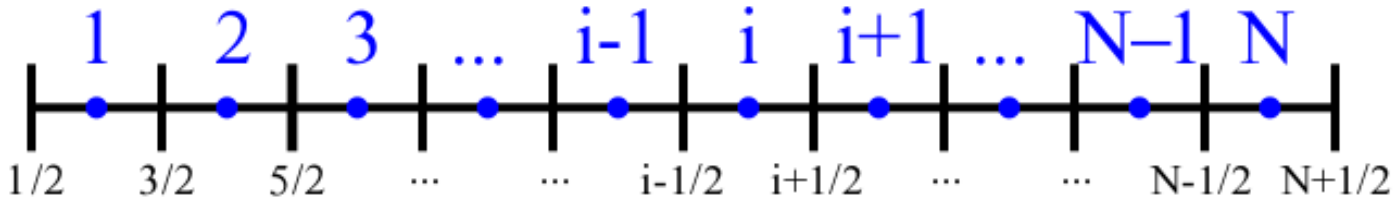


Formulación de balances sobre celda *i*–ésima

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \frac{d\hat{\Phi}}{d\hat{x}} d\hat{x} - \int_{x_{i-1/2}}^{x_{i+1/2}} \frac{1}{Pe} \frac{d}{d\hat{x}} \left(\frac{d\hat{\Phi}}{d\hat{x}}\right) d\hat{x} = 0$$

Aproximación de términos convectivos

Un ejemplo elemental



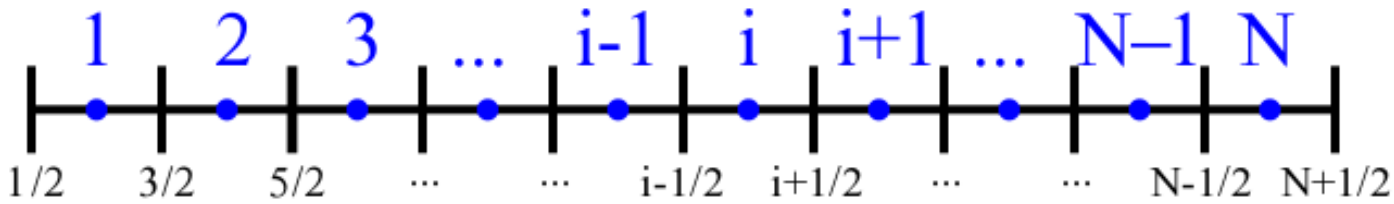
Tratamiento de términos difusivos

$$-\int_{x_{i-1/2}}^{x_{i+1/2}} \frac{1}{Pe}\frac{d}{d\hat{x}}\left(\frac{d\hat{\Phi}}{d\hat{x}}\right)d\hat{x} = \frac{1}{Pe}\left(\frac{d\hat{\Phi}}{d\hat{x}}\Big|_{x_{i+1/2}} - \frac{d\hat{\Phi}}{d\hat{x}}\Big|_{x_{i-1/2}}\right)$$

$$\simeq \frac{1}{Pe}\left(\frac{\hat{\Phi}_{i+1}-\hat{\Phi}_i}{\Delta\hat{x}} - \frac{\hat{\Phi}_i-\hat{\Phi}_{i-1}}{\Delta\hat{x}}\right) = \frac{1}{Pe}\frac{\hat{\Phi}_{i+1}-2\hat{\Phi}_i+\hat{\Phi}_{i-1}}{\Delta\hat{x}}$$

Aproximación de términos convectivos

Un ejemplo elemental



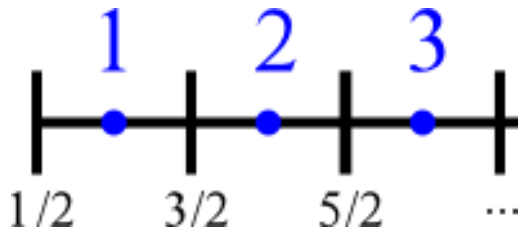
Aproximación de términos convectivos

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \frac{d\hat{\Phi}}{d\hat{x}} d\hat{x} = \hat{\Phi}|_{x_{i+1/2}} - \hat{\Phi}|_{x_{i-1/2}}$$

$$\simeq \frac{\hat{\Phi}_{i+1} + \hat{\Phi}_i}{2} - \frac{\hat{\Phi}_i + \hat{\Phi}_{i-1}}{2} = \frac{\hat{\Phi}_{i+1} - \hat{\Phi}_{i-1}}{2}$$

Aproximación de términos convectivos

Un ejemplo elemental



Balance en primera celda (condición de contorno en $x_{1/2}$)

