SIMULACIÓN DE TRANSITORIOS CON SOLVER PIMPLEFOAM

TRANSIENT SIMULATION WITH PIMPLEFOAM SOLVER

Guillermo Rolle

Palabras clave: CFD, OpenFOAM, pimpleFoam

Resumen. Este documento provee una guía para realizar una simulación de estado transitorio en OpenFOAM utilizando el solver pimpleFoam. La malla es importada de un informe anterior al igual que las condiciones de contorno e iniciales. El informe provee también una breve explicación sobre técnicas de post-proceso y conclusiones sobre el método implementado.

Keywords: CFD, OpenFOAM, pimpleFoam

Abstract. This document provides a guide to perform a transient state simulation in OpenFOAM using the pimpleFoam solver. Mesh is imported from a previous report like boundary and initial conditions. The report also provides a brief explanation of post-processing techniques and conclusions about the implemented method.

1. INTRODUCCIÓN

En el informe anterior se hicieron simulaciones en 3D sobre el mezclador de agroquímicos. Los campos de velocidad y presión fueron obtenidos con un solver estacionario. Se había comprobado que el solver no era capaz de reducir los residuales al valor requerido por no poder obtenerse una solución estacionaria. En este informe se simula el mismo caso pero con el solver transitorio pimpleFoam. Se usa una copia de la malla previa y el informe se basará exclusivamente en las modificaciones del caso para ejecutar el nuevo solver.

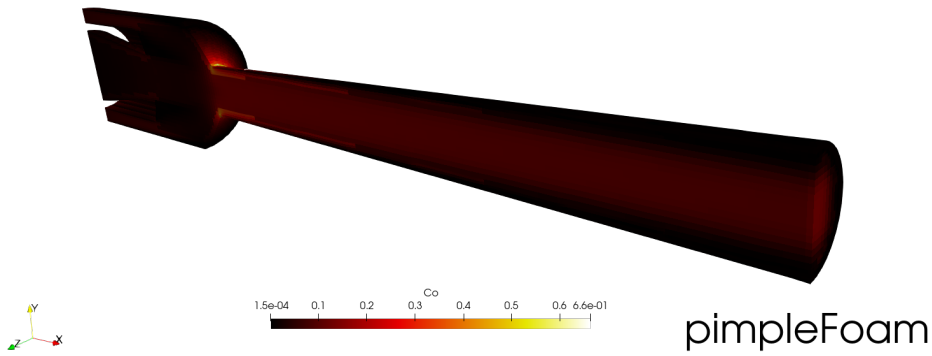

Figura 1: Caso 3D en cuestión

Este informe es parte de la serie de informes basados en un mezclador de agroquímicos en línea. El material correspondiente para este caso se encuentra en el repositorio online.

2. PREPARACIÓN DEL CASO

Usamos el caso 3D anterior como base para preparar el nuevo caso. Copiamos además el meshCase para generar la malla. Notar que no modificaremos los parámetros de mallado pero aprovecharemos el script Allrun que requiere de la existencia de este caso.

No son muchas las modificaciones que hay que hacer para pasar de una simulación con `simpleFoam` a una con `pimpleFoam`.

Primero, en el archivo controlDict debemos llenar el campo `application` con `pimpleFoam`. Además, en el solver estacionario el campo `deltaT` representa el paso en la numeración de iteraciones y no representa más que un número de iteración. En un solver transitorio en cambio, sí es importante ya que representa el paso de tiempo entre una iteración y la siguiente. Lo mismo aplica para el campo `endTime`. Para la simulación de transitorio simularemos un tiempo total de 2 segundos con paso de tiempo de 1ms como se ve en la figura 2. Por último, aumentamos la frecuencia de escritura para poder analizar mejor las variaciones en el tiempo.

Nota: Las cambios representados en las imágenes de este informe pueden diferir de las modificaciones finales reales. La última versión de cada archivo puede verse en el repositorio online.

En el archivo fvSchemes modificamos el campo `ddtSchemes` y agregamos una línea a `gradSchemes`. Los cambios se muestran en la figura 3.

