ANEXO I

MODELADO 3D
Y MALLADO

I.1. Introducción

El propósito de este *Anexo I* es explicar detalladamente cómo se obtiene el modelo 3D y el mallado del sistema estudiado. A pesar de que la descripción hace referencia al perfil NACA 0015, el proceso seguido con el perfil FX63 – 137 es idéntico.

Como se ha comentado anteriormente en la *Memoria*, el programa elegido para llevar a cabo esta parte del proyecto es *Salome*, mientras que el software encargado de realizar los cálculos numéricos es *OpenFOAM*. La utilización de estos determinados programas condiciona notablemente el diseño del modelo CAD.

En un principio, el sistema a estudiar quedaría perfectamente representado por un modelo 2D, compuesto por un plano provisto de un orificio central con la forma del perfil aerodinámico. Sin embargo, con este diseño la exportación del mallado desde *Salome* a *OpenFOAM* reporta un error debido a que *OpenFOAM* necesita una malla 3D cuando ésta es exportada con el formato de fichero usado en *Salome*. Para otros programas de diseño CAD los requisitos a la hora de exportar pueden ser distintos.

De este modo, un diseño 2D queda descartado por incompatibilidad de archivos entre el programa de modelado y el de cálculo computacional. Con todo esto, la solución más inmediata consiste en darle profundidad al anterior diseño, es decir, obtener un modelo 'falso 3D'. Es necesario prestar atención en el proceso de mallado para impedir subdivisiones a lo largo del espesor del modelo. Finalmente se deben reajustar las condiciones de contorno para que el sistema trabaje como un diseño 2D.

Así, el primer paso consiste evidentemente en obtener el modelo 3D representativo del sistema. A continuación se procede a la elaboración de la malla, la cual debe de disponer de una distribución adecuada y un número suficiente de nodos para obtener posteriormente unos resultados numéricos válidos. Por último, se realizan una serie de operaciones finales para la correcta exportación de la malla al programa de cálculo.

Finalmente, antes de comenzar la descripción de estas tareas comentar que el procedimiento se explica según el programa y versión empleados (*Salome 6.4.0*). El diseño con otra versión puede diferir ligeramente aunque el proceso general será bastante similar. En cambio, la utilización de un programa distinto impedirá un seguimiento conciso, si bien la sistemática para su obtención se mantendrá.

I.2. Obtención del modelo 3D

I.2.1. Introducción

En este apartado se describe el proceso de creación del modelo 3D del sistema que se analiza numéricamente. Esta explicación viene acompañada por una serie de imágenes con el fin de facilitar su entendimiento.

Un aspecto realmente importante a la hora de abordar el proceso de diseño 3D es que se encuentra íntimamente ligado al mallado que se desea obtener finalmente.

En primer lugar, la necesidad de realizar una malla de elementos rectangulares conlleva una mayor dificultad en el proceso de diseño. Esto se debe a que *Salome*, al igual que otros programas similares, exige que todo el dominio disponga de subdivisiones de 4 aristas para poder efectuar un mallado rectangular. Además, cada una de estas aristas no puede estar subdividida, si bien éstas pueden ser rectas o curvas.

Por otra parte, se desea que en las inmediaciones del perfil aerodinámico el mallado sea más compacto, mientras que en el contorno exterior la malla puede ser más basta. Por esta razón es aconsejable la división del dominio en dos regiones diferenciadas: una zona inmediata al perfil y el resto del dominio hasta el contorno exterior. De este modo se facilitan las posteriores operaciones de submallado en cada una de las zonas.

Por último, añadir que varios de los pasos intermedios en el modelado se podrían abordar y realizar de forma distinta, utilizando para ello diferentes herramientas del programa. La manera de proceder que se describe a continuación está basada en la sencillez y en la fácil disposición de cada uno de los elementos que componen este modelo.

I.2.2. Perfil aerodinámico

El primer paso en el proceso de diseño es obtener la geometría del perfil aerodinámico a estudiar.

Una técnica bastante utilizada en estos casos es la de importar un archivo previamente programado que, al ejecutarlo en el propio programa de diseño, genere la geometría buscada. Obtener el perfil introduciendo los puntos a mano implicaría mucho tiempo y esfuerzo. Concretamente, *Salome* permite importar scripts (archivos de texto) en lenguaje *Python*. El archivo con el código que genera el perfil aerodinámico se muestra en la *Figura I.2.1* y en la *Figura I.2.2*:

```
import geompy

# Se ha dibujado el perfil aerodinamico a trozos mediante interpol. En x se avanza con step constantes.

# Datos geometricos del perfil aerodinamico
c = 0.4
t = 0.15

# Numero de puntos en cada tramo
n = 2200.0
x = 0.0
y = 0.0
z = 0.0

i = 1
step = c/n
ptList = []
v = geompy.MakeVertex(x, y, z)
geompy.addToStudy(v, "Vertex_%f"%x )
ptList.append(v)
x += step

# Generacion del tramo superior
while (x<=c):
        while (i<=200):
                y = ((t*c)/0.2)*((0.2969*((x/c)**0.5))-(0.1260*(x/c))-(0.3516*((x/c)**2))+(0.2843*((x/c)**3))-(0.1036*((x/c)**4)))
                v = geompy.MakeVertex(x, y, z)
                #geompy.addToStudy(v, "Vertex_%f"%x )
                ptList.append(v)
                i += 1
                x += step

        geompy.addToStudy(v, "Vertex_%f"%x )
        interpol = geompy.MakeInterpol(ptList)
        geompy.addToStudy(interpol, "interpol" )
        ptList = []
        ptList.append(v)
        i = 1
```

Figura I.2.1: Script programado en *Python* I

```
# Generacion del tramo inferior
x -= step
x -= step
while (x>(200*step)):
        while (i<=200):
                y = -((t*c)/0.2)*((0.2969*((x/c)**0.5))-(0.1260*(x/c))-(0.3516*((x/c)**2))+(0.2843*((x/c)**3))-(0.1036*((x/c)**4)))
                v = geompy.MakeVertex(x, y, z)
                #geompy.addToStudy(v, "Vertex_%f"%x )
                ptList.append(v)
                i += 1
                x -= step

        geompy.addToStudy(v, "Vertex_%f"%x )
        interpol = geompy.MakeInterpol(ptList)
        geompy.addToStudy(interpol, "interpol" )
        ptList = []
        ptList.append(v)
        i = 1

while (i<=199):
        y = -((t*c)/0.2)*((0.2969*((x/c)**0.5))-(0.1260*(x/c))-(0.3516*((x/c)**2))+(0.2843*((x/c)**3))-(0.1036*((x/c)**4)))
        v = geompy.MakeVertex(x, y, z)
        #geompy.addToStudy(v, "Vertex_%f"%x )
        ptList.append(v)
        i += 1
        x -= step

x = 0.0
y = 0.0
v = geompy.MakeVertex(x, y, z)
#geompy.addToStudy(v, "Vertex_%f"%x )
ptList.append(v)
Interpol = geompy.MakeInterpol(ptList)
geompy.addToStudy(Interpol, "interpol" )

print "Done"
```

Figura I.2.2: Script programado en *Python* II

Este archivo contiene básicamente los datos geométricos del perfil, la ecuación característica para los perfiles NACA simétricos de 4 dígitos (en este caso 0015), los datos para su representación discreta y las acciones de generación y unión de los elementos que lo componen.

La importación del script anterior genera en *Salome* la geometría que se muestra en la *Figura I.2.3*:

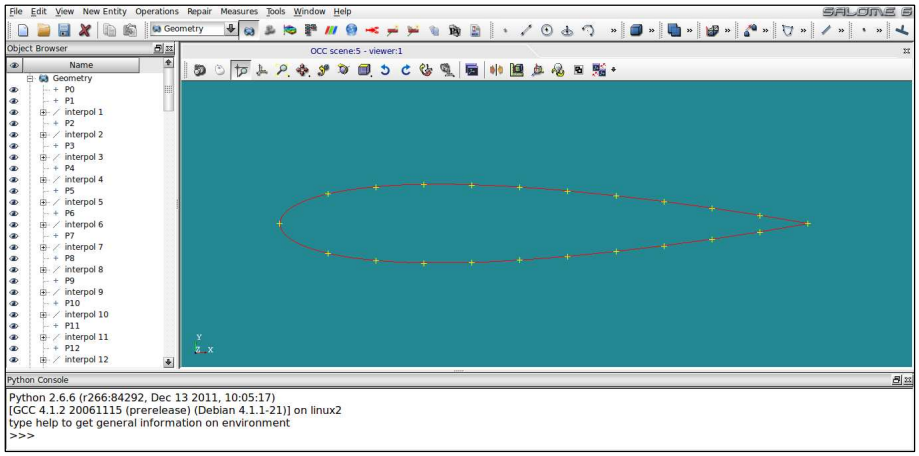


Figura I.2.3: Perfil aerodinámico

I.2.3. Zona inmediata al perfil

Una vez que se dispone del perfil aerodinámico se puede proceder al diseño de la zona inmediata a éste, en la cual se realiza más tarde un mallado compacto. Al igual que en otros apartados del diseño, la justificación de la geometría buscada se detalla en la *Memoria*.

El contorno exterior que delimita esta parte del dominio debe respetar la curvatura del perfil, acabando de forma rectangular en la parte posterior de éste. A continuación se crean las aristas que subdividirán en primera instancia esta parte del dominio. Estas aristas tienen que ser perpendiculares a la curvatura del perfil para que tras obtener la malla final, ésta sea normal al perfil en sus proximidades.

A la hora de obtener el perfil exterior de esta parte del dominio, el primer paso consiste en crear una serie de puntos homólogos a los dispuestos en el contorno del perfil. Estos nuevos puntos deben de mantener con sus homólogos una equidistancia. Además, su disposición debe ser tal que, unidos por una recta, ésta sea perpendicular a la recta tangente al perfil que pasa por el punto inicial.

Al igual que en la definición de la relación que existe entre las parejas de puntos, para su obtención es necesario utilizar varios elementos auxiliares.

En primer lugar se crea un punto próximo a su homólogo del perfil, aplicando previamente un zoom en la zona a trabajar para que esta distancia sea suficientemente pequeña. Este punto ha de colocarse sobre la línea del contorno del perfil. A continuación se selecciona la herramienta 2D sketch y se crea una recta desde el punto nuevo al punto original del perfil. Después, se selecciona la opción dirección perpendicular y se introduce la longitud de esta nueva recta.

En la *Figura I.2.4*, en la *Figura I.2.5* y en la *Figura I.2.6* se representa este proceso:

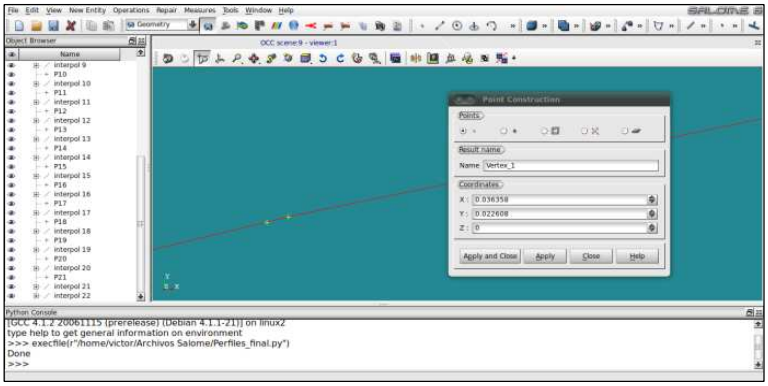


Figura I.2.4: Punto homólogo I

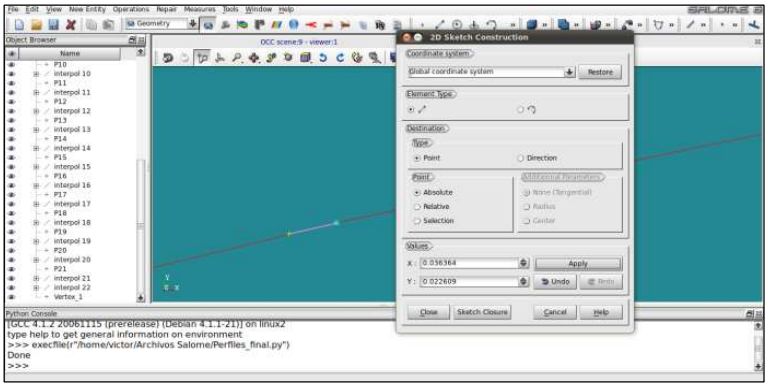


Figura I.2.5: Punto homólogo II

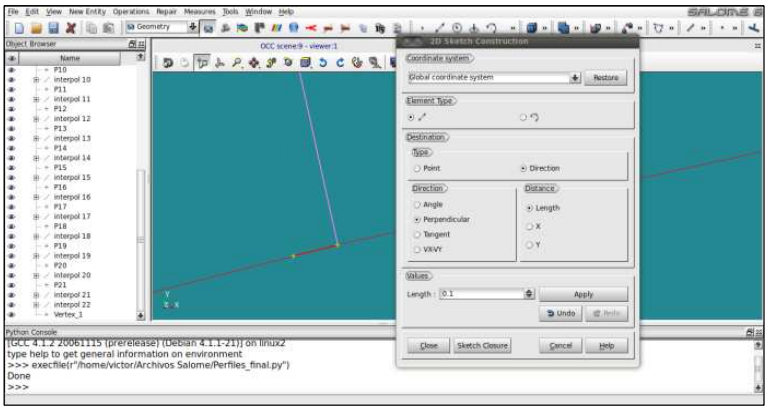


Figura I.2.6: Punto homólogo III

Cerrada la herramienta 2D sketch, se coloca un punto en el extremo de la recta perpendicular. En la *Figura I.2.7* se muestra este primer punto del contorno exterior:

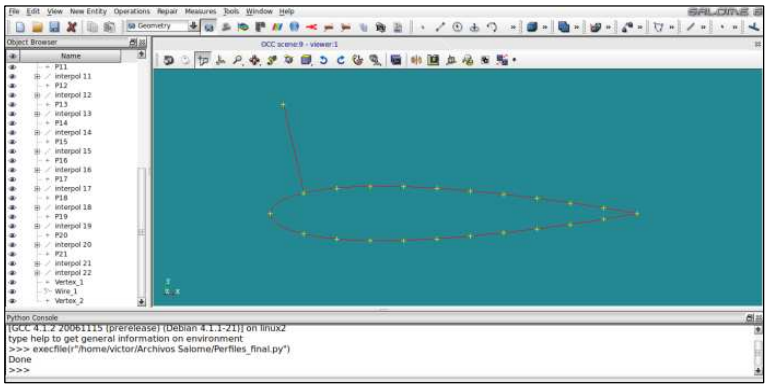


Figura I.2.7: Primer punto del contorno exterior

En la siguiente operación se realiza este proceso de forma repetida para cada uno de los puntos del contorno exterior del perfil. Además, se ha añadido un total de 6 puntos en la parte frontal de la curva, con el fin de mejorar el aspecto final del spline que conecta los puntos del contorno exterior. Recordad que en la parte frontal se producen cambios geométricos muy significativos que sería difícil de representar con tan solo 3 puntos.

El resultado final tras eliminar las rectas auxiliares se observa en la *Figura I.2.8*:

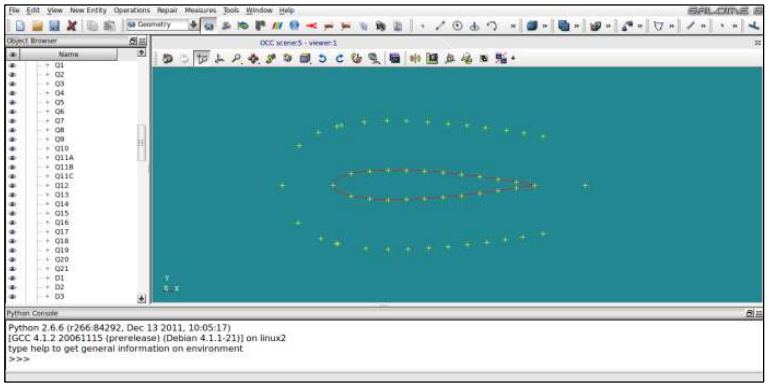


Figura I.2.8: Puntos del contorno exterior

A continuación, se crea el spline que pasa por todos los puntos del contorno exterior. Además, se añaden los puntos que forman la parte posterior rectangular y se unen al spline anterior, cerrando de este modo el contorno exterior. Para las operaciones sucesivas se pueden ocultar los puntos auxiliares de la parte frontal, quedando como resultado la geometría mostrada en la *Figura I.2.9*.

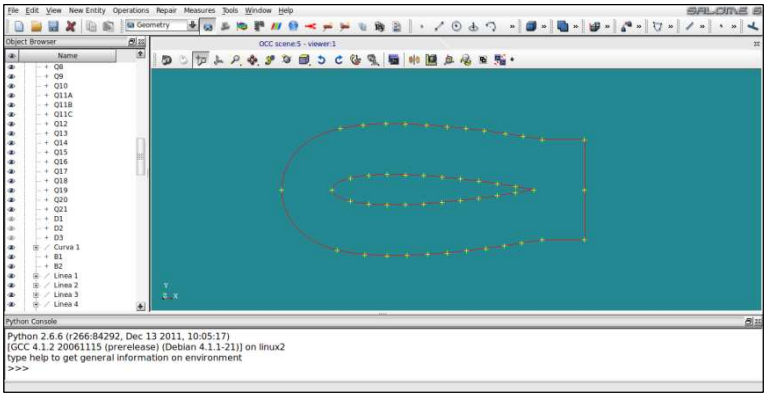


Figura I.2.9: Contorno exterior

Con el contorno exterior acabado, el siguiente paso consiste en crear las aristas que subdividan esta parte del dominio. Para ello se une cada pareja de puntos mediante una recta, tal y como se puede observar en la *Figura I.2.10*:

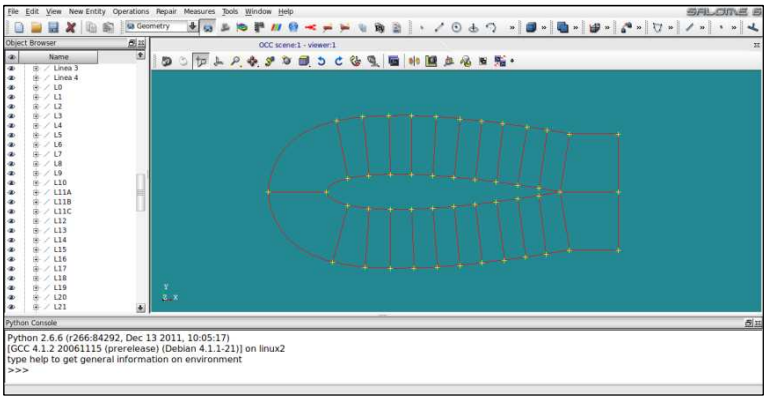


Figura I.2.10: Proceso de división

Después, se genera el plano relativo a esta parte del dominio. Este plano mostrado en la *Figura I.2.11* está formado por las líneas del contorno del perfil, por el spline y por las rectas que forman el contorno exterior.

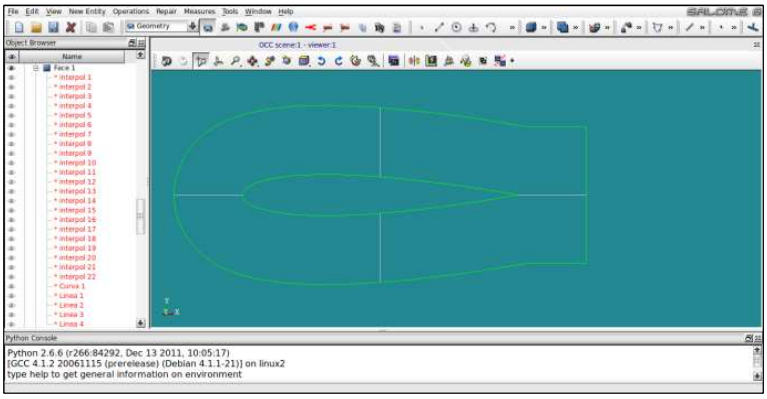


Figura I.2.11: Plano de la zona inmediata al perfil

En el último paso se efectúa una partición de este plano mediante las rectas divisorias, generando de este modo *Partition 1*. La realización de una partición se debe a la necesidad de disponer del plano subdividido en un único elemento geométrico. En la *Figura I.2.12* se representa el resultado final de todo este conjunto de operaciones:

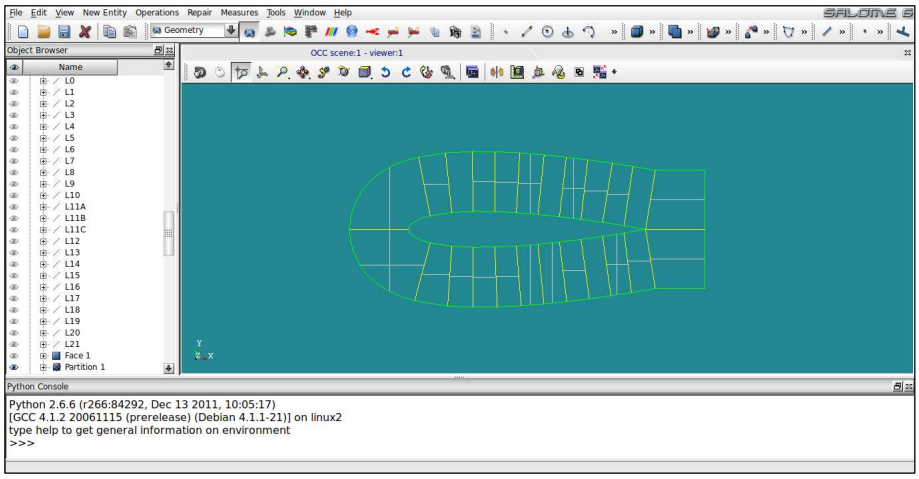


Figura I.2.12: *Partition 1*

I.2.4. Zona exterior

En cuanto se finaliza el diseño de la zona próxima al perfil aerodinámico, se procede a la creación del resto del dominio.

El primer paso consiste en situar un plano que delimite el dominio de estudio. La disposición más lógica de este plano es centrado respecto al perfil. Para ello se utiliza un vector auxiliar normal a *Partition 1* y se indica la dimensión deseada del plano. La geometría resultante se muestra en la *Figura I.2.13:*

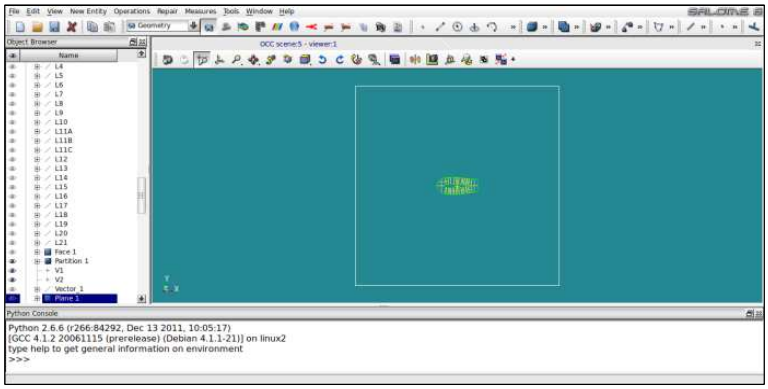


Figura I.2.13: Plano de estudio

A continuación se realiza al plano anterior un orificio central con la forma del perfil aerodinámico. Para ello es necesario crear un plano auxiliar con la forma del perfil, para posteriormente y mediante la herramienta booleana cut, efectuar el corte al plano principal. En la *Figura I.2.14* se muestra el resultado de la operación de corte:

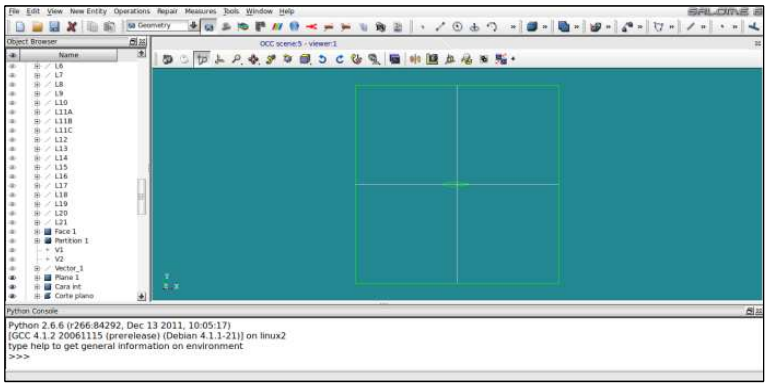


Figura I.2.14: Operación de corte

Después se realiza una serie de subdivisiones del dominio que permitan generar más tarde un mallado correcto. El objetivo es que la zona de transición desde *Partition 1* hasta el exterior del dominio quede correctamente subdividida, es decir, cada porción de superficie debe estar compuesta de 4 aristas. Para llevar a cabo estas operaciones es necesaria la visualización conjunta de *Partition 1* y del plano con el orificio central, por lo que se parte de la representación mostrada en la *Figura I.2.15*:

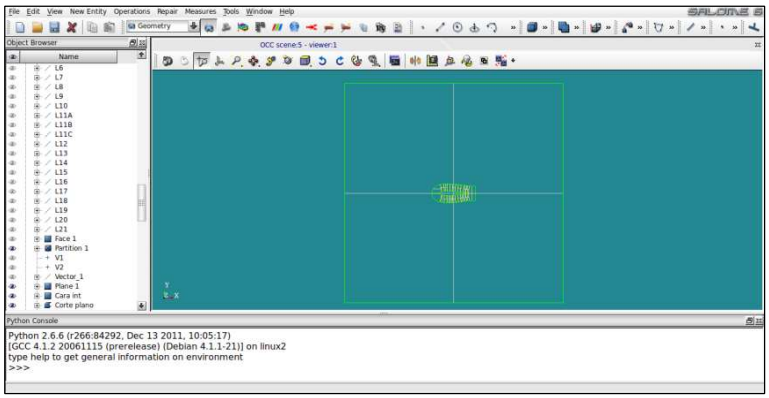


Figura I.2.15: Visualización conjunta

El primer paso para conseguir la división comentada anteriormente es crear una serie de puntos en el exterior del dominio de trabajo. Estos puntos junto con sus homólogos de la zona próxima al perfil se utilizan para crear las rectas divisorias. Algunos de los puntos a crear comparten alguna de las coordenadas de sus homólogos, mientras que otros obedecen a una relación de equidistancia. Para todo esto se utiliza la información disponible para cada uno de los puntos (nombre, coordenada x, coordenada y, coordenada z).

En la *Figura I.2.16* se observa la distribución de estos puntos del contorno exterior del plano. Una vez ubicados correctamente, la unión de cada pareja de puntos supone la subdivisión mostrada en la *Figura I.2.17*.

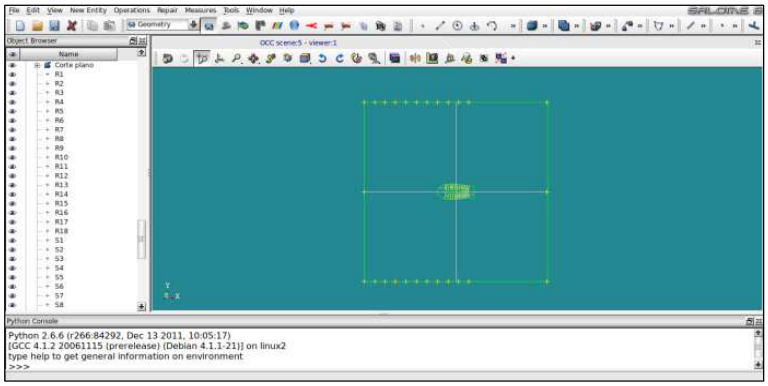


Figura I.2.16: Puntos del contorno exterior del plano

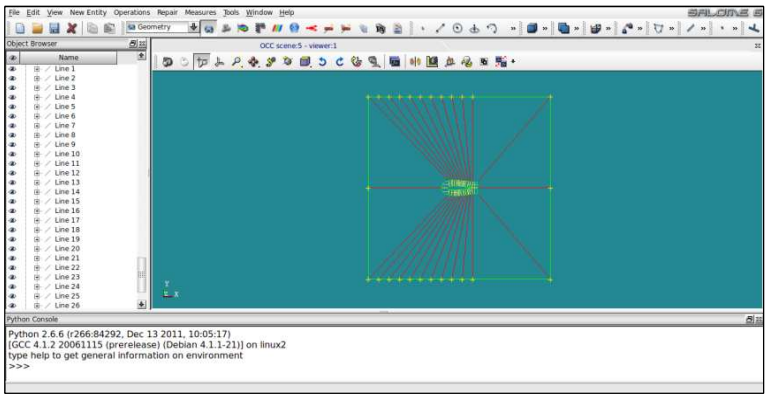


Figura I.2.17: División de la zona exterior

Seguidamente, se realiza una partición del plano haciendo uso de cada una de las rectas subdivisorias anteriores, obteniendo el elemento geométrico *Partition 2* mostrado en la *Figura I.2.18*:

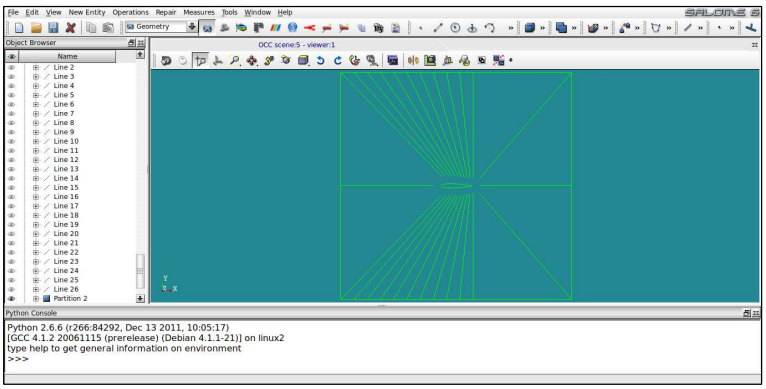


Figura I.2.18: *Partition 2*

Utilizando la herramienta booleana *fuse*, se obtiene finalmente el modelo 2D del sistema a estudiar (objeto *Fuse 1*). Este modelo se representa en la *Figura I.2.19*:

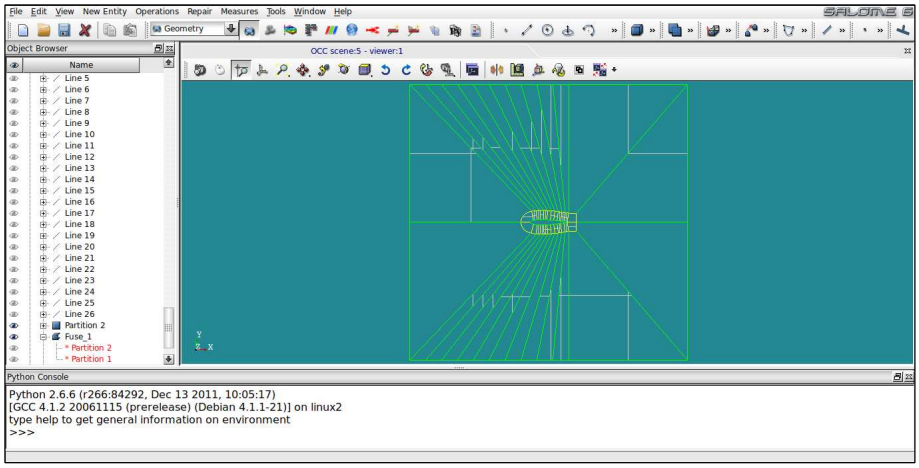


Figura I.2.19: Modelo 2D

I.2.5. Operaciones finales

En este momento se dispone de un modelo 2D que representa correctamente el sistema a estudiar. De este modo, el siguiente paso consiste en obtener a partir de este modelo el correspondiente en 3D, o como se ha denominado anteriormente, el modelo 'falso 3D'.

Esto se consigue fácilmente sometiendo al objeto *Fuse 1* obtenido anteriormente a una operación de extrusión. Utilizando un vector normal a la superficie e introduciendo el espesor deseado se obtiene el resultado mostrado en la *Figura I.2.20* y en la *Figura I.2.21*.

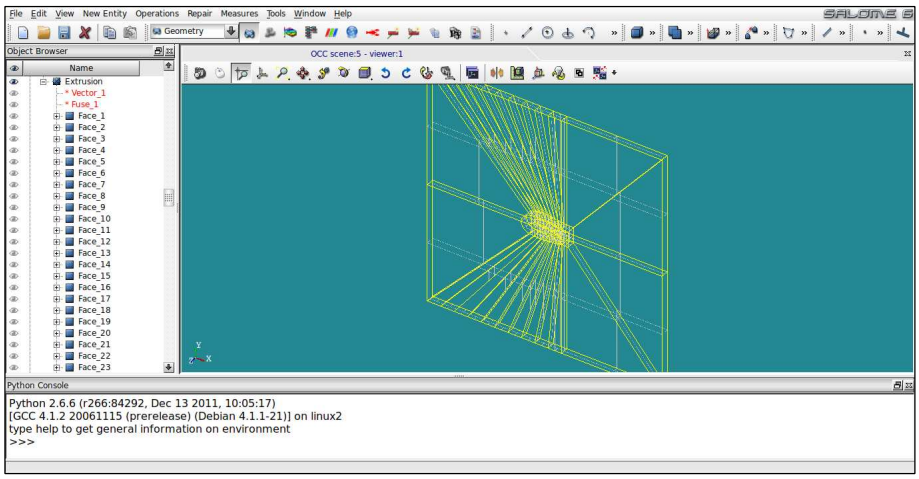


Figura I.2.20: Modelo 'falso 3D' I

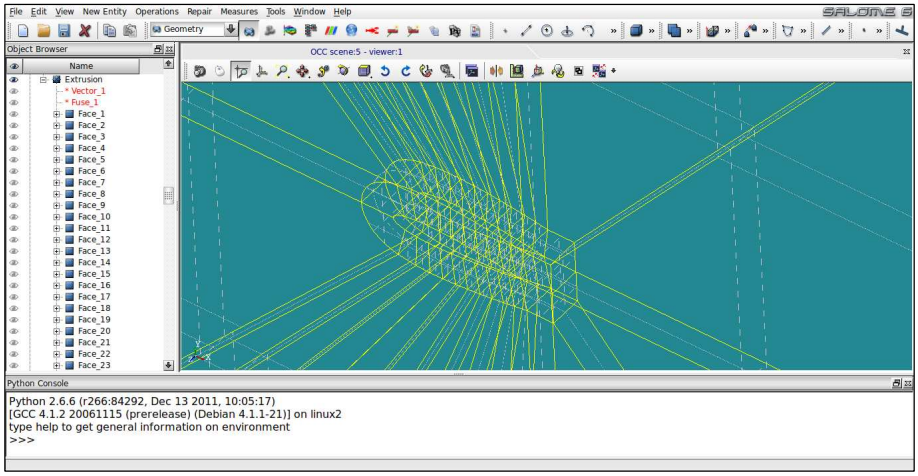


Figura I.2.21: Modelo 'falso 3D' II

Finalizado el proceso de diseño 3D, es necesario realizar una serie de operaciones finales antes de proceder a la realización del mallado. Estas operaciones consisten en la creación de una serie de grupos que faciliten las futuras labores de mallado en cada una de las distintas zonas del dominio.

Cada grupo está formado por varios elementos geométricos que comparten ciertas características. En este diseño se utilizan un total de 17 grupos que se describen a continuación. Aclarar que *FRONT* hace referencia a la cara frontal del 'falso 3D' mientras que, *BACK* hace referencia a la cara posterior.

Grupo 1

Caras de la zona inmediata al perfil aerodinámico (*FRONT*).

Grupo 2

Caras de la zona inmediata al perfil aerodinámico (*BACK*).

Grupo 3

Caras del resto del dominio sin subdivisión fina (*FRONT*).

Grupo 4

Caras del resto del dominio sin subdivisión fina (*BACK*).

Grupo 5

Caras del resto del dominio con subdivisión fina (*FRONT*).

Grupo 6

Caras del resto del dominio con subdivisión fina (*BACK*).

Grupo 7

Aristas de la zona inmediata al perfil (*FRONT & BACK*). Concretamente, las aristas perpendiculares al contorno del perfil más las dos verticales de la parte posterior.

Grupo 8

Aristas de la zona inmediata al perfil (*FRONT & BACK*). Concretamente, las aristas horizontales de la parte posterior.

Grupo 9

Aristas curvas frontales de la zona inmediata al perfil (*FRONT & BACK*).

Grupo 10

Aristas perímetro exterior sin subdivisión fina (*FRONT & BACK*).

Grupo 11

Aristas perímetro exterior con subdivisión fina (*FRONT & BACK*).

Grupo 12

Aristas subdivisorias del dominio no inmediato al perfil (*FRONT & BACK*).

Grupo 13

Conjunto de caras que forman el patch *Wall*.

Grupo 14

Conjunto de caras que forman el patch *Inlet*.

Grupo 15

Conjunto de caras que forman el patch *Outlet*.

Grupo 16

Conjunto de caras que forman el patch *Front*.

Grupo 17

Conjunto de caras que forman el patch *Back*.

De este modo, tras realizar todas estas operaciones se puede dar por finalizado el proceso de diseño del modelo 3D, por lo que se está en disposición de comenzar las tareas de generación del mallado.

En este punto se vuelve a insistir en el hecho de que, aunque sería necesario modificar ligeramente alguna de las operaciones realizadas, el proceso para obtener el modelo 3D del perfil FX63 – 137 sería idéntico.

I.3. Generación del mallado

I.3.1. Introducción

En este último capítulo se describe el proceso de generación del mallado, partiendo para ello del modelo 3D diseñado previamente.

Para empezar, se recuerda que para obtener unos resultados numéricos válidos es imprescindible generar una malla apropiada. En el caso a estudio conviene efectuar un mallado rectangular en todo el dominio de estudio. Además, éste debe ser más fino en las proximidades del perfil aerodinámico, mientras que en la región más alejada puede ser menos compacto.

Sin embargo, no hay que perder de vista de que trabajando con *Salome*, no es posible exportar a *OpenFOAM* la malla de un diseño 2D, llegando al planteamiento final de trabajar con un modelo 'falso 3D'. Este hecho conlleva la realización de un mallado hexaédrico que disponga a su vez de una malla rectangular a nivel superficial.

El proceso de mallado debe llevarse a cabo por etapas, partiendo de un mallado básico para todo el modelo. A continuación se realizan consecutivamente tantos submallados como sean necesarios. De este modo, cada submallado proporciona las características buscadas en su zona de influencia. Así, la acción conjunta del mallado principal junto con cada uno de los diferentes submallados permite alcanzar el resultado perseguido.

Finalmente, antes de exportar el mallado y proceder a los cálculos, es necesaria la generación de un grupo para cada uno de los patches del modelo (*Wall, Inlet, Outlet, Front, Back*). Esto permite identificar los patches en los distintos archivos de *OpenFOAM* y, de esta forma, delimitarlos y definir las condiciones de contorno que afectan a cada uno de ellos.

I.3.2. Mallado

El primer paso para generar el mallado del modelo 3D es crear y definir la malla principal *Mesh*. Ésta debe ser hexaédrica, rectangular a nivel superficial y con un número de divisiones igual a la unidad a nivel unidimensional. Para conseguir esta caracterización se introducen todas estas hipótesis en el cuadro de edición.

En la *Figura I.3.1* se puede observar el proceso de edición del mallado principal *Mesh* en *Salome*:

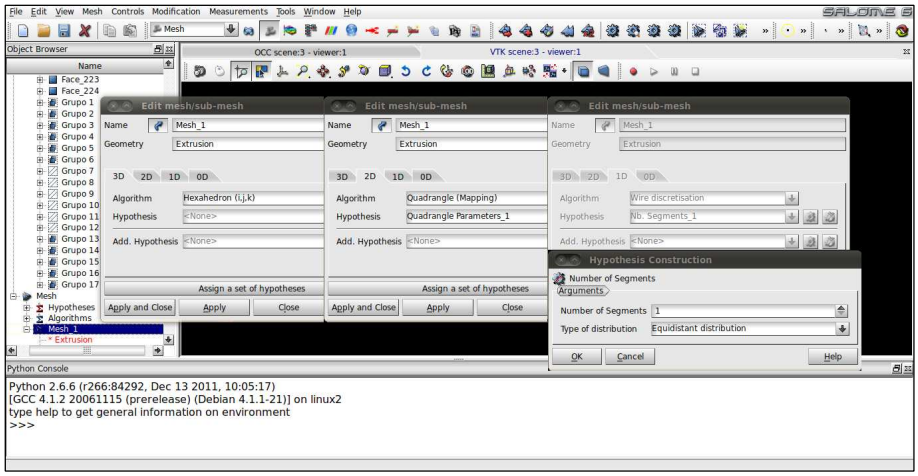


Figura I.3.1: Edición del mallado principal *Mesh*

Apuntar que es necesario utilizar la herramienta *Compute* para hacer efectivos los cambios realizados en el mallado principal o en los posteriores submallados.

Por último comentar que, partiendo de este punto, resulta difícil para el ordenador realizar correctamente todas las modificaciones que introducen cada uno de los submallados. De esta forma, pueden aparecer problemas durante las sucesivas reconstrucciones de la malla.

Para evitar este contratiempo, se ha editado el mallado principal *Mesh* tal y como se explica anteriormente, salvo que no se han introducido hipótesis 3D. De esta forma, se realiza un mallado superficial en el modelo, generando una malla 'hueca'. Una vez se hayan creado y editado todos los submallados, se regresa al cuadro de edición del mallado *Mesh*. Se introduce como hipótesis 3D un mallado hexaédrico y se reconstruye con *Compute*, consiguiendo finalmente un mallado correcto.

I.3.3. Submallado

El siguiente paso consiste en la creación de una serie de submallados que redefinan la malla hasta obtener el resultado buscado.

La interacción entre el mallado y los distintos submallados obedece a una ley de jerarquía. Así pues, un submallado puede modificar una zona mallada si impone unas restricciones más fuertes que las asignadas por un mallado o submallado anterior. De este modo, la forma de proceder consiste en partir de un mallado general que sea válido en una zona amplia del modelo para, sucesivamente, redefinir zonas específicas hasta alcanzar el mallado pretendido.

Para el análisis del perfil NACA 0015 se ha necesitado un total de 8 submallados para conseguir el resultado deseado. Algunos comparten características, si bien mientras que uno redefine la malla de la cara frontal (*FRONT*), el otro redefine la de la cara posterior (*BACK*). A continuación se detallan las características de cada uno de ellos y su proceso de edición.

I.3.3.1 Submallado 1 – 2

En este punto se describe el proceso de edición de los submallados 1 y 2. Aunque ambos generan el mismo mallado en la zona inmediata al perfil aerodinámico, mientras que el submallado 1 está ligado con la cara frontal del modelo, el submallado 2 está relacionado con la cara posterior.

Así, el submallado 1 (*SubMesh 1*) redefine la malla de la geometría perteneciente al *Grupo 1*, mientras que el submallado 2 (*Submesh 2*) modifica la relativa al *Grupo 2*. Como el proceso es igual para ambos, su descripción se realiza de forma conjunta.

El mallado es de tipo rectangular con un número de divisiones equidistantes igual a 10. Esta división se efectúa a todas y cada una de las aristas pertenecientes a esta parte del dominio. En la *Figura I.3.2* se muestra la malla generada por *SubMesh 1*:

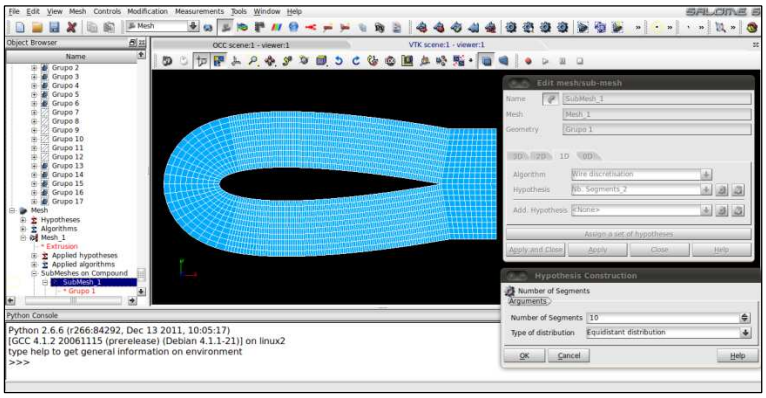


Figura I.3.2: *SubMesh 1*

Utilizando la herramienta *Mesh Information* es posible obtener la información y las características principales del mallado principal o de un submallado. La información junto con el resultado final de *SubMesh 2* (idéntico al anterior) se muestra en la *Figura I.3.3*:

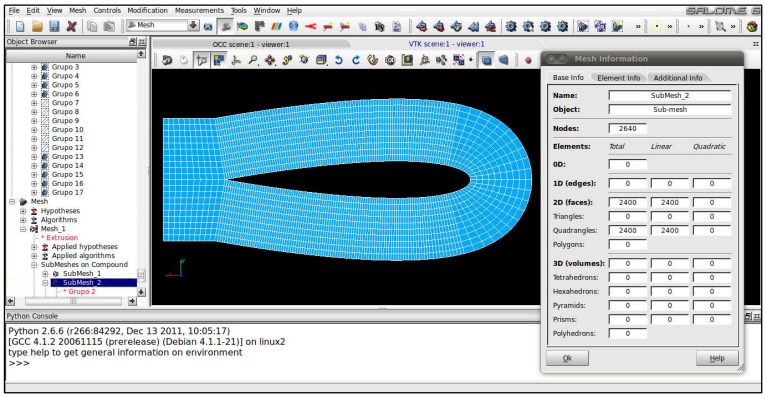


Figura I.3.3: *SubMesh 2*

I.3.3.2 Submallado 3 – 4

El submallado 3 (*SubMesh 3*) y el submallado 4 (*SubMesh 4*) redefinen el mallado generado en el apartado anterior. Concretamente, afectan a las aristas normales al contorno exterior del perfil aerodinámico. Al igual que en el caso anterior, la única diferencia entre ambos submallados es el dominio de aplicación. Así, *SubMesh 3* está ligado a la geometría del *Grupo 7* y por tanto a la cara frontal, mientras que *SubMesh 4* está relacionado con el *Grupo 8* y con el mallado de la cara posterior.

Cada uno de los submallados realiza una división en 20 segmentos de cada una de las aristas perpendiculares. Además, esta división se dispone de forma mucho más compacta en las inmediaciones del perfil, utilizando para ello la opción de *Scale Distribution*. Por último es necesario invertir el sentido de mallado en los ejes que lo precisen, quedando la malla tal y como aparece en la *Figura I.3.4*:

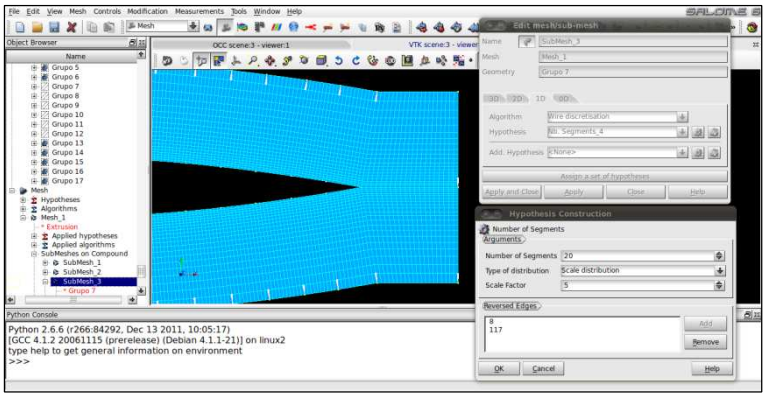


Figura I.3.4: *SubMesh 3 – 4*

I.3.3.3 Submallado 5 – 6

En este apartado se definen el submallado 5 (*SubMesh 5*) y el submallado 6 (*SubMesh 6*). Con estas operaciones el mallado en la zona inmediata al perfil aerodinámico queda finalizado. El propósito de estos submallados es mejorar el mallado en la parte delantera del perfil. Así pues, *SubMesh 5* redefine el mallado realizado al *Grupo 9*, mientras que *SubMesh 6* redefine el relativo al *Grupo 10*.

El rediseño de la parte delantera del perfil aerodinámico consiste en pasar de una división en 10 segmentos de las aristas curvas, a una división en 20 segmentos. El resultado final queda reflejado en la *Figura I.3.5*. Por otro lado, en la *Figura I.3.6* se muestra la acción conjunta del mallado principal y de todos los submallados creados hasta este punto.

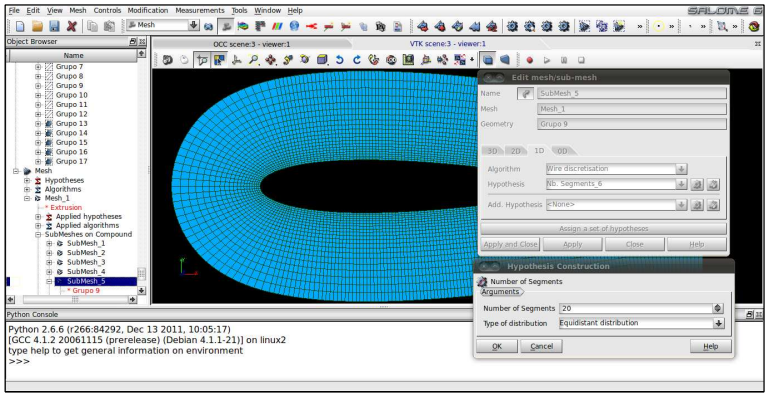


Figura I.3.5: *SubMesh 5 – 6*

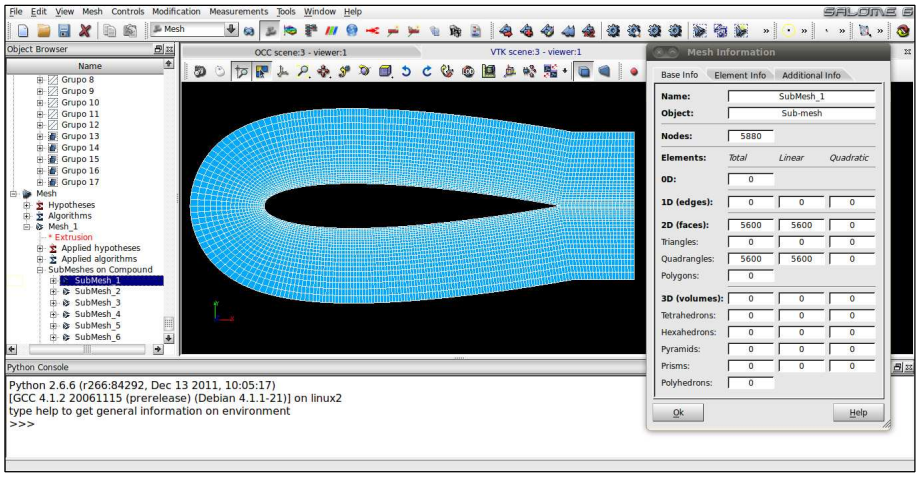


Figura I.3.6: Mallado final de la zona inmediata al perfil

Si se compara con las imágenes iniciales de *SubMesh 1* y *SubMesh 2* se aprecia claramente el proceso de rediseño del mallado. Además, destacar que partiendo de 2400 caras iniciales, se ha llegado finalmente a un total de 5600, lo que refleja de forma cuantitativa el refinamiento realizado a la malla.

I.3.3.4 Submallado 7

Una vez mallado correctamente la zona próxima al perfil, el siguiente paso consiste en caracterizar el mallado del resto del dominio.

El submallado 7 (*SubMesh 7*) se encarga de dividir correctamente parte del contorno exterior del dominio. En este punto se trabaja tanto con la geometría perteneciente a la parte frontal (*FRONT*) como a la de la parte posterior (*BACK*). Concretamente, *SubMesh 7* actúa sobre la geometría relativa al *Grupo 10*, es decir, al contorno exterior que no ha sido previamente subdividido en el modelado 3D.

En cuanto a las características de este submallado, *SubMesh 7* realiza un total de 20 divisiones en cada una de las aristas que pertenecen a este grupo. La *Figura I.3.7* muestra el cuadro de edición de este submallado:

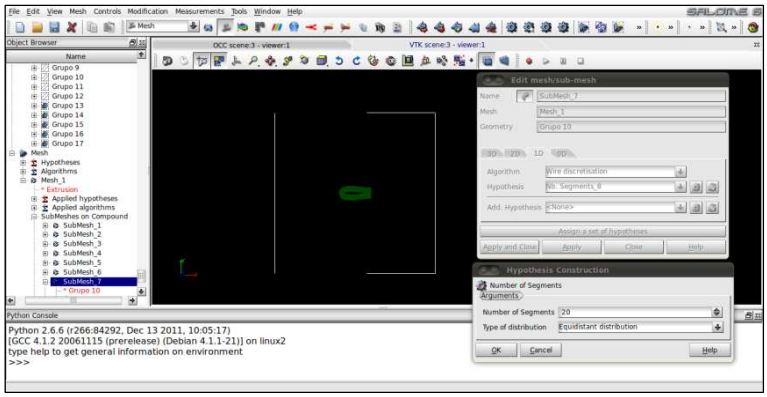


Figura I.3.7: *SubMesh 7*

I.3.3.5 Submallado 8

Para finalizar el proceso de mallado falta por redefinir el contorno exterior que no ha sido tratado en el apartado anterior. Así pues, el submallado 8 (*SubMesh 8*) modifica la geometría relativa al *Grupo 11*.

En este submallado se ha optado por dividir en 4 segmentos cada uno de los elementos pertenecientes al *Grupo 11*. Como consecuencia de esto, se genera un número total de divisiones en esta parte del contorno exterior distinto al del contorno de la zona próxima al perfil. Esto se soluciona mediante una transición que permite reajustar el mallado según se necesite.

El proceso de edición de este submallado aparece en la *Figura I.3.8*.

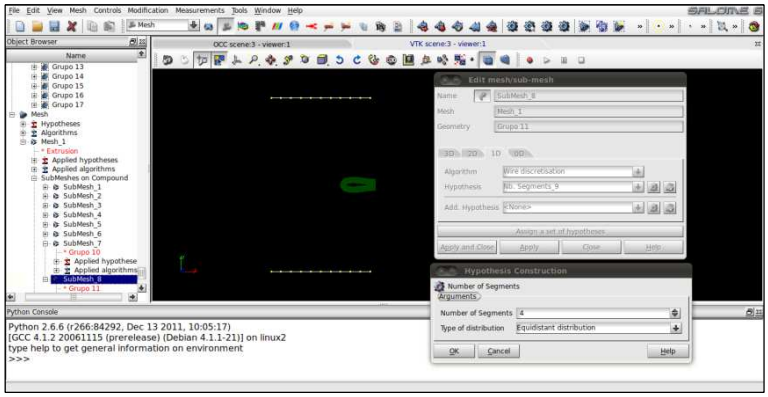


Figura I.3.8: *SubMesh 8*

I.3.4. Resultado final

En este apartado se presenta el resultado final que se obtiene al seguir paso a paso lo expuesto en el *Anexo I*.

A continuación, en la *Figura I.3.9* y en la *Figura I.3.10* se muestra el mallado con el que se analiza posteriormente el comportamiento del perfil NACA 0015. Destacar que en la *Figura I.3.10* se puede apreciar la transición comentada en el apartado en el que se describe *SubMesh 8*. Se observa cómo la malla es reajustada para que el número de divisiones coincida finalmente con el correspondiente a la zona del contorno exterior del dominio.

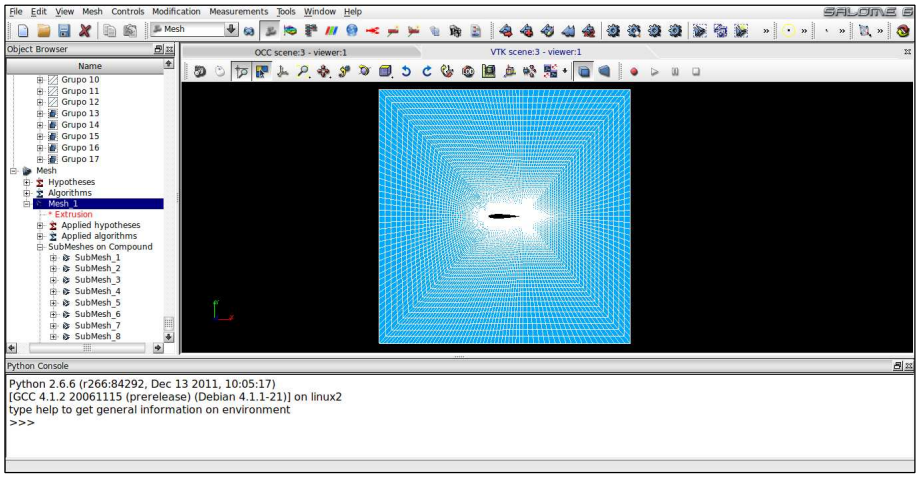


Figura I.3.9: Resultado final del mallado I

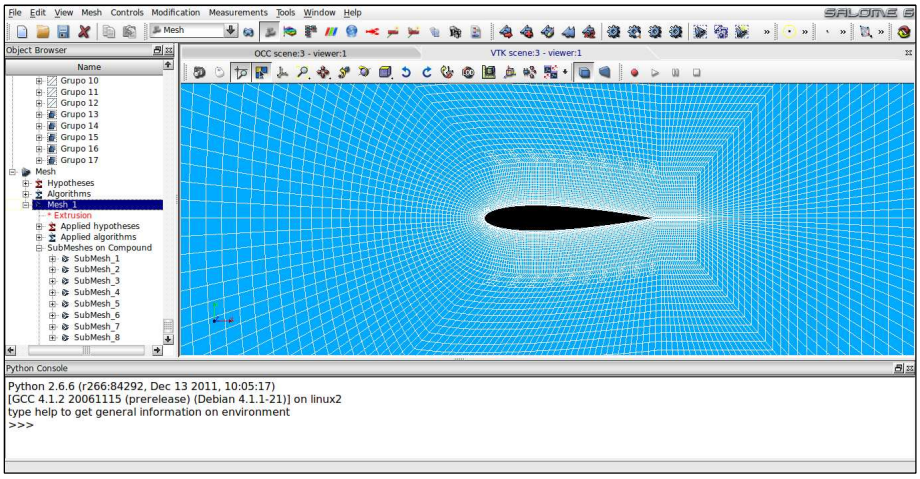


Figura I.3.10: Resultado final del mallado II

En la *Figura I.3.11* queda reflejado el hecho de que no se ha realizado ninguna subdivisión a lo largo del espesor, obteniendo de esta forma el modelo 'falso 3D':

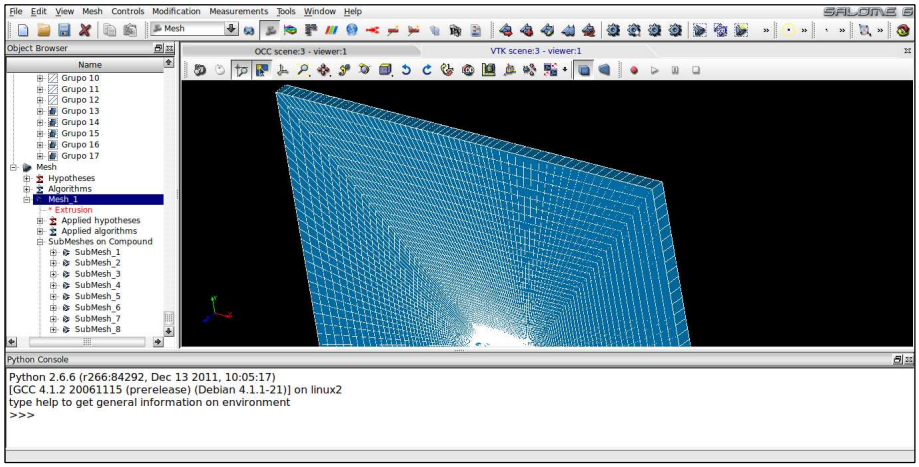


Figura I.3.11: Resultado final del mallado III

Por último, en la *Figura I.3.12* se ofrece una vista general del mallado acompañada de un cuadro con la información más relevante. Destacar el número total de nodos (28360), así como la cantidad de caras rectangulares (30200) y volúmenes hexaédricos (13960) que forman el mallado.

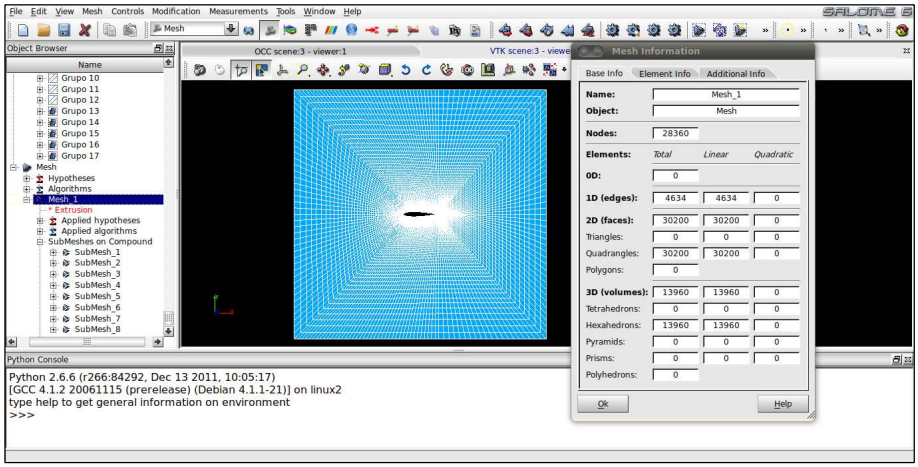


Figura I.3.12: Resultado final del mallado IV

I.3.5. Grupos

En este momento se dispone de un mallado que cumple correctamente con todos los requisitos planteados inicialmente. No obstante, antes de proceder a su exportación es necesario llevar a cabo una serie de operaciones finales. Estas operaciones consisten en la creación de varios grupos de caras utilizando para ello los grupos geométricos del modelo 3D. De este modo, cada uno de los nuevos grupos formará un patch, lo que facilita los pasos previos al cálculo numérico.

Finalmente se dispone de un total de 5 patches: *Wall* (superficie del hueco relativo al perfil aerodinámico – *Grupo 13*), *Inlet* (superficie del contorno exterior en las que se especifican las condiciones de entrada – *Grupo 14*), *Outlet* (superficie del contorno exterior de salida – *Grupo 15*), *Front* (superficie frontal del modelo – Grupo *16*) y *Back* (superficie posterior del modelo – *Grupo 17*). En la *Figura I.3.13* se muestra el mallado final una vez creados estos grupos:

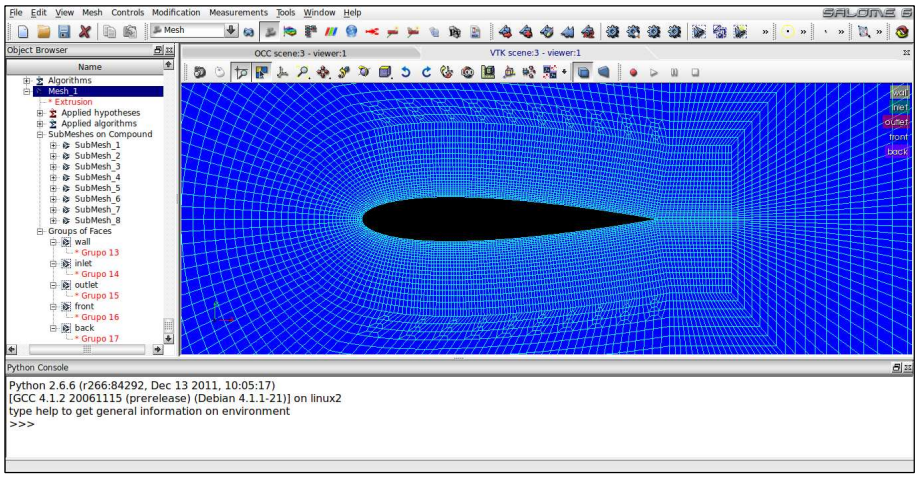


Figura I.3.13: Mallado final con patches

ANEXO II

RESULTADOS DEL CÁLCULO COMPUTACIONAL

II.1. Introducción

La elaboración de este *Anexo II* tiene como finalidad recoger en un mismo documento los resultados más importantes de cada una de las simulaciones numéricas realizadas a lo largo del proyecto.

En el segundo capítulo se presentan los resultados procedentes de los distintos estudios realizados al perfil aerodinámico NACA 0015, mientras que en el tercer capítulo se muestran los resultados correspondientes al perfil FX63 – 137.

Ambos capítulos mantienen una distribución de contenidos similar. Así, en el primer apartado se realiza una breve introducción comentando los aspectos más relevantes del perfil analizado y describiendo el conjunto de simulaciones efectuadas. En los siguientes apartados se exponen de manera ordenada los resultados obtenidos en cada uno de los casos estudiados. El último apartado se encarga de recoger en varias tablas todos los coeficientes aerodinámicos procedentes del cálculo, junto con sus correspondientes valores experimentales. Además, en este apartado final también se incluyen una serie de gráficas que facilitan la comparación de estos coeficientes.

Asimismo, cada uno de los apartados reservados a la presentación de resultados está estructurado del mismo modo.

Inicialmente se muestra la información relativa a los coeficientes aerodinámicos mostrados por *OpenFOAM* al finalizar el proceso de cálculo. También queda reflejado si el proceso ha terminado debido a que el error alcanzado es menor que el error máximo permitido o, si por el contrario, se ha alcanzado el número máximo de iteraciones impuesto. A continuación se exponen sendas gráficas con el estudio de la convergencia de los coeficientes aerodinámicos de sustentación (C_L) y de resistencia o arrastre (C_D).

En el caso de que se observe que durante el proceso de cálculo los resultados numéricos han convergido correctamente hasta un resultado final, la información y gráficas anteriores se acompañan de una serie de imágenes. Estas imágenes muestran la evolución de la presión y de la velocidad en el dominio de estudio. Por el contrario, en caso de que los resultados no presenten cierto nivel de convergencia, carece de sentido observar el comportamiento de estas variables ya que los resultados no tienen ni validez ni credibilidad.

En este punto es importante aclarar que, con el fin de no saturar de forma excesiva el *Anexo II*, se ha decidido no incluir una serie de imágenes que contienen información relativa a cada uno de los casos analizados. Concretamente, estas imágenes muestran las líneas de corriente alrededor del perfil, así como la evolución a lo largo del dominio de las variables turbulentas de disipación turbulenta (ε), energía cinética turbulenta (κ) y viscosidad turbulenta (μ_t).

No obstante, toda esta información junto con la disponible en este documento puede ser consultada en el CD incluido en el proyecto (*CD – Resultados*).

Por último, antes de proceder a la presentación de los resultados es aconsejable aclarar varias cuestiones relacionadas con las gráficas que aparecen en los apartados sucesivos.

En todas las imágenes que muestran la evolución de la presión en el dominio de estudio, se está representando una presión relativa. Además, ésta viene expresada en unidades SI (Pa). En el caso de las imágenes relacionadas con la variable velocidad, ésta también mantiene las unidades SI (m/s).

Asimismo, aunque no estén disponibles en el *Anexo II*, las imágenes con la evolución de las distintas variables turbulentas también están expresadas en unidades SI. De esta forma, las unidades de la disipación turbulenta ε son m/s^3, las de la energía cinética turbulenta κ son m^2/s^2 y las de la viscosidad turbulenta μ_t son m^2/s.

Finalmente y aunque parezca obvio, indicar que en estas imágenes el carácter utilizado como separador decimal es el punto.

II.2. Cálculos numéricos del perfil NACA 0015

II.2.1. Introducción

En este apartado se describe el conjunto de análisis numéricos realizados al perfil NACA 0015. Como se ha comentado anteriormente en la *Memoria*, la decisión de analizar tal cantidad de casos atiende a consideraciones geométricas y aspectos relacionados con parámetros del proceso de cálculo.

En primer lugar, la principal característica geométrica del perfil NACA 0015 es su simetría. Este hecho invita a pensar que los resultados obtenidos para un determinado ángulo de ataque deberían ser muy similares a los correspondientes a un ángulo de ataque negativo y del mismo valor, por lo que se descarta analizar situaciones para ángulos de ataque negativos.

Para determinar el conjunto de ángulos a estudiar, al razonamiento anterior se suman otras cuestiones como la convergencia de los resultados según el ángulo fijado, la disposición de un número limitado de datos experimentales para su comparación, etc. De este modo, se opta finalmente por realizar simulaciones que abarquen desde un ángulo de ataque de 0º hasta un ángulo de 16º, avanzando de grado en grado.

Asimismo, atendiendo a aspectos propios del proceso de cálculo, se decide estudiar 3 casos diferentes para cada uno de los ángulos previamente especificados. Los parámetros asociados a cada uno de los casos pueden ser consultados en el capítulo 3 de la *Memoria*.

Por último, para comprobar que los mallados diseñados son adecuados se realizan simulaciones para este perfil aerodinámico utilizando dos mallas nuevas más finas. Así, mientras que en la primera de ellas se ha doblado el número de nodos en la zona inmediata al perfil aerodinámico, en la segunda se ha utilizado dos veces el número inicial de nodos en todo el dominio de estudio.

Estos análisis se llevan a cabo para valores del ángulo α de 10º y 11º, estudiando casos homólogos a los casos 1, 2 y 3. Concretamente, los casos 4, 5 y 6 corresponden a los estudios realizados con la segunda versión de la malla y con unas condiciones equivalentes a las de los casos 1, 2 y 3 respectivamente. En la misma línea, en los casos 7 y 8 los cálculos se realizan utilizando la tercera versión de la malla y para las mismas condiciones que en los casos 1 y 2.

II.2.2. Resultados ángulo de ataque 0º

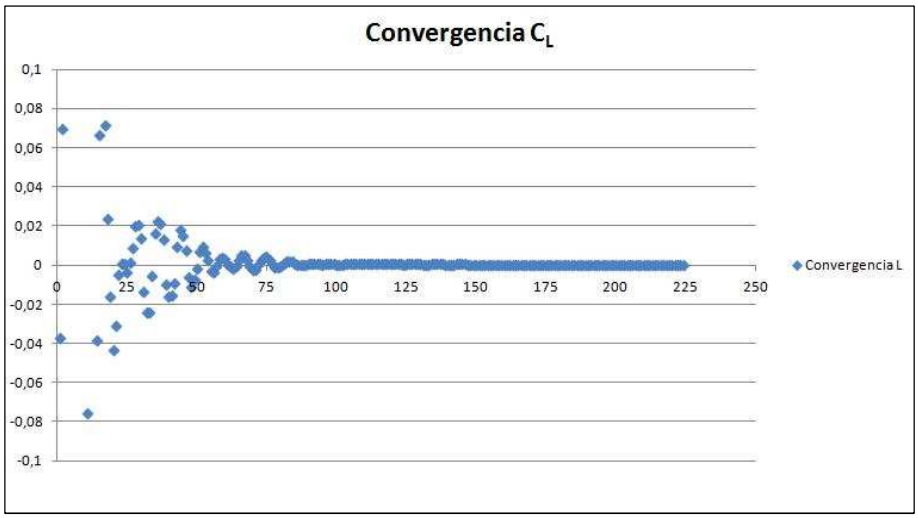
II.2.2.1. Caso 1 (Ángulo de ataque 0º)

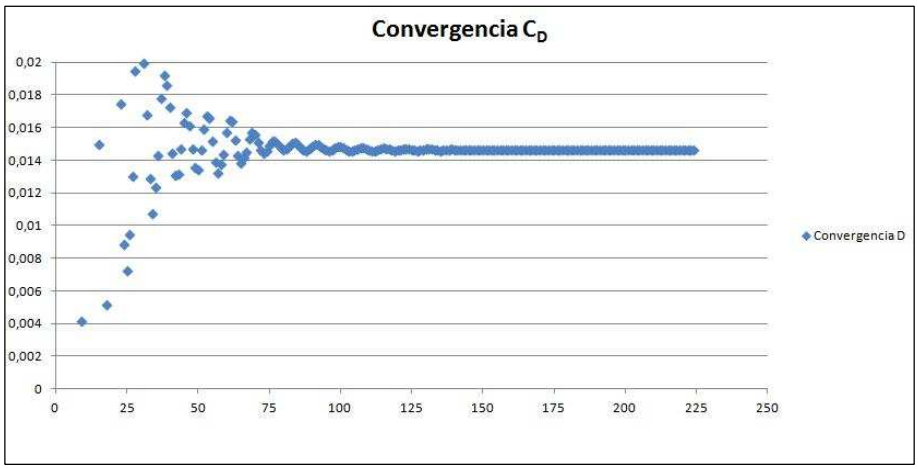
SIMPLE solution converged in 224 iterations.

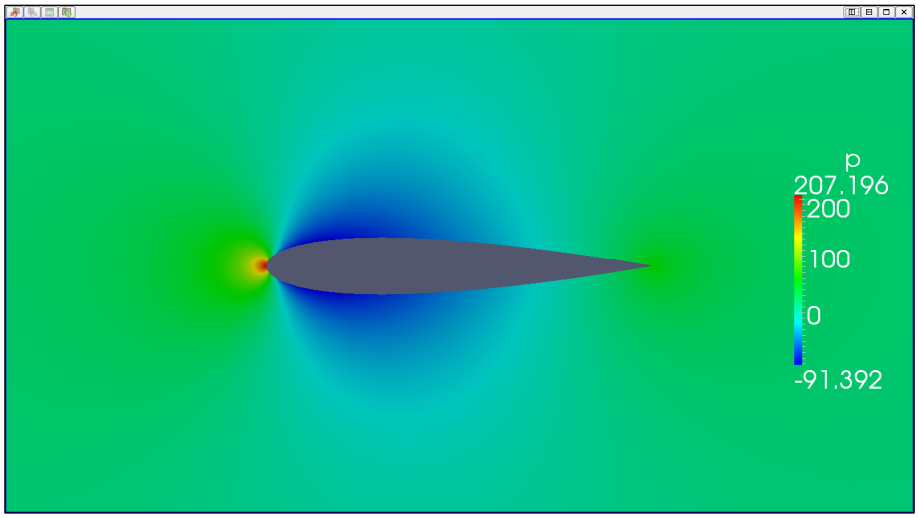
ForceCoeffs output:

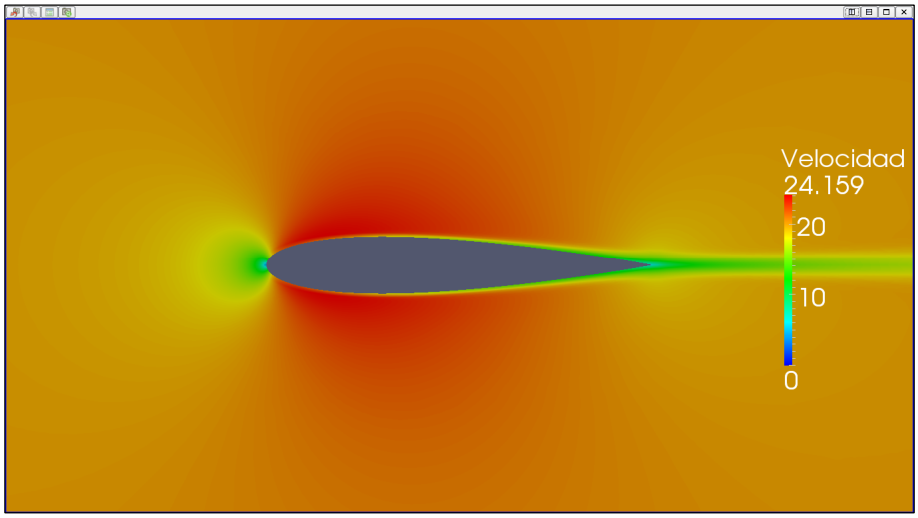
$$C_L = 6{,}1922 \cdot 10^{-05}$$

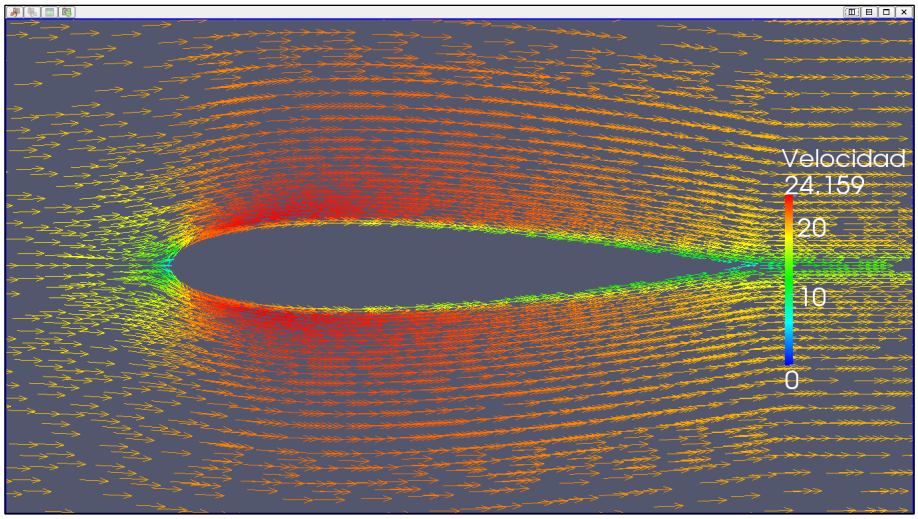
$$C_D = 0{,}0146$$






p
207.196
200
100
0
-91.392


Velocidad
24.159
20
10
0


Velocidad
24.159
20
10
0

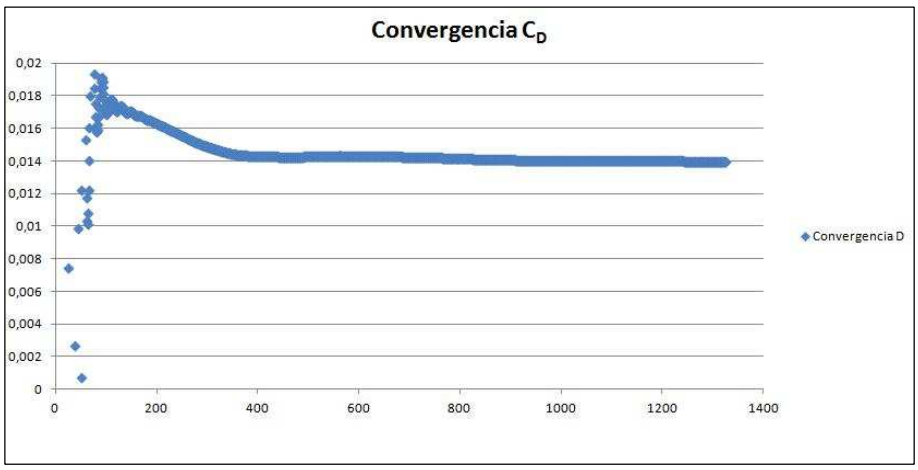
II.2.2.2. Caso 2 (Ángulo de ataque 0º)

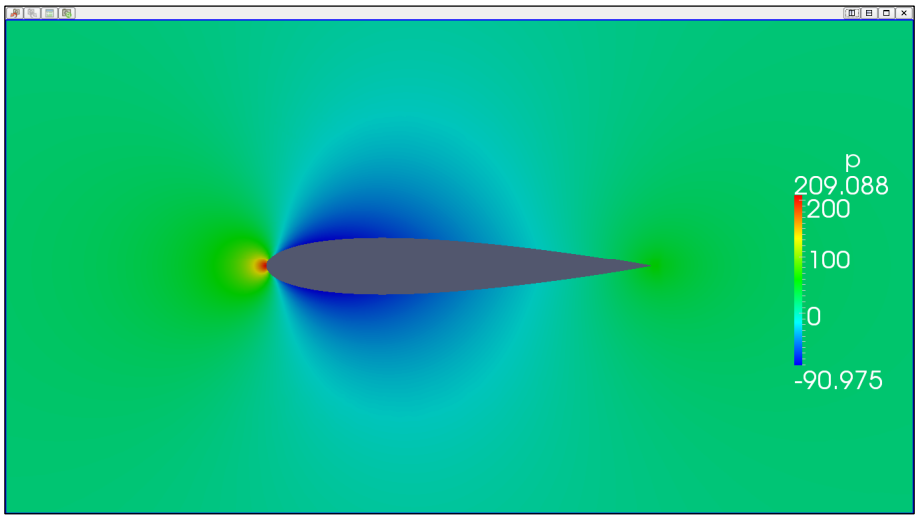
SIMPLE solution converged in 1326 iterations.

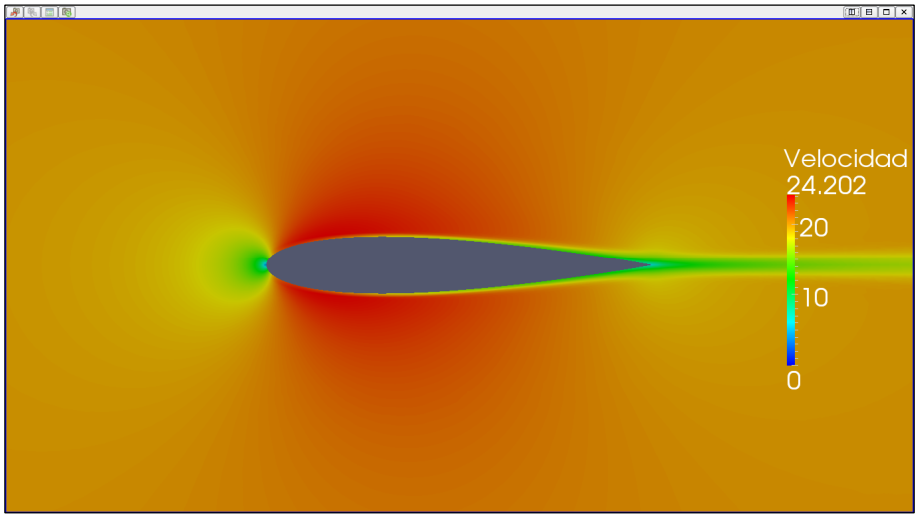
ForceCoeffs output:

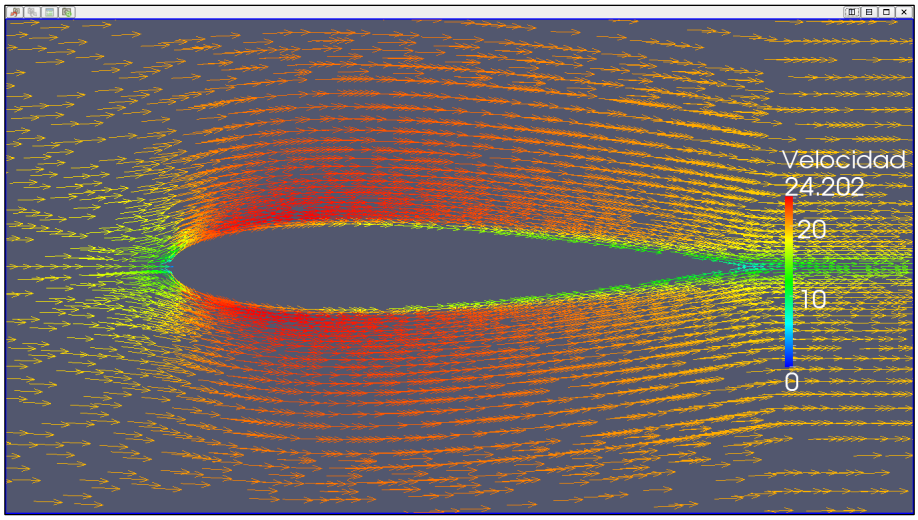
$$C_L = -5{,}3426 \cdot 10^{-05}$$

$$C_D = 0{,}0140$$




p
209.088
200
100
0
-90.975


Velocidad
24.202
20
10
0


Velocidad
24.202
20
10
0

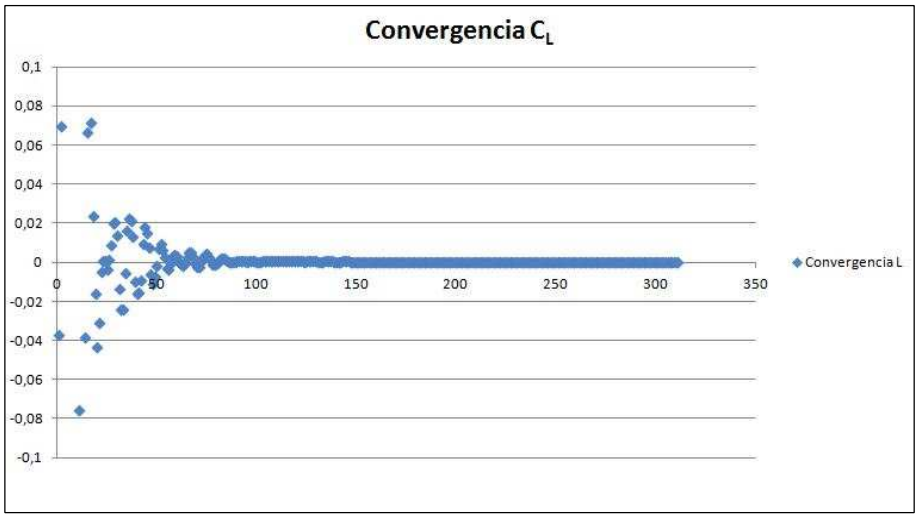
II.2.2.3. Caso 3 (Ángulo de ataque 0º)

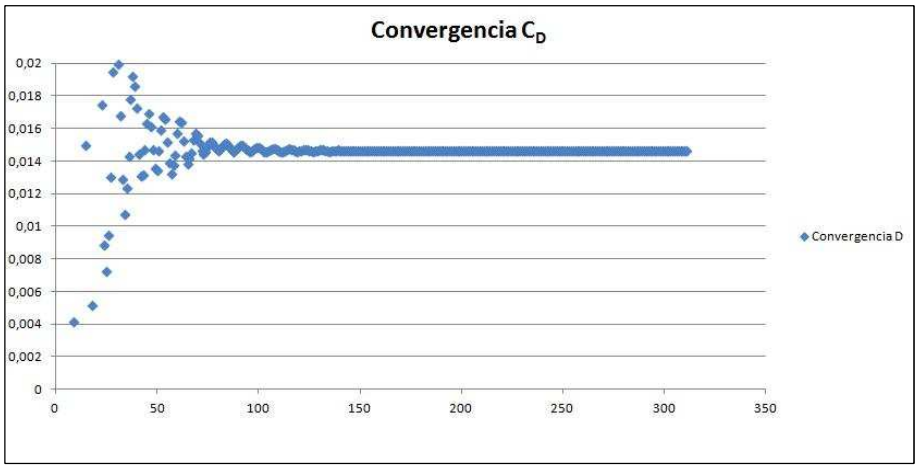
SIMPLE solution converged in 311 iterations.

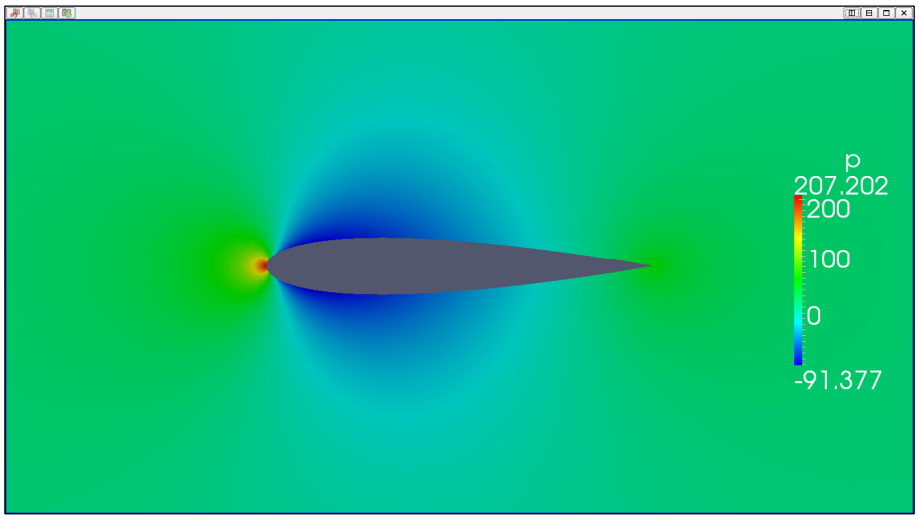
ForceCoeffs output:

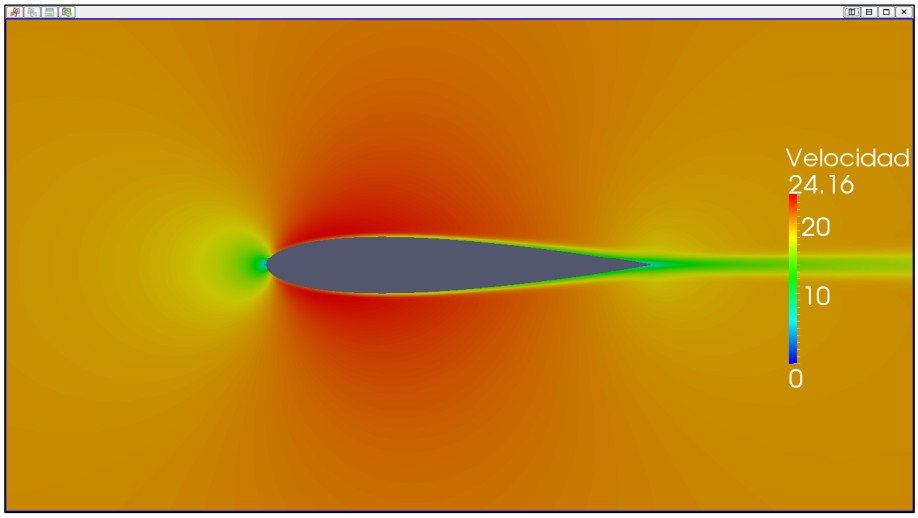
$C_L = 5{,}6614 \cdot 10^{-06}$

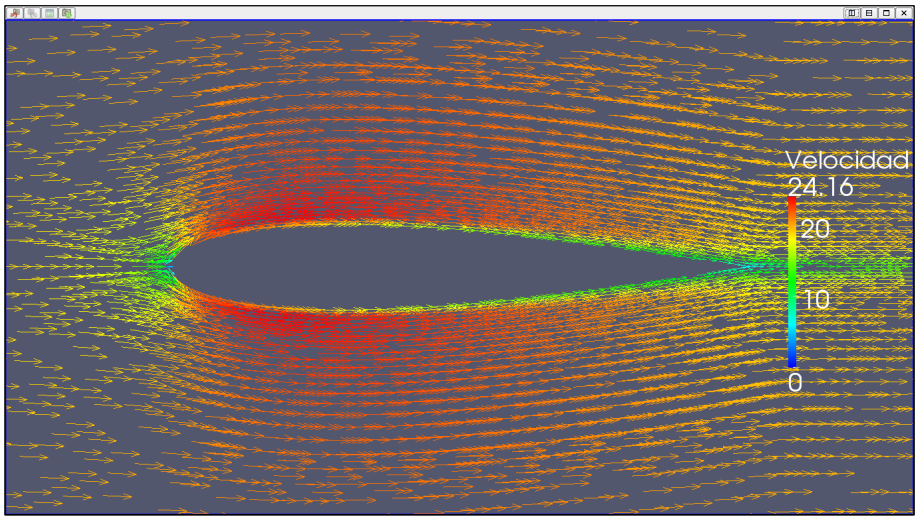
$C_D = 0{,}0146$






p
207.202
200
100
0
-91.377


Velocidad
24.16
20
10
0


Velocidad
24.16
20
10
0

II.2.3. Resultados ángulo de ataque 1º

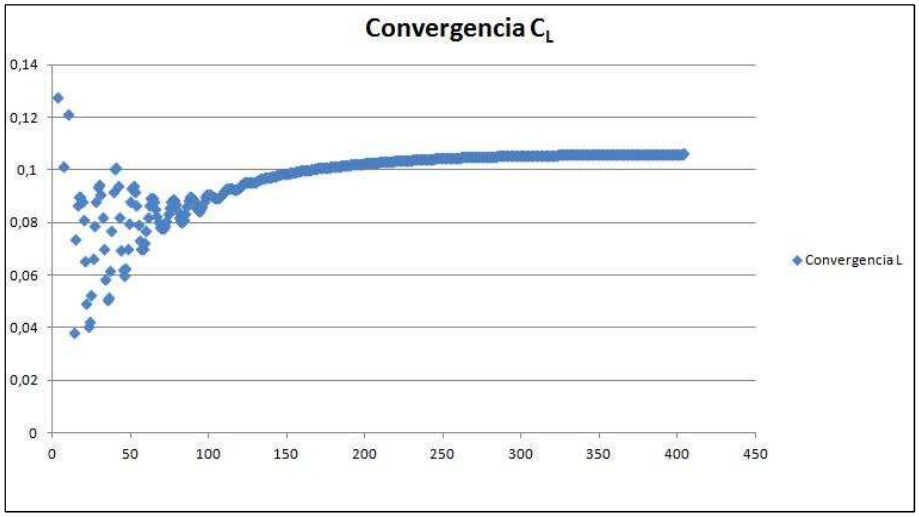
II.2.3.1. Caso 1 (Ángulo de ataque 1º)

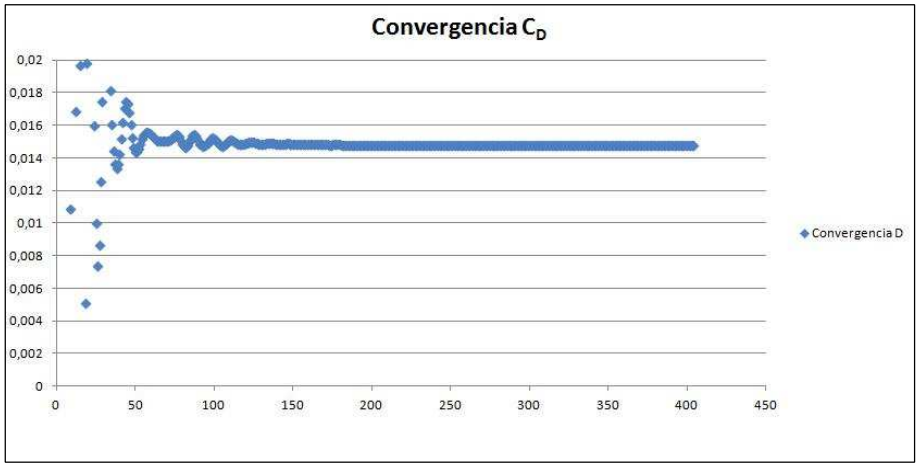
SIMPLE solution converged in 404 iterations.

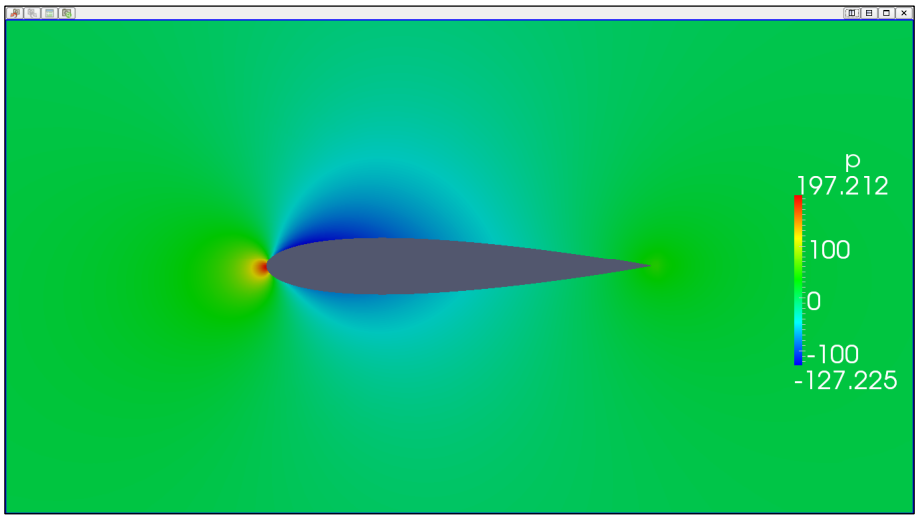
ForceCoeffs output:

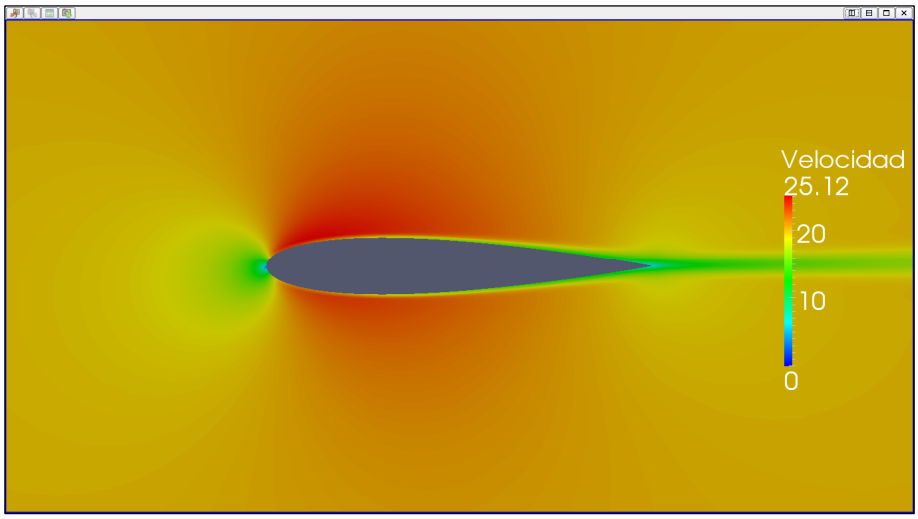
$C_L = 0{,}1061$

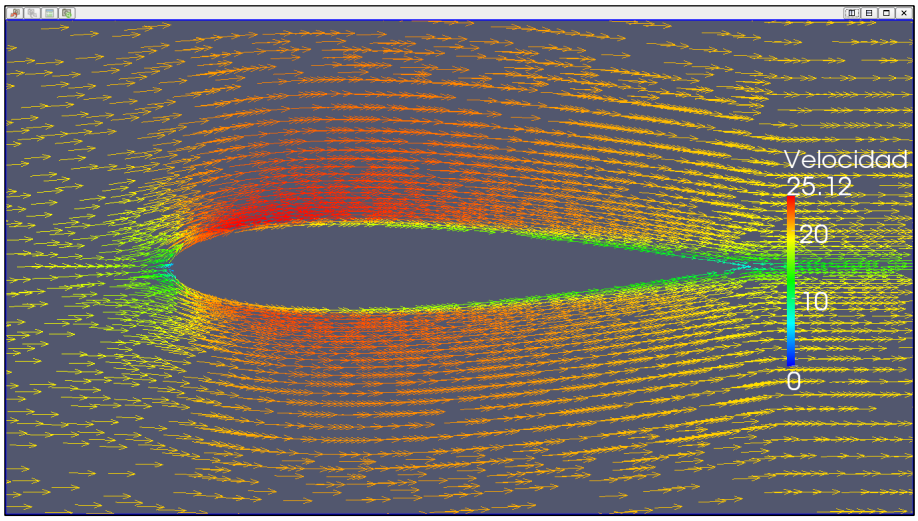
$C_D = 0{,}0147$






p
197.212
100
0
-100
-127.225


Velocidad
25.12
20
10
0


Velocidad
25.12
20
10
0

II.2.3.2. Caso 2 (Ángulo de ataque 1º)

Time: 30000

Solving U_X: Initial residual $= 1{,}261 \cdot 10^{-05}$, Final residual $= 1{,}441 \cdot 10^{-08}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 6{,}944 \cdot 10^{-06}$, Final residual $= 1{,}103 \cdot 10^{-08}$, No Iterations $= 2$.

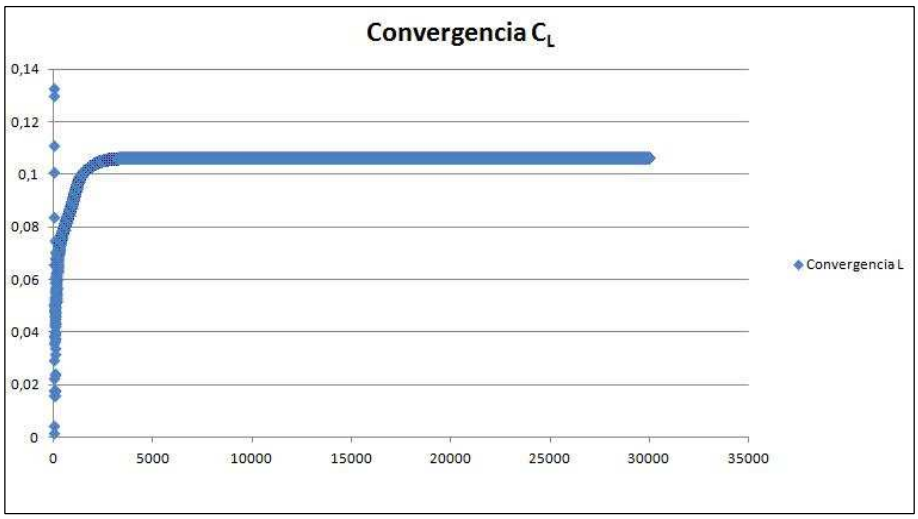
Solving p: Initial residual $= 6{,}246 \cdot 10^{-04}$, Final residual $= 5{,}961 \cdot 10^{-05}$, No Iterations $= 1$.

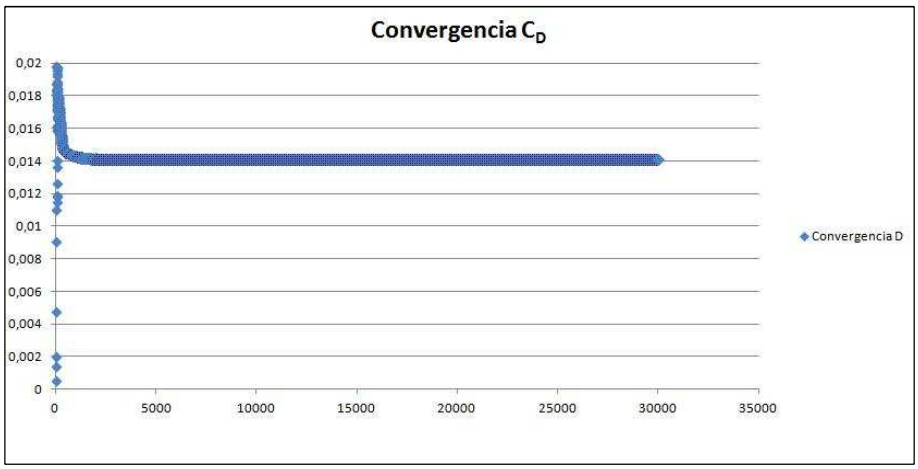
ExecutionTime: 2763,62 s. ClockTime: 2769 s.