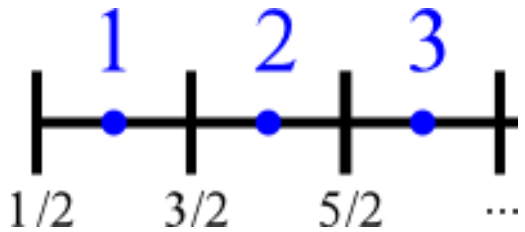
- Contribución de término difusivo

$$-\int_{x_{1/2}}^{x_{3/2}} \frac{1}{Pe}\frac{d}{d\hat{x}}\left(\frac{d\hat{\Phi}}{d\hat{x}}\right)d\hat{x} = \frac{1}{Pe}\left(\frac{d\hat{\Phi}}{d\hat{x}}|_{x_{3/2}} - \frac{d\hat{\Phi}}{d\hat{x}}|_{x_{1/2}}\right)$$

$$\simeq \frac{1}{Pe}\left(\frac{\hat{\Phi}_2 - \hat{\Phi}_1}{\Delta\hat{x}} - \frac{\hat{\Phi}_1 - \hat{\Phi}_{cc}}{\frac{1}{2}\Delta\hat{x}}\right)$$

Aproximación de términos convectivos

Un ejemplo elemental



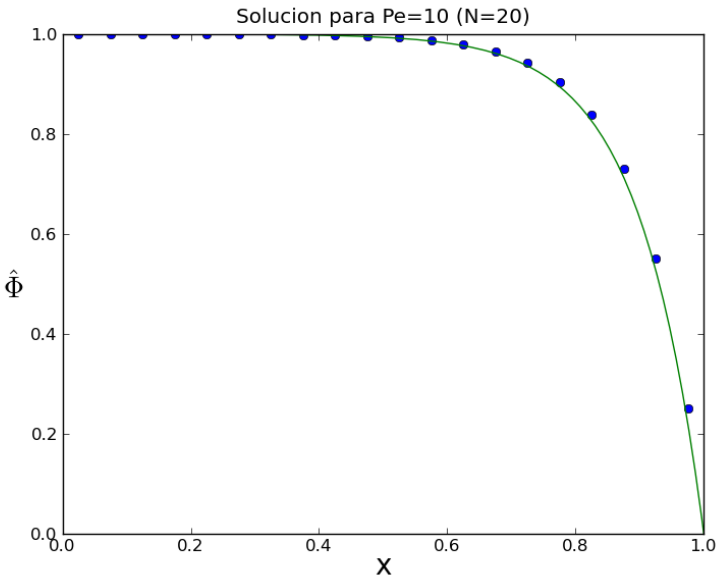
Balance en primera celda (condición de contorno en $x_{1/2}$)

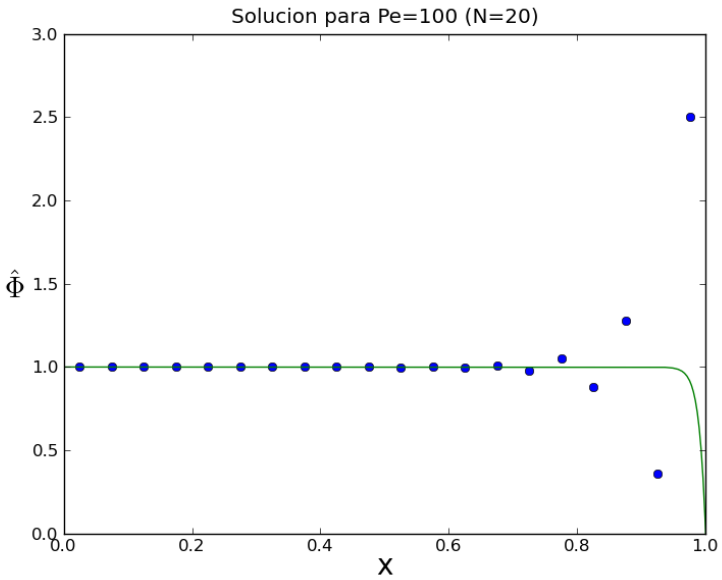
- Contribución de término convectivo

$$\int_{x_{1/2}}^{x_{3/2}} \frac{d\hat{\Phi}}{d\hat{x}} d\hat{x} = \hat{\Phi}|_{x_{3/2}} - \hat{\Phi}|_{x_{1/2}} \simeq \frac{\hat{\Phi}_1 + \hat{\Phi}_2}{2} - \hat{\Phi}_{cc}$$