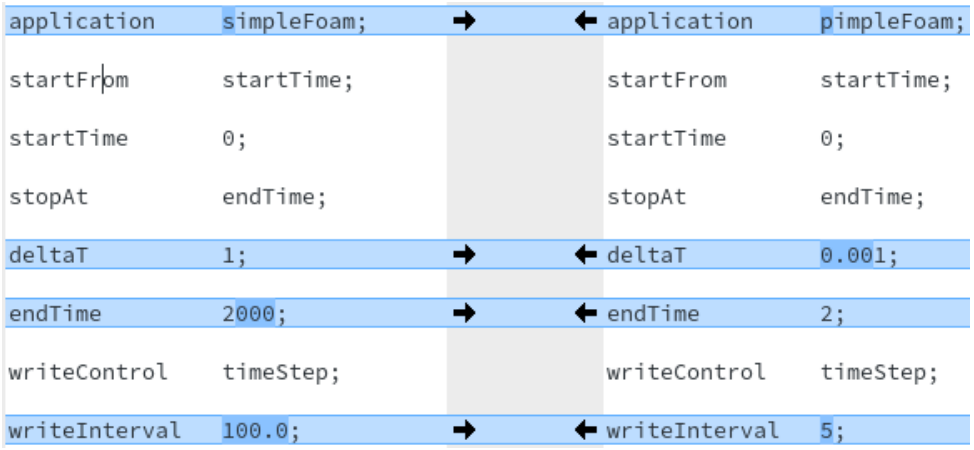

Figura 2: cambios en controlDict: caso estacionario (izquierda) vs caso transitorio (derecha)

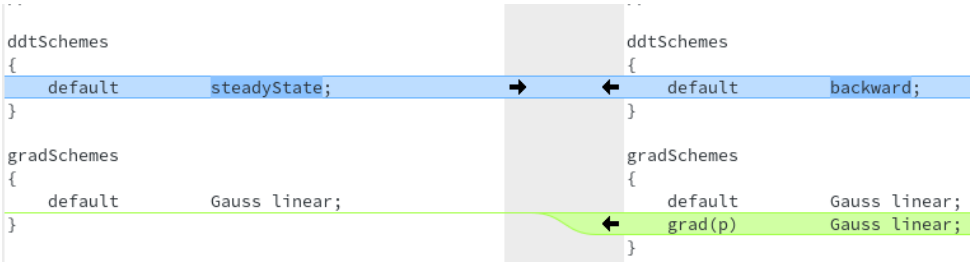

Figura 3: cambios en fvSchemes: caso estacionario (izquierda) vs caso transitorio (derecha)

Por último, modificamos las tolerancias de los solvers para `p` y `U` en el archivo fvSolution. Además, agregamos los modificadores para el campo `PIMPLE`. los cambios se muestran en la figura 4

Figura 4: cambios en fvSolution: caso estacionario (izquierda) vs caso transitorio (derecha)

3. SIMULACIÓN

Antes de poder ejecutar el archivo Allrun debemos modificarlo para que ejecute el solver `pimpleFoam` (ver figura 5). Además, recordar que en cada computadora donde se desee ejecutar este caso, se debe modificar el campo `FOAMDIR` del mismo archivo, y la cantidad de procesadores a utilizar con `mpirun`. Este último cambio debe corresponderse con el valor del campo `numberOfSubdomains` en el archivo decomposeParDict.

Figura 5: cambios en Allrun: caso estacionario (izquierda) vs caso transitorio (derecha)

Con esto estamos listos para ejecutar la simulación:

```
$ ./Allrun
```

Es posible que la simulación se ejecute durante un tiempo prolongado. Por suerte, podemos observar el caso en *ParaView* mientras aún se está ejecutando como se ve en la figura 6. Para esto, en otra terminal ir al directorio del caso y ejecutar:

```
$ paraFoam -builtin
```

Seleccionamos las opciones *"Decomposed Case"* y *"Skip Zero Time"* y podremos trabajar con los datos cargados actualmente. Al hacer click en *Refresh"* podremos cargar los nuevos pasos de tiempo que hayan sido guardados en el disco.