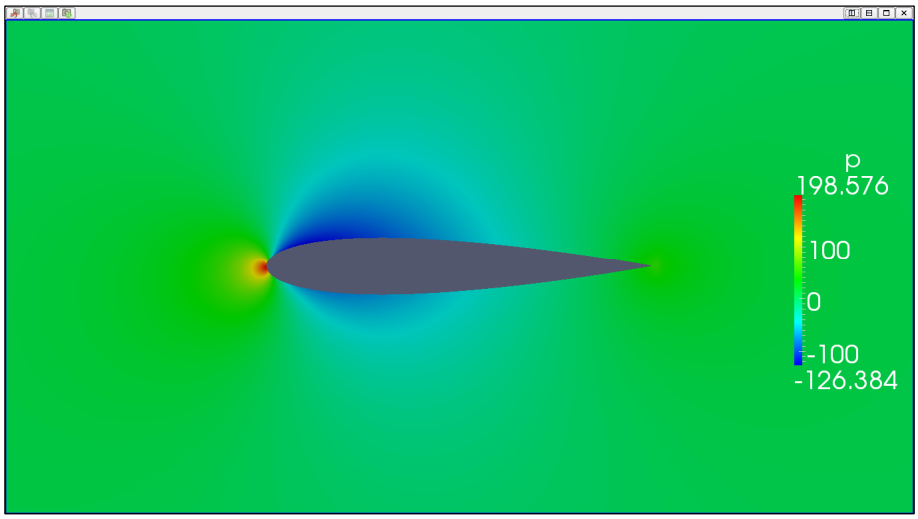
ForceCoeffs output:

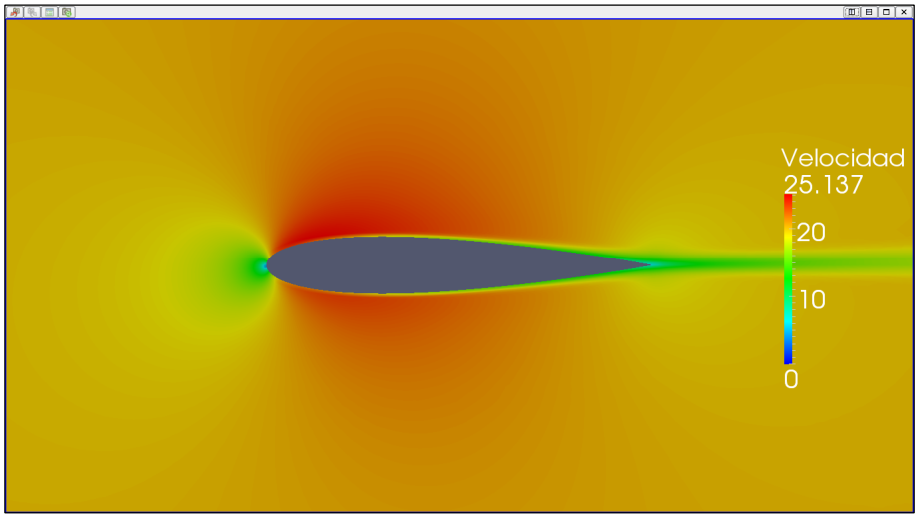
$C_L = 0{,}1065$

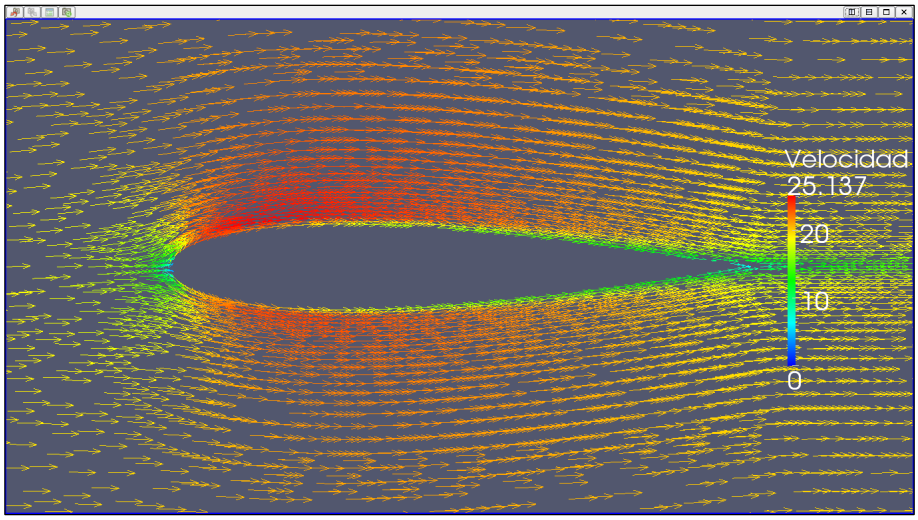
$C_D = 0{,}0141$






p
198.576
100
0
-100
-126.384


Velocidad
25.137
20
10
0


Velocidad
25.137
20
10
0

II.2.3.3. Caso 3 (Ángulo de ataque 1º)

Time: 50000

Solving U_X: Initial residual $= 4{,}670 \cdot 10^{-06}$, Final residual $= 2{,}846 \cdot 10^{-07}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 5{,}148 \cdot 10^{-06}$, Final residual $= 3{,}500 \cdot 10^{-07}$, No Iterations $= 4$.

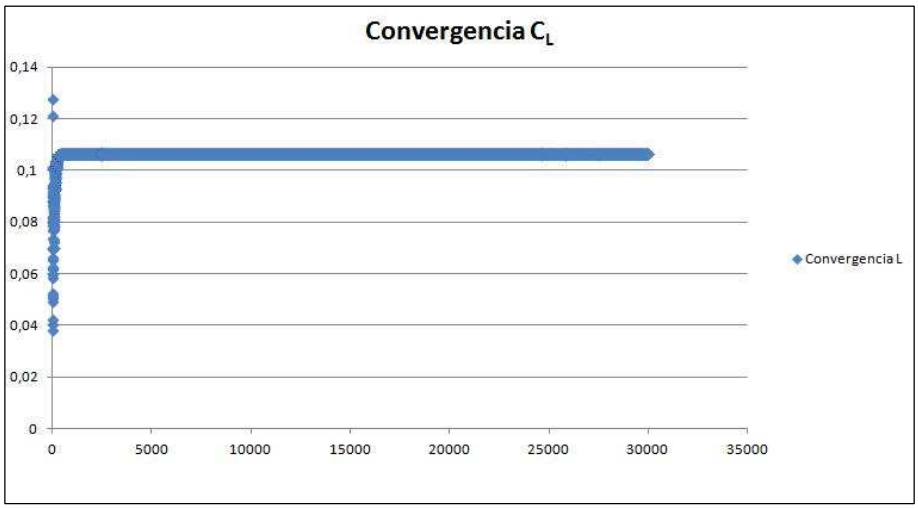
Solving p: Initial residual $= 1{,}575 \cdot 10^{-04}$, Final residual $= 1{,}497 \cdot 10^{-05}$, No Iterations $= 2$.

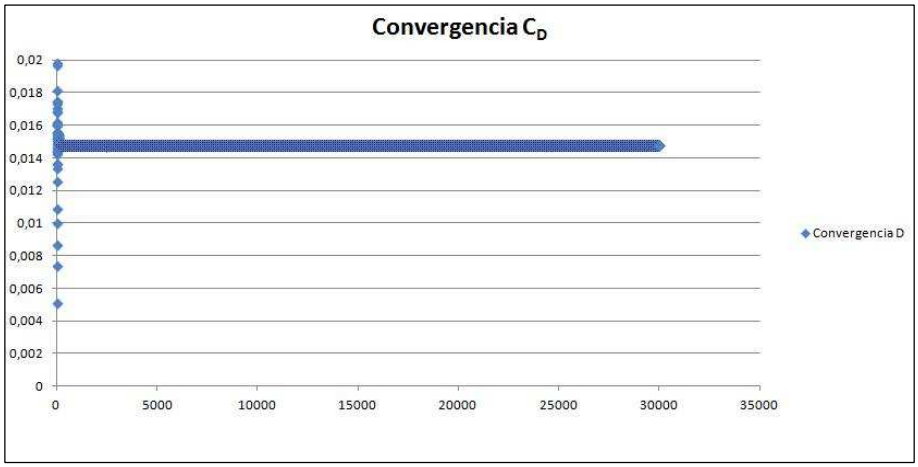
ExecutionTime: 5133,79 s. ClockTime: 5142 s.

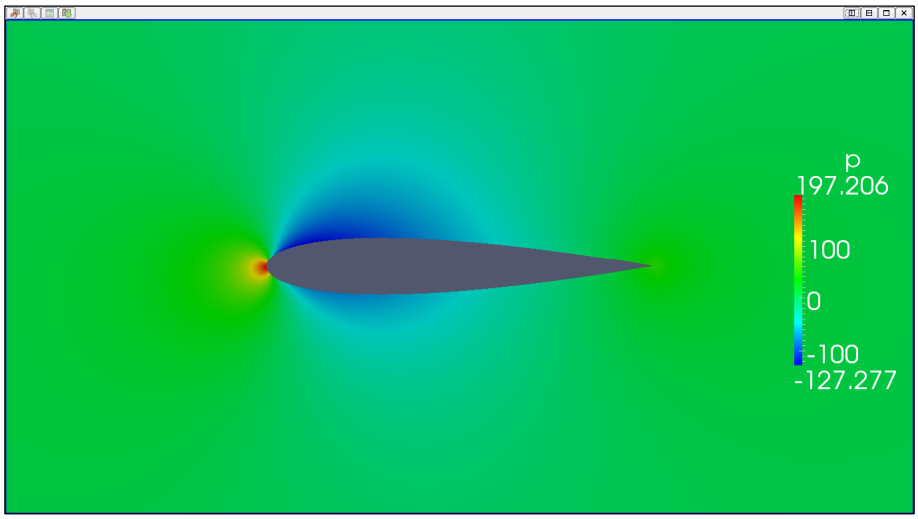
ForceCoeffs output:

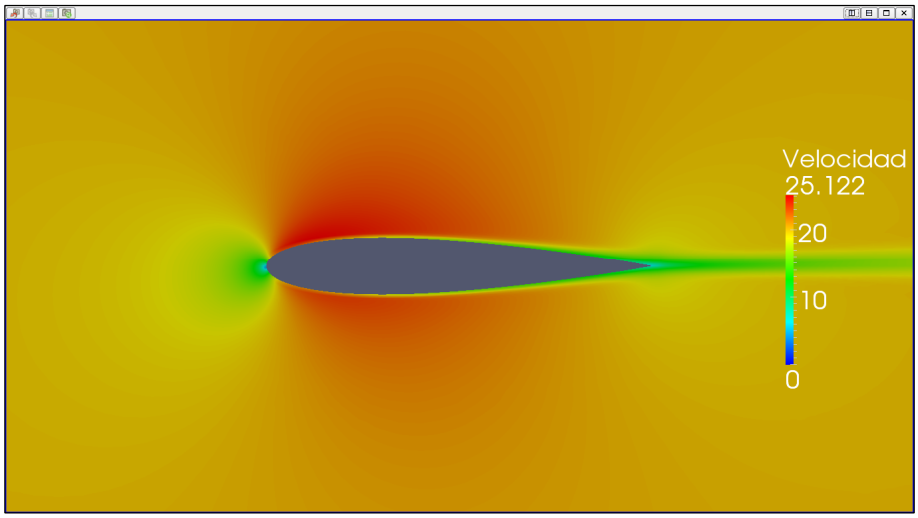
$C_L = 0{,}1063$

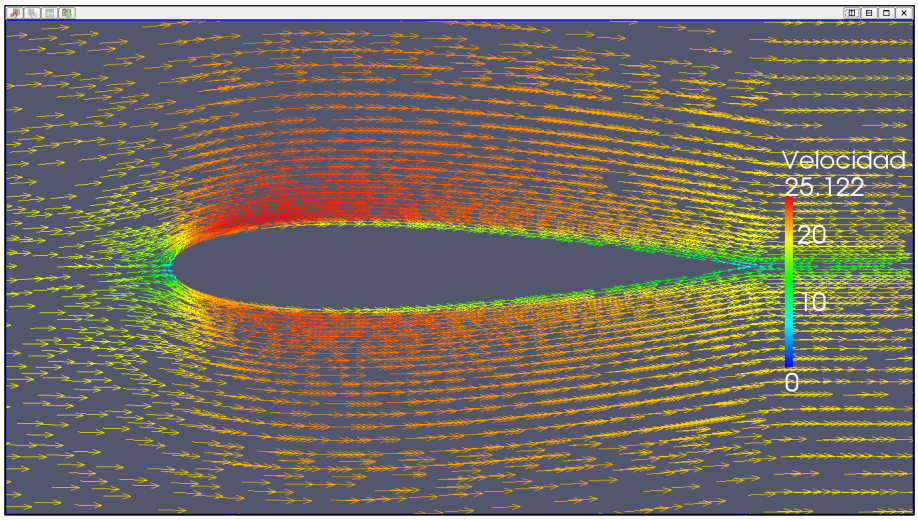
$C_D = 0{,}0147$






p
197.206
100
0
-100
-127.277


Velocidad
25.122
20
10
0


Velocidad
25.122
20
10
0

II.2.4. Resultados ángulo de ataque 2º

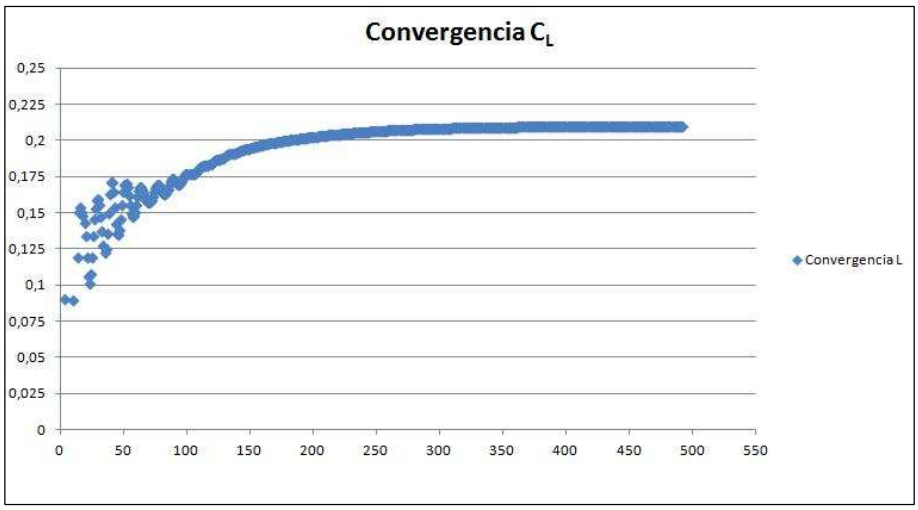
II.2.4.1. Caso 1 (Ángulo de ataque 2º)

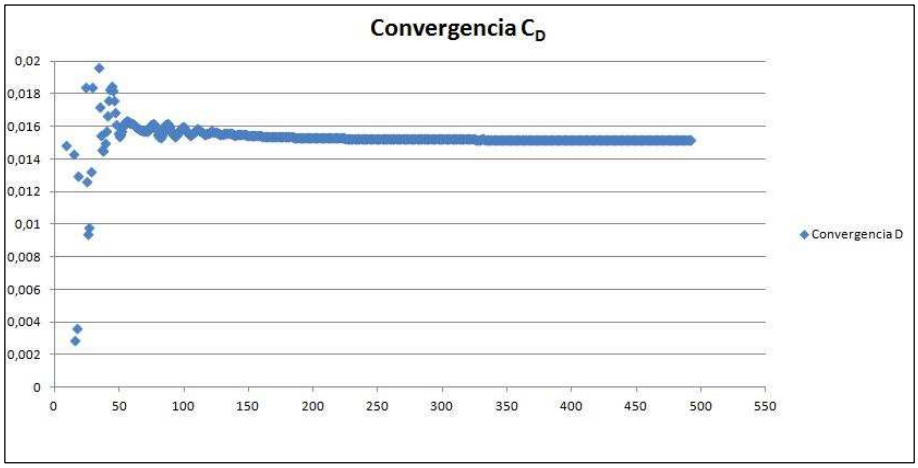
SIMPLE solution converged in 492 iterations.

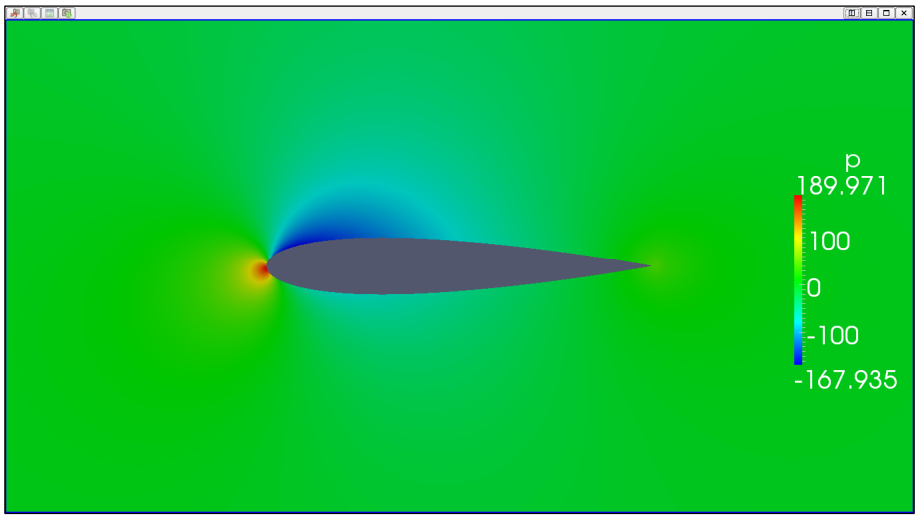
ForceCoeffs output:

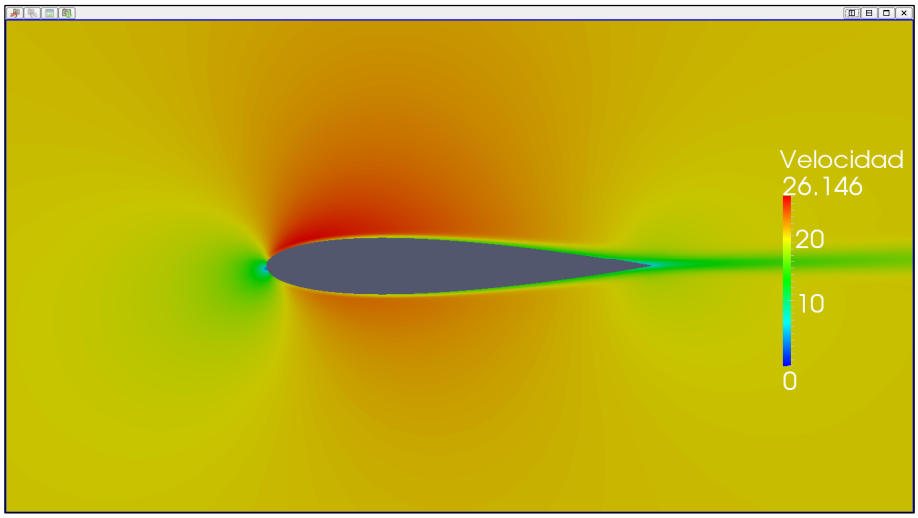
$C_L = 0{,}2100$

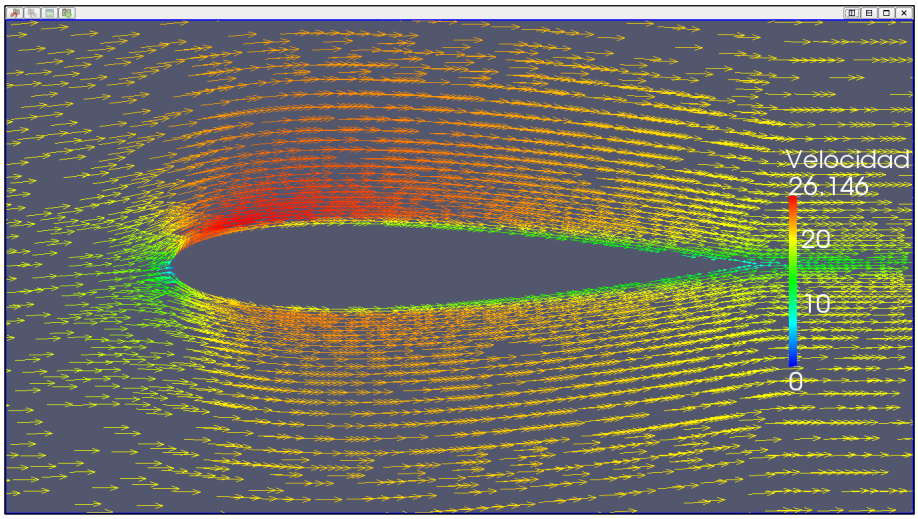
$C_D = 0{,}0152$






p
189.971
100
0
-100
-167.935


Velocidad
26.146
20
10
0


Velocidad
26.146
20
10
0

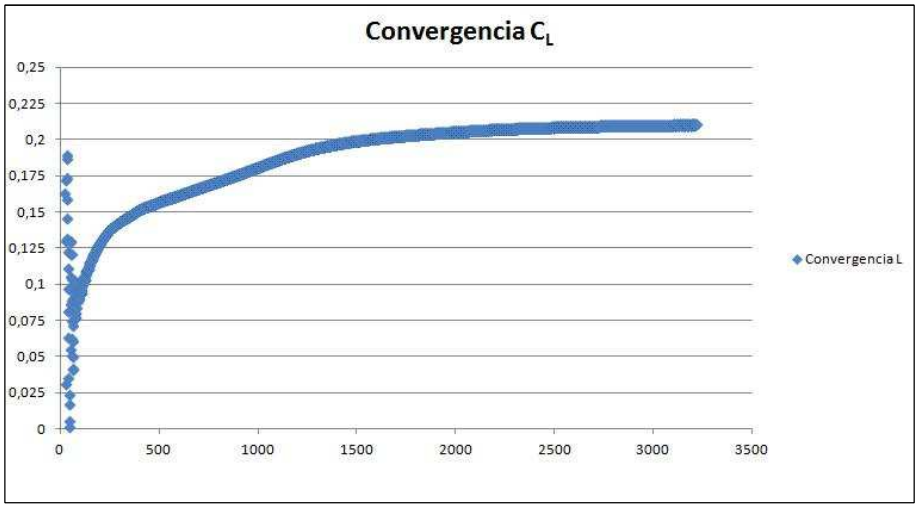
II.2.4.2. Caso 2 (Ángulo de ataque 2º)

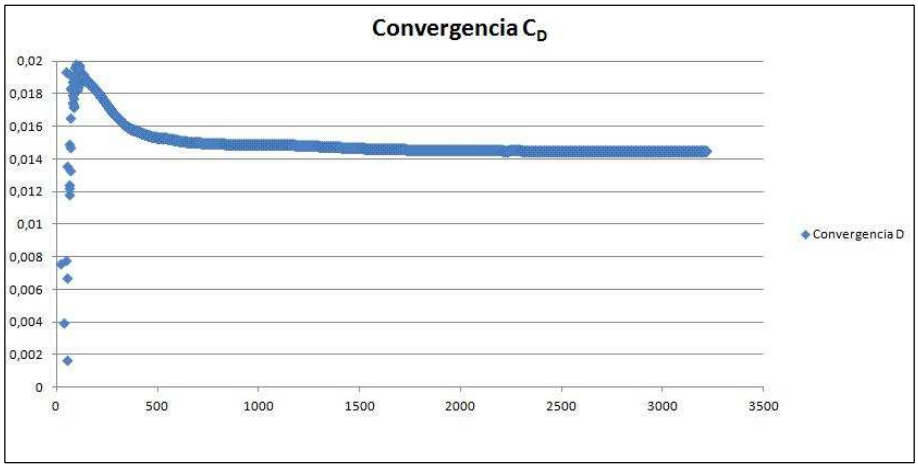
SIMPLE solution converged in 3219 iterations.

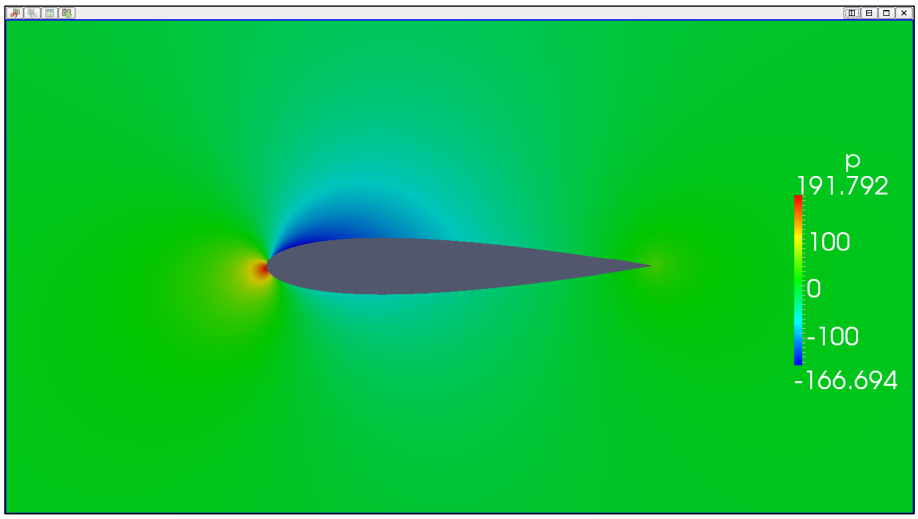
ForceCoeffs output:

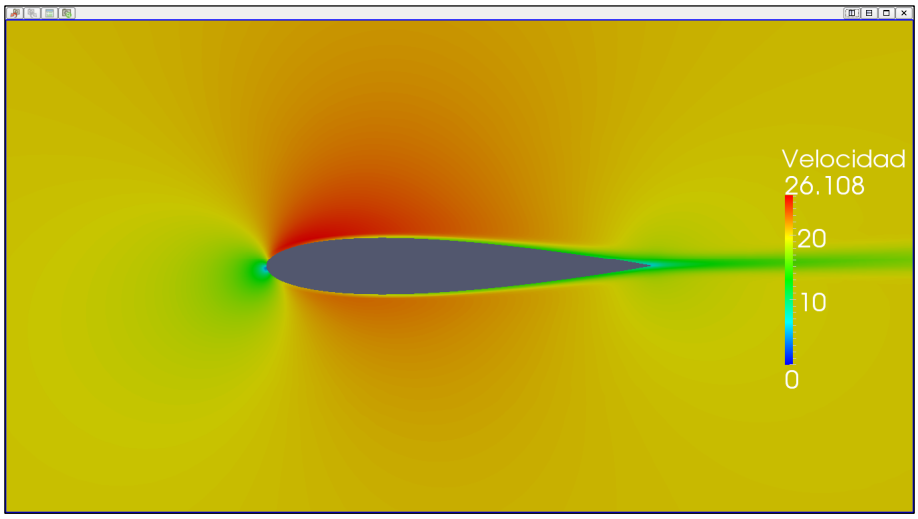
$C_L = 0{,}2102$

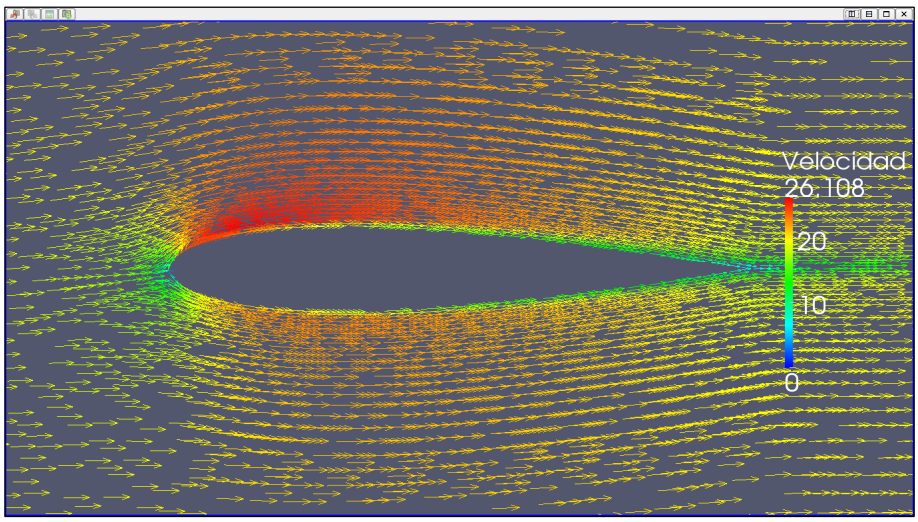
$C_D = 0{,}0145$






p
191.792
100
0
-100
-166.694


Velocidad
26.108
20
10
0


Velocidad
26.108
20
10
0

II.2.4.3. Caso 3 (Ángulo de ataque 2º)

Time: 50000

Solving U_X: Initial residual $= 8{,}948 \cdot 10^{-06}$, Final residual $= 5{,}959 \cdot 10^{-07}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 1{,}086 \cdot 10^{-05}$, Final residual $= 6{,}970 \cdot 10^{-07}$, No Iterations $= 4$.

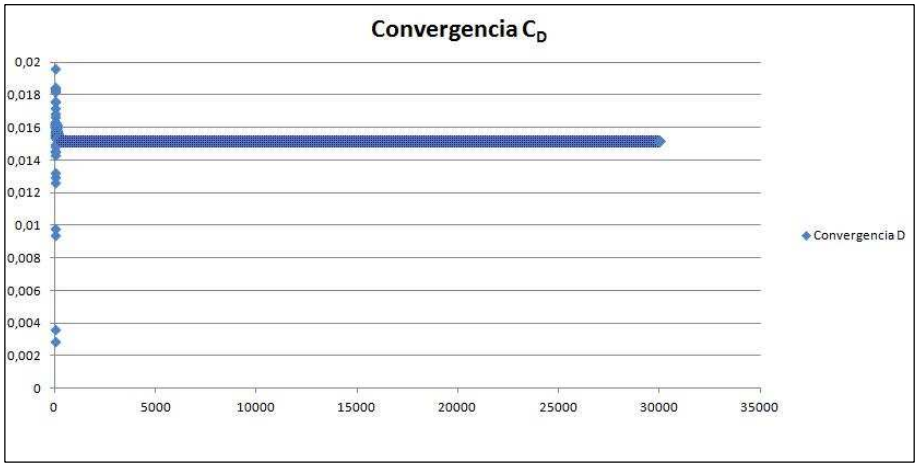
Solving p: Initial residual $= 2{,}403 \cdot 10^{-04}$, Final residual $= 2{,}307 \cdot 10^{-05}$, No Iterations $= 2$.

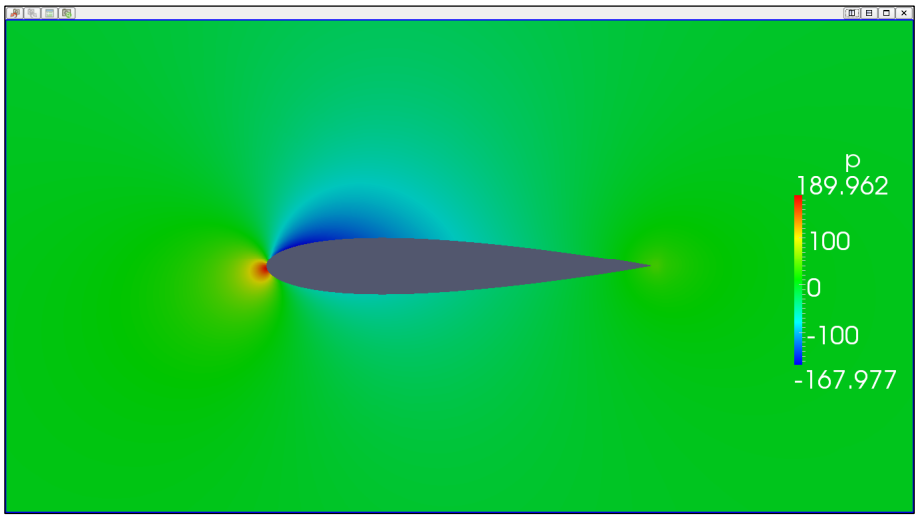
ExecutionTime: 5120,88 s. ClockTime: 5129 s.

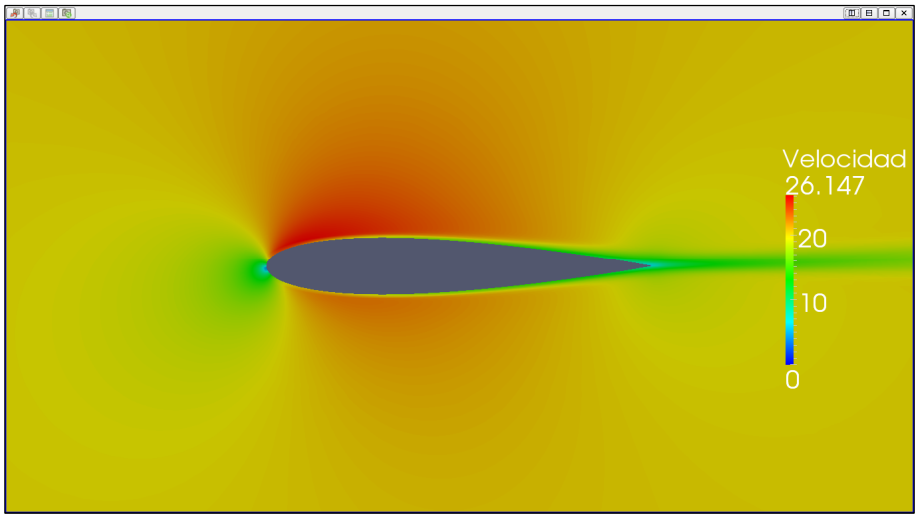
ForceCoeffs output:

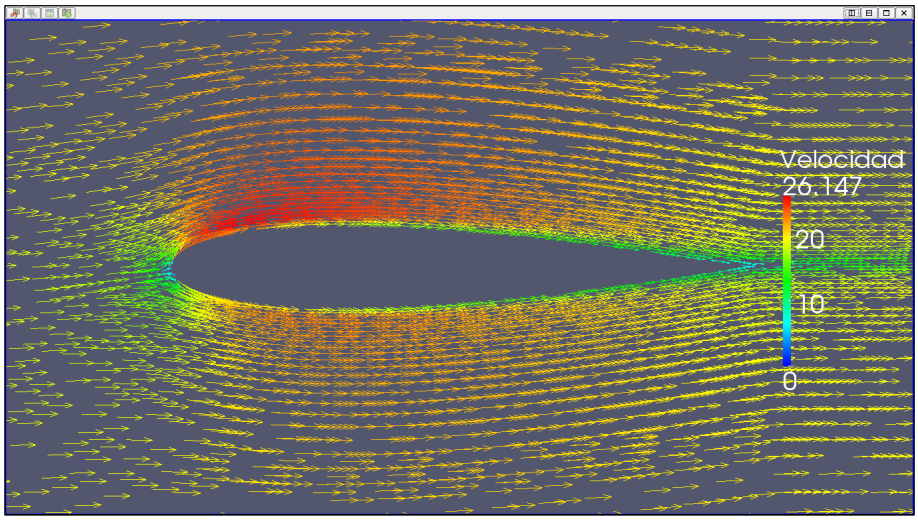
$C_L = 0{,}2100$

$C_D = 0{,}0152$




p
189.962
100
0
-100
-167.977


Velocidad
26.147
20
10
0


Velocidad
26.147
20
10
0

II.2.5. Resultados ángulo de ataque 3º

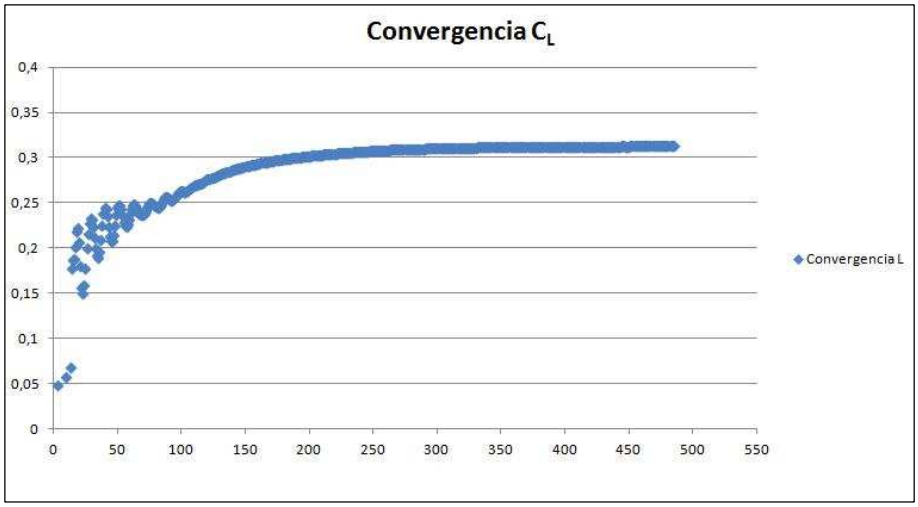
II.2.5.1. Caso 1 (Ángulo de ataque 3º)

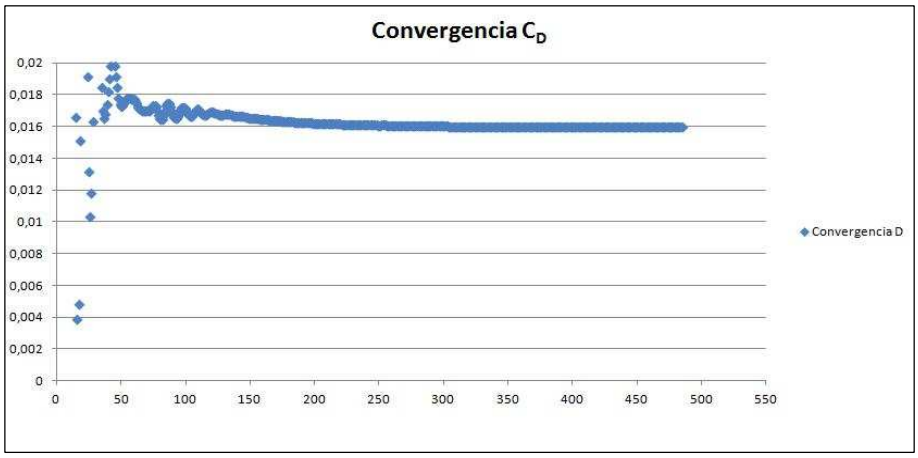
SIMPLE solution converged in 485 iterations.

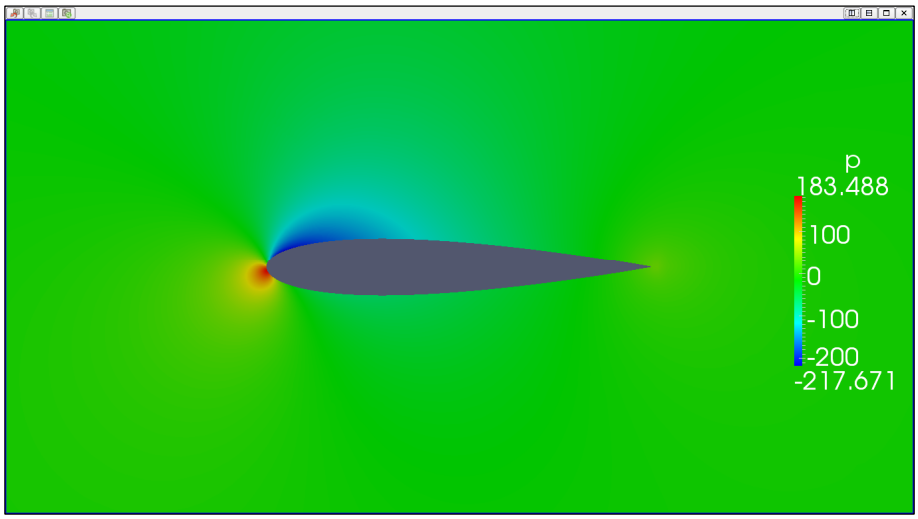
ForceCoeffs output:

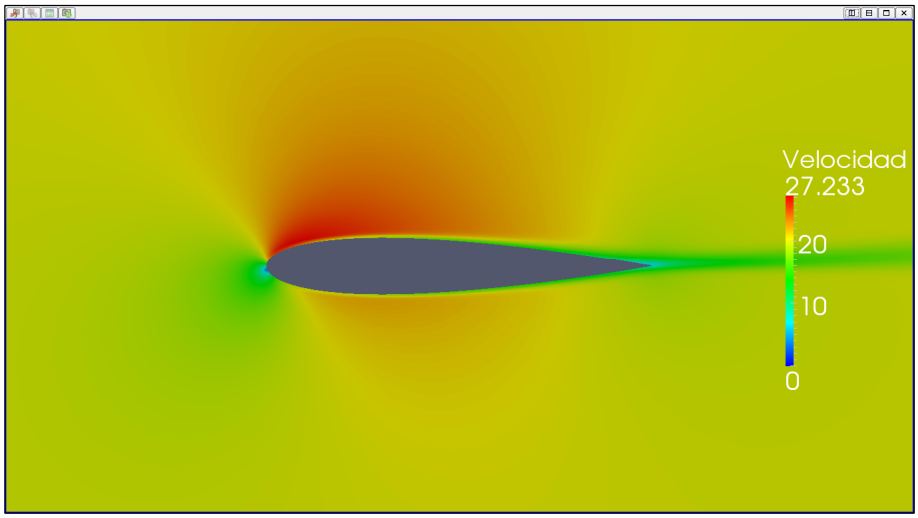
$C_L = 0{,}3126$

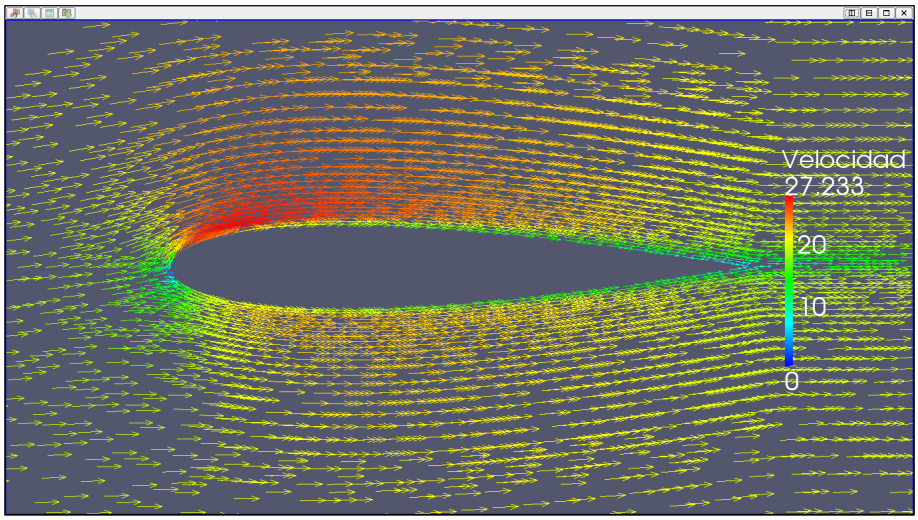
$C_D = 0{,}0159$






p
183.488
100
0
-100
-200
-217.671


Velocidad
27.233
20
10
0


Velocidad
27.233
20
10
0

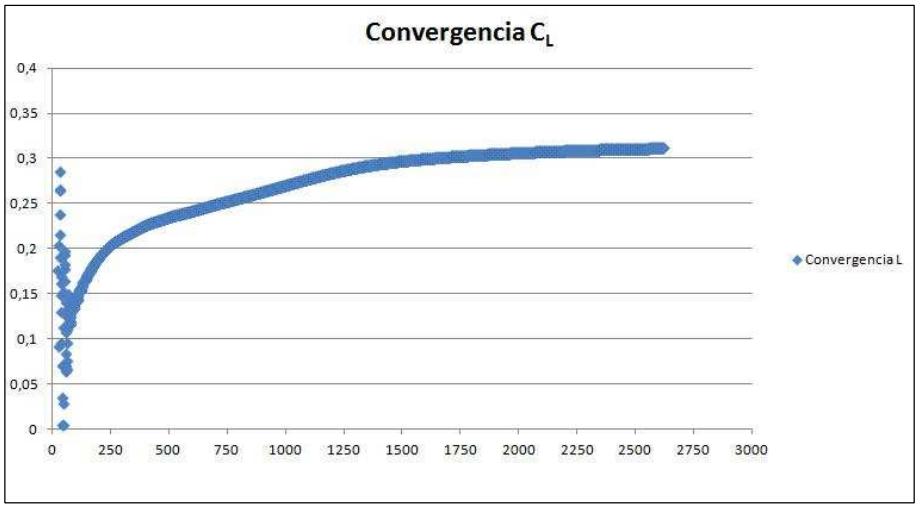
II.2.5.2. Caso 2 (Ángulo de ataque 3º)

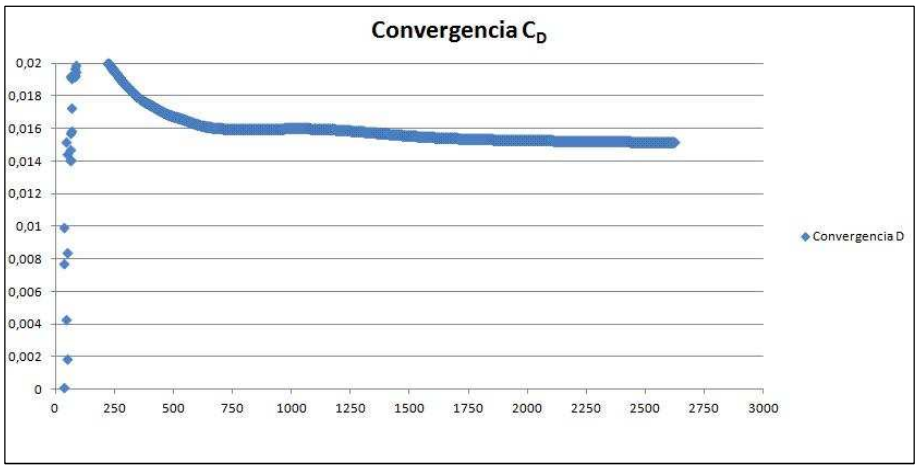
SIMPLE solution converged in 2622 iterations.

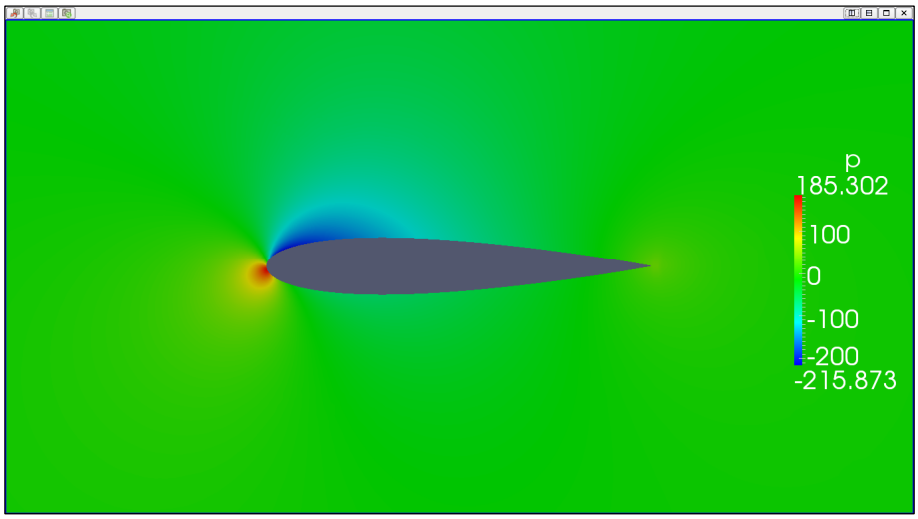
ForceCoeffs output:

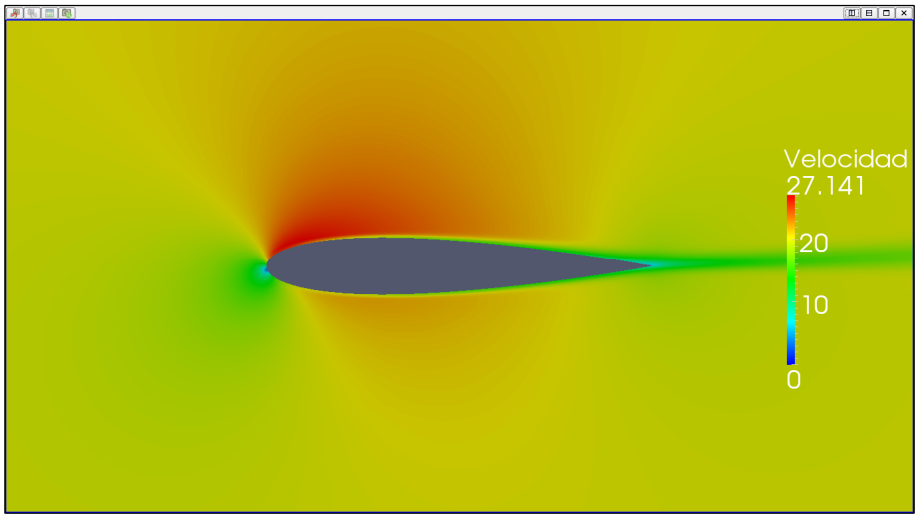
$C_L = 0{,}3115$

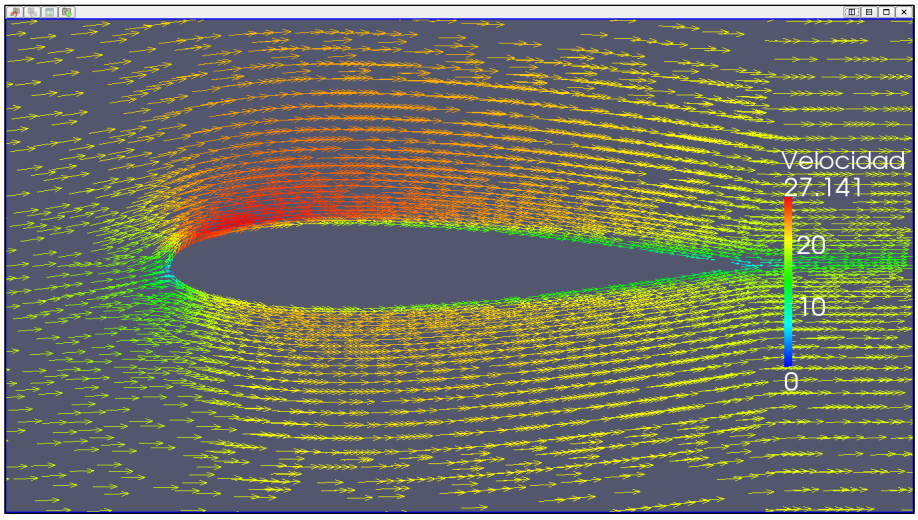
$C_D = 0{,}0152$






p
185.302
100
0
-100
-200
-215.873


Velocidad
27.141
20
10
0


Velocidad
27.141
20
10
0

II.2.5.3. Caso 3 (Ángulo de ataque 3º)

Time: 50000

Solving U_X: Initial residual $= 1{,}120 \cdot 10^{-05}$, Final residual $= 8{,}270 \cdot 10^{-07}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 7{,}249 \cdot 10^{-06}$, Final residual $= 5{,}182 \cdot 10^{-07}$, No Iterations $= 4$.

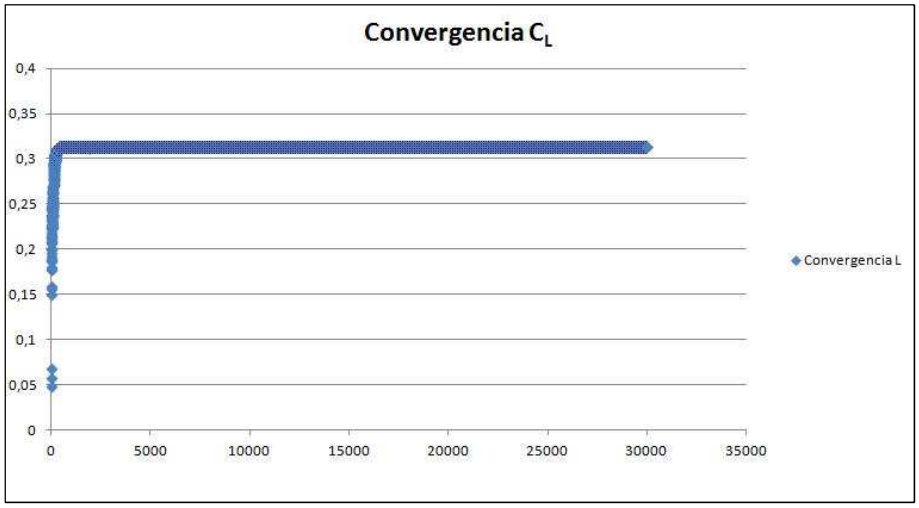
Solving p: Initial residual $= 2{,}819 \cdot 10^{-04}$, Final residual $= 1{,}574 \cdot 10^{-05}$, No Iterations $= 3$.

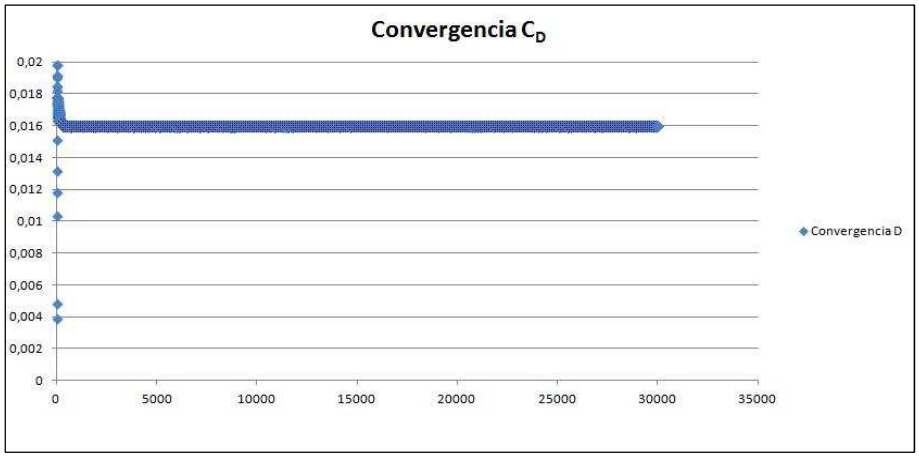
ExecutionTime: 5329,15 s. ClockTime: 5339 s.

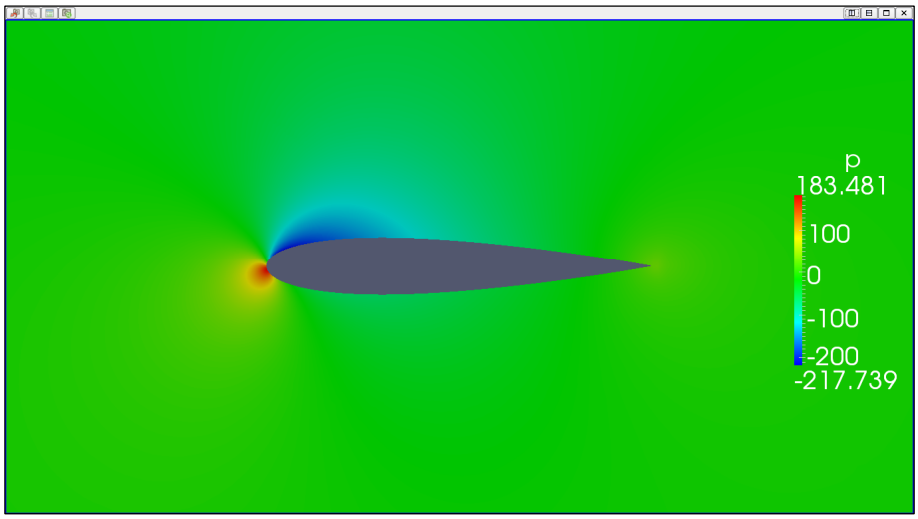
ForceCoeffs output:

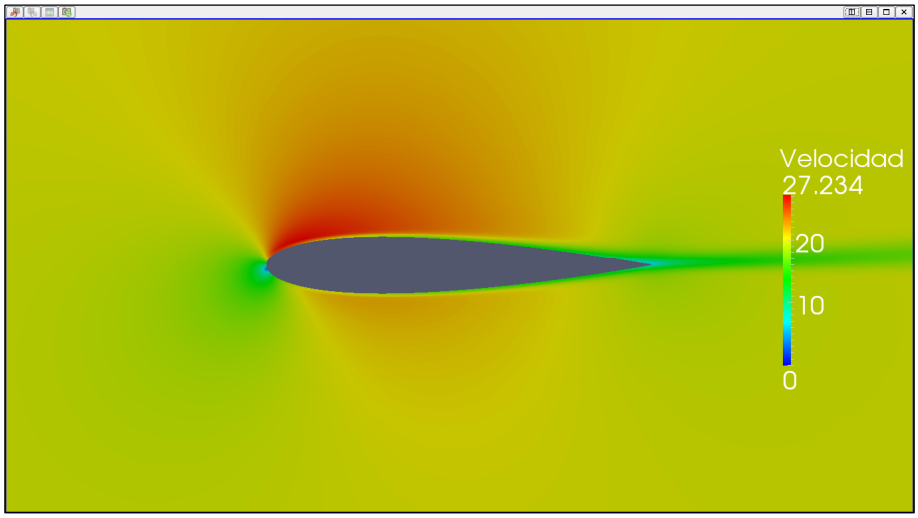
$C_L = 0{,}3128$

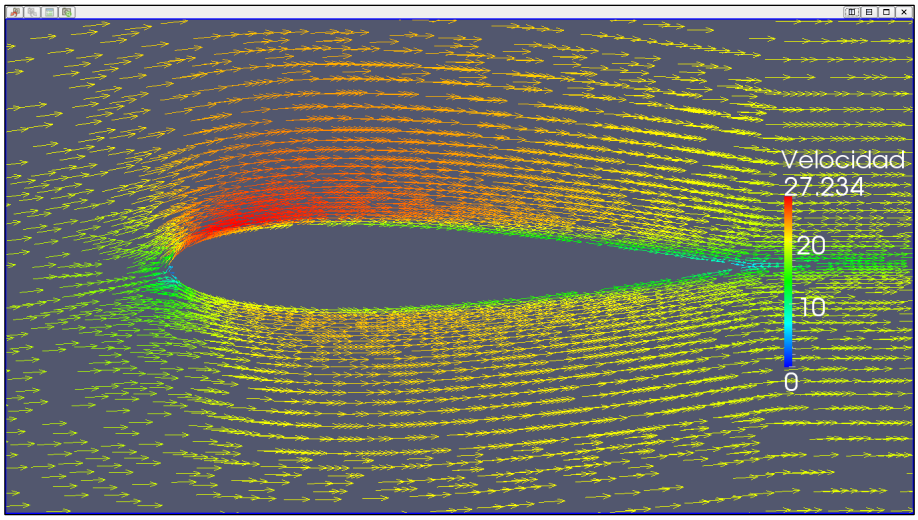
$C_D = 0{,}0159$






p
183.481
100
0
-100
-200
-217.739


Velocidad
27.234
20
10
0


Velocidad
27.234
20
10
0

II.2.6. Resultados ángulo de ataque 4º

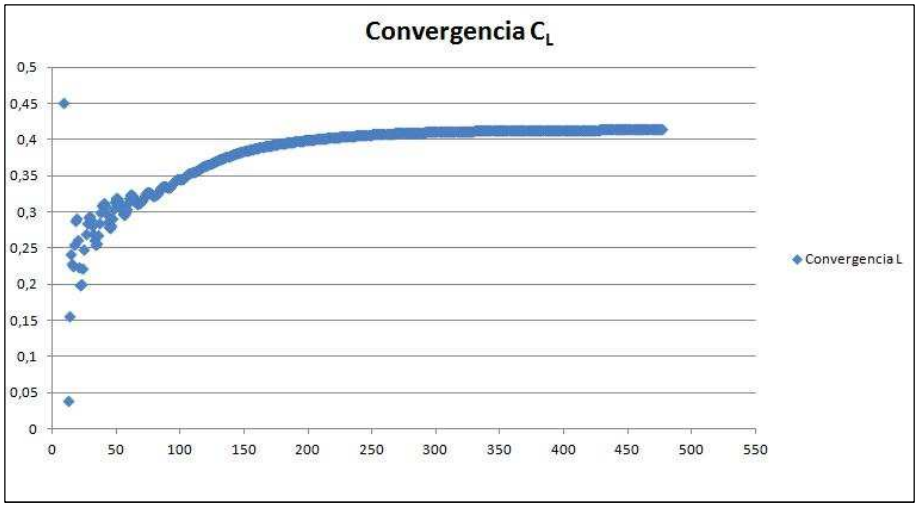
II.2.6.1. Caso 1 (Ángulo de ataque 4º)

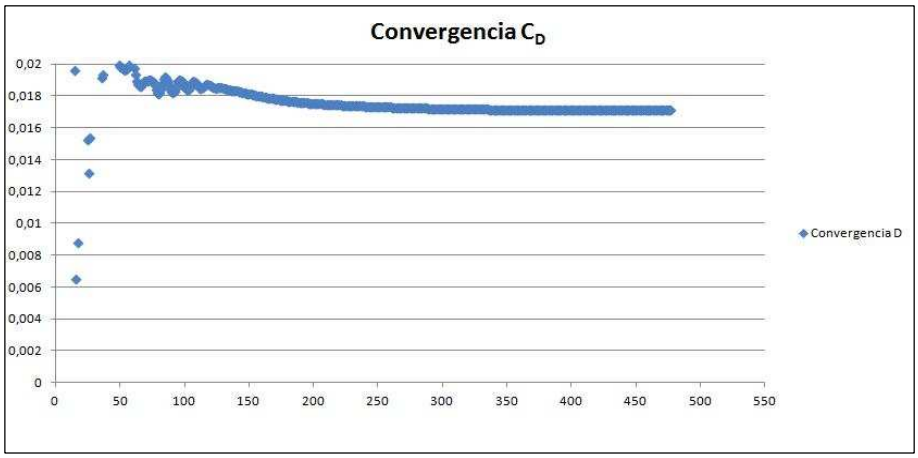
SIMPLE solution converged in 477 iterations.

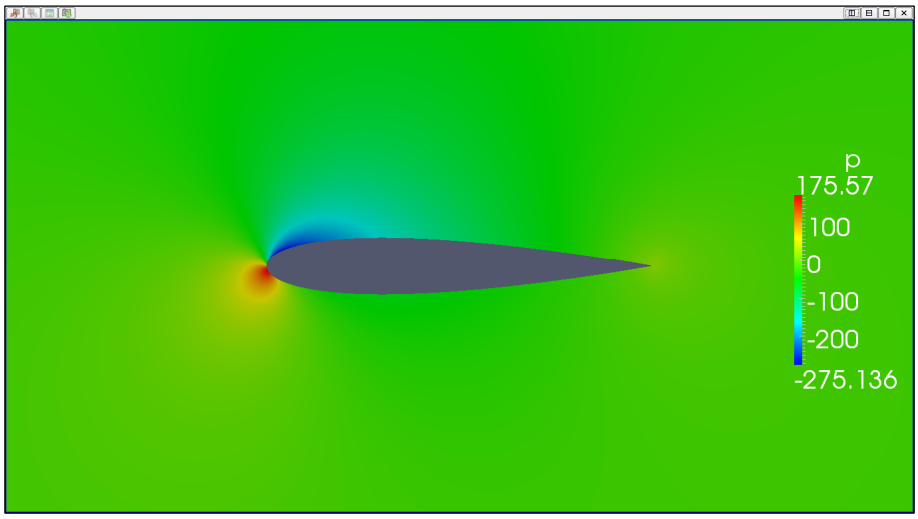
ForceCoeffs output:

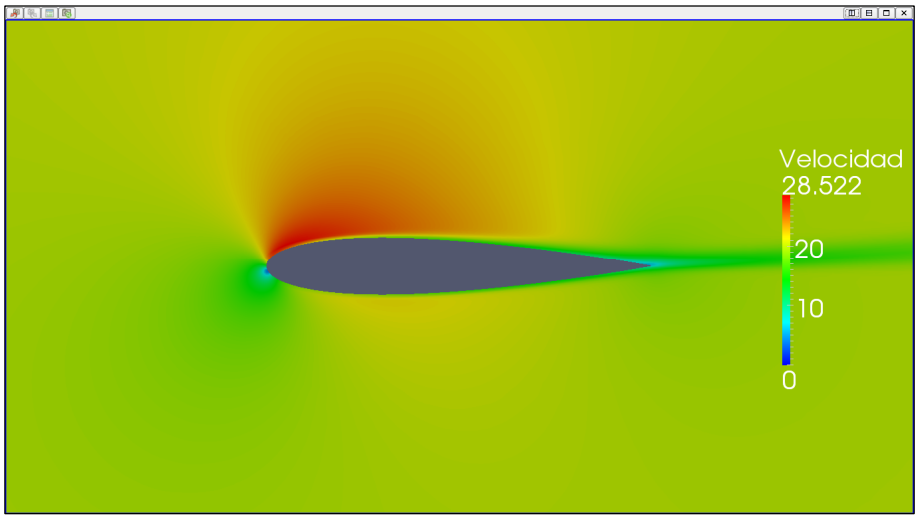
$C_L = 0{,}4140$

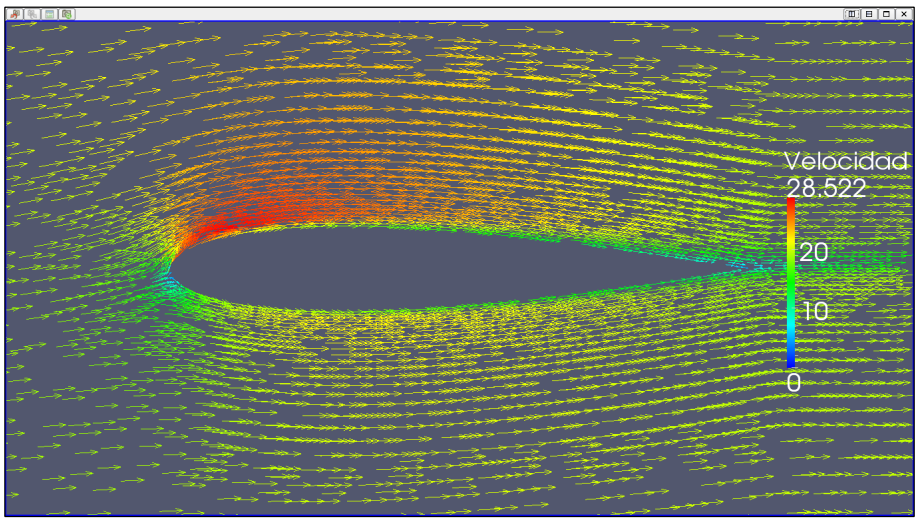
$C_D = 0{,}0171$






p
175.57
100
0
-100
-200
-275.136


Velocidad
28.522
20
10
0


Velocidad
28.522
20
10
0

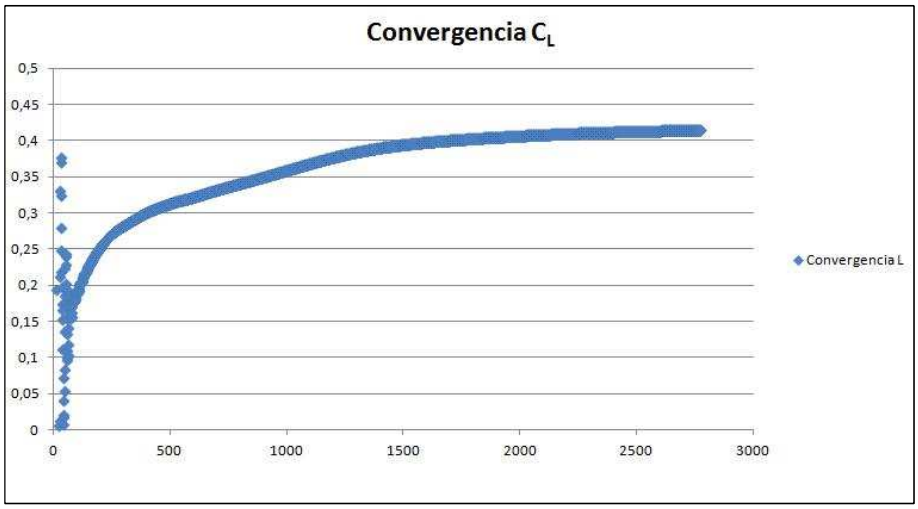
II.2.6.2. Caso 2 (Ángulo de ataque 4º)

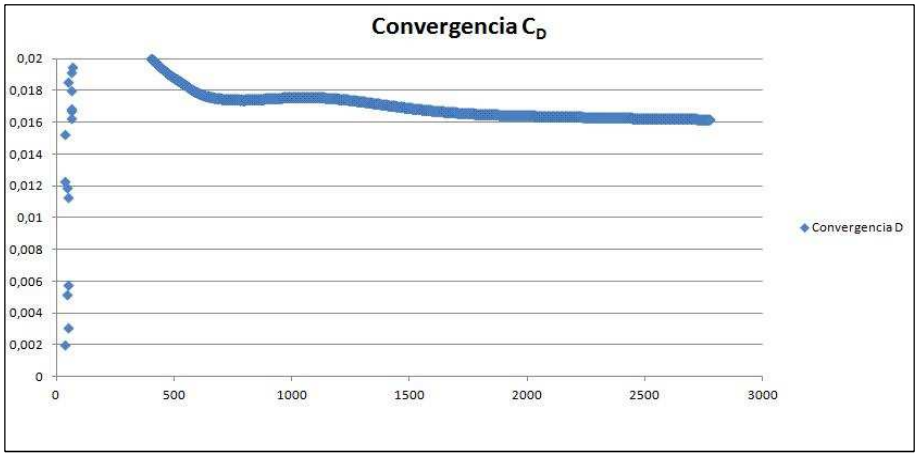
SIMPLE solution converged in 2775 iterations.

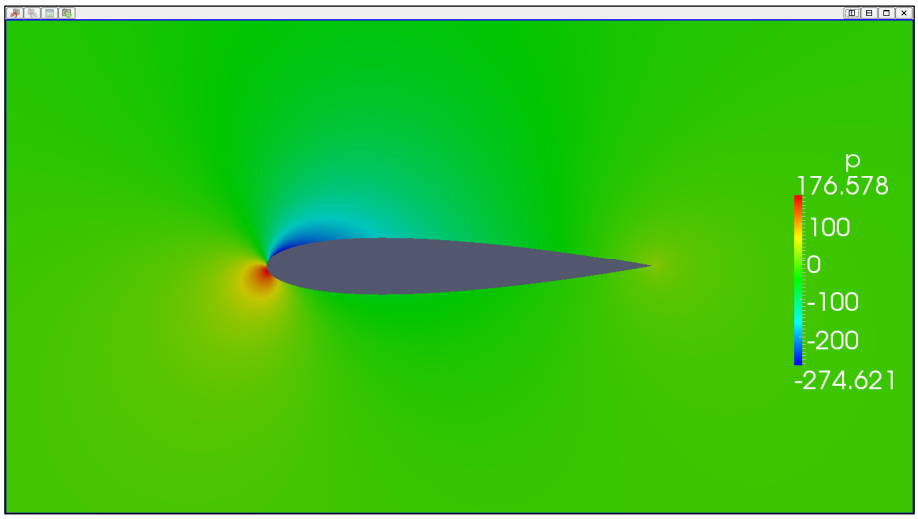
ForceCoeffs output:

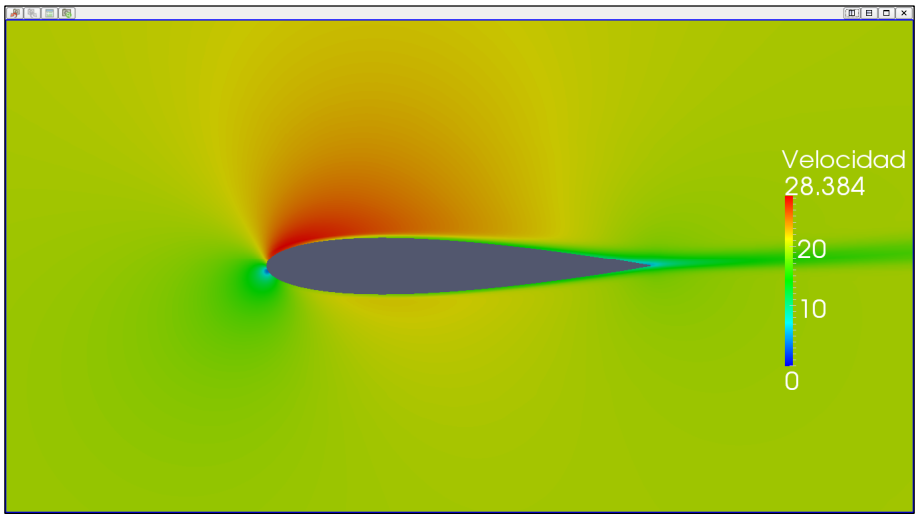
$C_L = 0{,}4146$

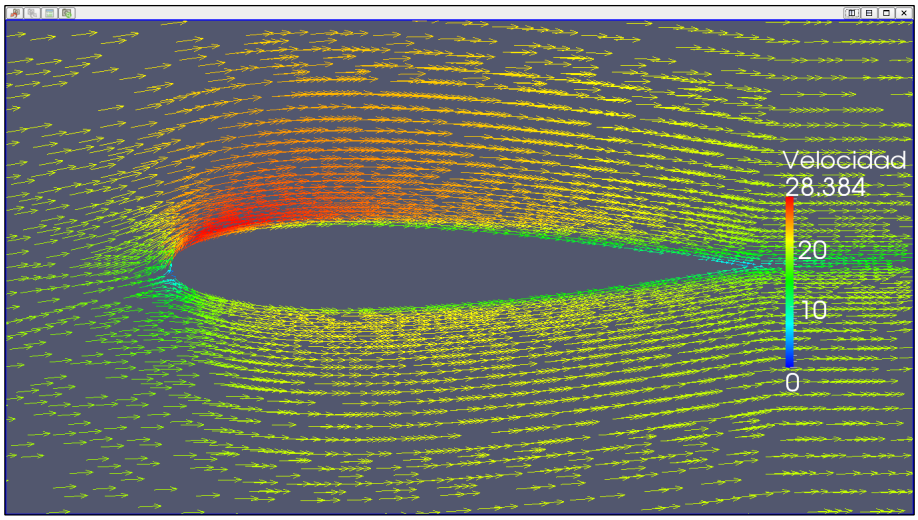
$C_D = 0{,}0162$






p
176.578
100
0
-100
-200
-274.621


Velocidad
28.384
20
10
0


Velocidad
28.384
20
10
0

II.2.6.3. Caso 3 (Ángulo de ataque 4º)

Time: 50000

Solving U_X: Initial residual $= 4{,}298 \cdot 10^{-06}$, Final residual $= 2{,}315 \cdot 10^{-07}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 3{,}932 \cdot 10^{-06}$, Final residual $= 2{,}365 \cdot 10^{-07}$, No Iterations $= 4$.

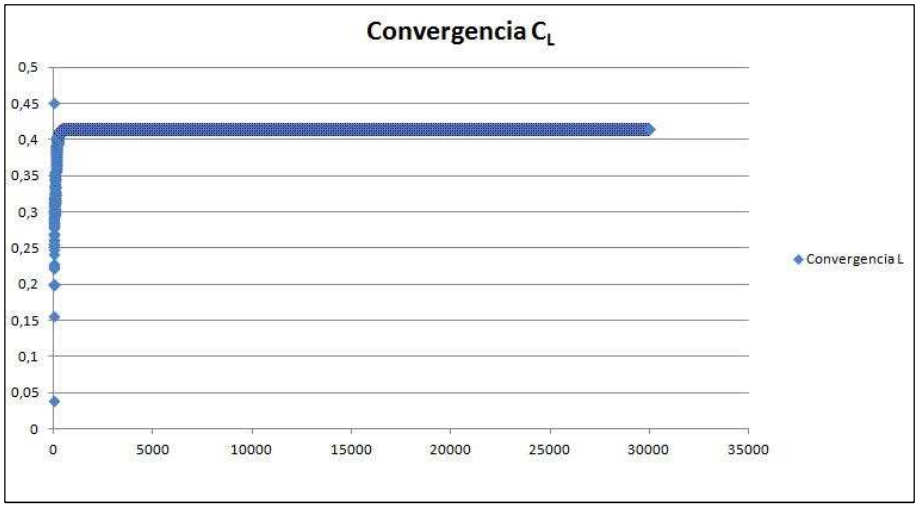
Solving p: Initial residual $= 1{,}422 \cdot 10^{-04}$, Final residual $= 8{,}976 \cdot 10^{-06}$, No Iterations $= 3$.

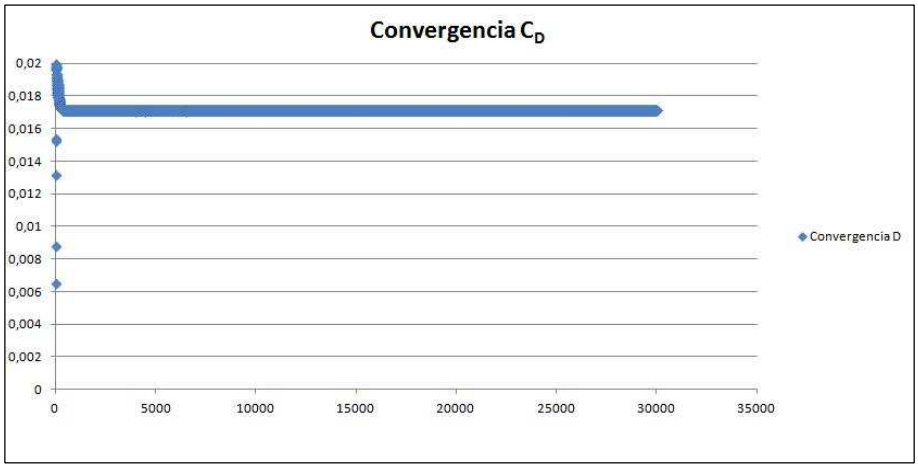
ExecutionTime: 5131,91 s. ClockTime: 5140 s.

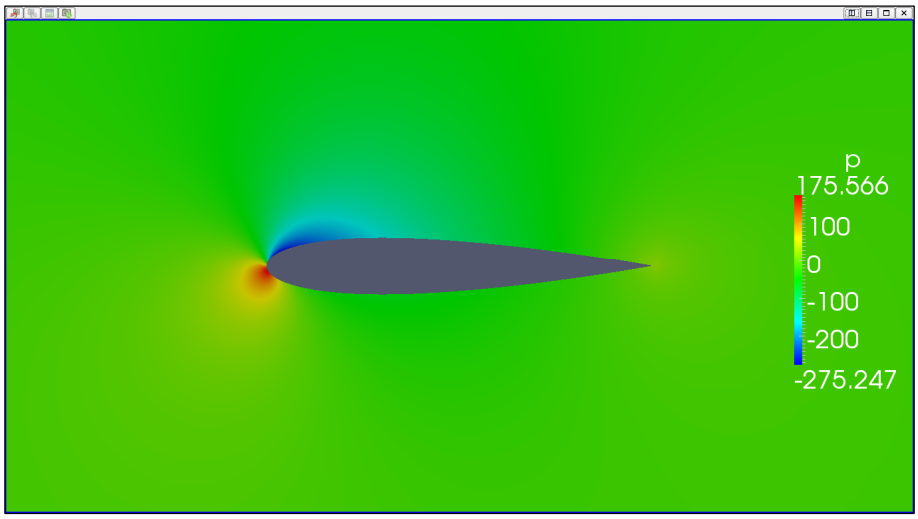
ForceCoeffs output:

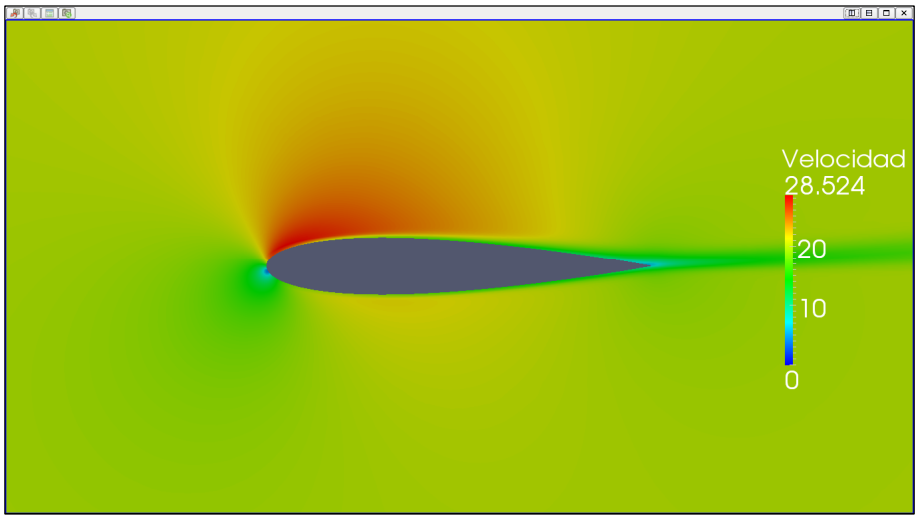
$C_L = 0{,}4142$

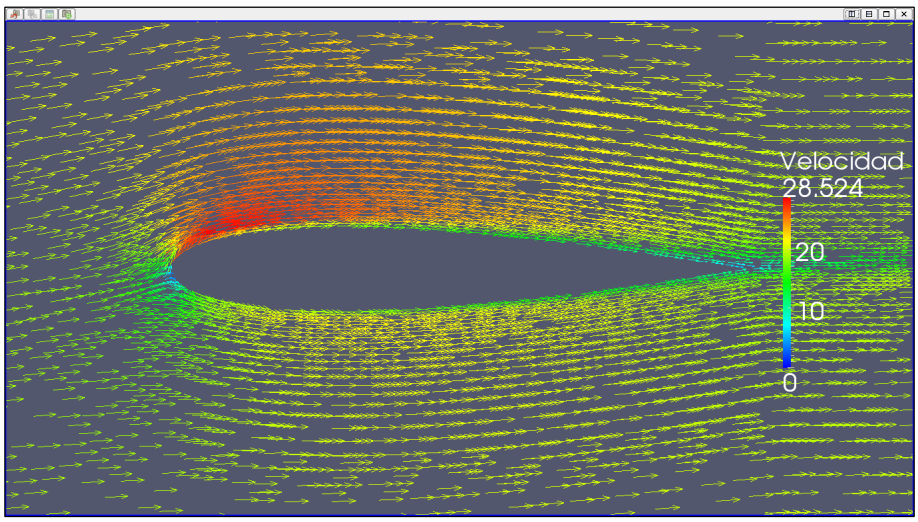
$C_D = 0{,}0171$






p
175.566
100
0
-100
-200
-275.247


Velocidad
28.524
20
10
0


Velocidad
28.524
20
10
0

II.2.7. Resultados ángulo de ataque 5º

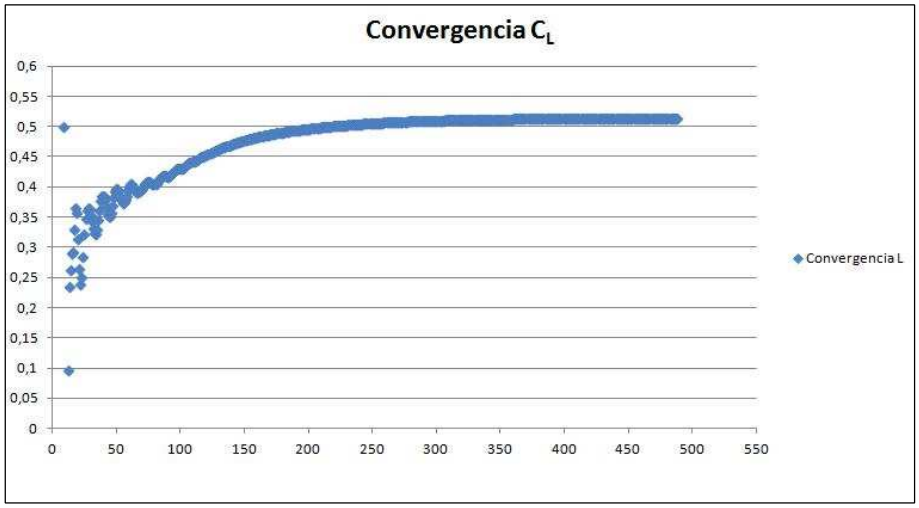
II.2.7.1. Caso 1 (Ángulo de ataque 5º)

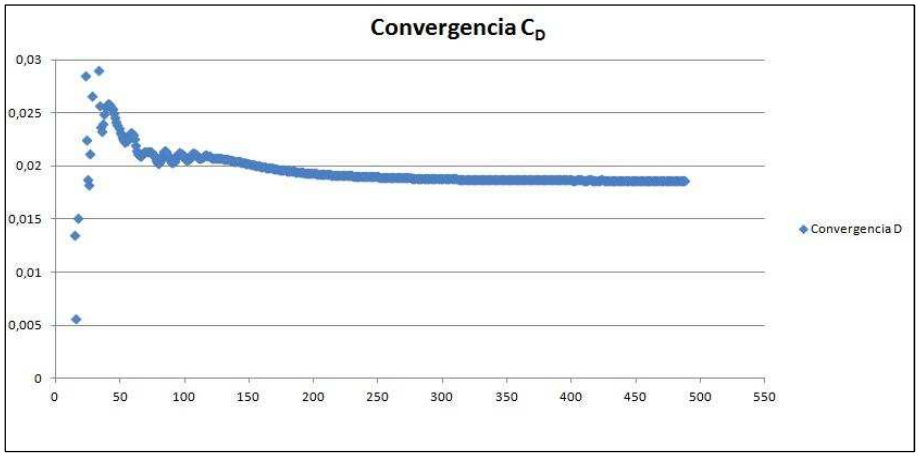
SIMPLE solution converged in 488 iterations.

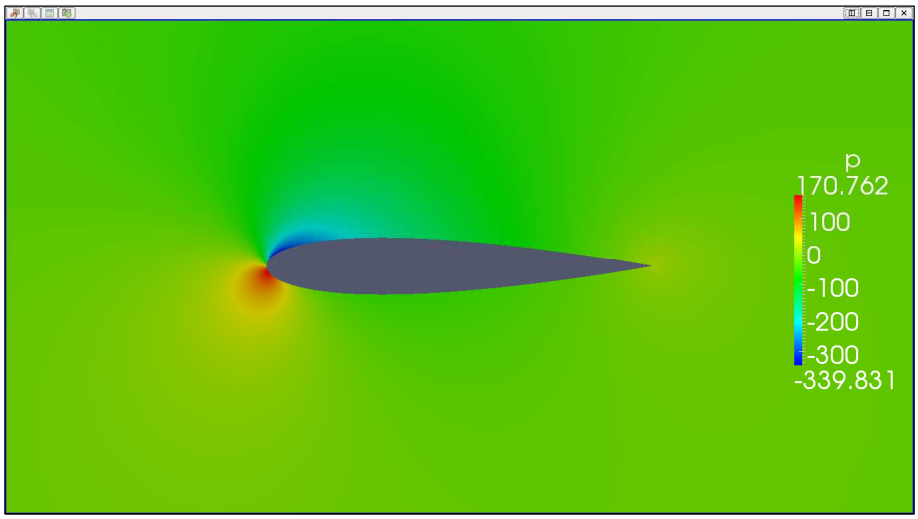
ForceCoeffs output:

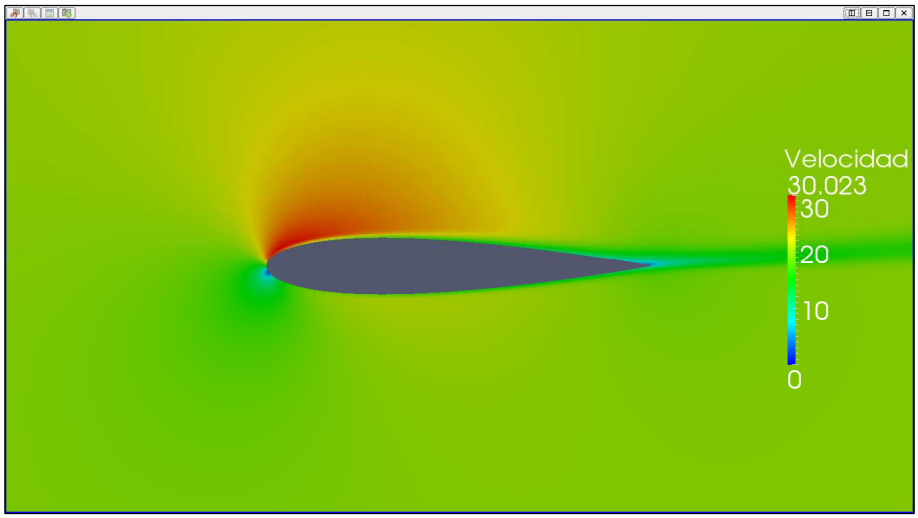
$C_L = 0{,}5137$

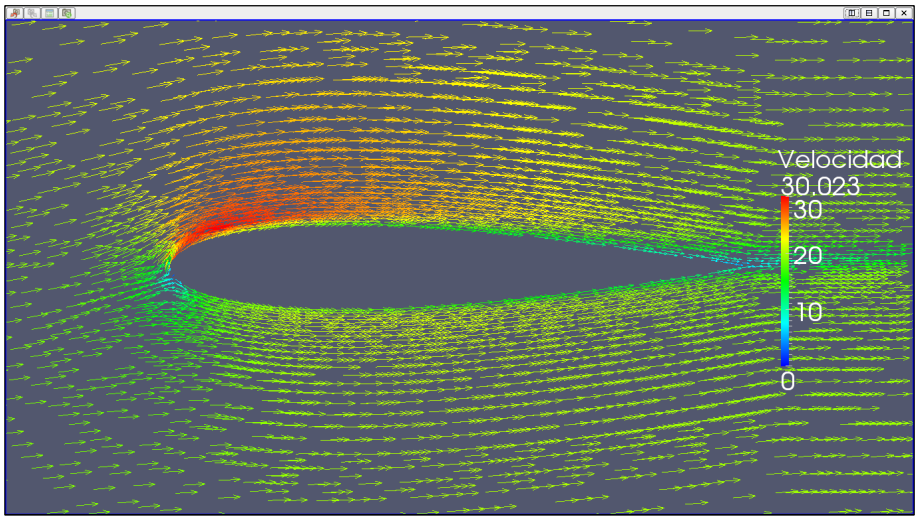
$C_D = 0{,}0186$






p
170.762
100
0
-100
-200
-300
-339.831


Velocidad
30.023
30
20
10
0


Velocidad
30.023
30
20
10
0

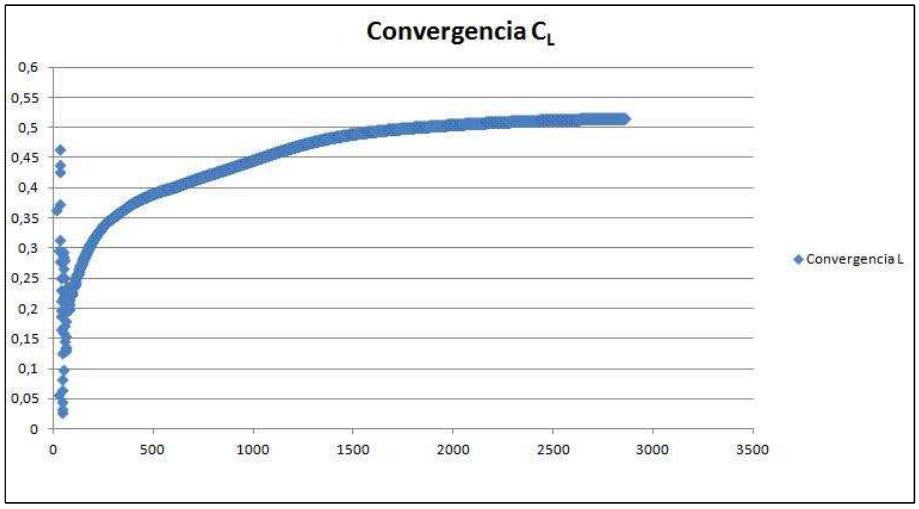
II.2.7.2. Caso 2 (Ángulo de ataque 5º)

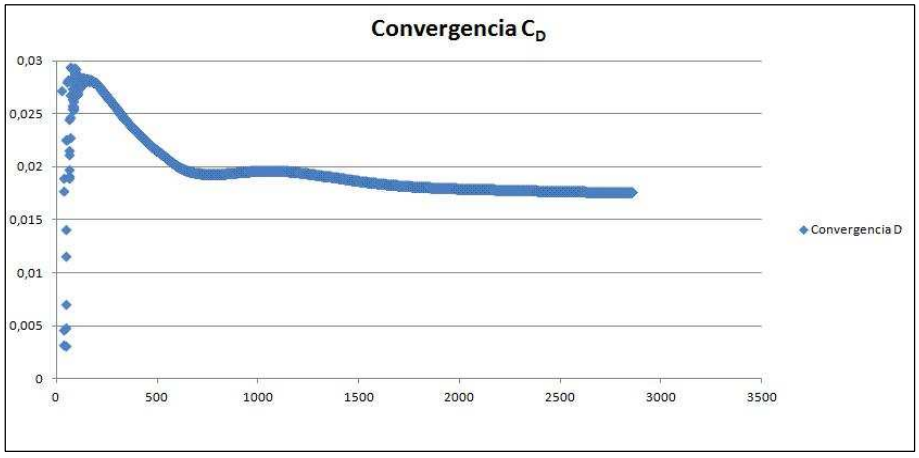
SIMPLE solution converged in 2858 iterations.

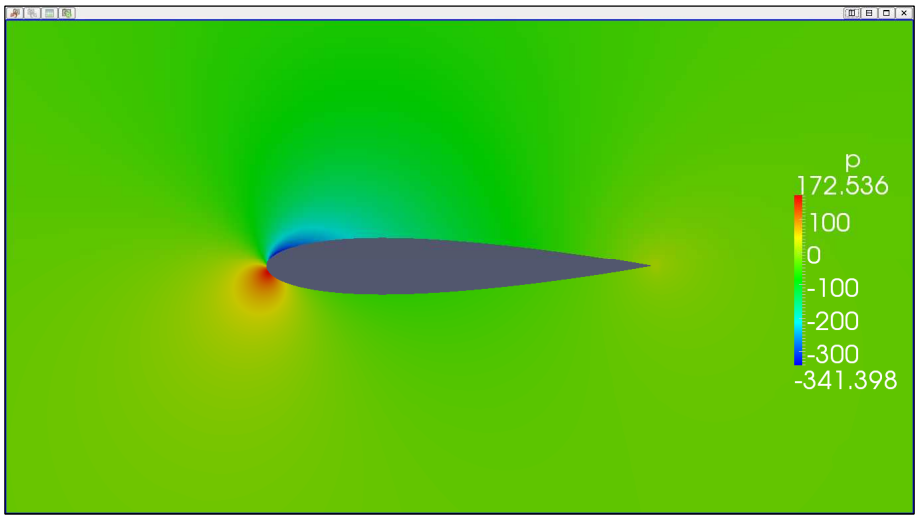
ForceCoeffs output:

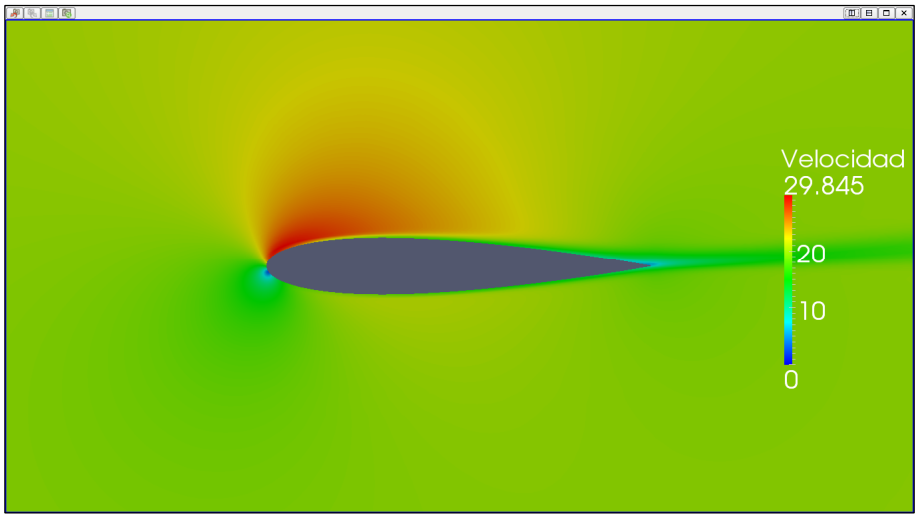
$C_L = 0{,}5157$

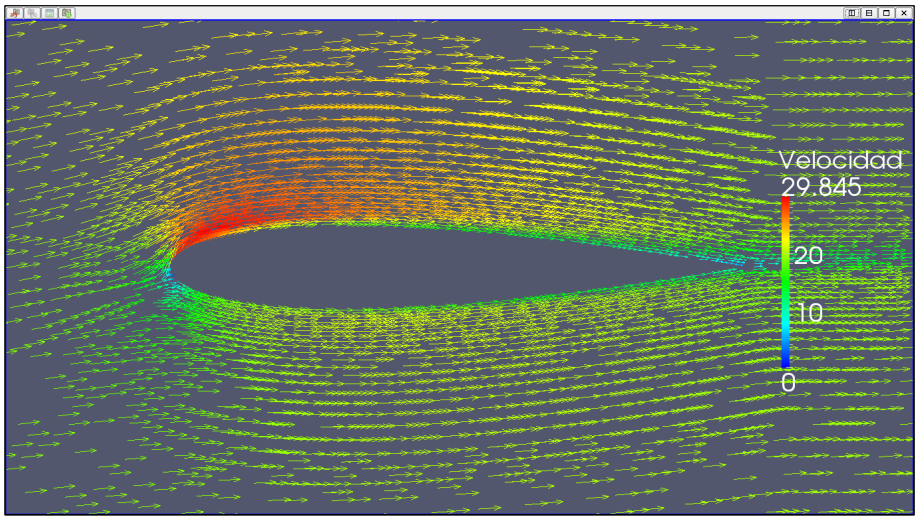
$C_D = 0{,}0176$






p
172.536
100
0
-100
-200
-300
-341.398


Velocidad
29.845
20
10
0


Velocidad
29.845
20
10
0

II.2.7.3. Caso 3 (Ángulo de ataque 5º)

Time: 50000

Solving U_X: Initial residual $= 1{,}395 \cdot 10^{-05}$, Final residual $= 6{,}011 \cdot 10^{-07}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 7{,}708 \cdot 10^{-06}$, Final residual $= 2{,}557 \cdot 10^{-07}$, No Iterations $= 4$.

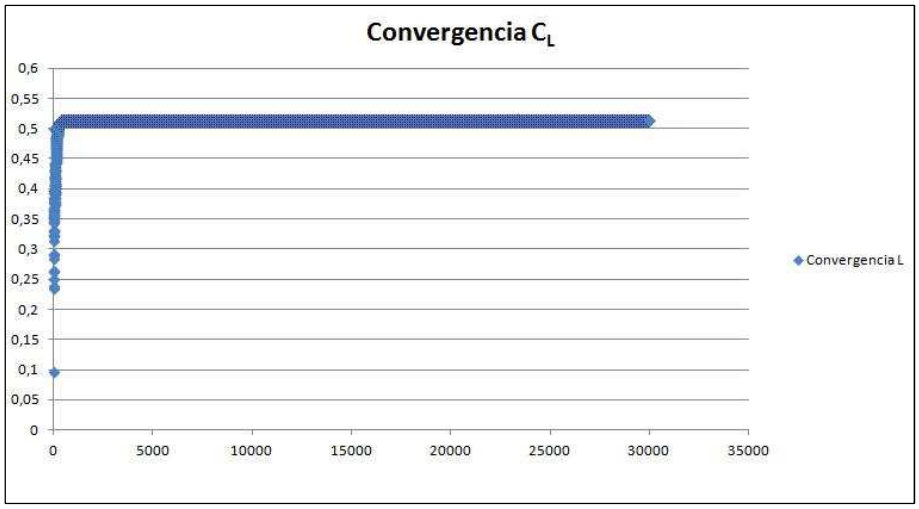
Solving p: Initial residual $= 3{,}931 \cdot 10^{-04}$, Final residual $= 2{,}610 \cdot 10^{-05}$, No Iterations $= 3$.

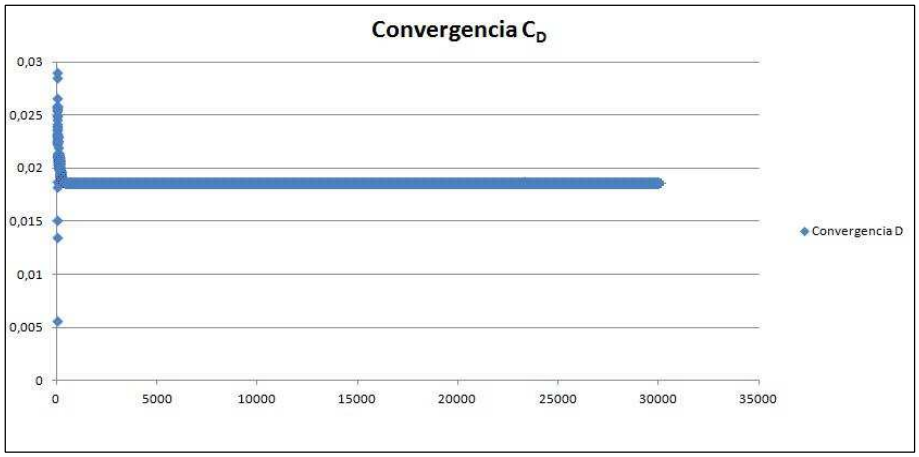
ExecutionTime: 5128,87 s. ClockTime: 5136 s.