Aproximación de términos convectivos

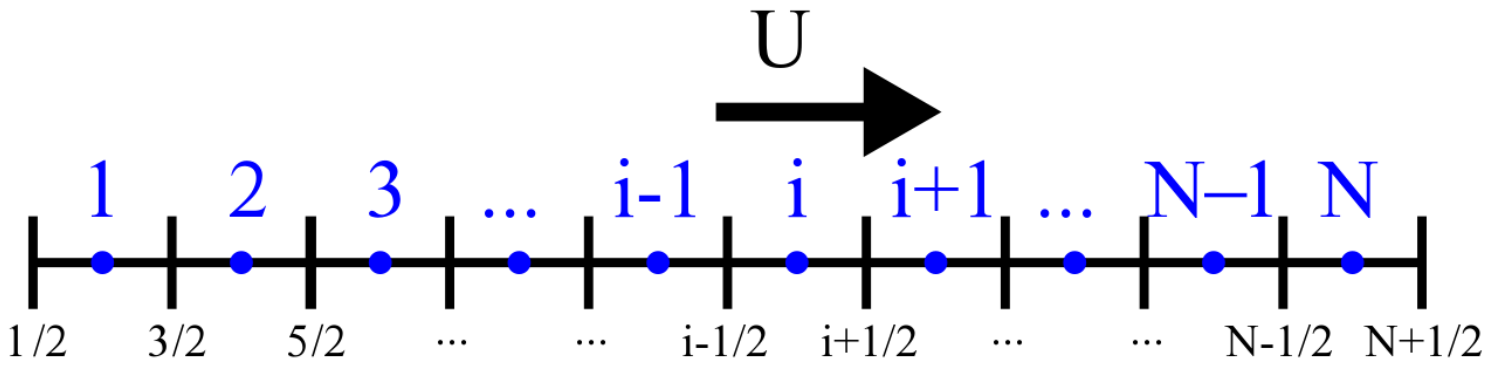
Un ejemplo elemental





Aproximación de términos convectivos

Un ejemplo elemental

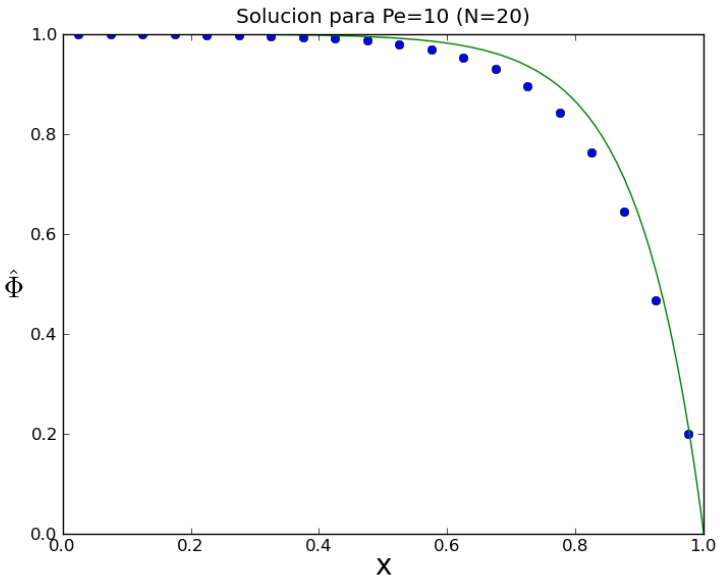


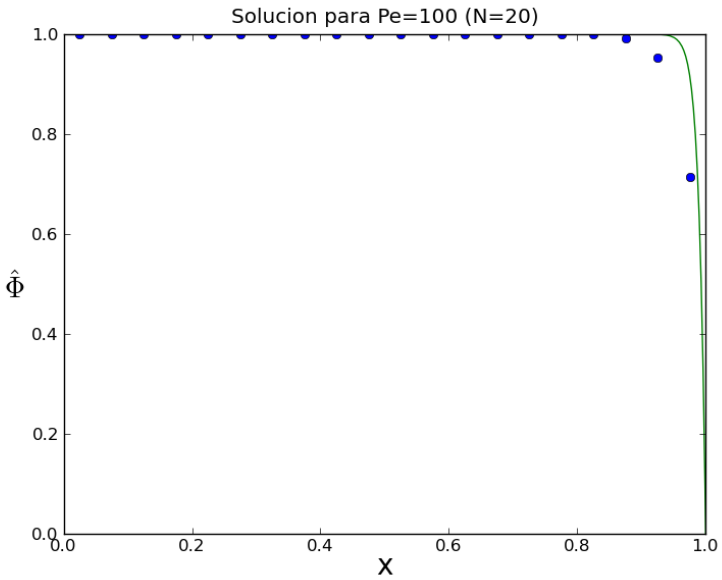
Una aproximación estable de términos convectivos

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \frac{d\hat{\Phi}}{d\hat{x}} d\hat{x} = \hat{\Phi}|_{x_{i+1/2}} - \hat{\Phi}|_{x_{i-1/2}} \simeq \hat{\Phi}_i - \hat{\Phi}|_{i-1}$$