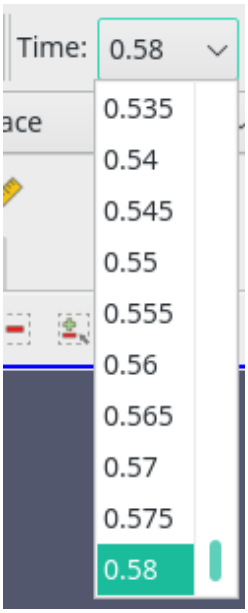

Figura 6: Lista de tiempos incompleta en ParaView. El tiempo total de simulación es 2

```
Courant Number mean: 0.67335769 max: 6.6853766
Time = 0.986
```

Figura 7: Valor medio y máximo del número de Courant para una iteración arbitraria

Mientras se ejecuta el caso es importante observar los valores del número de Courant. Este número representa una medida de la velocidad con la que la información es transportada bajo la influencia de un campo de flujo. Es un factor limitante en el rendimiento de los esquemas numéricos. Es recomendable que este número no sea superior a que 1. En la figura 7 se ve que el valor máximo de Courant para ese tiempo es muy alto (6.69). Esto lo podemos comprobar en la figura 8. En la imagen, la escala está limitada a un valor máximo de 1. Hay una cantidad importante de celdas donde el valor de Courant es muy alto.

Es posible que la simulación sea aceptable incluso para estos valores del número de Courant, pero para mejorar los resultados de la simulación, se cancela la simulación actual (`Ctrl+C` en la terminal) y se simula nuevamente con un paso de tiempo 10 veces más chico. Se espera que el número de Courant varíe linealmente con el paso de tiempo por su definición. Esta simulación es considerablemente más lenta (tardó aproximadamente 6 horas).

En las figuras 9 y 10 se observa la clara disminución del número de Courant. Como mencionamos en el párrafo anterior, los valores medios y máximos (así como en todo el campo de flujo) se redujo aproximadamente 10 veces estos valores.

4. TIEMPO TRANSITORIO

Con esta simulación podemos determinar el tiempo que demora el dispositivo en llegar al régimen estacionario. El solver *pimpleFoam* deja de iterar cuando se satisfacen las tolerancias definidas en el archivo `fvSolution`. Podemos aprovechar el archivo `log` generado para determinar cuándo llegó a la solución estacionaria.

`$ foamLog log.pimpleFoam` En la carpeta `$CASE_DIR/logs` se encuentran todos los archivos de post-proceso producidos por el comando anterior. En la figura 11 se observa que para un tiempo aproximadamente mayor a 0.6 podemos asumir que el solver ya ha convergido

Figura 8: Valor medio y máximo del número de Courant para una iteración arbitraria con h=0.001s

```
Courant Number mean: 0.067053114 max: 0.66257557
Time = 1
```

Figura 9: Valor medio y máximo del número de Courant para una iteración arbitraria con h=0.0001s

lo suficiente a la solución estacionaria.

En la figura 12 se compara la simulación en tres tiempos diferentes para observar esas mínimas diferencias en el campo de velocidad (imperceptibles).

5. COMPARACIÓN ENTRE SOLVERS

Otro aspecto interesante es verificar si las simulaciones `simpleFoam` y `pimpleFoam` convergen a la misma solución estacionaria (o lo suficientemente parecidas). En la figura 13 se comparan las dos simulaciones. Los resultados visibles son idénticos. También se comparan los campos de presión en la figura 14.