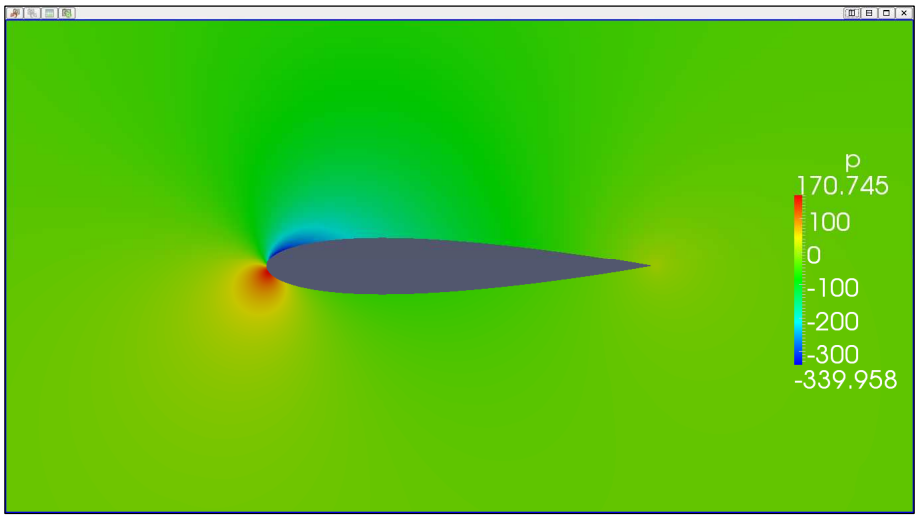
ForceCoeffs output:

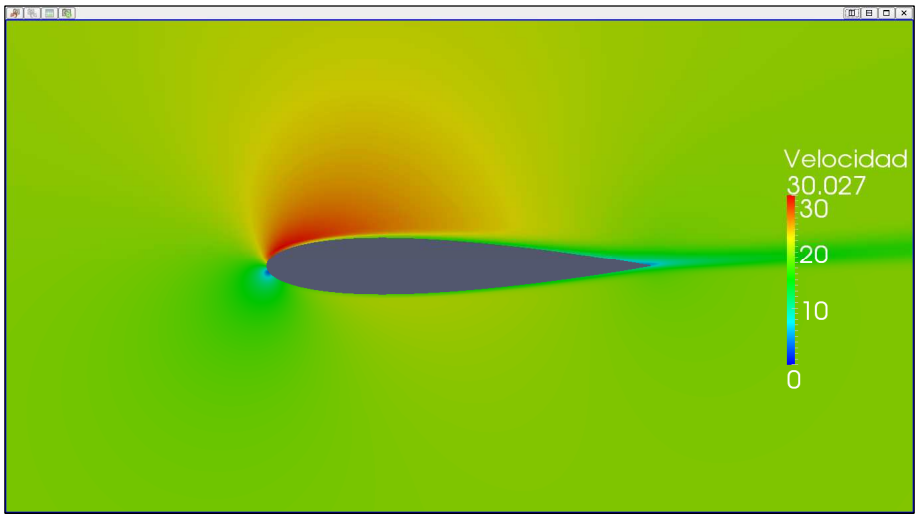
$C_L = 0{,}5139$

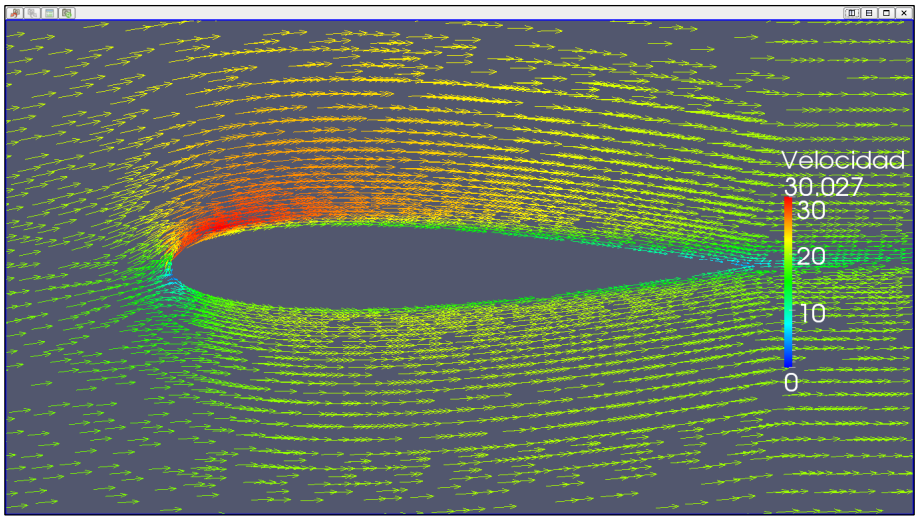
$C_D = 0{,}0186$






p
170.745
100
0
-100
-200
-300
-339.958


Velocidad
30.027
30
20
10
0


Velocidad
30.027
30
20
10
0

II.2.8. Resultados ángulo de ataque 6º

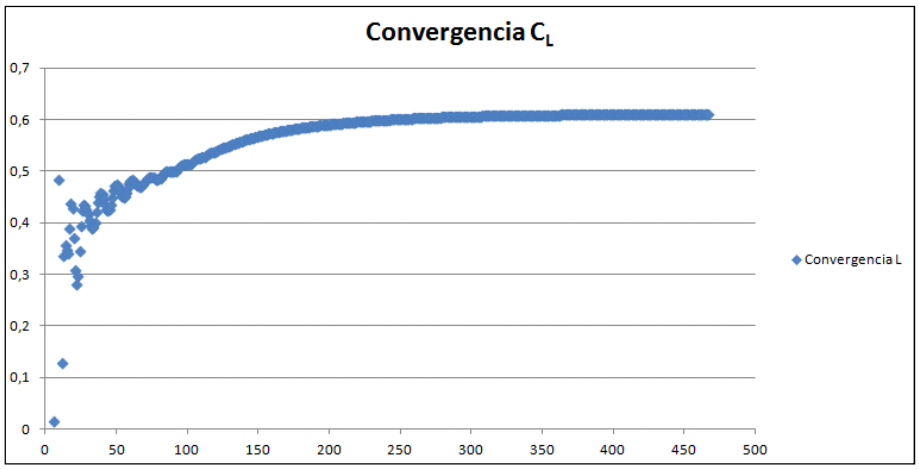
II.2.8.1. Caso 1 (Ángulo de ataque 6º)

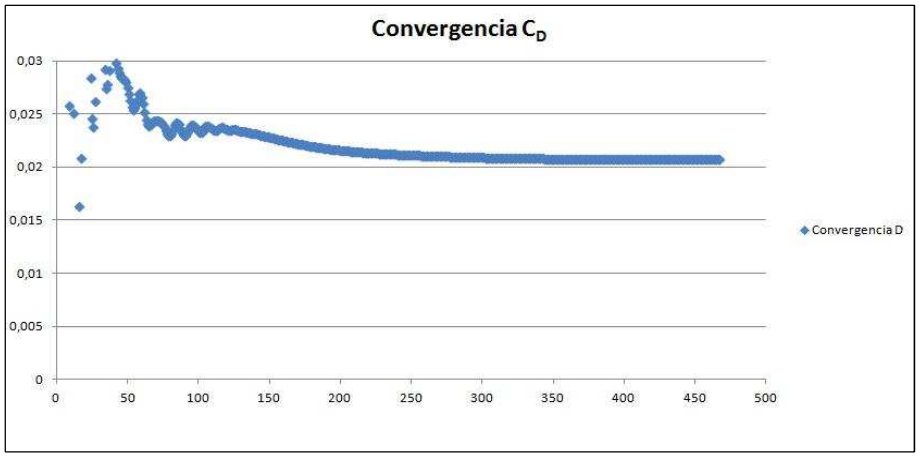
SIMPLE solution converged in 467 iterations.

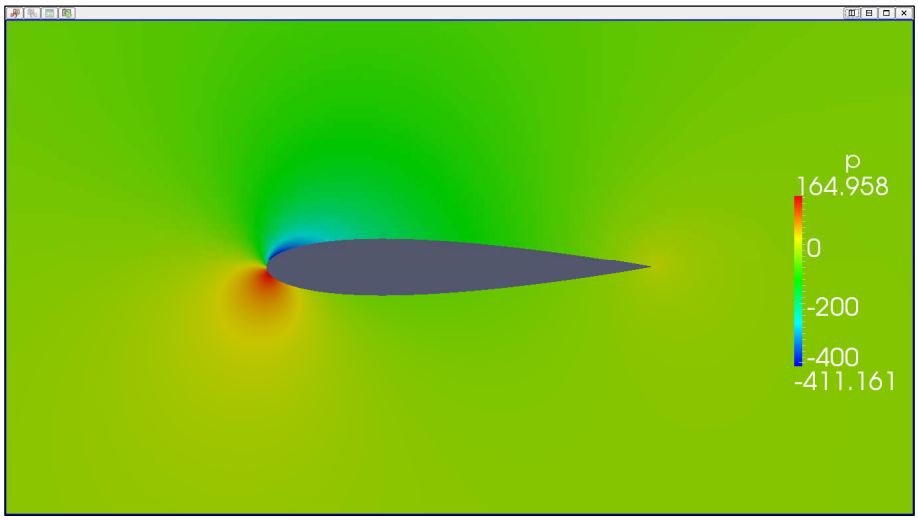
ForceCoeffs output:

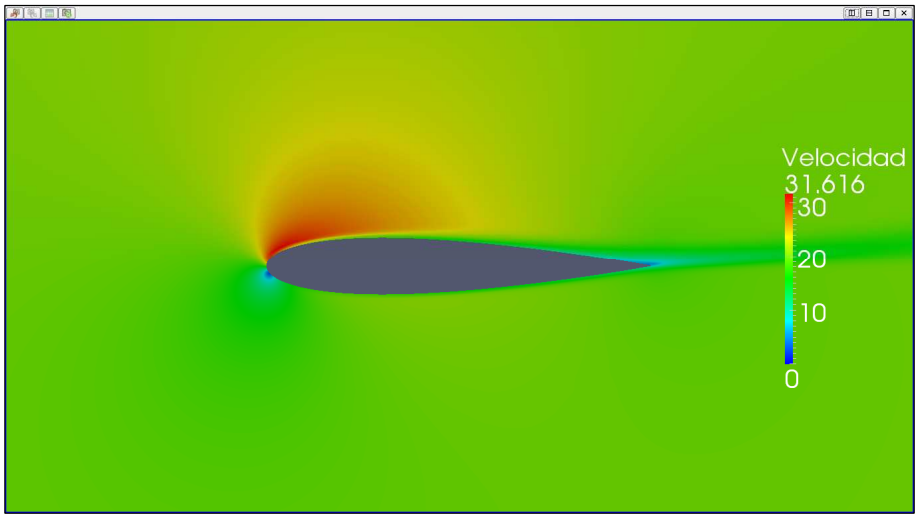
$C_L = 0{,}6106$

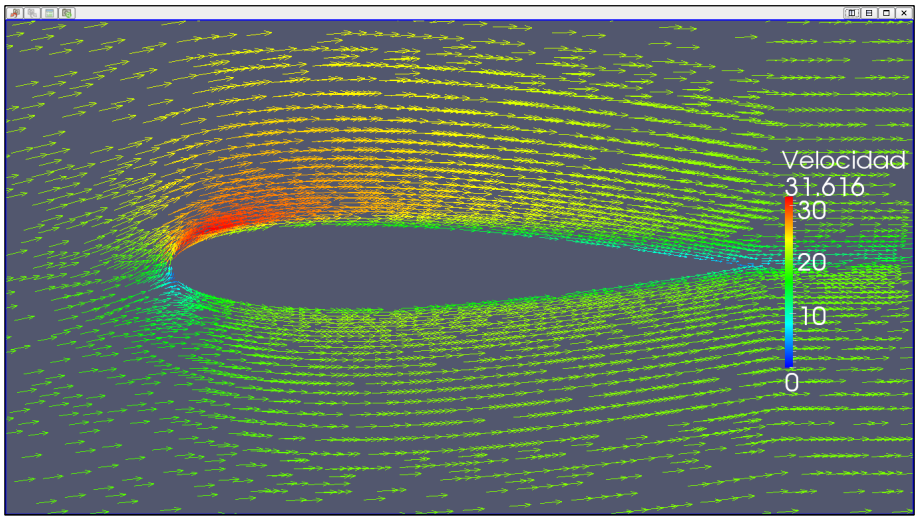
$C_D = 0{,}0207$






p
164.958
0
-200
-400
-411.161


Velocidad
31.616
30
20
10
0


Velocidad
31.616
30
20
10
0

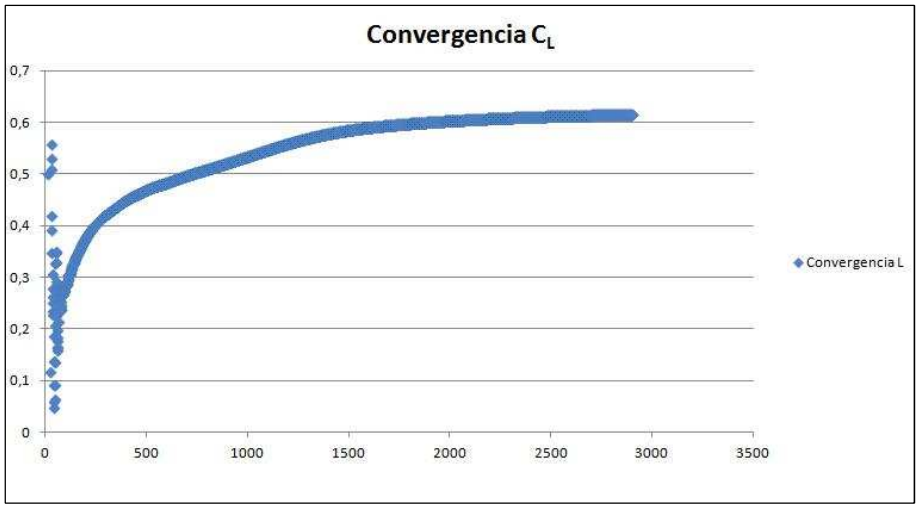
II.2.8.2. Caso 2 (Ángulo de ataque 6º)

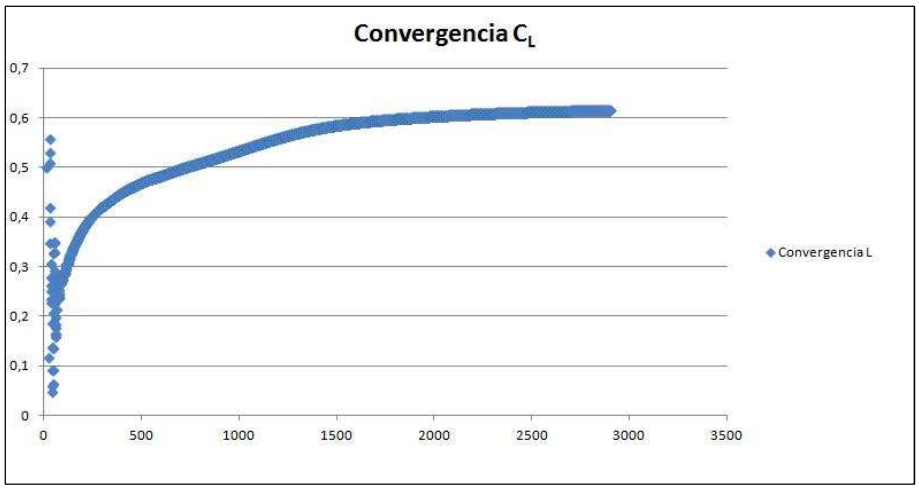
SIMPLE solution converged in 2903 iterations.

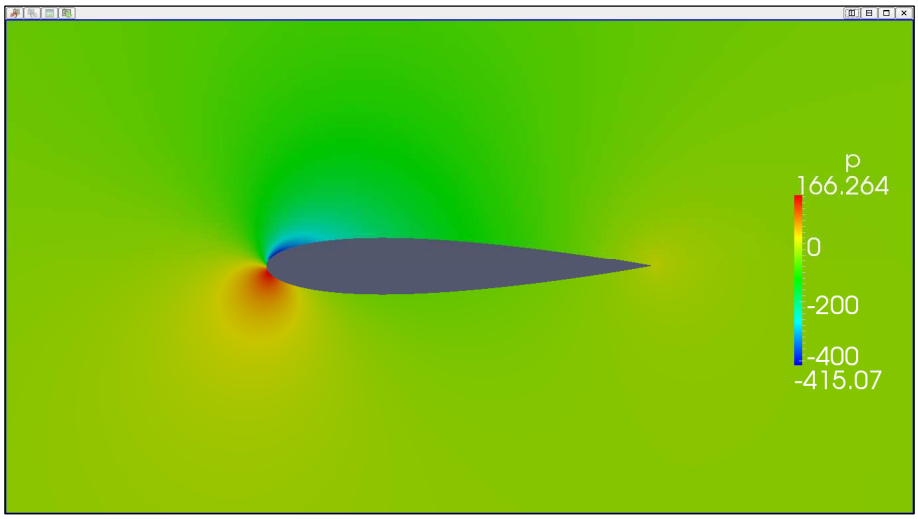
ForceCoeffs output:

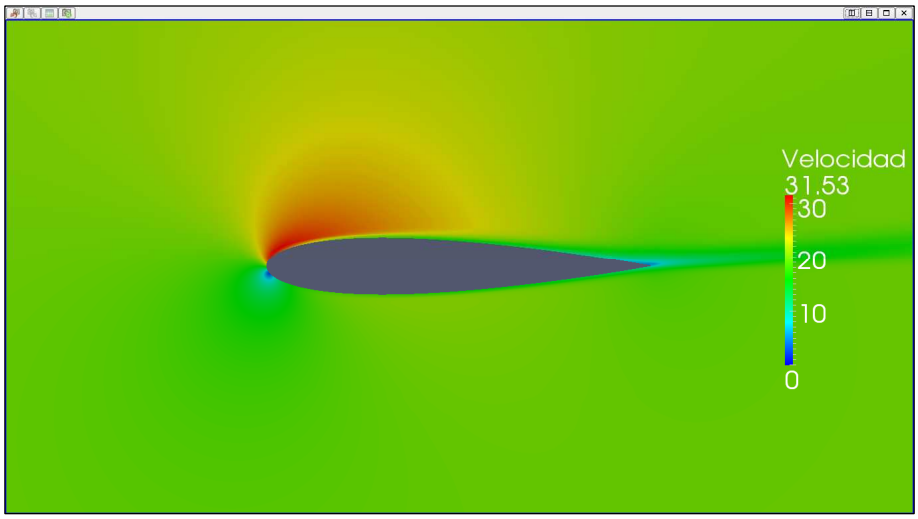
$C_L = 0{,}6152$

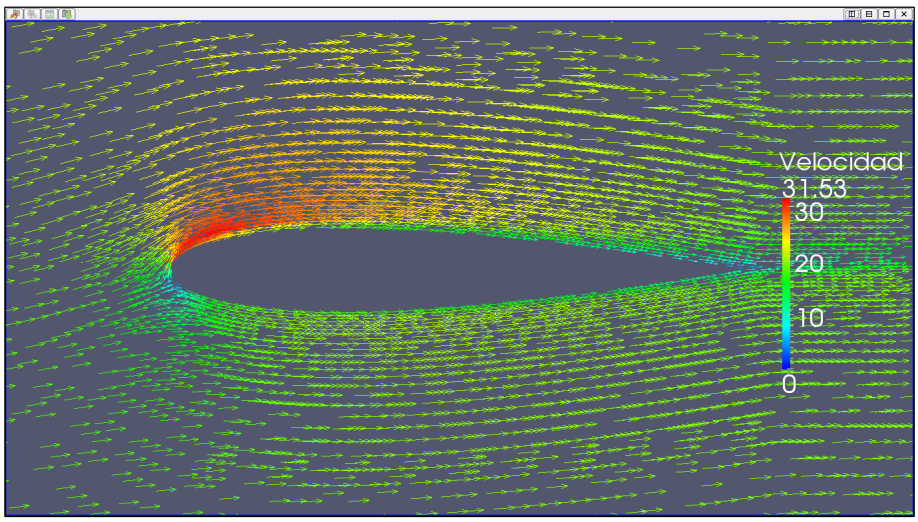
$C_D = 0{,}0194$






p
166.264
0
-200
-400
-415.07


Velocidad
31.53
30
20
10
0


Velocidad
31.53
30
20
10
0

II.2.8.3. Caso 3 (Ángulo de ataque 6º)

Time: 50000

Solving U_X: Initial residual $= 4{,}049 \cdot 10^{-07}$, Final residual $= 2{,}025 \cdot 10^{-08}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 2{,}926 \cdot 10^{-07}$, Final residual $= 1{,}881 \cdot 10^{-08}$, No Iterations $= 4$.

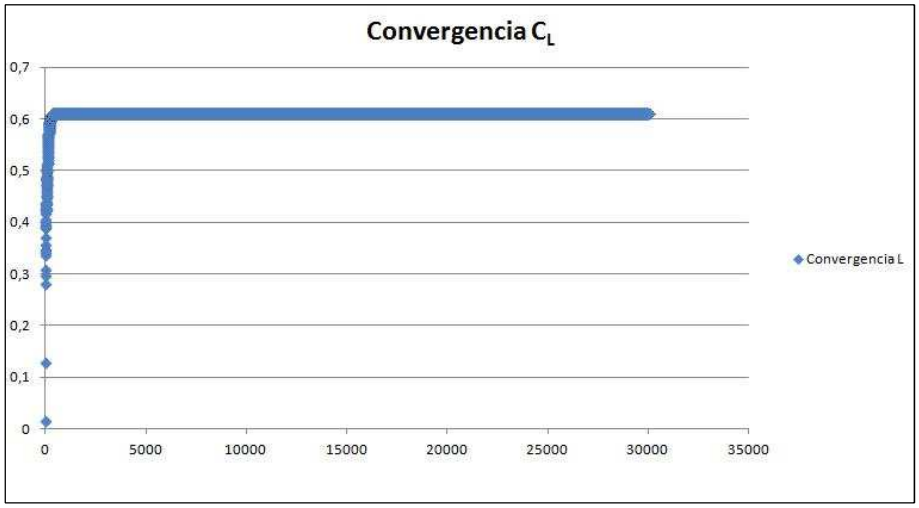
Solving p: Initial residual $= 1{,}108 \cdot 10^{-05}$, Final residual $= 1{,}022 \cdot 10^{-06}$, No Iterations $= 3$.

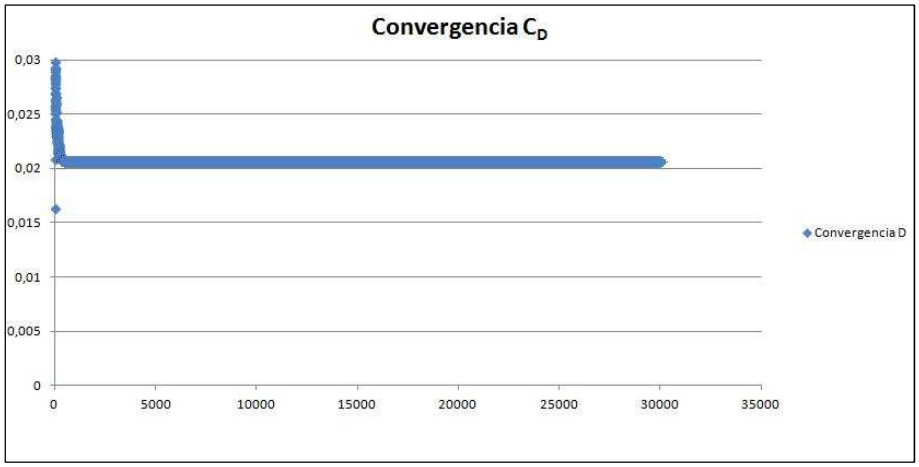
ExecutionTime: 5106,49 s. ClockTime: 5116 s.

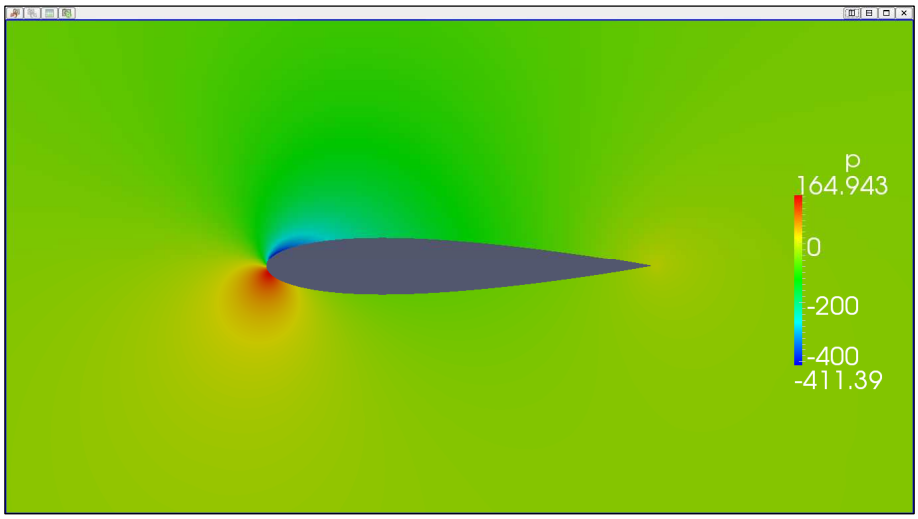
ForceCoeffs output:

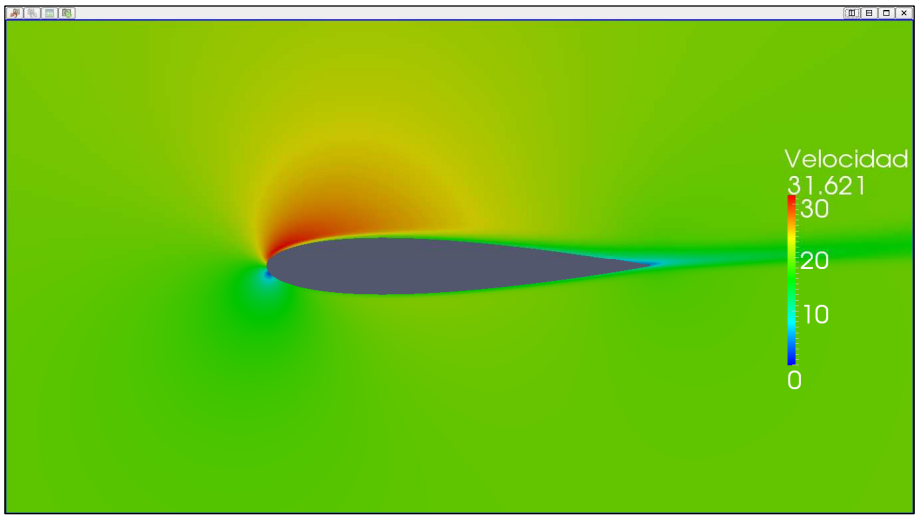
$C_L = 0{,}6110$

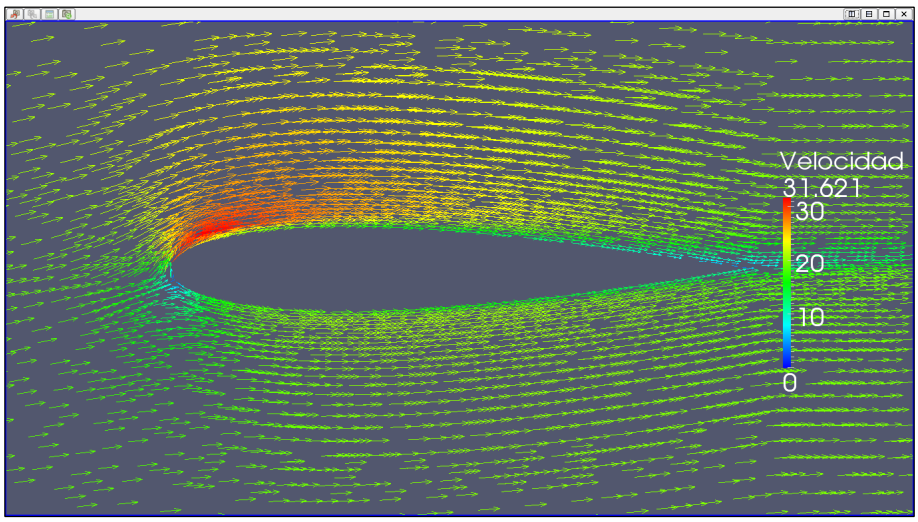
$C_D = 0{,}0207$






p
164.943
0
-200
-400
-411.39


Velocidad
31.621
30
20
10
0


Velocidad
31.621
30
20
10
0

II.2.9. Resultados ángulo de ataque 7º

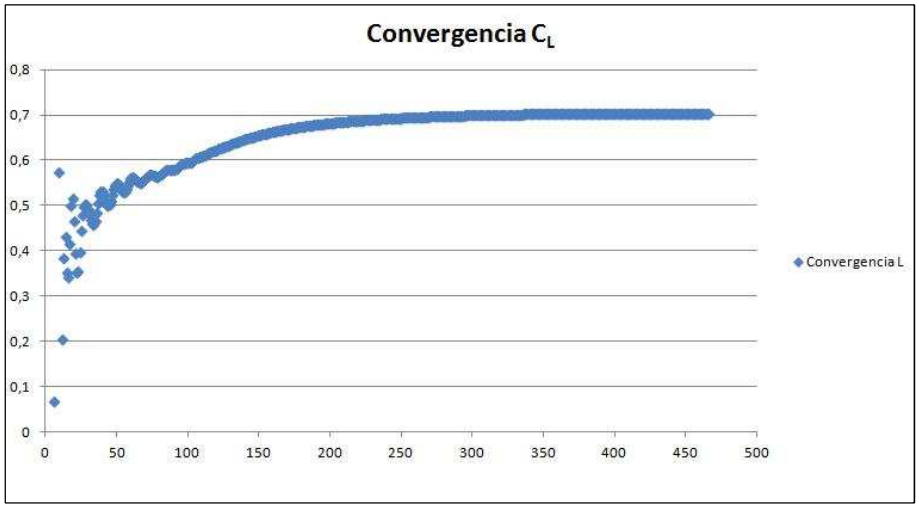
II.2.9.1. Caso 1 (Ángulo de ataque 7º)

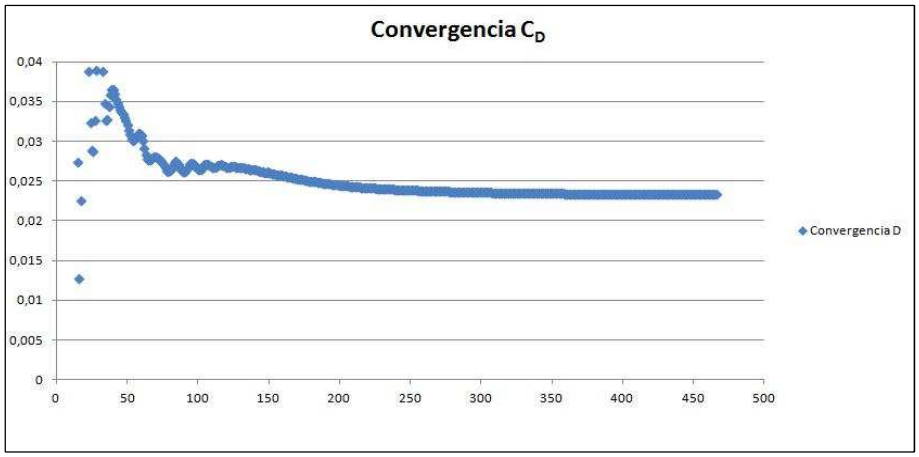
SIMPLE solution converged in 466 iterations.

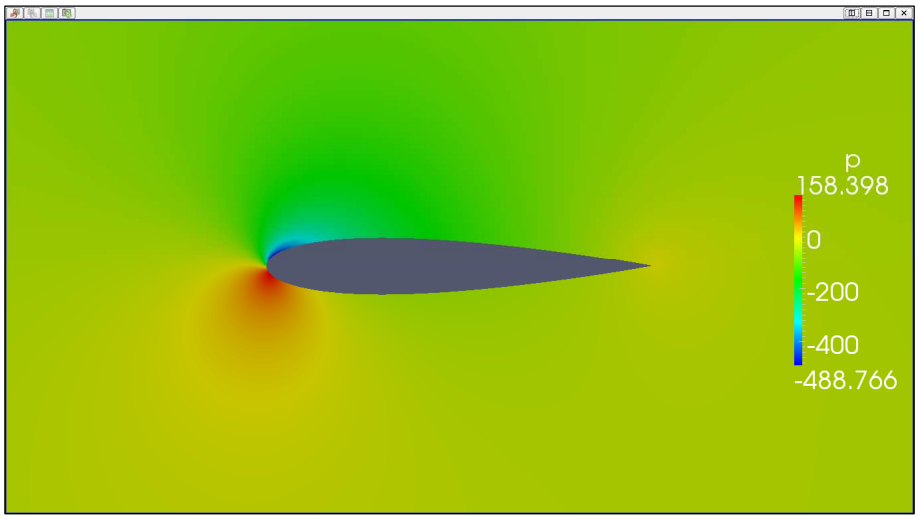
ForceCoeffs output:

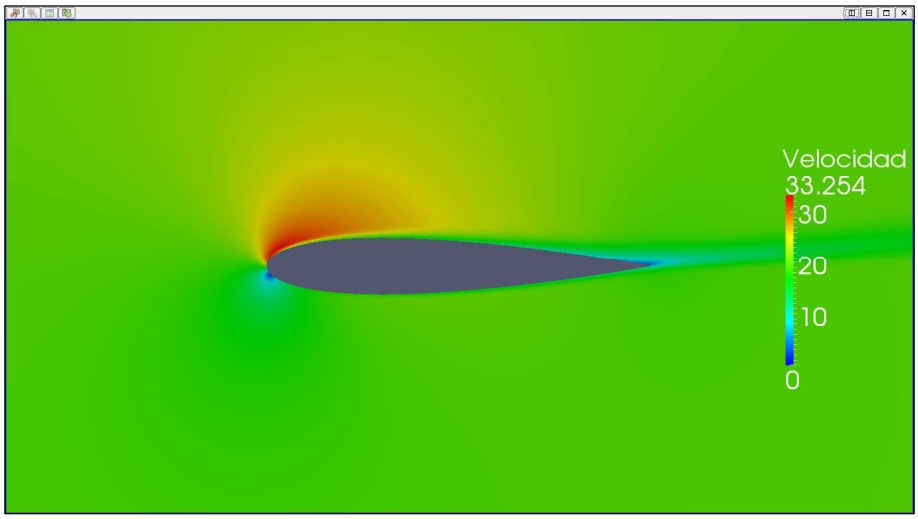
$C_L = 0{,}7040$

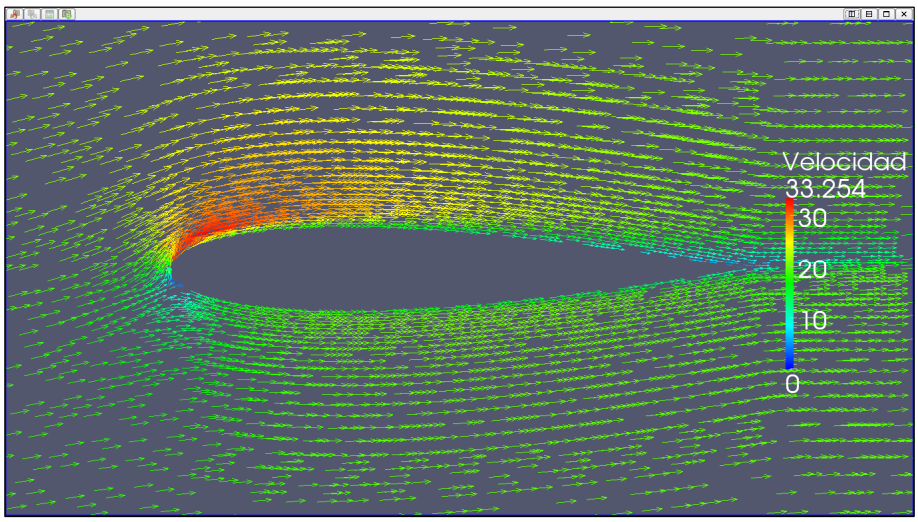
$C_D = 0{,}0233$






p
158.398
0
-200
-400
-488.766


Velocidad
33.254
30
20
10
0


Velocidad
33.254
30
20
10
0

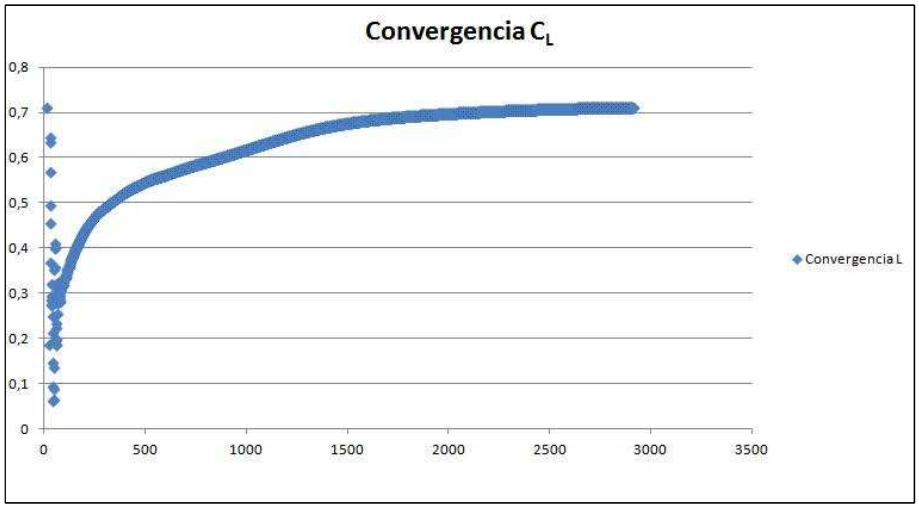
II.2.9.2. Caso 2 (Ángulo de ataque 7º)

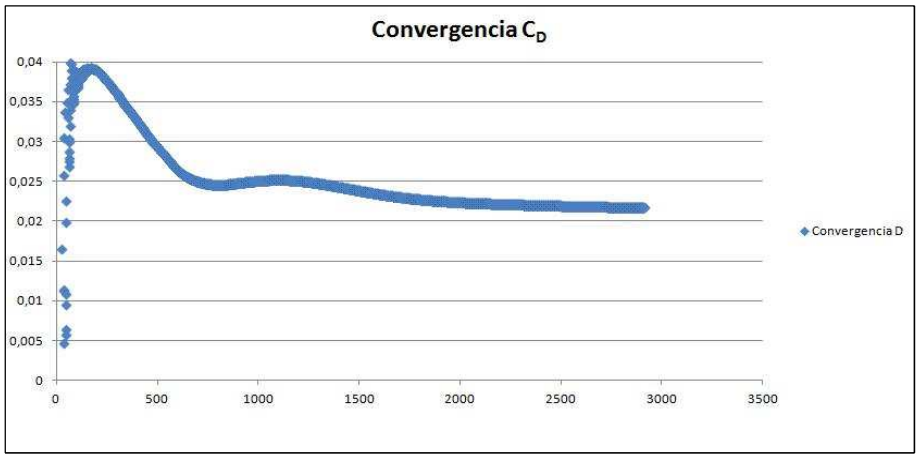
SIMPLE solution converged in 2914 iterations.

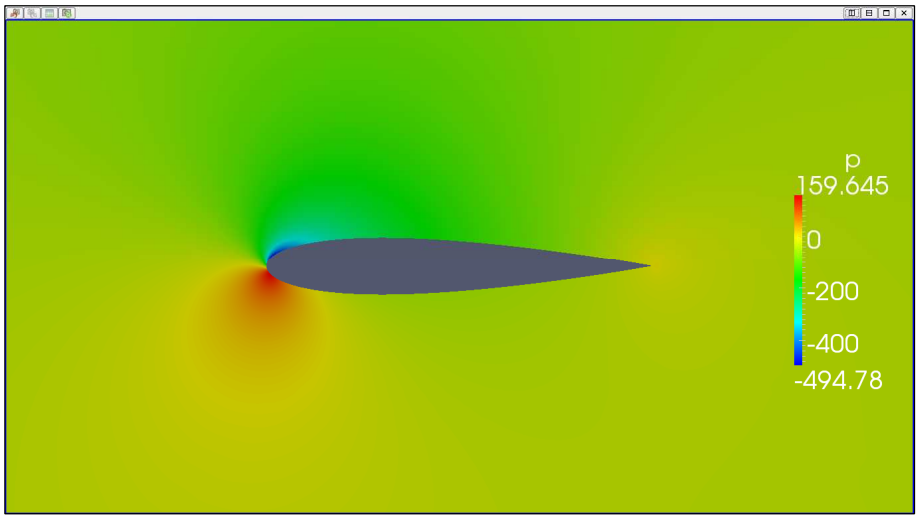
ForceCoeffs output:

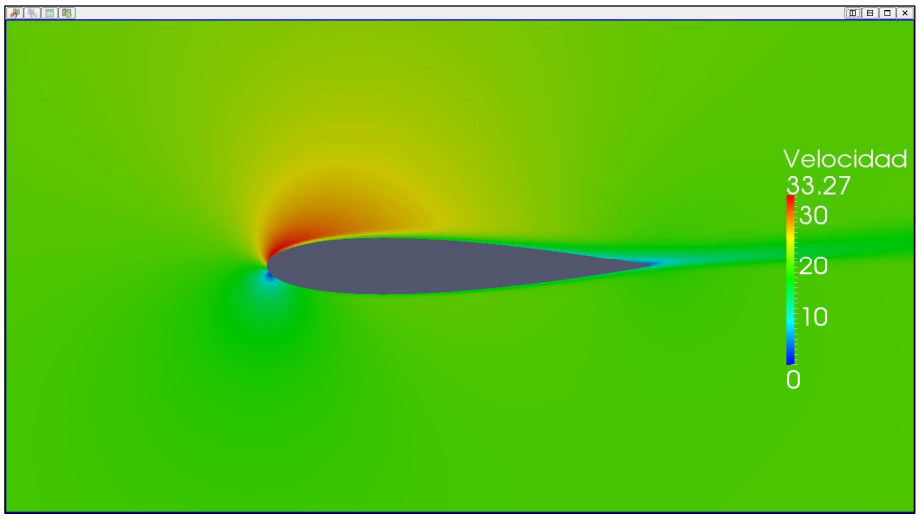
$C_L = 0{,}7117$

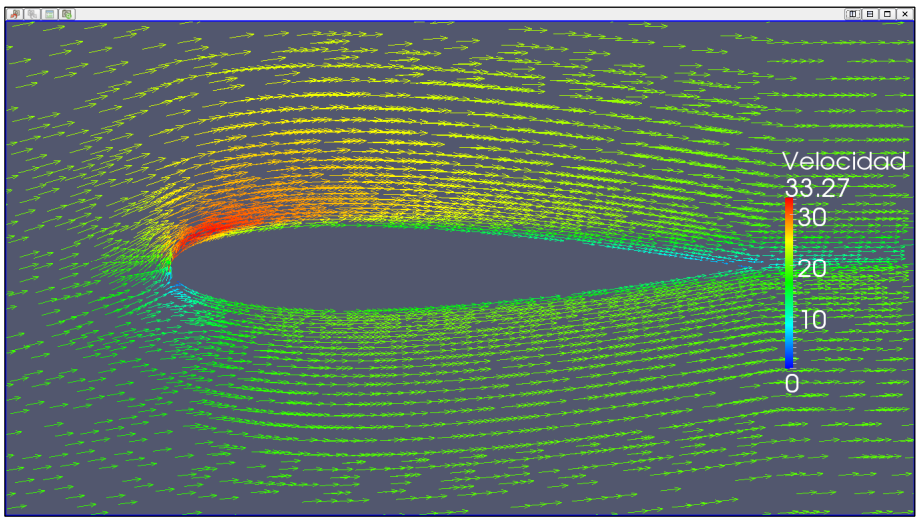
$C_D = 0{,}0217$






p
159.645
0
-200
-400
-494.78


Velocidad
33.27
30
20
10
0


Velocidad
33.27
30
20
10
0

II.2.9.3. Caso 3 (Ángulo de ataque 7º)

Time: 50000

Solving U_X: Initial residual $= 4{,}014 \cdot 10^{-07}$, Final residual $= 1{,}834 \cdot 10^{-08}$, No Iterations $= 6$.

Solving U_Y: Initial residual $= 3{,}251 \cdot 10^{-07}$, Final residual $= 2{,}989 \cdot 10^{-08}$, No Iterations $= 4$.

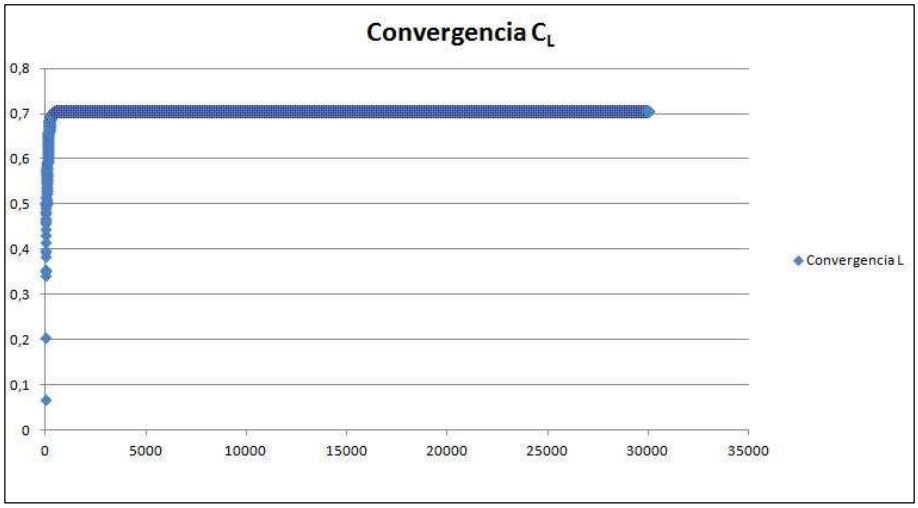
Solving p: Initial residual $= 1{,}890 \cdot 10^{-05}$, Final residual $= 1{,}797 \cdot 10^{-06}$, No Iterations $= 2$.

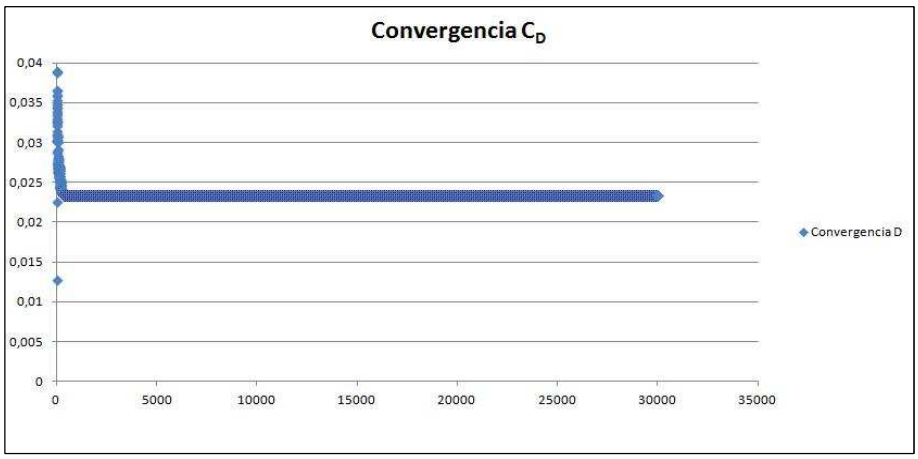
ExecutionTime: 5093,03 s. ClockTime: 5101 s.

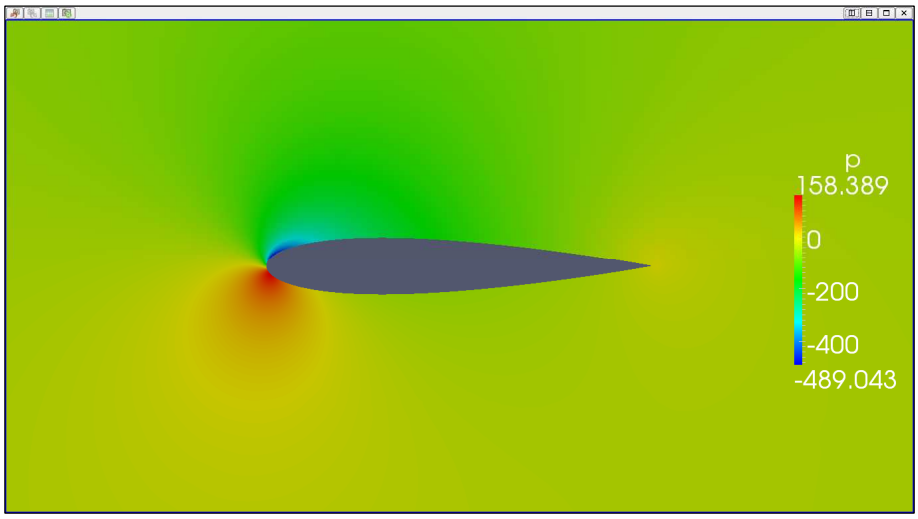
ForceCoeffs output:

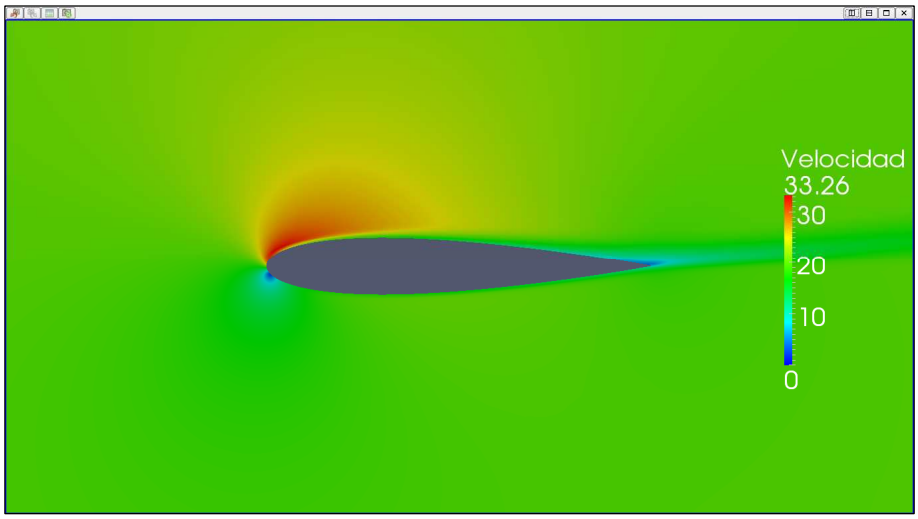
$C_L = 0{,}7044$

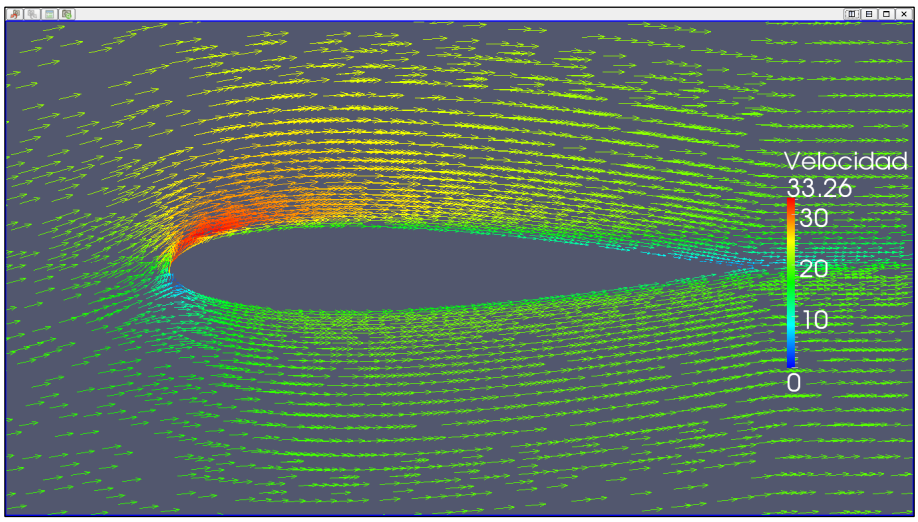
$C_D = 0{,}0233$






p
158.389
0
-200
-400
-489.043


Velocidad
33.26
30
20
10
0


Velocidad
33.26
30
20
10
0

II.2.10. Resultados ángulo de ataque 8º

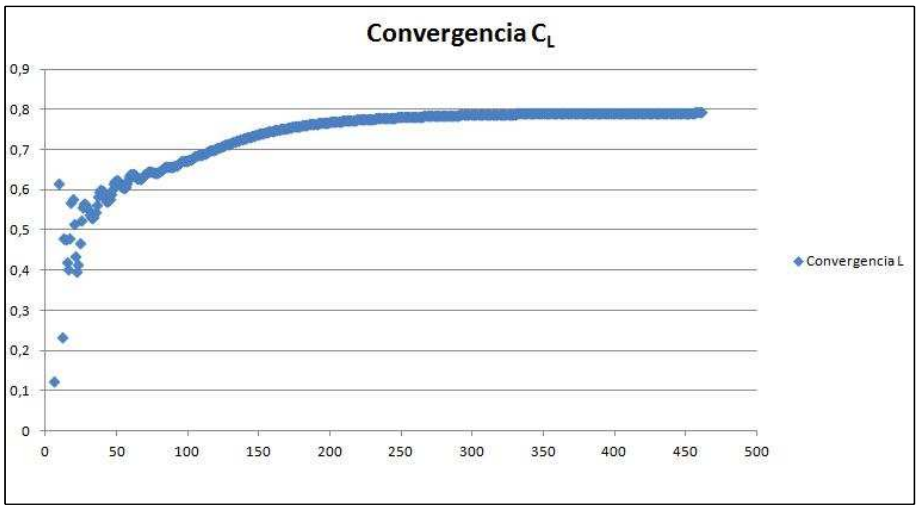
II.2.10.1. Caso 1 (Ángulo de ataque 8º)

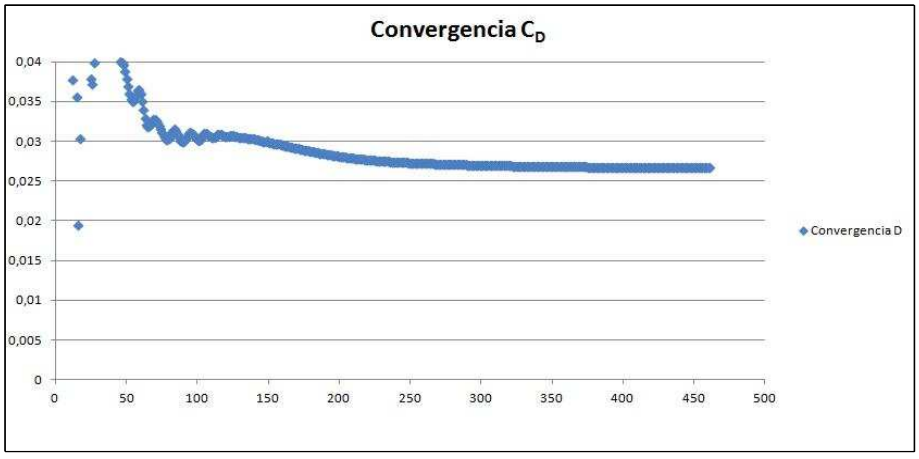
SIMPLE solution converged in 461 iterations.

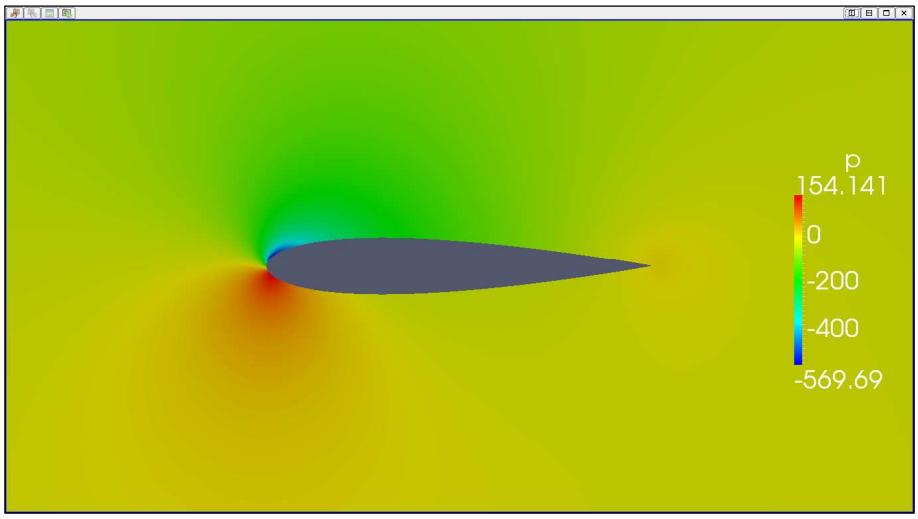
ForceCoeffs output:

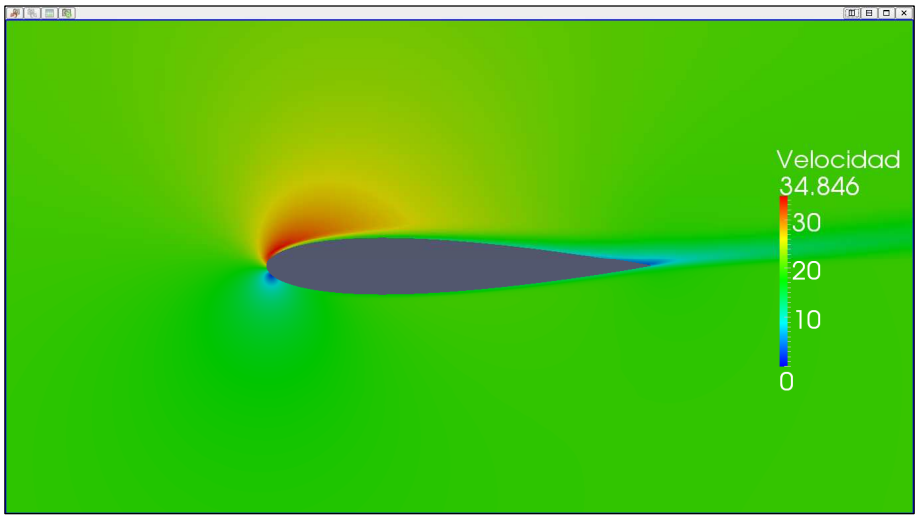
$C_L = 0{,}7921$

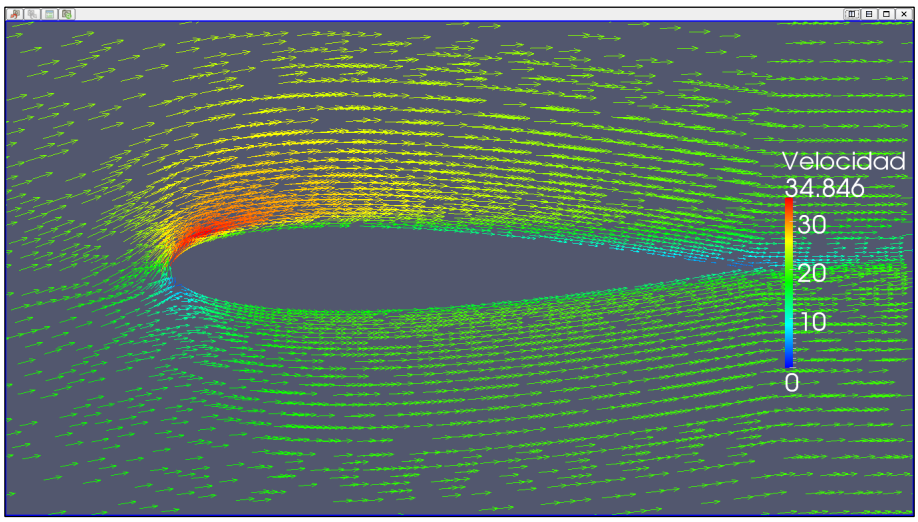
$C_D = 0{,}0267$






p
154.141
0
-200
-400
-569.69


Velocidad
34.846
30
20
10
0


Velocidad
34.846
30
20
10
0

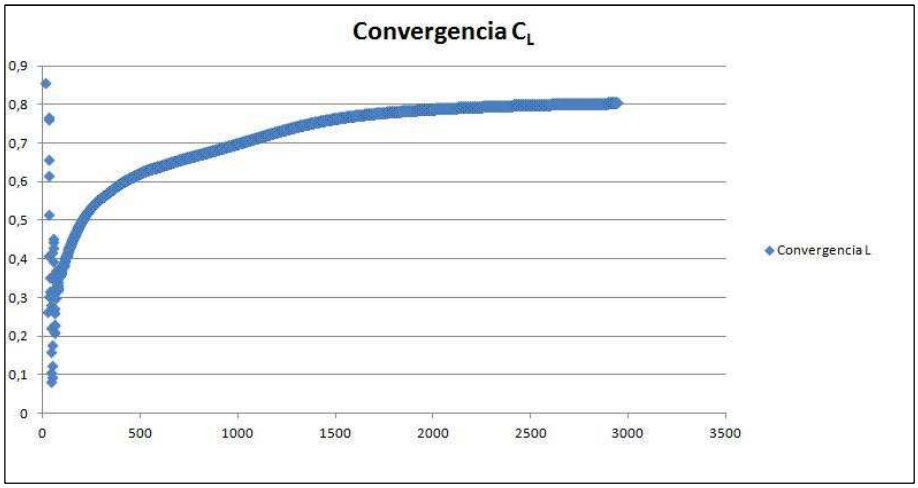
II.2.10.2. Caso 2 (Ángulo de ataque 8º)

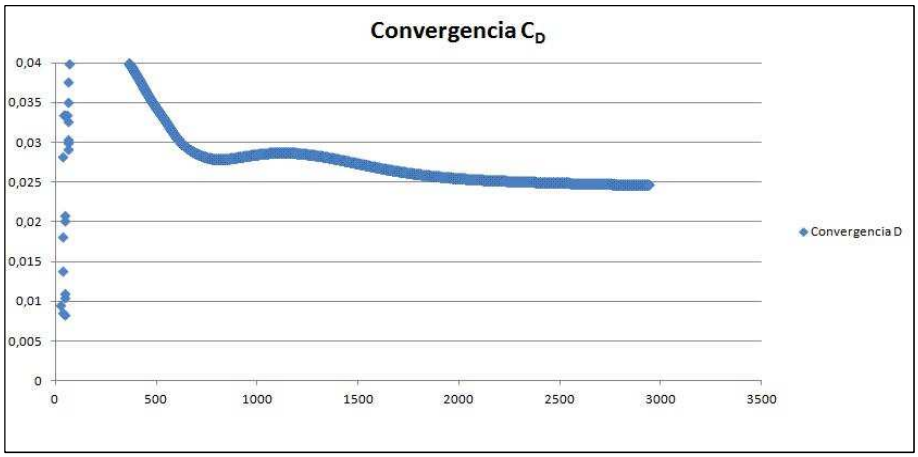
SIMPLE solution converged in 2943 iterations.

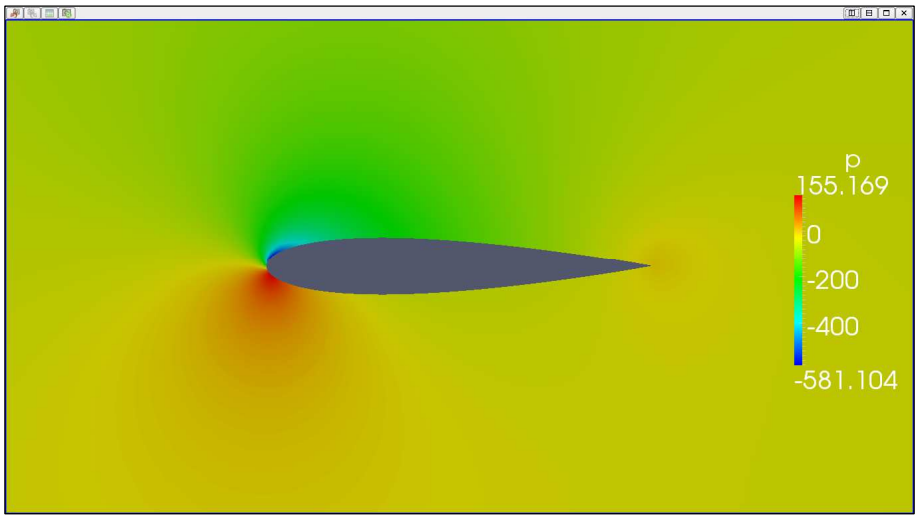
ForceCoeffs output:

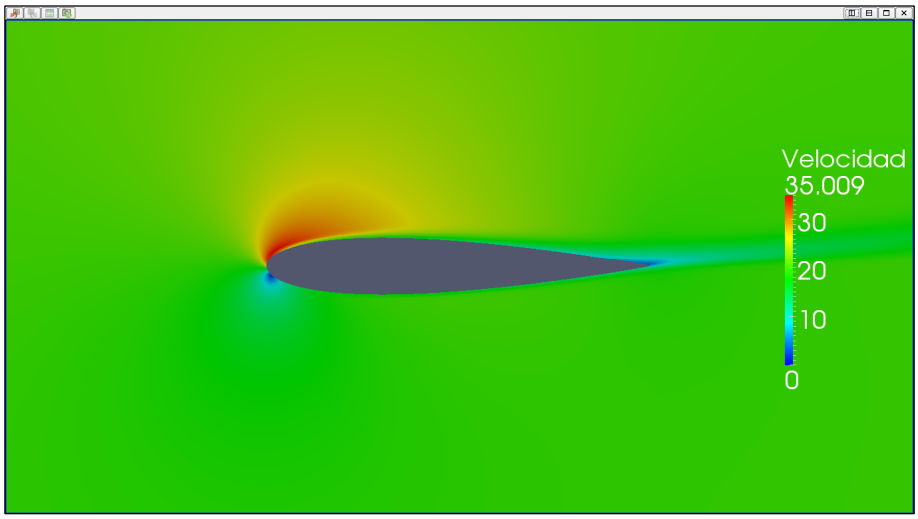
$C_L = 0{,}8043$

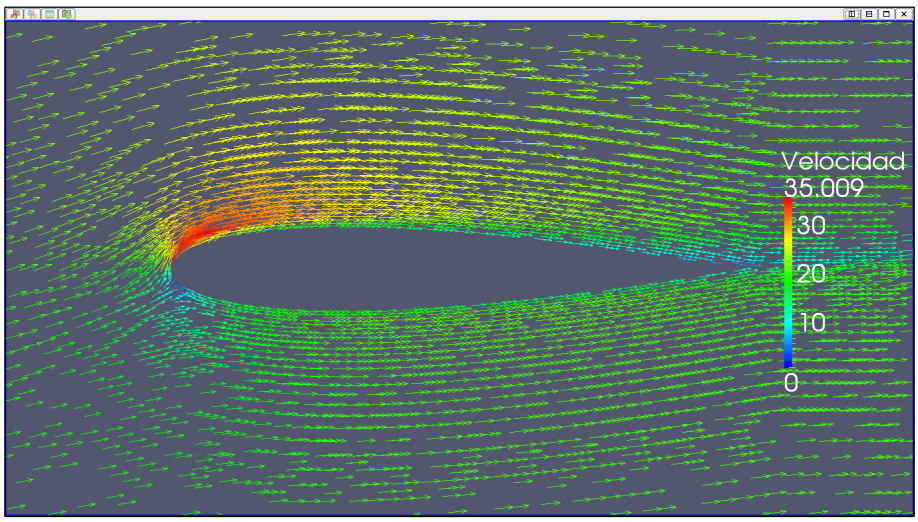
$C_D = 0{,}0247$






p
155.169
0
-200
-400
-581.104


Velocidad
35.009
30
20
10
0


Velocidad
35.009
30
20
10
0

II.2.10.3. Caso 3 (Ángulo de ataque 8º)

Time: 50000

Solving U_X: Initial residual $= 7{,}149 \cdot 10^{-07}$, Final residual $= 5{,}223 \cdot 10^{-08}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 6{,}018 \cdot 10^{-07}$, Final residual $= 3{,}219 \cdot 10^{-08}$, No Iterations $= 2$.

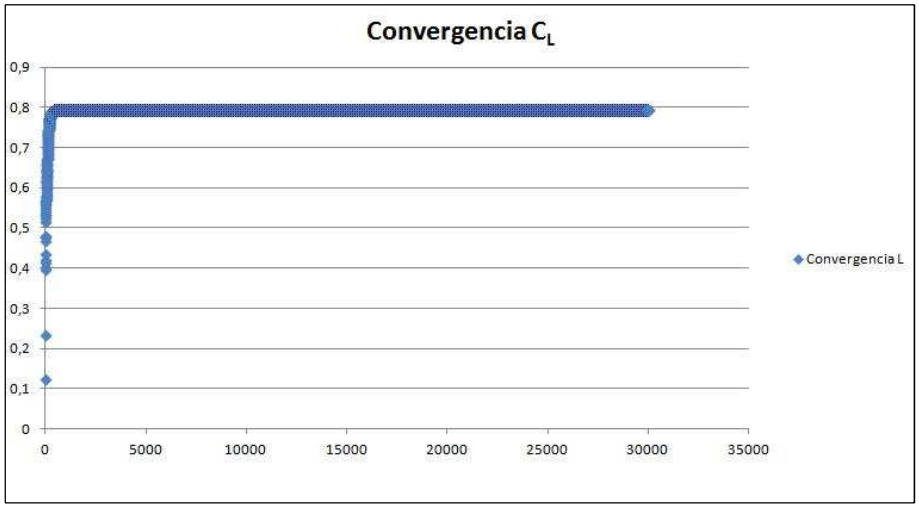
Solving p: Initial residual $= 3{,}368 \cdot 10^{-05}$, Final residual $= 2{,}816 \cdot 10^{-06}$, No Iterations $= 2$.

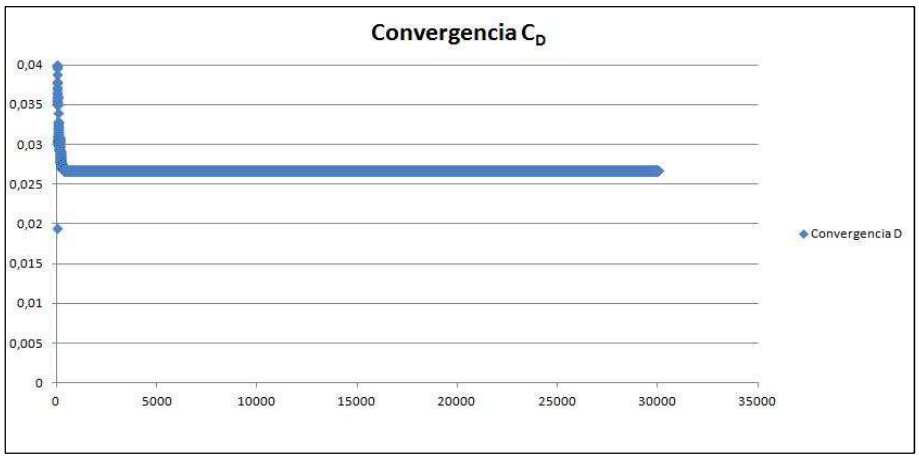
ExecutionTime: 4968,78 s. ClockTime: 4976 s.

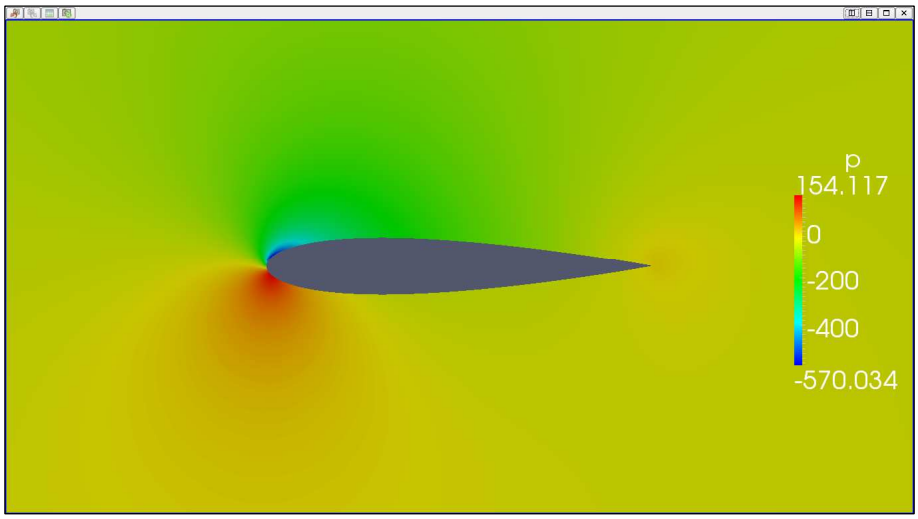
ForceCoeffs output:

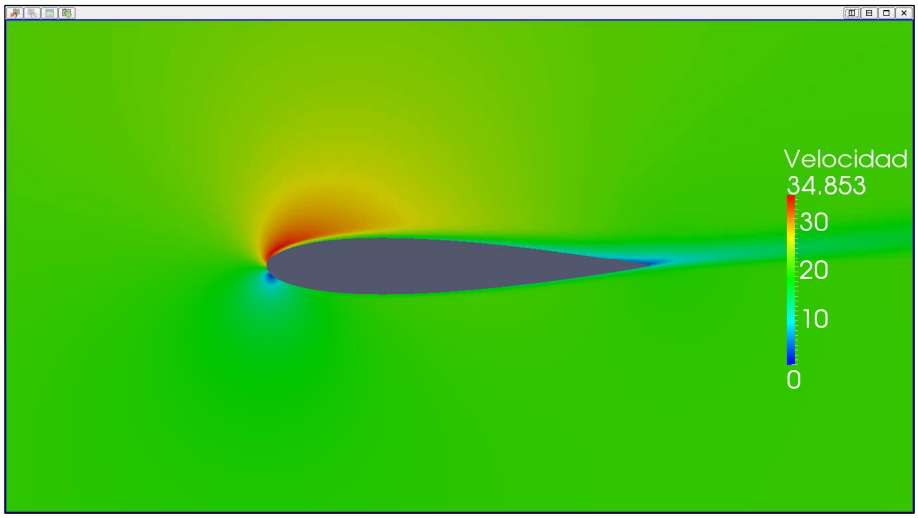
$C_L = 0{,}7926$

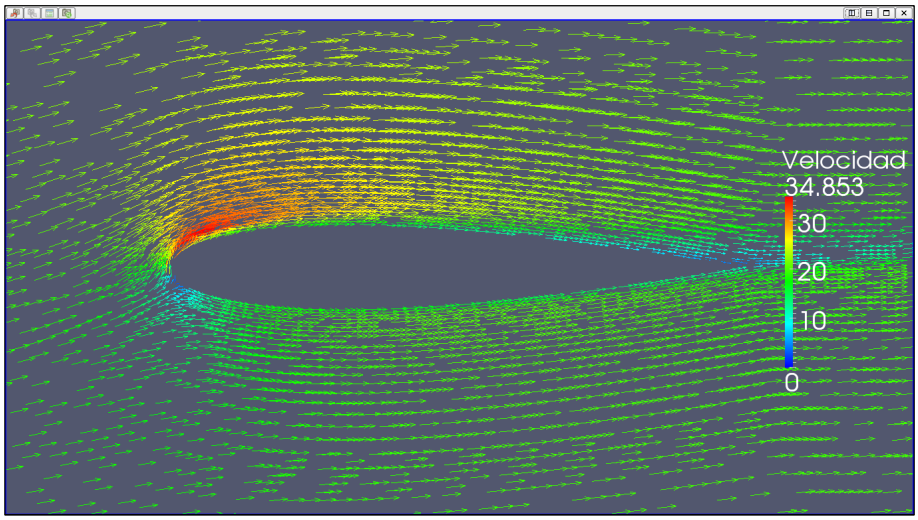
$C_D = 0{,}0267$






p
154.117
0
-200
-400
-570.034


Velocidad
34.853
30
20
10
0


Velocidad
34.853
30
20
10
0

II.2.11. Resultados ángulo de ataque 9º

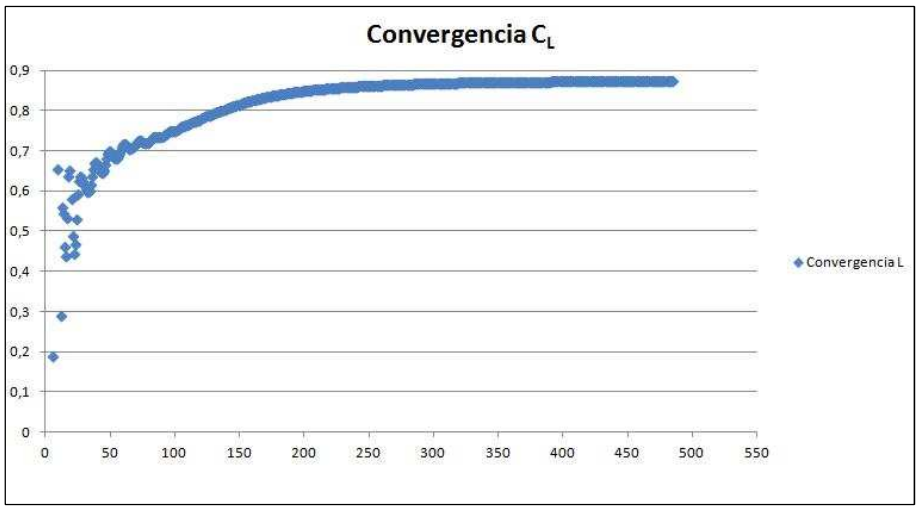
II.2.11.1. Caso 1 (Ángulo de ataque 9º)

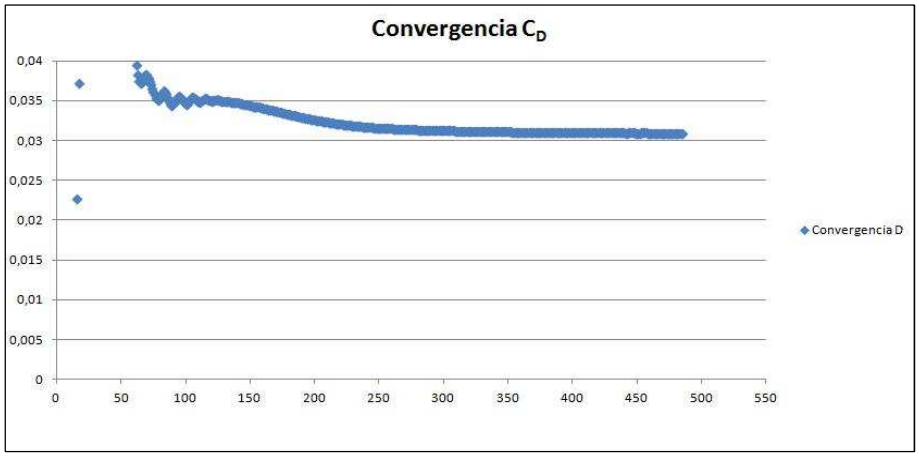
SIMPLE solution converged in 485 iterations.

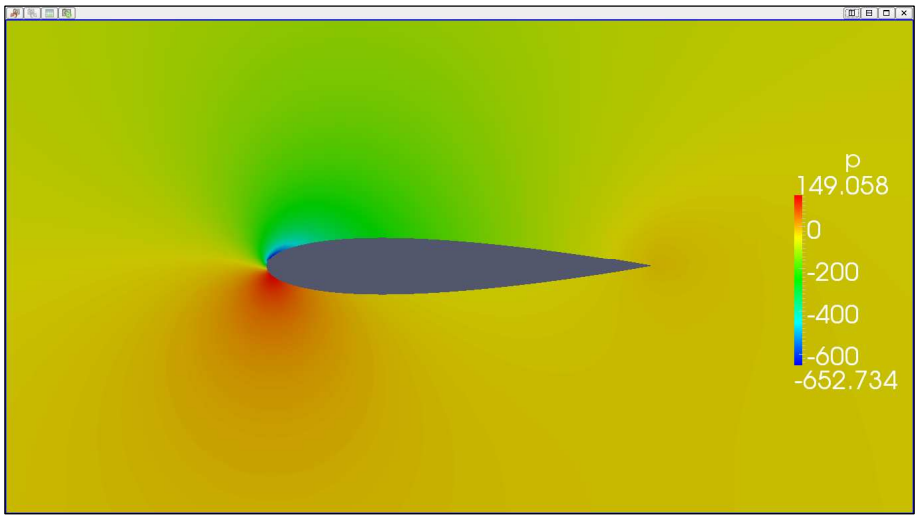
ForceCoeffs output:

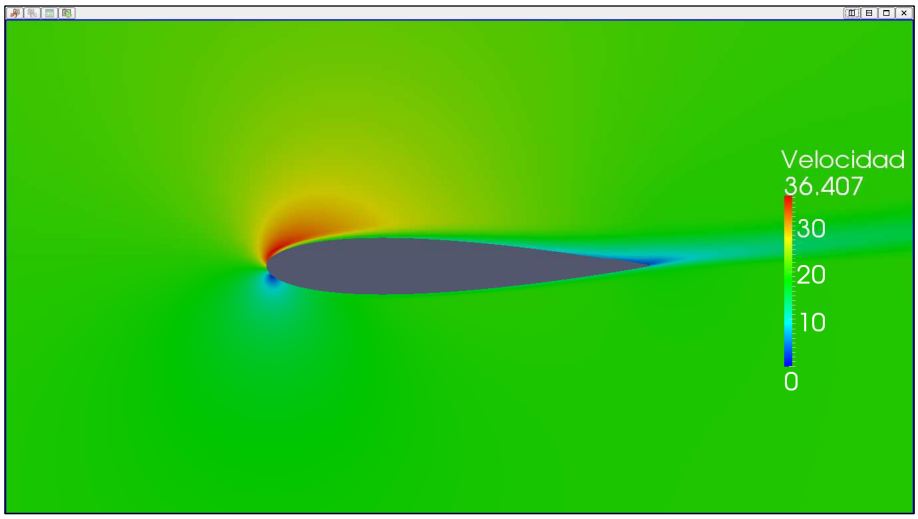
$C_L = 0{,}8732$

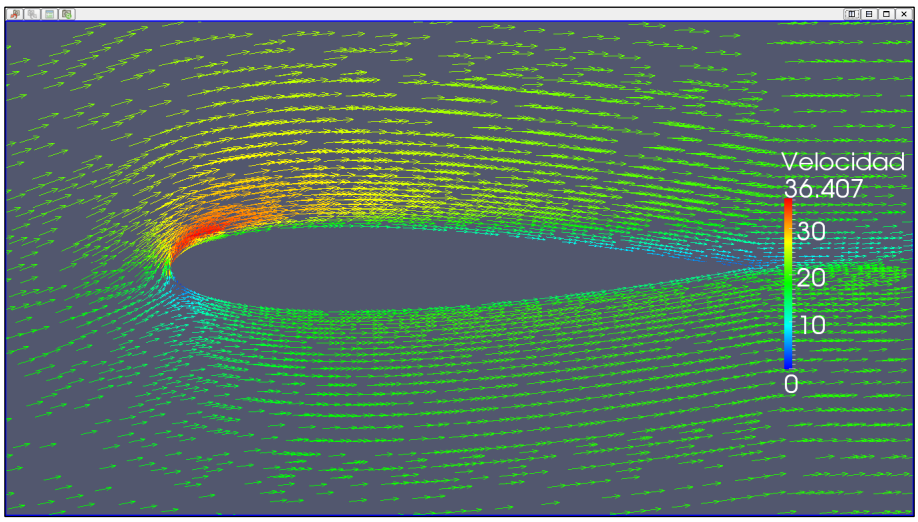
$C_D = 0{,}0309$






p
149.058
0
-200
-400
-600
-652.734


Velocidad
36.407
30
20
10
0


Velocidad
36.407
30
20
10
0

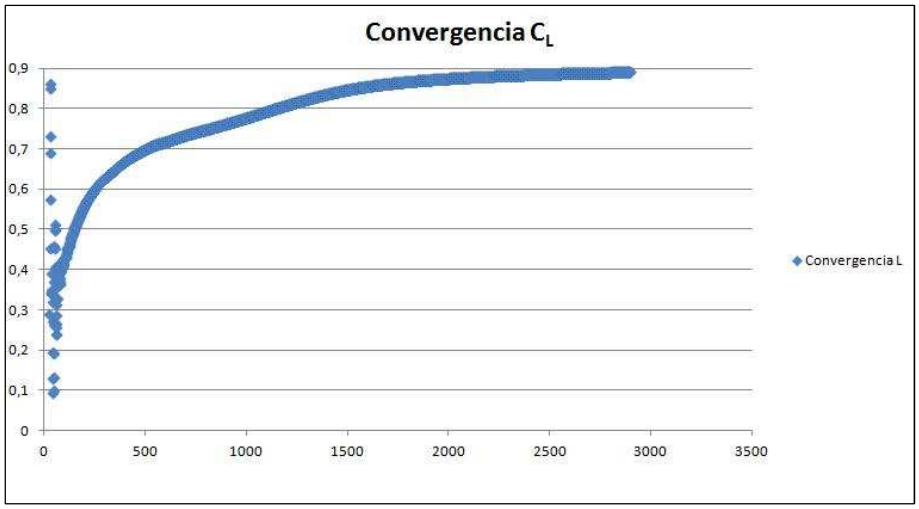
II.2.11.2. Caso 2 (Ángulo de ataque 9º)

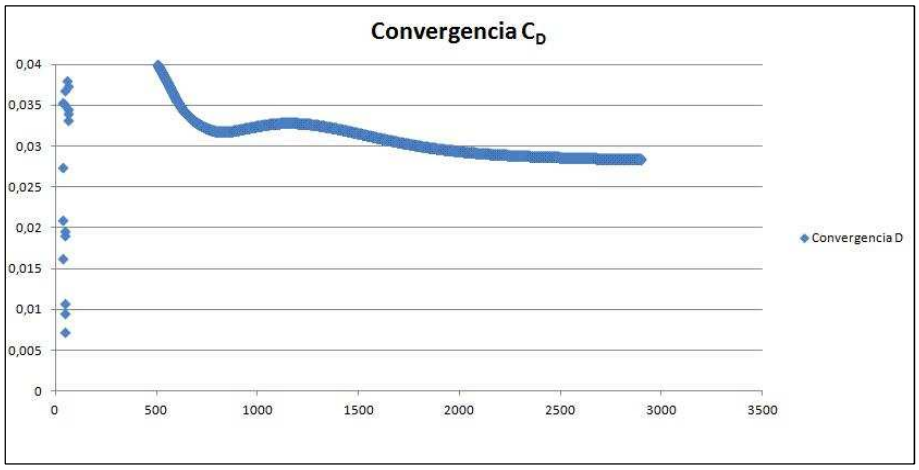
SIMPLE solution converged in 2899 iterations.

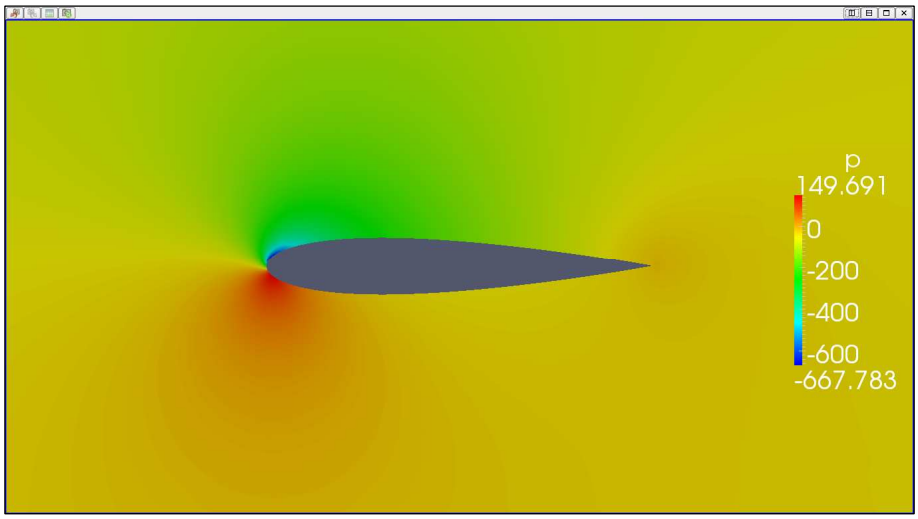
ForceCoeffs output:

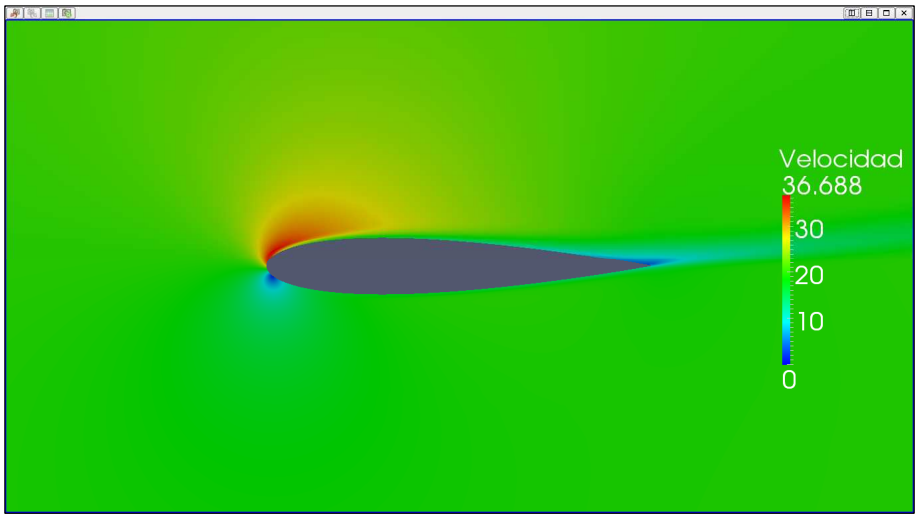
$C_L = 0{,}8908$

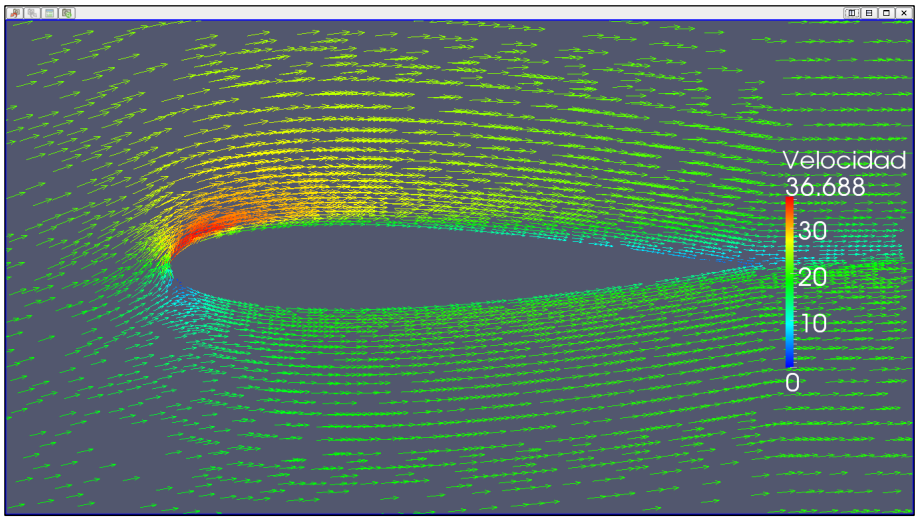
$C_D = 0{,}0284$






p
149.691
0
-200
-400
-600
-667.783


Velocidad
36.688
30
20
10
0


Velocidad
36.688
30
20
10
0

II.2.11.3. Caso 3 (Ángulo de ataque 9º)

Time: 50000

Solving U_X: Initial residual $= 2{,}219 \cdot 10^{-07}$, Final residual $= 1{,}086 \cdot 10^{-08}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 1{,}716 \cdot 10^{-07}$, Final residual $= 1{,}037 \cdot 10^{-08}$, No Iterations $= 4$.

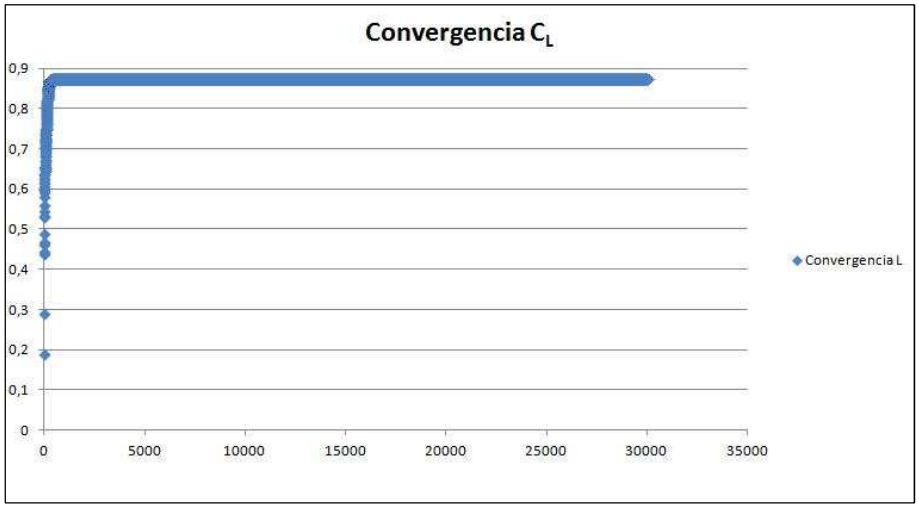
Solving p: Initial residual $= 8{,}334 \cdot 10^{-06}$, Final residual $= 8{,}332 \cdot 10^{-07}$, No Iterations $= 3$.

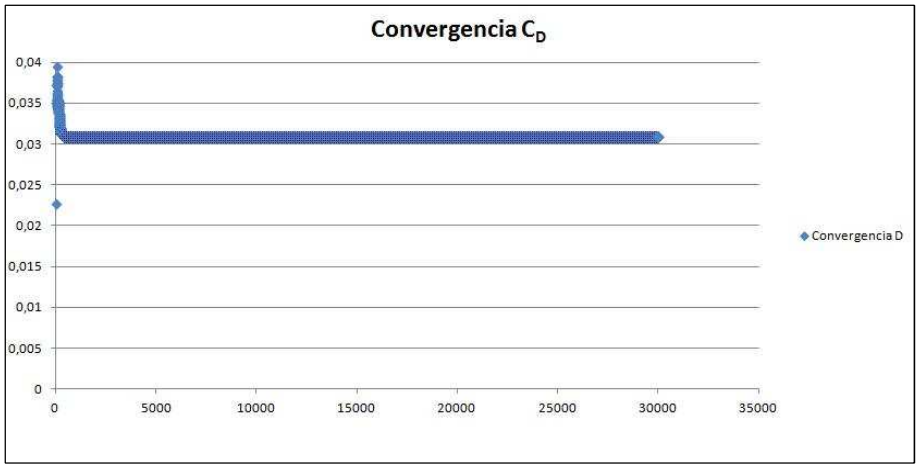
ExecutionTime: 5207,85 s. ClockTime: 5217 s.

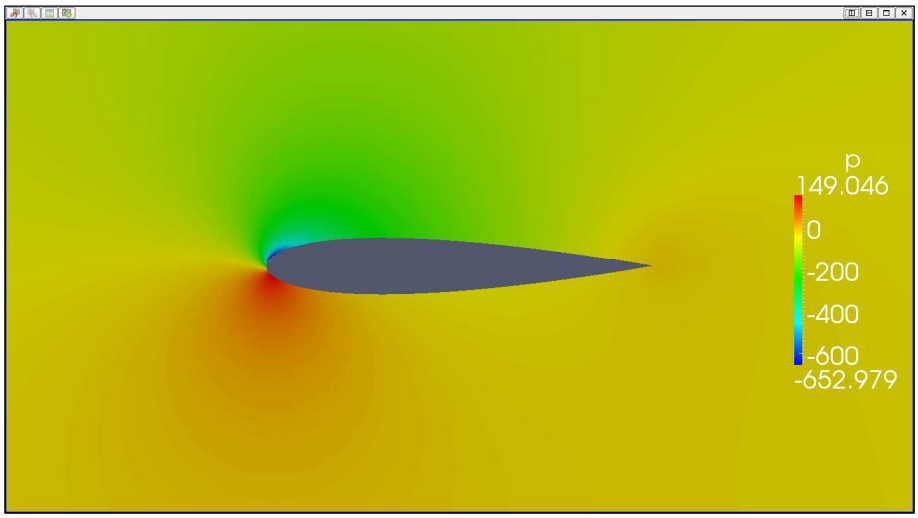
ForceCoeffs output:

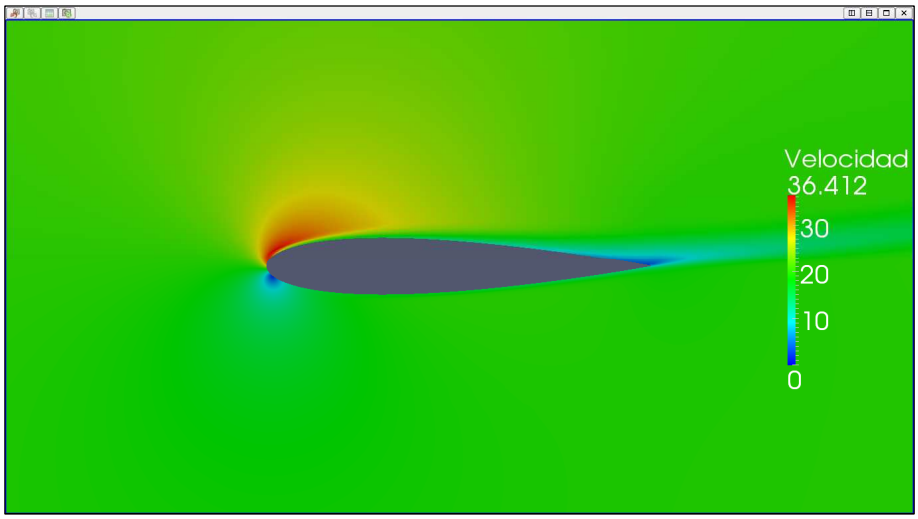
$C_L = 0{,}8735$

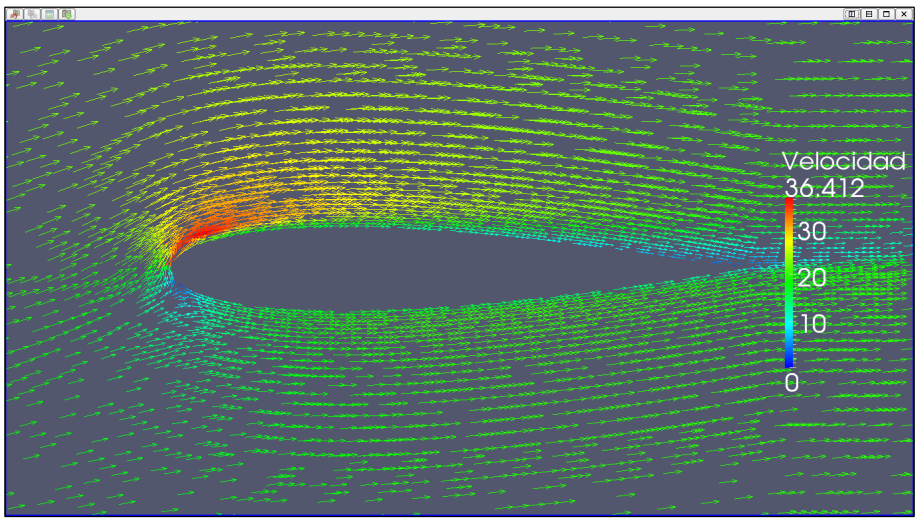
$C_D = 0{,}0309$






p
149.046
0
-200
-400
-600
-652.979


Velocidad
36.412
30
20
10
0


Velocidad
36.412
30
20
10
0

II.2.12. Resultados ángulo de ataque 10º

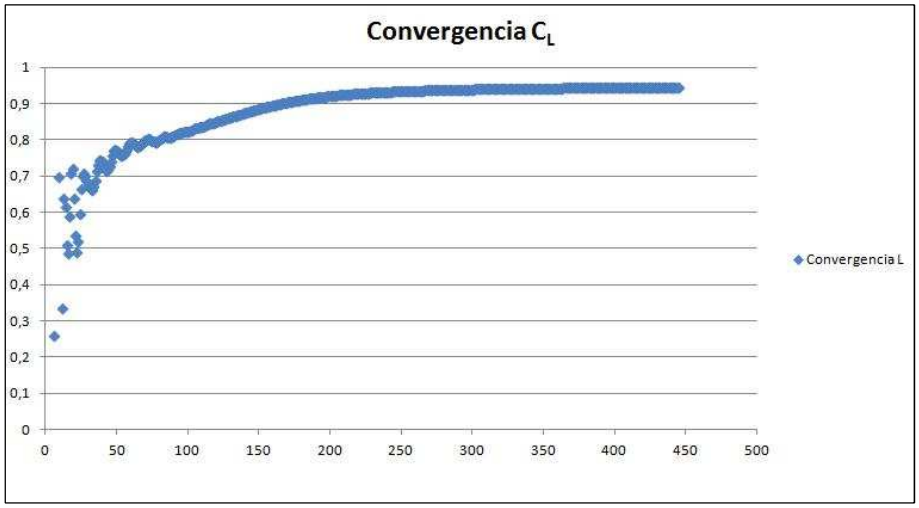
II.2.12.1. Caso 1 (Ángulo de ataque 10º)

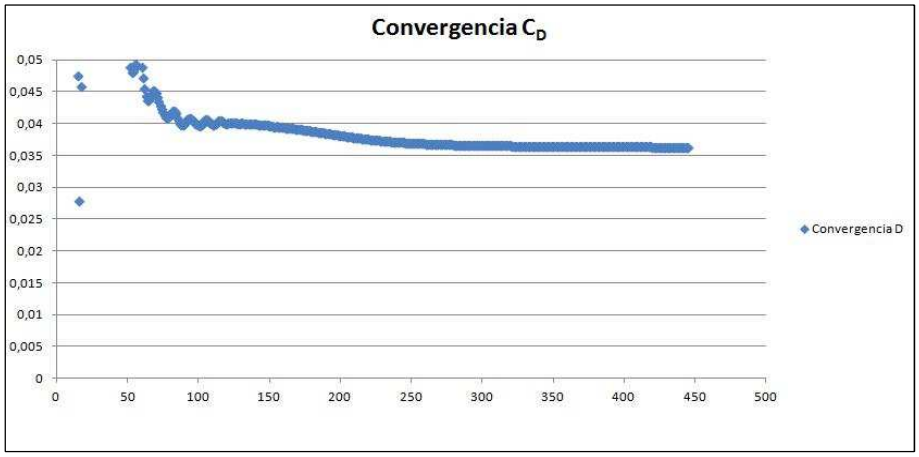
SIMPLE solution converged in 445 iterations.