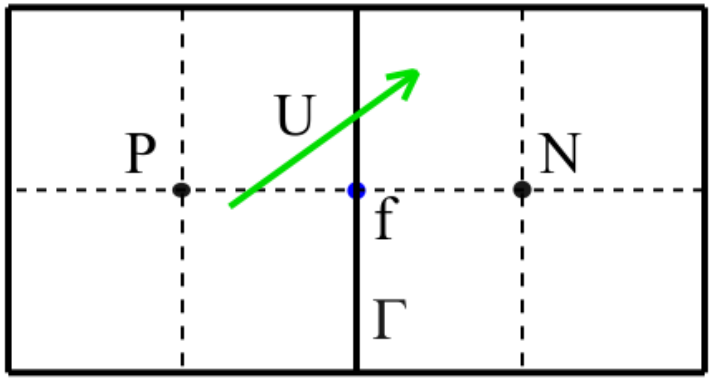
Aproximación de términos convectivos

Un ejemplo elemental





Aproximación de flujos convectivos

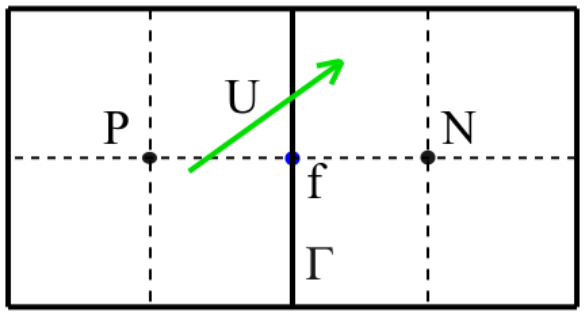


Contribución de flujos convectivos

$$\frac{d}{dt}\int_{\Omega}\rho\Phi dV+\underbrace{\int_{\Gamma}\rho\vec{U}\Phi\cdot d\vec{S}}-\int_{\Gamma}\rho\Gamma_{\Phi}\vec{\nabla}\Phi\cdot d\vec{S}=\int_{\Omega}S(\Phi)dV$$

$$\int_{\Gamma}\rho\vec{U}\Phi\cdot d\vec{S}\simeq(\rho\vec{U})_{f}\Phi_{f}\cdot\vec{S}_{f}$$

Aproximación de flujos convectivos



Esquemas básicos

- aproximación centrada (*central differencing*)

$$\Phi_f = (\Phi_f)_{CD} = f_x \Phi_P + (1 - f_x)\Phi_N$$

- aproximación descentrada (*upwind differencing*)

$$\Phi_f = (\Phi_f)_{UD} = \Phi_P$$

- aproximación mixta (*blended differencing*)

$$\Phi_f = (1 - \gamma)(\Phi)_{UD} + \gamma(\Phi)_{CD}$$

Aproximación de flujos convectivos

Otras aproximaciones de flujos convectivos

- descentrado de orden superior:
  - *linear upwinding*
  - QUICK
- esquemas TVD y limitadores de flujo:
  - van Leer
  - SUPERBEE
  - MINMOD
  - MUSCL

Control de discretización espacial en OpenFOAM

Tipos de esquemas de discretización espacial

- Interpolación general (obtiene valores sobre aristas/caras)

  `interpolationSchemes`
- Discretización de términos convectivos:

  `divSchemes`
- Discretización de términos difusivos:

  `laplacianSchemes`
- Discretización de términos en gradiente:

  `gradSchemes`

Control de discretización espacial

Esquemas de interpolación sobre aristas/cara

Centred schemes	
`linear`	Linear interpolation (central differencing)
`cubicCorrection`	Cubic scheme
`midPoint`	Linear interpolation with symmetric weighting
Upwinded convection schemes	
`upwind`	Upwind differencing
`linearUpwind`	Linear upwind differencing
`skewLinear`	Linear with skewness correction
`filteredLinear2`	Linear with filtering for high-frequency ringing
TVD schemes	
`limitedLinear`	limited linear differencing
`vanLeer`	van Leer limiter
`MUSCL`	MUSCL limiter
`limitedCubic`	Cubic limiter
NVD schemes	
`SFCD`	Self-filtered central differencing
`Gamma` ψ	Gamma differencing

Control de discretización espacial

Términos convectivos: `divSchemes`

- Discretización de términos de forma $\mathrm{div}(\rho\vec{U}\vec{U})$ mediante:

  `div(phi,U) Gauss` *`InterpScheme`*

  donde `phi` corresponde a $\vec{\phi} = \rho\vec{U}$
- Esquemas (usuales) de interpolación:

Scheme	Numerical behaviour
`linear`	Second order, unbounded
`skewLinear`	Second order, (more) unbounded, skewness correction
`cubicCorrected`	Fourth order, unbounded
`upwind`	First order, bounded
`linearUpwind`	First/second order, bounded
`QUICK`	First/second order, bounded
TVD schemes	First/second order, bounded
`SFCD`	Second order, bounded
NVD schemes	First/second order, bounded

Control de discretización espacial

Términos difusivos: `laplacianSchemes`

- Discretización de términos de forma $\mathrm{div}(\nu\vec{\nabla}U)$ mediante:

  `laplacian(nu,U) Gauss` *`InterpScheme SnScheme`*
  - Esquema de interpolación según lista general
  - Esquemas de aproximación de normales:

Scheme	Numerical behaviour
`corrected`	Unbounded, second order, conservative
`uncorrected`	Bounded, first order, non-conservative
`limited` ψ	Blend of `corrected` and `uncorrected`
`bounded`	First order for bounded scalars
`fourth`	Unbounded, fourth order, conservative