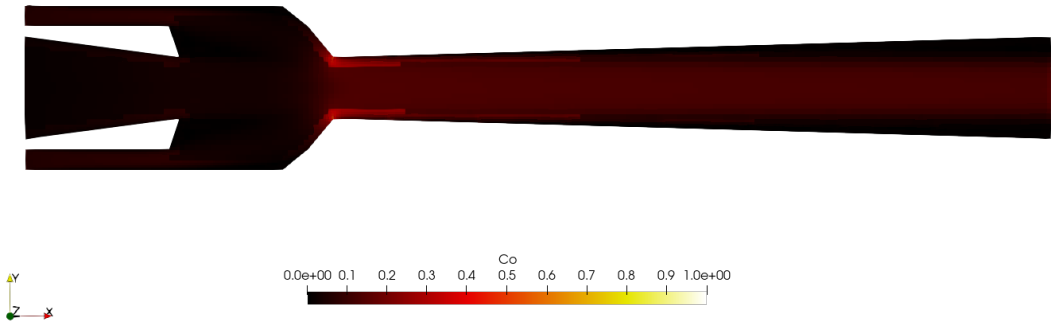

Figura 10: Valor medio y máximo del número de Courant para una iteración arbitraria con h=0.0001s

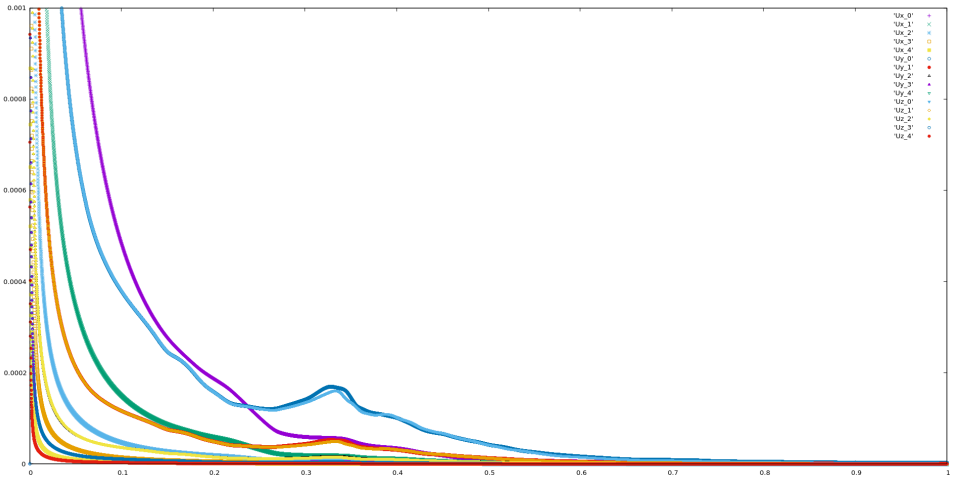

Figura 11: Variaciones en U en cada iteración del método

Por último, vamos a comparar las diferencias en los campos en las primeras iteraciones de cada uno de los solvers. Esto se ve en la figura 15 . Observar que el *simpleFoam* con 100 iteraciones ya prácticamente ha convergido a la solución final. En un solver transitorio, la simulación varía con el número de iteración según el paso de tiempo. En la figura 15 se visualiza la iteración 500 de *pimpleFoam*, es decir, el tiempo 0.05 s y se nota claramente que el flujo no está desarrollado y los transitorios no son comparables entre solvers.