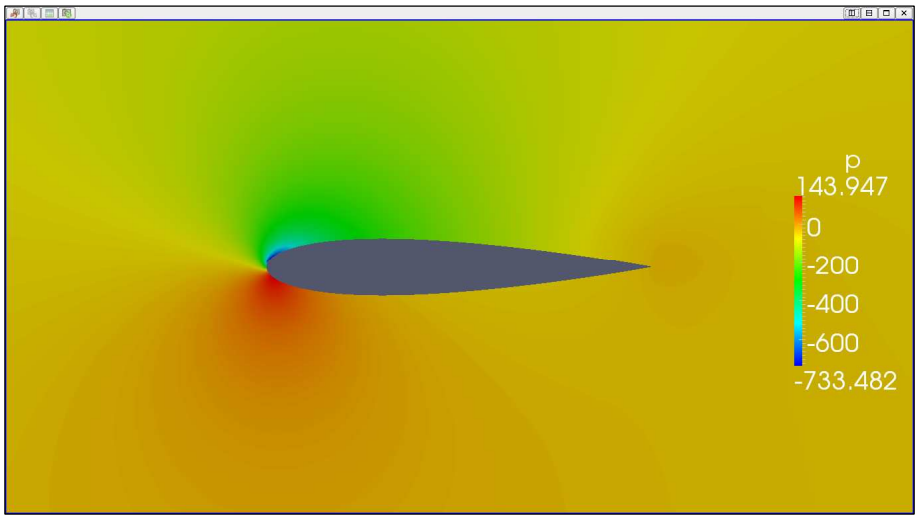
ForceCoeffs output:

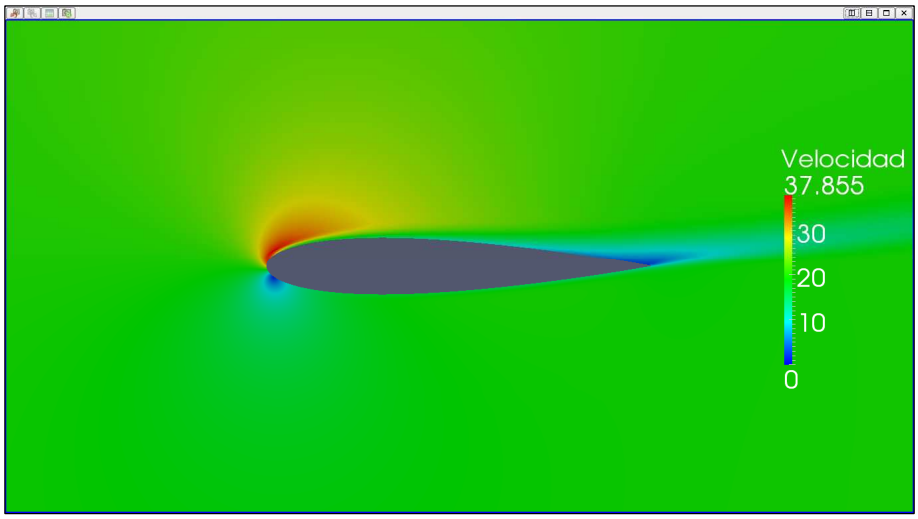
$C_L = 0{,}9441$

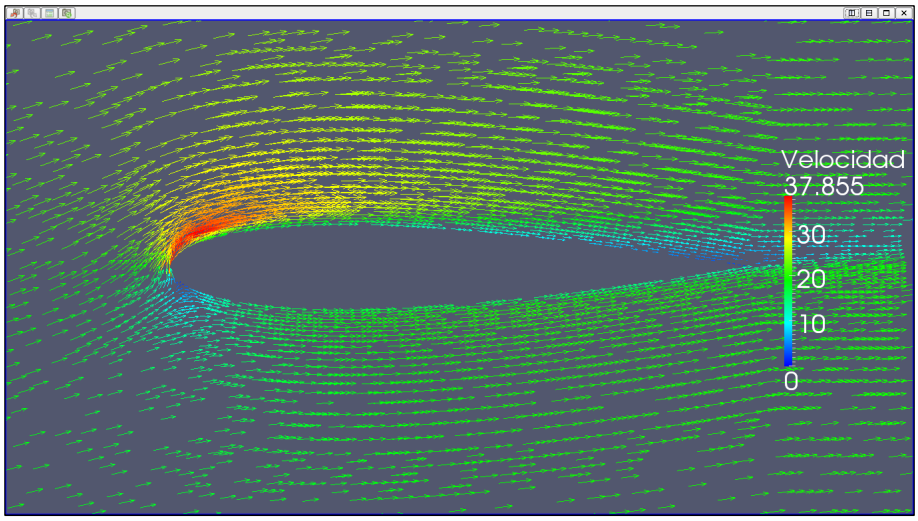
$C_D = 0{,}0363$






p
143.947
0
-200
-400
-600
-733.482


Velocidad
37.855
30
20
10
0


Velocidad
37.855
30
20
10
0

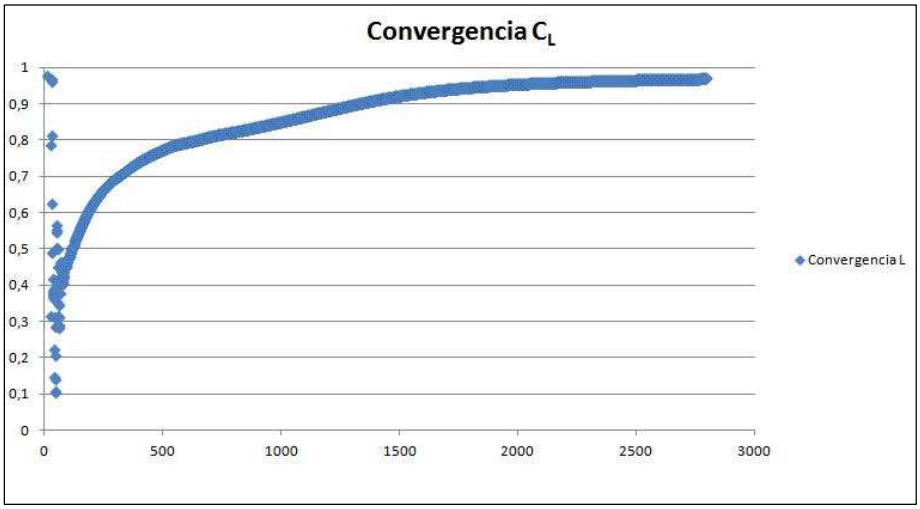
II.2.12.2. Caso 2 (Ángulo de ataque 10º)

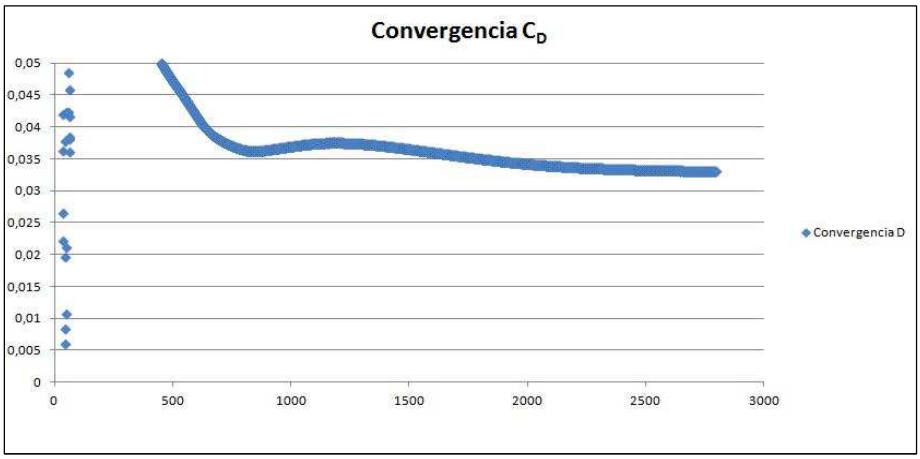
SIMPLE solution converged in 2797 iterations.

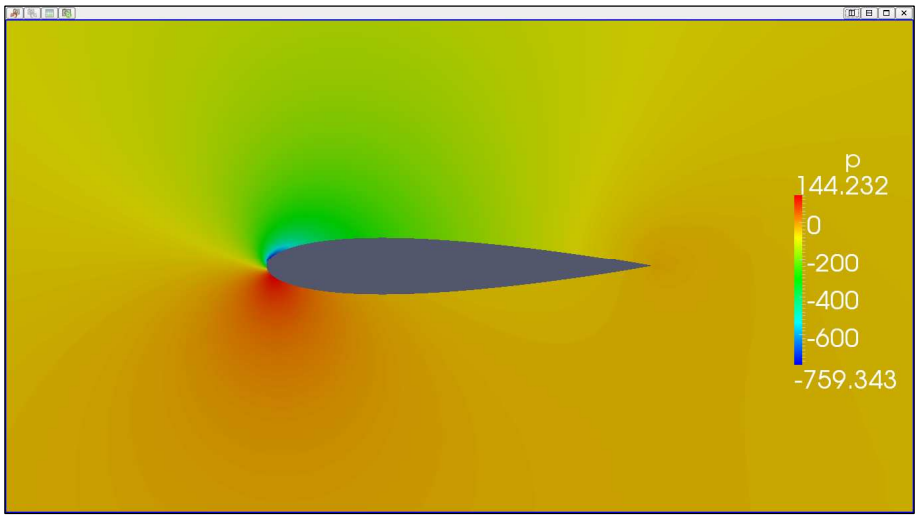
ForceCoeffs output:

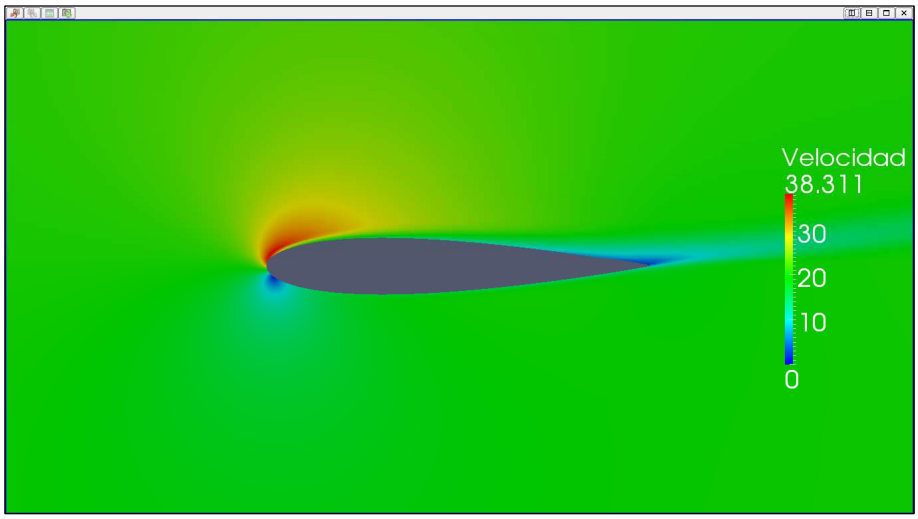
$C_L = 0{,}9693$

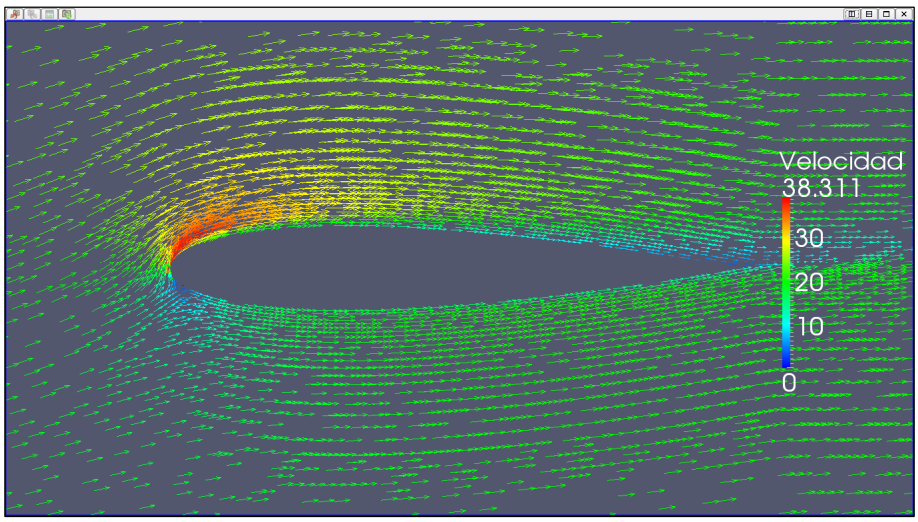
$C_D = 0{,}0330$






p
144.232
0
-200
-400
-600
-759.343


Velocidad
38.311
30
20
10
0


Velocidad
38.311
30
20
10
0

II.2.12.3. Caso 3 (Ángulo de ataque 10º)

Time: 50000

Solving U_X: Initial residual $= 1{,}892 \cdot 10^{-05}$, Final residual $= 1{,}871 \cdot 10^{-06}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 1{,}616 \cdot 10^{-05}$, Final residual $= 1{,}430 \cdot 10^{-06}$, No Iterations $= 2$.

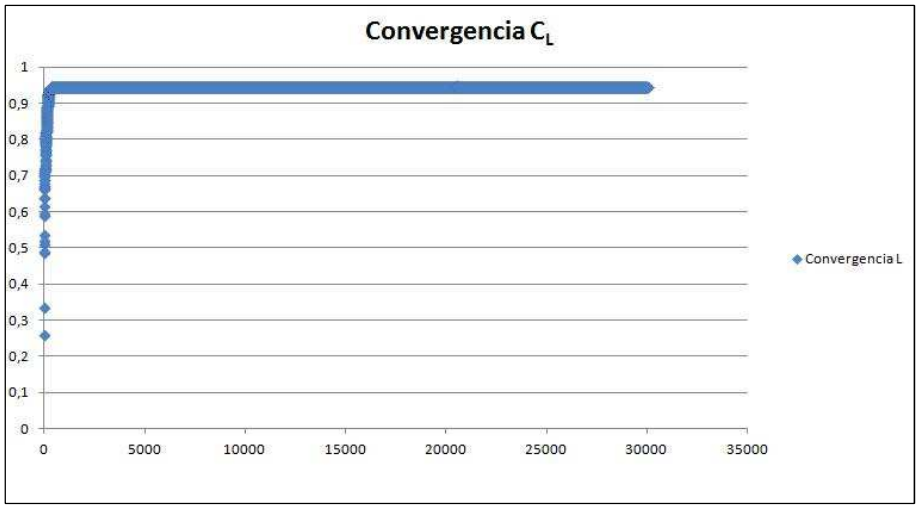
Solving p: Initial residual $= 1{,}460 \cdot 10^{-03}$, Final residual $= 1{,}425 \cdot 10^{-04}$, No Iterations $= 2$.

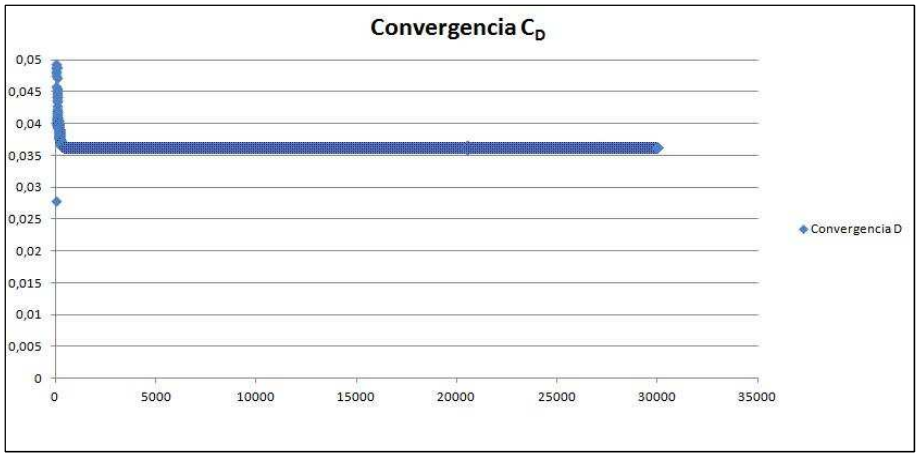
ExecutionTime: 5127,12 s. ClockTime: 5135 s.

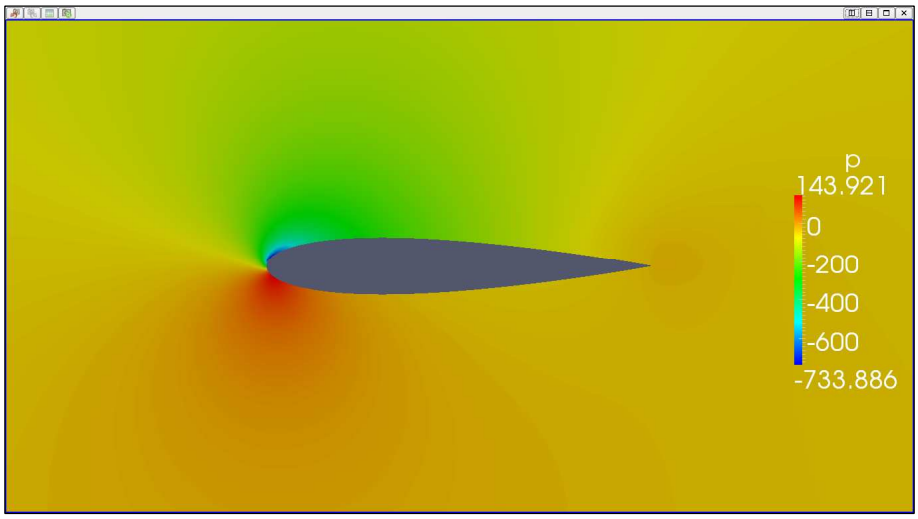
ForceCoeffs output:

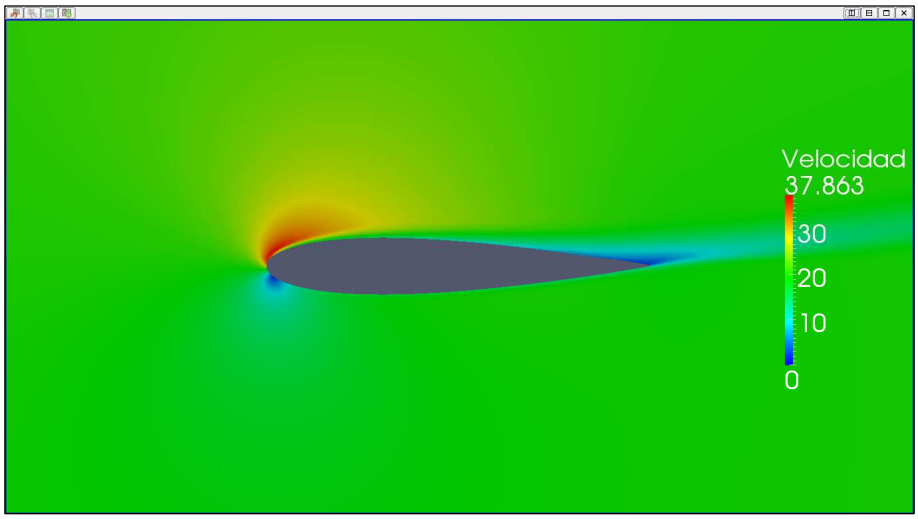
$C_L = 0{,}9446$

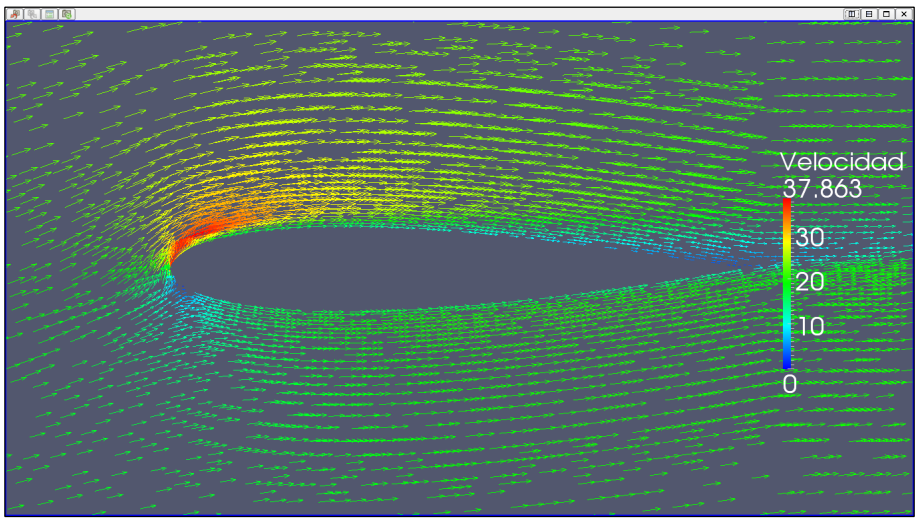
$C_D = 0{,}0363$






p
143.921
0
-200
-400
-600
-733.886


Velocidad
37.863
30
20
10
0


Velocidad
37.863
30
20
10
0

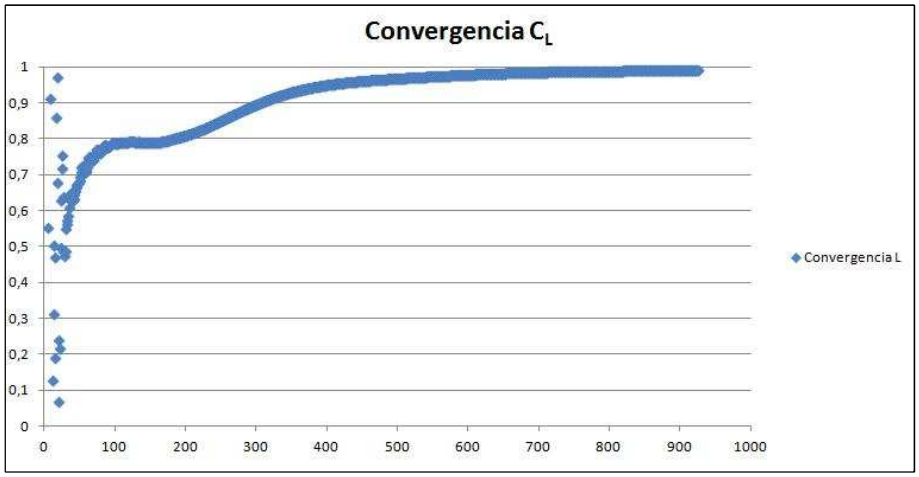
II.2.12.4. Caso 4 (Ángulo de ataque 10º)

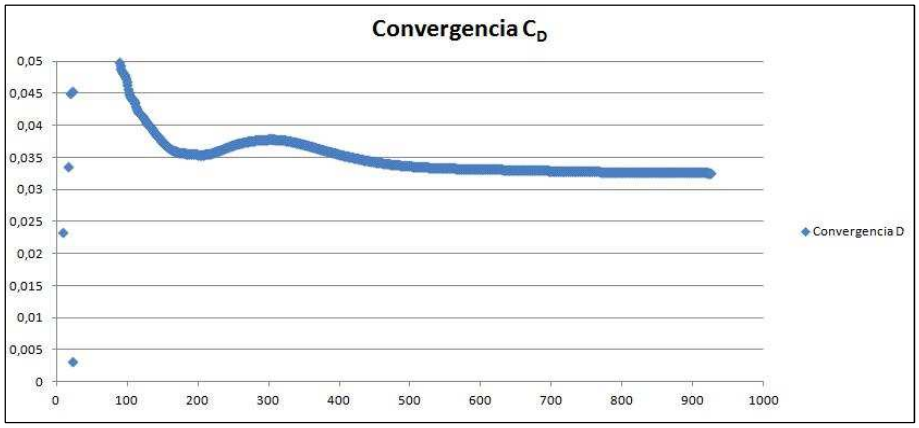
SIMPLE solution converged in 957 iterations.

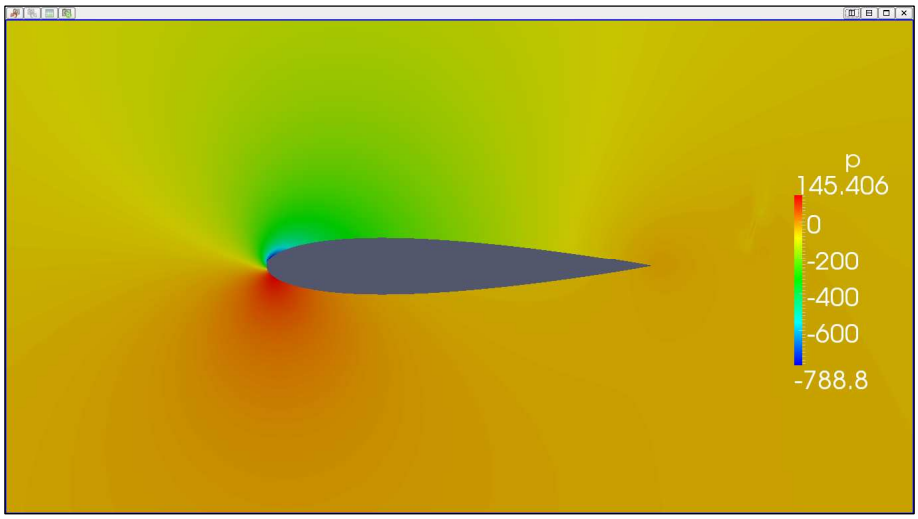
ForceCoeffs output:

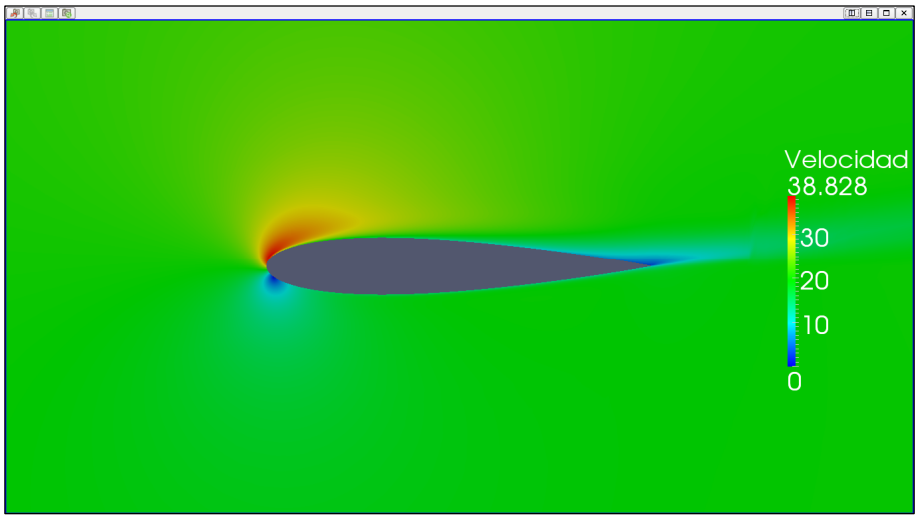
$C_L = 0{,}9904$

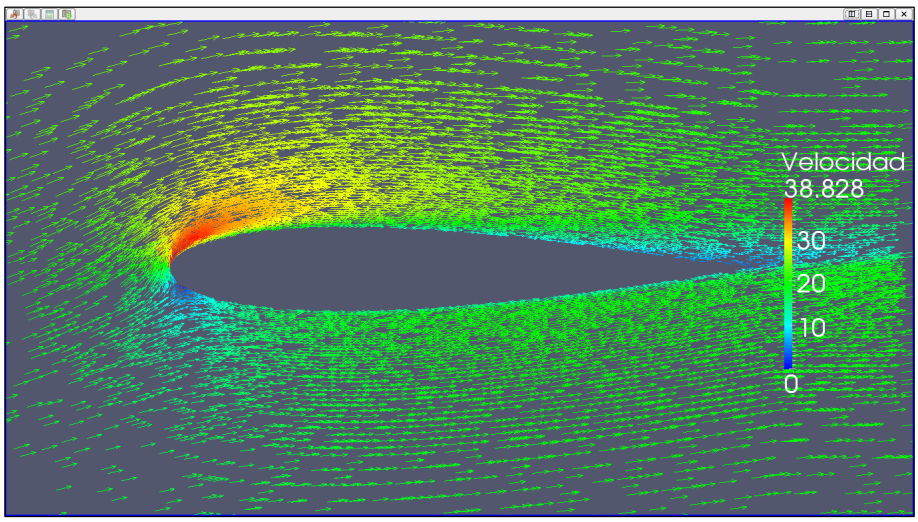
$C_D = 0{,}0326$






p
145.406
0
-200
-400
-600
-788.8


Velocidad
38.828
30
20
10
0


Velocidad
38.828
30
20
10
0

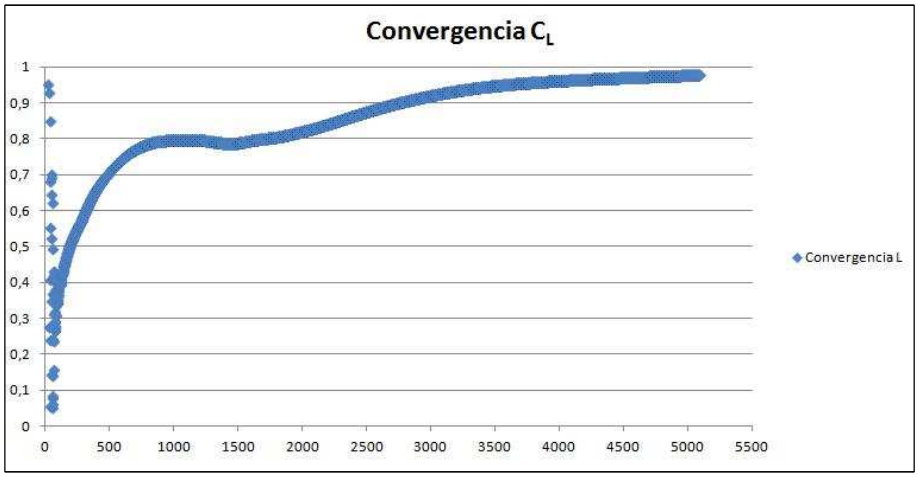
II.2.12.5. Caso 5 (Ángulo de ataque 10º)

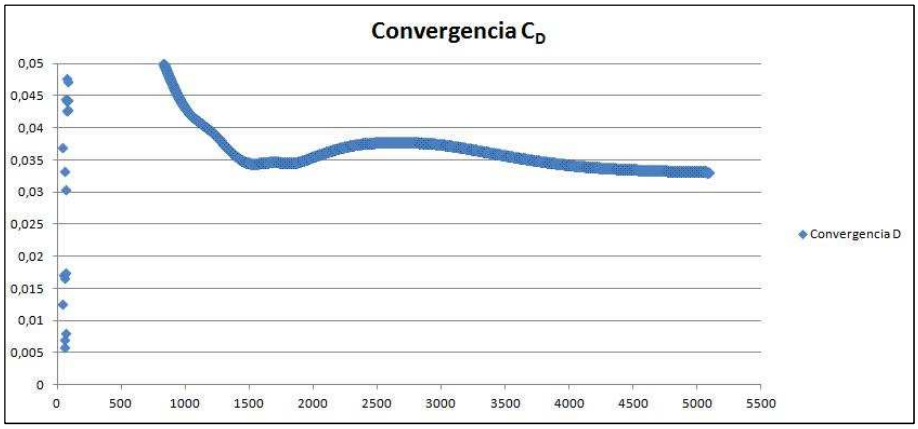
SIMPLE solution converged in 5094 iterations.

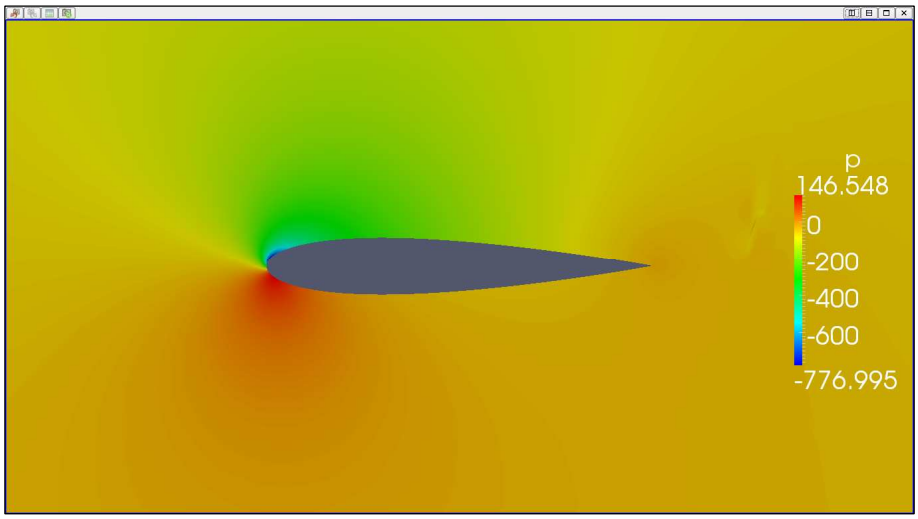
ForceCoeffs output:

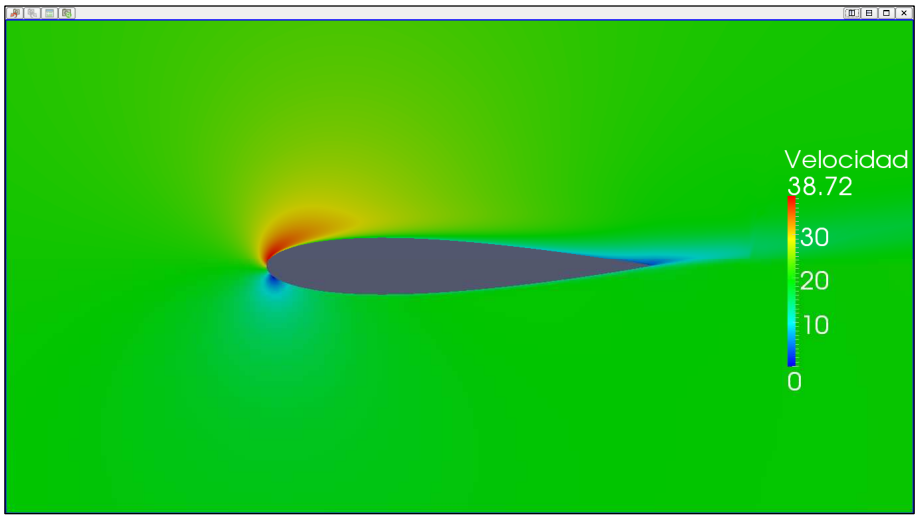
$C_L = 0{,}9774$

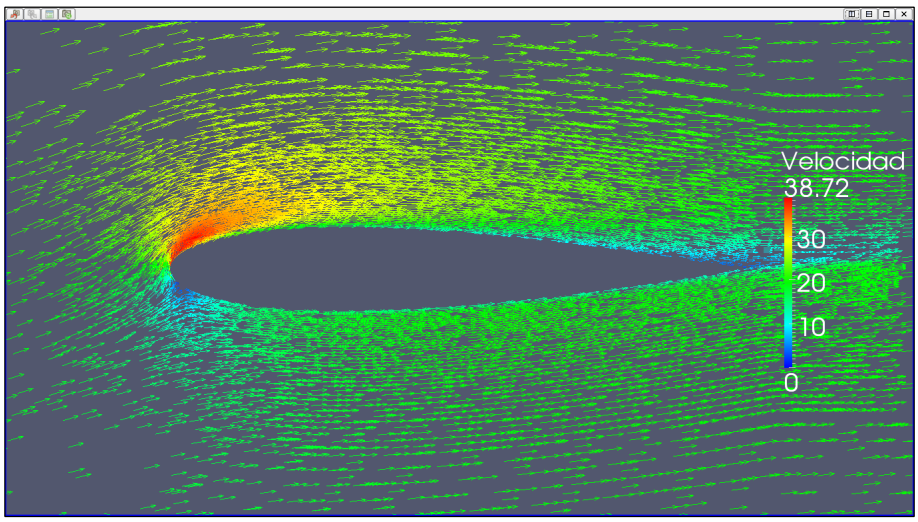
$C_D = 0{,}0331$






p
146.548
0
-200
-400
-600
-776.995


Velocidad
38.72
30
20
10
0


Velocidad
38.72
30
20
10
0

II.2.12.6. Caso 6 (Ángulo de ataque 10º)

Time: 50000

Solving U_X: Initial residual $= 9{,}474 \cdot 10^{-08}$, Final residual $= 4{,}331 \cdot 10^{-09}$, No Iterations $= 6$.

Solving U_Y: Initial residual $= 7{,}358 \cdot 10^{-08}$, Final residual $= 9{,}374 \cdot 10^{-09}$, No Iterations $= 4$.

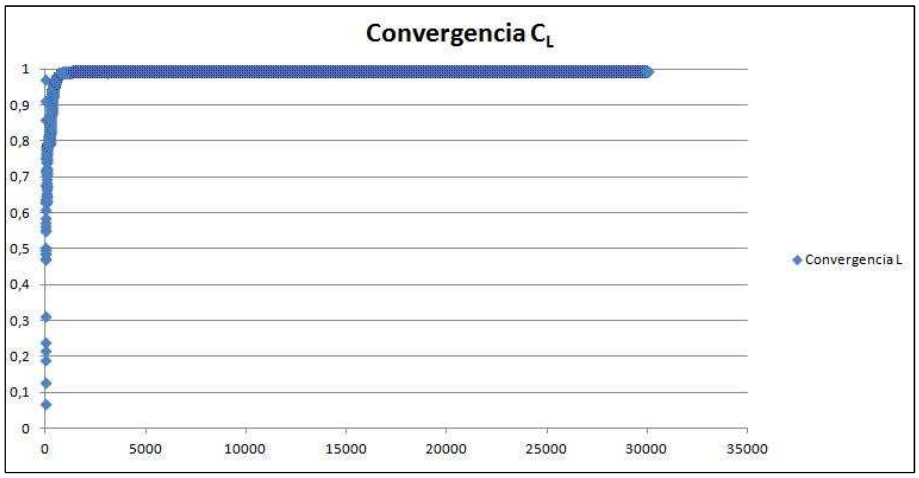
Solving p: Initial residual $= 7{,}139 \cdot 10^{-06}$, Final residual $= 9{,}999 \cdot 10^{-07}$, No Iterations $= 2$.

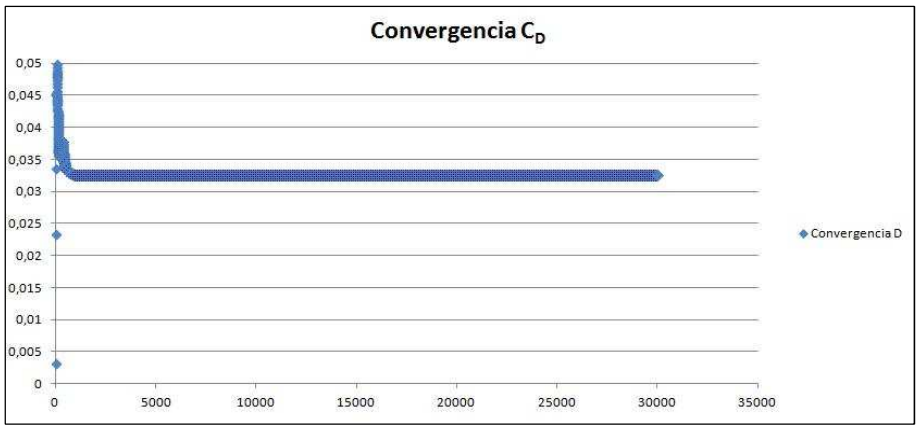
ExecutionTime: 13586,20 s. ClockTime: 13602 s.

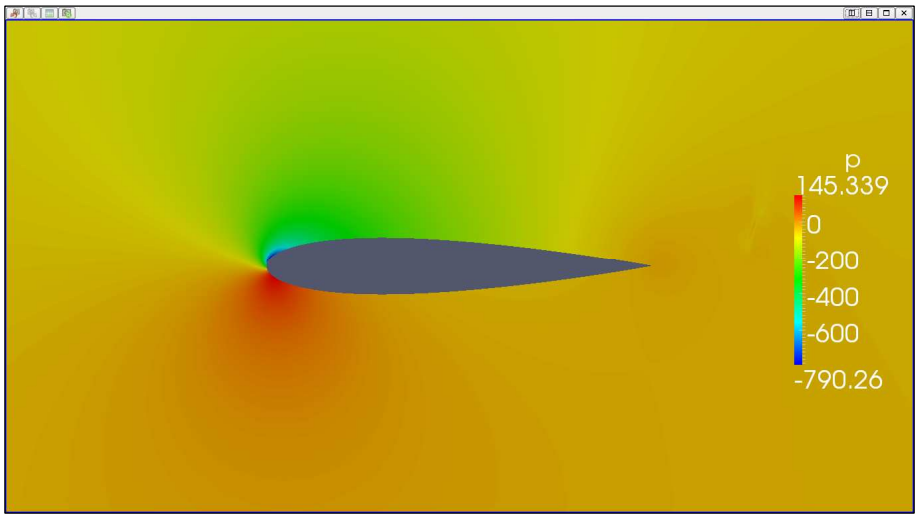
ForceCoeffs output:

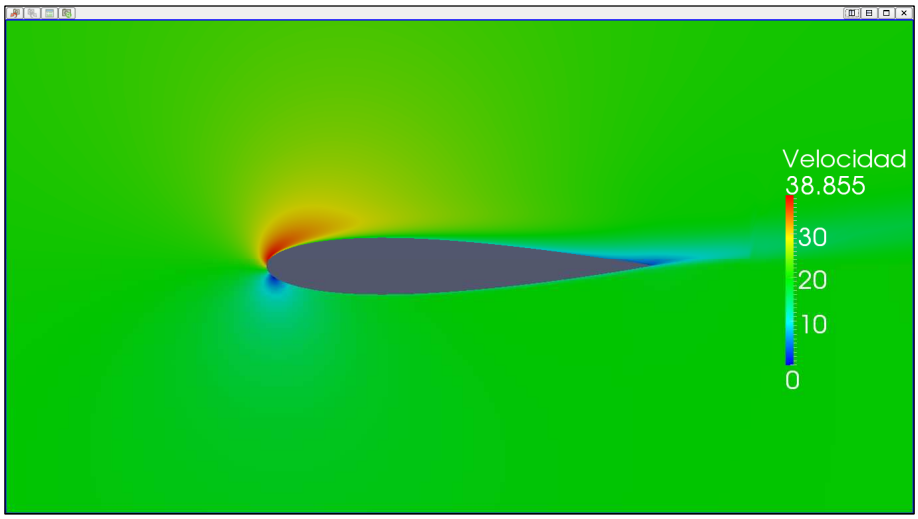
$C_L = 0{,}9922$

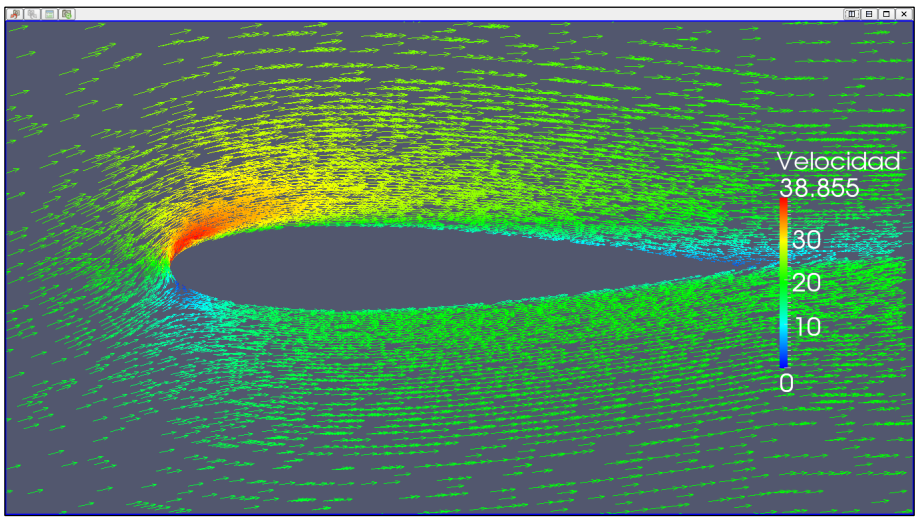
$C_D = 0{,}0325$






p
145.339
0
-200
-400
-600
-790.26


Velocidad
38.855
30
20
10
0


Velocidad
38.855
30
20
10
0

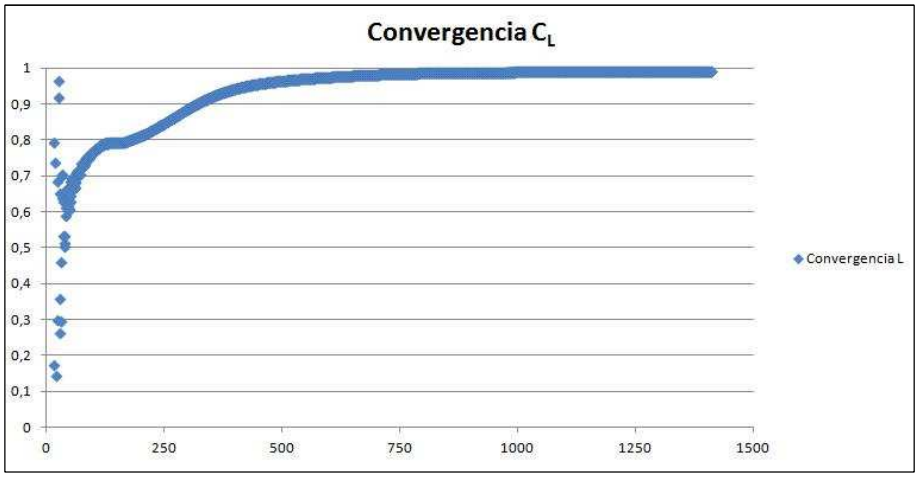
II.2.12.7. Caso 7 (Ángulo de ataque 10º)

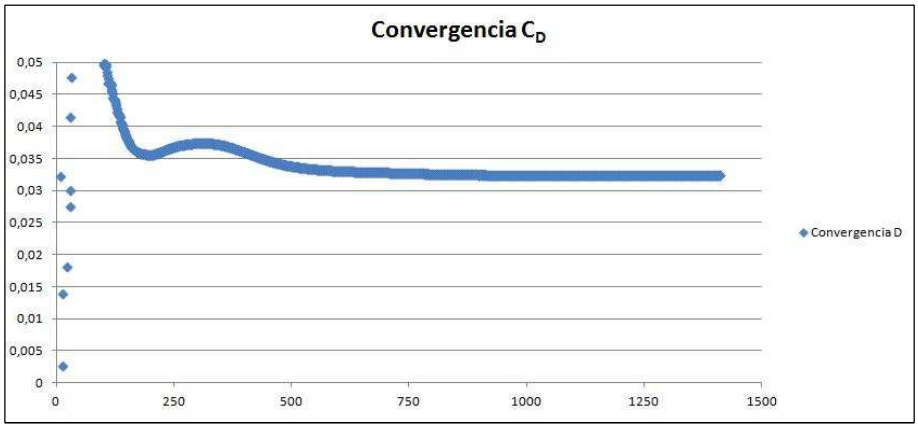
SIMPLE solution converged in 1412 iterations.

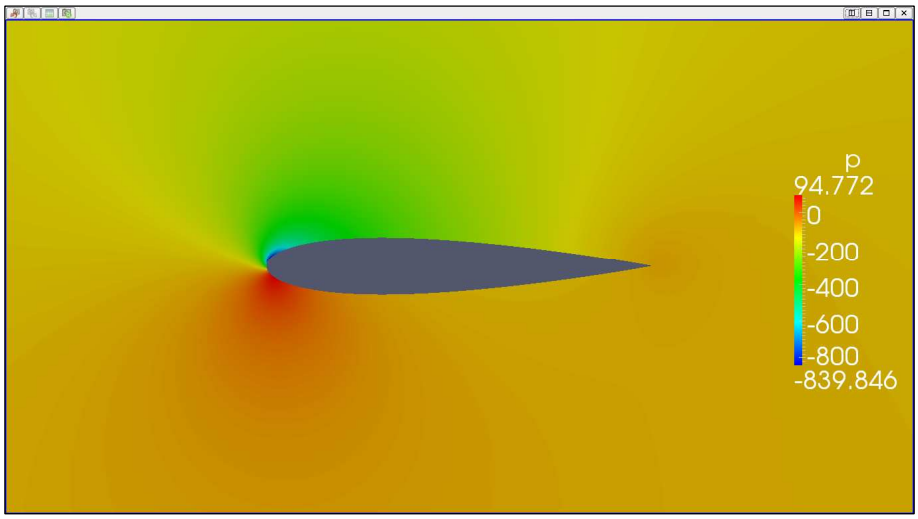
ForceCoeffs output:

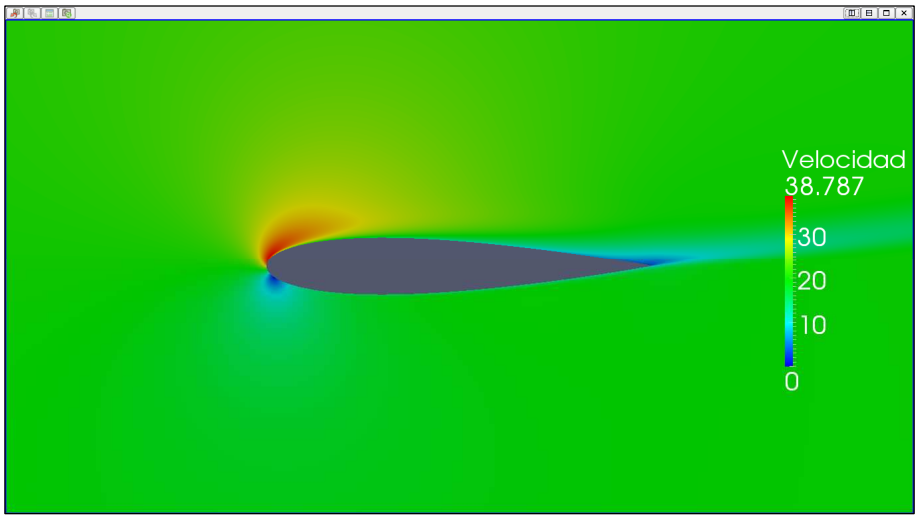
$C_L = 0{,}9901$

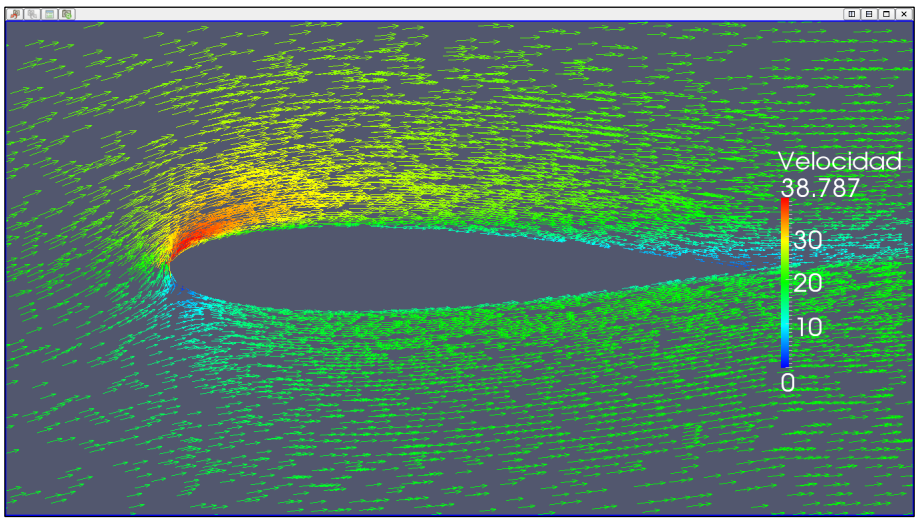
$C_D = 0{,}0323$






p
94.772
0
-200
-400
-600
-800
-839.846


Velocidad
38.787
30
20
10
0


Velocidad
38.787
30
20
10
0

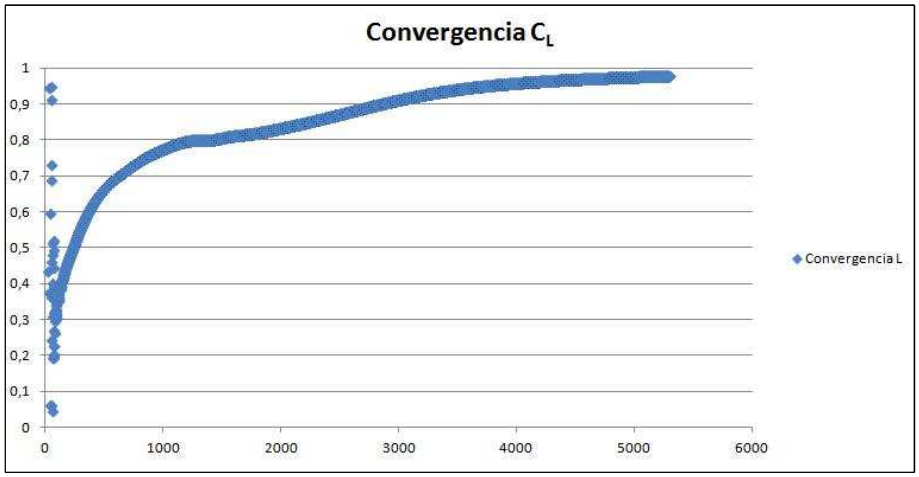
II.2.12.8. Caso 8 (Ángulo de ataque 10º)

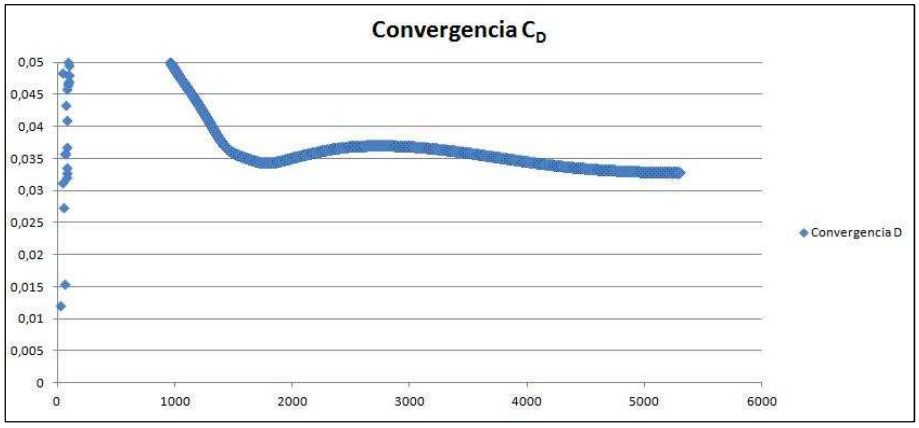
SIMPLE solution converged in 5299 iterations.

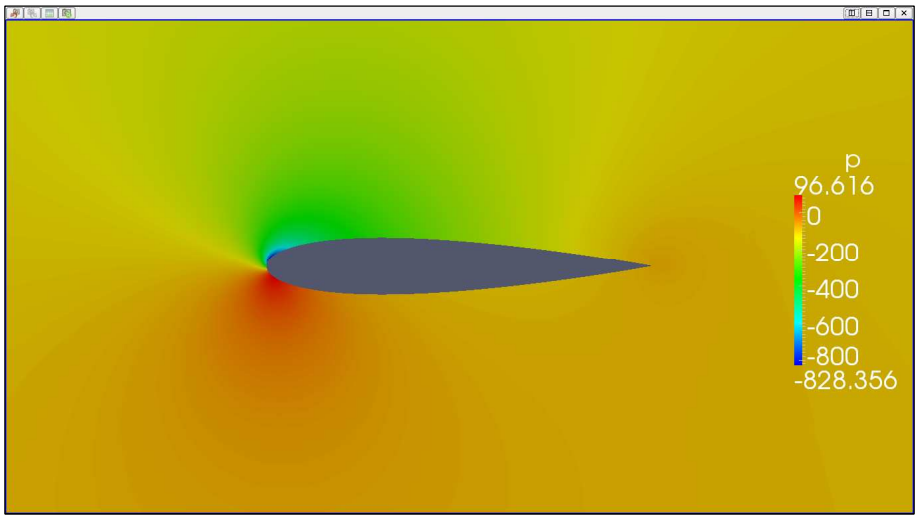
ForceCoeffs output:

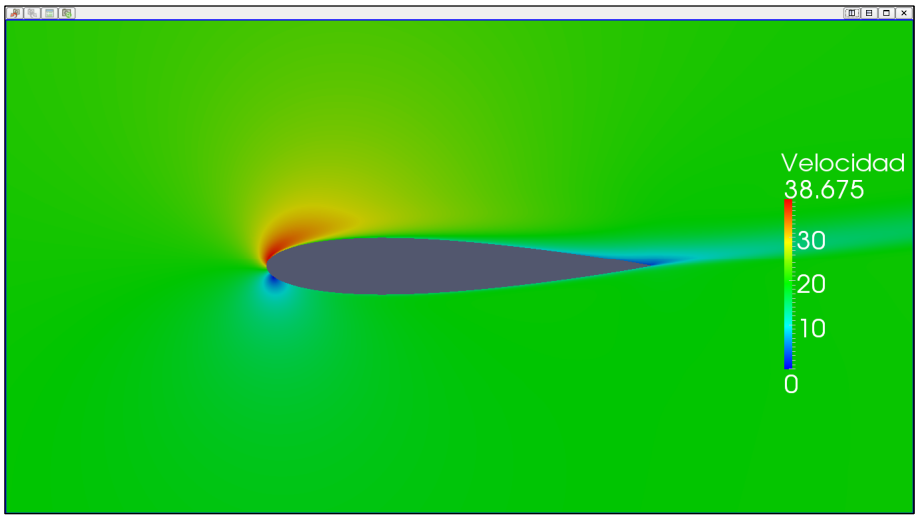
$C_L = 0{,}9786$

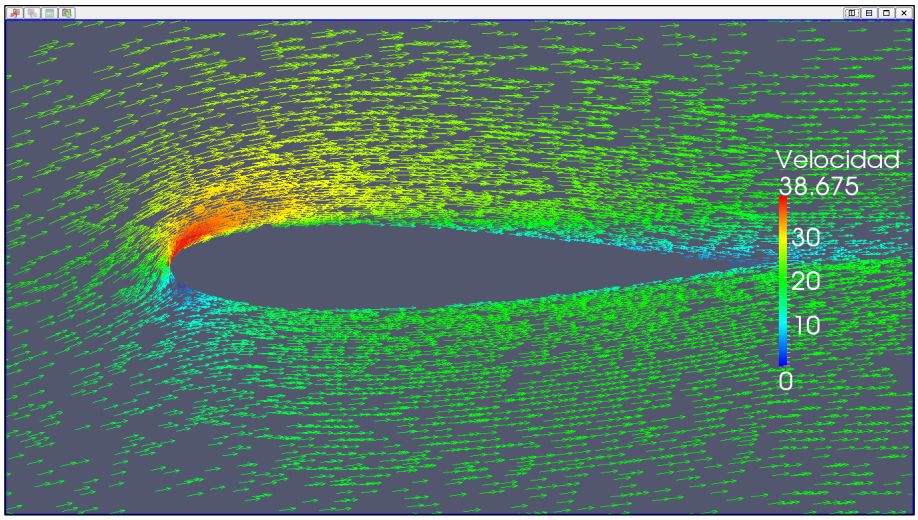
$C_D = 0{,}0328$






p
96.616
0
-200
-400
-600
-800
-828.356


Velocidad
38.675
30
20
10
0


Velocidad
38.675
30
20
10
0

II.2.13. Resultados ángulo de ataque 11º

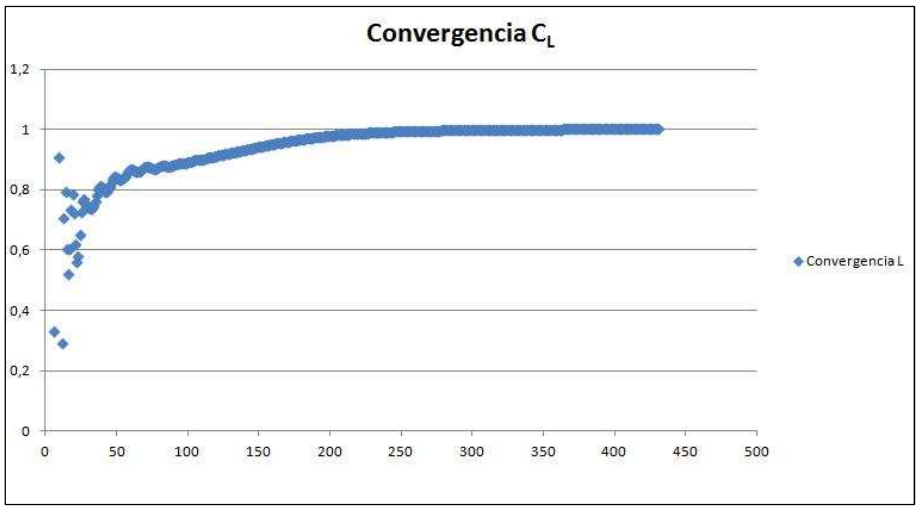
II.2.13.1. Caso 1 (Ángulo de ataque 11º)

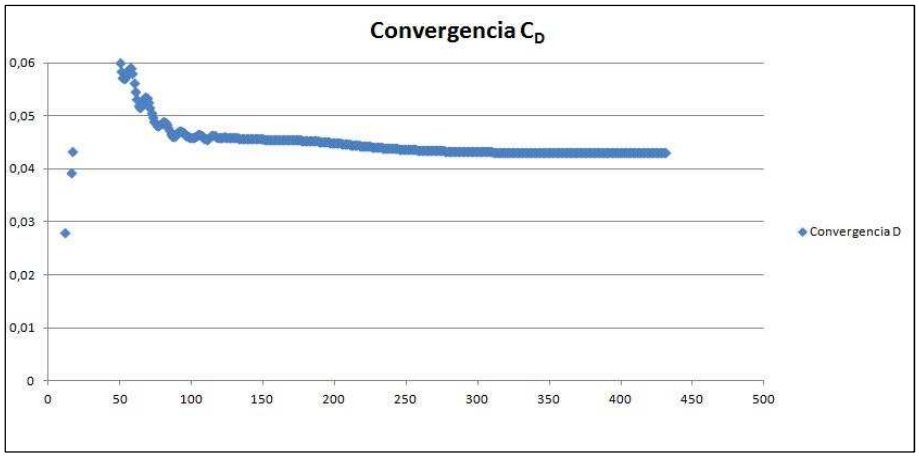
SIMPLE solution converged in 431 iterations.

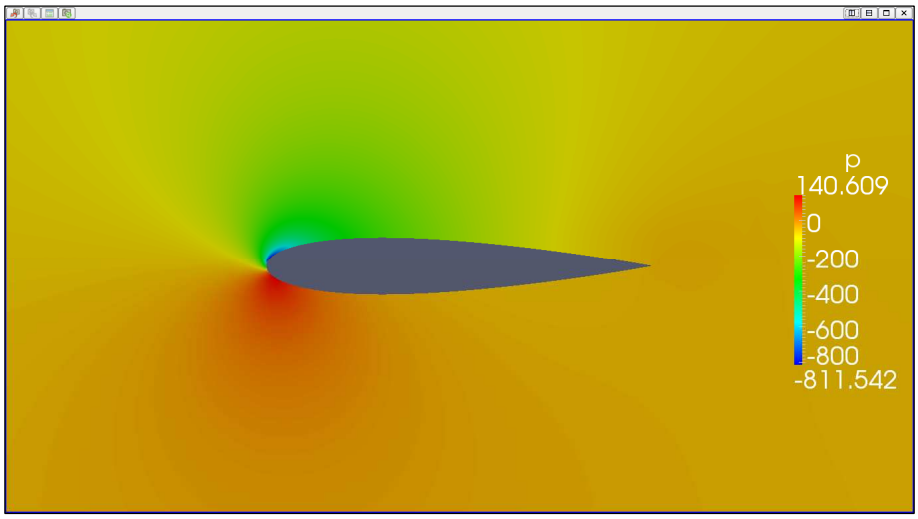
ForceCoeffs output:

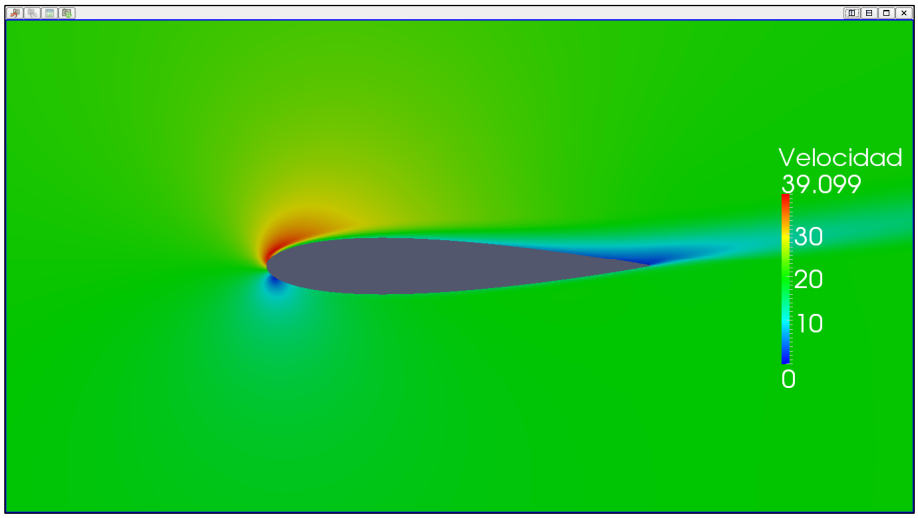
$C_L = 1{,}0020$

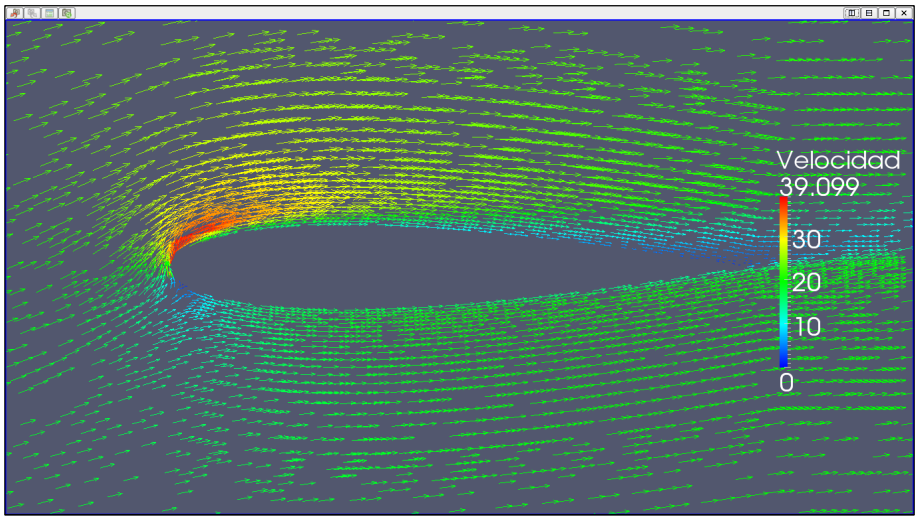
$C_D = 0{,}0430$






p
140.609
0
-200
-400
-600
-800
-811.542


Velocidad
39.099
30
20
10
0


Velocidad
39.099
30
20
10
0

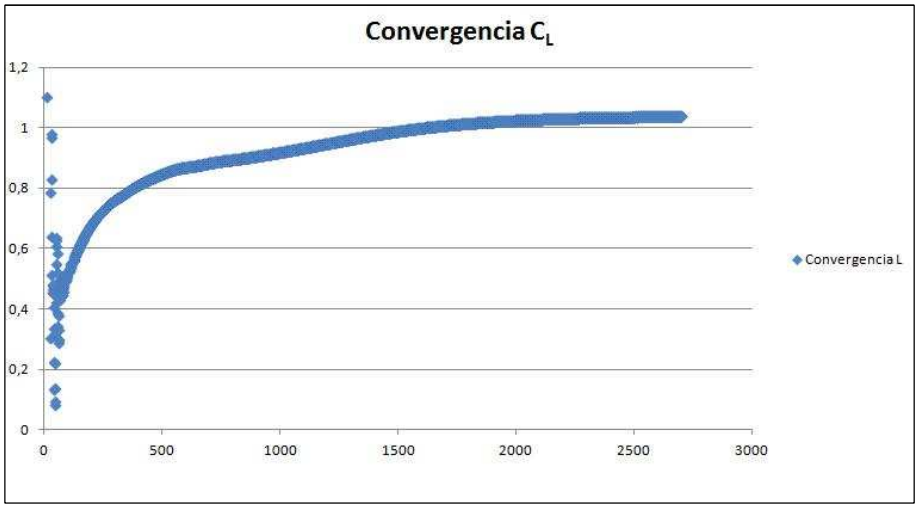
II.2.13.2. Caso 2 (Ángulo de ataque 11º)

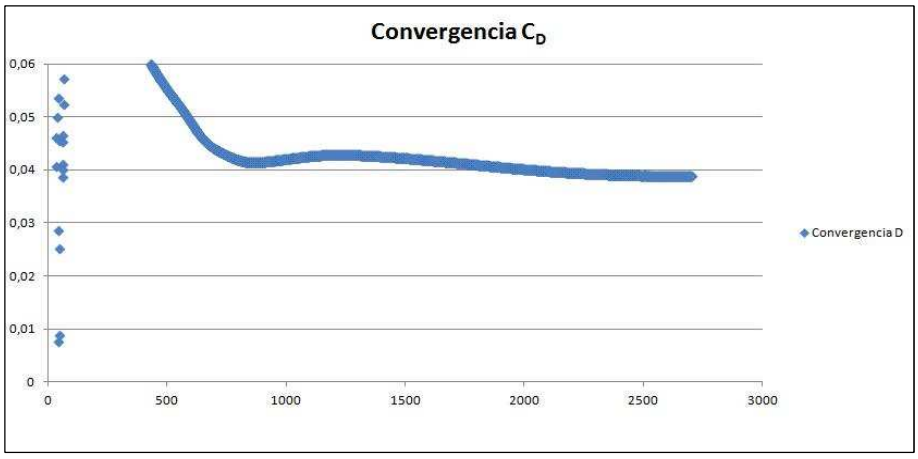
SIMPLE solution converged in 2703 iterations.

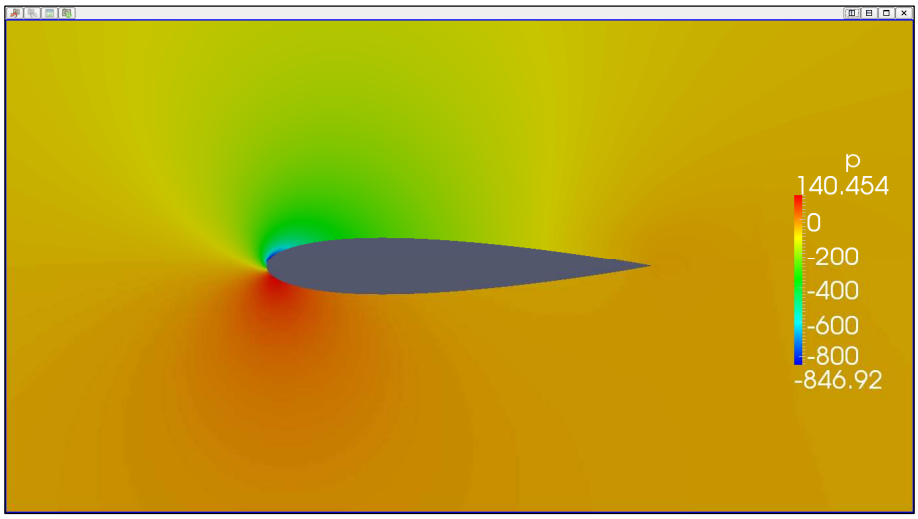
ForceCoeffs output:

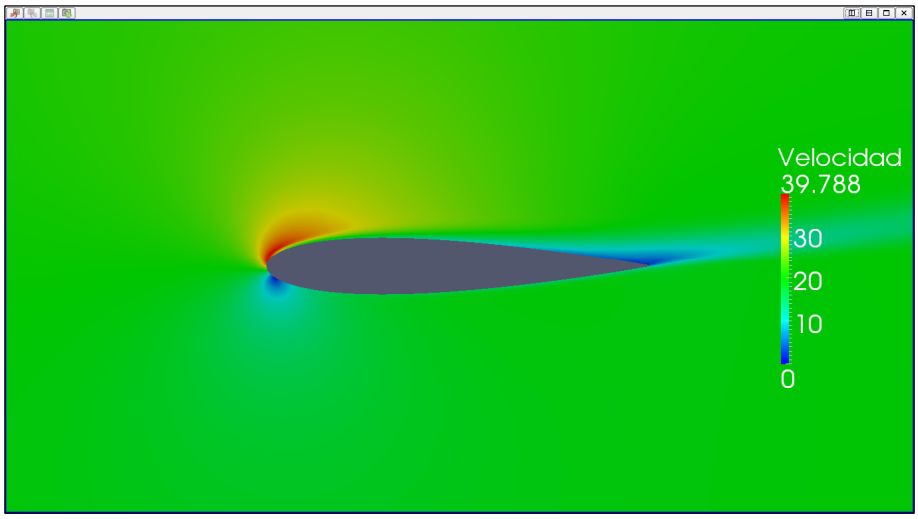
$C_L = 1{,}0386$

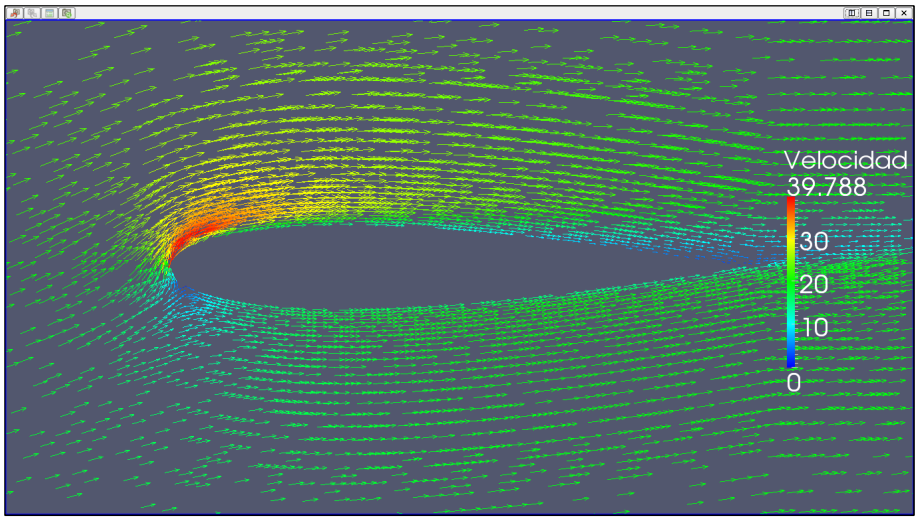
$C_D = 0{,}0387$






p
140.454
0
-200
-400
-600
-800
-846.92


Velocidad
39.788
30
20
10
0


Velocidad
39.788
30
20
10
0

II.2.13.3. Caso 3 (Ángulo de ataque 11º)

Time: 50000

Solving U_X: Initial residual $= 5{,}574 \cdot 10^{-07}$, Final residual $= 3{,}595 \cdot 10^{-08}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 3{,}476 \cdot 10^{-07}$, Final residual $= 2{,}609 \cdot 10^{-08}$, No Iterations $= 4$.

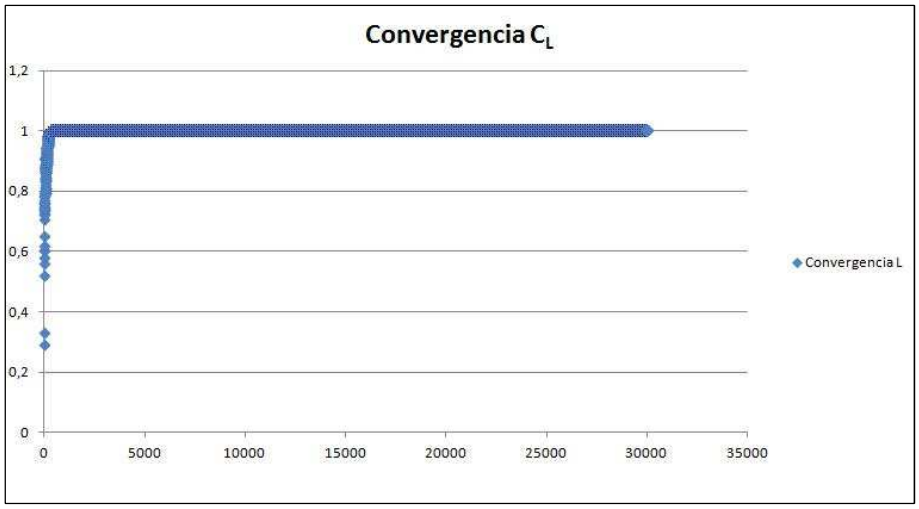
Solving p: Initial residual $= 2{,}083 \cdot 10^{-05}$, Final residual $= 1{,}947 \cdot 10^{-06}$, No Iterations $= 3$.

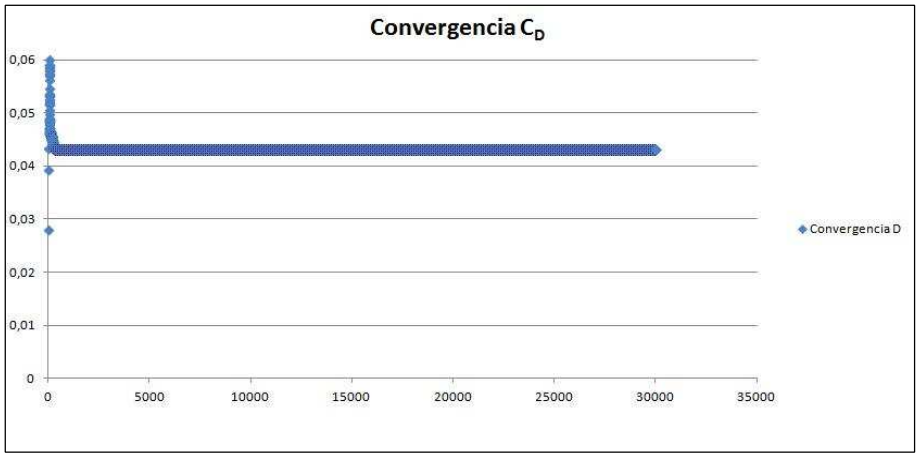
ExecutionTime: 5218,13 s. ClockTime: 5228 s.

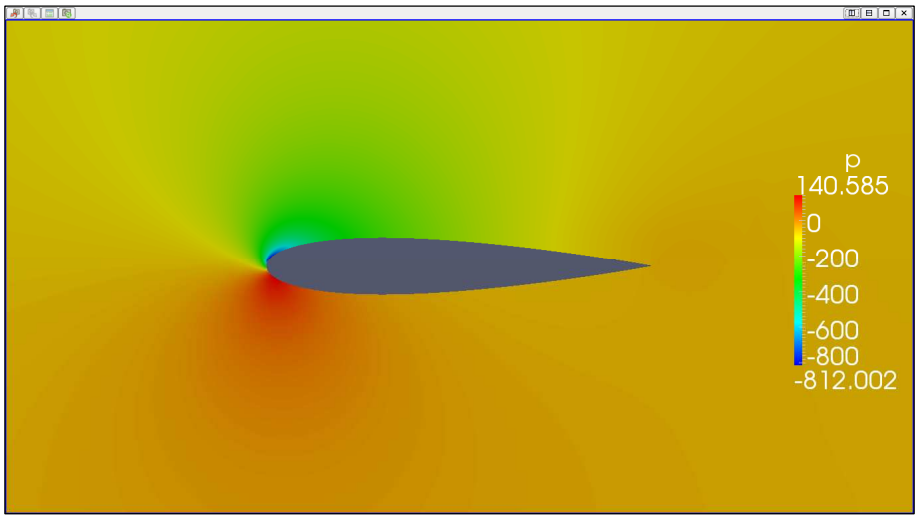
ForceCoeffs output:

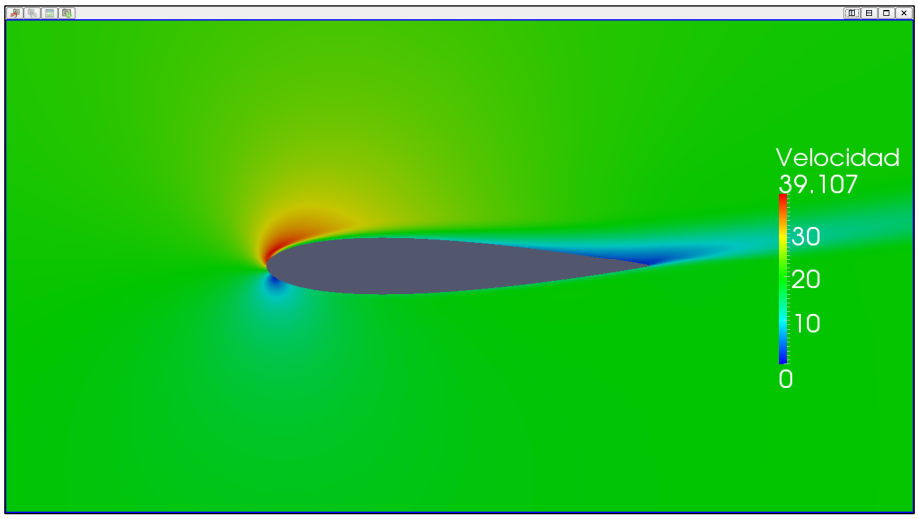
$C_L = 1{,}0025$

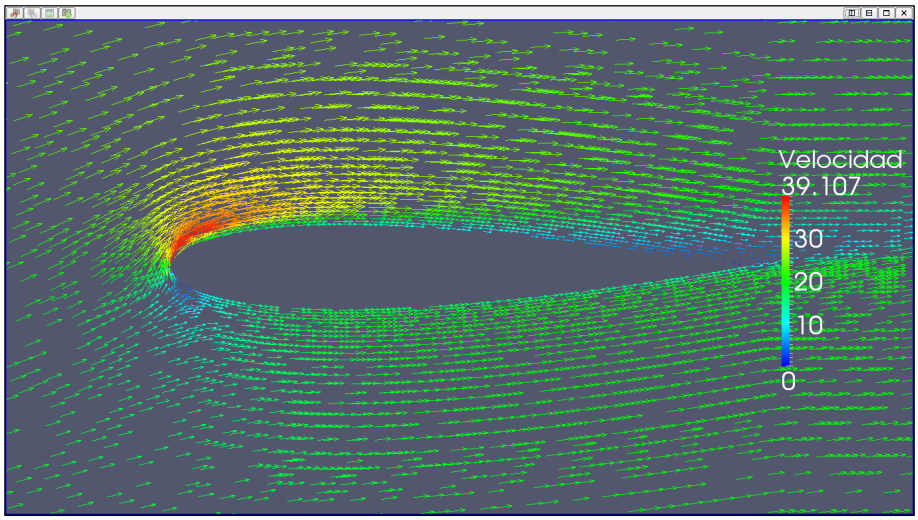
$C_D = 0{,}0430$






p
140.585
0
-200
-400
-600
-800
-812.002


Velocidad
39.107
30
20
10
0


Velocidad
39.107
30
20
10
0

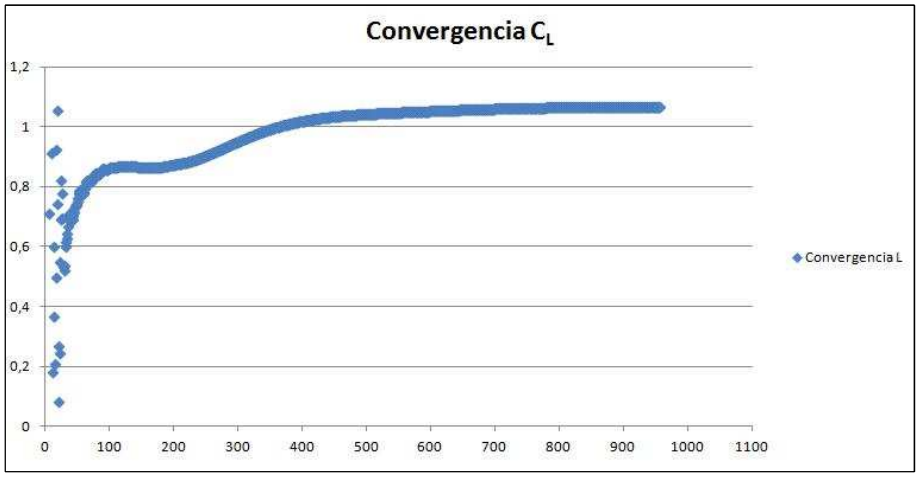
II.2.13.4. Caso 4 (Ángulo de ataque 11º)

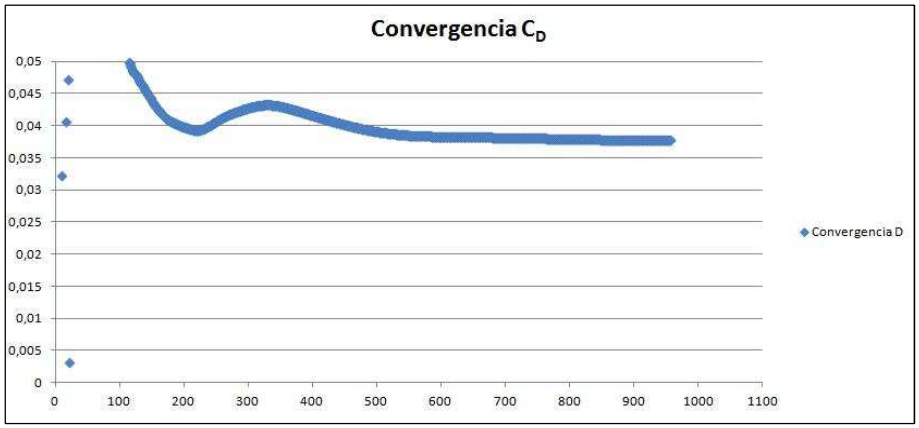
SIMPLE solution converged in 957 iterations.

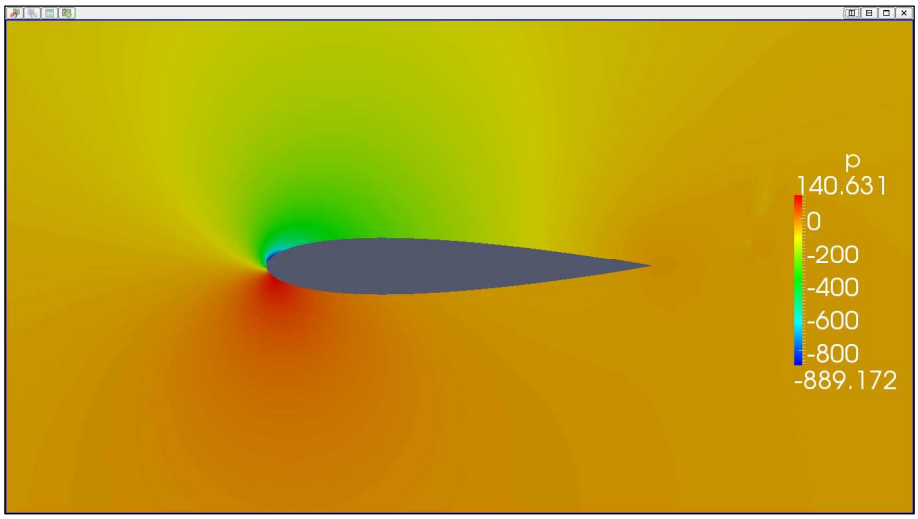
ForceCoeffs output:

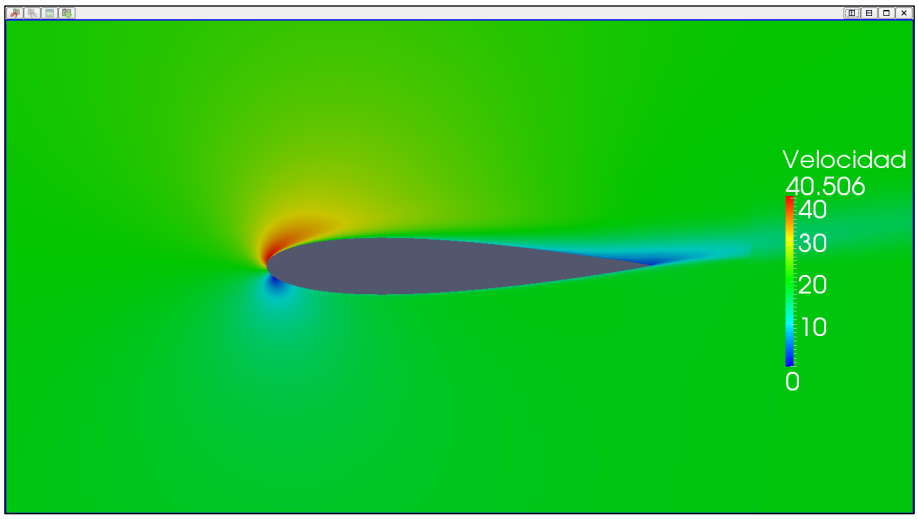
$C_L = 1{,}0675$

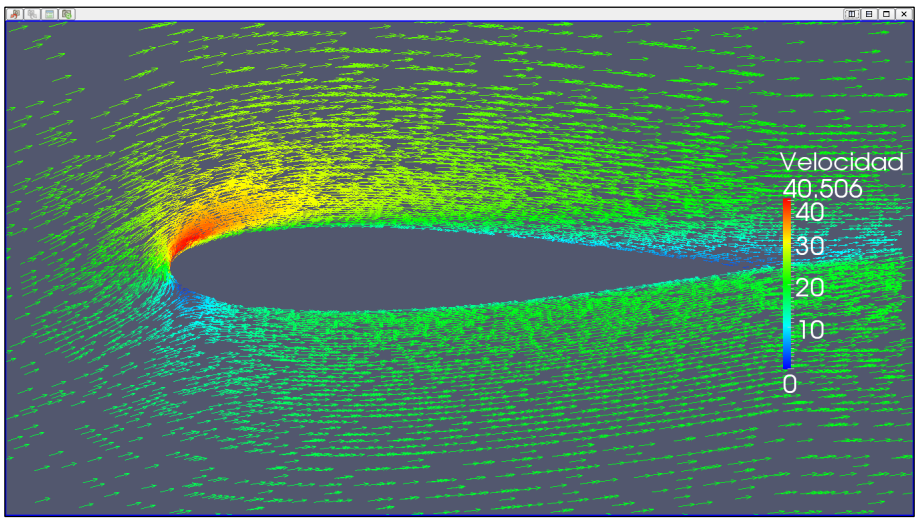
$C_D = 0{,}0377$






p
140.631
0
-200
-400
-600
-800
-889.172


Velocidad
40.506
40
30
20
10
0


Velocidad
40.506
40
30
20
10
0

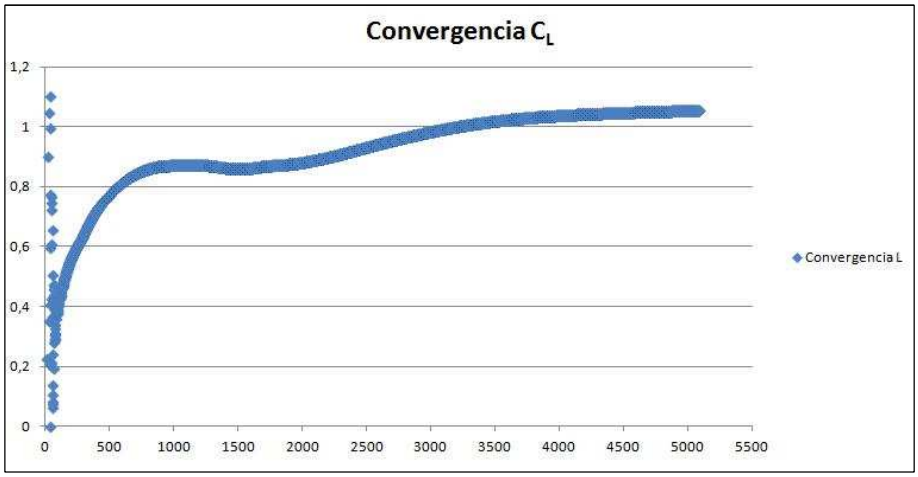
II.2.13.5. Caso 5 (Ángulo de ataque 11º)

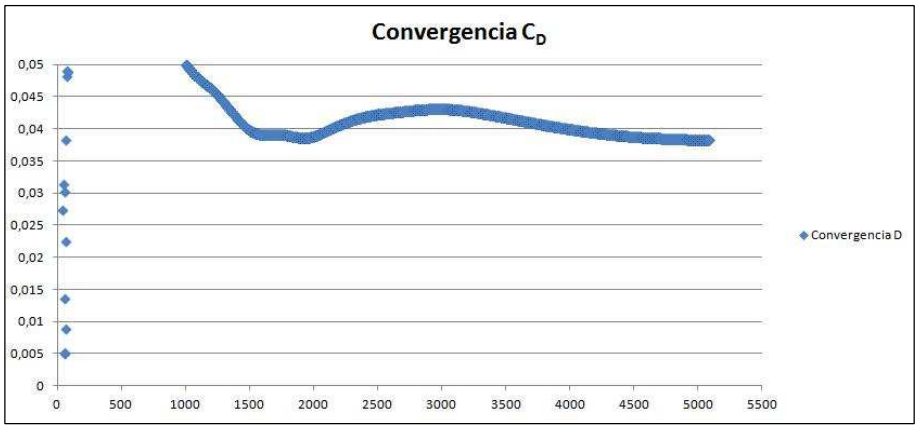
SIMPLE solution converged in 5090 iterations.

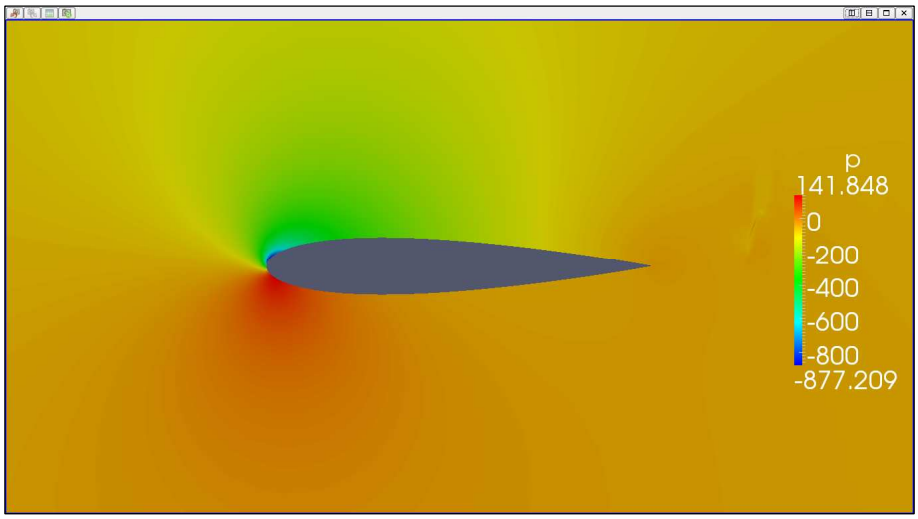
ForceCoeffs output:

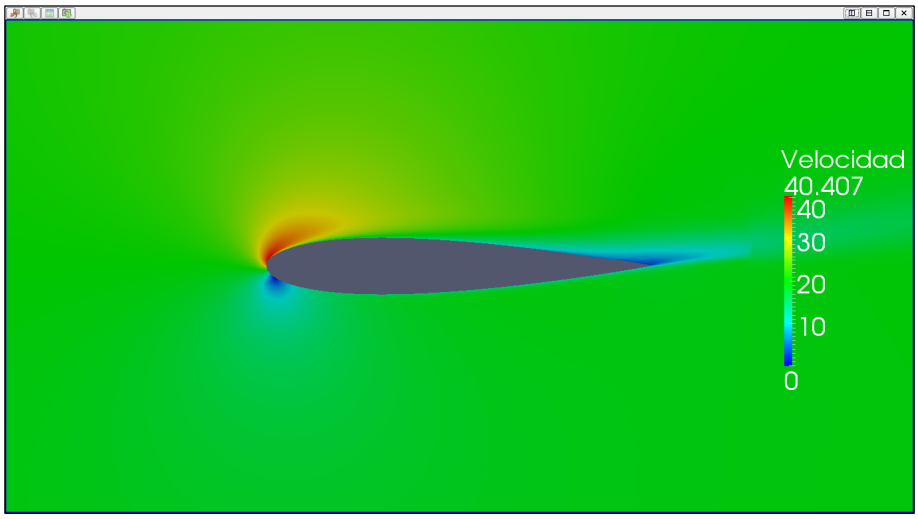
$C_L = 1{,}0548$

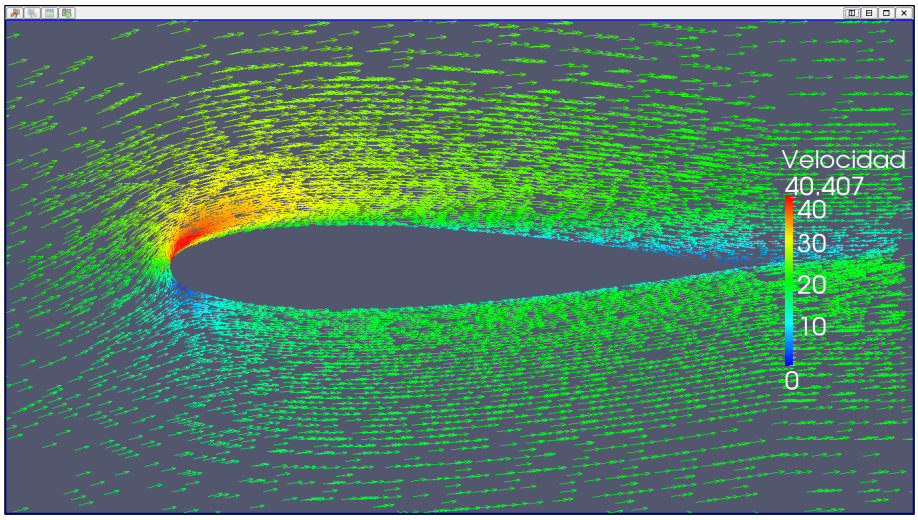
$C_D = 0{,}0382$






p
141.848
0
-200
-400
-600
-800
-877.209


Velocidad
40.407
40
30
20
10
0


Velocidad
40.407
40
30
20
10
0

II.2.13.6. Caso 6 (Ángulo de ataque 11º)

Time: 50000

Solving U_X: Initial residual $= 2{,}091 \cdot 10^{-07}$, Final residual $= 2{,}053 \cdot 10^{-08}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 1{,}694 \cdot 10^{-07}$, Final residual $= 9{,}499 \cdot 10^{-09}$, No Iterations $= 6$.

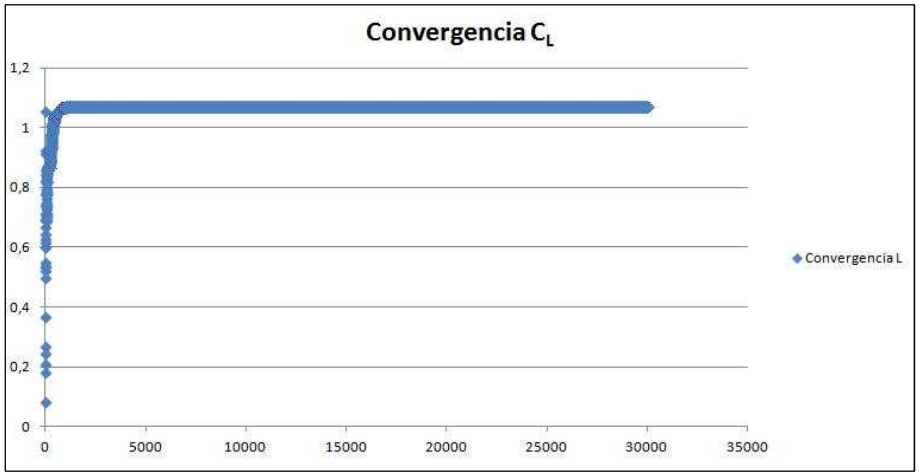
Solving p: Initial residual $= 1{,}139 \cdot 10^{-05}$, Final residual $= 7{,}942 \cdot 10^{-07}$, No Iterations $= 4$.

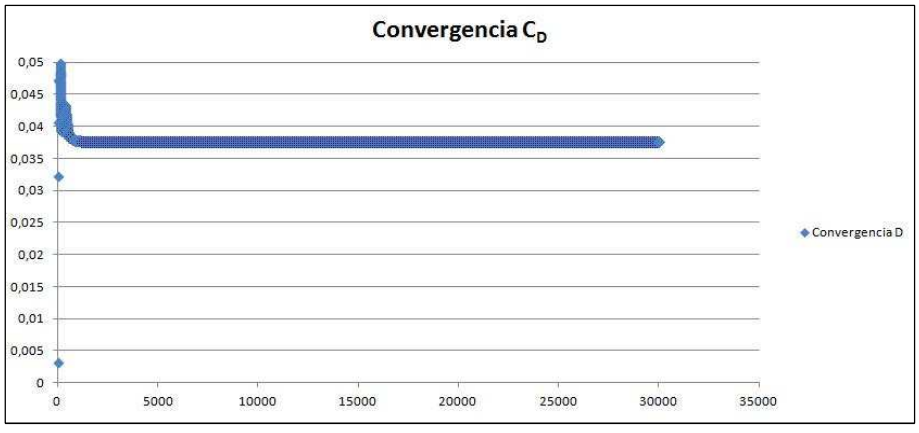
ExecutionTime: 14090,60 s. ClockTime: 14112 s.

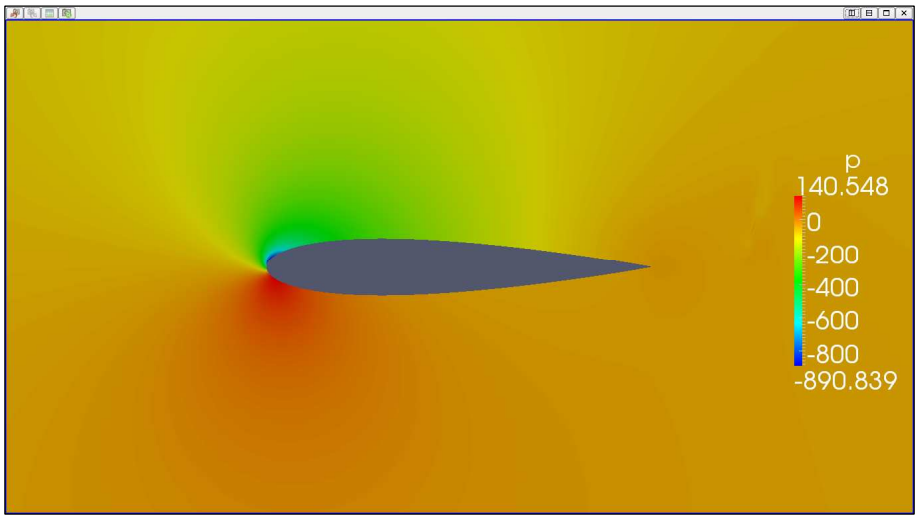
ForceCoeffs output:

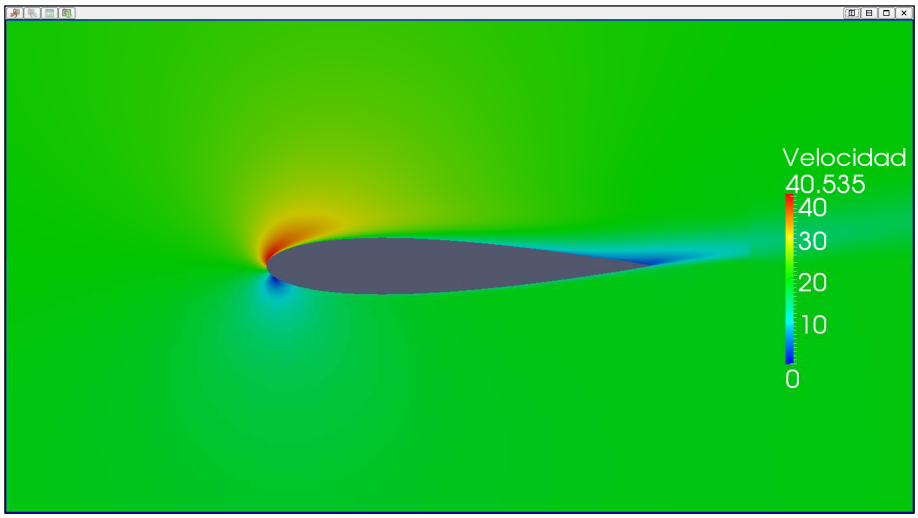
$C_L = 1{,}0695$

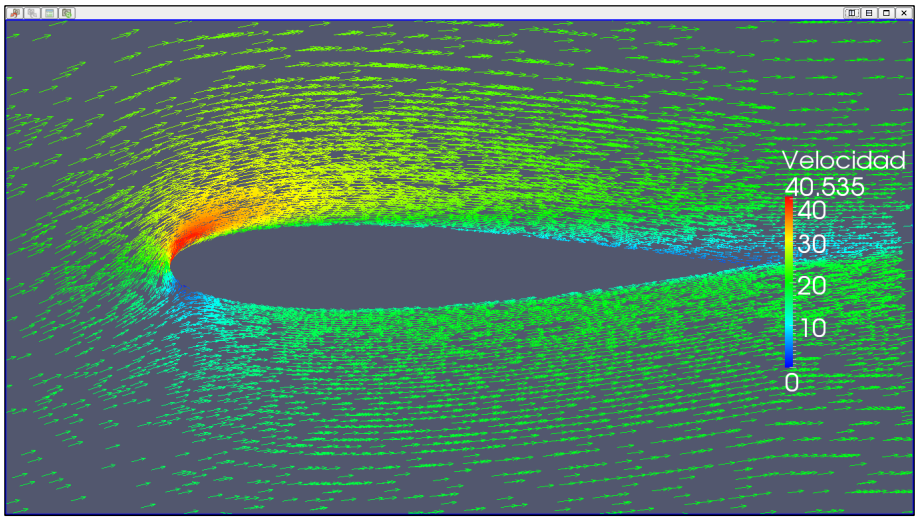
$C_D = 0{,}0376$






p
140.548
0
-200
-400
-600
-800
-890.839


Velocidad
40.535
40
30
20
10
0


Velocidad
40.535
40
30
20
10
0

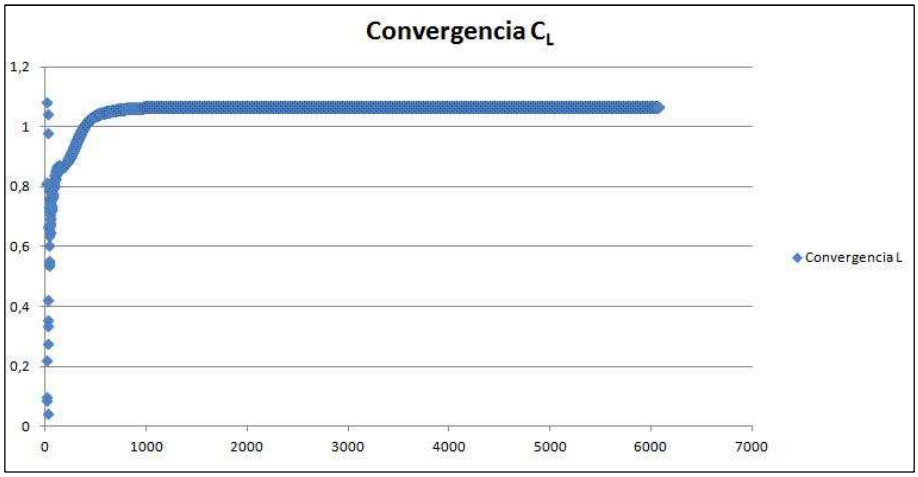
II.2.13.7. Caso 7 (Ángulo de ataque 11º)

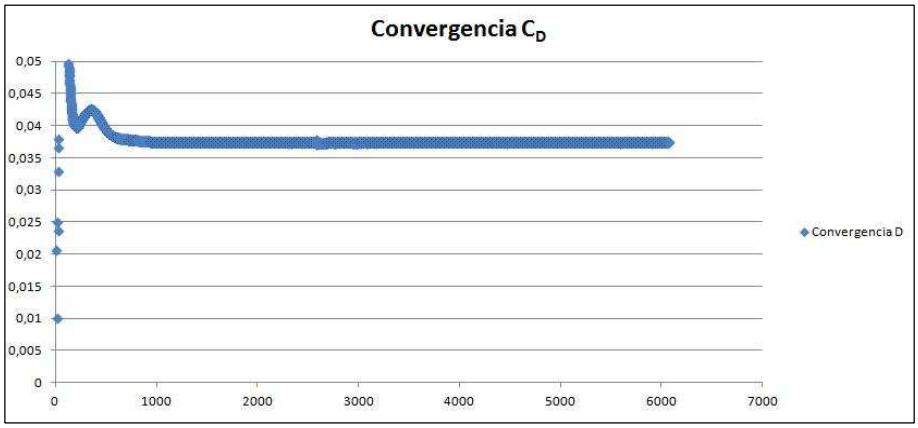
SIMPLE solution converged in 6074 iterations.