- Ejemplo:

  `laplacian(nu,U) Gauss linear corrected`

Control de discretización espacial

Términos en gradiente: `gradSchemes`

Discretisation scheme	Description
`Gauss <interpolationScheme>`	Second order, Gaussian integration
`leastSquares`	Second order, least squares
`fourth`	Fourth order, least squares
`cellLimited <gradScheme>`	Cell limited version of one of the above schemes
`faceLimited <gradScheme>`	Face limited version of one of the above schemes

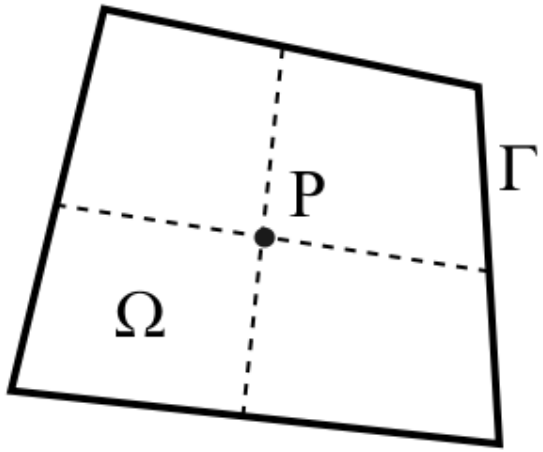
Más información

- *OpenFOAM User's Guide*
- *OpenFOAM Programmer's Guide*
- OpenFOAM wiki en: `http://openfoamwiki.net`
- H. Jasak; *Error Analysis and Estimation for the Finite Volume Method with Applications to Fluid Flows*, PhD Dissertation, 1996.

Integración temporal

Formulación de balance sobre celda Ω

$$\frac{d}{dt}\int_{\Omega}\rho\Phi dV+\int_{\Gamma}\rho\vec{U}\Phi\cdot d\vec{S}-\int_{\Gamma}\rho\Gamma_{\Phi}\vec{\nabla}\Phi\cdot d\vec{S}=\int_{\Omega}S(\Phi)dV$$



Integración temporal

Formulación semidiscretizada en espacio

$$\rho_P V_P \frac{d\Phi_P}{dt} + \sum_f F\Phi_f - \sum_f (\rho\Gamma)_f \vec{S} \cdot (\vec{\nabla}\Phi)_f = S_0 V_P + S_1 V_P \Phi_P$$

Esquemas de discretización temporal

- Euler implícito
- Esquema BDF de orden 2
- Crank-Nicolson
- esquemas de Runge-Kutta

Integración temporal

Método de Euler implícito

$$\rho_P V_P \frac{\Phi_P^n - \Phi_P^0}{\Delta t} + \sum_f F\Phi_f^n - \sum_f (\rho\Gamma)_f \vec{S} \cdot (\vec{\nabla}\Phi)_f^n = S_0 V_P + S_1 V_P \Phi_P^n$$

Estabilidad del esquema

- incondicionalmente estable
- sobreamortiguamiento si *CFL* mucho mayor que 1

Integración temporal

Método BDF de orden 2

$$\rho_P V_P \frac{3/2\Phi_P^n - 2\Phi_P^0 + 1/2\Phi_P^{00}}{\Delta t}$$

$$+\sum_f F\Phi_f^n - \sum_f (\rho\Gamma)_f \vec{S}\cdot(\vec{\nabla}\Phi)_f^n = S_0 V_P + S_1 V_P \Phi_P^n$$

Método de Crank–Nicolson

$$\rho_P V_P \frac{\Phi_P^n - \Phi_P^n}{\Delta t} + \frac{1}{2}\sum_f F\Phi_f^n - \frac{1}{2}\sum_f (\rho\Gamma)_f \vec{S}\cdot(\vec{\nabla}\Phi)_f^n$$

$$+\frac{1}{2}\sum_f F\Phi_f^n - \frac{1}{2}\sum_f (\rho\Gamma)_f \vec{S}\cdot(\vec{\nabla}\Phi)_f^n = S_0 V_P + \frac{1}{2}S_1 V_P \Phi_P^n + \frac{1}{2}S_1 V_P \Phi_P^0$$

Integración temporal

Implementación en OpenFOAM

- términos implícitos (ensamblado de matriz):

  `fvm`: finiteVolumeMethod
- términos explícitos (ensamblado de vector):

  `fvc`: finiteVolumeCalculus

Ecuación del calor en `laplacianFoam`

Integración con esquema implícito de

$$\frac{\partial T}{\partial t} - \mathrm{div}(D_T \vec{\nabla} T) = 0$$

```
solve ( fvm::ddt(T) - fvm::laplacian(DT, T) );
```