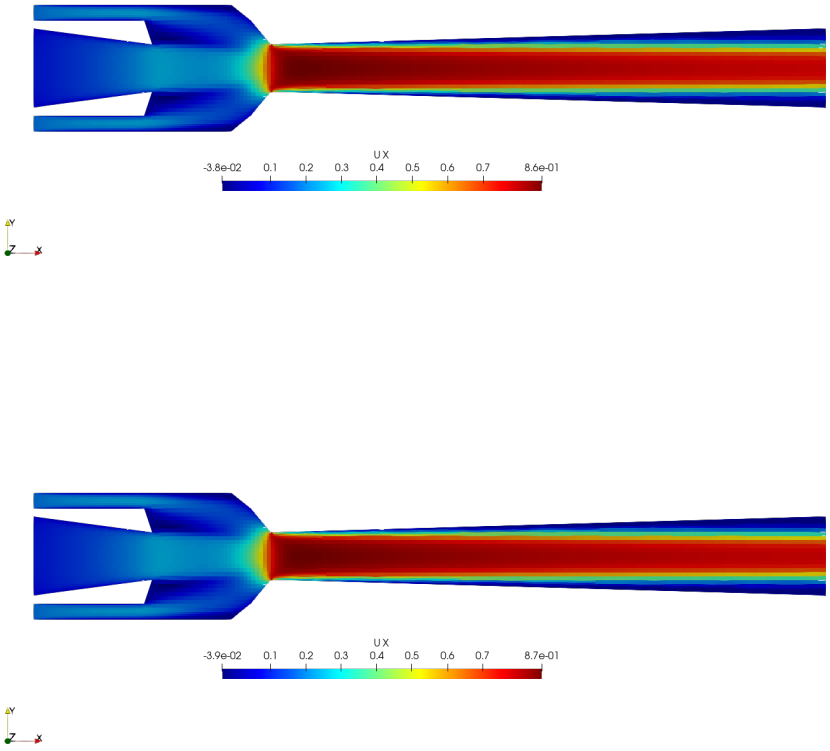

Figura 12: Campo Ux para t=0.6 (arriba) y t=0.8 (abajo)

6. CONCLUSIONES

En este informe ejecutamos la primera simulación transitoria con *pimpleFoam* sobre un caso previo de *simpleFoam*. Se analizaron algunas cuestiones propias de estos solvers como el *Número de Courant* y el tiempo de transitorio. También se verificó que los resultados del campo estacionario sean coherentes entre solvers.

En los informes anteriores se simulaba también la difusión y convección del agroquímico que es inyectado en el dispositivo. En este caso se omitió ese paso porque es el método utilizado es incompatible con este solver. Antes se obtenía el campo estacionario y luego, a partir de ese campo obtenido, se ejecutaba una simulación transitoria sobre la difusión del agroquímico (campo escalar). Como se utiliza un solver transitorio para calcular los campos de velocidad y presión en este informe, no es posible aplicar el mismo método.

El estudio de difusión de agroquímico en un campo transitorio se dejará para un informe futuro.