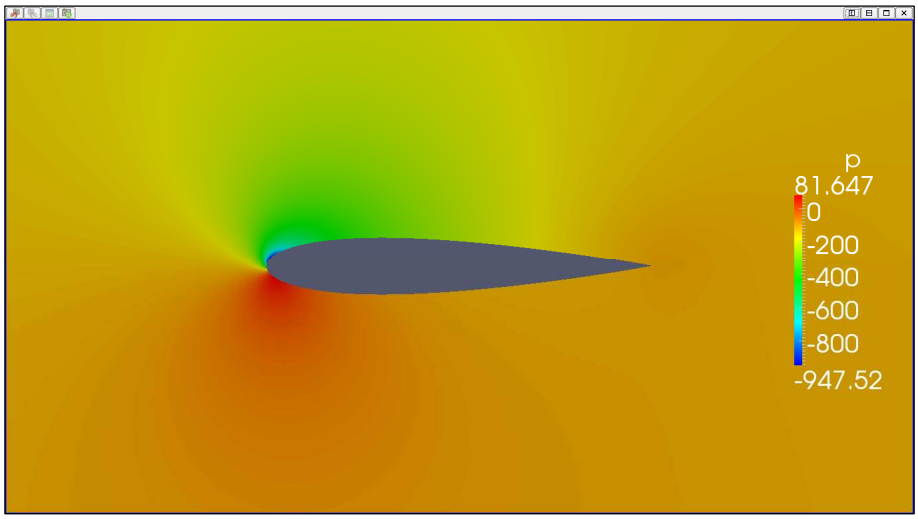
ForceCoeffs output:

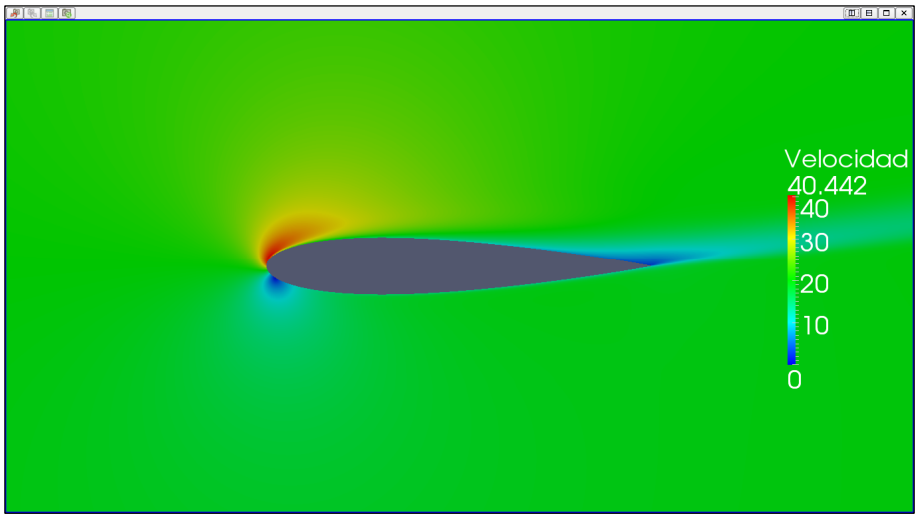
$C_L = 1{,}0659$

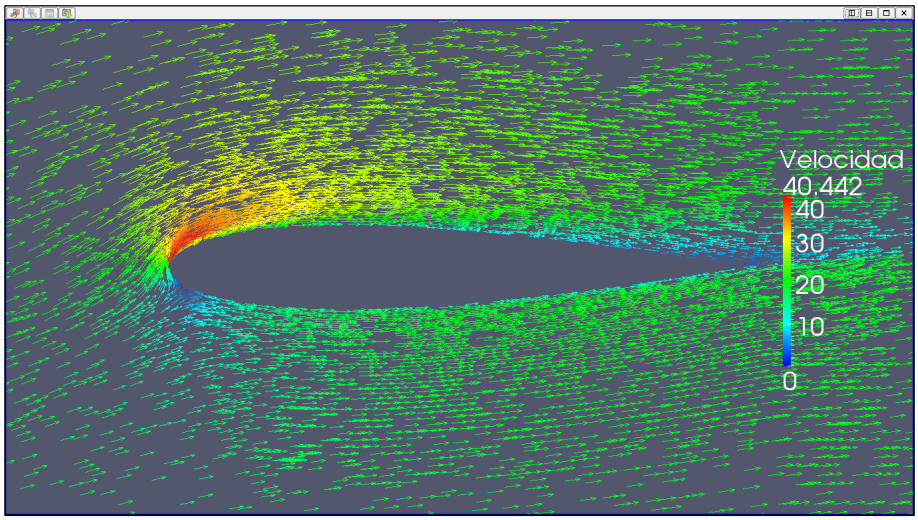
$C_D = 0{,}0373$






p
81.647
0
-200
-400
-600
-800
-947.52


Velocidad
40.442
40
30
20
10
0


Velocidad
40.442
40
30
20
10
0

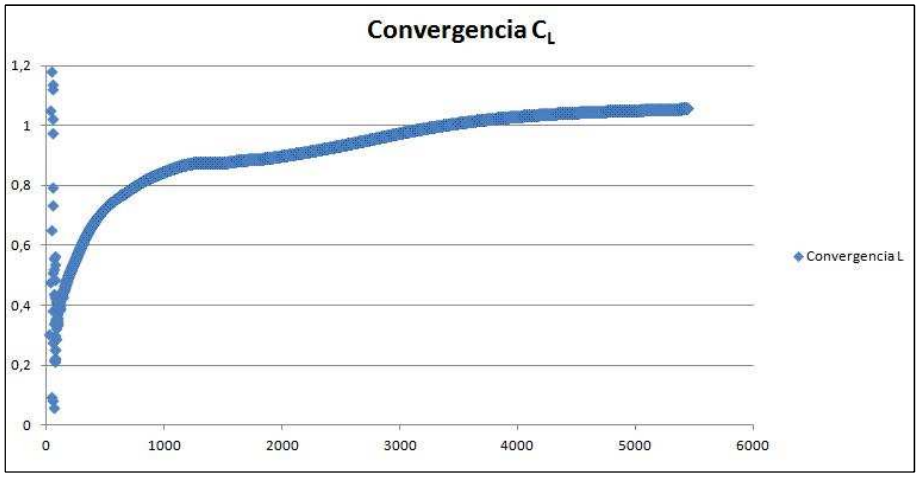
II.2.13.8. Caso 8 (Ángulo de ataque 11º)

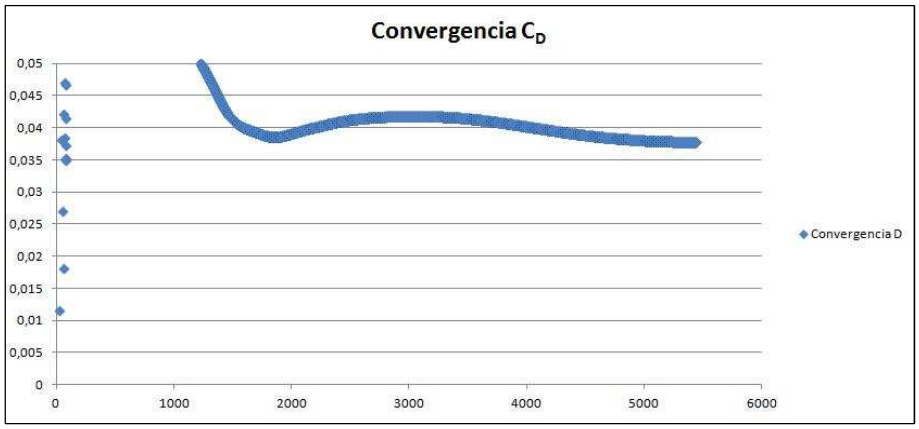
SIMPLE solution converged in 5444 iterations.

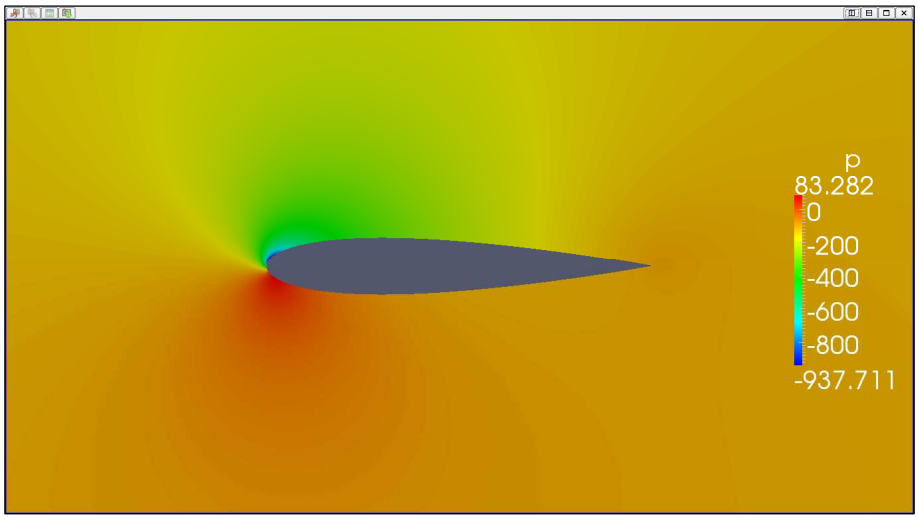
ForceCoeffs output:

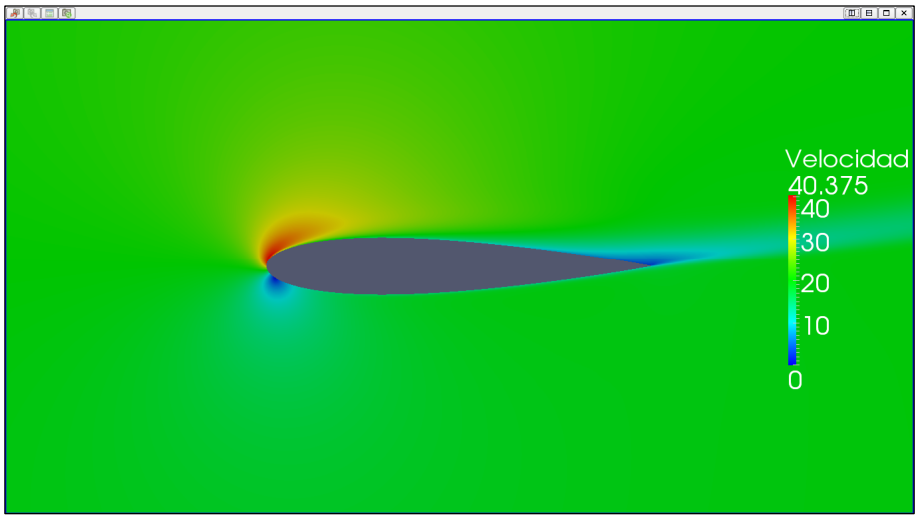
$C_L = 1{,}0567$

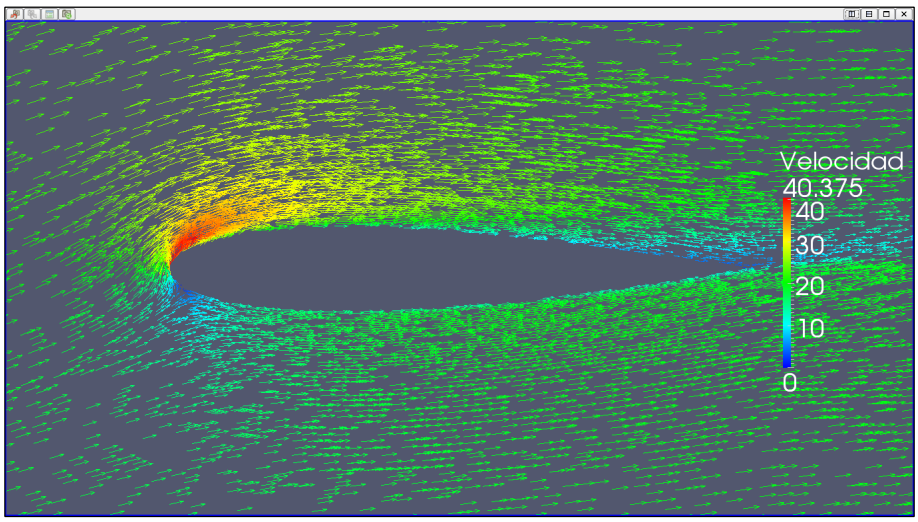
$C_D = 0{,}0377$






p
83.282
0
-200
-400
-600
-800
-937.711


Velocidad
40.375
40
30
20
10
0


Velocidad
40.375
40
30
20
10
0

II.2.14. Resultados ángulo de ataque 12º

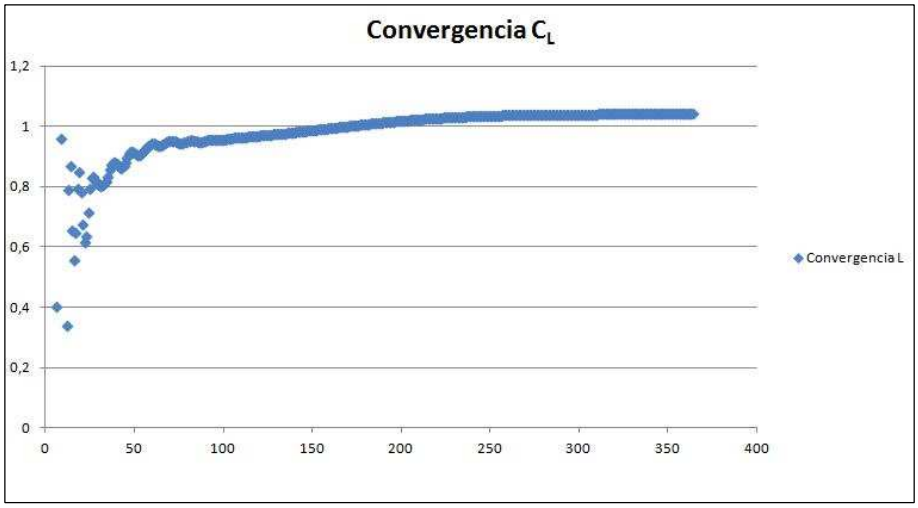
II.2.14.1. Caso 1 (Ángulo de ataque 12º)

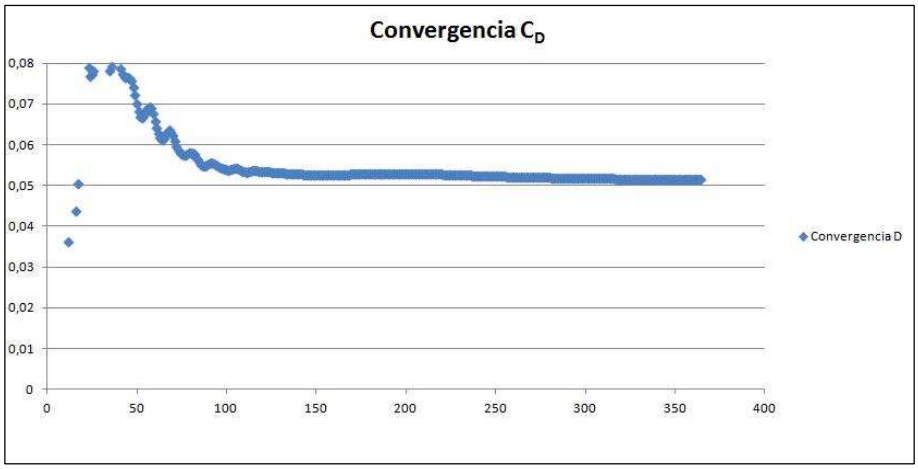
SIMPLE solution converged in 364 iterations.

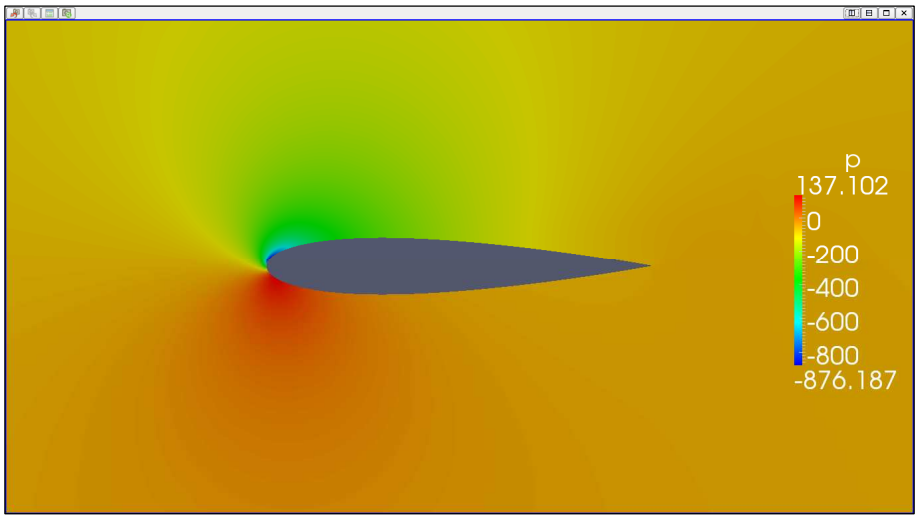
ForceCoeffs output:

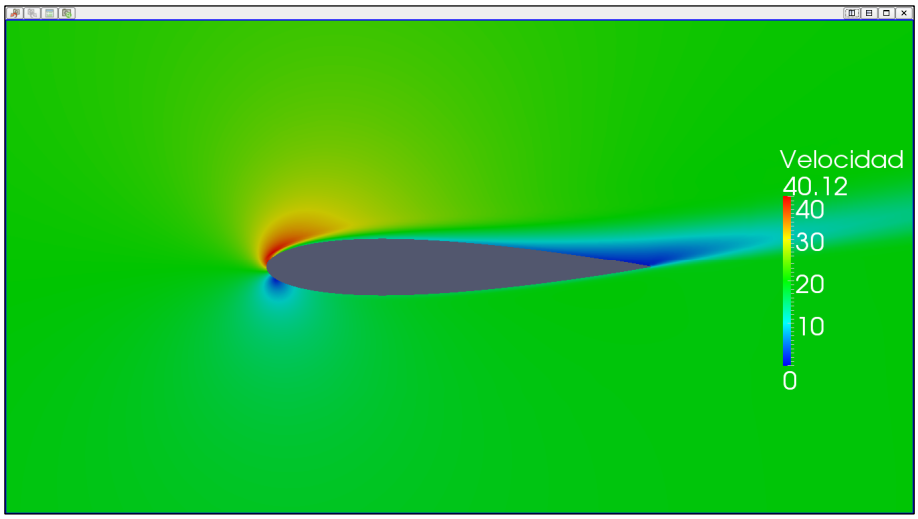
$C_L = 1{,}0407$

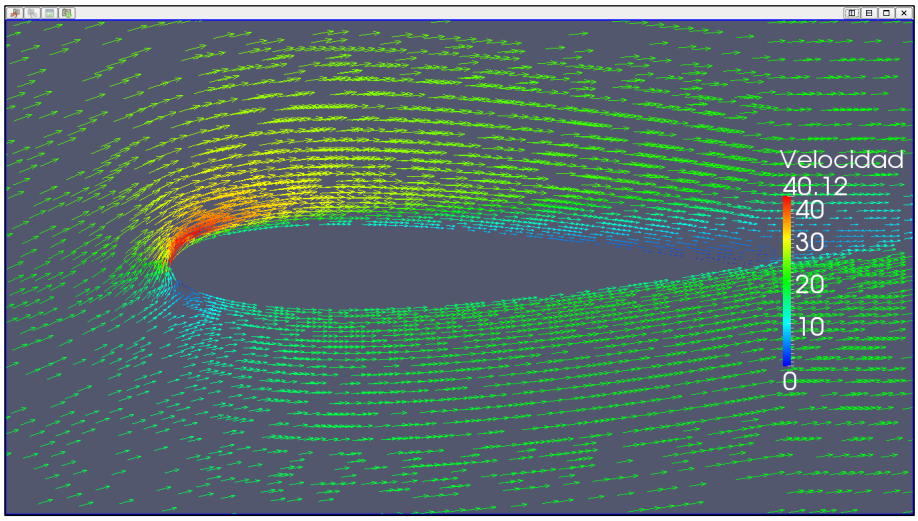
$C_D = 0{,}0516$






p
137.102
0
-200
-400
-600
-800
-876.187


Velocidad
40.12
40
30
20
10
0


Velocidad
40.12
40
30
20
10
0

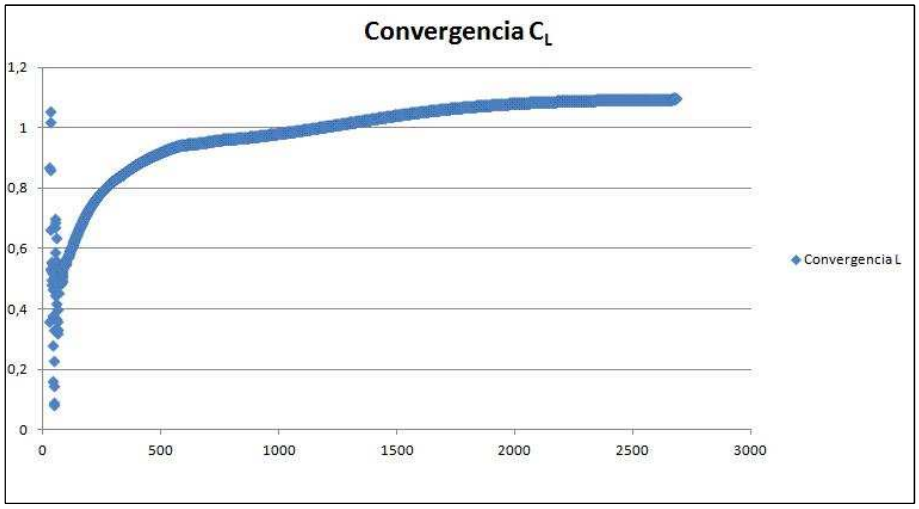
II.2.14.2. Caso 2 (Ángulo de ataque 12º)

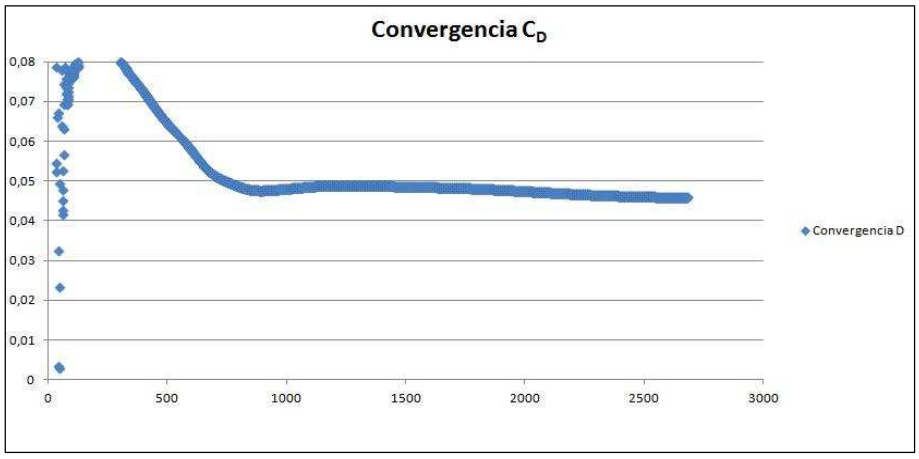
SIMPLE solution converged in 2684 iterations.

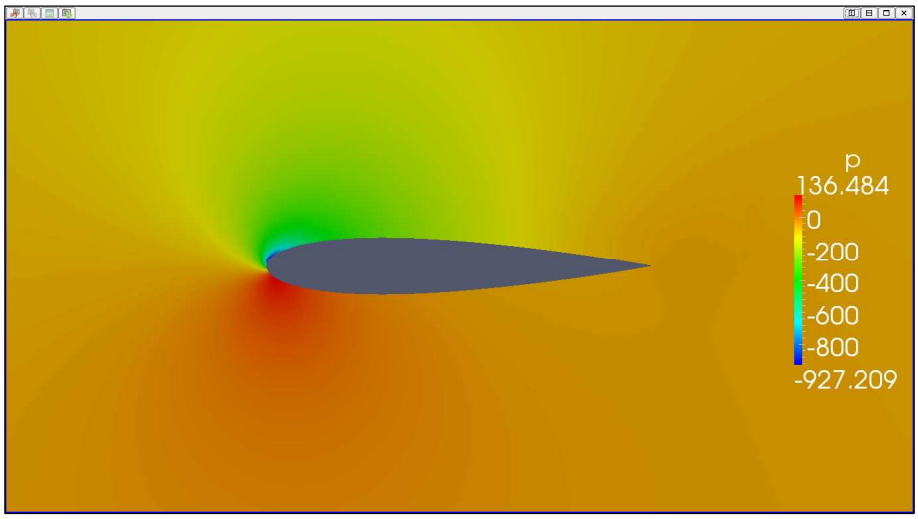
ForceCoeffs output:

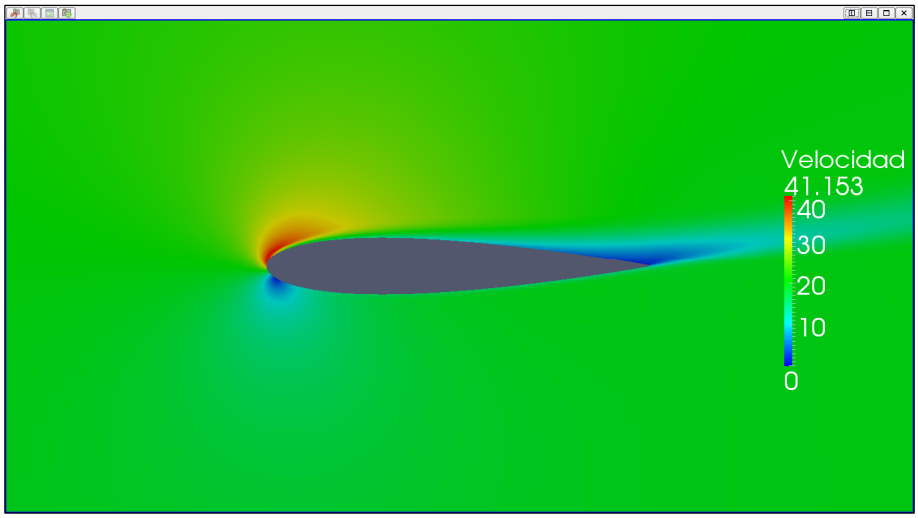
$C_L = 1{,}0957$

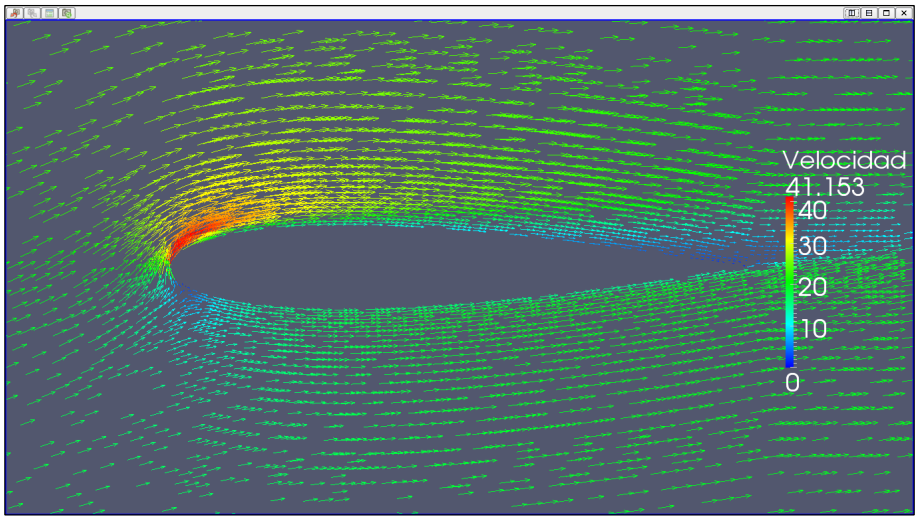
$C_D = 0{,}0458$






p
136.484
0
-200
-400
-600
-800
-927.209


Velocidad
41.153
40
30
20
10
0


Velocidad
41.153
40
30
20
10
0

II.2.14.3. Caso 3 (Ángulo de ataque 12º)

Time: 50000

Solving U_X: Initial residual $= 1{,}847 \cdot 10^{-07}$, Final residual $= 1{,}100 \cdot 10^{-08}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 1{,}513 \cdot 10^{-07}$, Final residual $= 1{,}064 \cdot 10^{-08}$, No Iterations $= 4$.

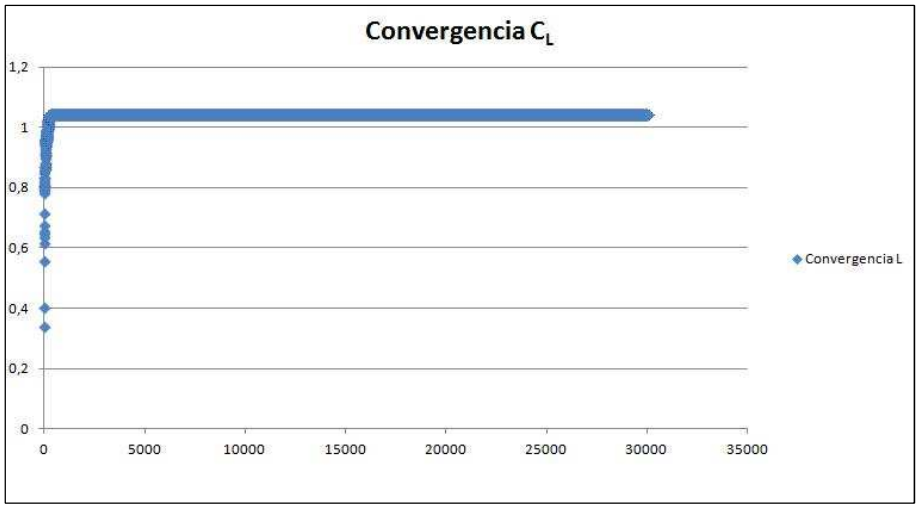
Solving p: Initial residual $= 1{,}024 \cdot 10^{-05}$, Final residual $= 8{,}255 \cdot 10^{-07}$, No Iterations $= 3$.

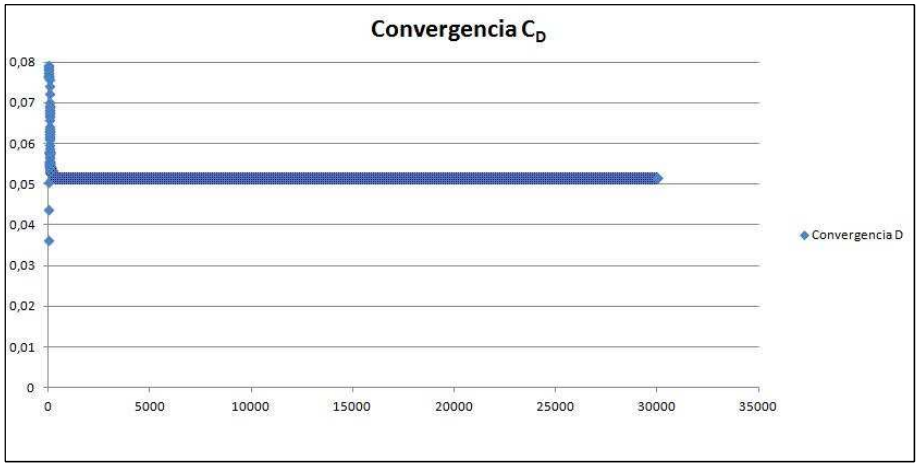
ExecutionTime: 5444,32 s. ClockTime: 5453 s.

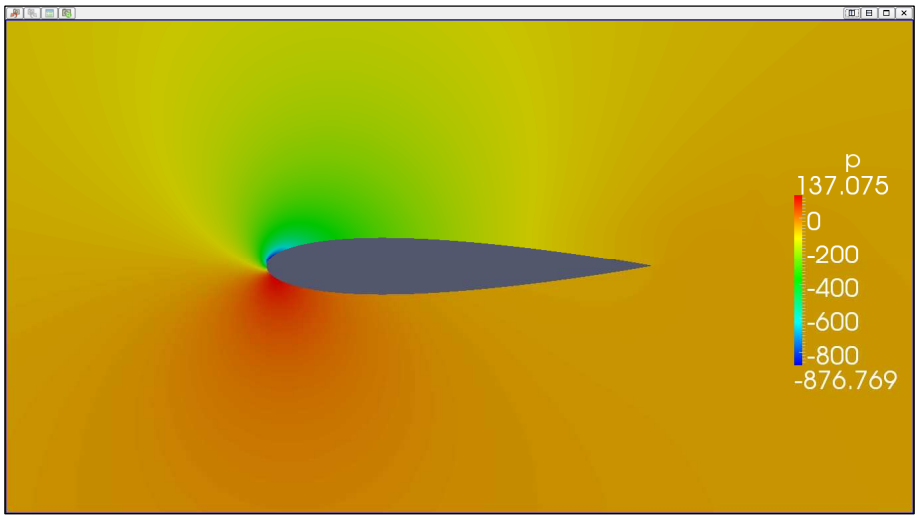
ForceCoeffs output:

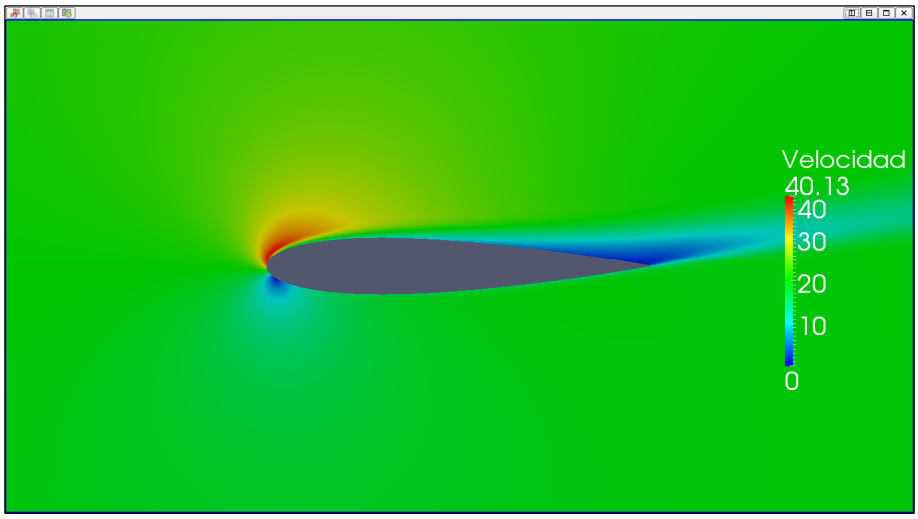
$C_L = 1{,}0415$

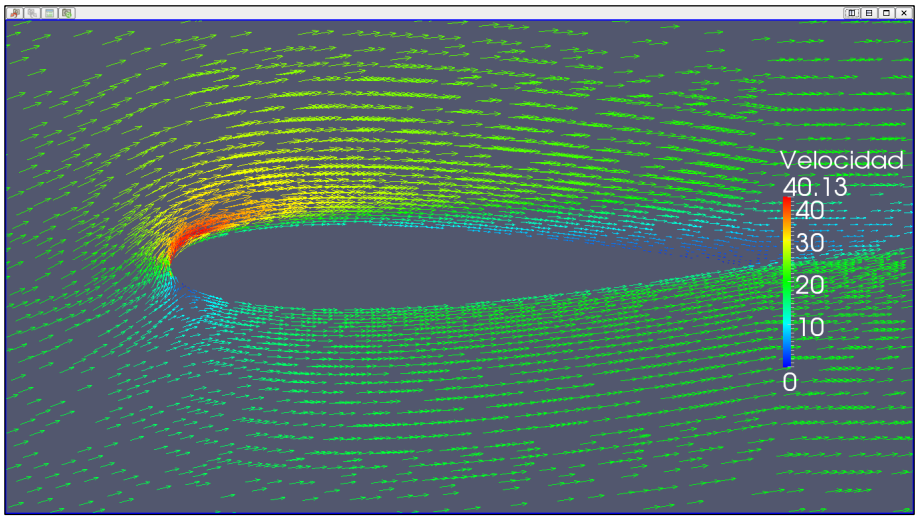
$C_D = 0{,}0516$






p
137.075
0
-200
-400
-600
-800
-876.769


Velocidad
40.13
40
30
20
10
0


Velocidad
40.13
40
30
20
10
0

II.2.15. Resultados ángulo de ataque 13º

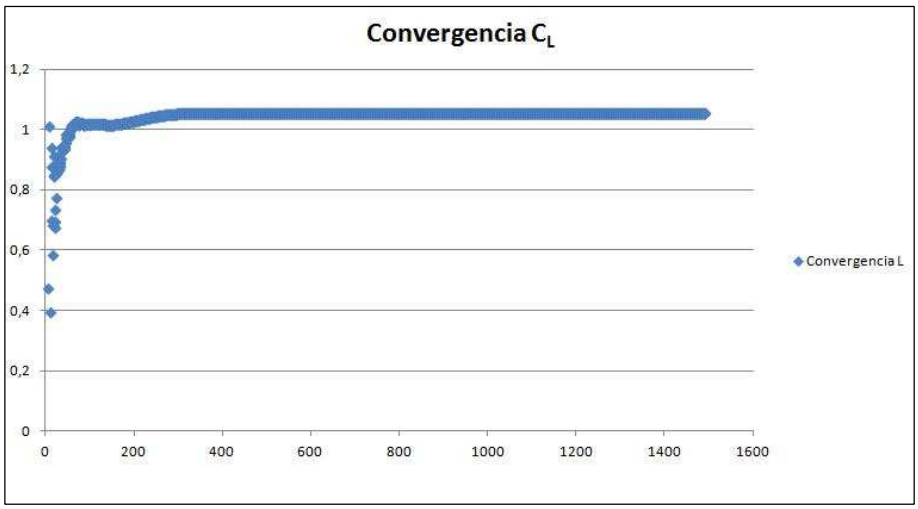
II.2.15.1. Caso 1 (Ángulo de ataque 13º)

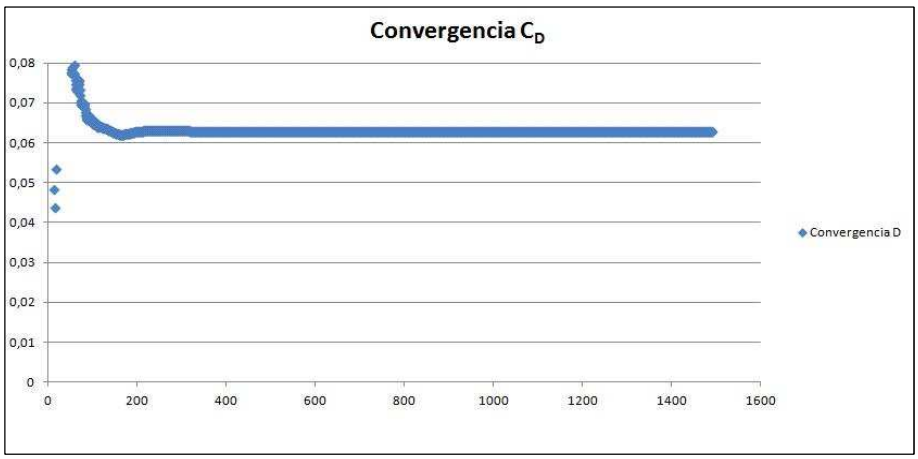
SIMPLE solution converged in 1492 iterations.

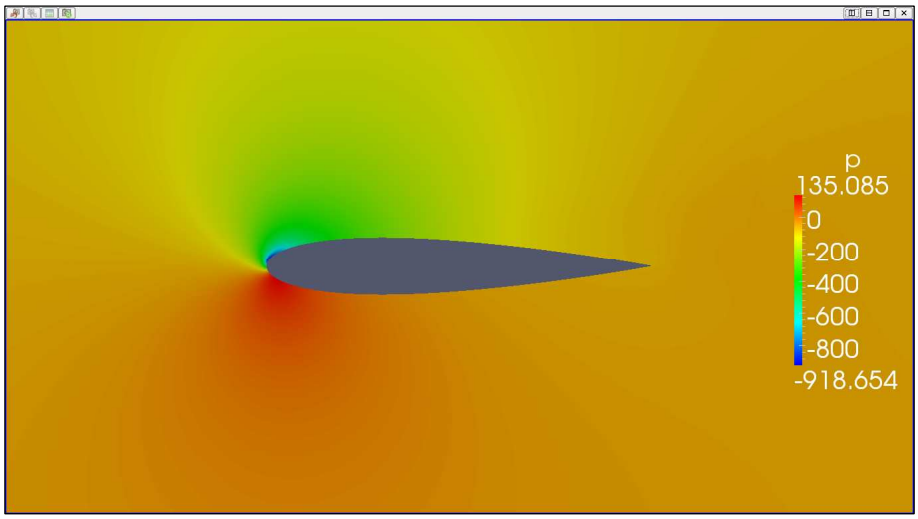
ForceCoeffs output:

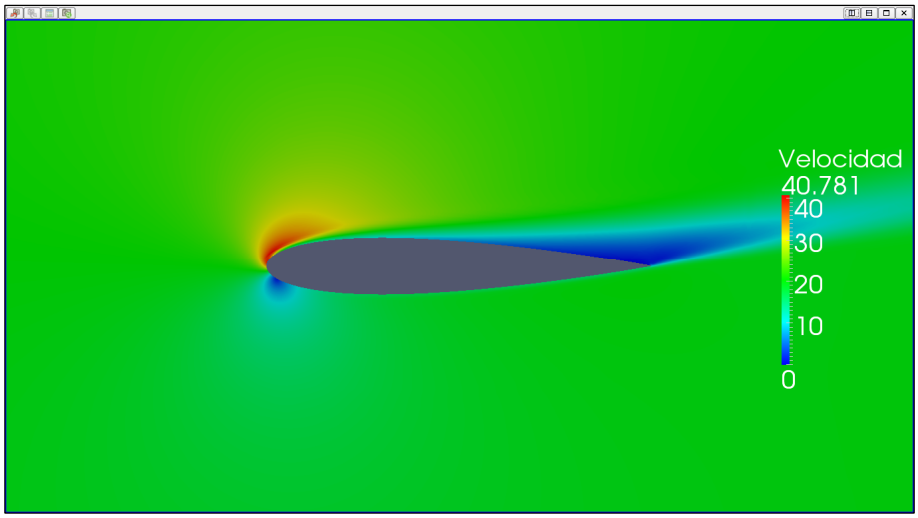
$C_L = 1{,}0526$

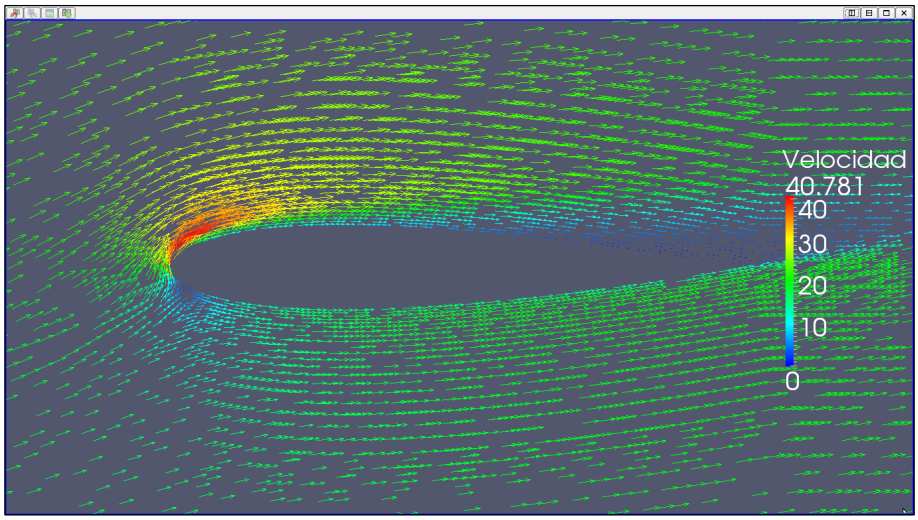
$C_D = 0{,}0627$






p
135.085
0
-200
-400
-600
-800
-918.654


Velocidad
40.781
40
30
20
10
0


Velocidad
40.781
40
30
20
10
0

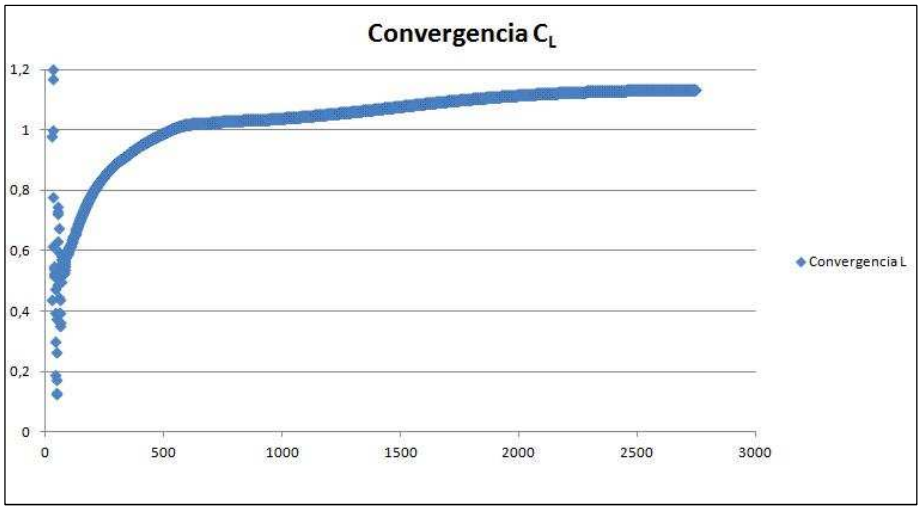
II.2.15.2. Caso 2 (Ángulo de ataque 13º)

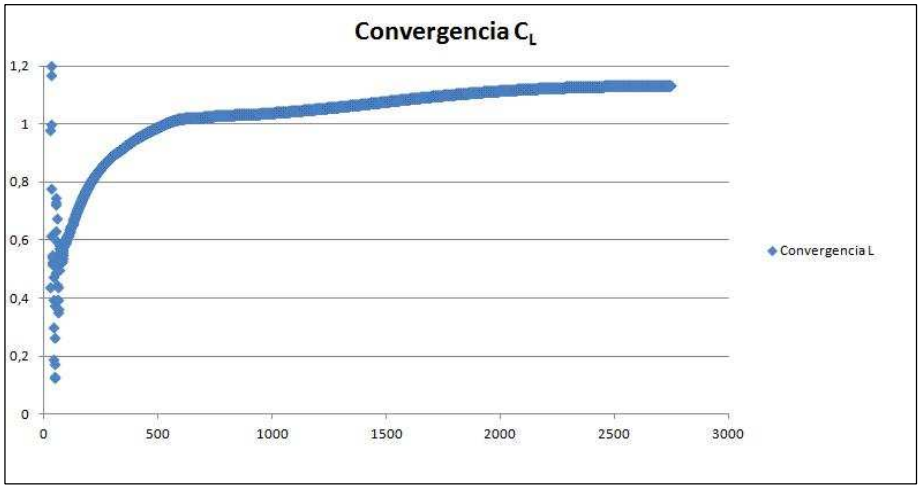
SIMPLE solution converged in 2747 iterations.

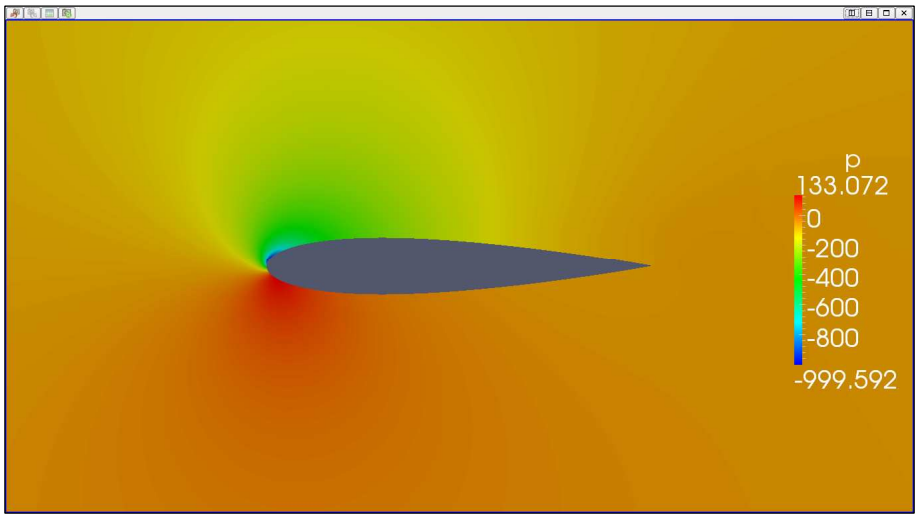
ForceCoeffs output:

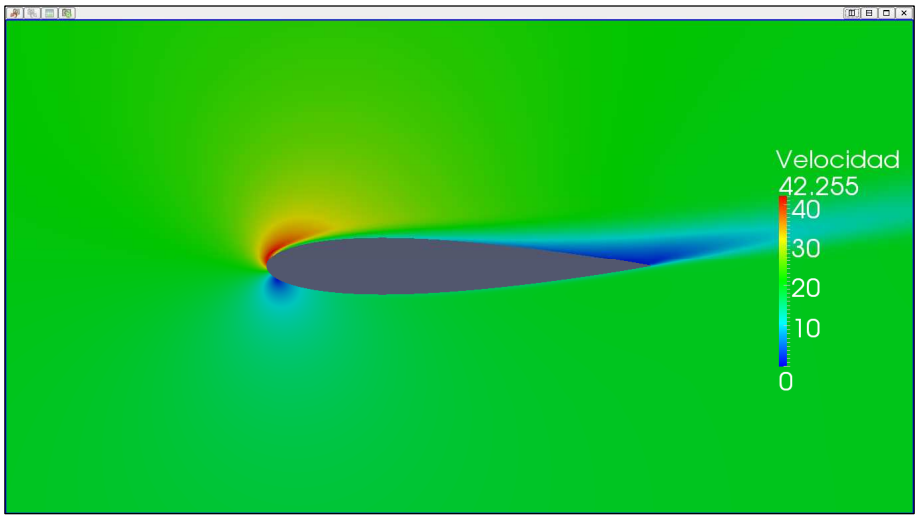
$C_L = 1{,}1343$

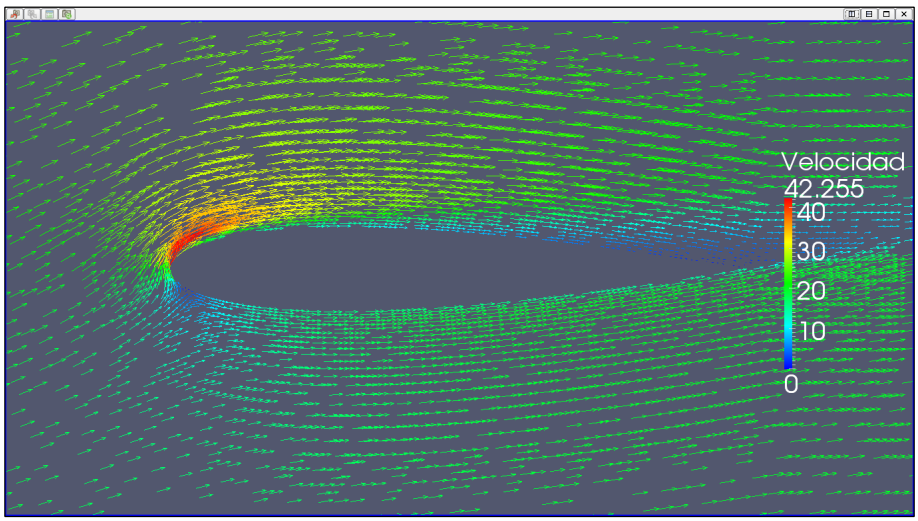
$C_D = 0{,}0548$






p
133.072
0
-200
-400
-600
-800
-999.592


Velocidad
42.255
40
30
20
10
0


Velocidad
42.255
40
30
20
10
0

II.2.15.3. Caso 3 (Ángulo de ataque 13º)

Time: 50000

Solving U_X: Initial residual $= 9{,}149 \cdot 10^{-06}$, Final residual $= 2{,}847 \cdot 10^{-07}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 1{,}168 \cdot 10^{-05}$, Final residual $= 3{,}607 \cdot 10^{-07}$, No Iterations $= 4$.

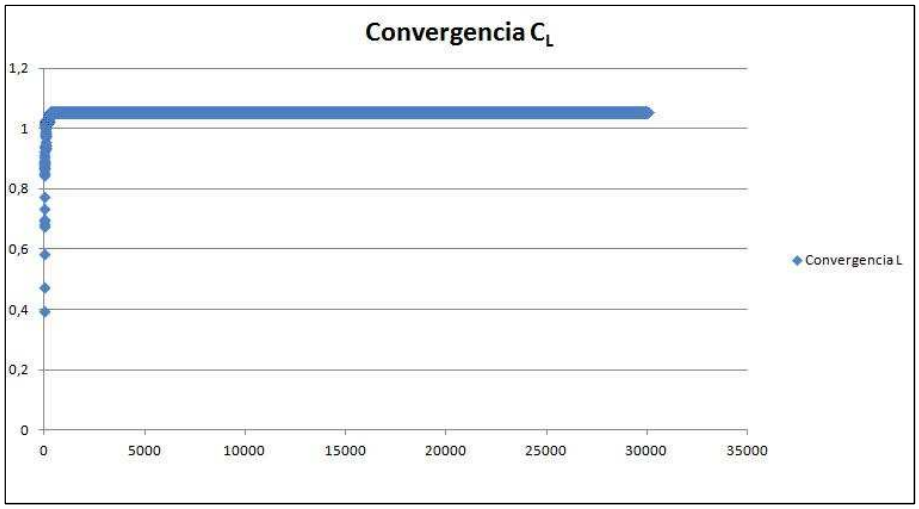
Solving p: Initial residual $= 9{,}563 \cdot 10^{-04}$, Final residual $= 4{,}327 \cdot 10^{-05}$, No Iterations $= 3$.

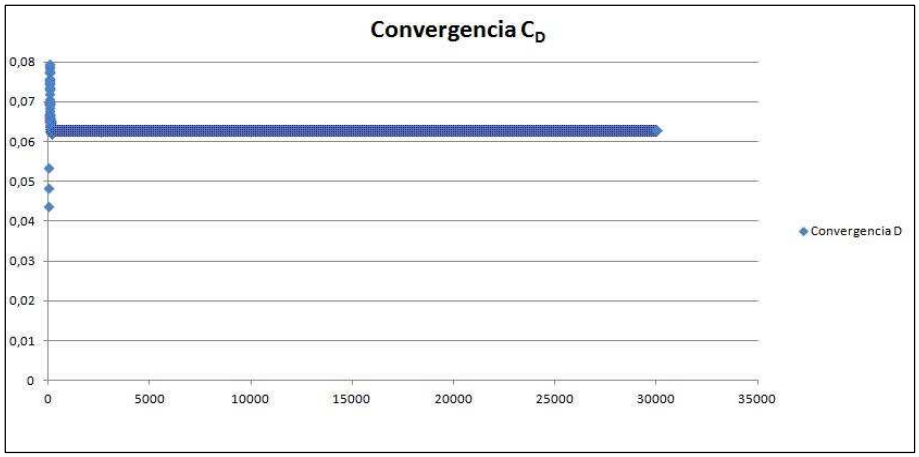
ExecutionTime: 5116,67 s. ClockTime: 5124 s.

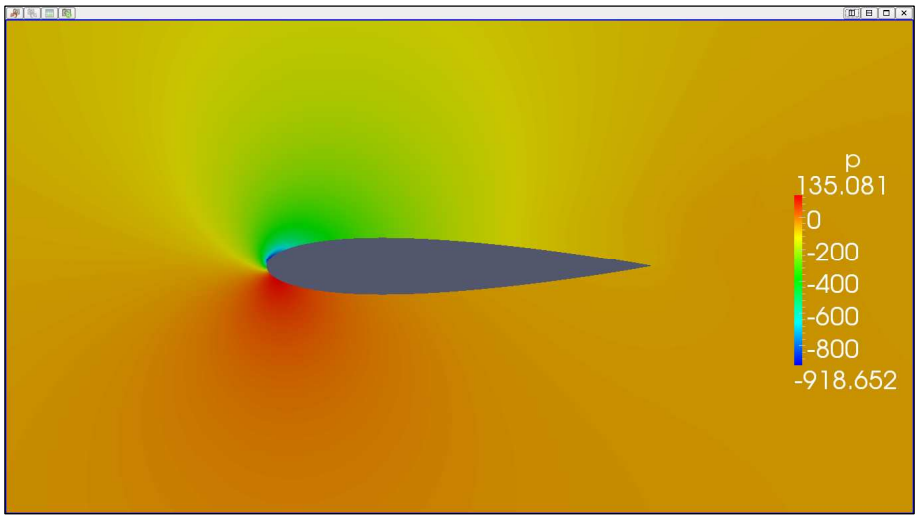
ForceCoeffs output:

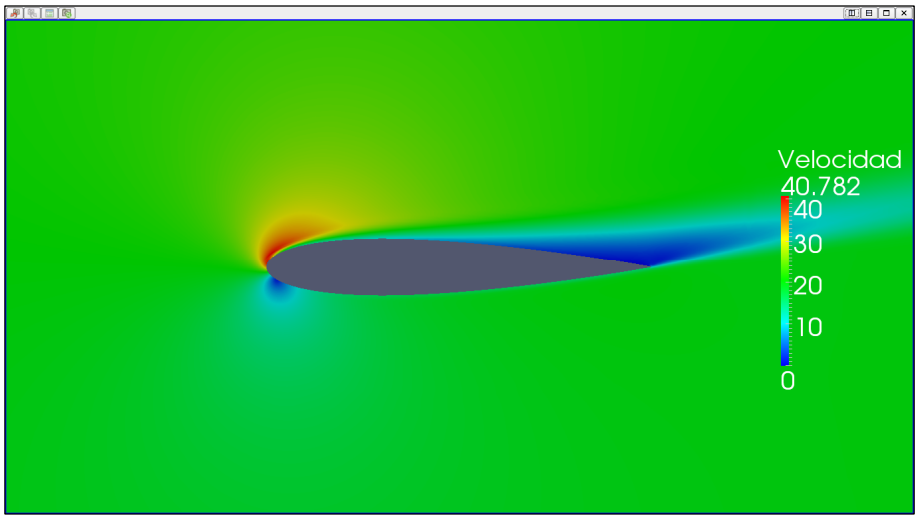
$C_L = 1{,}0525$

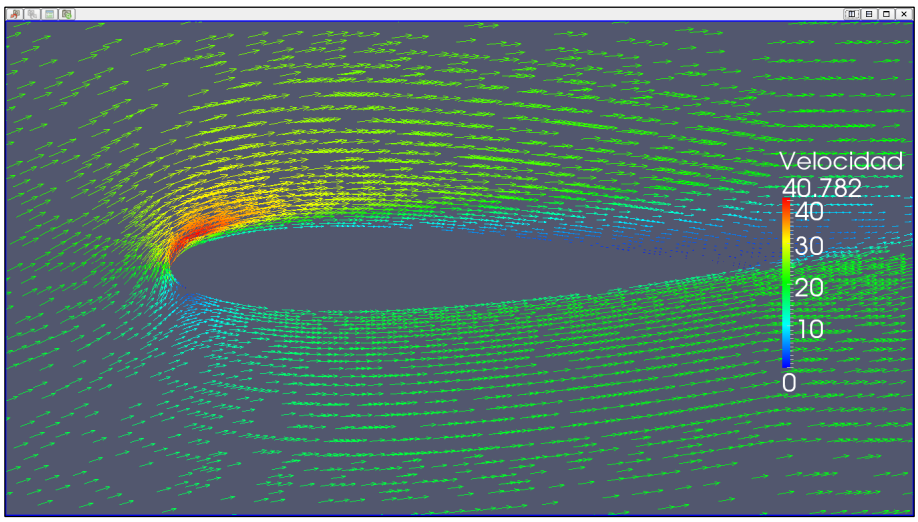
$C_D = 0{,}0627$






p
135.081
0
-200
-400
-600
-800
-918.652


Velocidad
40.782
40
30
20
10
0


Velocidad
40.782
40
30
20
10
0

II.2.16. Resultados ángulo de ataque 14º

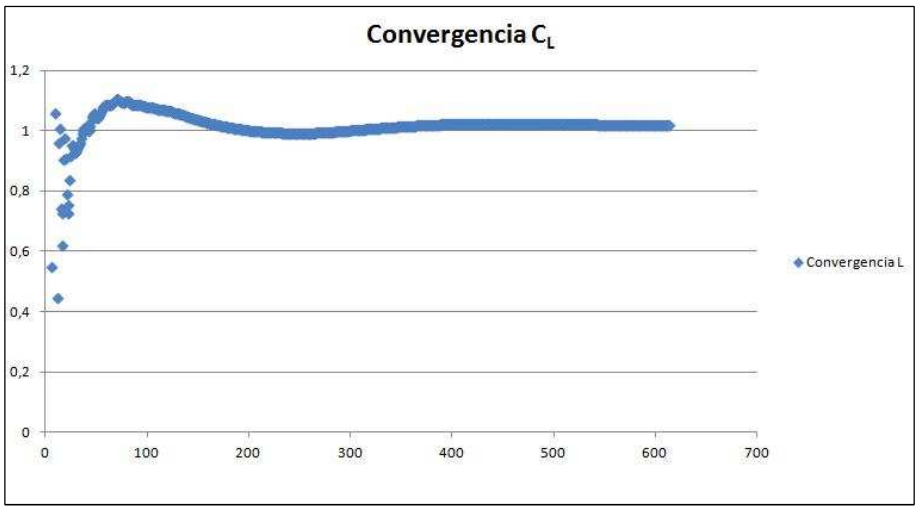
II.2.16.1. Caso 1 (Ángulo de ataque 14º)

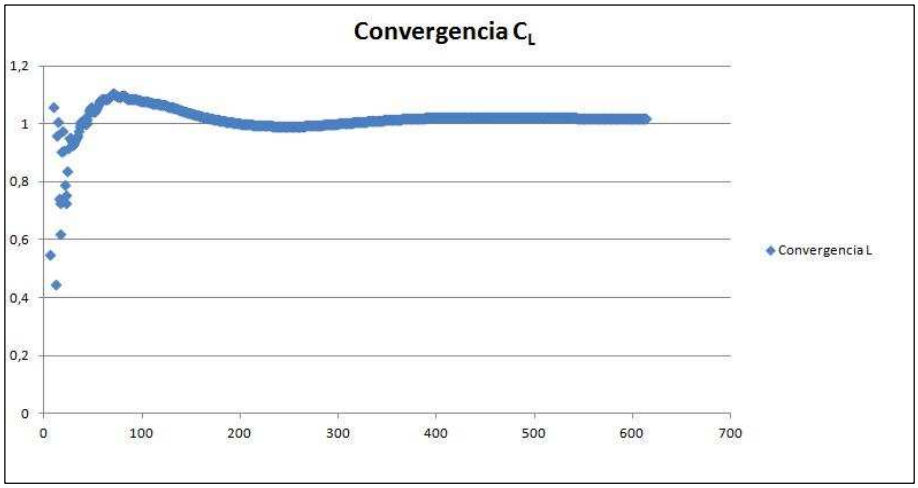
SIMPLE solution converged in 614 iterations.

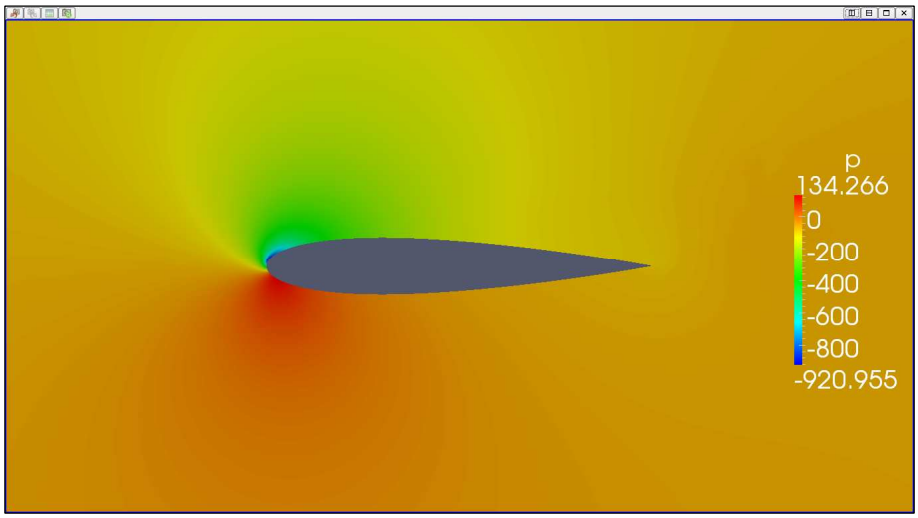
ForceCoeffs output:

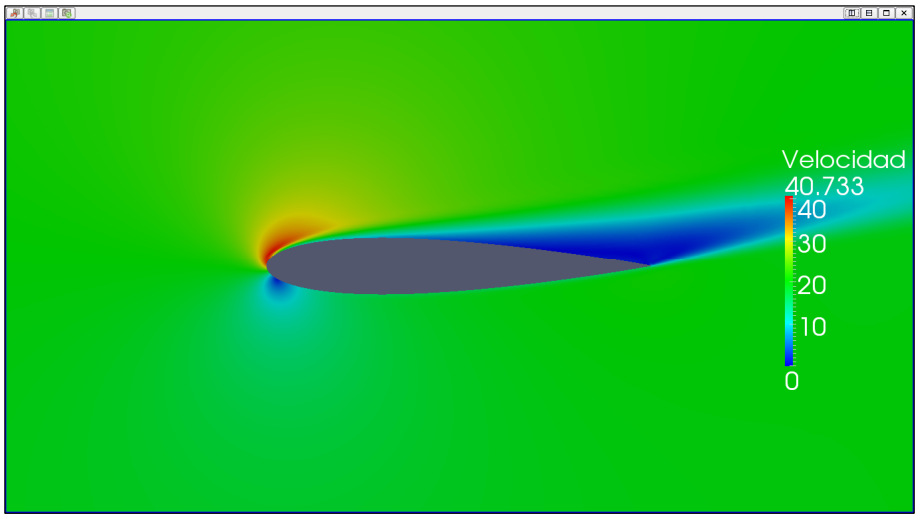
$C_L = 1{,}0188$

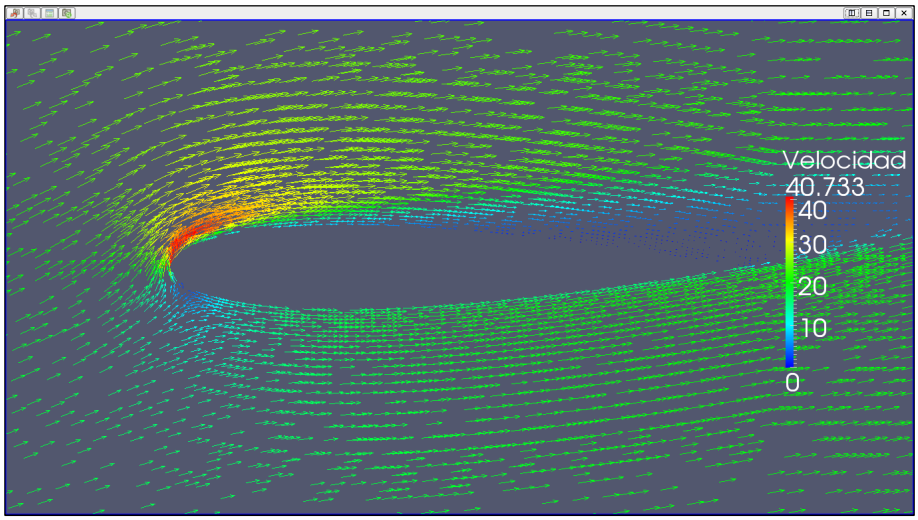
$C_D = 0{,}0780$






p
134.266
0
-200
-400
-600
-800
-920.955


Velocidad
40.733
40
30
20
10
0


Velocidad
40.733
40
30
20
10
0

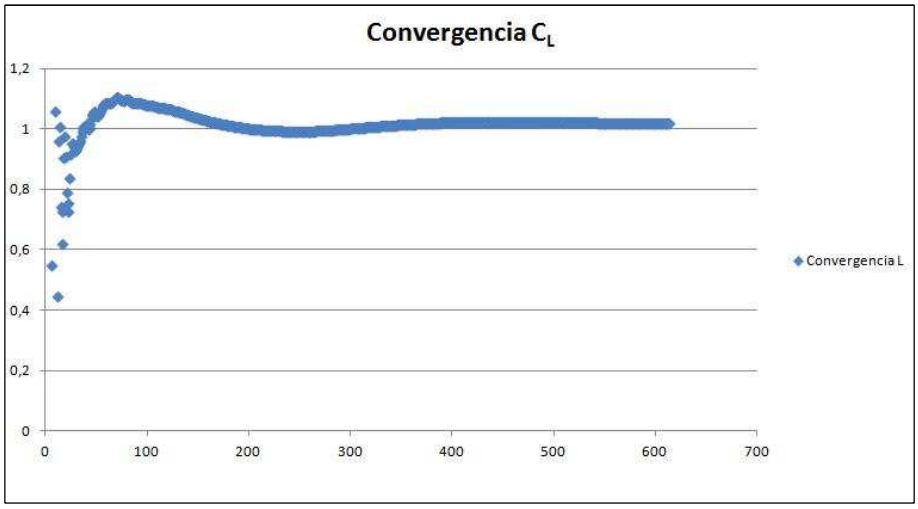
II.2.16.2. Caso 2 (Ángulo de ataque 14º)

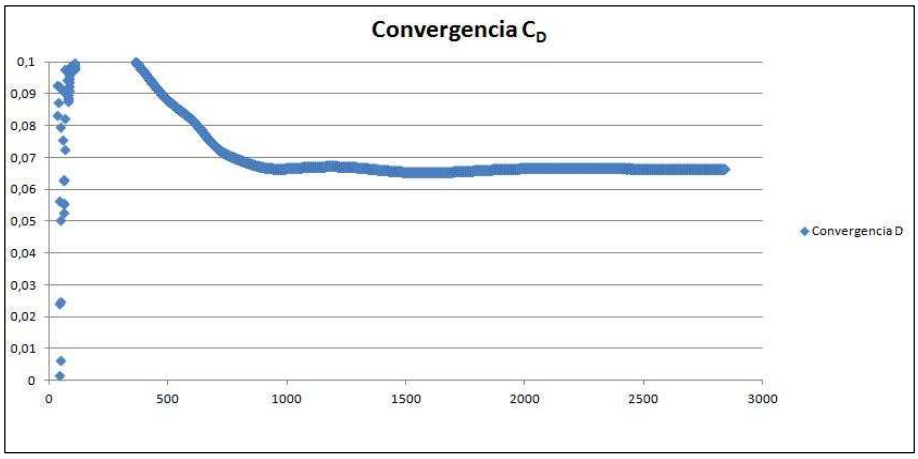
SIMPLE solution converged in 2840 iterations.

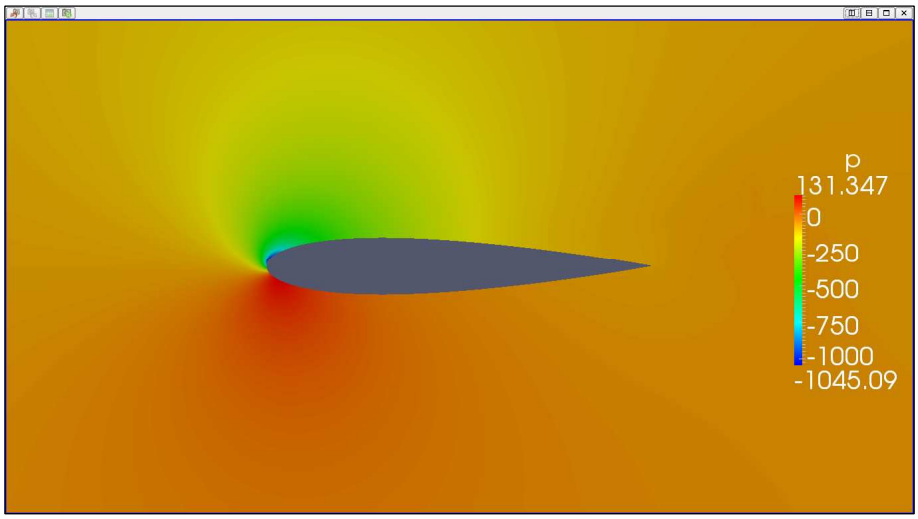
ForceCoeffs output:

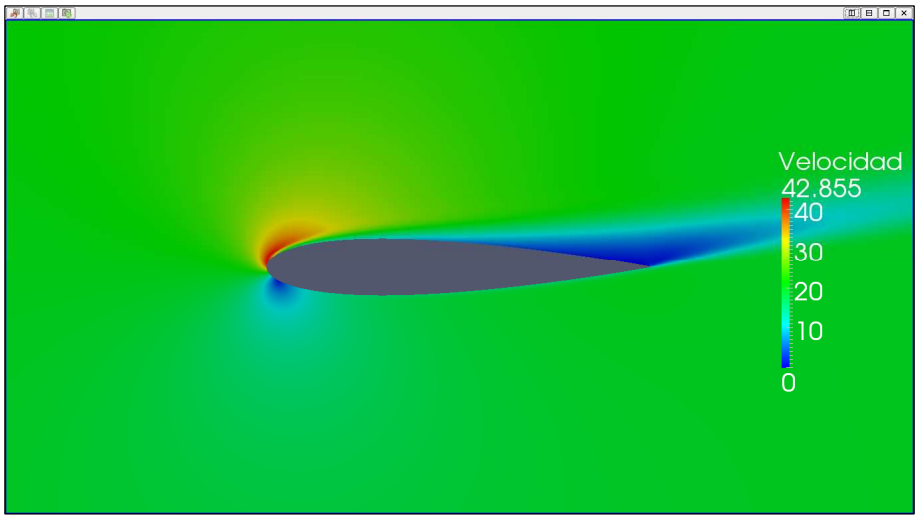
$C_L = 1{,}1405$

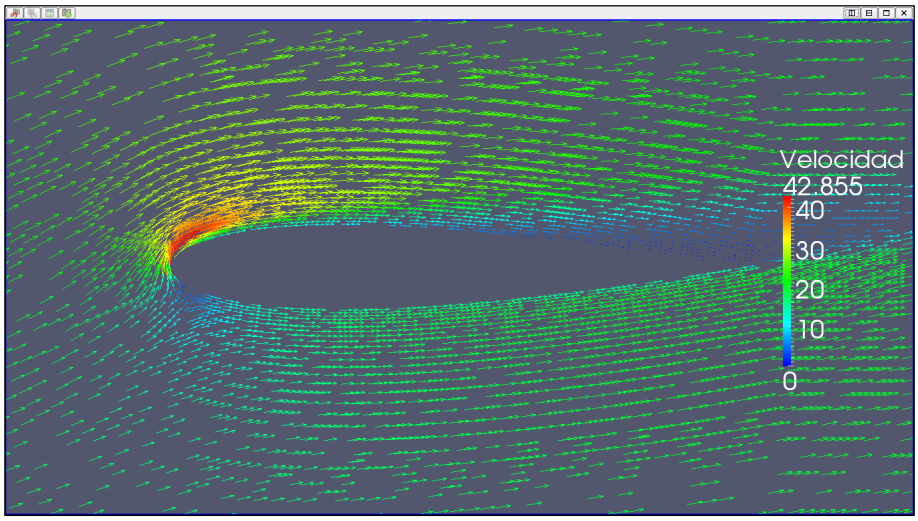
$C_D = 0{,}0665$






p
131.347
0
-250
-500
-750
-1000
-1045.09


Velocidad
42.855
40
30
20
10
0


Velocidad
42.855
40
30
20
10
0

II.2.16.3. Caso 3 (Ángulo de ataque 14º)

Time: 50000

Solving U_X: Initial residual $= 2{,}896 \cdot 10^{-07}$, Final residual $= 1{,}704 \cdot 10^{-08}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 3{,}416 \cdot 10^{-07}$, Final residual $= 1{,}936 \cdot 10^{-08}$, No Iterations $= 4$.

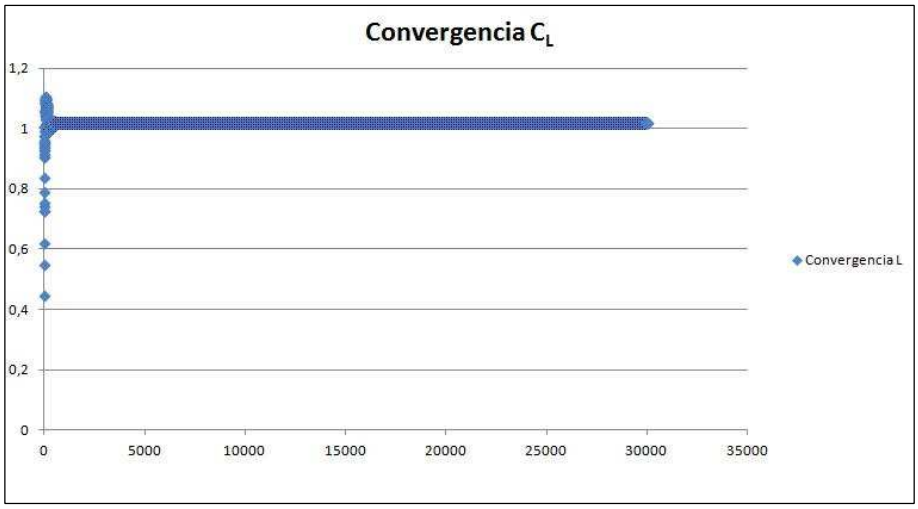
Solving p: Initial residual $= 1{,}561 \cdot 10^{-05}$, Final residual $= 1{,}309 \cdot 10^{-06}$, No Iterations $= 3$.

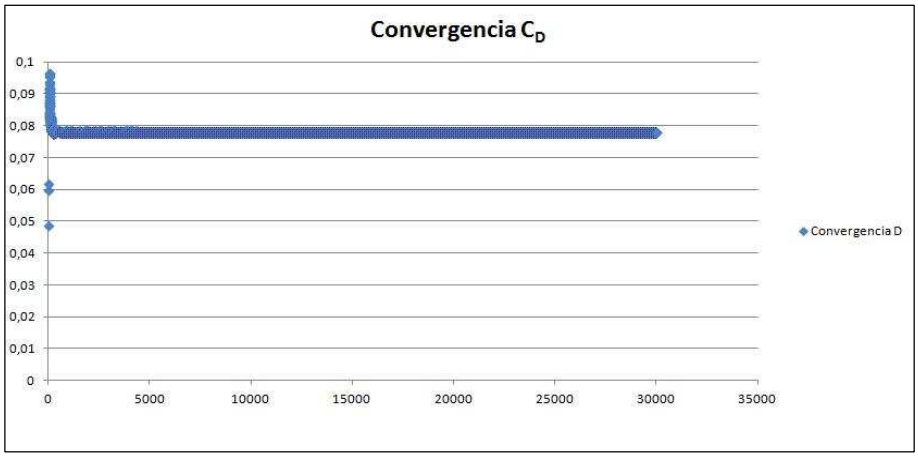
ExecutionTime: 5196,04 s. ClockTime: 5206 s.

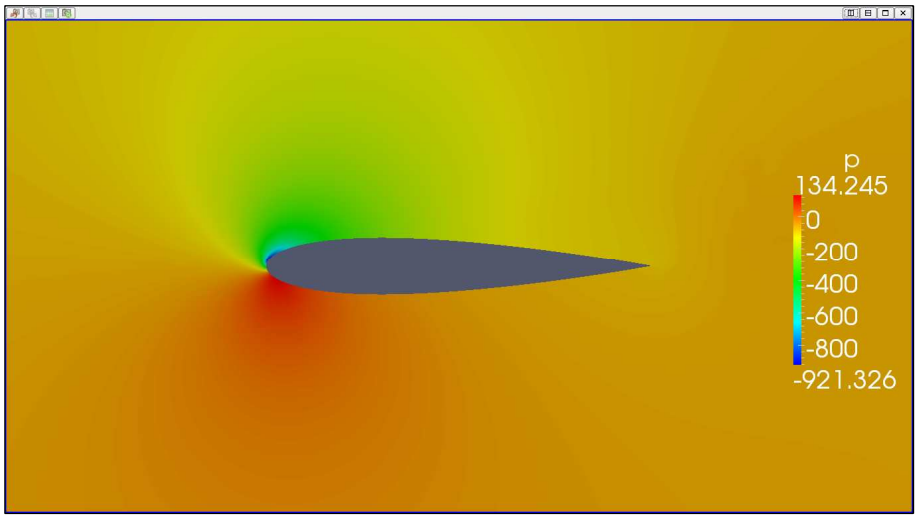
ForceCoeffs output:

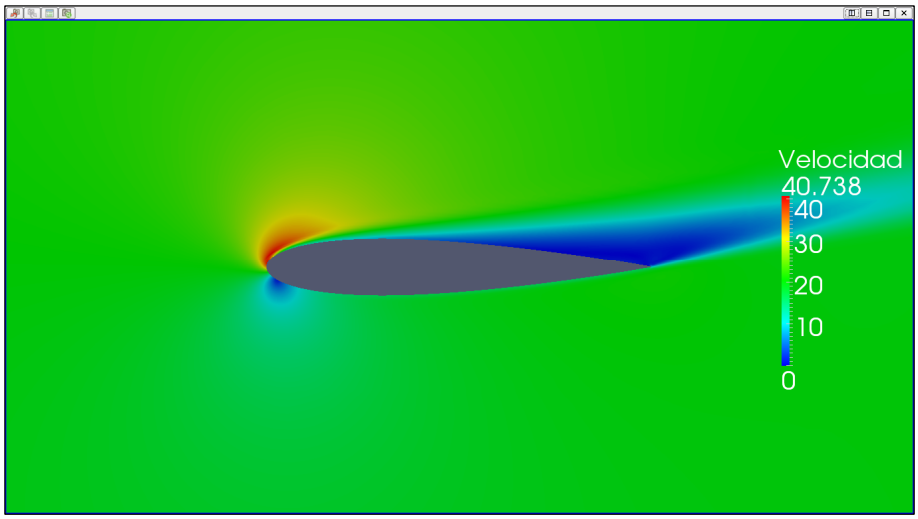
$C_L = 1{,}0194$

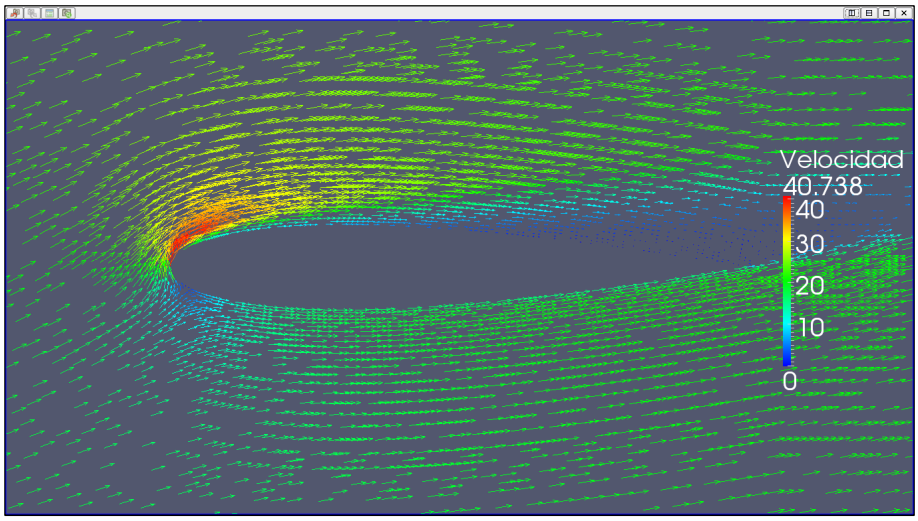
$C_D = 0{,}0780$






p
134.245
0
-200
-400
-600
-800
-921.326


Velocidad
40.738
40
30
20
10
0


Velocidad
40.738
40
30
20
10
0

II.2.17. Resultados ángulo de ataque 15º

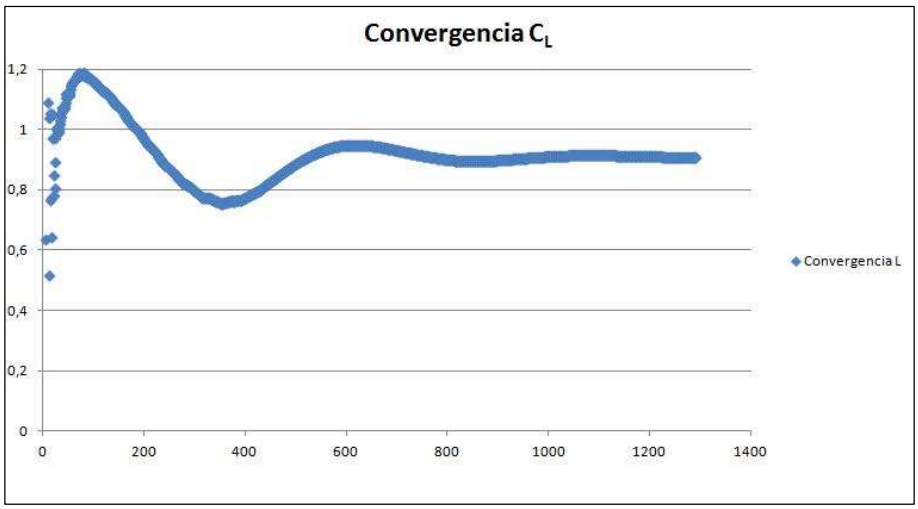
II.2.17.1. Caso 1 (Ángulo de ataque 15º)

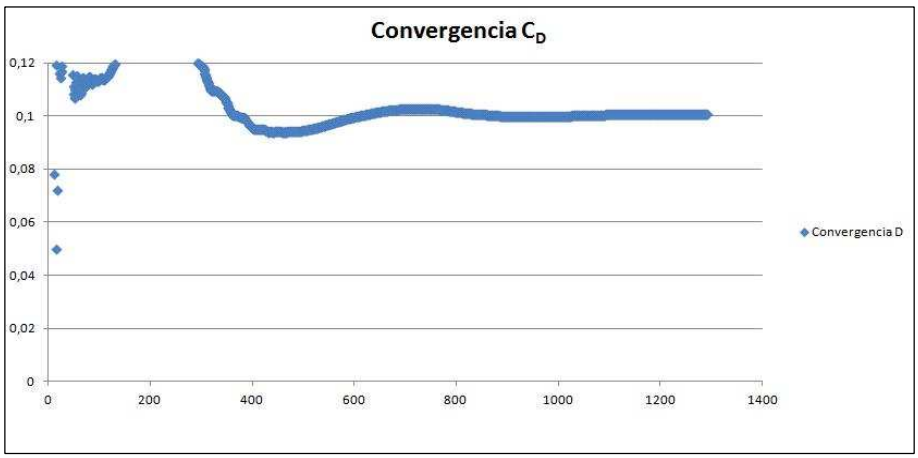
SIMPLE solution converged in 1292 iterations.

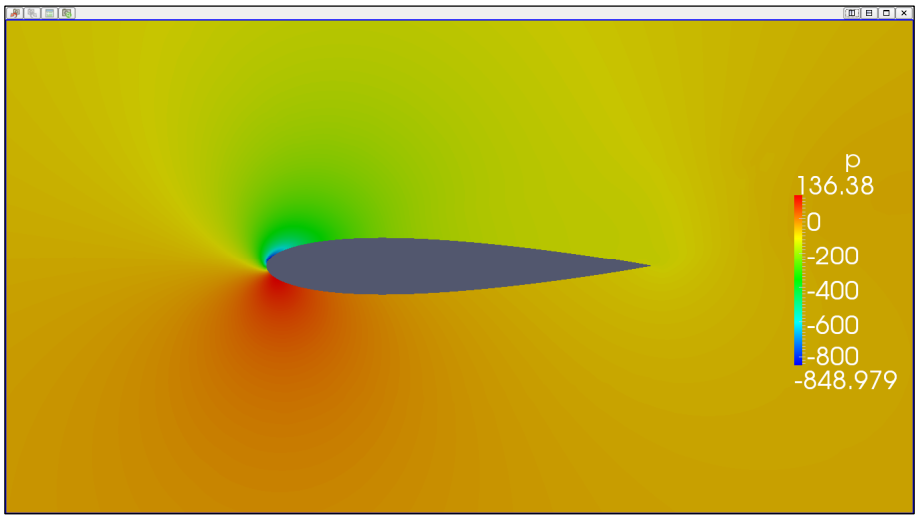
ForceCoeffs output:

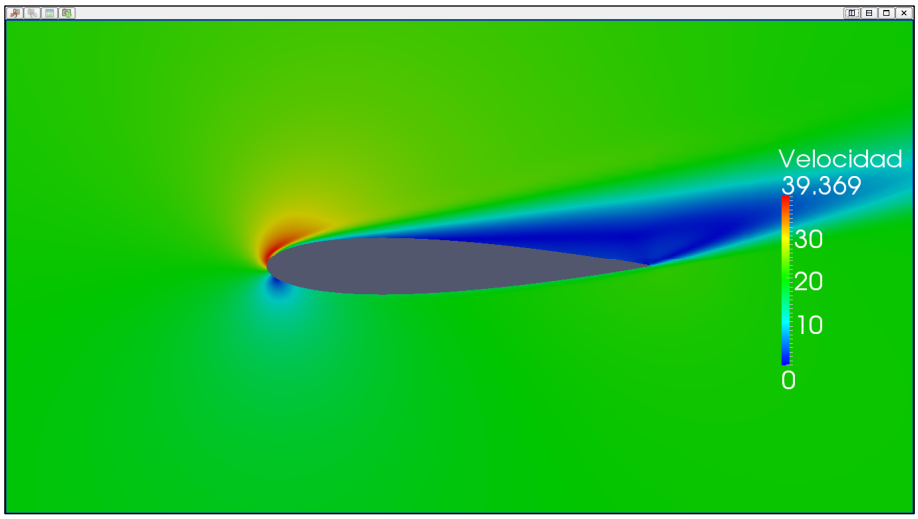
$C_L = 0{,}9081$

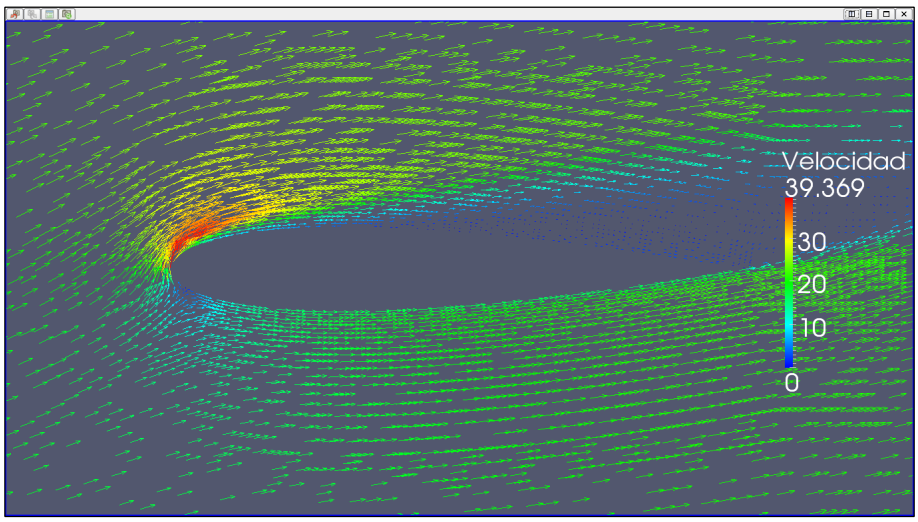
$C_D = 0{,}1005$






p
136.38
0
-200
-400
-600
-800
-848.979


Velocidad
39.369
30
20
10
0


Velocidad
39.369
30
20
10
0

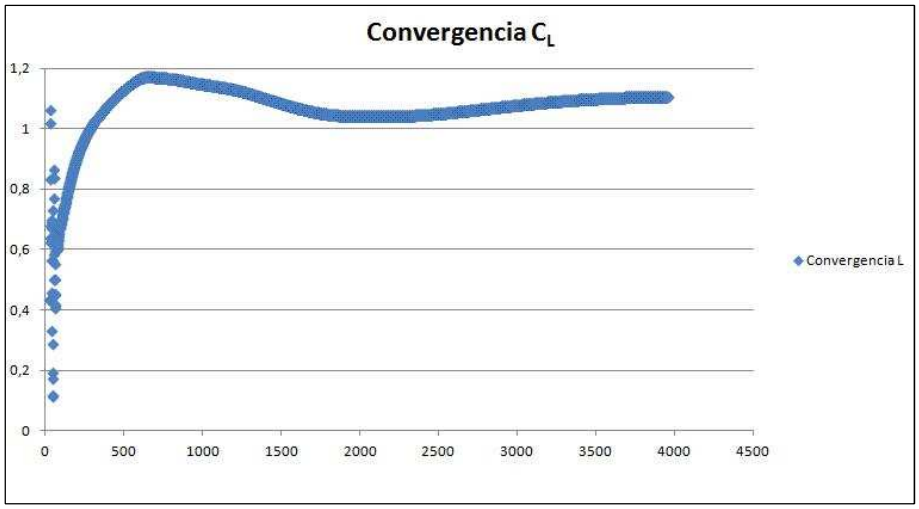
II.2.17.2. Caso 2 (Ángulo de ataque 15º)

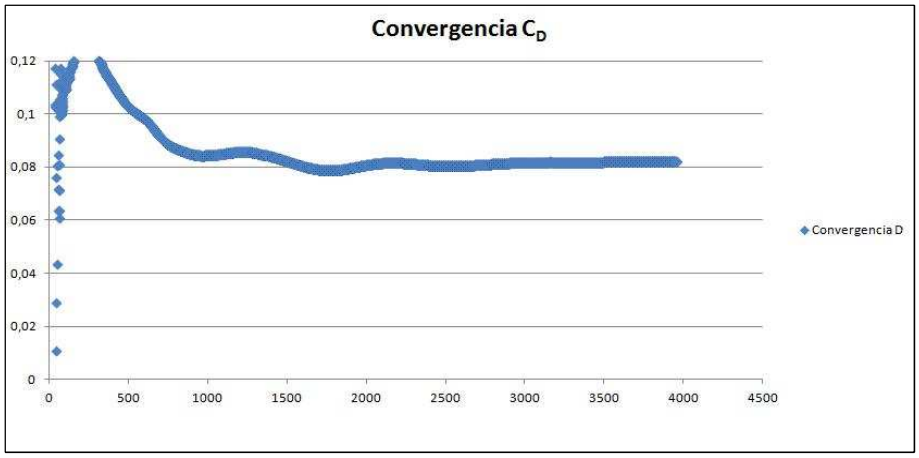
SIMPLE solution converged in 3958 iterations.

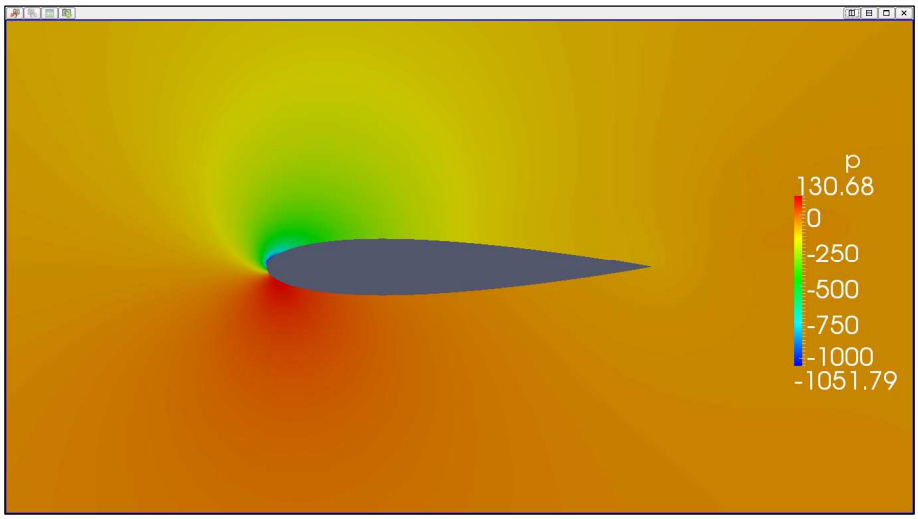
ForceCoeffs output:

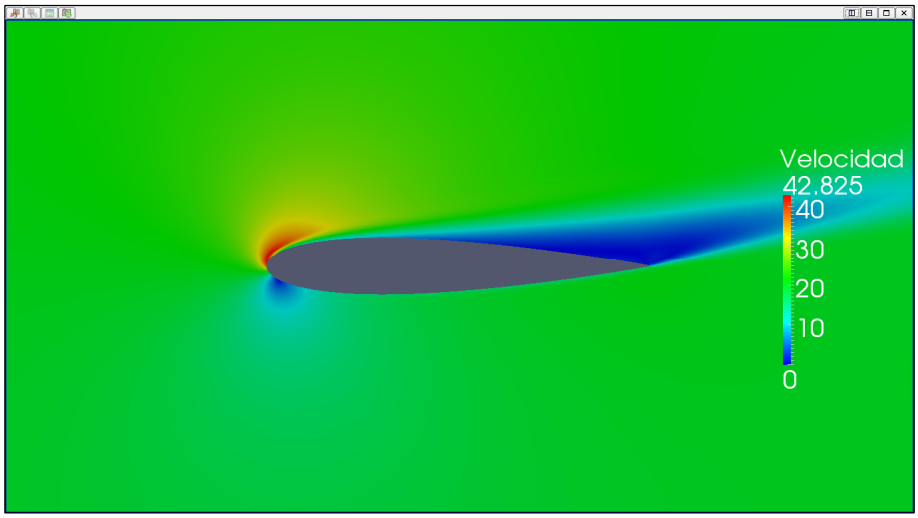
$C_L = 1{,}1064$

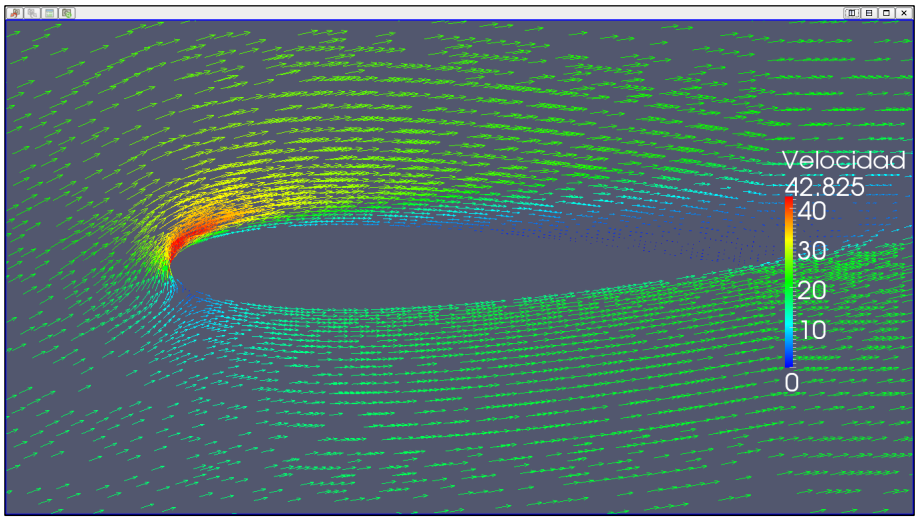
$C_D = 0{,}0822$






p
130.68
0
-250
-500
-750
-1000
-1051.79


Velocidad
42.825
40
30
20
10
0


Velocidad
42.825
40
30
20
10
0

II.2.17.3. Caso 3 (Ángulo de ataque 15º)

Time: 50000

Solving U_X: Initial residual $= 2{,}181 \cdot 10^{-07}$, Final residual $= 1{,}454 \cdot 10^{-08}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 3{,}719 \cdot 10^{-07}$, Final residual $= 3{,}056 \cdot 10^{-08}$, No Iterations $= 4$.