Plan

Dificultades en tratamiento de presión

Ecuaciones de Stokes

$$-\mu\Delta\vec{v} + \vec{\nabla}p = 0$$

$$\mathrm{div}\vec{v} = 0$$

Formulación sobre celda (2D)

$$-\sum_f \int_\Gamma \mu\frac{\partial u}{\partial n}\, dS + \sum_f \int_\Gamma p n_x\, dS = 0$$

$$-\sum_f \int_\Gamma \mu\frac{\partial v}{\partial n}\, dS + \sum_f \int_\Gamma p n_y\, dS = 0$$

$$\sum_f \int_\Gamma \vec{v}\cdot\vec{n}\, dS = 0$$

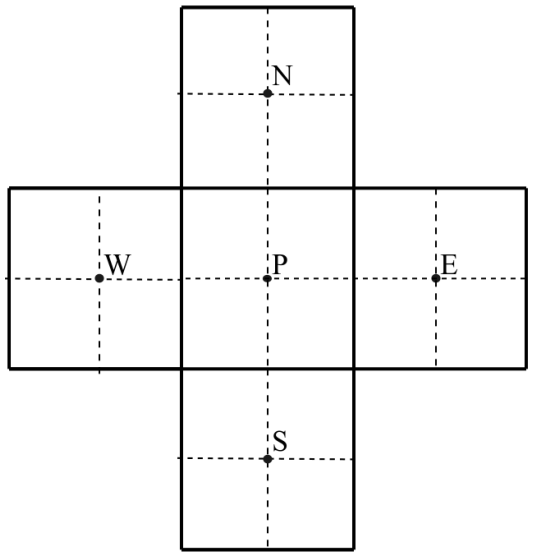
Dificultades en tratamiento de presión

Discretización de ecuación de Stokes

$$-\mu(u_E + u_W + u_N + u_S - 4u_P) + h(p_E - p_W) = 0$$

$$-\mu(v_E + v_W + v_N + v_S - 4v_P) + h(p_N - p_S) = 0$$

$$v_N - v_S + u_E - u_W = 0$$



Dificultades en tratamiento de presión

Ecuaciones de Stokes

$$-\mu\Delta\vec{v} + \vec{\nabla}p = 0$$

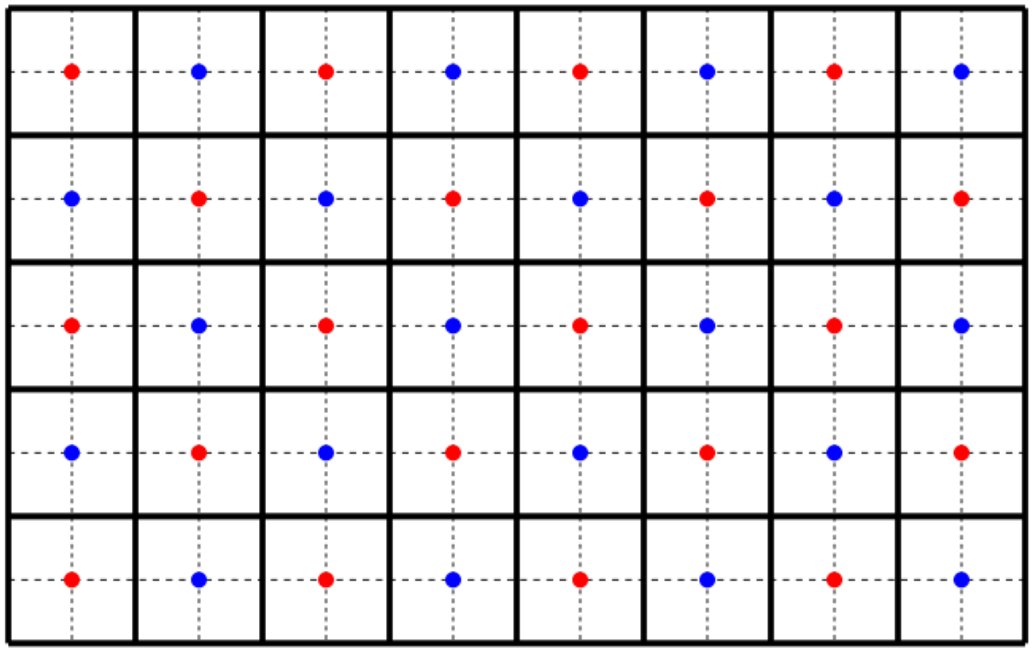
$$\mathrm{div}\vec{v} = 0$$

Observación

Con balance nulo de fuerzas de viscosidad ($\mu\Delta\vec{v} = 0$):
p debe ser constante

Dificultades en tratamiento de presión

$$-\mu(u_E + u_W + u_N + u_S - 4u_P) + h(p_E - p_W) = 0$$
$$-\mu(v_E + v_W + v_N + v_S - 4v_P) + h(p_N - p_S) = 0$$



Dificultades en tratamiento de presión

Alternativas de esquemas numéricos estables

- uso de mallas decaladas (*staggered grids*):
  - presión y velocidad **no** se aproximan en los mismos puntos
  - ejemplo: MAC (Marker–and–Cell)
- métodos de proyección (o segregados):
  - las ecuaciones se resuelven en varios pasos
  - se separa conservación de momentos e incompresibilidad
  - ejemplo: PISO y SIMPLE