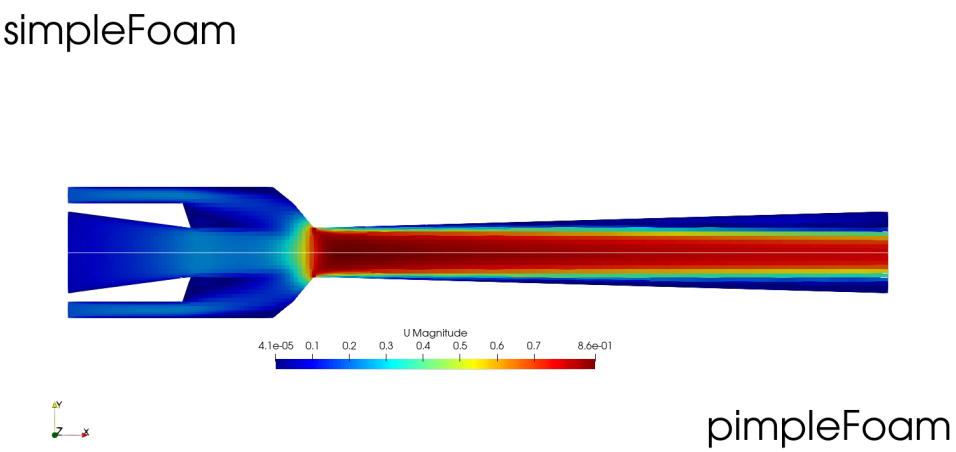

Figura 13: Comparación campo U entre ambos solvers

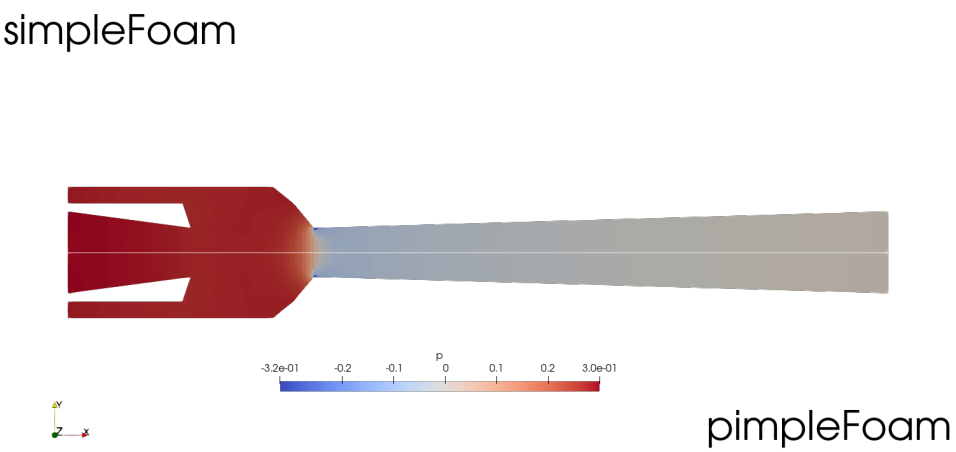

Figura 14: Comparación campo P entre ambos solvers

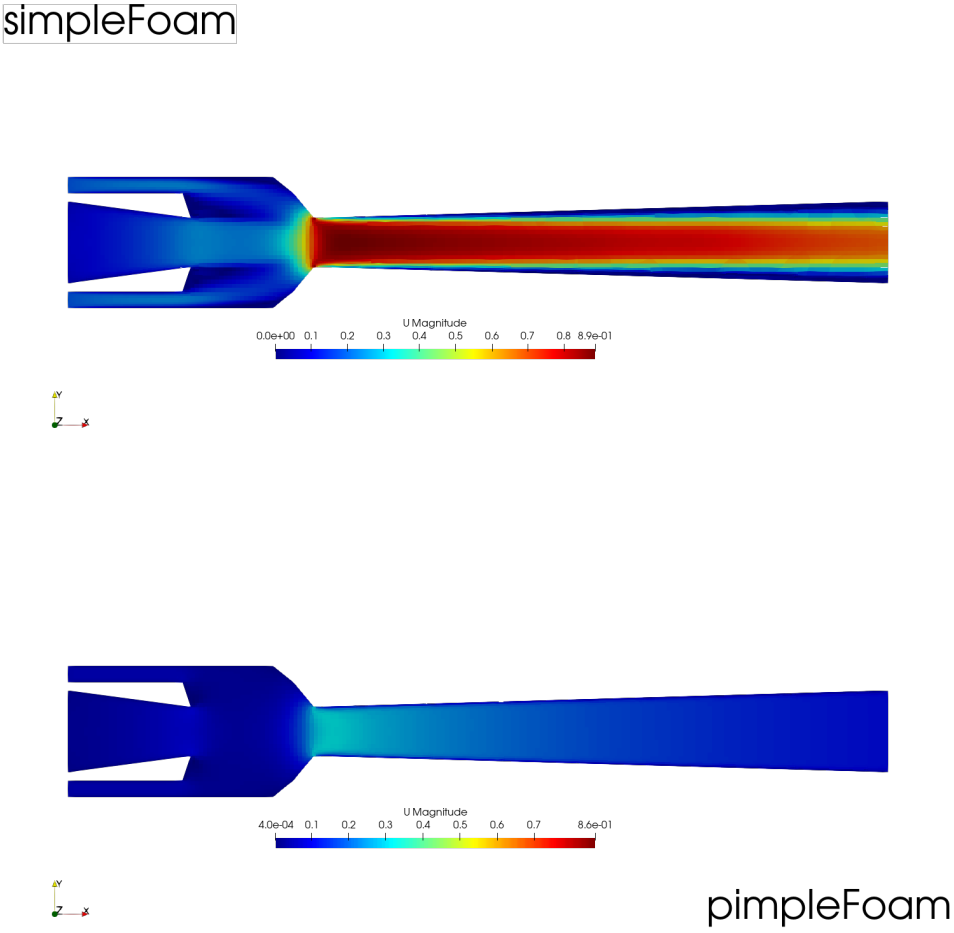

Figura 15: It. 100 de simpleFoam (arriba), It. 500 de pimpleFoam (abajo)