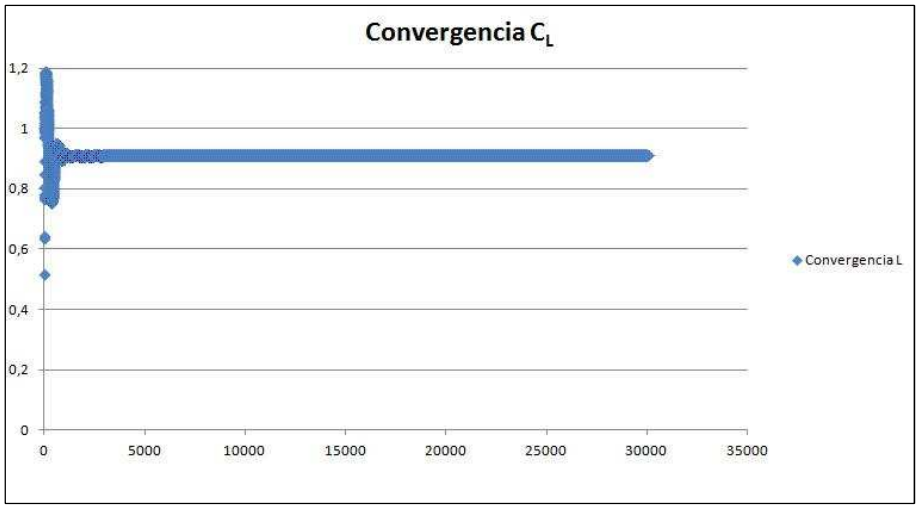
Solving p: Initial residual $= 1{,}938 \cdot 10^{-05}$, Final residual $= 1{,}800 \cdot 10^{-06}$, No Iterations $= 2$.

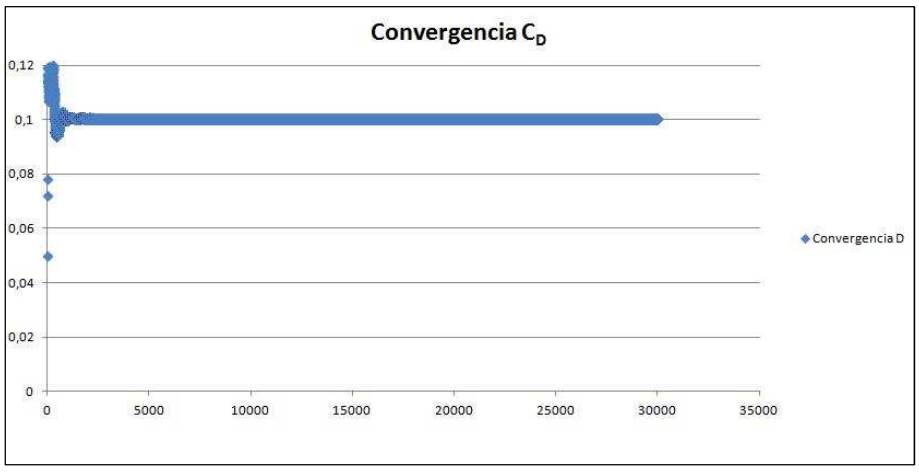
ExecutionTime: 5141,17 s. ClockTime: 5152 s.

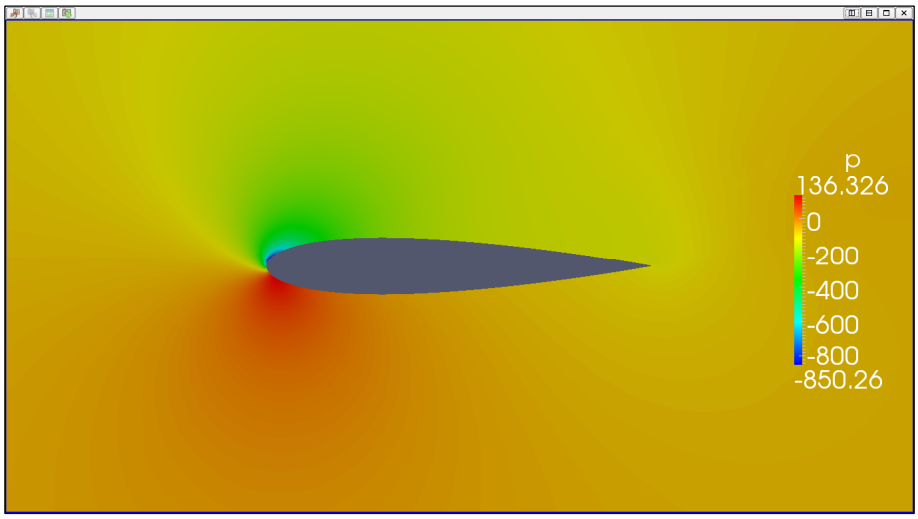
ForceCoeffs output:

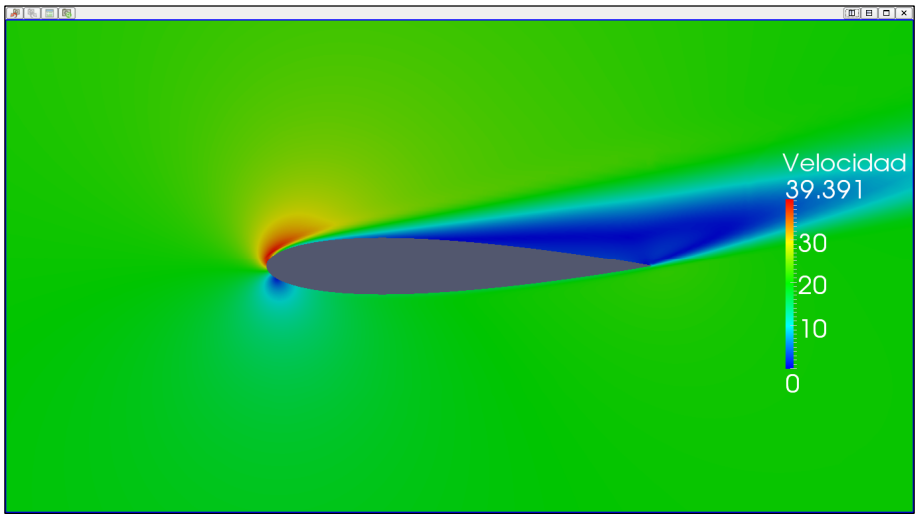
$C_L = 0{,}9098$

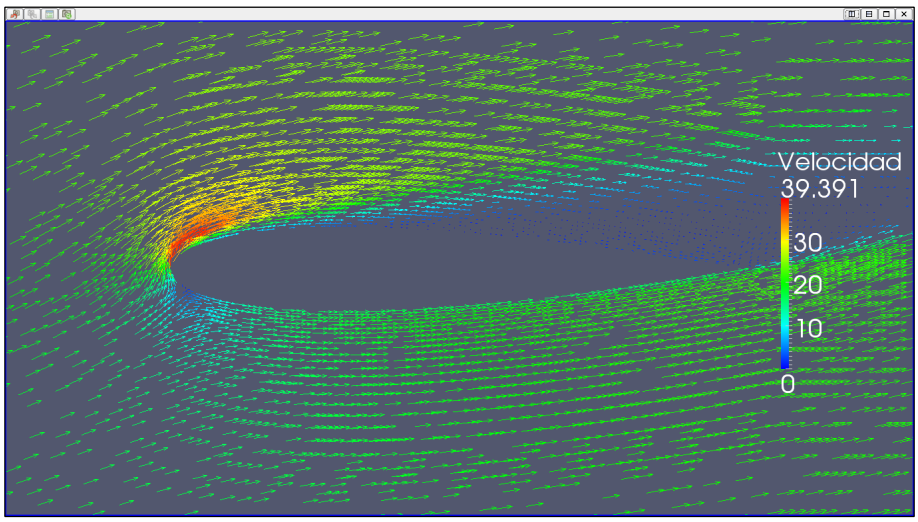
$C_D = 0{,}1005$






p
136.326
0
-200
-400
-600
-800
-850.26


Velocidad
39.391
30
20
10
0


Velocidad
39.391
30
20
10
0

II.2.18. Resultados ángulo de ataque 16º

II.2.18.1. Caso 1 (Ángulo de ataque 16º)

Time: 30000

Solving U_X: Initial residual $= 4{,}053 \cdot 10^{-03}$, Final residual $= 1{,}338 \cdot 10^{-04}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 8{,}742 \cdot 10^{-03}$, Final residual $= 3{,}694 \cdot 10^{-04}$, No Iterations $= 4$.

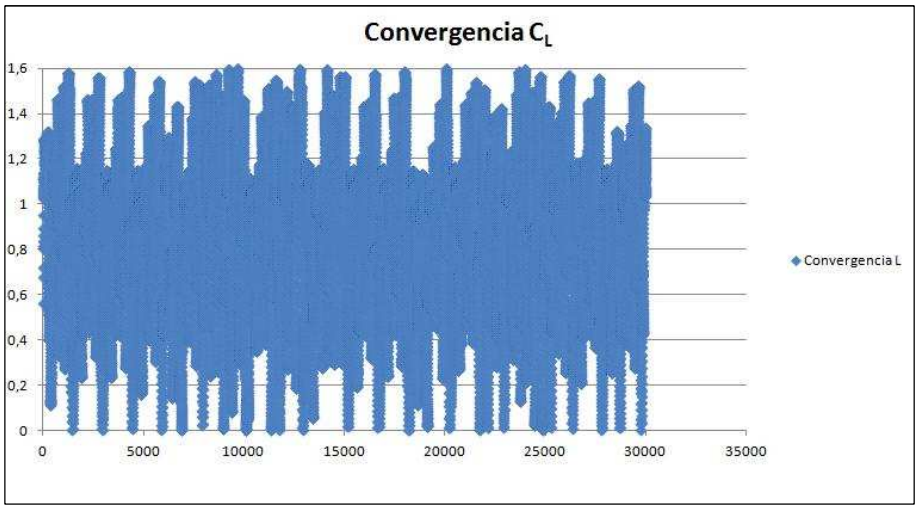
Solving p: Initial residual $= 8{,}674 \cdot 10^{-02}$, Final residual $= 4{,}297 \cdot 10^{-03}$, No Iterations $= 3$.

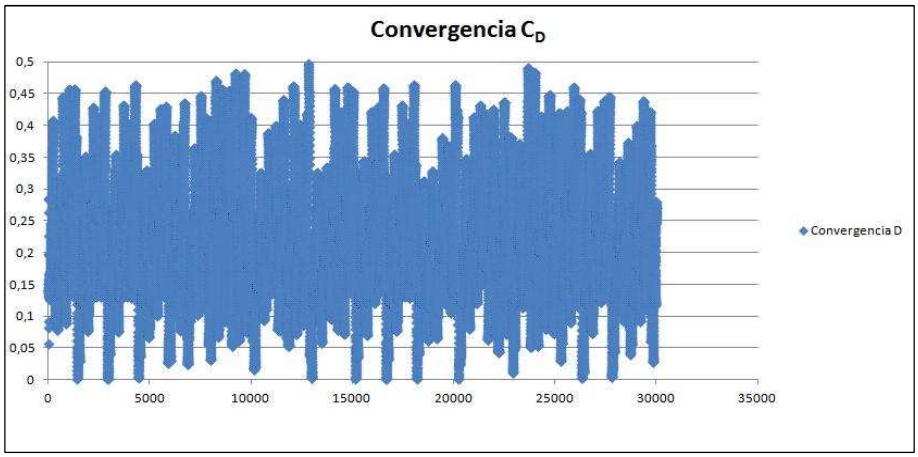
ExecutionTime: 3188,45 s. ClockTime: 3196 s.

ForceCoeffs output:

$C_L = 1{,}3376$

$C_D = 0{,}2806$





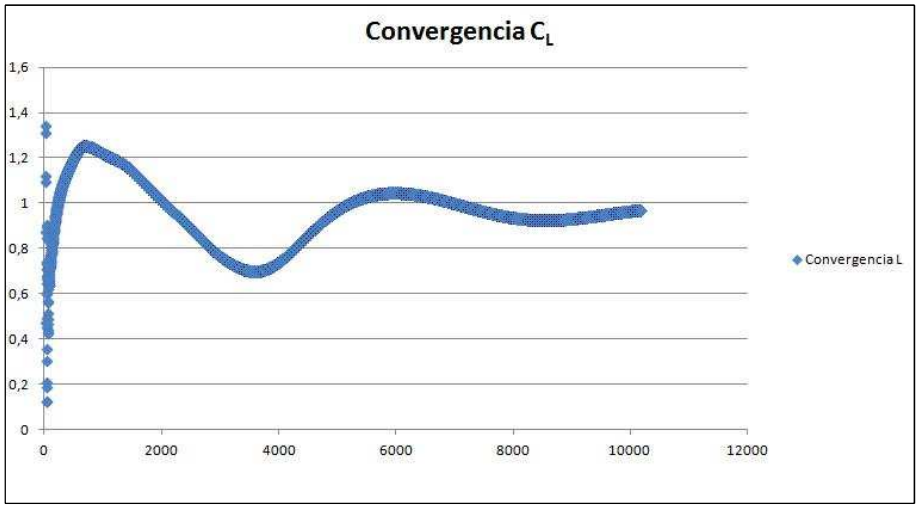
II.2.18.2. Caso 2 (Ángulo de ataque 16º)

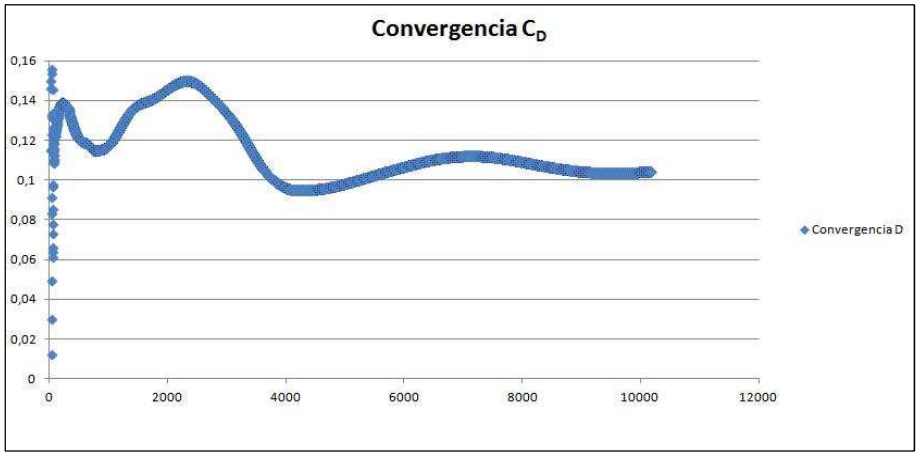
SIMPLE solution converged in 10178 iterations.

ForceCoeffs output:

$C_L = 0{,}9700$

$C_D = 0{,}1042$





II.2.18.3. Caso 3 (Ángulo de ataque 16º)

Time: 50000

Solving U_X: Initial residual $= 7{,}734 \cdot 10^{-03}$, Final residual $= 3{,}583 \cdot 10^{-04}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 1{,}277 \cdot 10^{-02}$, Final residual $= 7{,}220 \cdot 10^{-04}$, No Iterations $= 4$.

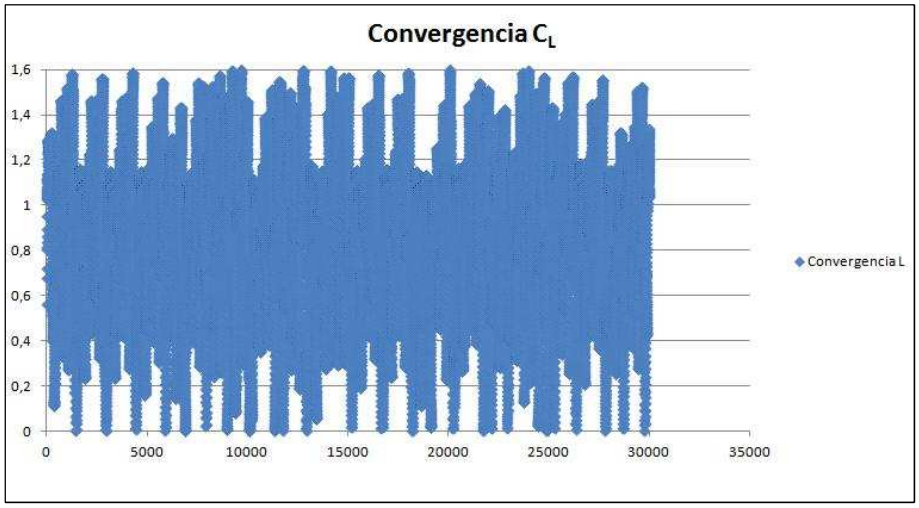
Solving p: Initial residual $= 1{,}410 \cdot 10^{-01}$, Final residual $= 1{,}398 \cdot 10^{-02}$, No Iterations $= 2$.

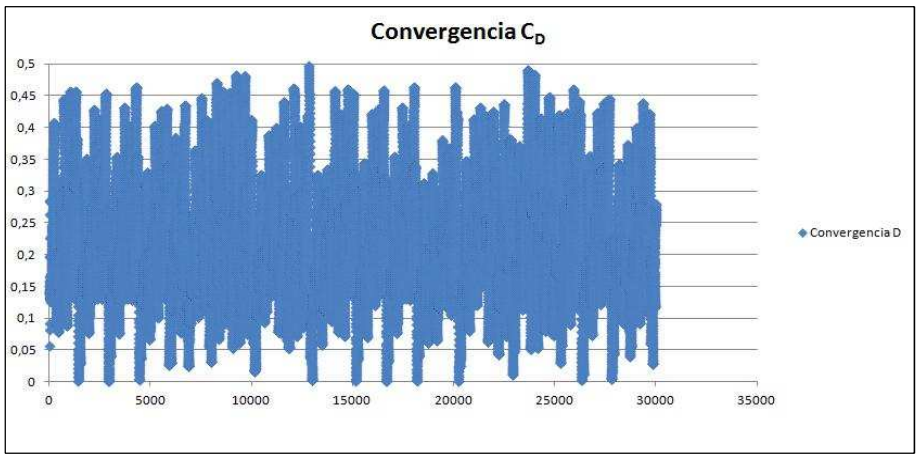
ExecutionTime: 5236,38 s. ClockTime: 5245 s.

ForceCoeffs output:

$C_L = 1{,}5359$

$C_D = 0{,}4339$

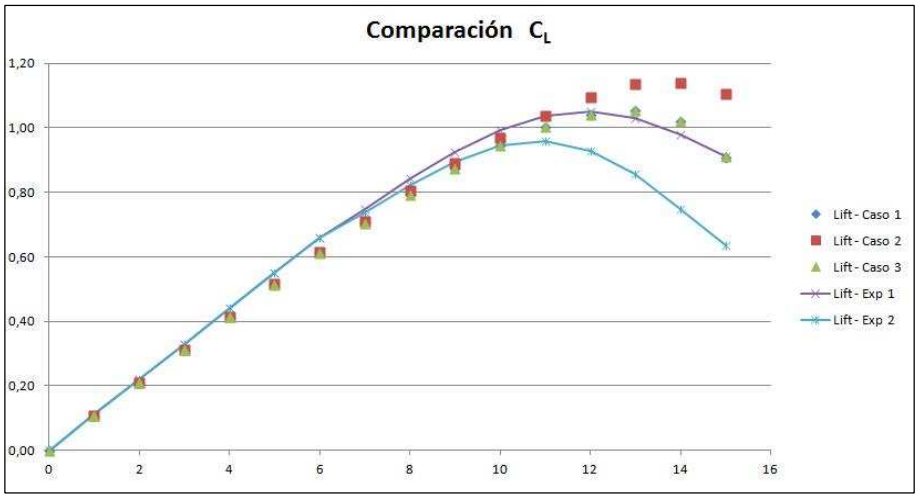


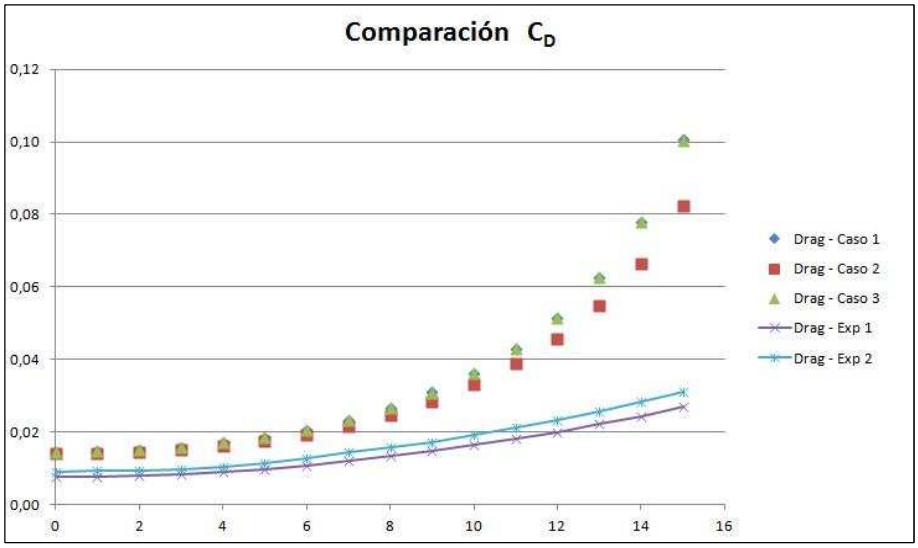


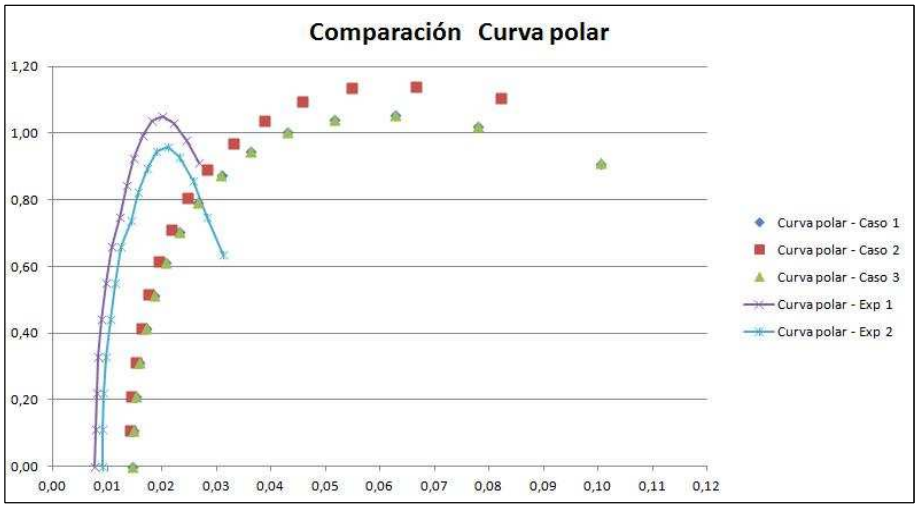
II.2.19. Resumen de resultados y comparación

Ángulo	C_L - Caso1	C_L - Caso2	C_L - Caso3	C_D - Caso1	C_D - Caso2	C_D - Caso3
0º	0,0001	-0,0001	0,0000	0,0146	0,0140	0,0146
1º	0,1061	0,1065	0,1063	0,0147	0,0141	0,0147
2º	0,2099	0,2102	0,2100	0,0152	0,0145	0,0152
3º	0,3126	0,3115	0,3128	0,0159	0,0152	0,0159
4º	0,4140	0,4146	0,4142	0,0171	0,0162	0,0171
5º	0,5137	0,5157	0,5139	0,0186	0,0176	0,0186
6º	0,6106	0,6152	0,6110	0,0207	0,0194	0,0207
7º	0,7040	0,7117	0,7045	0,0233	0,0217	0,0233
8º	0,7921	0,8043	0,7926	0,0267	0,0247	0,0267
9º	0,8732	0,8908	0,8735	0,0309	0,0284	0,0309
10º	0,9441	0,9693	0,9446	0,0363	0,0330	0,0363
11º	1,0020	1,0386	1,0025	0,0430	0,0387	0,0430
12º	1,0407	1,0957	1,0415	0,0516	0,0458	0,0516
13º	1,0526	1,1343	1,0525	0,0628	0,0548	0,0627
14º	1,0188	1,1405	1,0194	0,0780	0,0665	0,0780
15º	0,9081	1,1064	0,9098	0,1005	0,0822	0,1005

Ángulo	C_L - Exp1	C_L - Exp2	C_D - Exp1	C_D - Exp2
Re	700000	360000	700000	360000
0º	0,0000	0,0000	0,0077	0,0091
1º	0,1100	0,1100	0,0078	0,0092
2º	0,2200	0,2200	0,0080	0,0094
3º	0,3300	0,3300	0,0083	0,0098
4º	0,4400	0,4400	0,0089	0,0105
5º	0,5500	0,5500	0,0098	0,0114
6º	0,6600	0,6600	0,0108	0,0126
7º	0,7483	0,7390	0,0122	0,0143
8º	0,8442	0,8240	0,0135	0,0157
9º	0,9260	0,8946	0,0149	0,0173
10º	0,9937	0,9440	0,0164	0,0191
11º	1,0363	0,9572	0,0182	0,0211
12º	1,0508	0,9285	0,0200	0,0233
13º	1,0302	0,8562	0,0221	0,0257
14º	0,9801	0,7483	0,0244	0,0283
15º	0,9119	0,6350	0,0269	0,0312


Comparación C_L
1,20
1,00
0,80
0,60
0,40
0,20
0,00
0
2
4
6
8
10
12
14
16
Lift - Caso 1
Lift - Caso 2
Lift - Caso 3
Lift - Exp 1
Lift - Exp 2


Comparación C_D
0,12
0,10
0,08
0,06
0,04
0,02
0,00
0
2
4
6
8
10
12
14
16
Drag - Caso 1
Drag - Caso 2
Drag - Caso 3
Drag - Exp 1
Drag - Exp 2


Comparación Curva polar
1,20
1,00
0,80
0,60
0,40
0,20
0,00
0,00
0,01
0,02
0,03
0,04
0,05
0,06
0,07
0,08
0,09
0,10
0,11
0,12
Curva polar - Caso 1
Curva polar - Caso 2
Curva polar - Caso 3
Curva polar - Exp 1
Curva polar - Exp 2

II.3. Cálculos numéricos del perfil FX63 – 137

II.3.1. Introducción

Este apartado se encarga de especificar el conjunto de simulaciones efectuadas al perfil FX63 – 137. Como se ha comentado con anterioridad en la *Memoria*, la decisión de analizar tal cantidad de casos atiende a consideraciones geométricas y aspectos relacionados con parámetros del proceso de cálculo.

En primer lugar, destacar que el perfil FX63 – 137 presenta un nivel de asimetría importante, además de disponer en su parte posterior de una geometría de cierta complejidad. Estas particularidades repercuten de forma considerable en el comportamiento aerodinámico del perfil. De hecho, existen ángulos de ataque negativos para los cuales se obtiene una fuerza de sustentación positiva. De este modo y a diferencia del perfil anterior, es necesario estudiar su comportamiento para ángulos inferiores a 0º.

Para determinar el conjunto de ángulos a estudiar, al razonamiento anterior se suman otras cuestiones como la convergencia de los resultados según el ángulo fijado, la disposición de un número limitado de datos experimentales para su comparación, etc. De esta forma, se decide finalmente estudiar el conjunto de ángulos comprendidos entre −8º y 18º con un paso de 2º.

Atendiendo a aspectos relacionados con el proceso de cálculo, se opta por analizar 3 casos distintos para cada uno de los ángulos de ataque anteriormente especificados. Los parámetros asociados a cada uno de los casos pueden ser consultados en el capítulo 3 de la *Memoria*.

Finalmente comentar que durante el análisis del perfil FX63 – 137 no se han realizado simulaciones con diferentes mallados.

II.3.2. Resultados ángulo de ataque -8º

II.3.2.1. Caso 1 (Ángulo de ataque -8º)

Time: 20000

Solving U_X: Initial residual $= 8{,}682 \cdot 10^{-04}$, Final residual $= 2{,}841 \cdot 10^{-05}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 2{,}052 \cdot 10^{-03}$, Final residual $= 8{,}904 \cdot 10^{-05}$, No Iterations $= 4$.

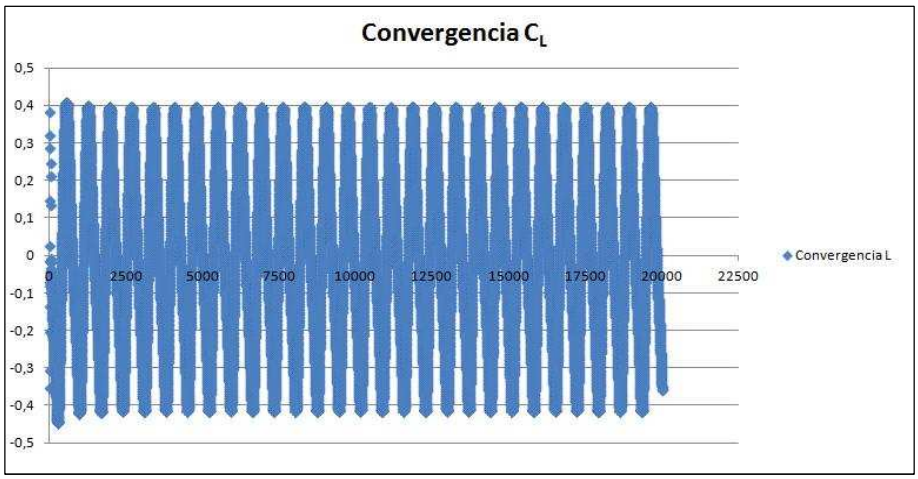
Solving p: Initial residual $= 8{,}429 \cdot 10^{-03}$, Final residual $= 5{,}624 \cdot 10^{-04}$, No Iterations $= 4$.

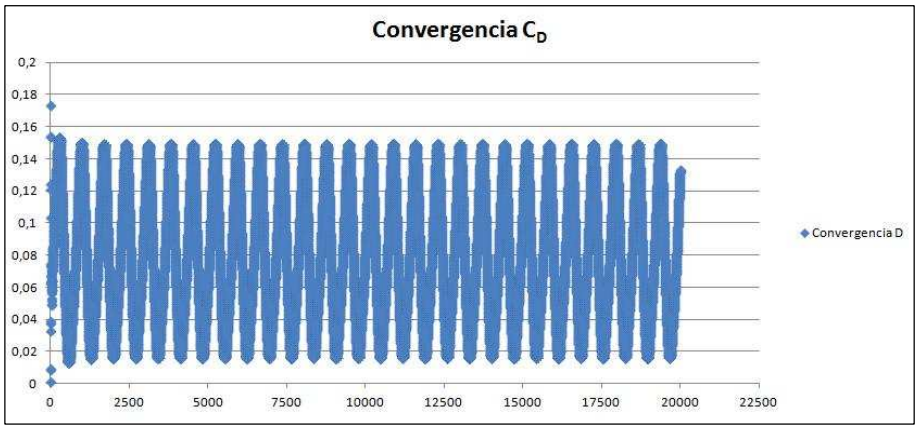
ExecutionTime: 6885,02 s. ClockTime: 6913 s.

ForceCoeffs output:

$C_L = -0{,}3602$

$C_D = 0{,}1329$





II.3.2.2. Caso 2 (Ángulo de ataque -8º)

Time: 20000

Solving U_X: Initial residual $= 1{,}168 \cdot 10^{-04}$, Final residual $= 8{,}371 \cdot 10^{-07}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 1{,}466 \cdot 10^{-04}$, Final residual $= 1{,}089 \cdot 10^{-06}$, No Iterations $= 2$.

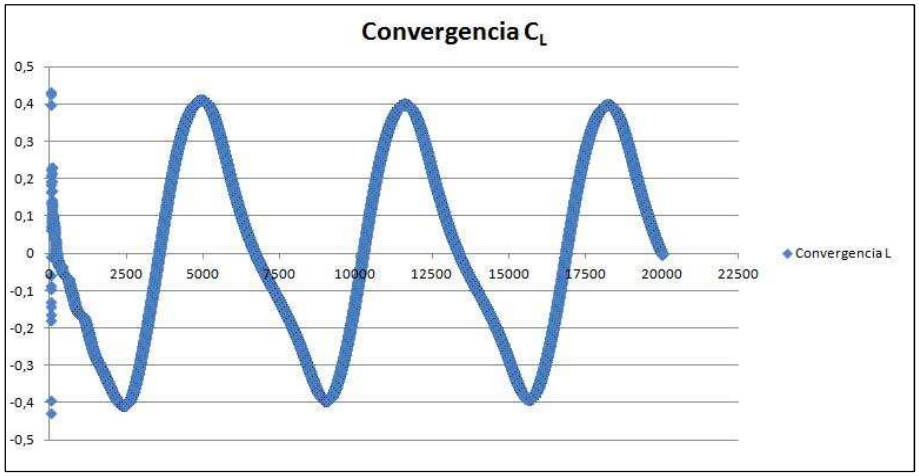
Solving p: Initial residual $= 9{,}762 \cdot 10^{-04}$, Final residual $= 4{,}351 \cdot 10^{-05}$, No Iterations $= 4$.

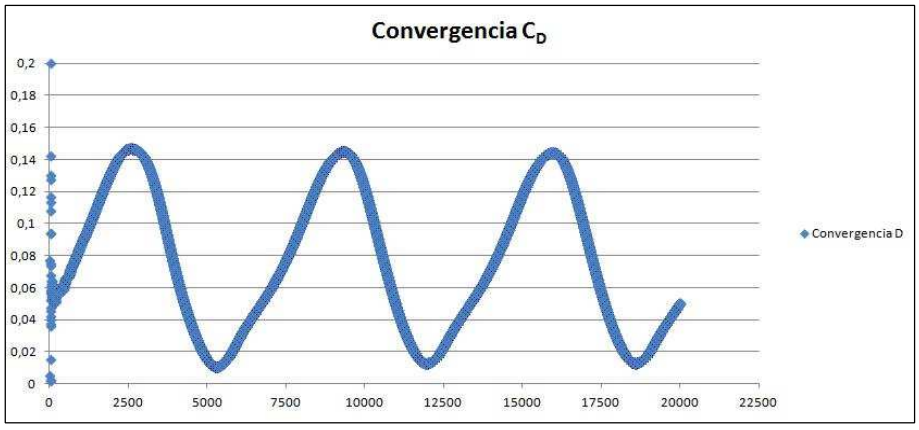
ExecutionTime: 6531,15 s. ClockTime: 6559 s.

ForceCoeffs output:

$C_L = -0{,}0024$

$C_D = 0{,}0501$





II.3.2.3. Caso 3 (Ángulo de ataque -8º)

Time: 30000

Solving U_X: Initial residual $= 1{,}006 \cdot 10^{-03}$, Final residual $= 4{,}958 \cdot 10^{-05}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 2{,}685 \cdot 10^{-03}$, Final residual $= 1{,}562 \cdot 10^{-04}$, No Iterations $= 4$.

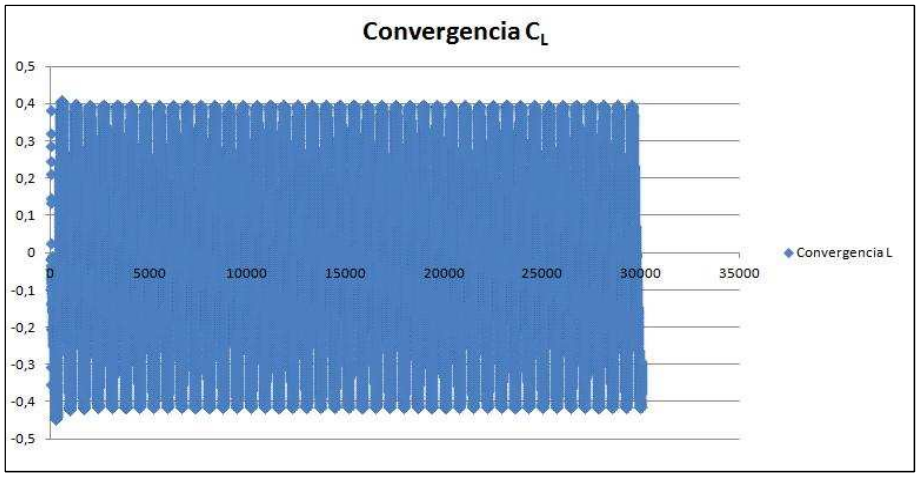
Solving p: Initial residual $= 1{,}166 \cdot 10^{-02}$, Final residual $= 1{,}162 \cdot 10^{-03}$, No Iterations $= 4$.

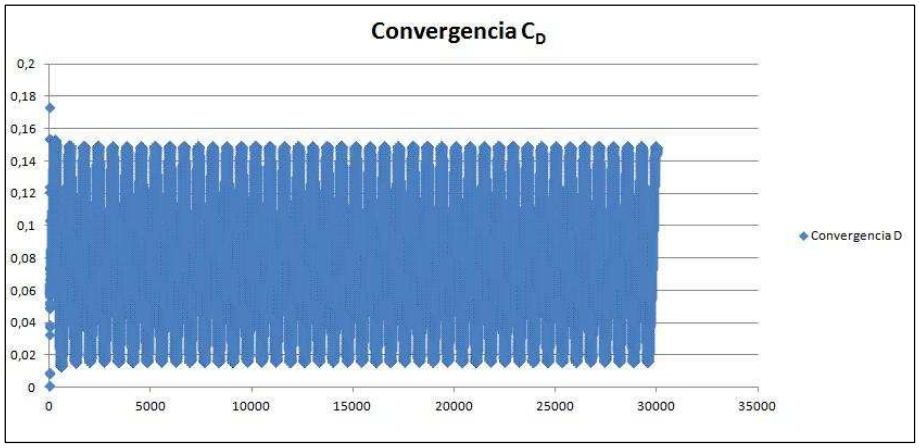
ExecutionTime: 10710,20 s. ClockTime: 10775 s.

ForceCoeffs output:

$C_L = -0{,}2902$

$C_D = 0{,}1443$





II.3.3. Resultados ángulo de ataque -6º

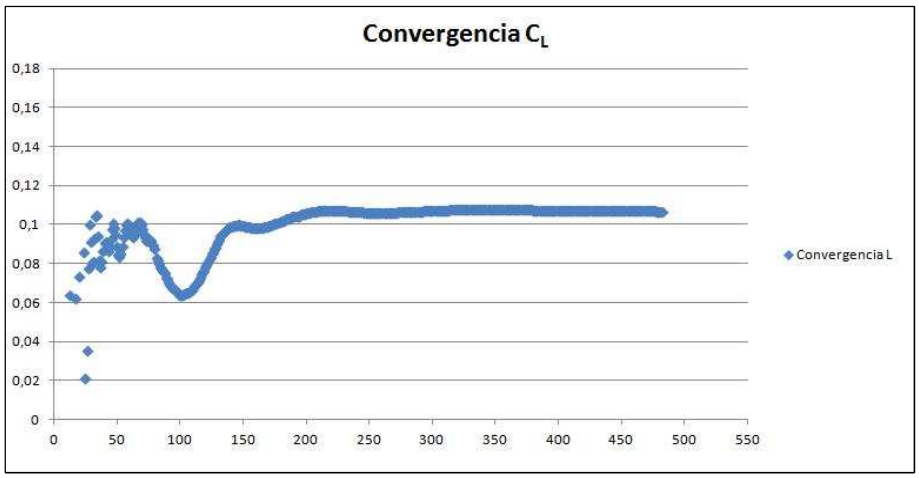
II.3.3.1. Caso 1 (Ángulo de ataque -6º)

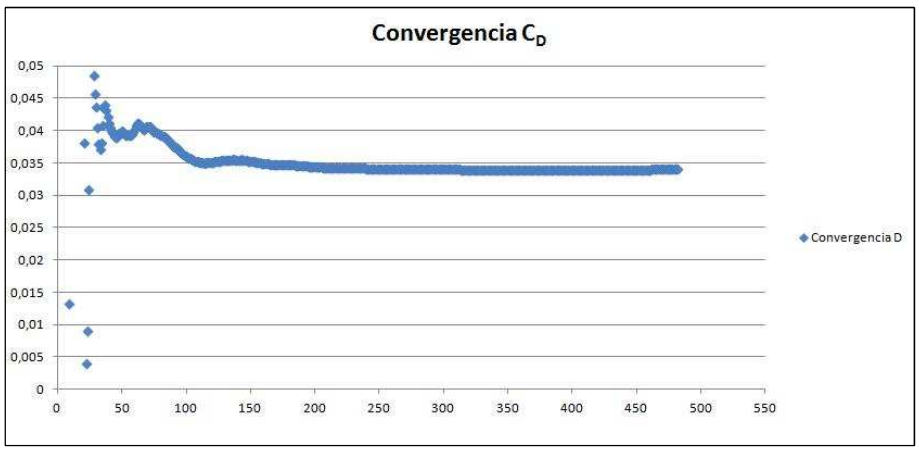
SIMPLE solution converged in 482 iterations.

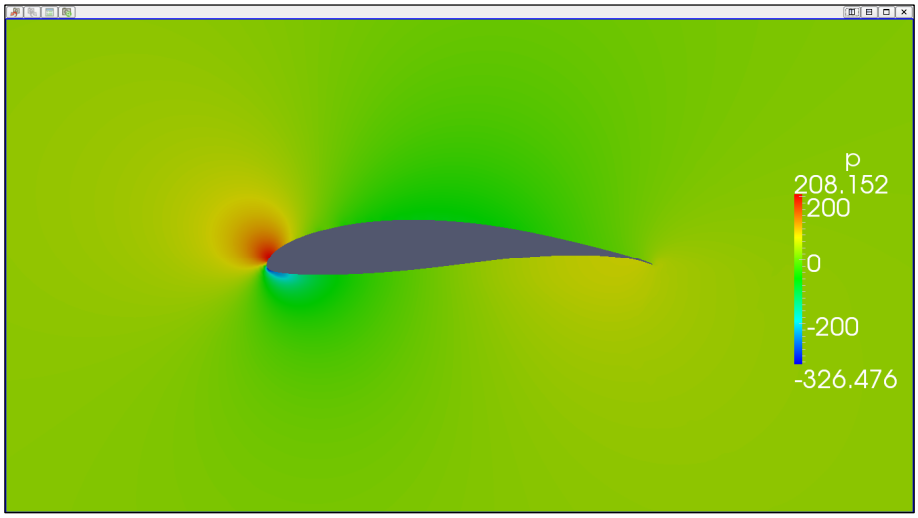
ForceCoeffs output:

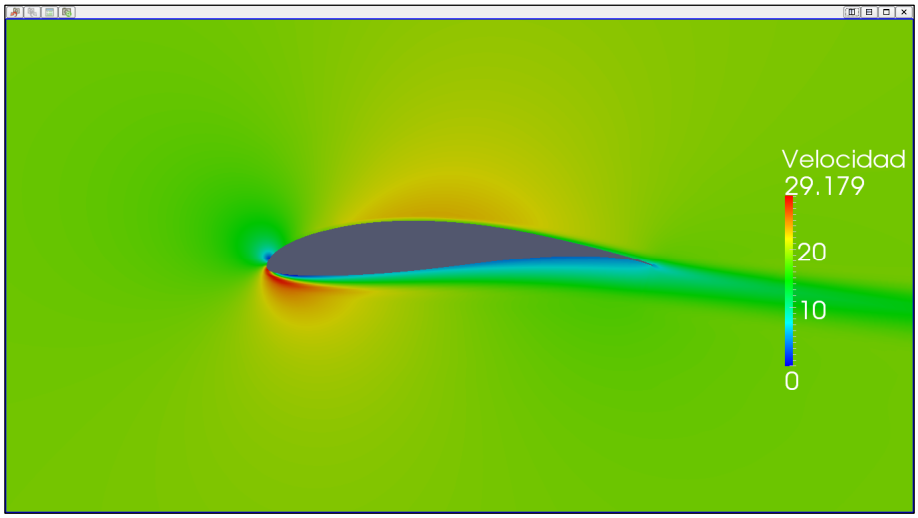
$C_L = 0{,}1068$

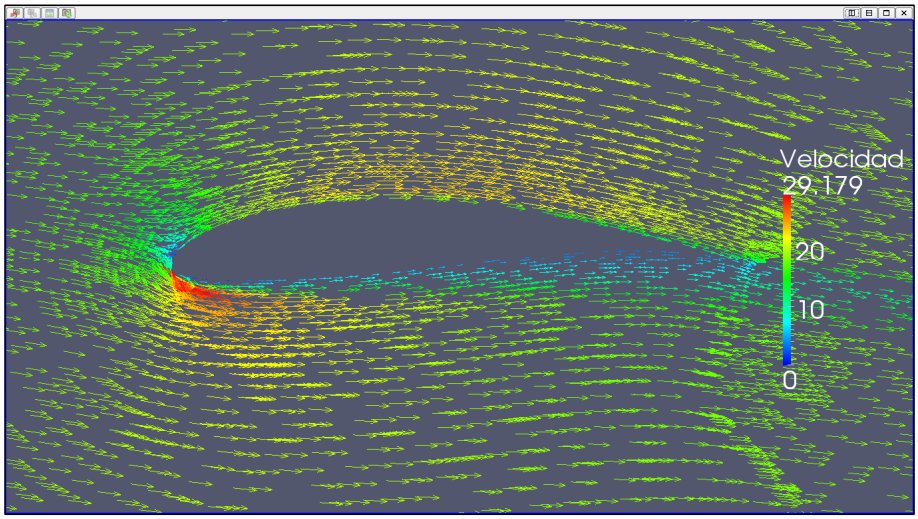
$C_D = 0{,}0340$






p
208.152
200
0
-200
-326.476


Velocidad
29.179
20
10
0


Velocidad
29.179
20
10
0

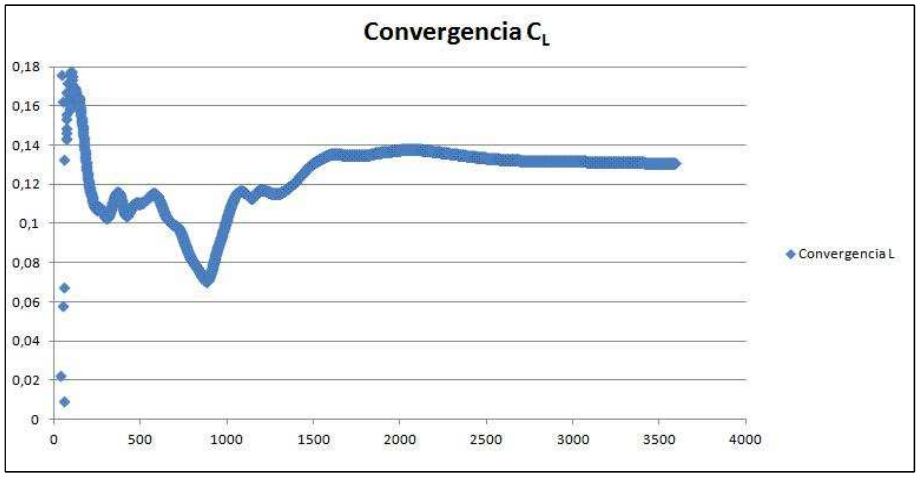
II.3.3.2. Caso 2 (Ángulo de ataque -6º)

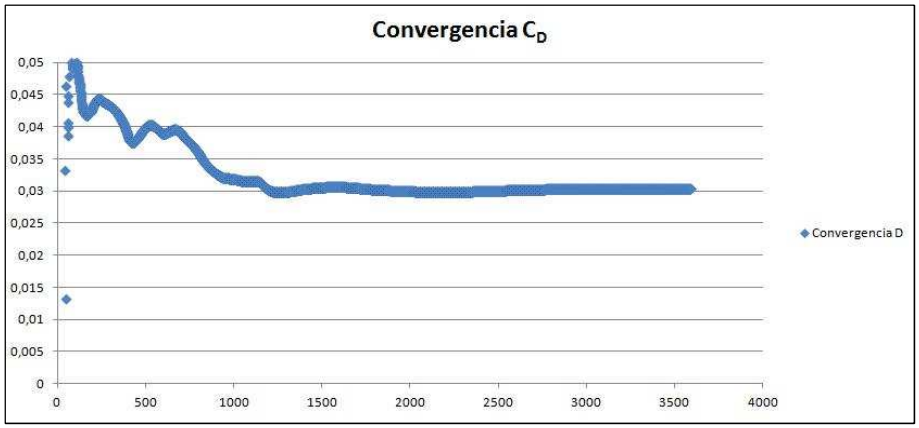
SIMPLE solution converged in 3592 iterations.

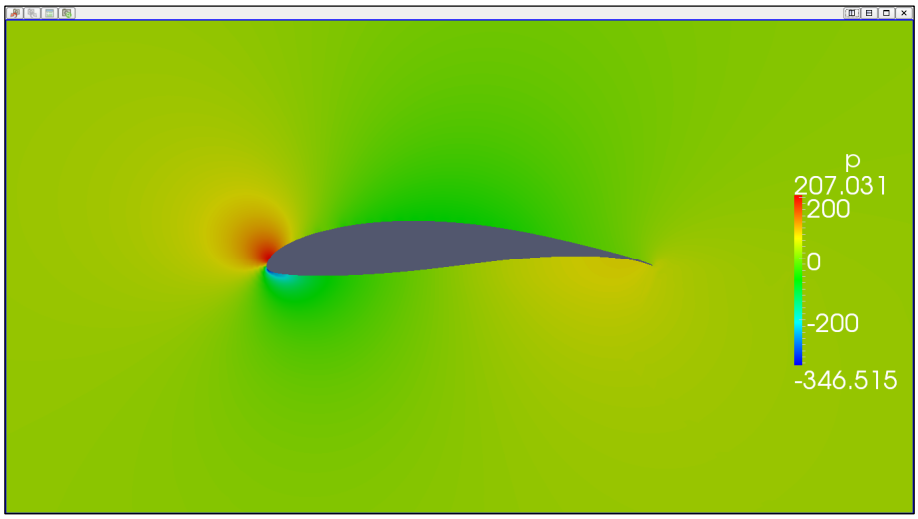
ForceCoeffs output:

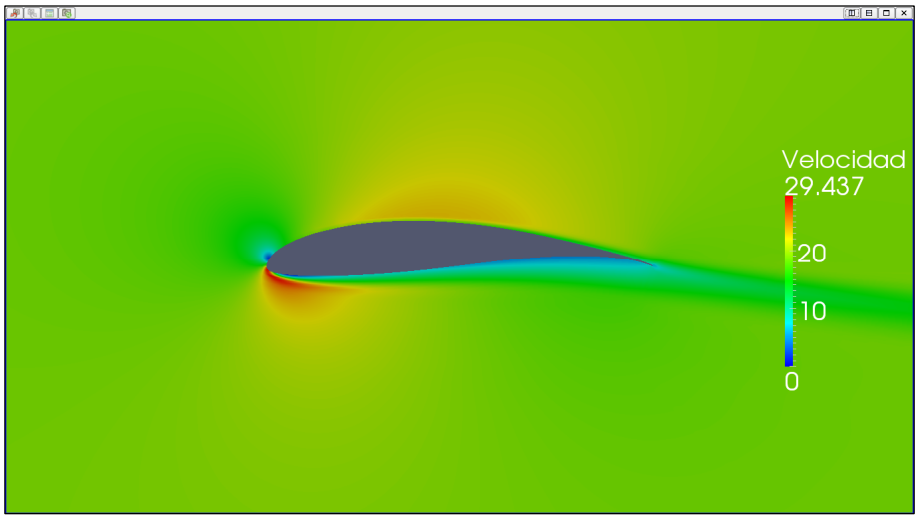
$C_L = 0{,}1307$

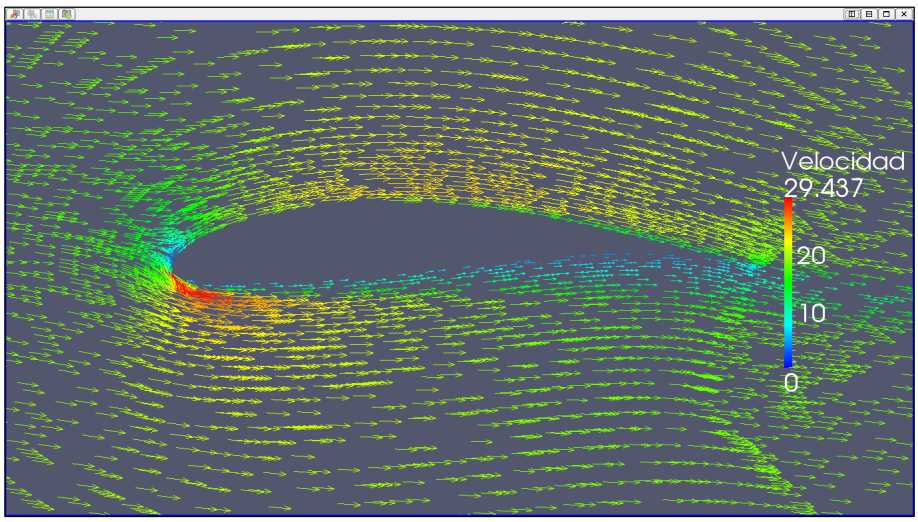
$C_D = 0{,}0304$






p
207.031
200
0
-200
-346.515


Velocidad
29.437
20
10
0


Velocidad
29.437
20
10
0

II.3.3.3. Caso 3 (Ángulo de ataque -6º)

Time: 30000

Solving U_X: Initial residual $= 1{,}188 \cdot 10^{-07}$, Final residual $= 9{,}697 \cdot 10^{-09}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 2{,}542 \cdot 10^{-07}$, Final residual $= 2{,}193 \cdot 10^{-08}$, No Iterations $= 2$.

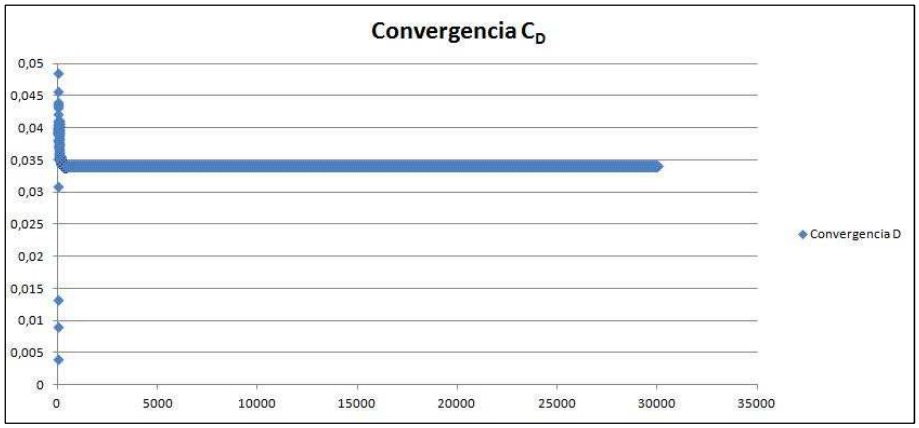
Solving p: Initial residual $= 1{,}558 \cdot 10^{-05}$, Final residual $= 1{,}368 \cdot 10^{-06}$, No Iterations $= 3$.

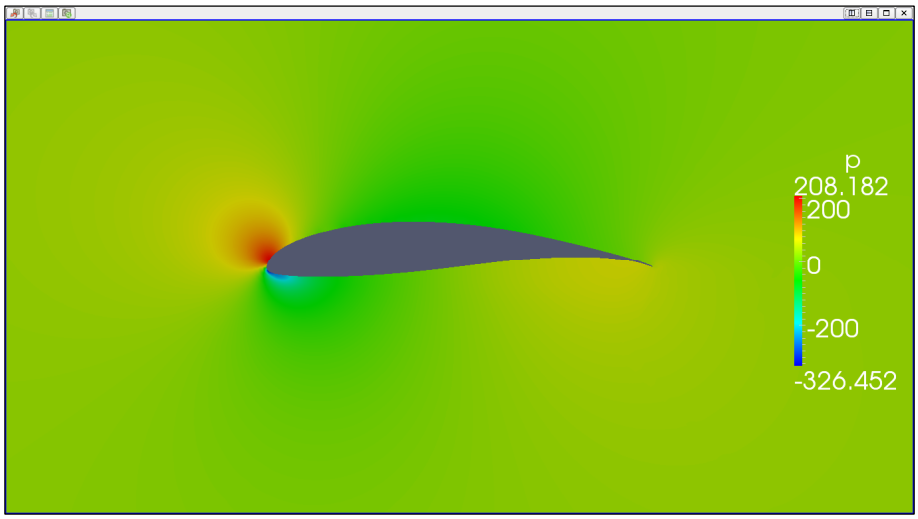
ExecutionTime: 10323,60 s. ClockTime: 10359 s.

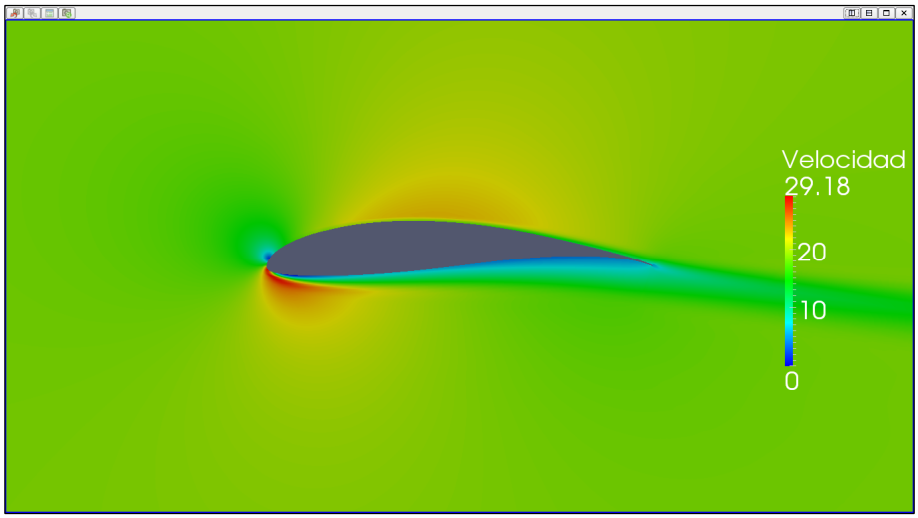
ForceCoeffs output:

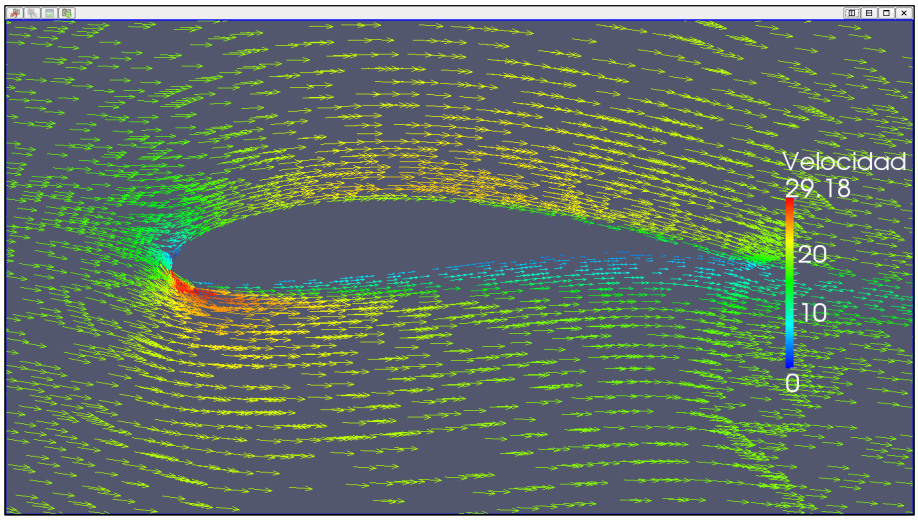
$C_L = 0{,}1063$

$C_D = 0{,}0340$




p
208.182
200
0
-200
-326.452


Velocidad
29.18
20
10
0


Velocidad
29.18
20
10
0

II.3.4. Resultados ángulo de ataque -4º

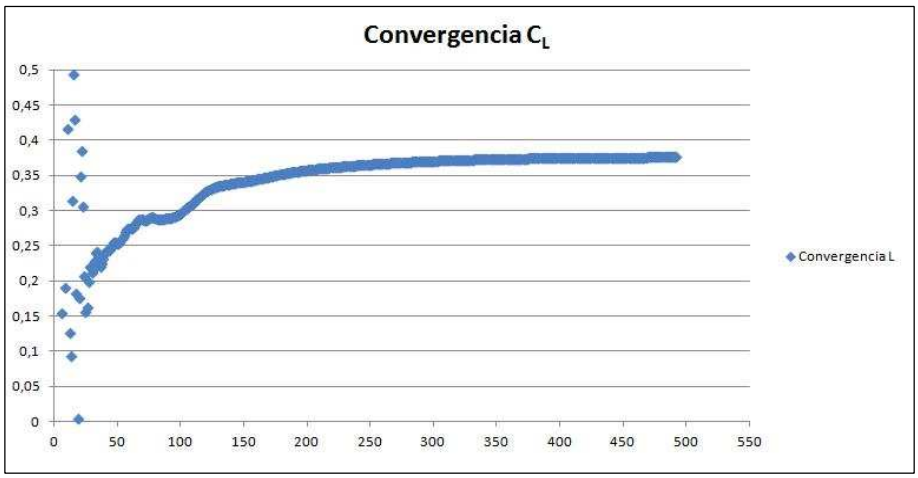
II.3.4.1. Caso 1 (Ángulo de ataque -4º)

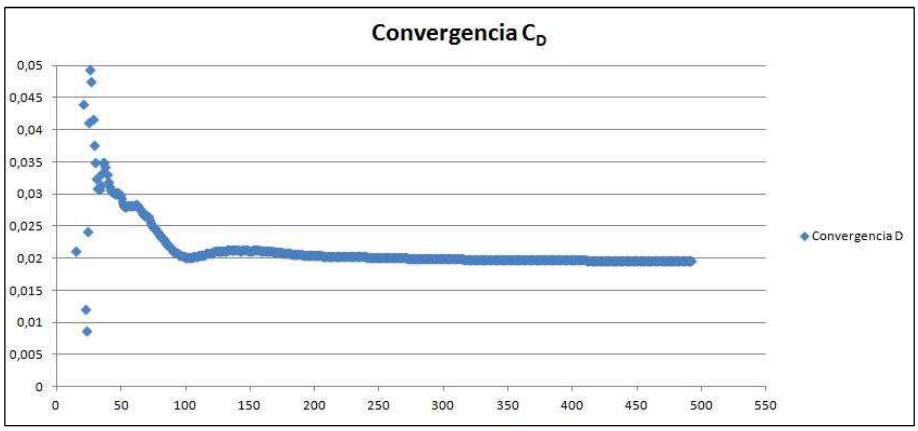
SIMPLE solution converged in 492 iterations.

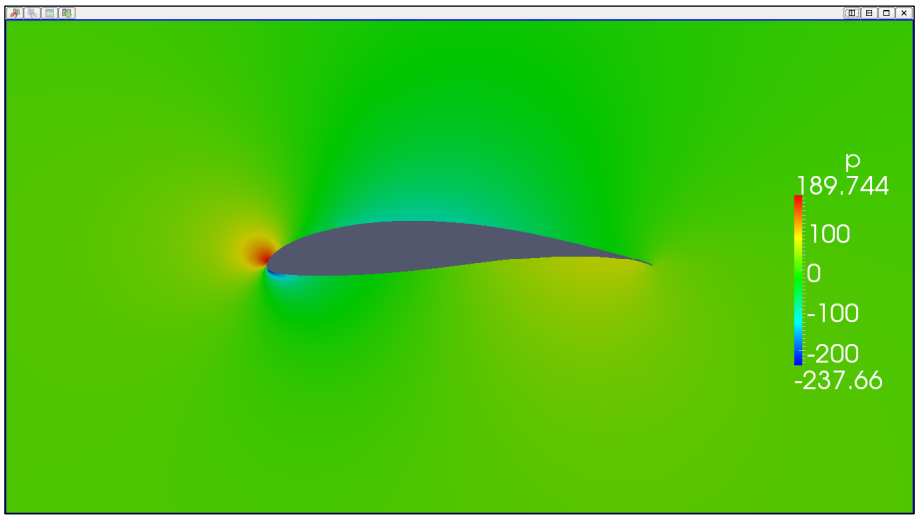
ForceCoeffs output:

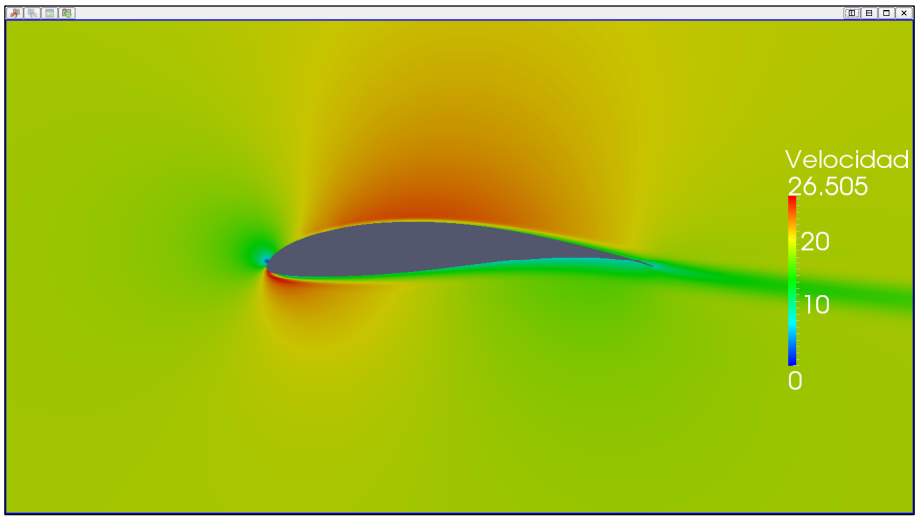
$C_L = 0{,}3760$

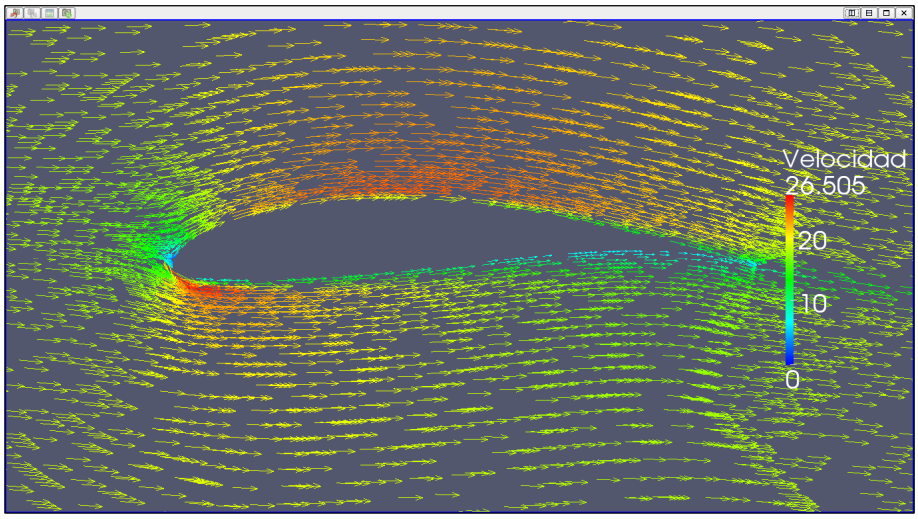
$C_D = 0{,}0196$






p
189.744
100
0
-100
-200
-237.66


Velocidad
26.505
20
10
0


Velocidad
26.505
20
10
0

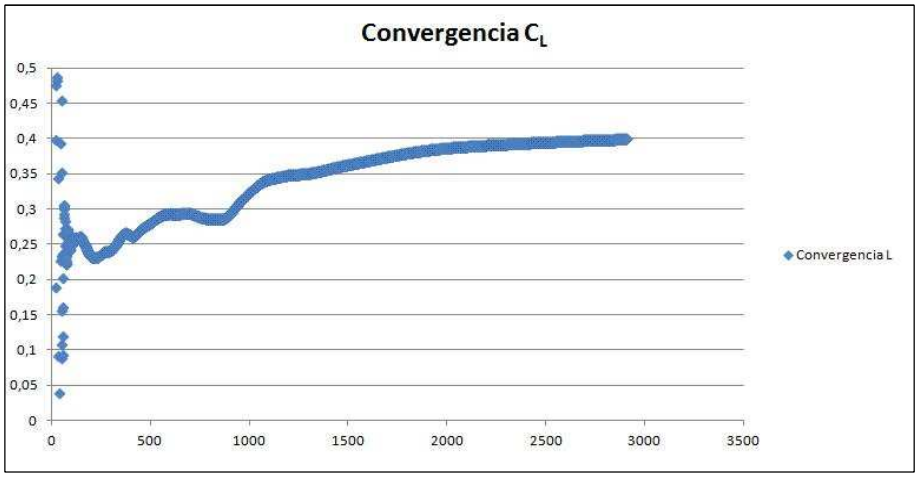
II.3.4.2. Caso 2 (Ángulo de ataque -4º)

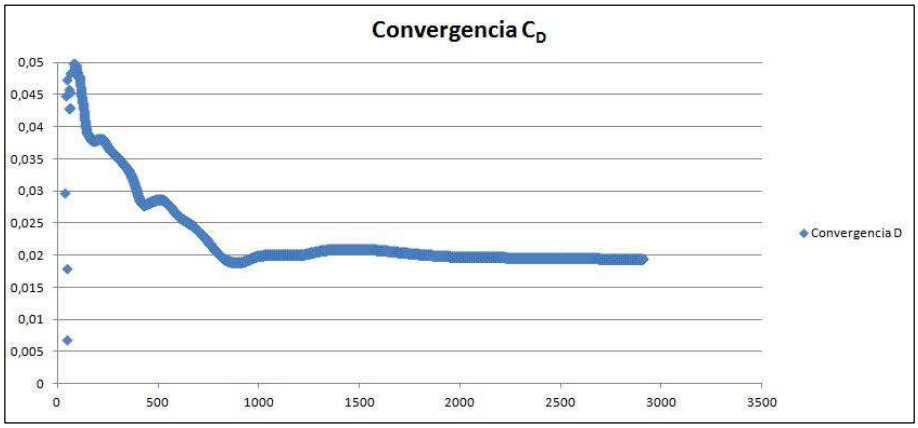
SIMPLE solution converged in 2910 iterations.

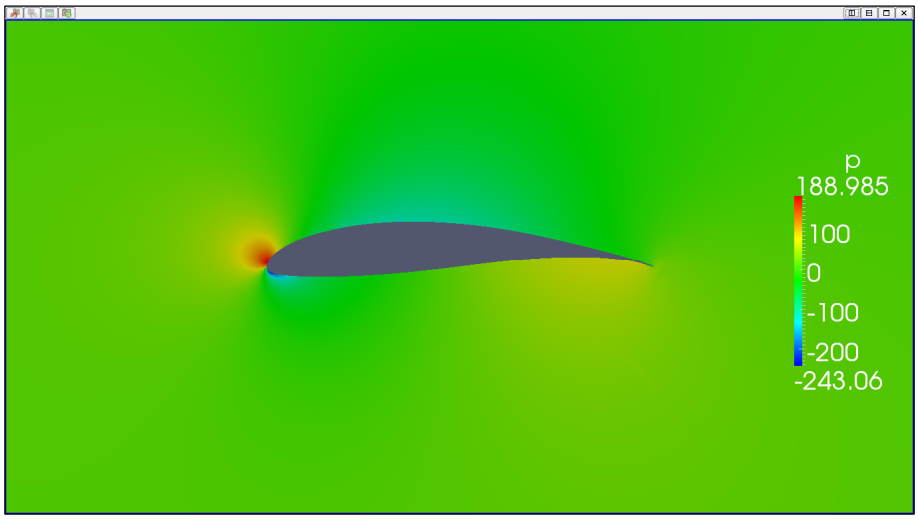
ForceCoeffs output:

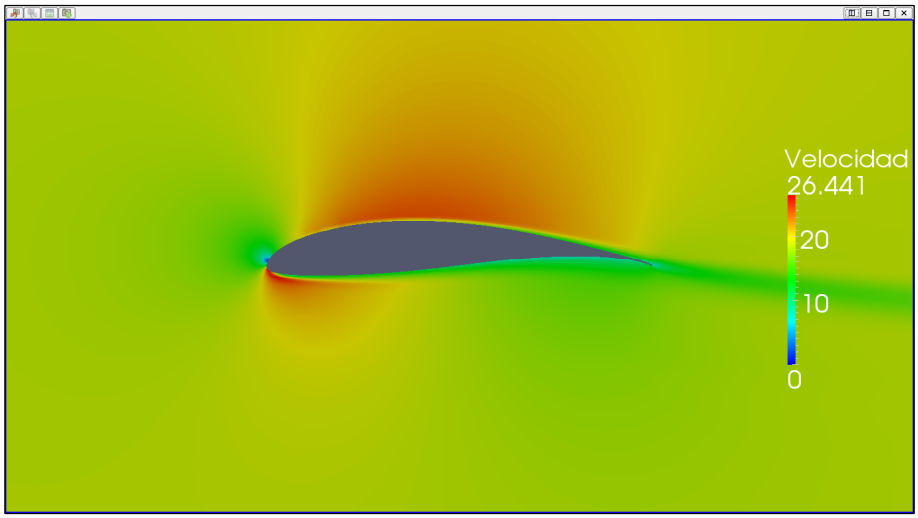
$C_L = 0{,}3994$

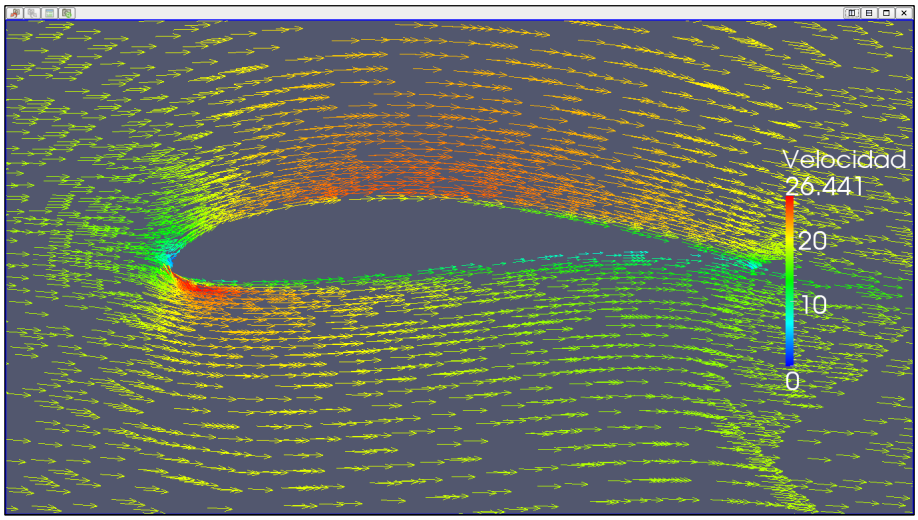
$C_D = 0{,}0194$






p
188.985
100
0
-100
-200
-243.06


Velocidad
26.441
20
10
0


Velocidad
26.441
20
10
0

II.3.4.3. Caso 3 (Ángulo de ataque -4º)

Time: 30000

Solving U_X: Initial residual $= 5{,}065 \cdot 10^{-07}$, Final residual $= 4{,}344 \cdot 10^{-09}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 7{,}320 \cdot 10^{-07}$, Final residual $= 6{,}256 \cdot 10^{-08}$, No Iterations $= 2$.

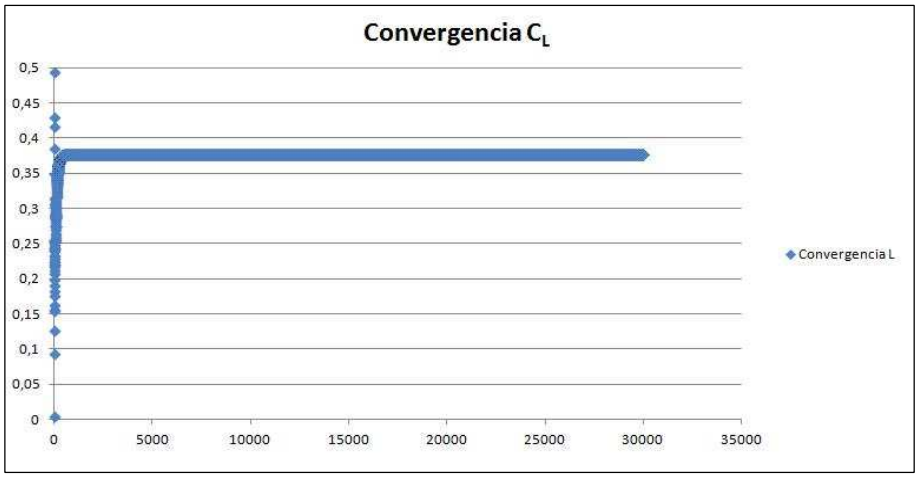
Solving p: Initial residual $= 3{,}188 \cdot 10^{-05}$, Final residual $= 2{,}810 \cdot 10^{-06}$, No Iterations $= 4$.

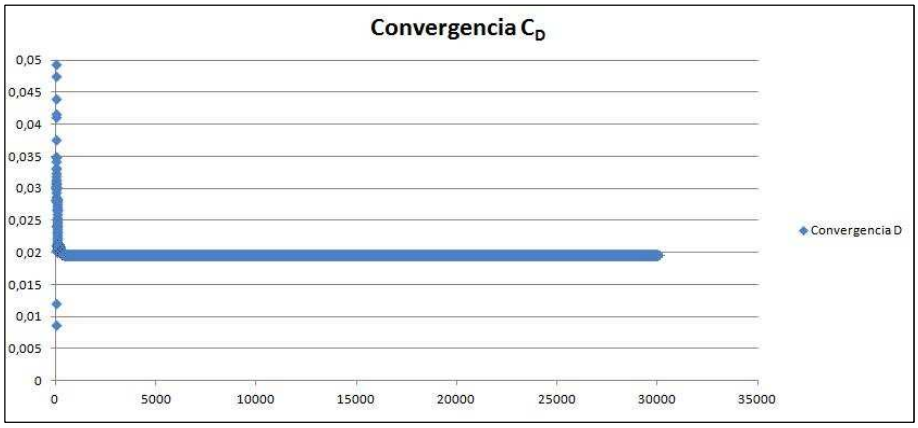
ExecutionTime: 9931,69 s. ClockTime: 9972 s.

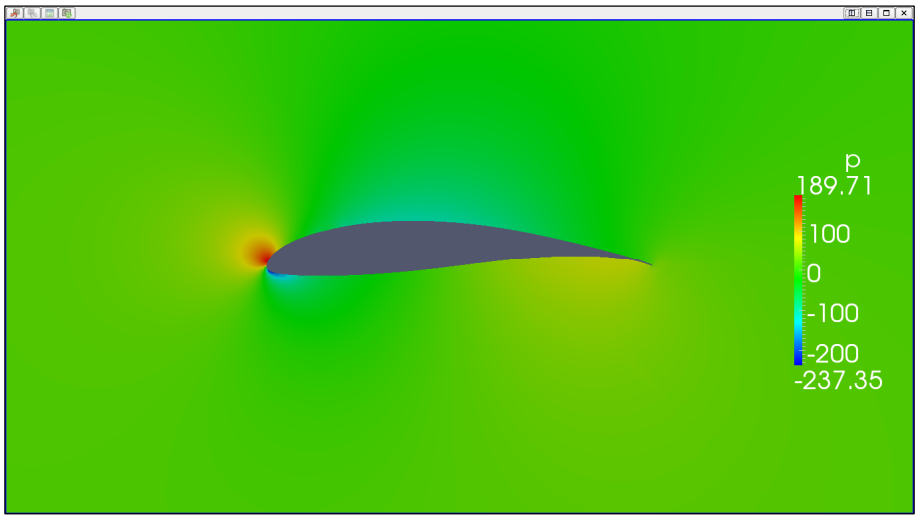
ForceCoeffs output:

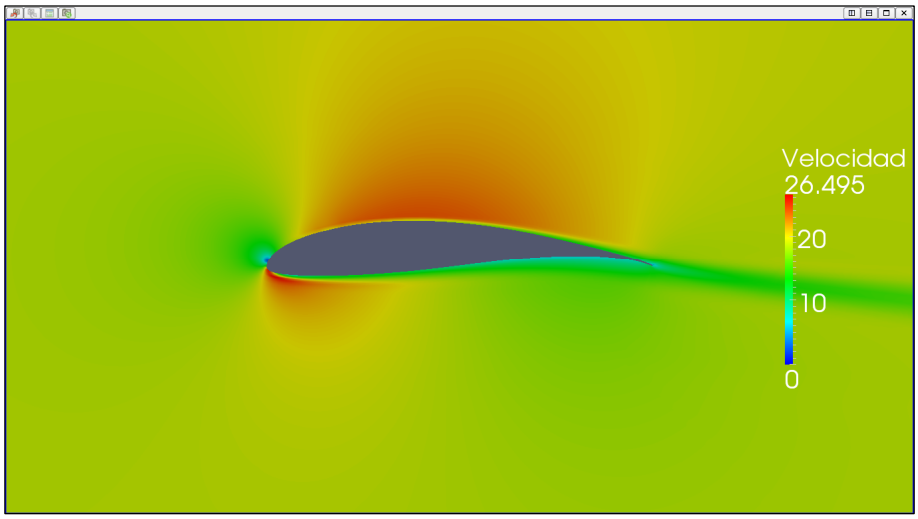
$C_L = 0{,}3766$

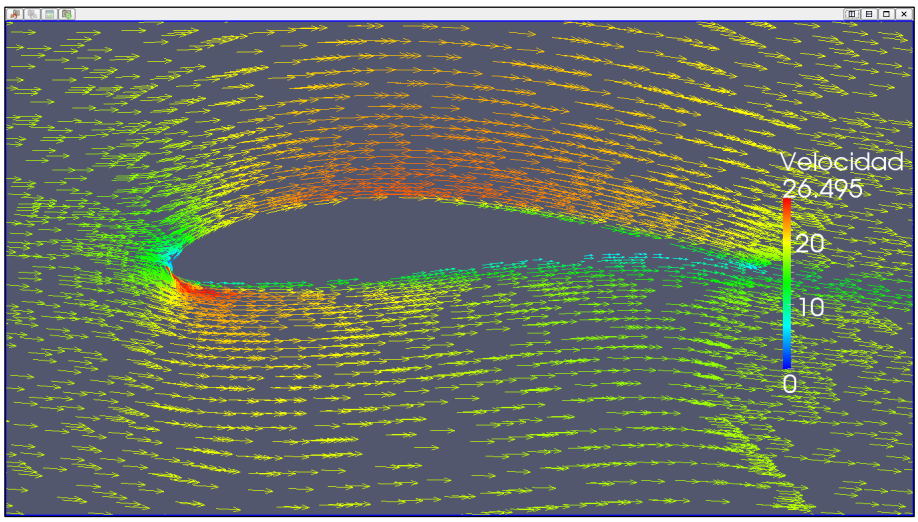
$C_D = 0{,}0196$






p
189.71
100
0
-100
-200
-237.35


Velocidad
26.495
20
10
0


Velocidad
26.495
20
10
0

II.3.5. Resultados ángulo de ataque -2º

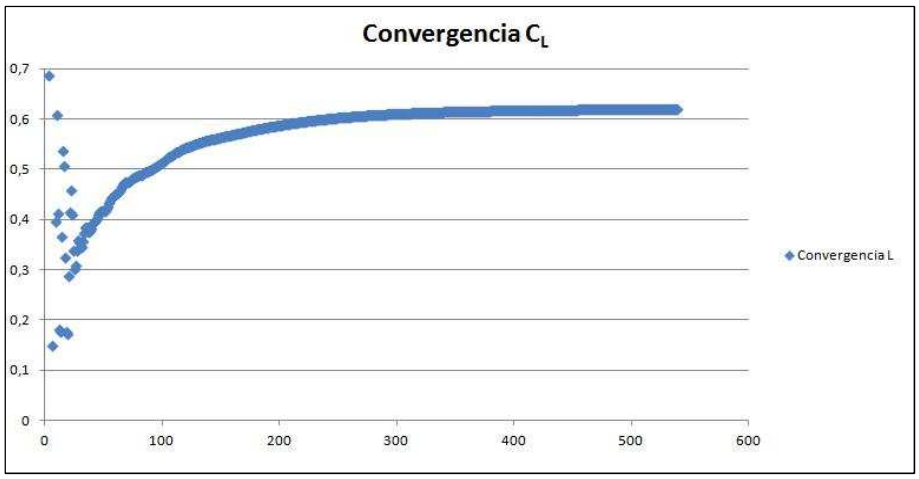
II.3.5.1. Caso 1 (Ángulo de ataque -2º)

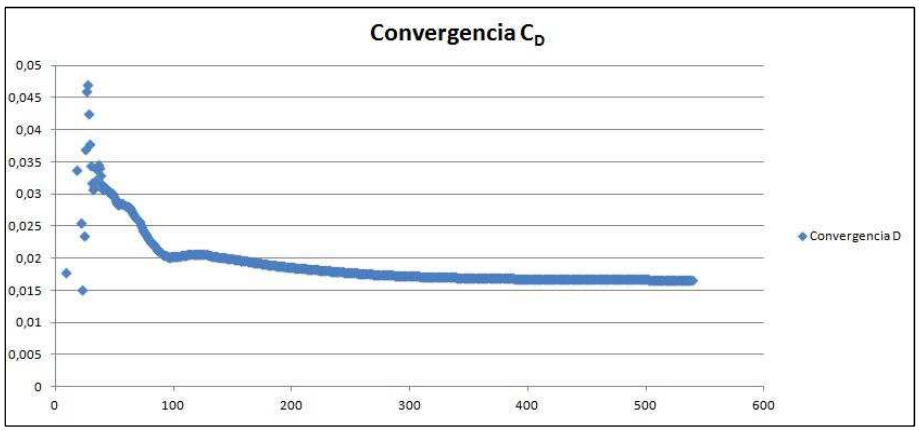
SIMPLE solution converged in 539 iterations.

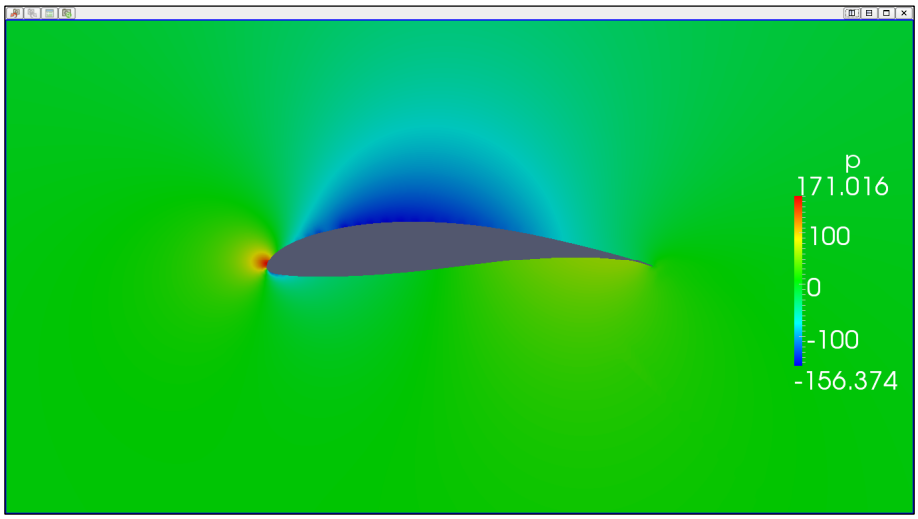
ForceCoeffs output:

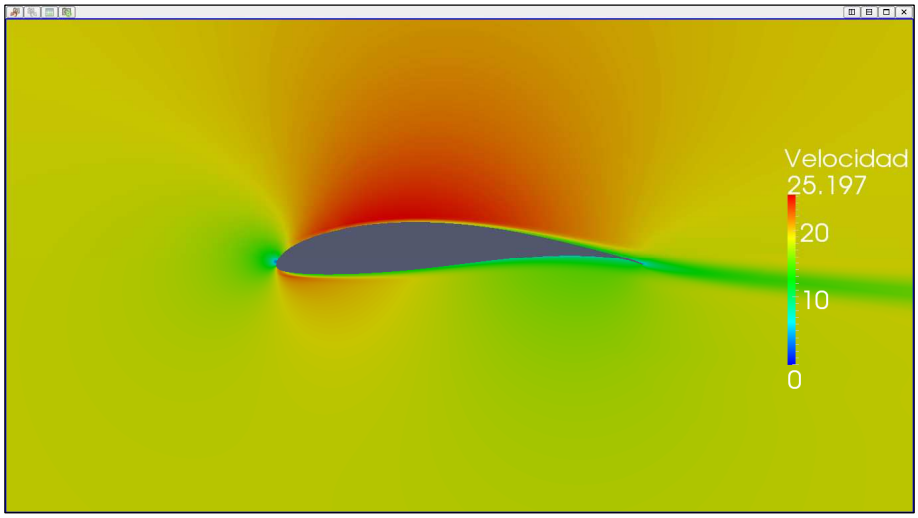
$C_L = 0{,}6194$

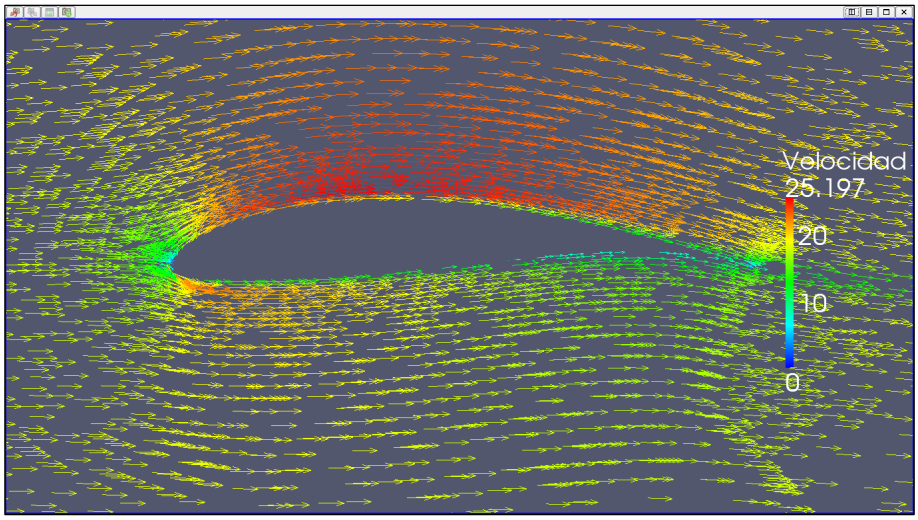
$C_D = 0{,}0166$






p
171.016
100
0
-100
-156.374


Velocidad
25.197
20
10
0


Velocidad
25.197
20
10
0

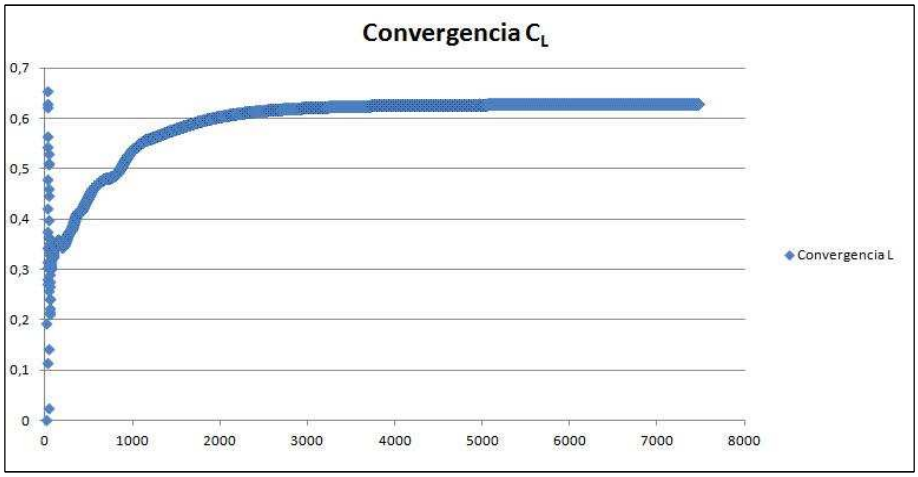
II.3.5.2. Caso 2 (Ángulo de ataque -2º)

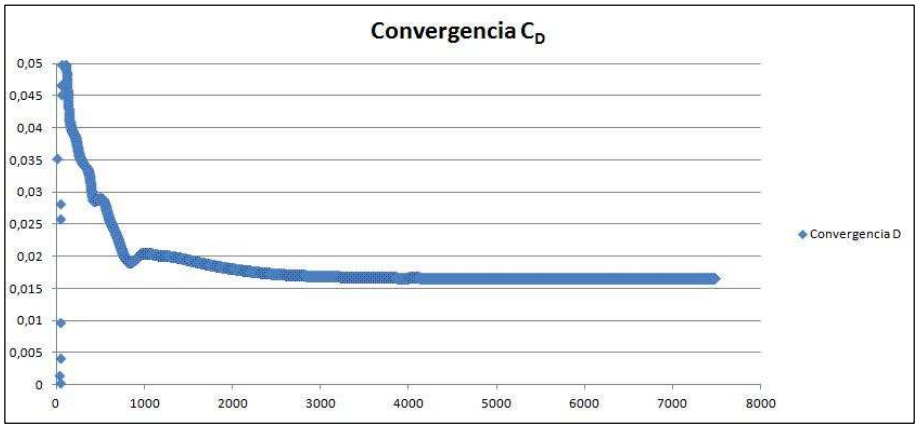
SIMPLE solution converged in 7478 iterations.

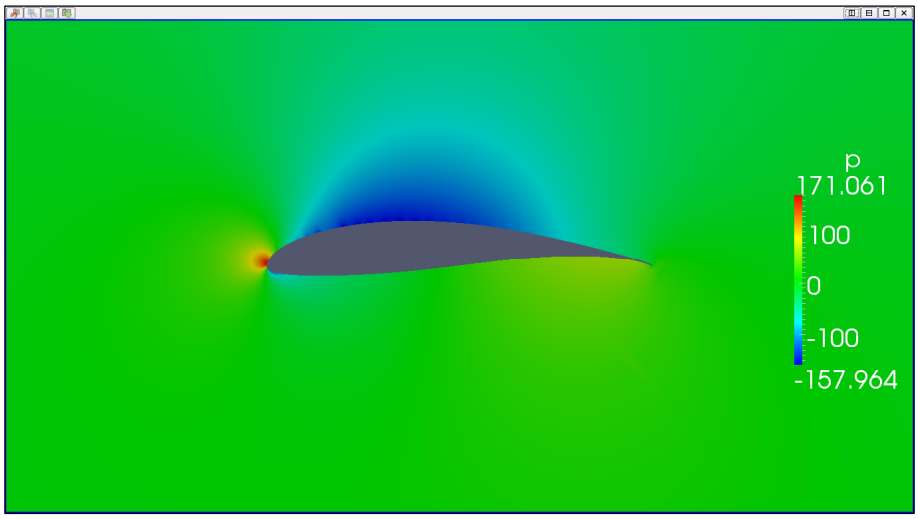
ForceCoeffs output:

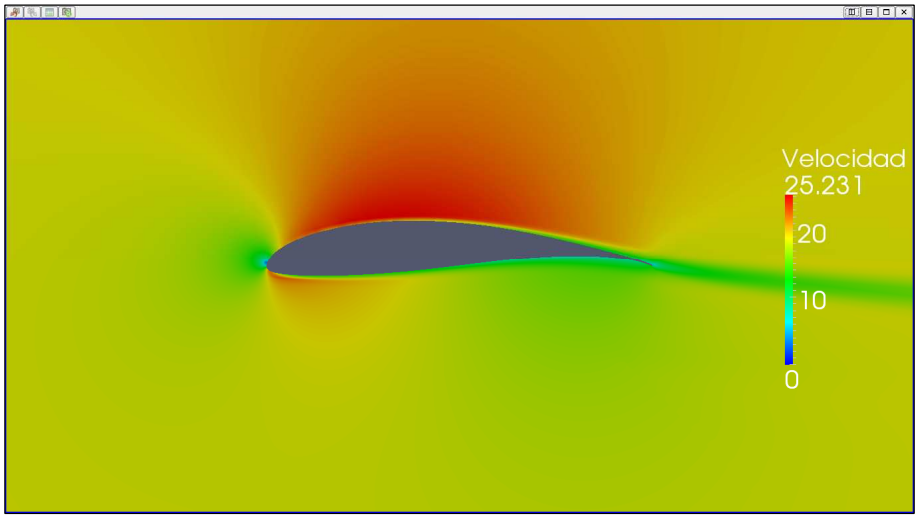
$C_L = 0{,}6283$

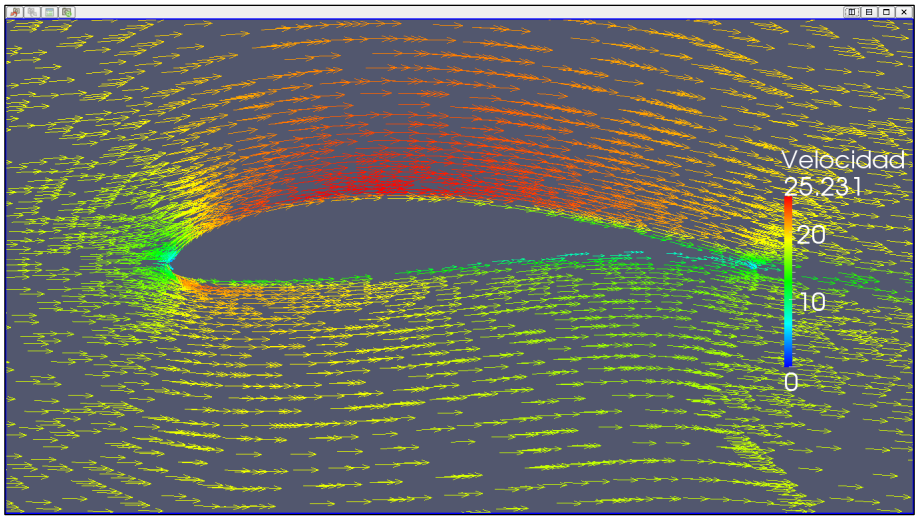
$C_D = 0{,}0165$






p
171.061
100
0
-100
-157.964


Velocidad
25.231
20
10
0


Velocidad
25.231
20
10
0

II.3.5.3. Caso 3 (Ángulo de ataque -2º)

Time: 30000

Solving U_X: Initial residual $= 8{,}482 \cdot 10^{-06}$, Final residual $= 5{,}294 \cdot 10^{-07}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 7{,}044 \cdot 10^{-06}$, Final residual $= 2{,}195 \cdot 10^{-07}$, No Iterations $= 2$.

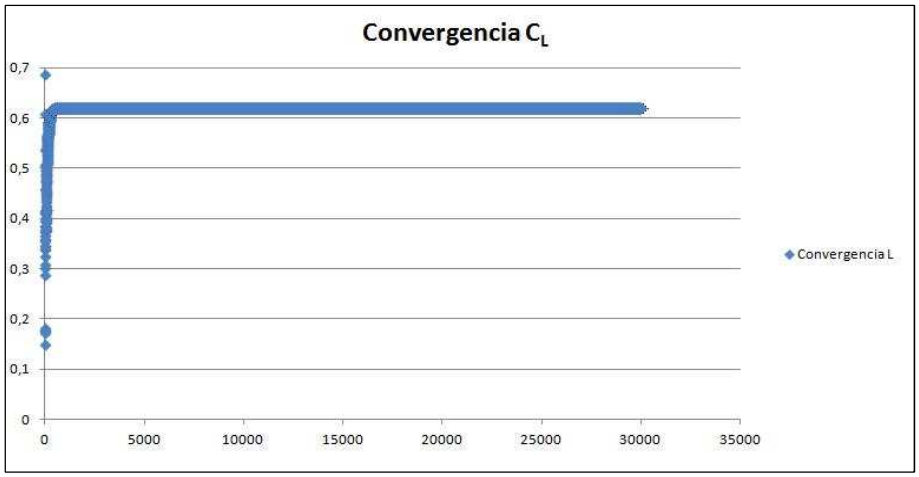
Solving p: Initial residual $= 3{,}350 \cdot 10^{-04}$, Final residual $= 2{,}993 \cdot 10^{-05}$, No Iterations $= 3$.

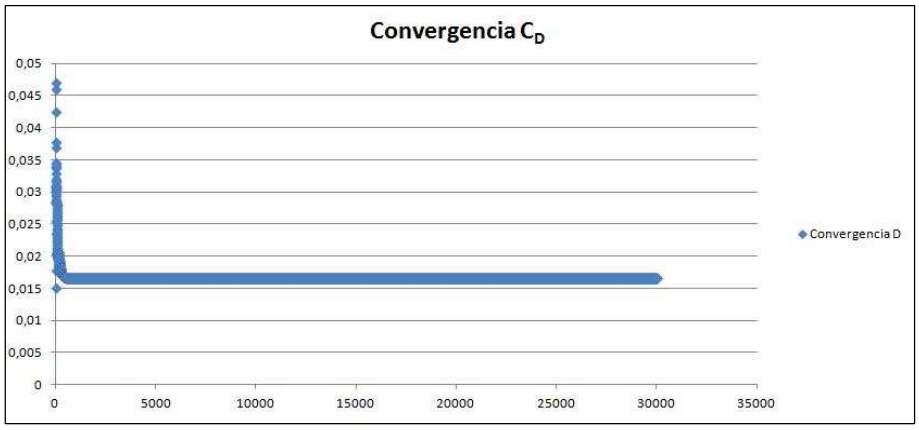
ExecutionTime: 9776,69 s. ClockTime: 9806 s.

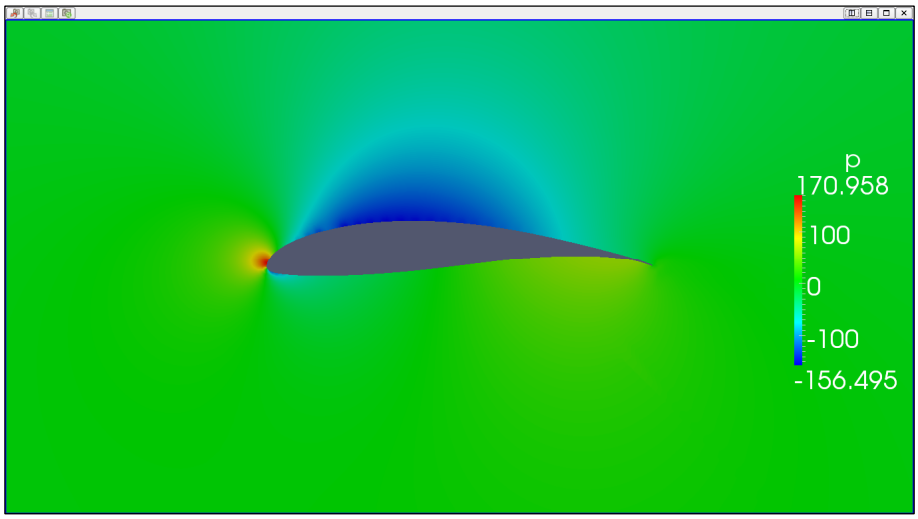
ForceCoeffs output:

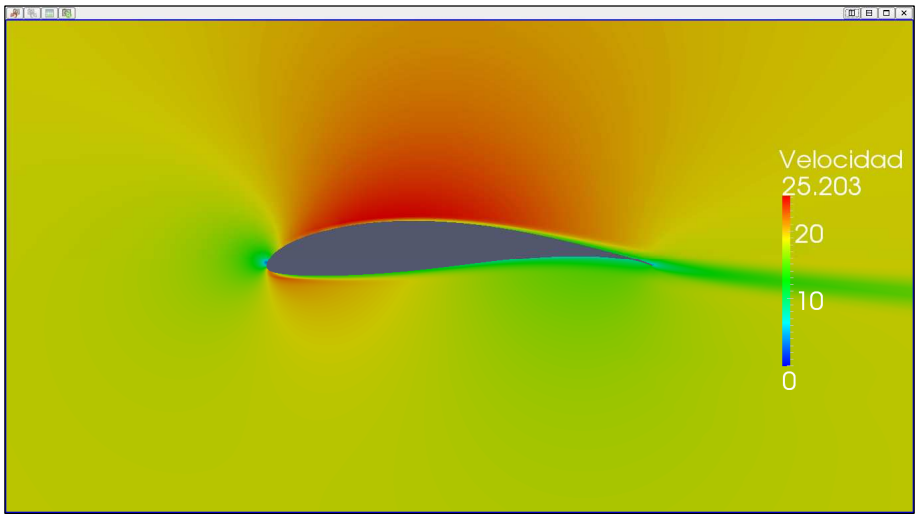
$C_L = 0{,}6200$

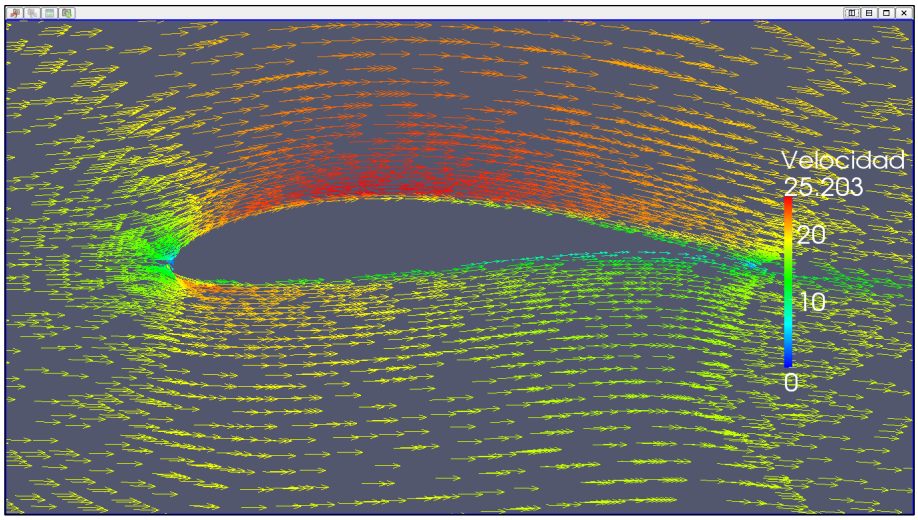
$C_D = 0{,}0166$






p
170.958
100
0
-100
-156.495


Velocidad
25.203
20
10
0


Velocidad
25.203
20
10
0

II.3.6. Resultados ángulo de ataque 0º

II.3.6.1. Caso 1 (Ángulo de ataque 0º)

Time: 20000

Solving U_X: Initial residual $= 9{,}803 \cdot 10^{-05}$, Final residual $= 4{,}504 \cdot 10^{-06}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 8{,}434 \cdot 10^{-05}$, Final residual $= 6{,}194 \cdot 10^{-06}$, No Iterations $= 2$.