Algoritmo SIMPLE

Semi–Implicit Method for Pressure–Linked Equations

Ecuaciones de Navier–Stokes incompresibles

$$\frac{\partial \vec{u}}{\partial t} + \mathrm{div}(\vec{u} \otimes \vec{u}) - \mathrm{div}(\nu \vec{\nabla} \vec{u}) + \vec{\nabla} p = \vec{0}$$

$$\mathrm{div} \vec{u} = 0$$

Idea del algoritmo SIMPLE (para problema estacionario)

- se itera en tiempo (linealizando en tiempo anterior)
- en capa paso de tiempo:
  - se descomponen los campos en predicción y corrección
  - se calcula una predicción de la velocidad (ec. de conservación de momentos con presión dada)
  - se calculan las correcciones en presión y velocidad (ec. de conservación de momentos e incompresibilidad)

Algoritmo SIMPLE

Descomposición de campos (predicción + corrección)

$$\vec{u}^{n+1} = \vec{u}^{*} + \vec{u}' \qquad\qquad p^{n+1} = p^{n} + p'$$

Ecuación de conservación de momentos

Despreciando términos cuadráticos y viscosos en $\vec{u}'$:

$$\frac{\partial \vec{u}^{*}}{\partial t} + \frac{\partial \vec{u}'}{\partial t} + \mathrm{div}(\vec{u}^{n} \otimes \vec{u}^{*}) - \mathrm{div}(\nu \vec{\nabla} \vec{u}^{*}) + \vec{\nabla} p^{n} + \vec{\nabla} p' = \vec{0}$$

Segregación de cálculos

$$\frac{\partial \vec{u}^{*}}{\partial t} + \mathrm{div}(\vec{u}^{n} \otimes \vec{u}^{*}) - \mathrm{div}(\nu \vec{\nabla} \vec{u}^{*}) + \vec{\nabla} p^{n} = \vec{0}$$

$$\frac{\partial \vec{u}'}{\partial t} + \vec{\nabla} p' = \vec{0}$$

Algoritmo SIMPLE

Etapa de predicción

- Se calcula $\vec{u}^*$ solución de

$$\mathrm{div}(\vec{u}^n \otimes \vec{u}^*) - \mathrm{div}(\nu \vec{\nabla} \vec{u}^*) + \vec{\nabla} p^n = \vec{0}$$

Etapa de corrección

- Se calculan $\vec{u}'$ y p' solución de

$$\frac{\partial \vec{u}'}{\partial t} + \vec{\nabla} p' = \vec{0}$$

$$\mathrm{div}(\vec{u}^* + \vec{u}') = 0$$

Algoritmo SIMPLE

Resolución de etapa de corrección

- Se deduce un problema de Poisson para la presión.
  Eliminando $\vec{u}'$ del sistema en $(\vec{u}', p')$:

$$-\Delta p' = -\frac{\partial}{\partial t}(\mathrm{div}\vec{u}^*)$$

  con condiciones de contorno *artificiales* sobre p'
- Calculada presión p' se calcula corrección de velocidad $\vec{u}'$:

$$\frac{\partial \vec{u}'}{\partial t} + \vec{\nabla} p' = \vec{0}$$

Algoritmo SIMPLE

Implementación del algoritmo SIMPLE

- La etapa de predicción puede simplificarse
  - p.e. convertirse en explícita (se retiene sólo diagonal del operador de discretización)
- El algoritmo itera en (pseudo–)tiempo hasta convergencia
  - justifica no considerar términos despreciados
- En cálculo de presión se itera para retener correcciones no–ortogonales
  - evita aumento de banda de matriz (reduce coste total)
- Se emplea relajación para mejorar la convergencia

Algoritmo SIMPLE

Implementación de SIMPLE en OpenFOAM

Archivo `simpleFoam.C` en directorio `applications/solvers/incompressible`

```
while (simple.loop())
{
    Info<< "Time = " << runTime.timeName() << nl << endl;

    // --- Pressure-velocity SIMPLE corrector
    {
        #include "UEqn.H"
        #include "pEqn.H"
    }

    turbulence->correct();

    runTime.write();

    Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
        << "  ClockTime = " << runTime.elapsedClockTime() << " s"
        << nl << endl;
}
```

Predicción de velocidades

Archivo `UEqn.H`

```
// Momentum predictor

tmp<fvVectorMatrix> UEqn
(
    fvm::div(phi, U)
  + turbulence->divDevReff(U)
  ==
    sources(U)
);

UEqn().relax();

sources.constrain(UEqn());

solve(UEqn() == -fvc::grad(p));
```