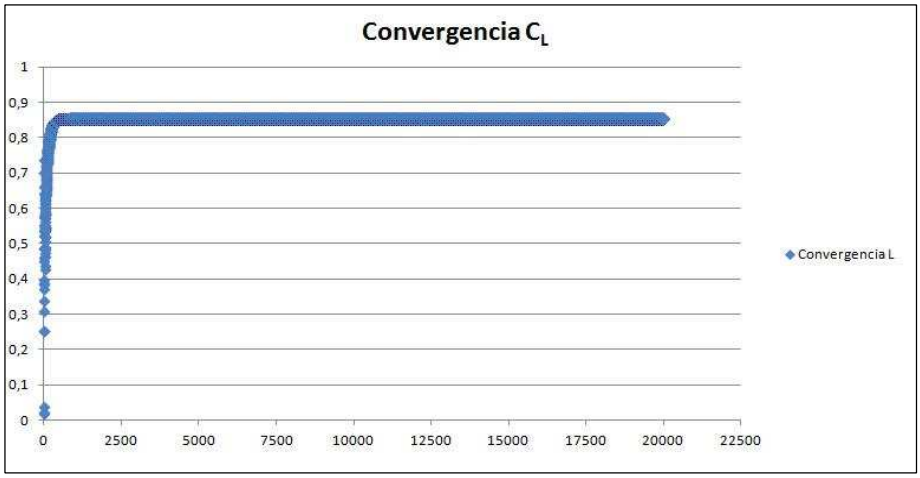
Solving p: Initial residual $= 1{,}318 \cdot 10^{-02}$, Final residual $= 4{,}564 \cdot 10^{-04}$, No Iterations $= 5$.

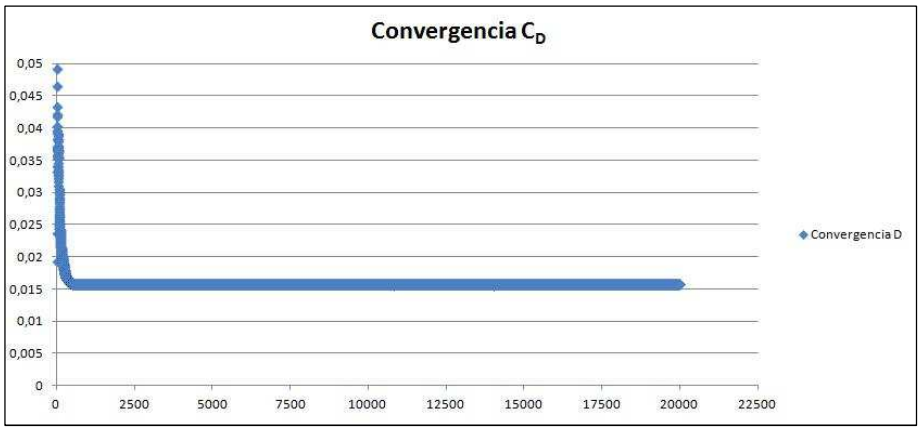
ExecutionTime: 6666,28 s. ClockTime: 6684 s.

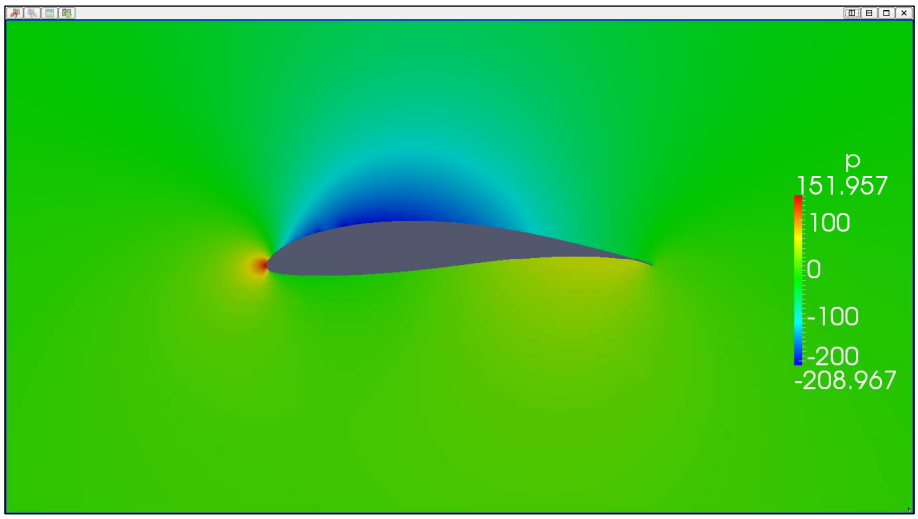
ForceCoeffs output:

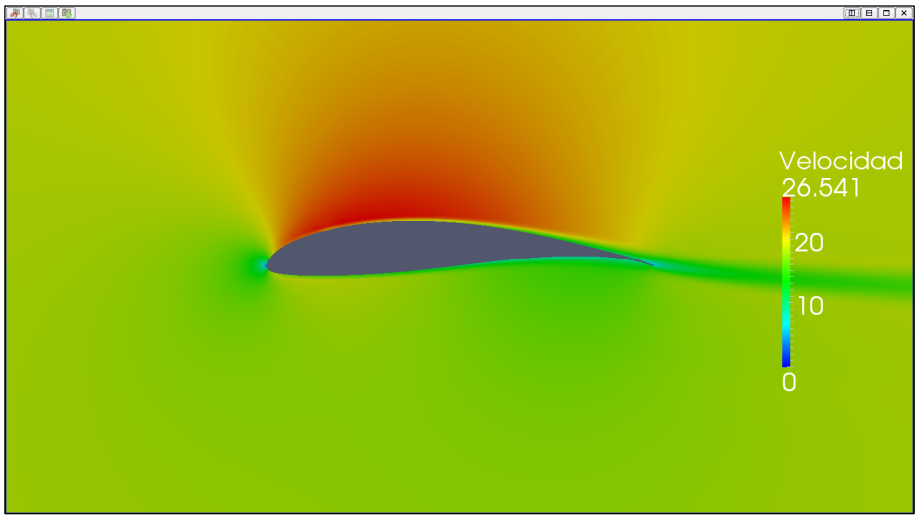
$C_L = 0{,}8537$

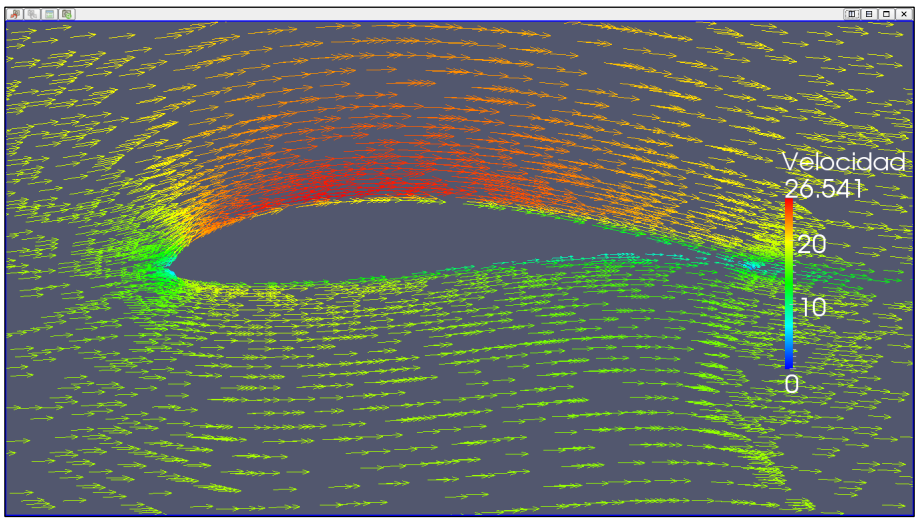
$C_D = 0{,}0157$






p
151.957
100
0
-100
-200
-208.967


Velocidad
26.541
20
10
0


Velocidad
26.541
20
10
0

II.3.6.2. Caso 2 (Ángulo de ataque 0º)

Time: 20000

Solving U_X: Initial residual $= 9{,}811 \cdot 10^{-06}$, Final residual $= 6{,}253 \cdot 10^{-08}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 8{,}525 \cdot 10^{-06}$, Final residual $= 6{,}755 \cdot 10^{-08}$, No Iterations $= 2$.

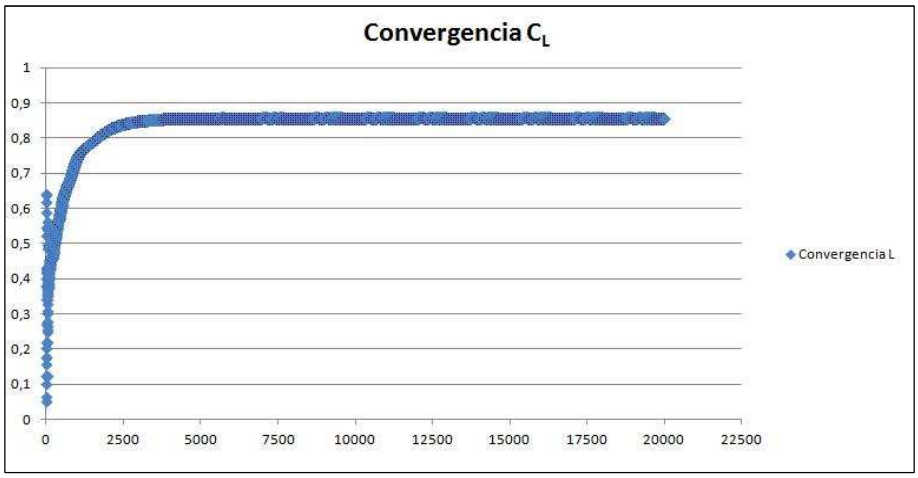
Solving p: Initial residual $= 1{,}312 \cdot 10^{-02}$, Final residual $= 1{,}085 \cdot 10^{-03}$, No Iterations $= 4$.

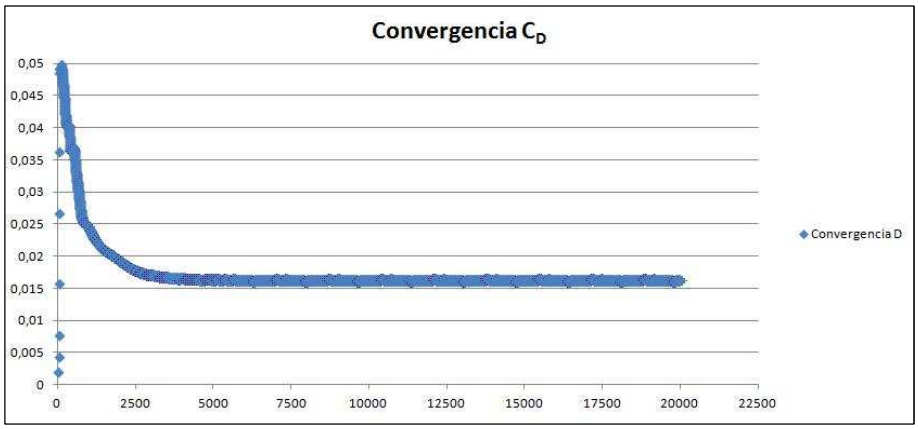
ExecutionTime: 6642,55 s. ClockTime: 6660 s.

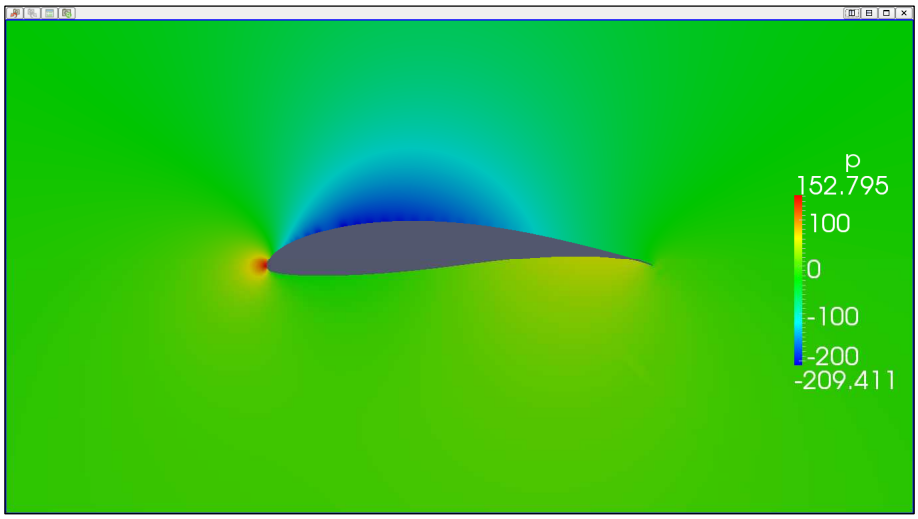
ForceCoeffs output:

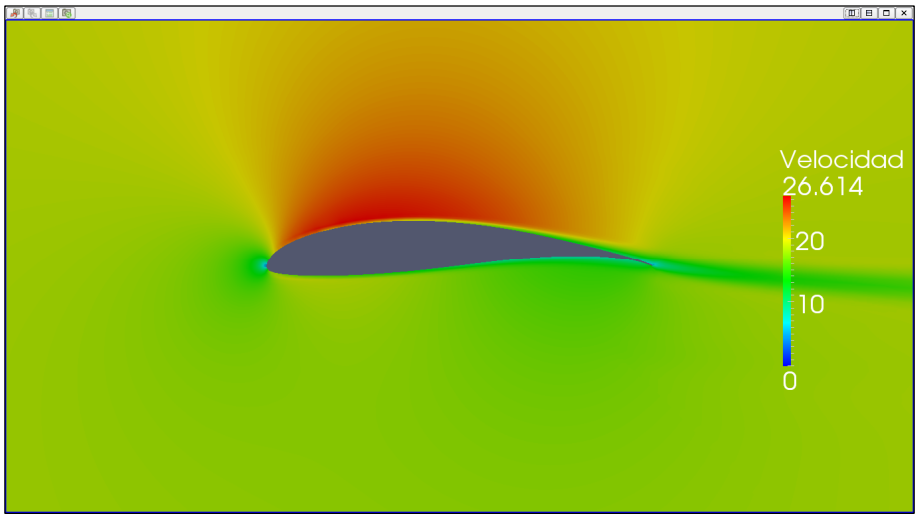
$C_L = 0{,}8567$

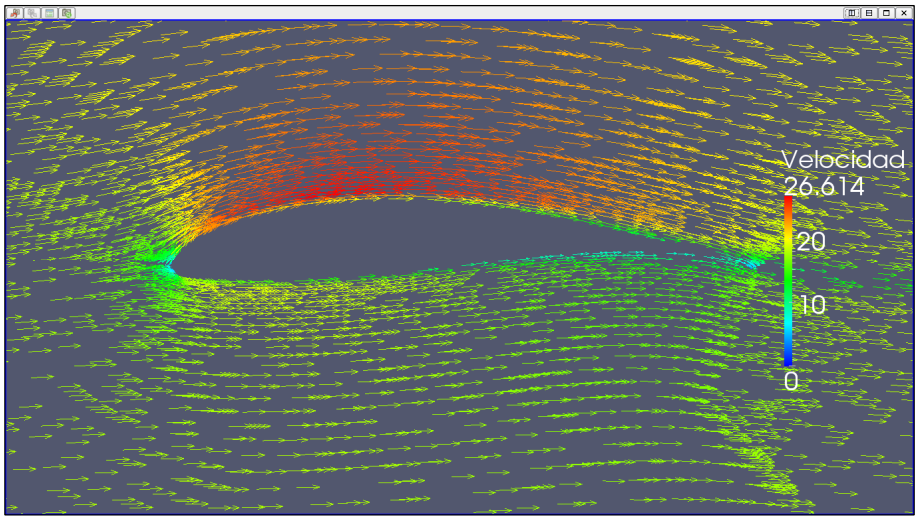
$C_D = 0{,}0162$






p
152.795
100
0
-100
-200
-209.411


Velocidad
26.614
20
10
0


Velocidad
26.614
20
10
0

II.3.6.3. Caso 3 (Ángulo de ataque 0º)

Time: 30000

Solving U_X: Initial residual $= 9{,}483 \cdot 10^{-05}$, Final residual $= 4{,}738 \cdot 10^{-06}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 8{,}336 \cdot 10^{-05}$, Final residual $= 6{,}064 \cdot 10^{-06}$, No Iterations $= 2$.

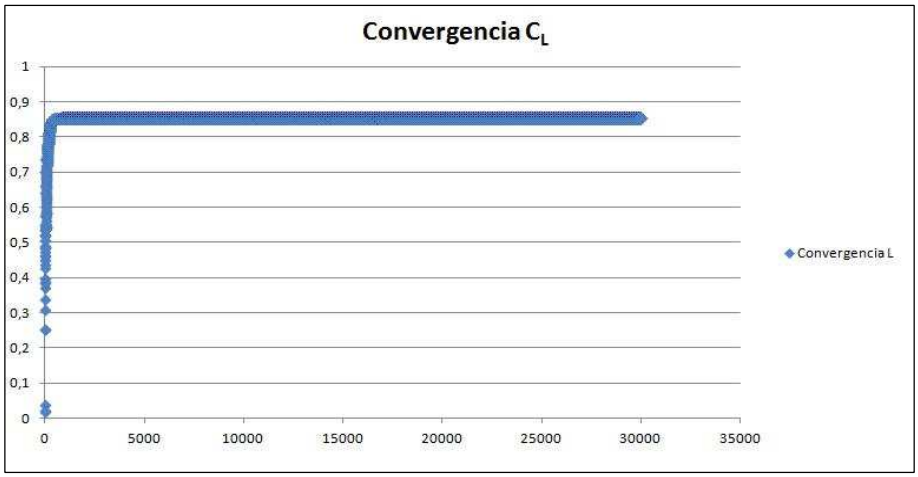
Solving p: Initial residual $= 1{,}314 \cdot 10^{-02}$, Final residual $= 4{,}566 \cdot 10^{-04}$, No Iterations $= 5$.

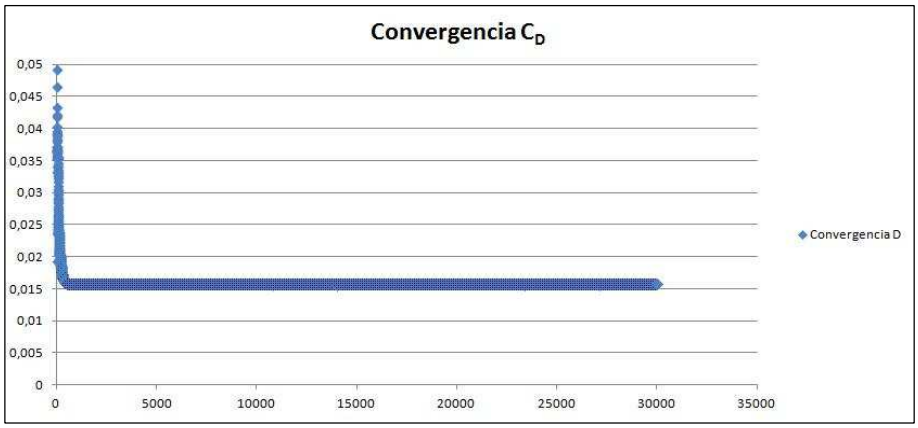
ExecutionTime: 9770,73 s. ClockTime: 9797 s.

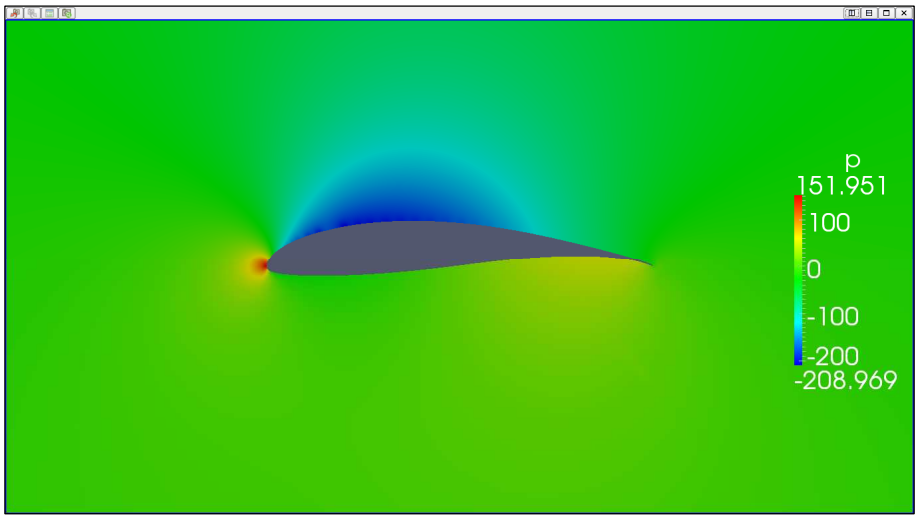
ForceCoeffs output:

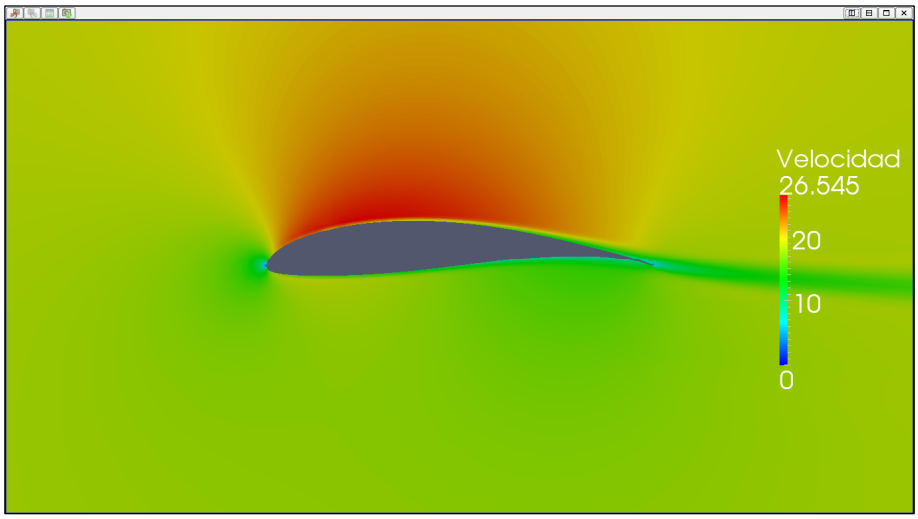
$C_L = 0{,}8537$

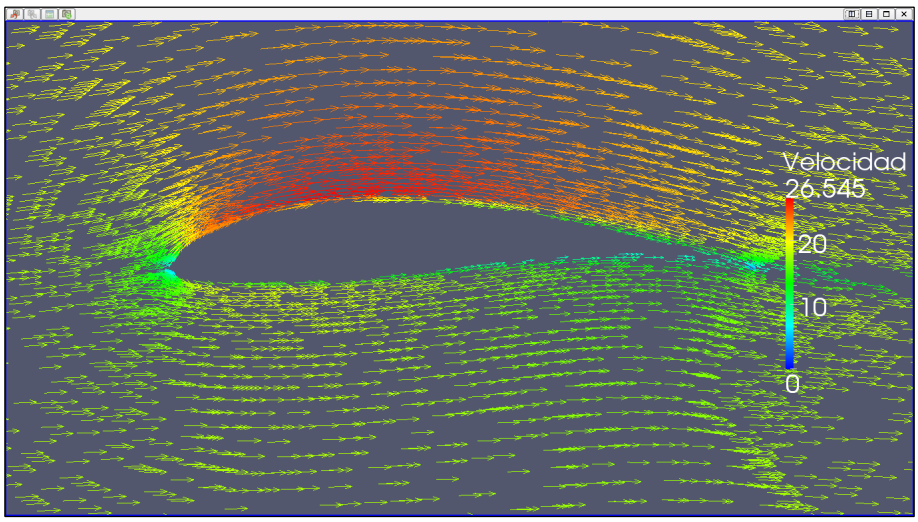
$C_D = 0{,}0157$






p
151.951
100
0
-100
-200
-208.969


Velocidad
26.545
20
10
0


Velocidad
26.545
20
10
0

II.3.7 Resultados ángulo de ataque 2º

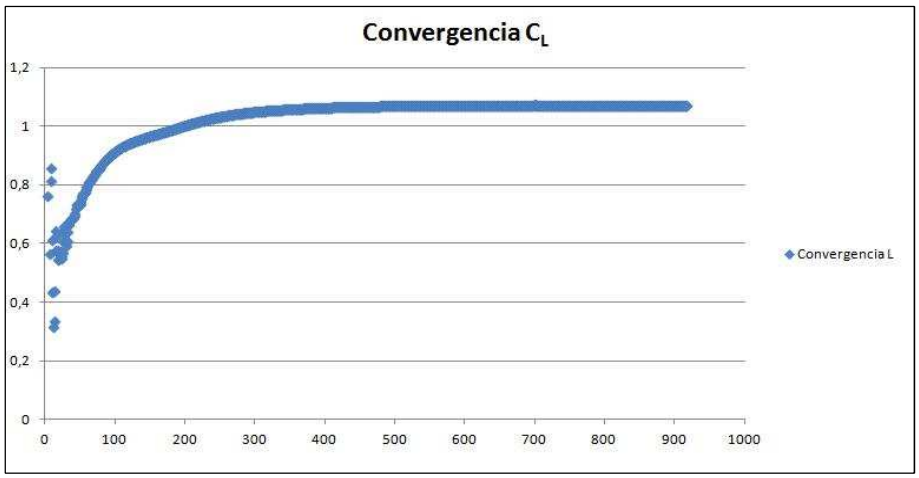
II.3.7.1. Caso 1 (Ángulo de ataque 2º)

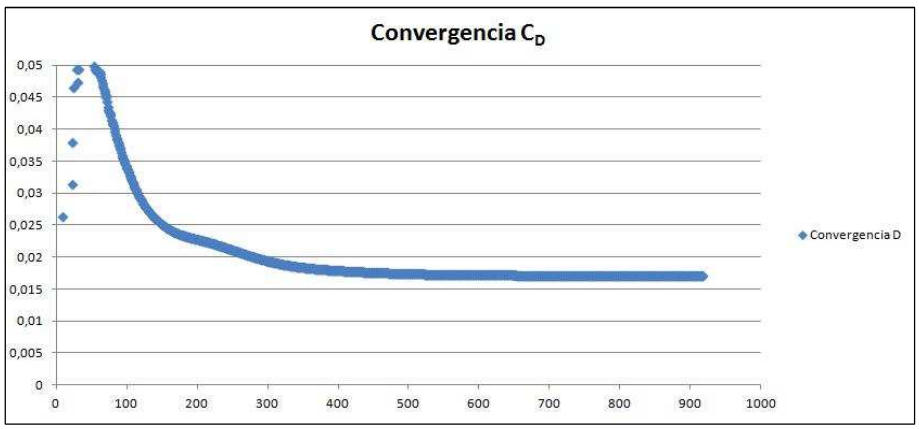
SIMPLE solution converged in 918 iterations.

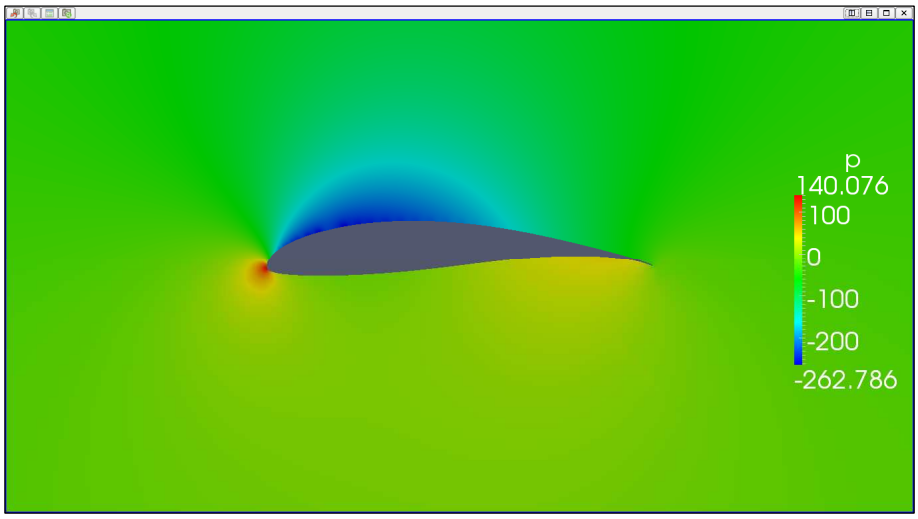
ForceCoeffs output:

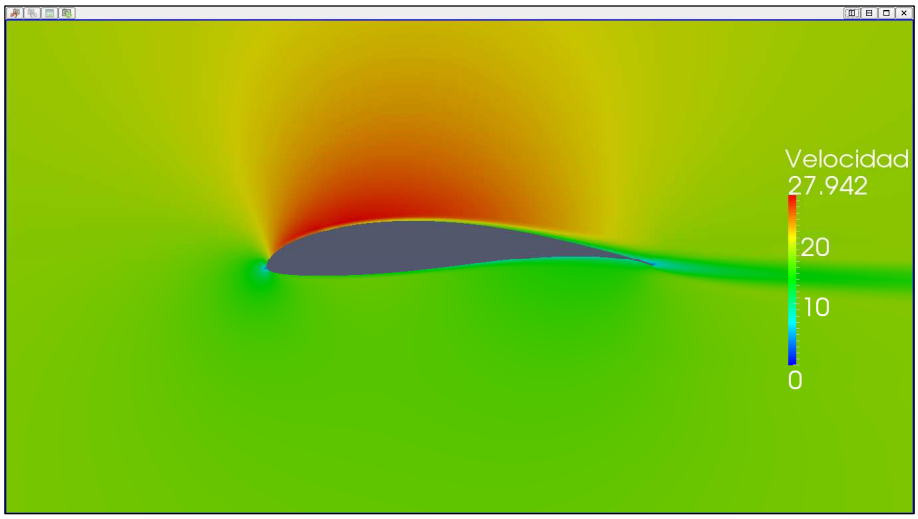
$C_L = 1{,}0718$

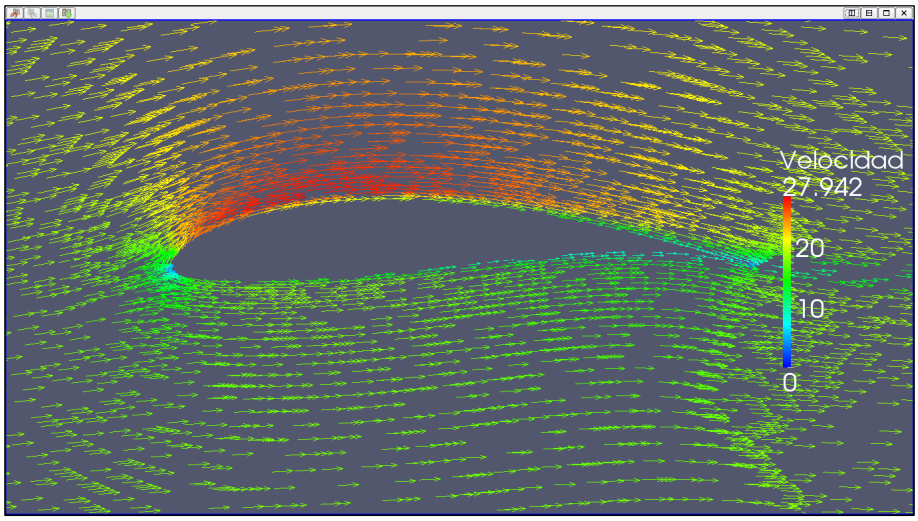
$C_D = 0{,}0171$






p
140.076
100
0
-100
-200
-262.786


Velocidad
27.942
20
10
0


Velocidad
27.942
20
10
0

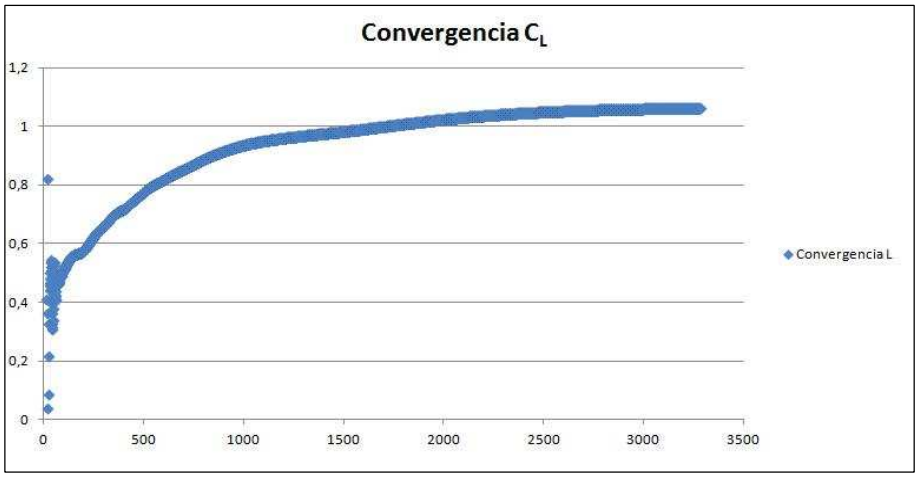
II.3.7.2. Caso 2 (Ángulo de ataque 2º)

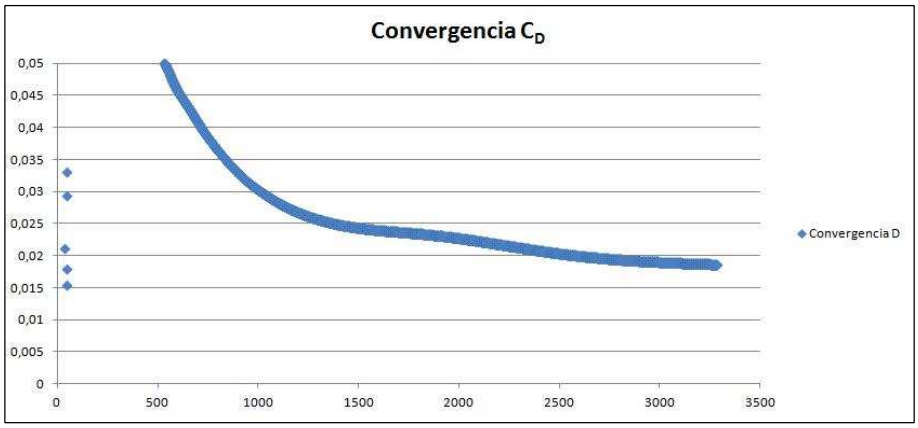
SIMPLE solution converged in 3285 iterations.

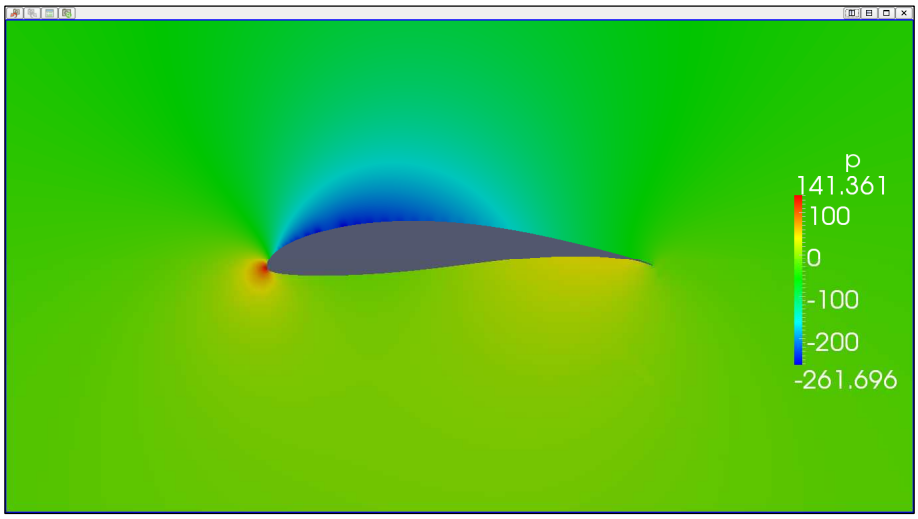
ForceCoeffs output:

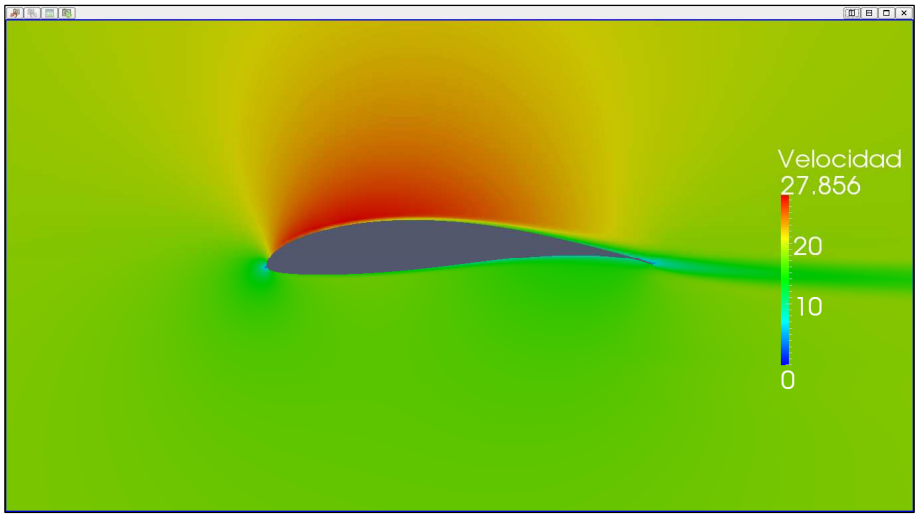
$C_L = 1{,}0639$

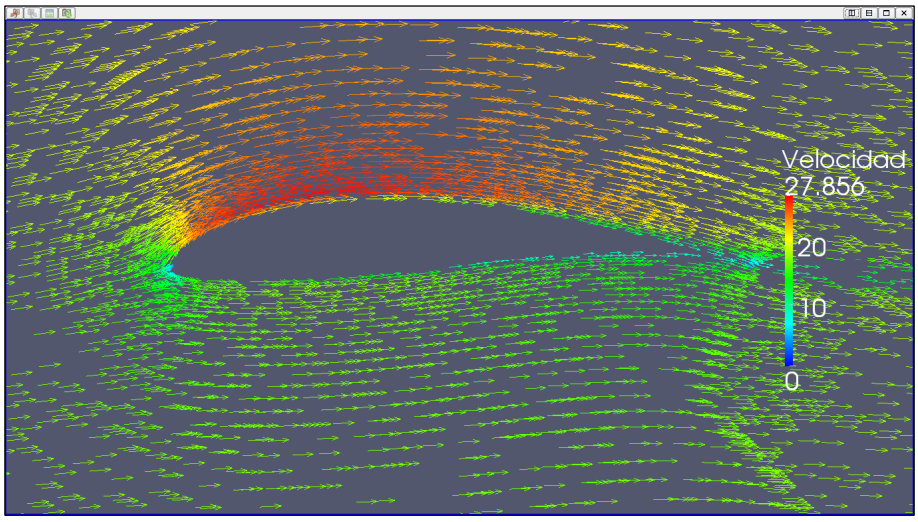
$C_D = 0{,}0186$






p
141.361
100
0
-100
-200
-261.696


Velocidad
27.856
20
10
0


Velocidad
27.856
20
10
0

II.3.7.3. Caso 3 (Ángulo de ataque 2º)

Time: 30000

Solving U_X: Initial residual $= 2{,}464 \cdot 10^{-06}$, Final residual $= 2{,}421 \cdot 10^{-07}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 2{,}810 \cdot 10^{-06}$, Final residual $= 1{,}149 \cdot 10^{-07}$, No Iterations $= 4$.

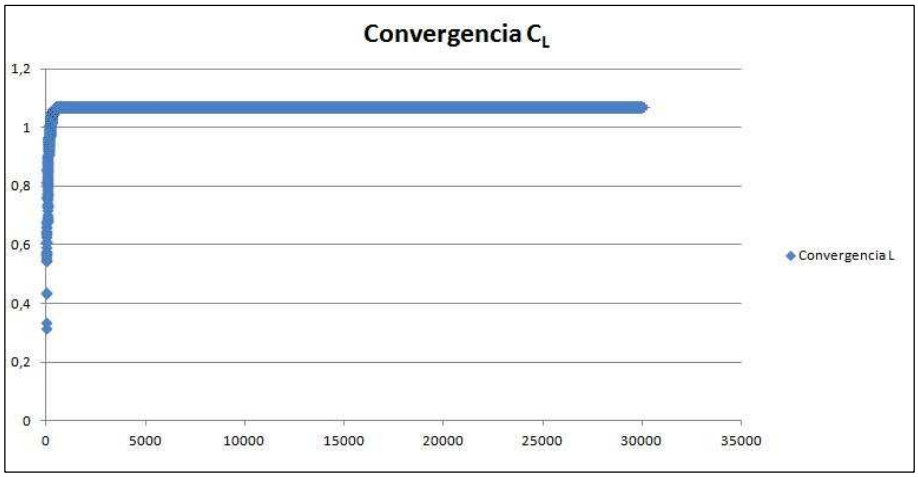
Solving p: Initial residual $= 1{,}780 \cdot 10^{-04}$, Final residual $= 8{,}522 \cdot 10^{-05}$, No Iterations $= 5$.

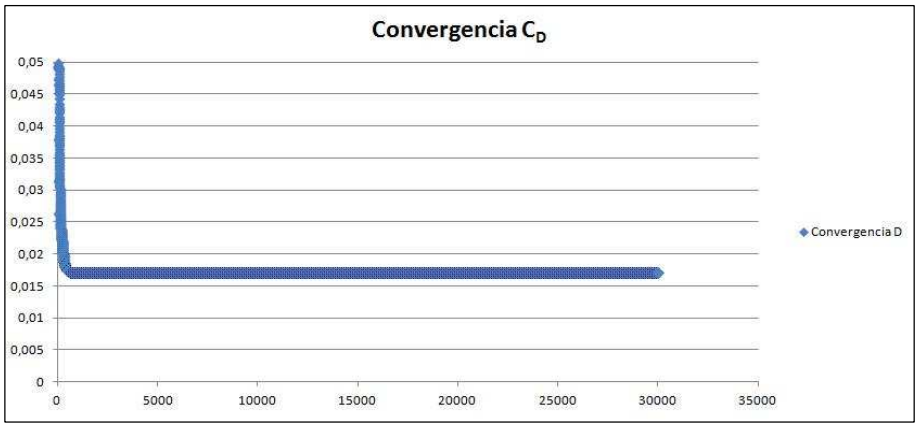
ExecutionTime: 9933,27 s. ClockTime: 9966 s.

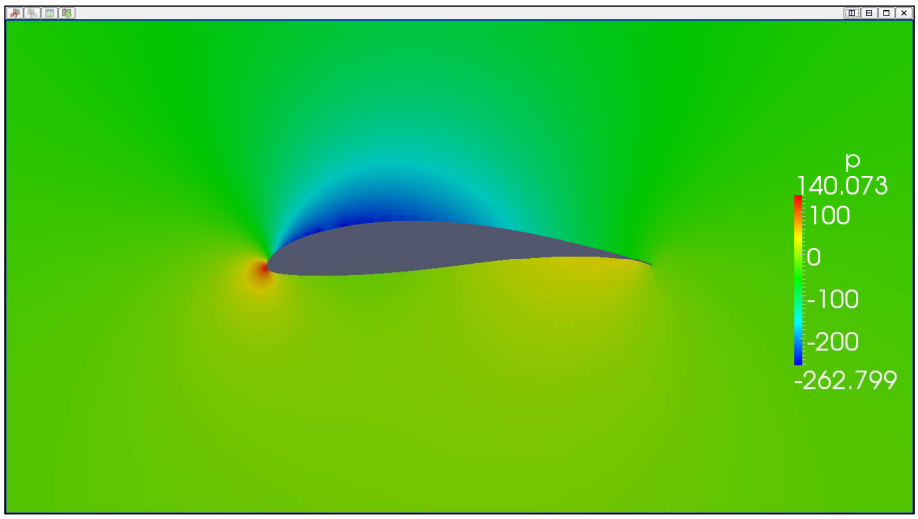
ForceCoeffs output:

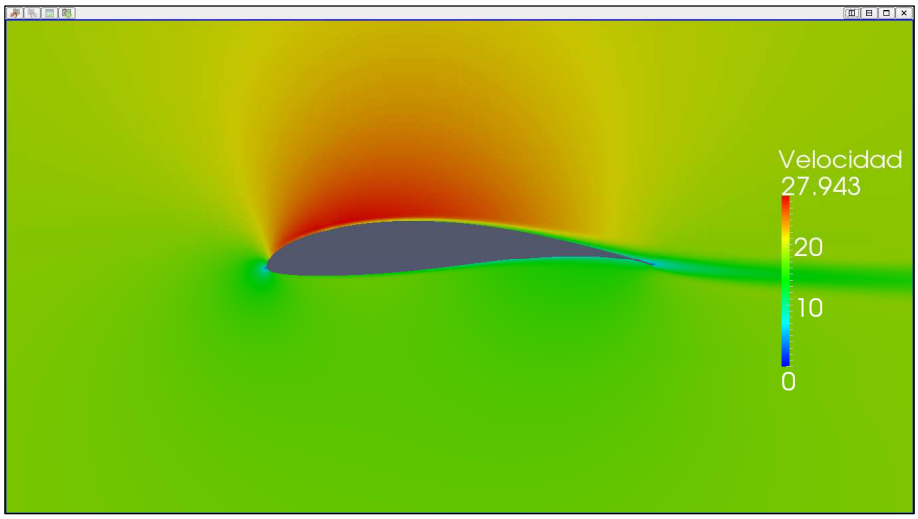
$C_L = 1{,}0718$

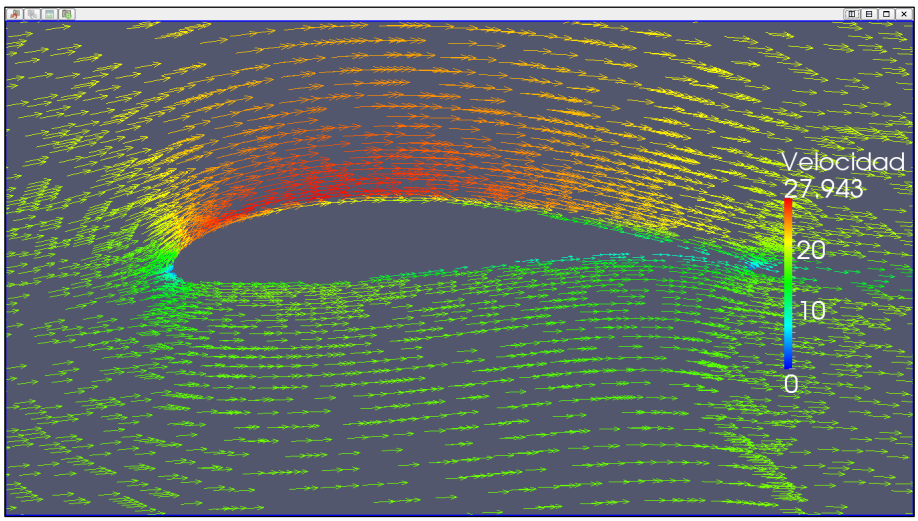
$C_D = 0{,}0171$






p
140.073
100
0
-100
-200
-262.799


Velocidad
27.943
20
10
0


Velocidad
27.943
20
10
0

II.3.8. Resultados ángulo de ataque 4º

II.3.8.1. Caso 1 (Ángulo de ataque 4º)

Time: 20000

Solving U_X: Initial residual $= 9{,}450 \cdot 10^{-06}$, Final residual $= 3{,}833 \cdot 10^{-07}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 1{,}002 \cdot 10^{-05}$, Final residual $= 4{,}436 \cdot 10^{-07}$, No Iterations $= 4$.

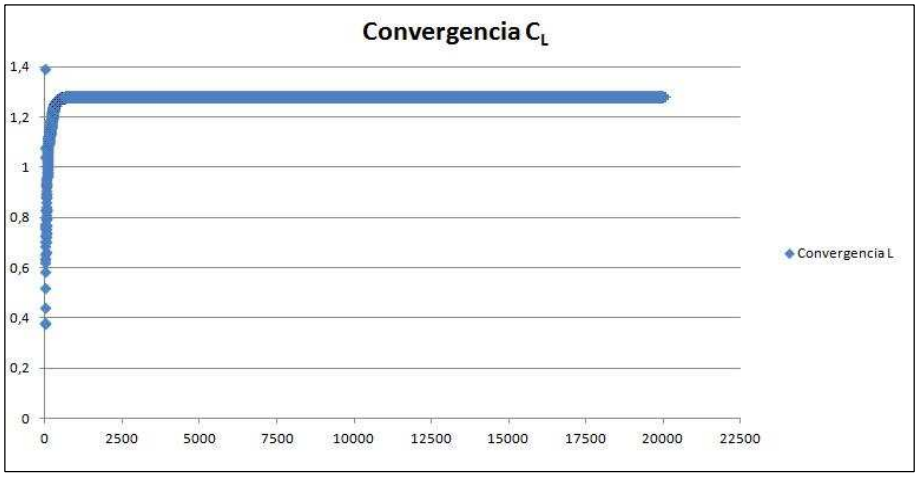
Solving p: Initial residual $= 6{,}142 \cdot 10^{-04}$, Final residual $= 3{,}814 \cdot 10^{-05}$, No Iterations $= 4$.

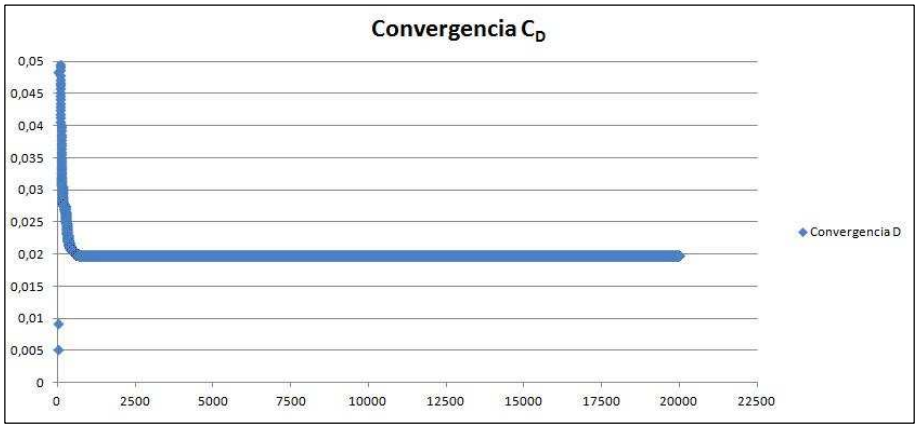
ExecutionTime: 6881.39 s. ClockTime: 6901 s.

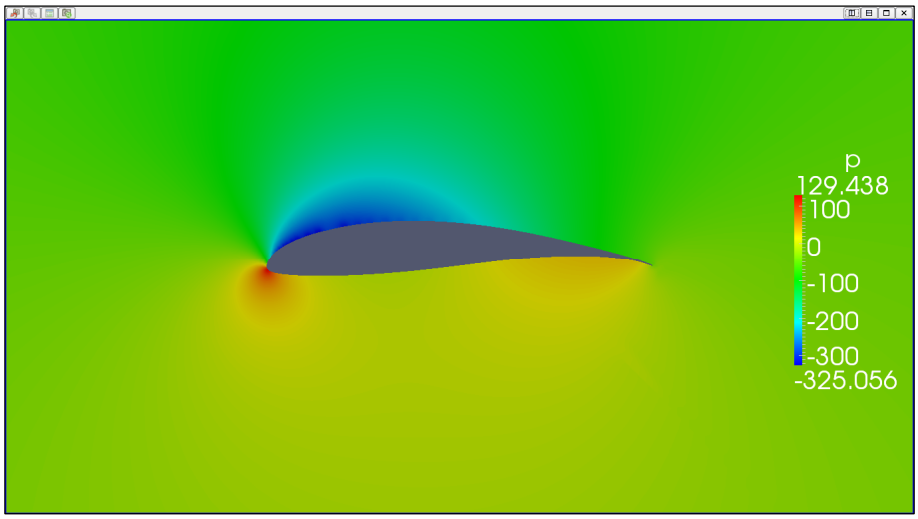
ForceCoeffs output:

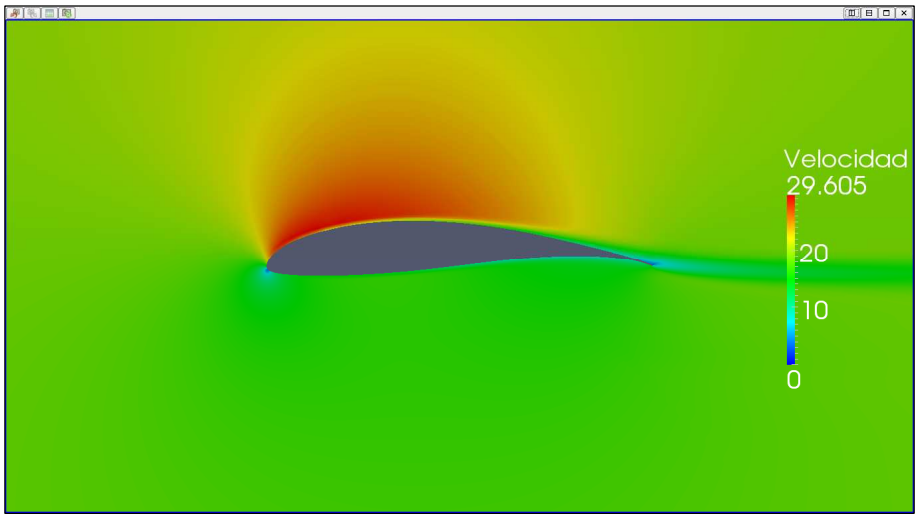
$C_L = 1{,}2796$

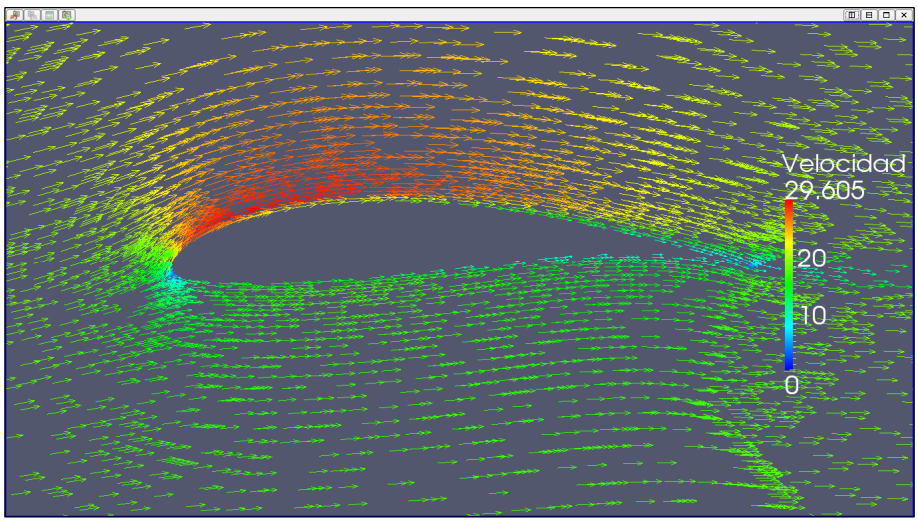
$C_D = 0{,}0197$






p
129.438
100
0
-100
-200
-300
-325.056


Velocidad
29.605
20
10
0


Velocidad
29.605
20
10
0

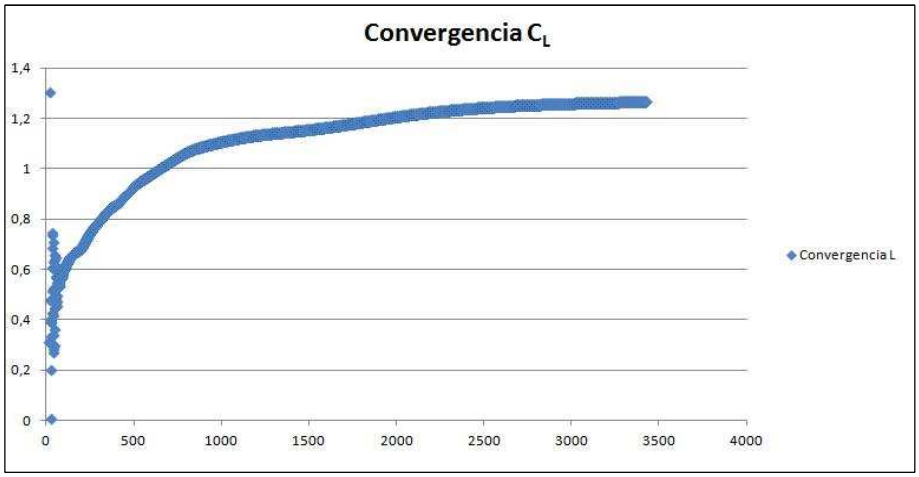
II.3.8.2. Caso 2 (Ángulo de ataque 4º)

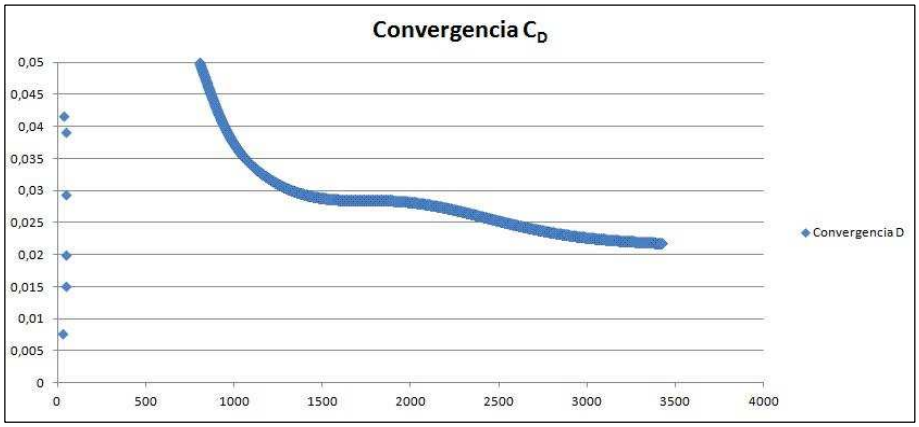
SIMPLE solution converged in 3426 iterations.

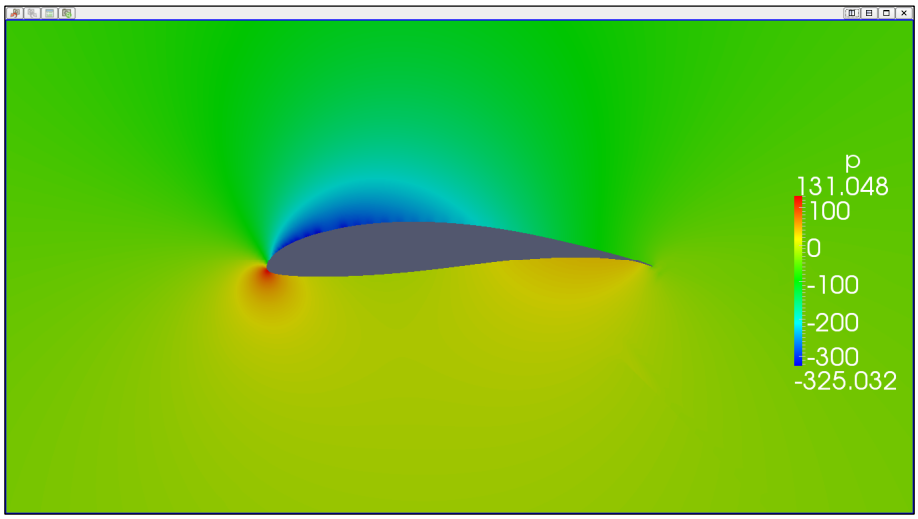
ForceCoeffs output:

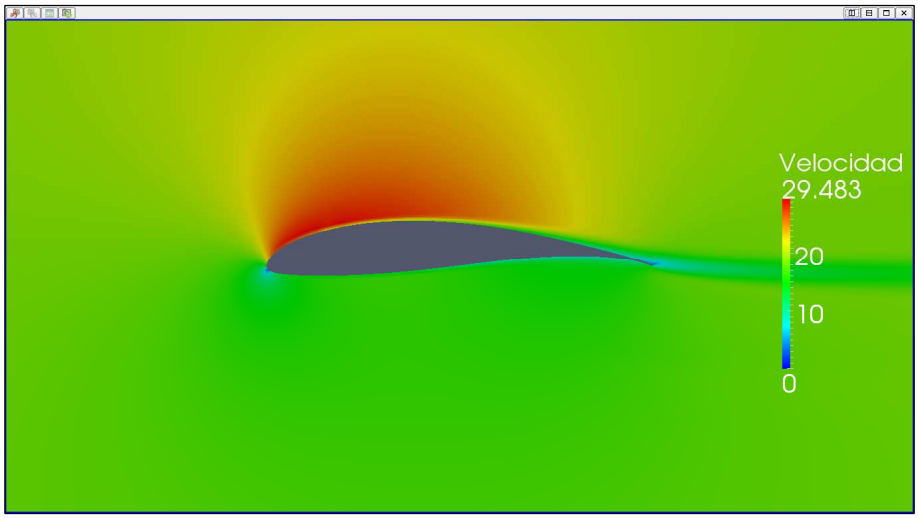
$C_L = 1{,}2668$

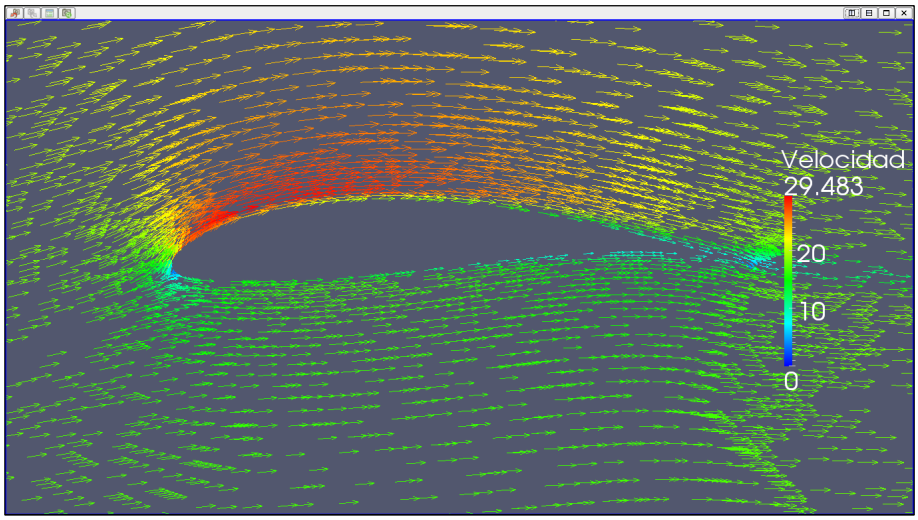
$C_D = 0{,}0218$






p
131.048
100
0
-100
-200
-300
-325.032


Velocidad
29.483
20
10
0


Velocidad
29.483
20
10
0

II.3.8.3. Caso 3 (Ángulo de ataque 4º)

Time: 30000

Solving U_X: Initial residual $= 7{,}039 \cdot 10^{-06}$, Final residual $= 6{,}406 \cdot 10^{-07}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 8{,}363 \cdot 10^{-06}$, Final residual $= 3{,}303 \cdot 10^{-07}$, No Iterations $= 4$.

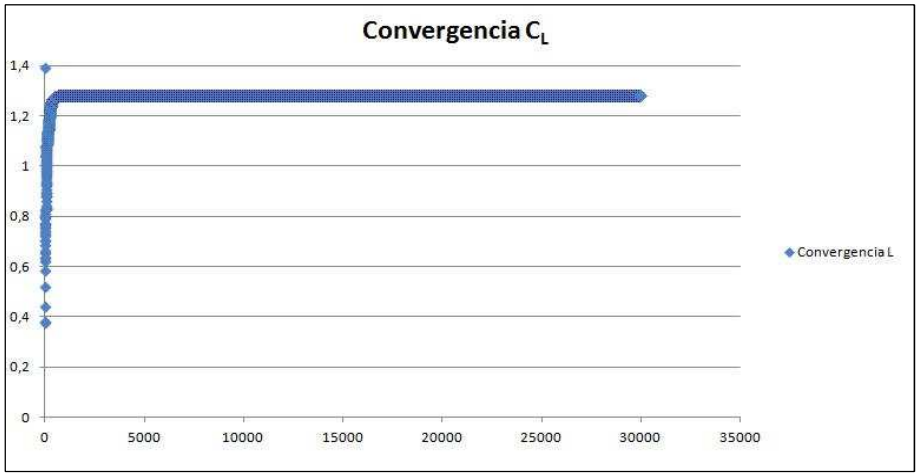
Solving p: Initial residual $= 7{,}027 \cdot 10^{-04}$, Final residual $= 3{,}040 \cdot 10^{-05}$, No Iterations $= 4$.

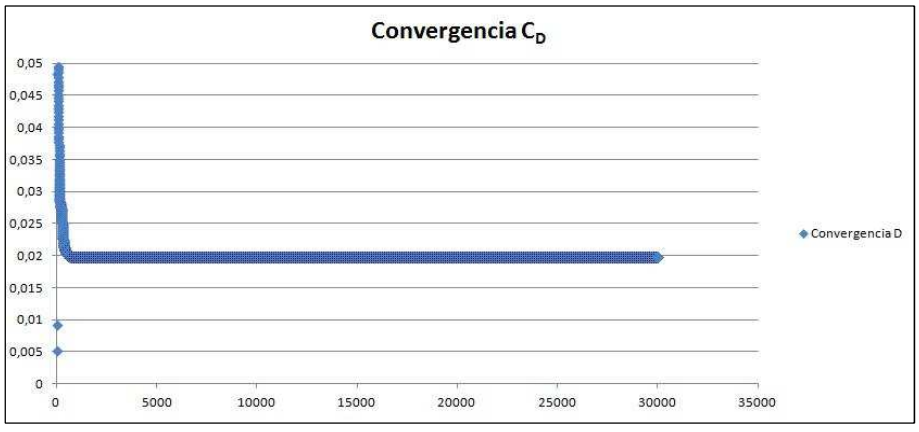
ExecutionTime: 9894,86 s. ClockTime: 9925 s.

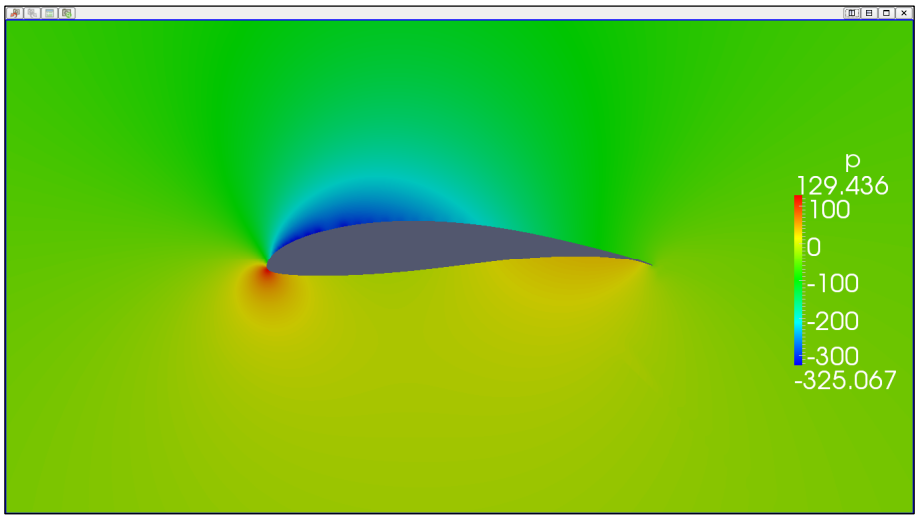
ForceCoeffs output:

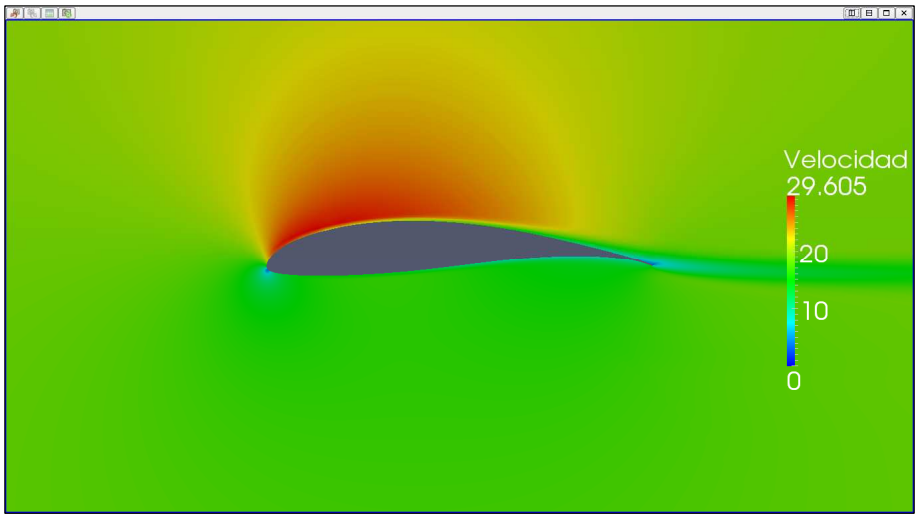
$C_L = 1{,}2797$

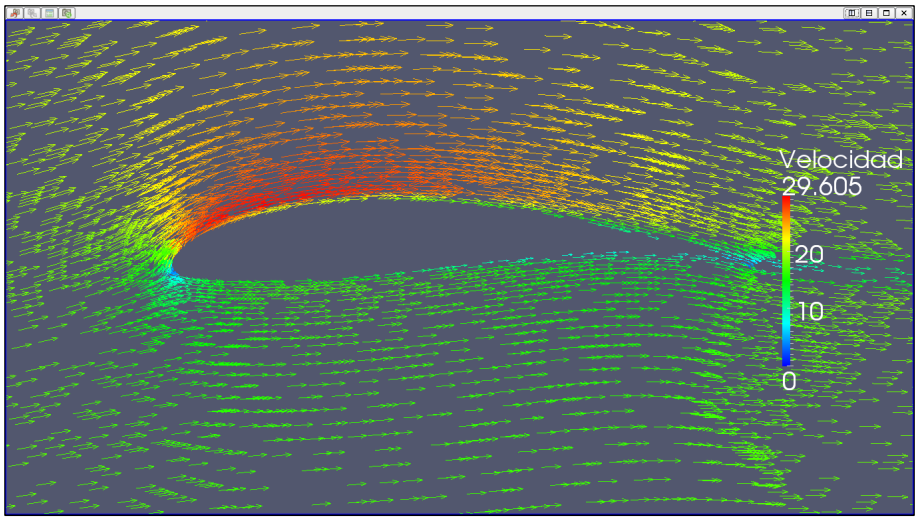
$C_D = 0{,}0197$






p
129.436
100
0
-100
-200
-300
-325.067


Velocidad
29.605
20
10
0


Velocidad
29.605
20
10
0

II.3.9. Resultados ángulo de ataque 6º

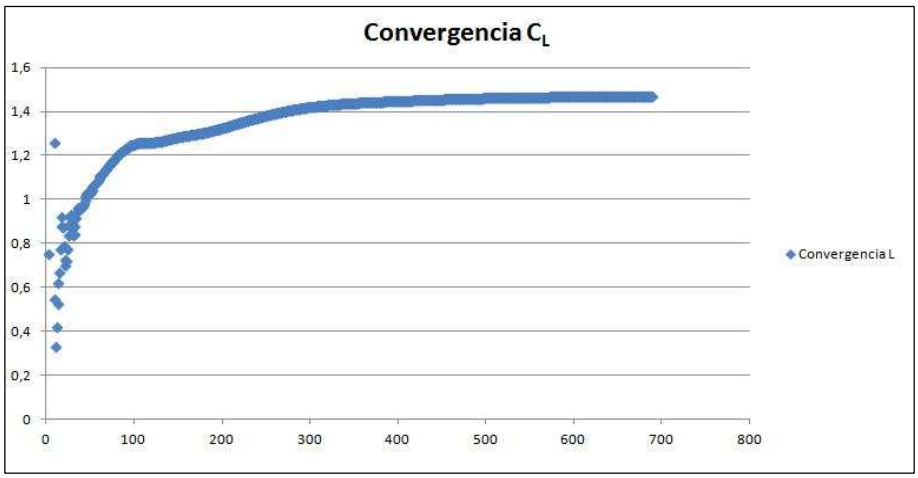
II.3.9.1. Caso 1 (Ángulo de ataque 6º)

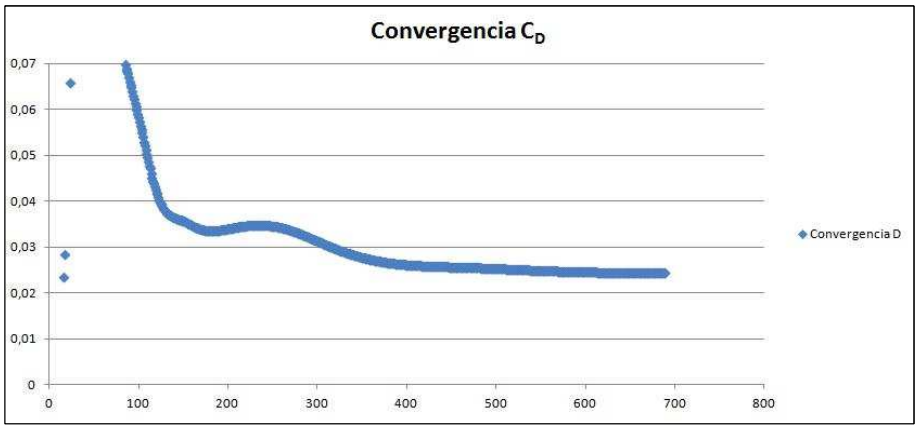
SIMPLE solution converged in 689 iterations.

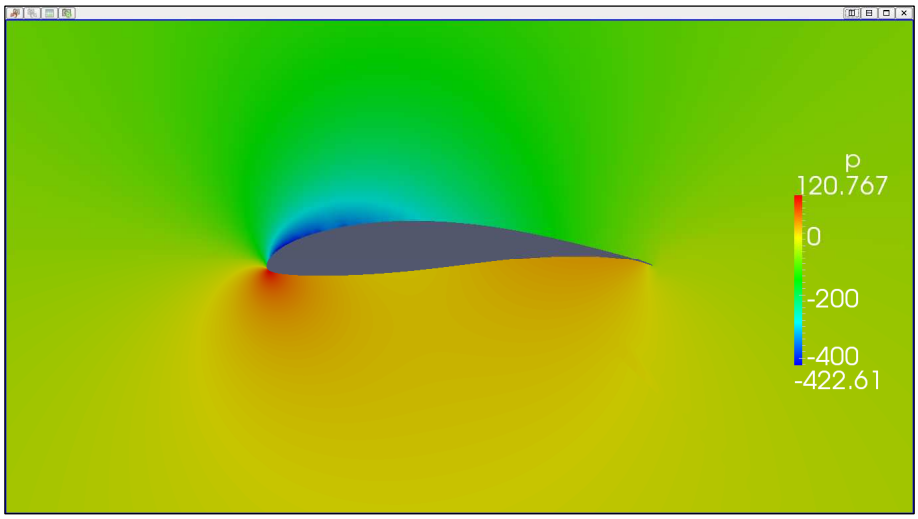
ForceCoeffs output:

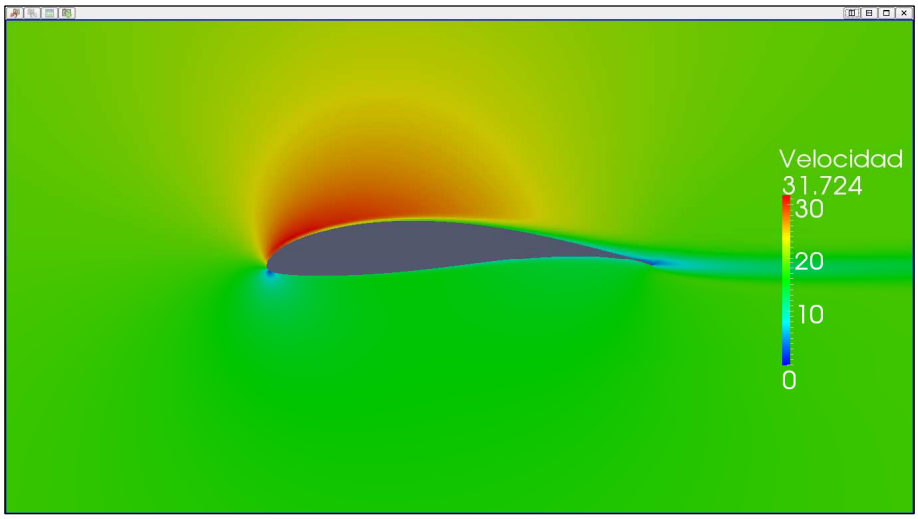
$C_L = 1{,}4693$

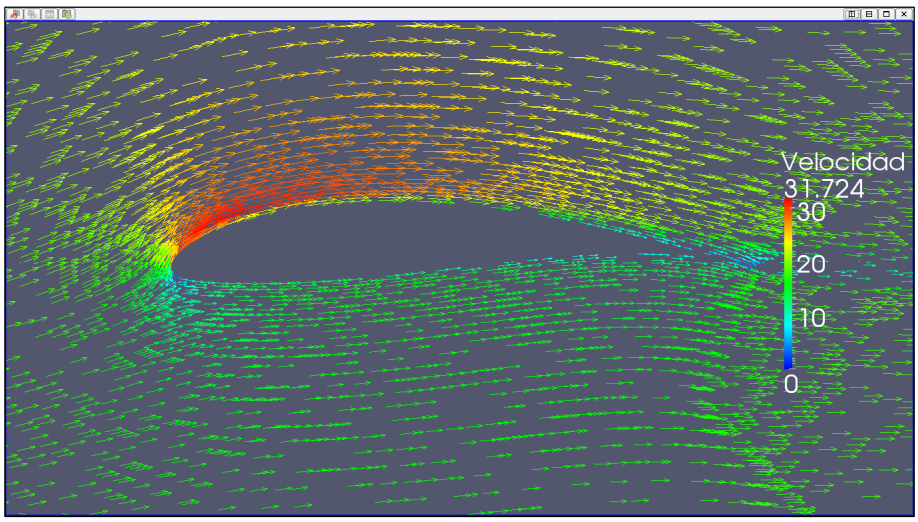
$C_D = 0{,}0243$






p
120.767
0
-200
-400
-422.61


Velocidad
31.724
30
20
10
0


Velocidad
31.724
30
20
10
0

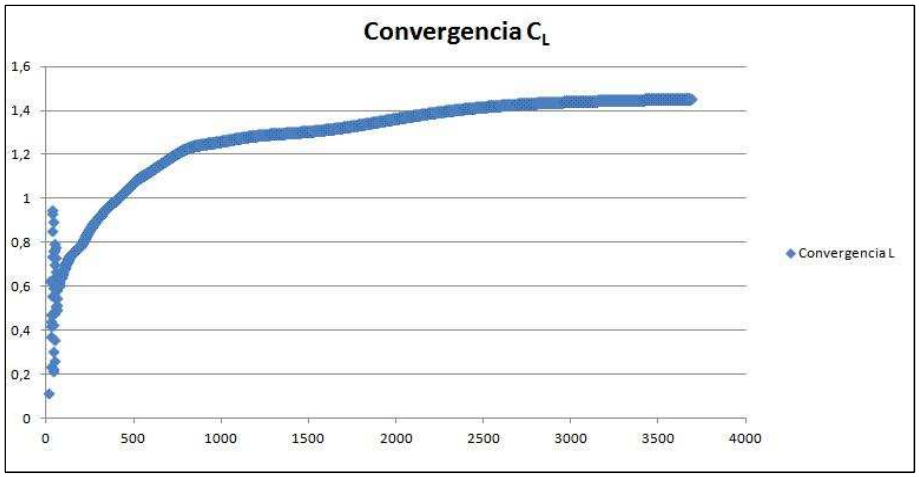
II.3.9.2. Caso 2 (Ángulo de ataque 6º)

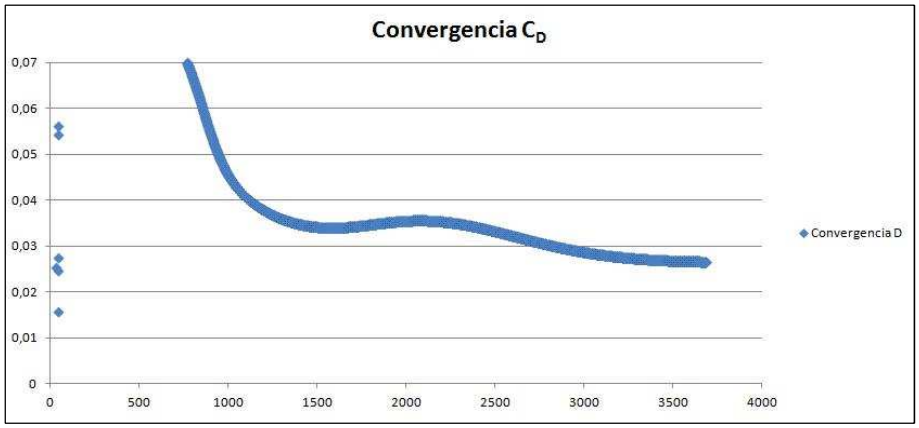
SIMPLE solution converged in 3687 iterations.

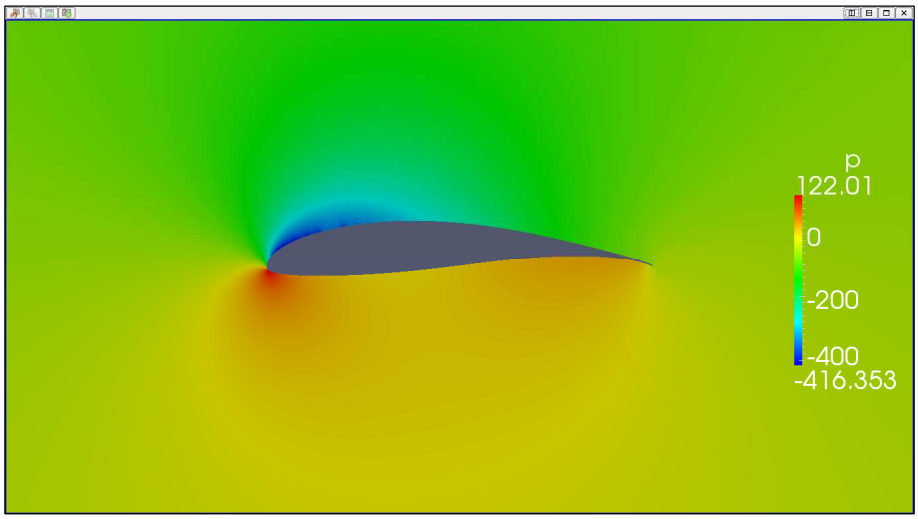
ForceCoeffs output:

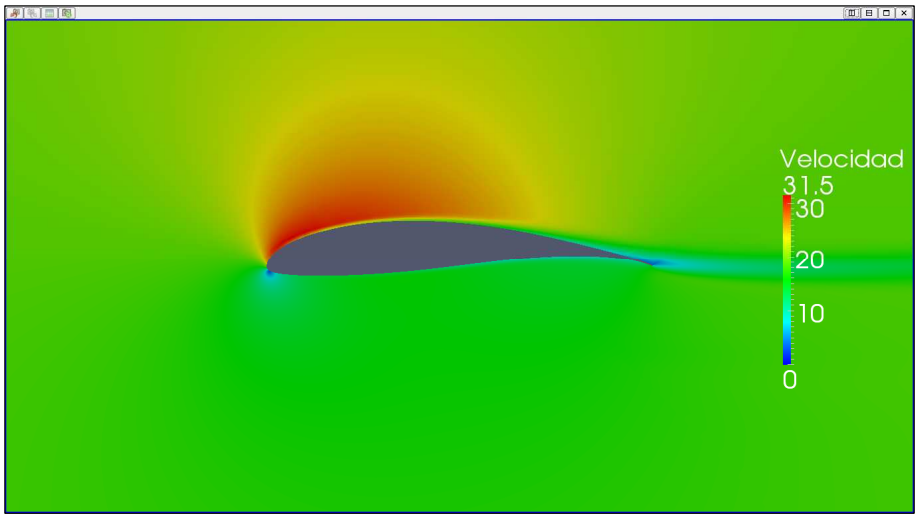
$C_L = 1{,}4548$

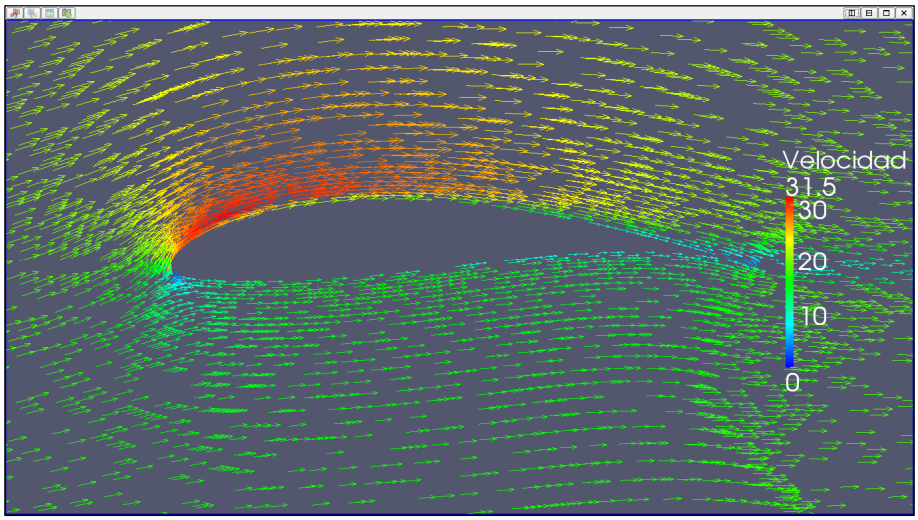
$C_D = 0{,}0266$






p
122.01
0
-200
-400
-416.353


Velocidad
31.5
30
20
10
0


Velocidad
31.5
30
20
10
0

II.3.9.3. Caso 3 (Ángulo de ataque 6º)

Time: 30000

Solving U_X: Initial residual $= 7{,}106 \cdot 10^{-06}$, Final residual $= 5{,}277 \cdot 10^{-07}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 7{,}109 \cdot 10^{-06}$, Final residual $= 6{,}568 \cdot 10^{-07}$, No Iterations $= 2$.

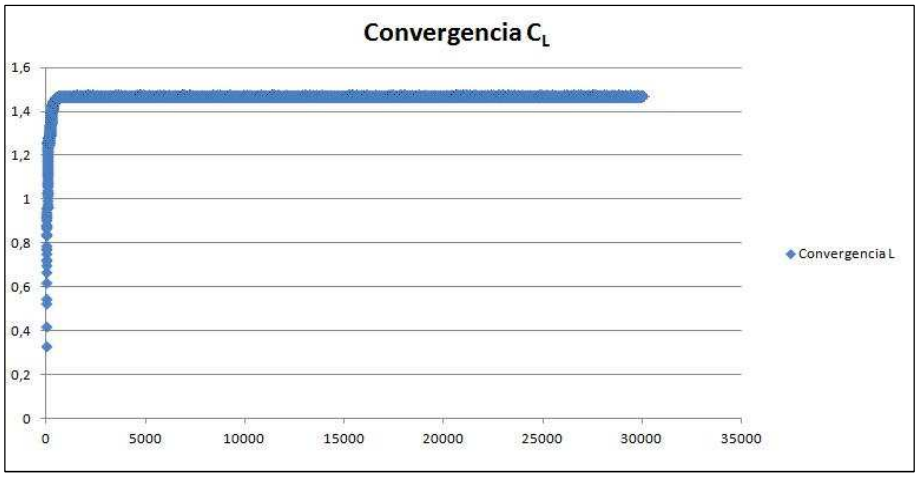
Solving p: Initial residual $= 7{,}464 \cdot 10^{-04}$, Final residual $= 4{,}652 \cdot 10^{-05}$, No Iterations $= 4$.

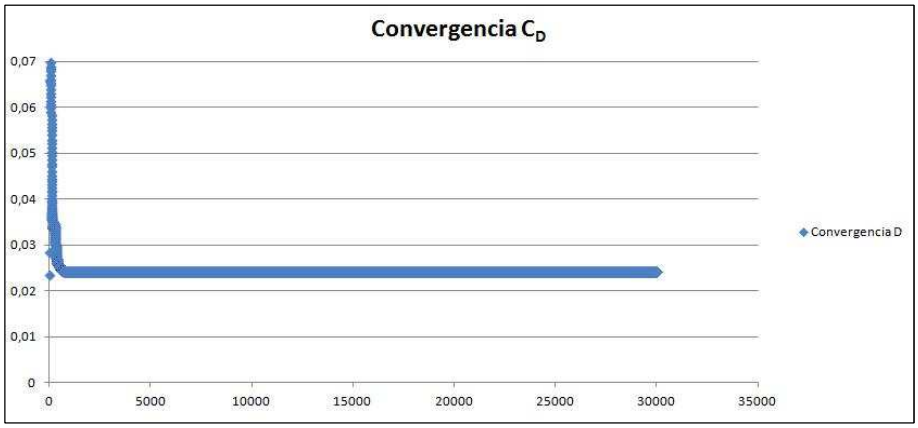
ExecutionTime: 10258,30 s. ClockTime: 10296 s.

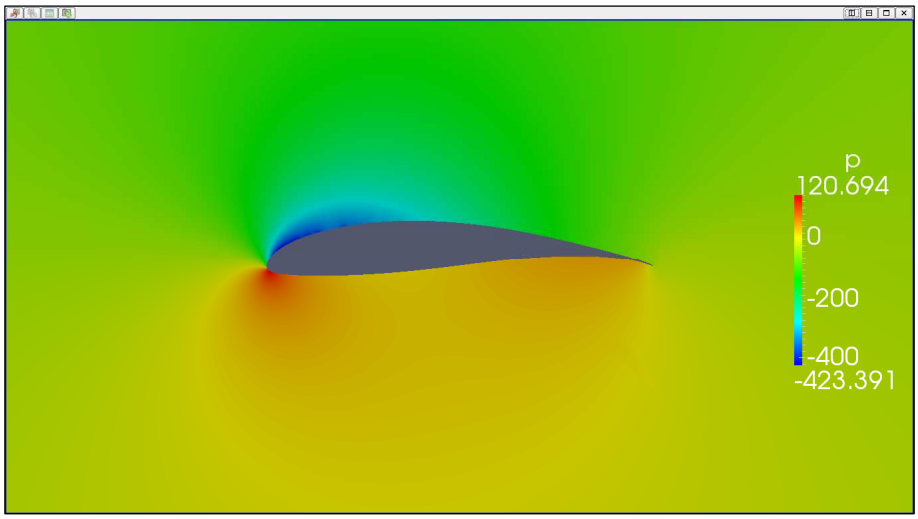
ForceCoeffs output:

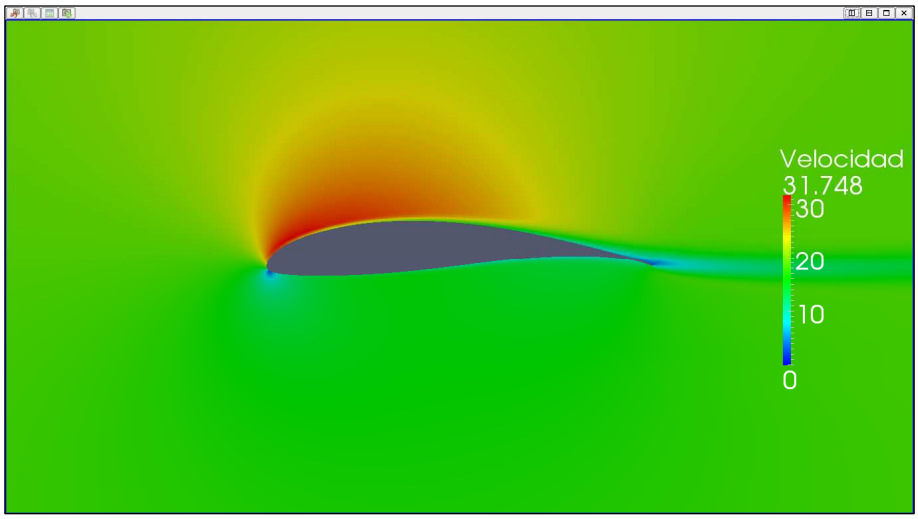
$C_L = 1{,}4713$

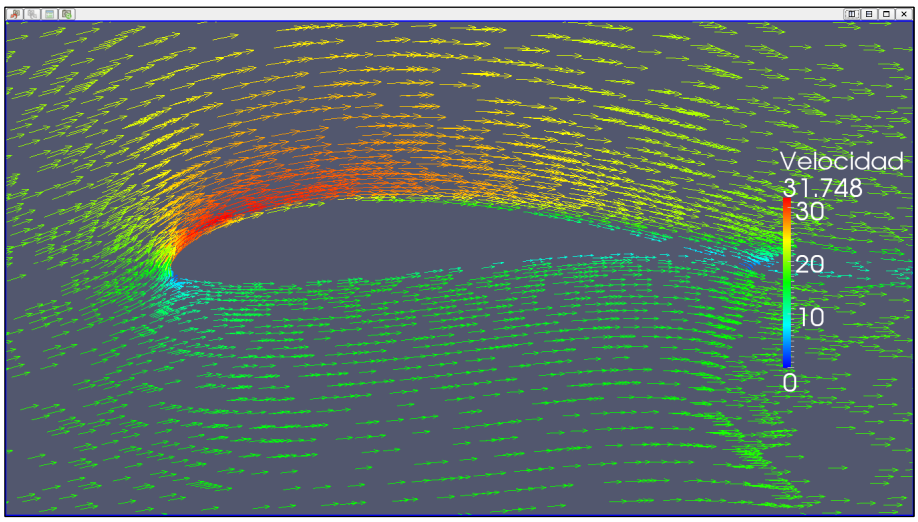
$C_D = 0{,}0241$






p
120.694
0
-200
-400
-423.391


Velocidad
31.748
30
20
10
0


Velocidad
31.748
30
20
10
0

II.3.10. Resultados ángulo de ataque 8º

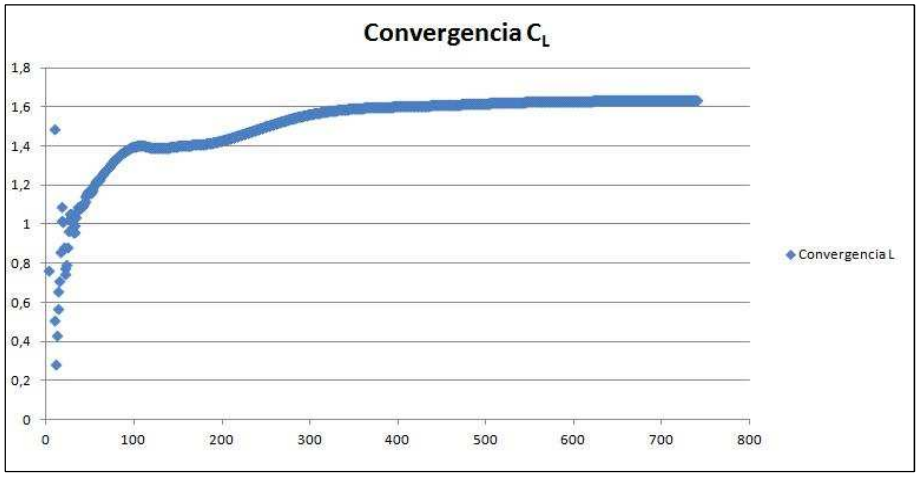
II.3.10.1. Caso 1 (Ángulo de ataque 8º)

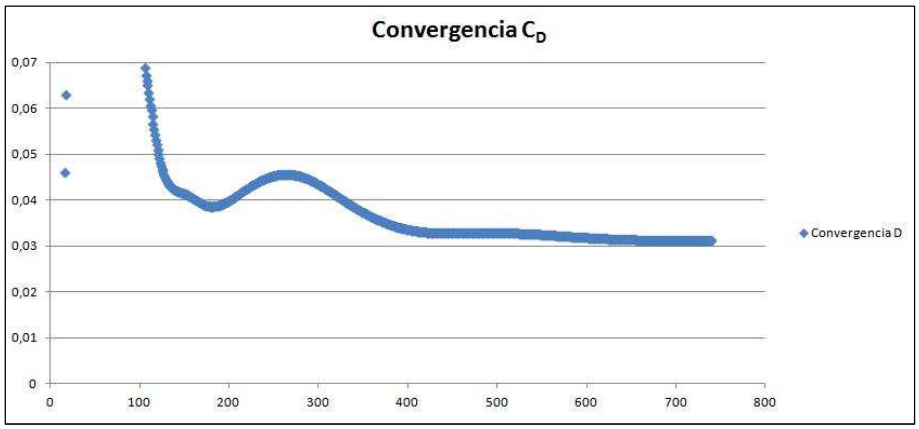
SIMPLE solution converged in 740 iterations.

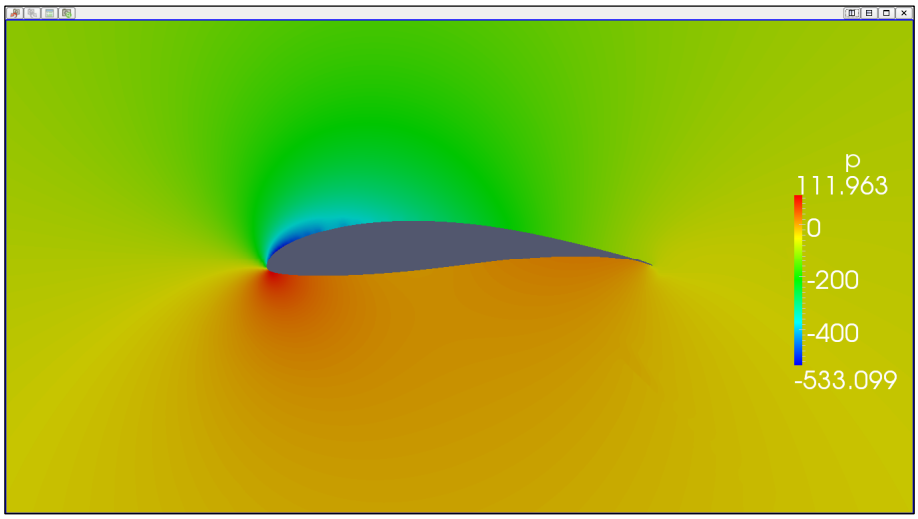
ForceCoeffs output:

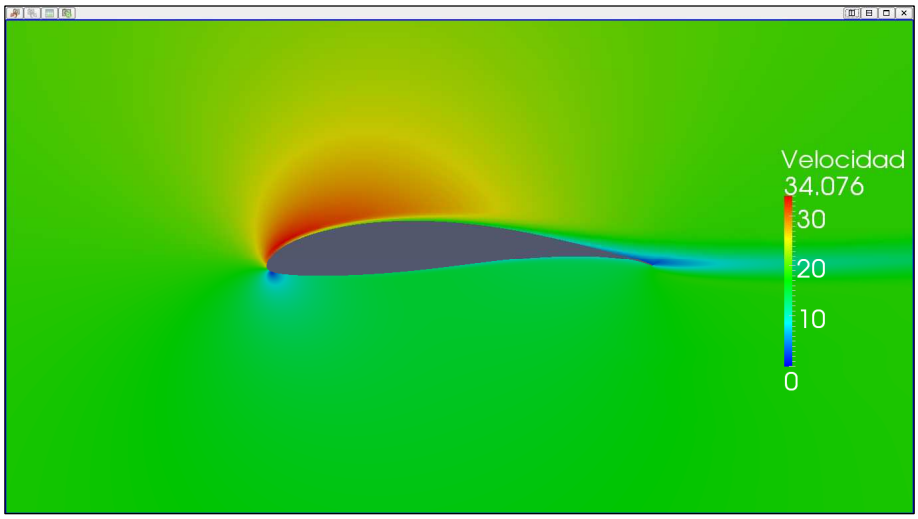
$C_L = 1{,}6349$

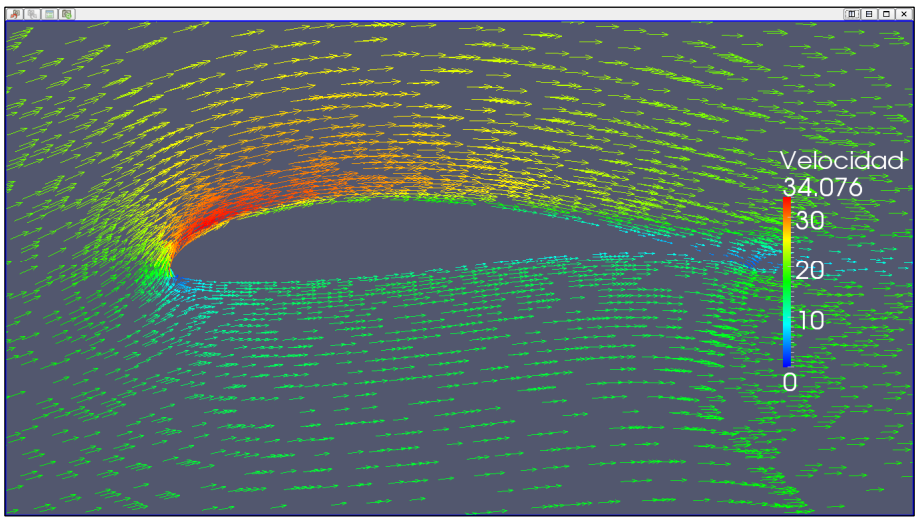
$C_D = 0{,}0311$






p
111.963
0
-200
-400
-533.099


Velocidad
34.076
30
20
10
0


Velocidad
34.076
30
20
10
0

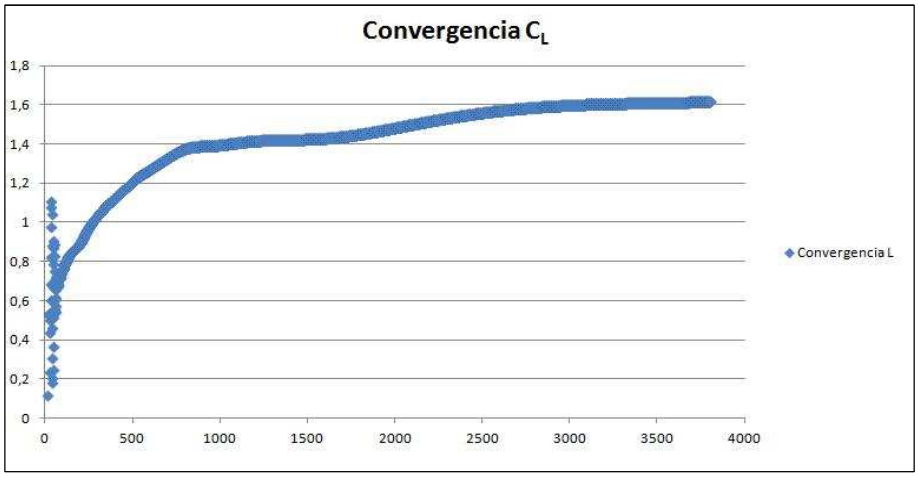
II.3.10.2. Caso 2 (Ángulo de ataque 8º)

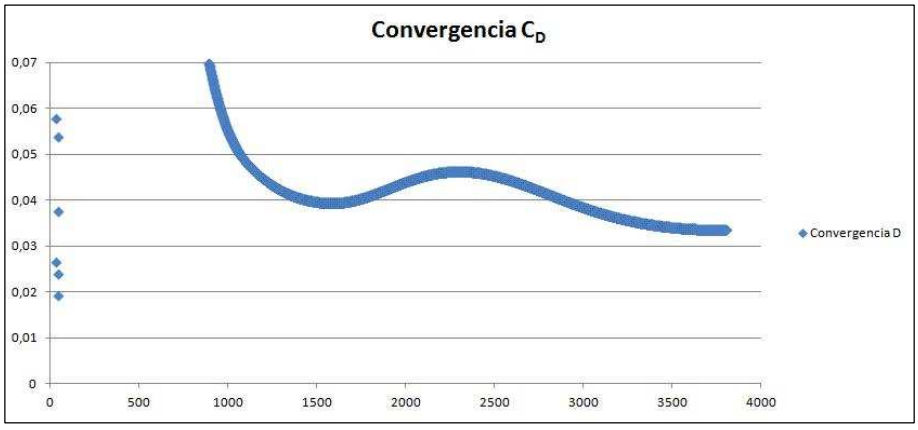
SIMPLE solution converged in 3805 iterations.