Etapa de corrección

Archivo `pEqn.H`

```
{
    p.boundaryField().updateCoeffs();

    volScalarField rAU(1.0/UEqn().A());
    U = rAU*UEqn().H();
    UEqn.clear();

    phi = fvc::interpolate(U, "interpolate(HbyA)") & mesh.Sf();
    adjustPhi(phi, U, p);

    // Non-orthogonal pressure corrector loop
    while (simple.correctNonOrthogonal())
    {
        fvScalarMatrix pEqn
        (
            fvm::laplacian(rAU, p) == fvc::div(phi)
        );

        pEqn.setReference(pRefCell, pRefValue);

        pEqn.solve();

        if (simple.finalNonOrthogonalIter())
        {
            phi -= pEqn.flux();
        }
    }

    #include "continuityErrs.H"

    // Explicitly relax pressure for momentum corrector
    p.relax();

    // Momentum corrector
    U -= rAU*fvc::grad(p);
    U.correctBoundaryConditions();
    sources.correct(U);
}
```

Algoritmo PISO

Pressure Implicit with Splitting of Operators

Adaptación de SIMPLE para problemas evolutivos

- términos despreciados (en cálculo de predicción $\vec{u}^*$) no se cancelan en transitorio
- términos truncados se incluyen en la corrección
- es preciso iterar en cálculo de correcciones

Esquema del algoritmo PISO

En cada paso de tiempo se resuelve:

- Cálculo de predicción $\vec{u}^*$ en t_{n+1}
- Bucle de cálculo de corrección $(\vec{u}', p')$ en t_{n+1}:
  - cálculo de p'_{k+1} a partir de $\vec{u}^*$ y $\vec{u}'_k$
  - corrección de velocidad $\vec{u}'_{k+1}$ a partir de p'_{k+1}

Implementación de PISO en OpenFOAM

Estructura de solver `icoFoam`

```
while (runTime.loop()) # bucle temporal
    fvVectorMatrix UEqn ( fvm::ddt(U)
      +fvm::div(phi,U)-fvm::laplacian(nu,U));
    solve(UEqn == -fvc::grad(p)); # prediccion
    # inicio bucle PISO:
    for (int corr=0; corr<nCorr; corr++)
        adjustPhi(phi, U, p);
        # inicio bucle calculo presion:
        for (int nonOrth=0; nonOrth<=...
            fvScalarMatrix pEqn (
                fvm::laplacian(rAU, p)
                == fvc::div(phi) );
            pEqn.solve();
        U -= rAU*fvc::grad(p); # correccion
```

Plan

Métodos (iterativos) de resolución

- métodos de Krylov:
  - método de gradiente conjugado
  - método de gradiente biconjugado
- métodos multimalla:
  - métodos multimalla geométricos
  - métodos multimalla algebraicos

Referencias

- `http://www.openfoam.com/features`
- OpenFOAM User Guide (sección 4.5)

Métodos de Krylov

Métodos disponibles

- para matrices simétricas y definidas positivas:
  - mét. de gradiente conjugado (precondicionado): `PCG`
- para matrices generales:
  - mét. de gradiente biconjugado (precondicionado): `PBiCG`

Precondicionadores

- factorizaciones incompletas: Cholesky (`DIC`) y LU (`DILU`)
- precondicionador diagonal (Jacobi): `diagonal`
- métodos multimalla (geométrico/algebraico): `GAMG`

Métodos multimalla

Métodos multimalla geométricos/algebraicos

- usuario no necesita proporcionar jerarquía de mallas
- malla grosera construida a partir de directrices en:

  `nCoarsestCells`, `agglomerator`
- control de suavizadores mediante:

  `smoother`

  `nPreSwepng`, `nPostSwepng`, `nFinestSwepng`
- control de niveles de multimalla con:

  `mergeLevels`

Un ejemplo: uso de `icoFoam`

Solver `icoFoam`

- *Transient solver for incompressible, laminar flow of Newtonian fluids*
- discretización basada en PISO
- usado en *Lid–driven cavity flow* (tutorial en *User's Guide*)

Código fuente de `icoFoam`

Archivo `icoFoam.C` en subdirectorio:
`applications/solvers/incompressible`

Un ejemplo: uso de `icoFoam`

Opciones en archivo `fvSolution`

```
solvers
{
    p
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0;
    }

    U
    {
        solver          PBiCG;
        preconditioner  DILU;
        tolerance       1e-05;
        relTol          0;
    }
}

PISO
{
    nCorrectors     2;
    nNonOrthogonalCorrectors 0;
    pRefCell        0;
    pRefValue       0;
}
```

Un ejemplo: uso de `icoFoam`

Opciones en archivo `fvSchemes`

```
ddtSchemes
{
    default         Euler;
}

gradSchemes
{
    default         Gauss linear;
    grad(p)         Gauss linear;
}

divSchemes
{
    default         none;
    div(phi,U)      Gauss linear;
}

laplacianSchemes
{
    default         none;
    laplacian(nu,U) Gauss linear corrected;
    laplacian((1|A(U)),p) Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
    interpolate(HbyA) linear;
}
```