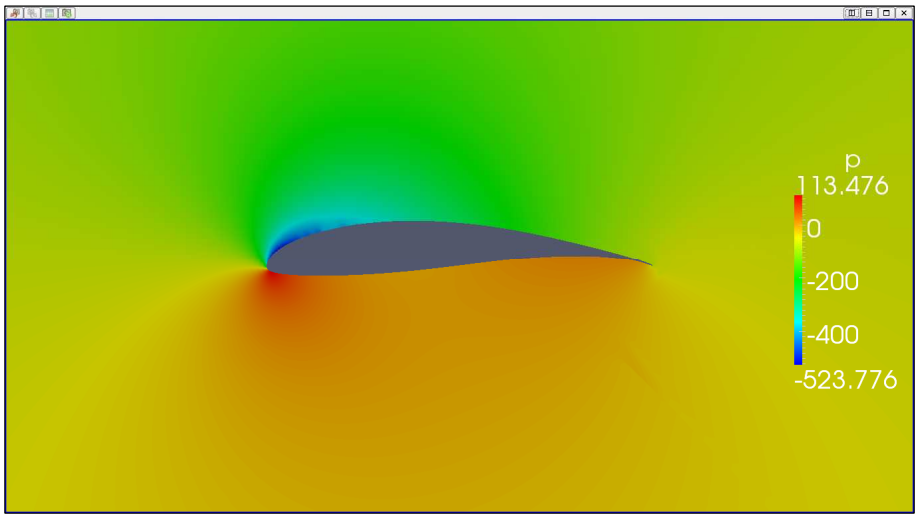
ForceCoeffs output:

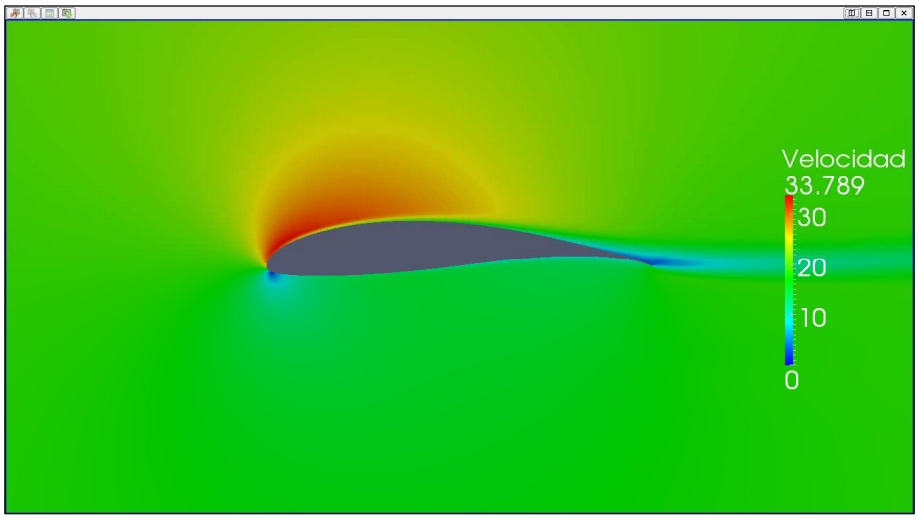
$C_L = 1{,}6159$

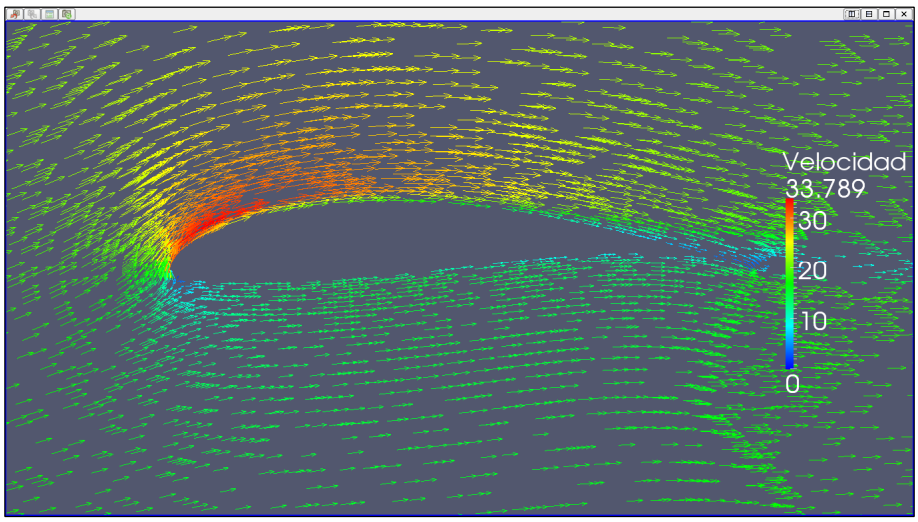
$C_D = 0{,}0335$






p
113.476
0
-200
-400
-523.776


Velocidad
33.789
30
20
10
0


Velocidad
33.789
30
20
10
0

II.3.10.3. Caso 3 (Ángulo de ataque 8º)

Time: 30000

Solving U_X: Initial residual $= 4{,}599 \cdot 10^{-07}$, Final residual $= 3{,}380 \cdot 10^{-08}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 4{,}418 \cdot 10^{-07}$, Final residual $= 3{,}692 \cdot 10^{-08}$, No Iterations $= 2$.

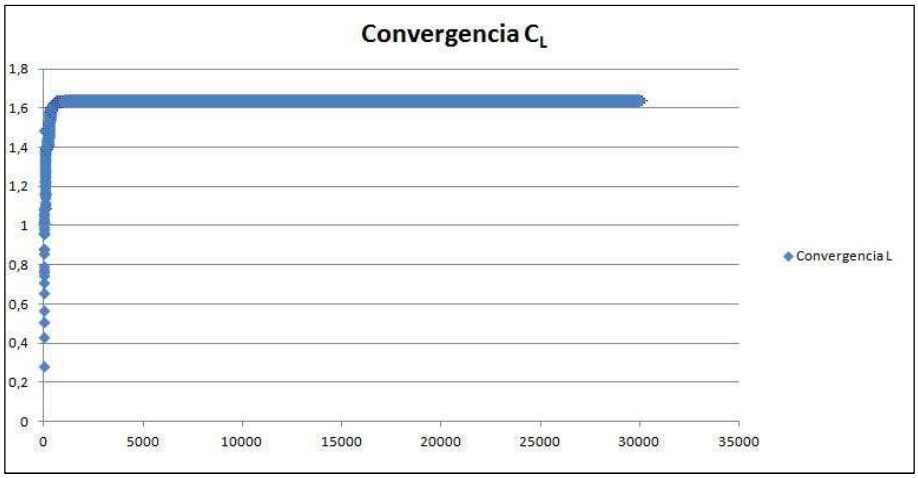
Solving p: Initial residual $= 2{,}668 \cdot 10^{-05}$, Final residual $= 2{,}062 \cdot 10^{-06}$, No Iterations $= 5$.

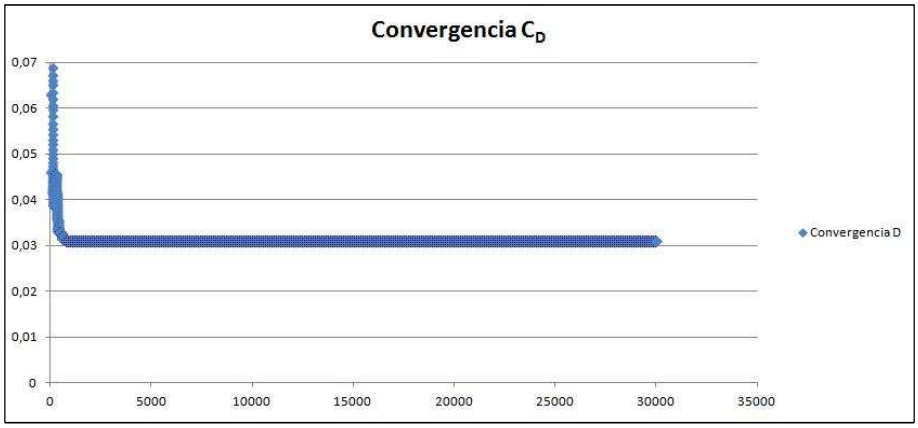
ExecutionTime: 10048,80 s. ClockTime: 10079 s.

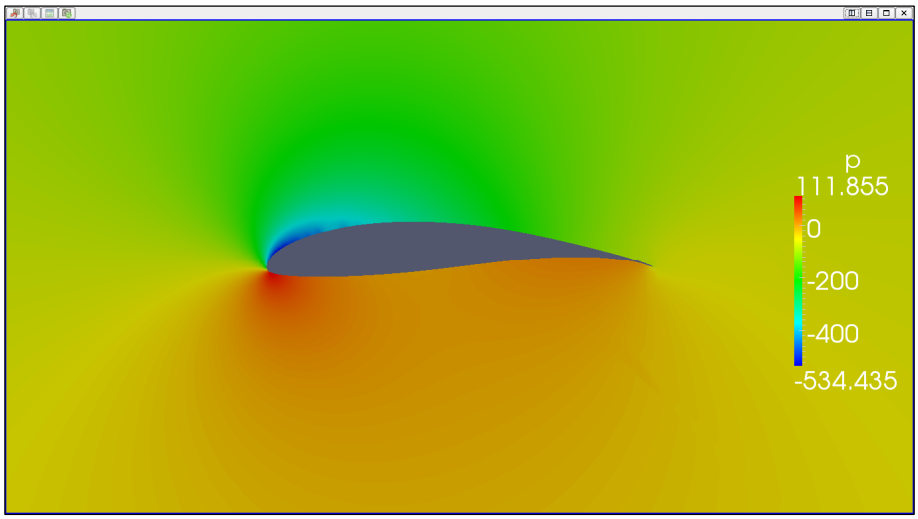
ForceCoeffs output:

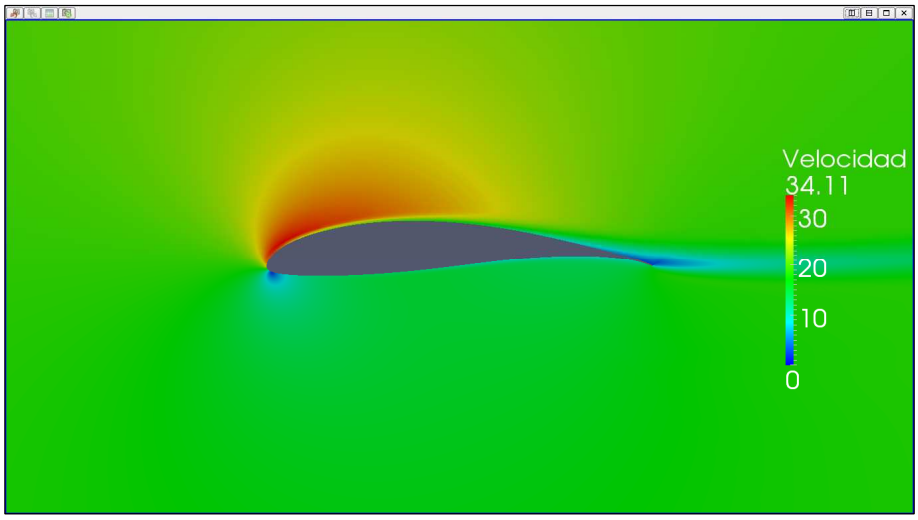
$C_L = 1{,}6378$

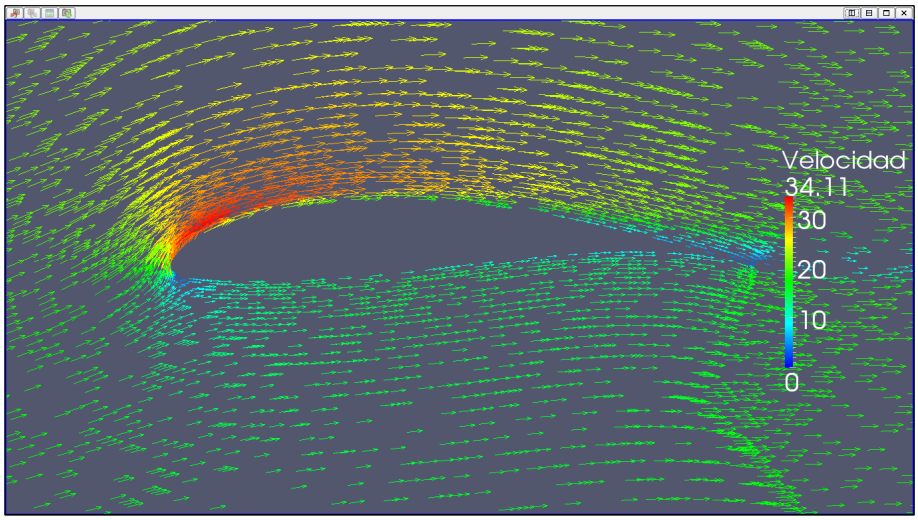
$C_D = 0{,}0308$






p
111.855
0
-200
-400
-534.435


Velocidad
34.11
30
20
10
0


Velocidad
34.11
30
20
10
0

II.3.11. Resultados ángulo de ataque 10º

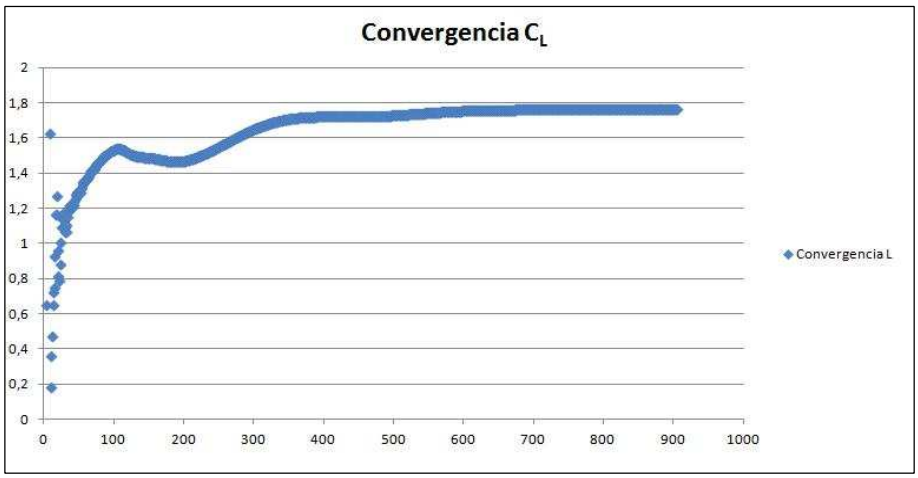
II.3.11.1. Caso 1 (Ángulo de ataque 10º)

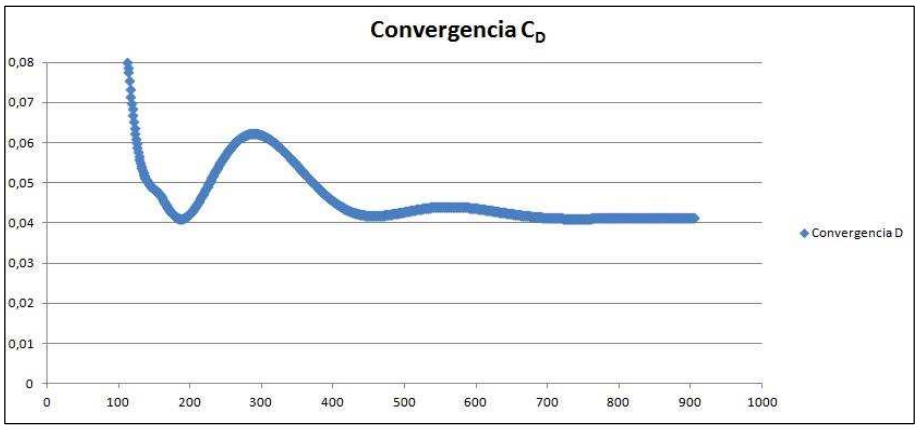
SIMPLE solution converged in 905 iterations.

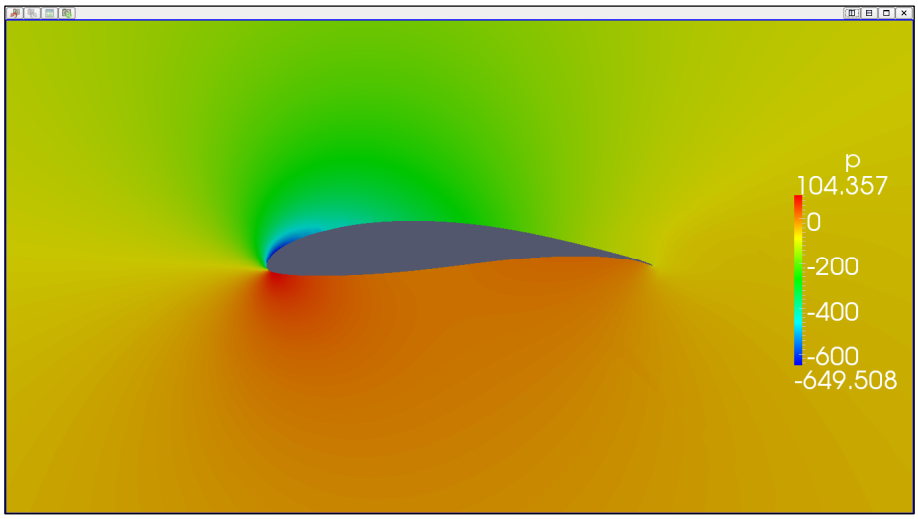
ForceCoeffs output:

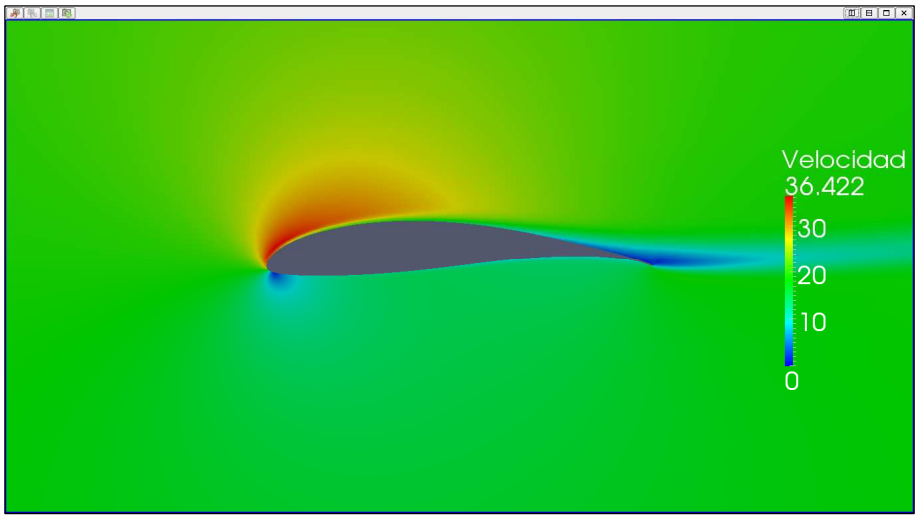
$C_L = 1{,}7658$

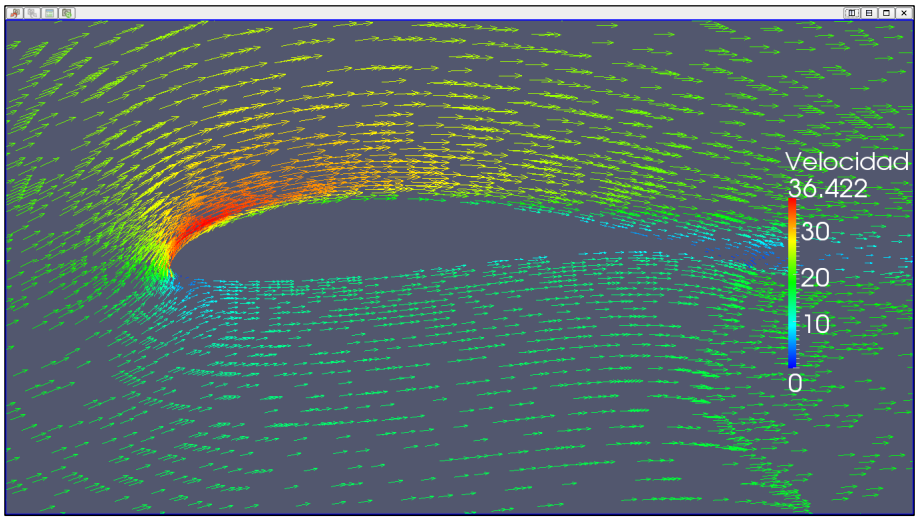
$C_D = 0{,}0412$






p
104.357
0
-200
-400
-600
-649.508


Velocidad
36.422
30
20
10
0


Velocidad
36.422
30
20
10
0

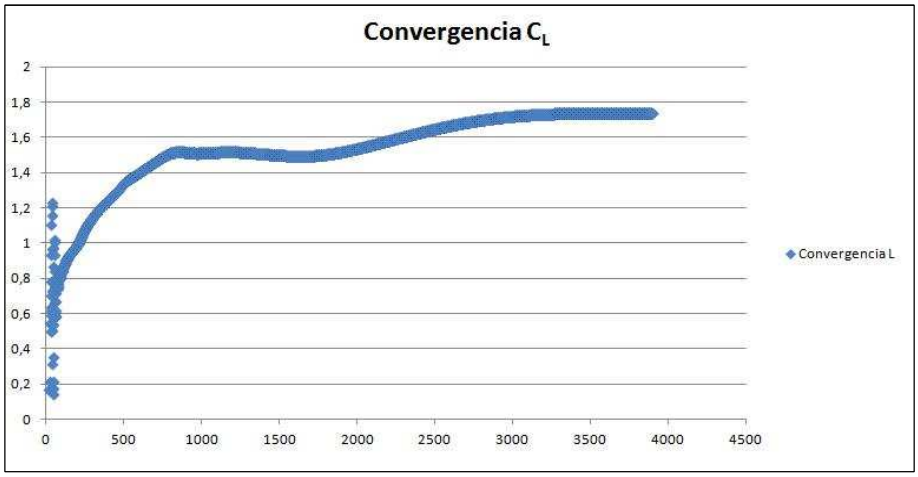
II.3.11.2. Caso 2 (Ángulo de ataque 10º)

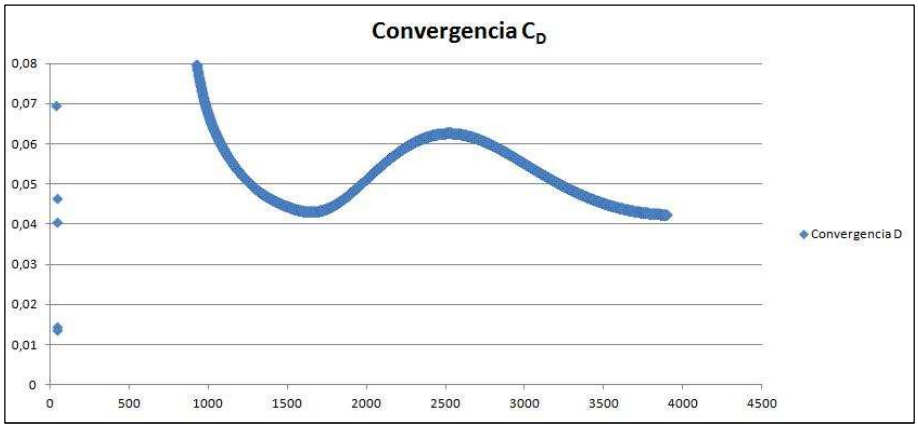
SIMPLE solution converged in 3899 iterations.

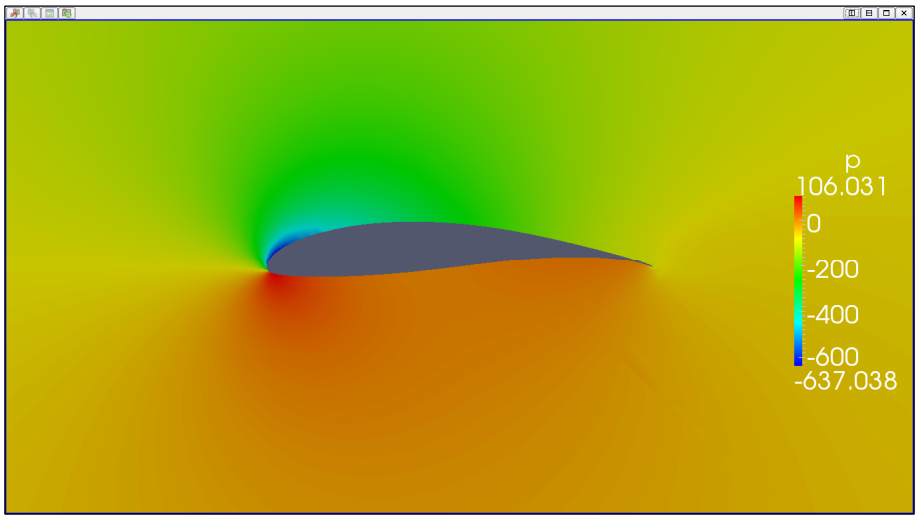
ForceCoeffs output:

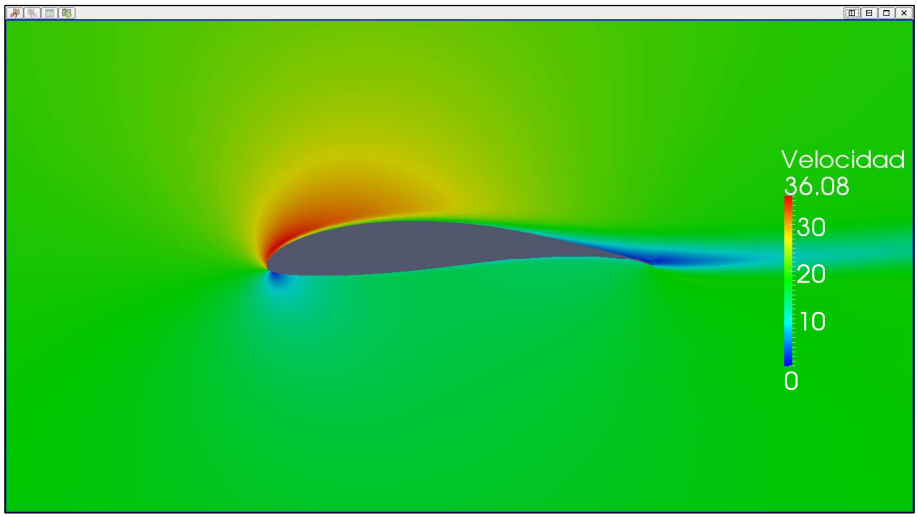
$C_L = 1{,}7390$

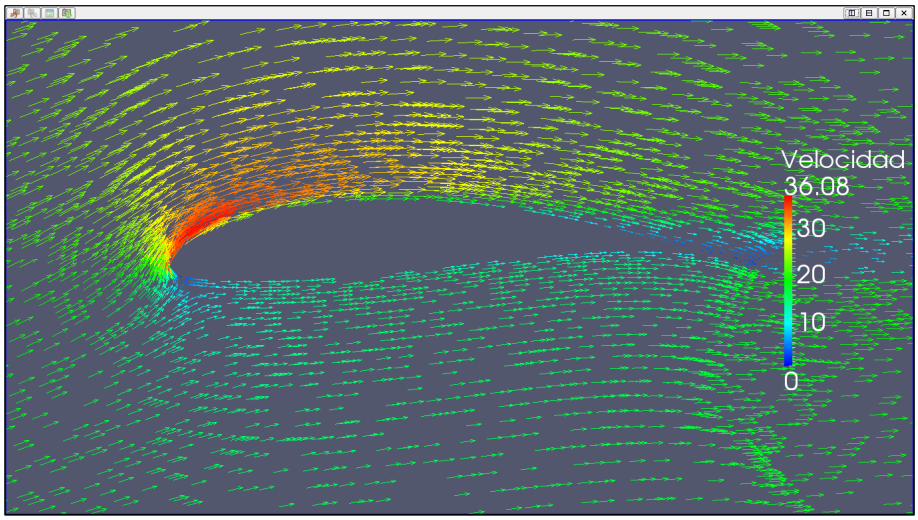
$C_D = 0{,}0424$






p
106.031
0
-200
-400
-600
-637.038


Velocidad
36.08
30
20
10
0


Velocidad
36.08
30
20
10
0

II.3.11.3. Caso 3 (Ángulo de ataque 10º)

Time: 30000

Solving U_X: Initial residual $= 3{,}657 \cdot 10^{-07}$, Final residual $= 2{,}044 \cdot 10^{-08}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 3{,}293 \cdot 10^{-07}$, Final residual $= 1{,}855 \cdot 10^{-08}$, No Iterations $= 2$.

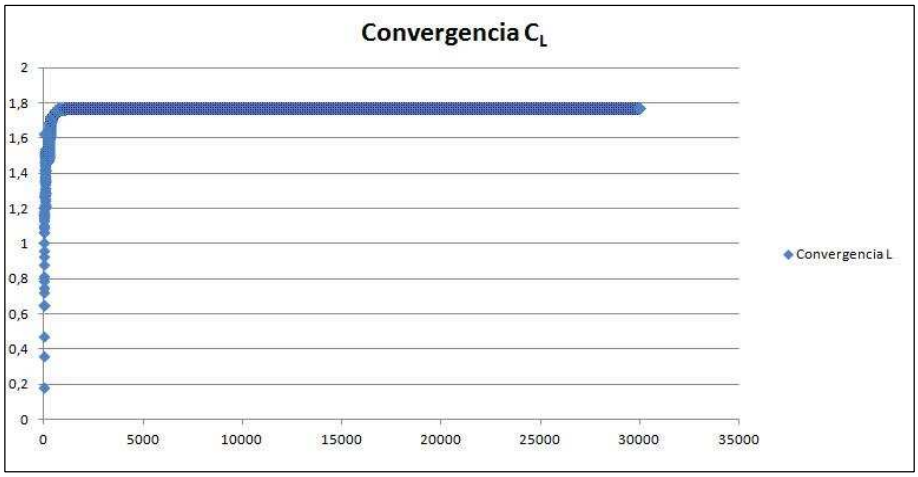
Solving p: Initial residual $= 2{,}629 \cdot 10^{-05}$, Final residual $= 2{,}439 \cdot 10^{-06}$, No Iterations $= 4$.

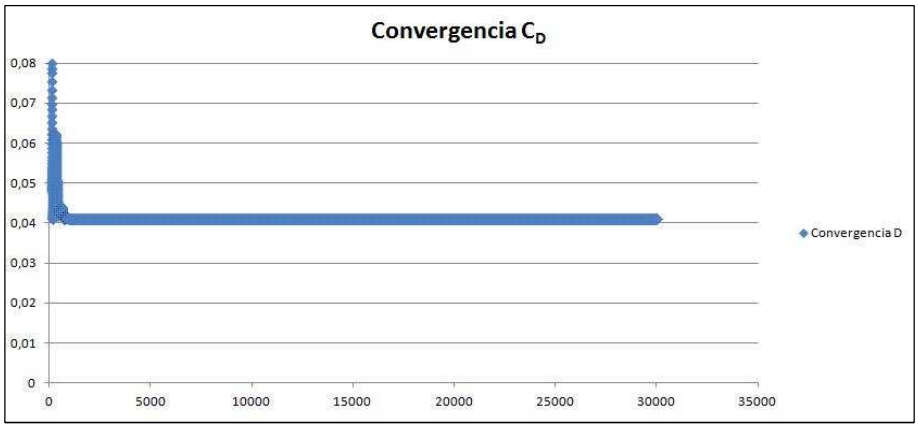
ExecutionTime: 9990,24 s. ClockTime: 10020 s.

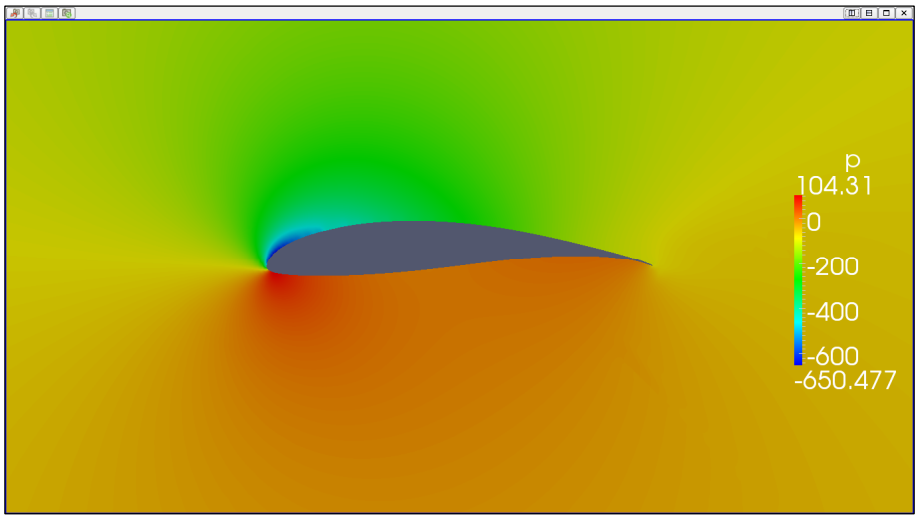
ForceCoeffs output:

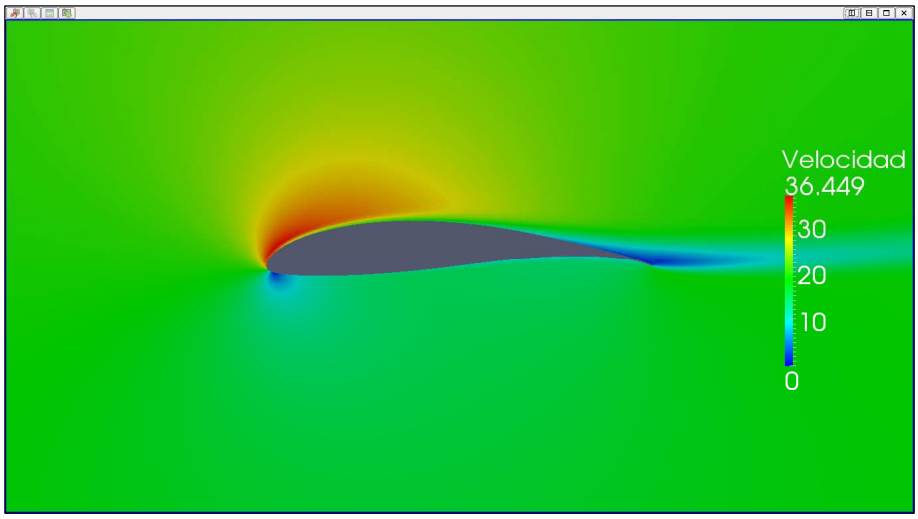
$C_L = 1{,}7673$

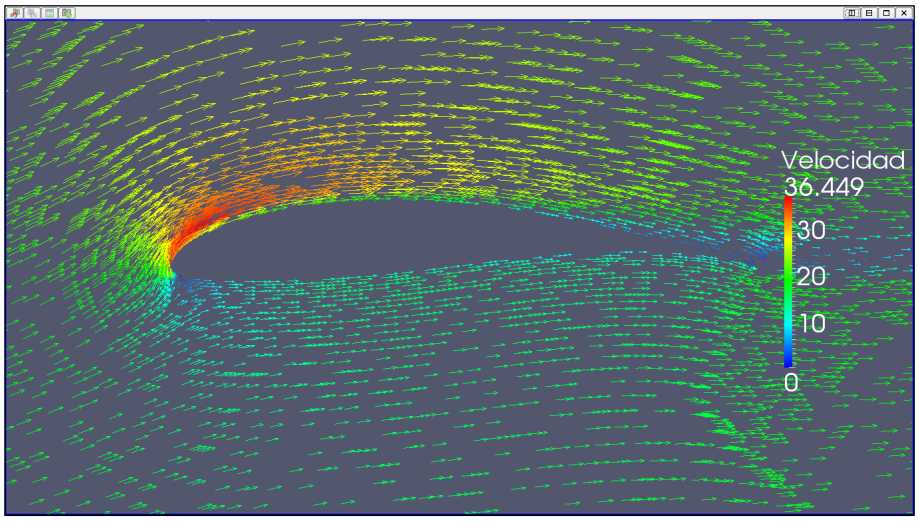
$C_D = 0{,}0409$






p
104.31
0
-200
-400
-600
-650.477


Velocidad
36.449
30
20
10
0


Velocidad
36.449
30
20
10
0

II.3.12. Resultados ángulo de ataque 12º

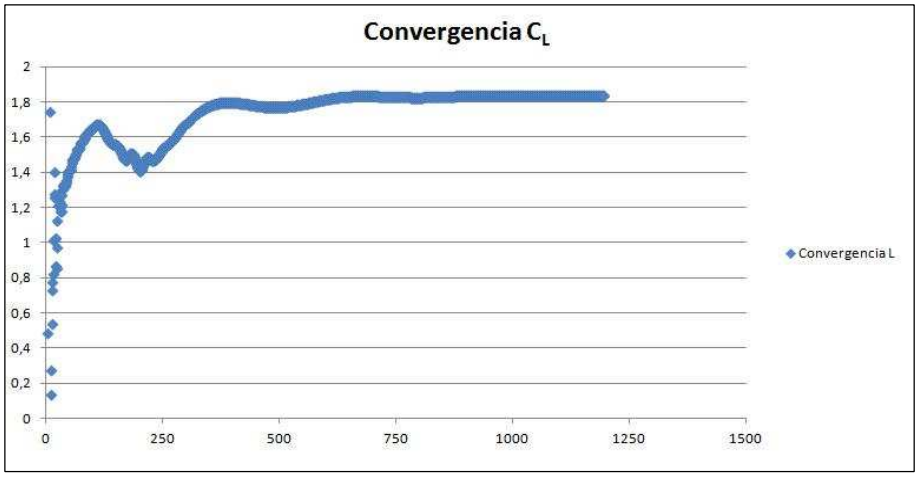
II.3.12.1. Caso 1 (Ángulo de ataque 12º)

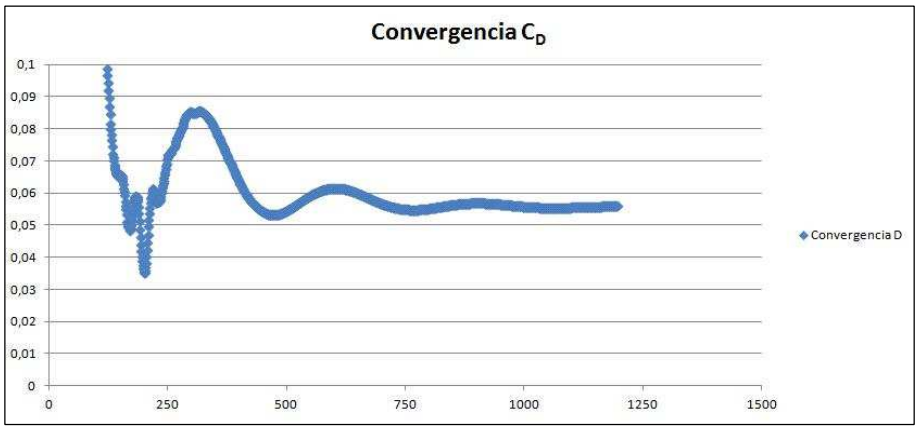
SIMPLE solution converged in 1196 iterations.

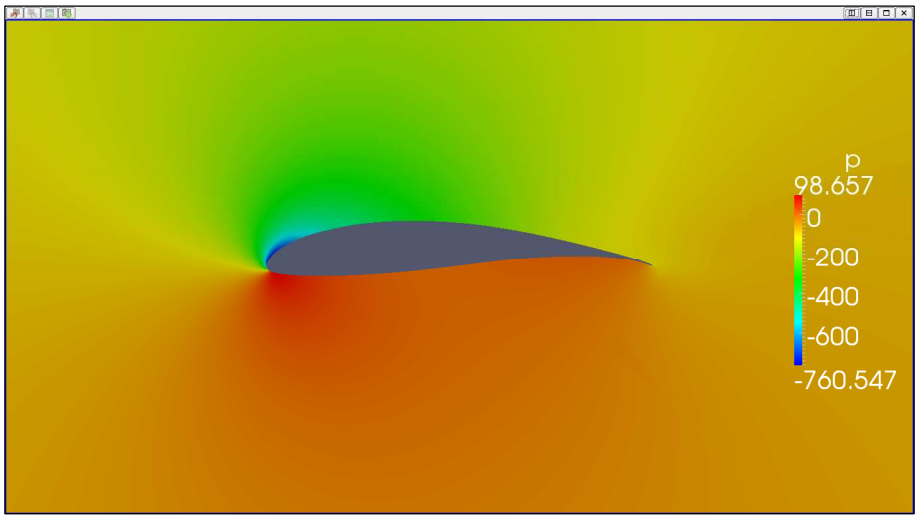
ForceCoeffs output:

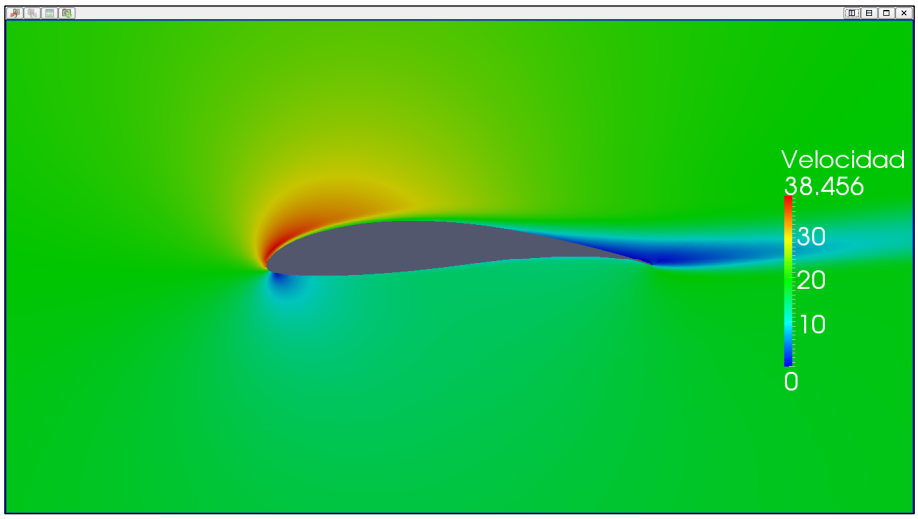
$C_L = 1{,}8384$

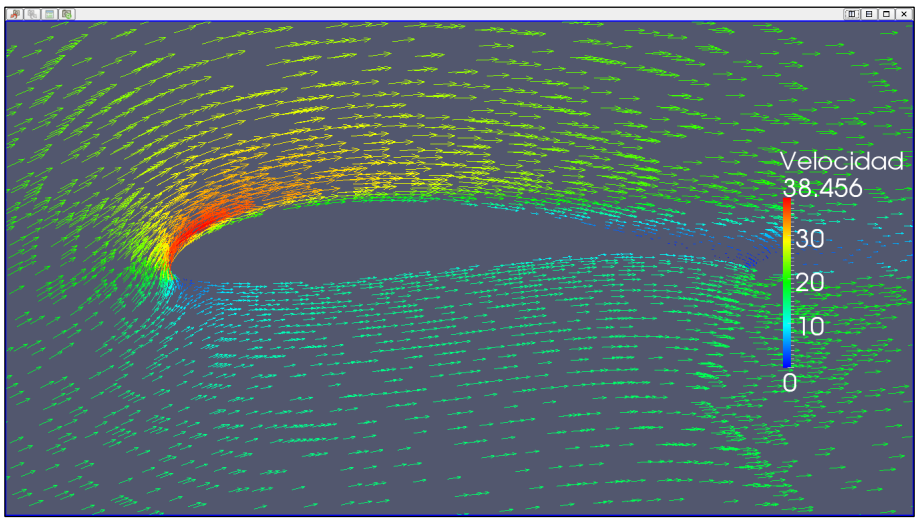
$C_D = 0{,}0559$






p
98.657
0
-200
-400
-600
-760.547


Velocidad
38.456
30
20
10
0


Velocidad
38.456
30
20
10
0

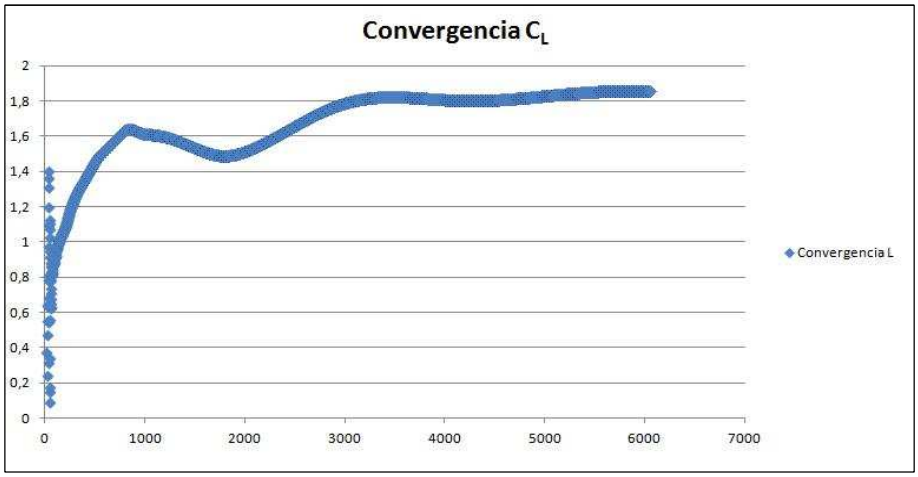
II.3.12.2. Caso 2 (Ángulo de ataque 12º)

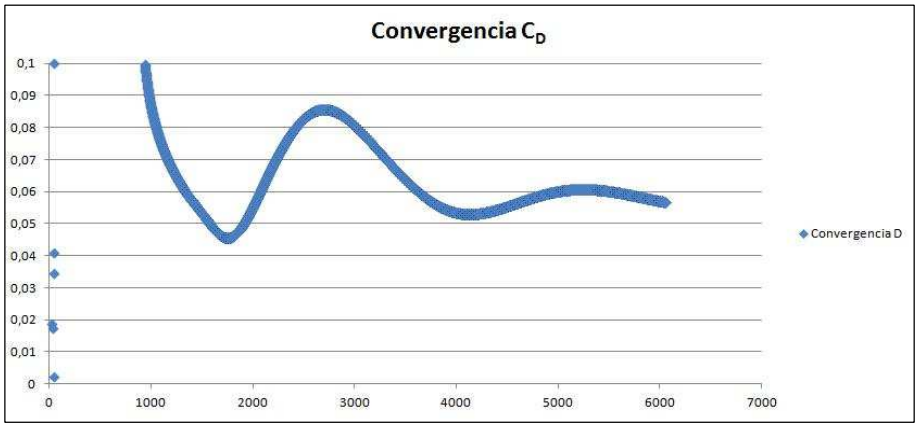
SIMPLE solution converged in 6059 iterations.

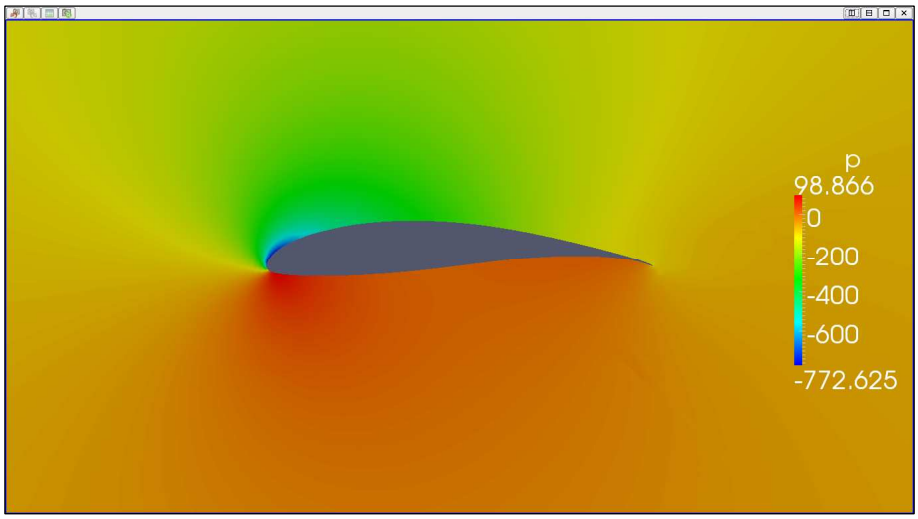
ForceCoeffs output:

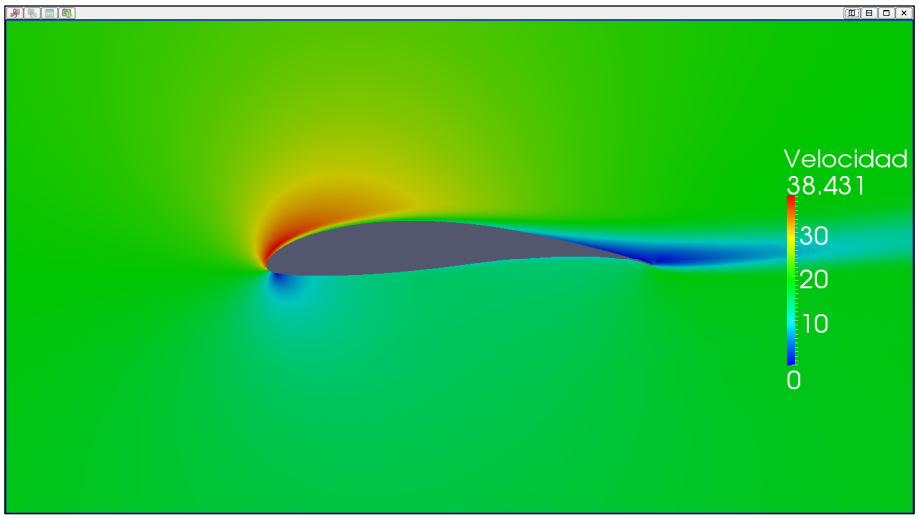
$C_L = 1{,}8587$

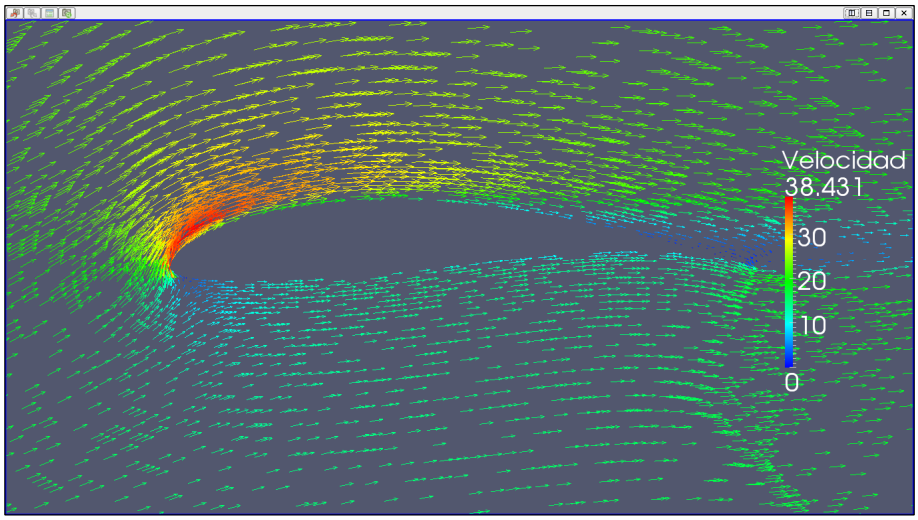
$C_D = 0{,}0566$






p
98.866
0
-200
-400
-600
-772.625


Velocidad
38.431
30
20
10
0


Velocidad
38.431
30
20
10
0

II.3.12.3. Caso 3 (Ángulo de ataque 12º)

Time: 30000

Solving U_X: Initial residual $= 7{,}934 \cdot 10^{-08}$, Final residual $= 2{,}404 \cdot 10^{-09}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 7{,}318 \cdot 10^{-08}$, Final residual $= 2{,}026 \cdot 10^{-09}$, No Iterations $= 2$.

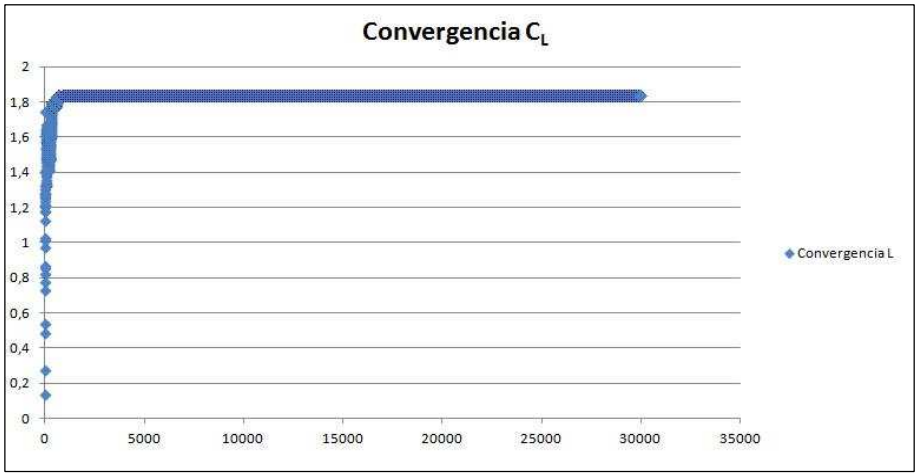
Solving p: Initial residual $= 6{,}889 \cdot 10^{-06}$, Final residual $= 7{,}816 \cdot 10^{-07}$, No Iterations $= 4$.

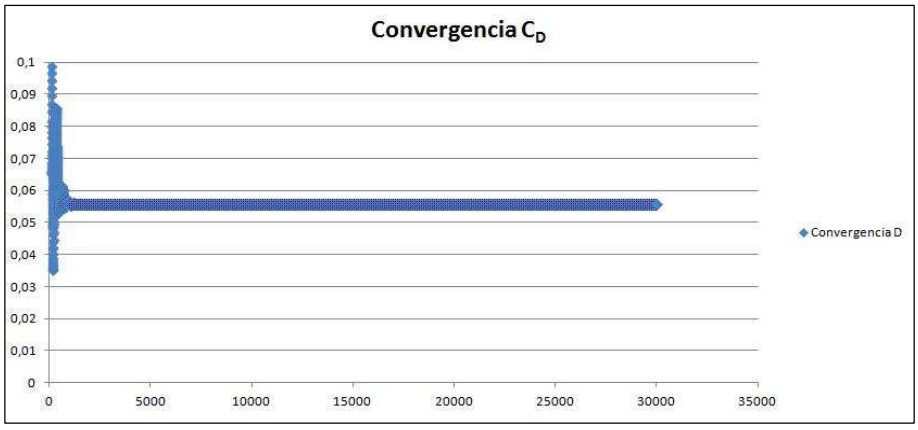
ExecutionTime: 9938,28 s. ClockTime: 9966 s.

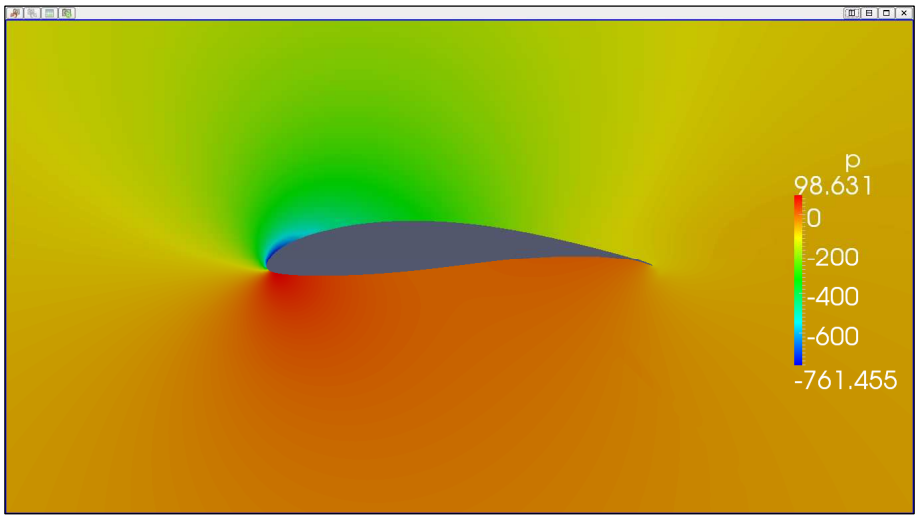
ForceCoeffs output:

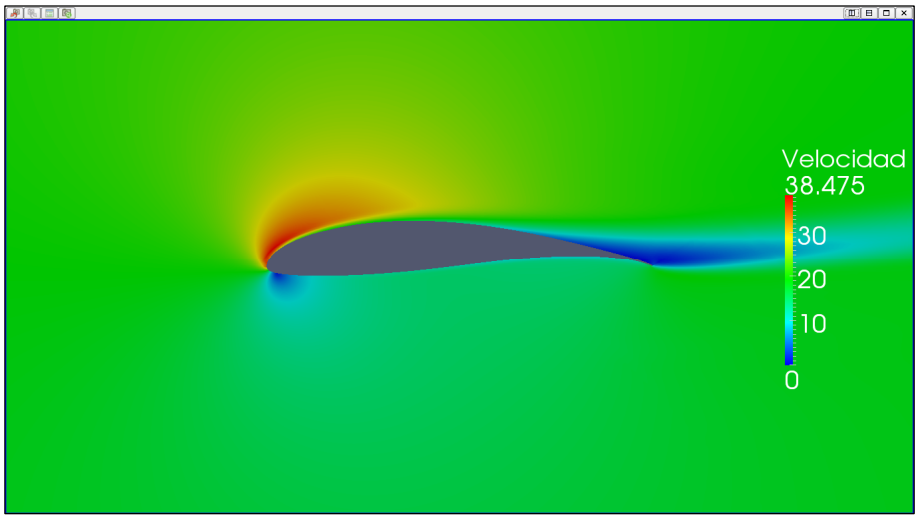
$C_L = 1{,}8391$

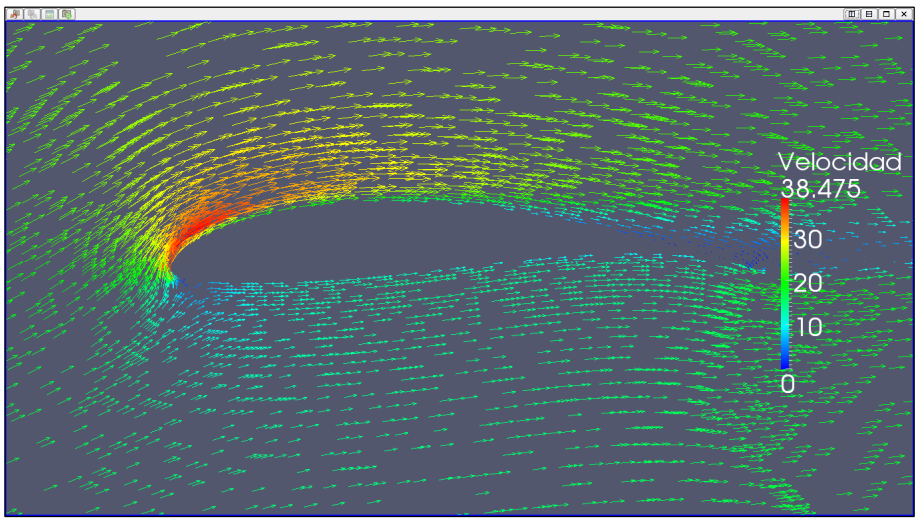
$C_D = 0{,}0556$






p
98.631
0
-200
-400
-600
-761.455


Velocidad
38.475
30
20
10
0


Velocidad
38.475
30
20
10
0

II.3.13. Resultados ángulo de ataque 14º

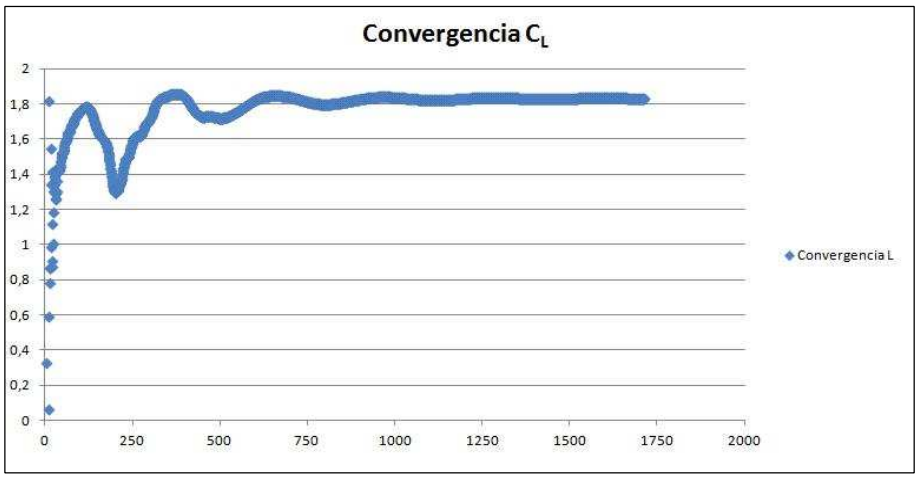
II.3.13.1. Caso 1 (Ángulo de ataque 14º)

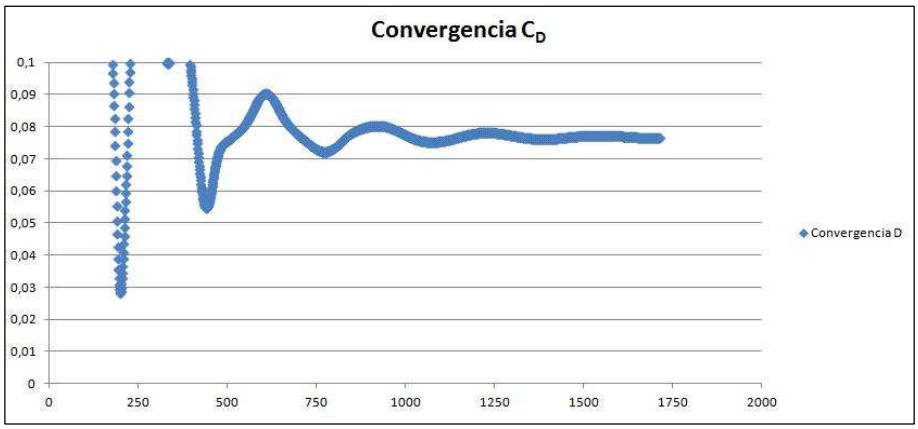
SIMPLE solution converged in 1714 iterations.

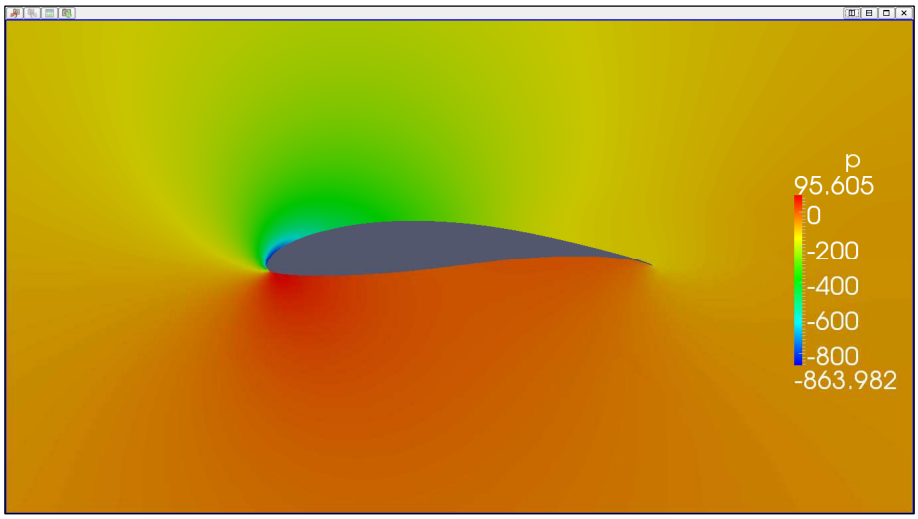
ForceCoeffs output:

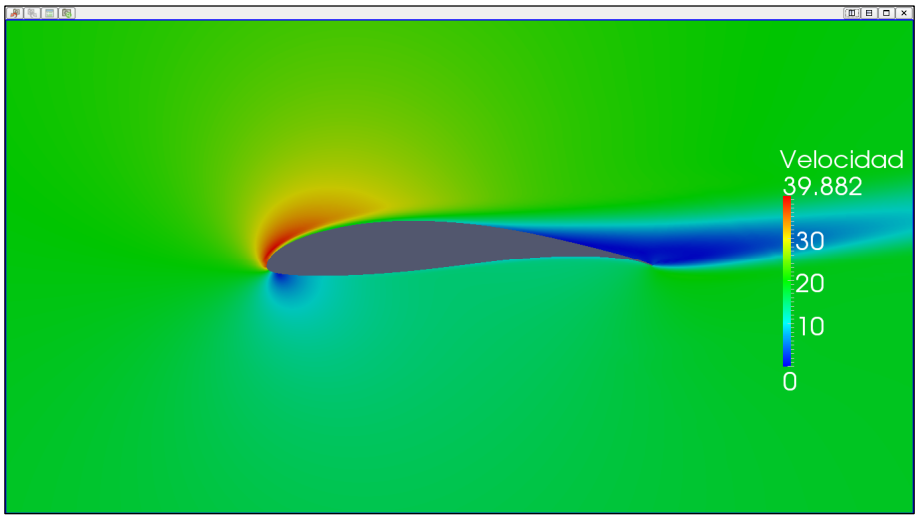
$C_L = 1{,}8314$

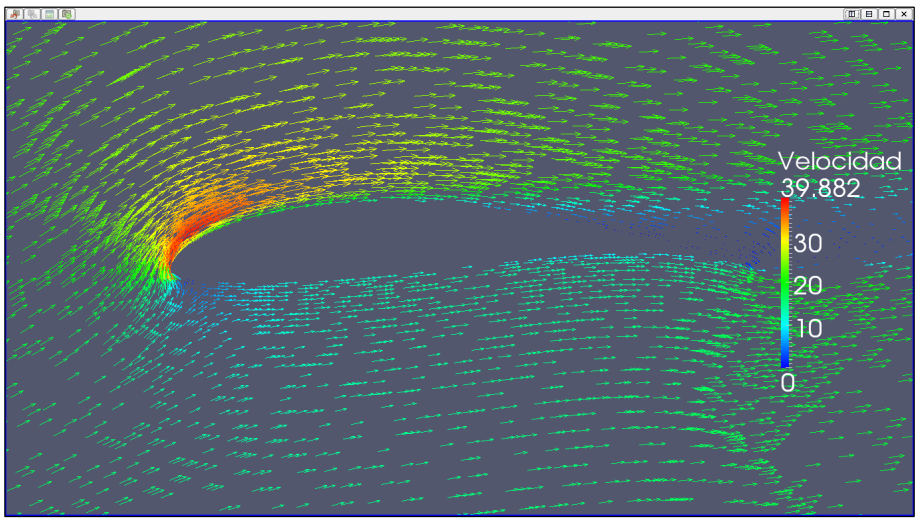
$C_D = 0{,}0766$






p
95.605
0
-200
-400
-600
-800
-863.982


Velocidad
39.882
30
20
10
0


Velocidad
39.882
30
20
10
0

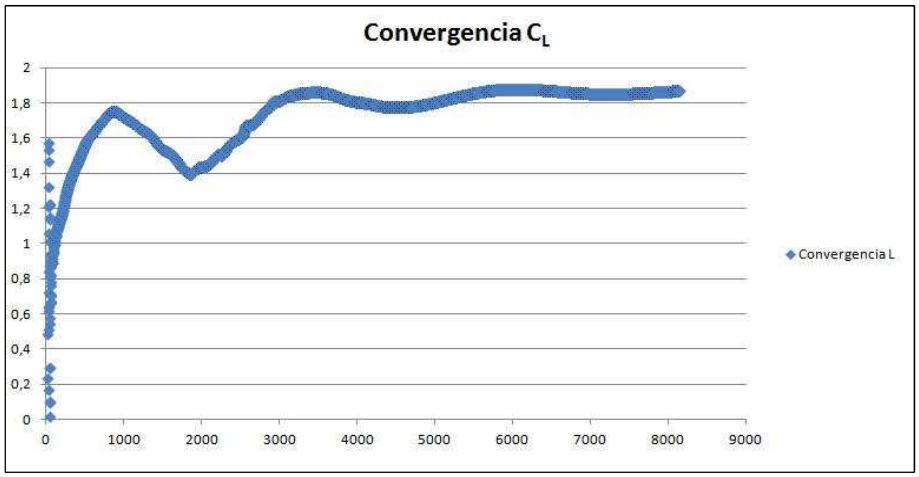
II.3.13.2. Caso 2 (Ángulo de ataque 14º)

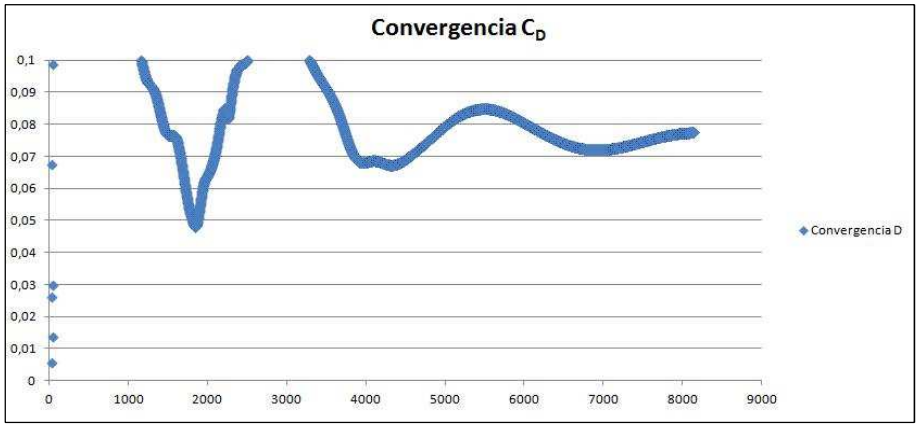
SIMPLE solution converged in 8144 iterations.

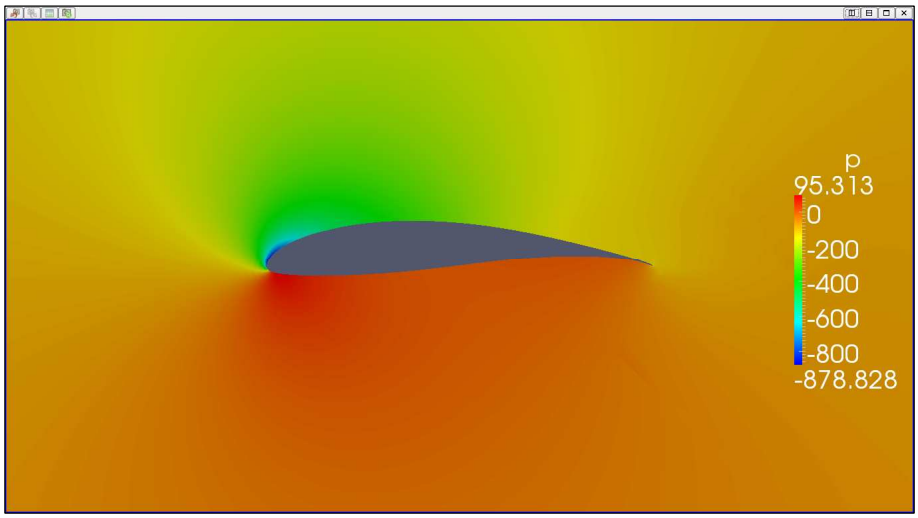
ForceCoeffs output:

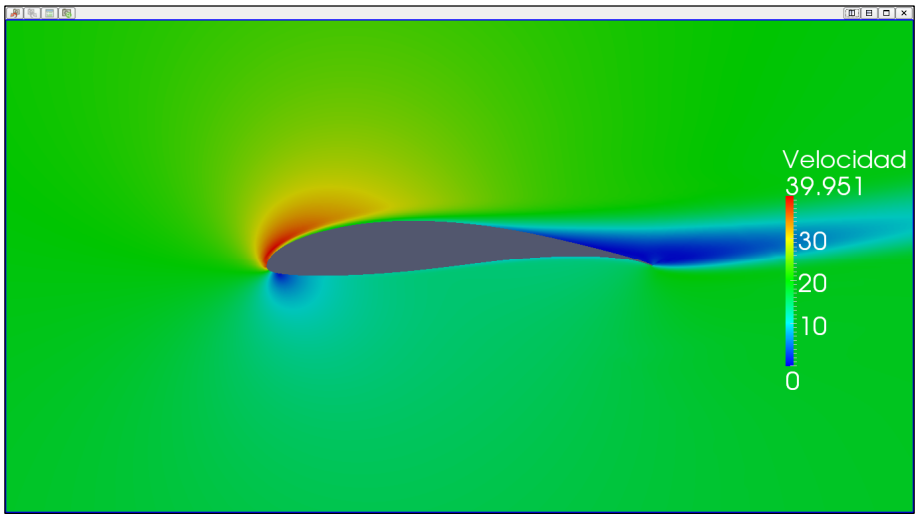
$C_L = 1{,}8680$

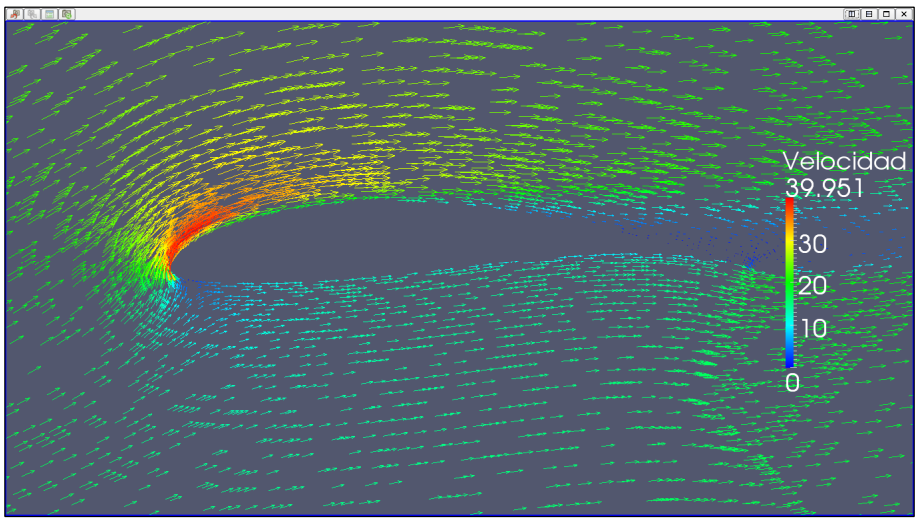
$C_D = 0{,}0774$






p
95.313
0
-200
-400
-600
-800
-878.828


Velocidad
39.951
30
20
10
0


Velocidad
39.951
30
20
10
0

II.3.13.3. Caso 3 (Ángulo de ataque 14º)

Time: 30000

Solving U_X: Initial residual $= 1{,}347 \cdot 10^{-07}$, Final residual $= 5{,}468 \cdot 10^{-09}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 1{,}562 \cdot 10^{-07}$, Final residual $= 6{,}874 \cdot 10^{-09}$, No Iterations $= 2$.

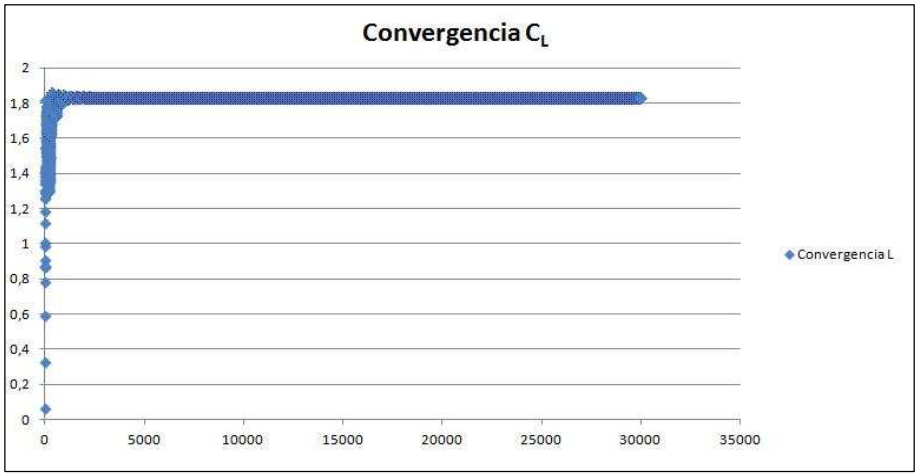
Solving p: Initial residual $= 7{,}533 \cdot 10^{-06}$, Final residual $= 8{,}121 \cdot 10^{-07}$, No Iterations $= 5$.

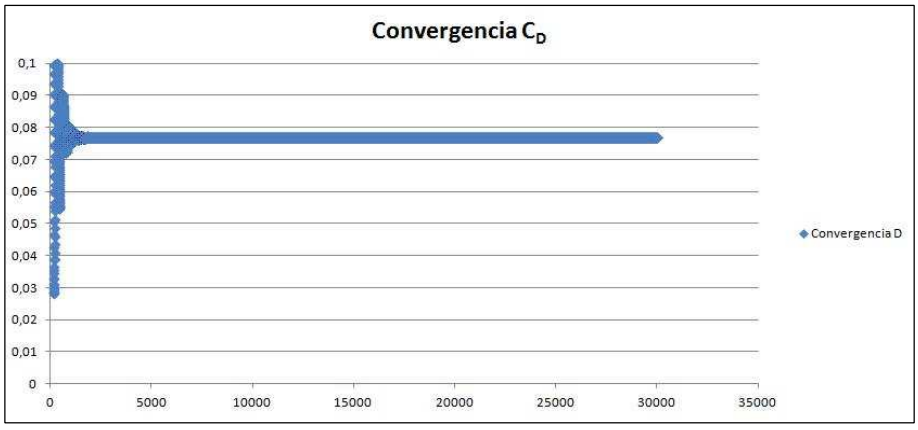
ExecutionTime: 10097,10 s. ClockTime: 6562 s.

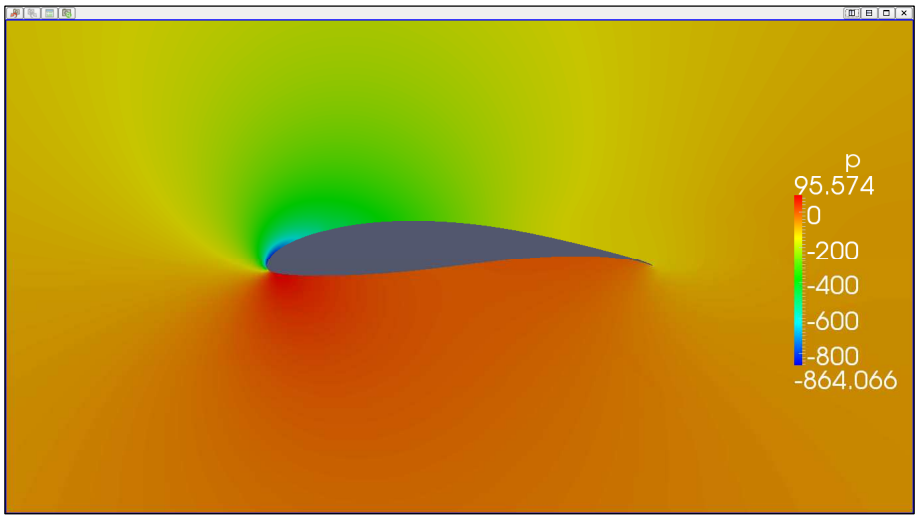
ForceCoeffs output:

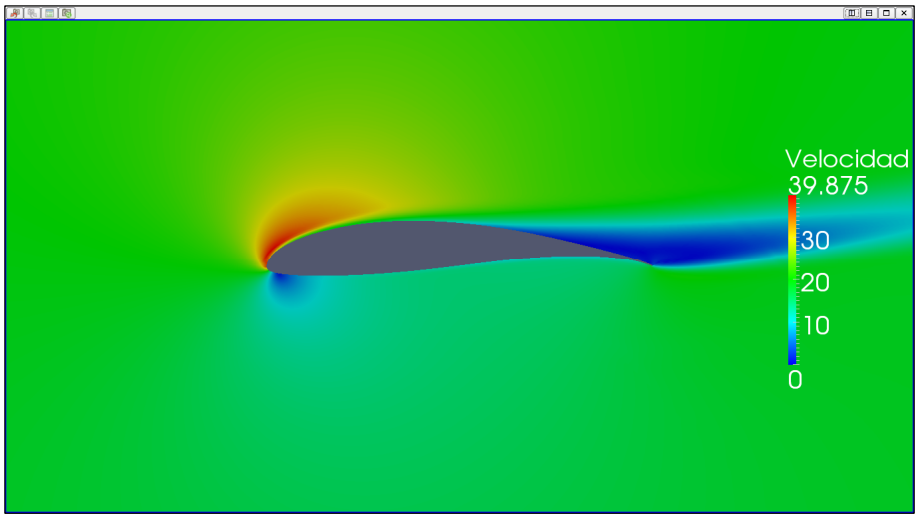
$C_L = 1{,}8324$

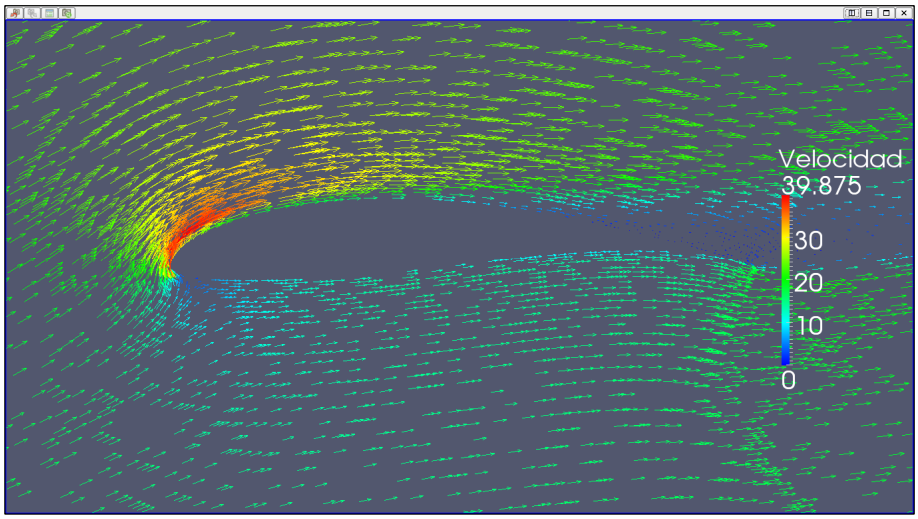
$C_D = 0{,}0768$






p
95.574
0
-200
-400
-600
-800
-864.066


Velocidad
39.875
30
20
10
0


Velocidad
39.875
30
20
10
0

II.3.14. Resultados ángulo de ataque 16º

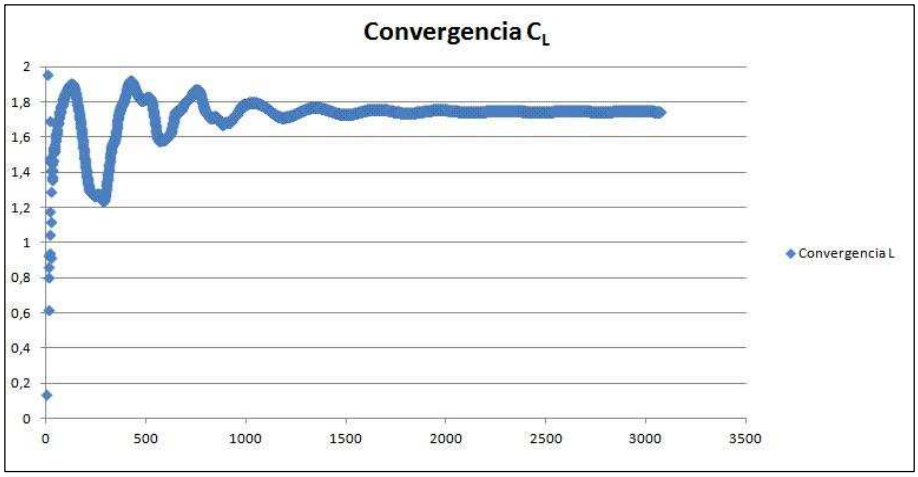
II.3.14.1. Caso 1 (Ángulo de ataque 16º)

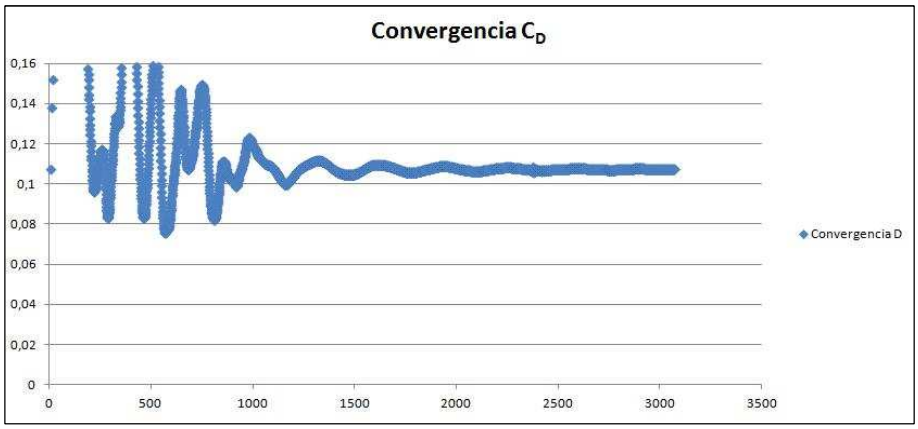
SIMPLE solution converged in 3072 iterations.

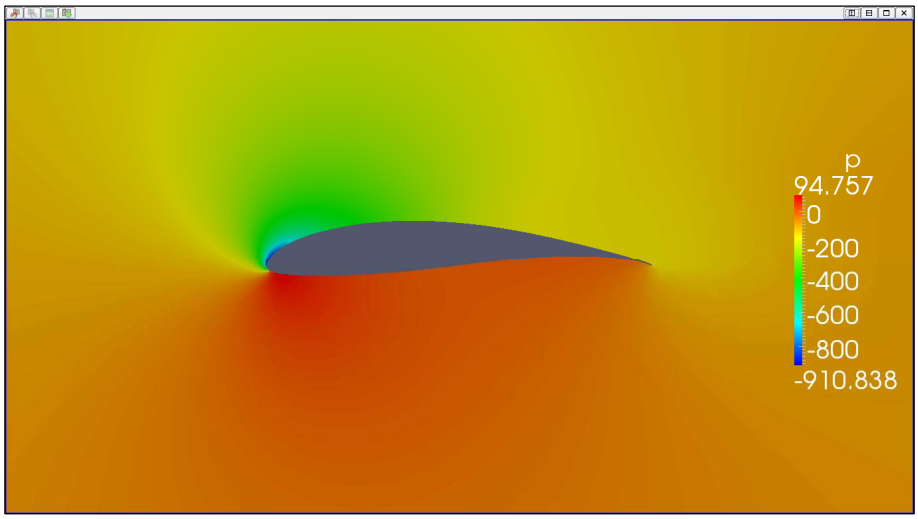
ForceCoeffs output:

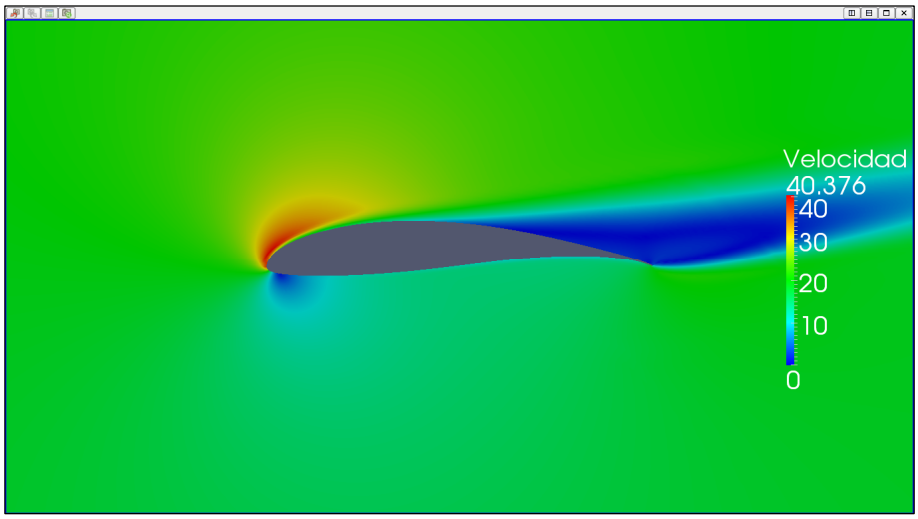
$C_L = 1{,}7465$

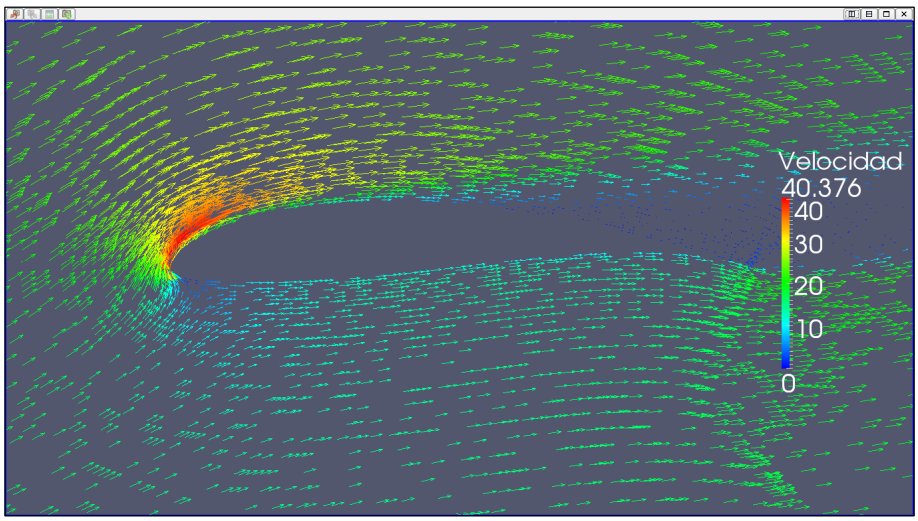
$C_D = 0{,}1072$






p
94.757
0
-200
-400
-600
-800
-910.838


Velocidad
40.376
40
30
20
10
0


Velocidad
40.376
40
30
20
10
0

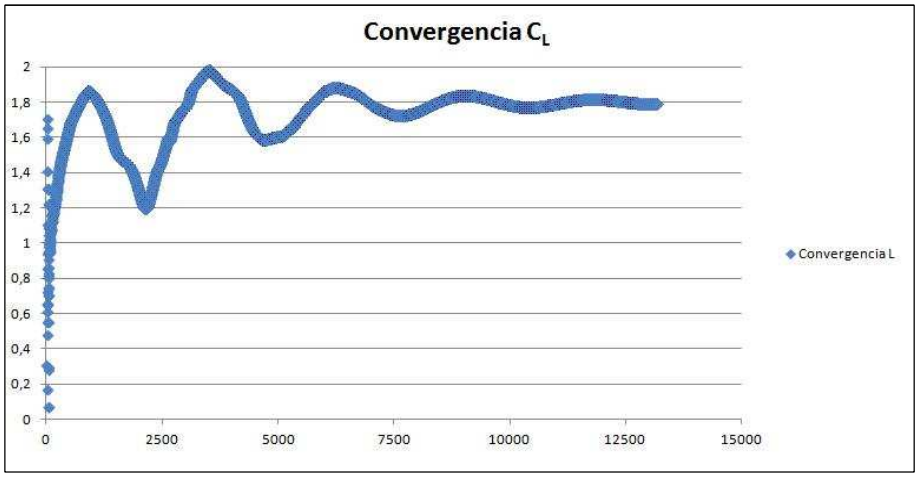
II.3.14.2. Caso 2 (Ángulo de ataque 16º)

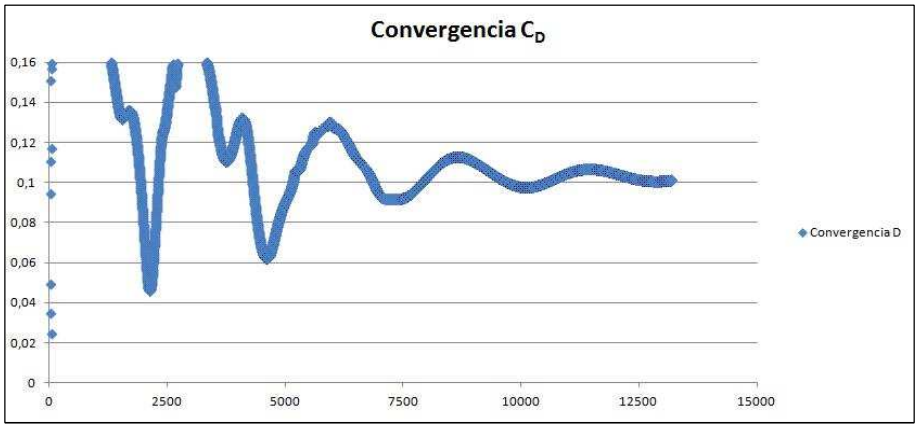
SIMPLE solution converged in 13188 iterations.

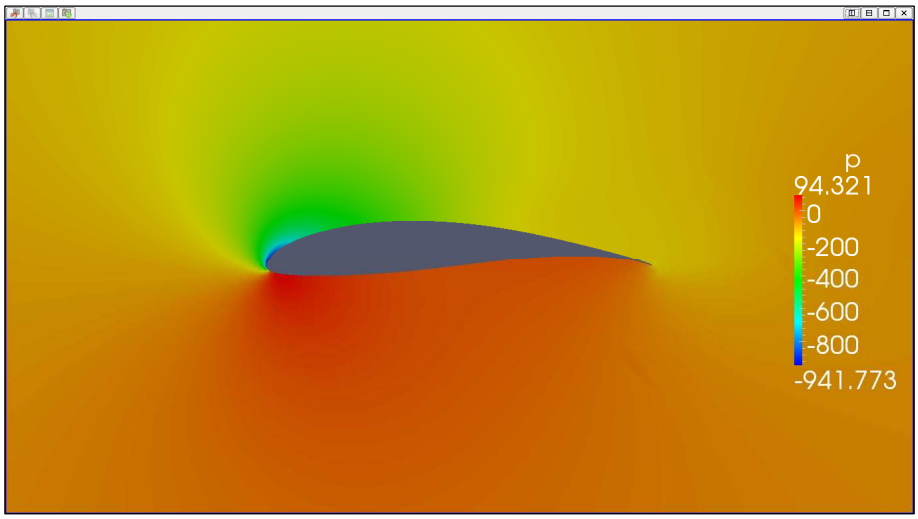
ForceCoeffs output:

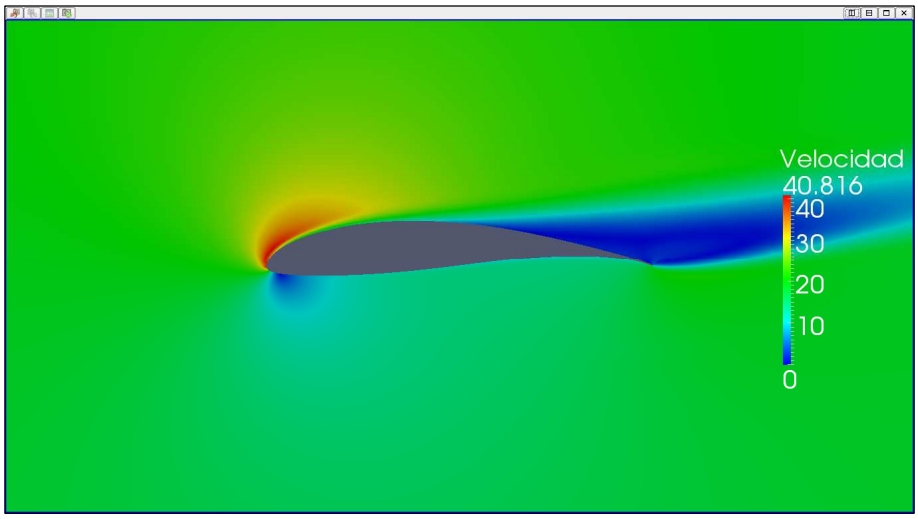
$C_L = 1{,}7873$

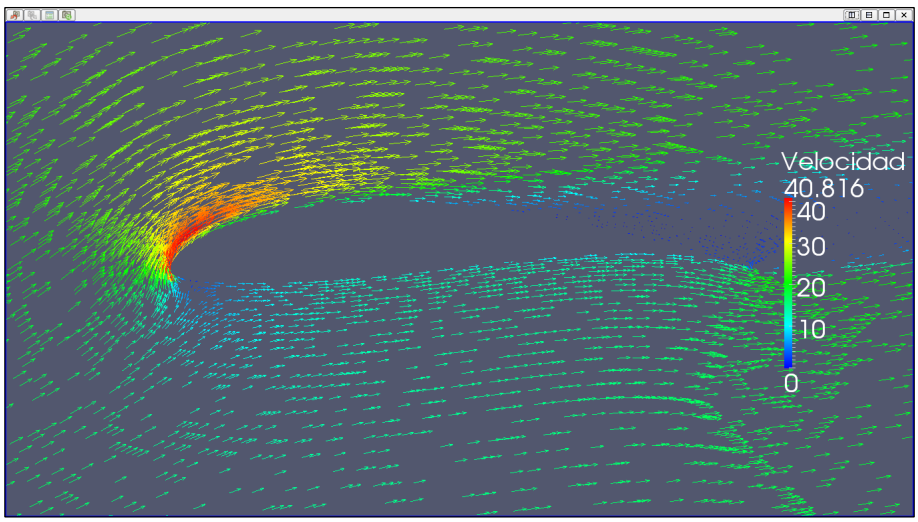
$C_D = 0{,}1012$






p
94.321
0
-200
-400
-600
-800
-941.773


Velocidad
40.816
40
30
20
10
0


Velocidad
40.816
40
30
20
10
0

II.3.14.3. Caso 3 (Ángulo de ataque 16º)

Time: 30000

Solving U_X: Initial residual $= 9{,}467 \cdot 10^{-08}$, Final residual $= 5{,}538 \cdot 10^{-09}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 1{,}175 \cdot 10^{-07}$, Final residual $= 7{,}205 \cdot 10^{-09}$, No Iterations $= 2$.

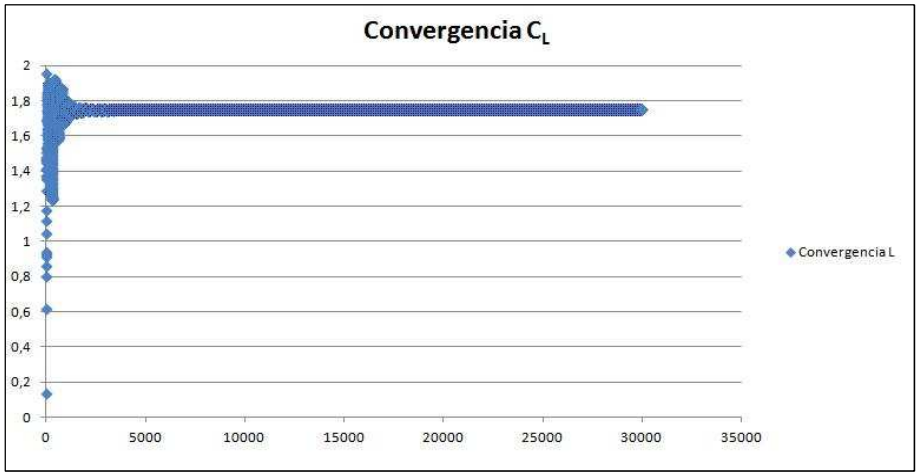
Solving p: Initial residual $= 5{,}633 \cdot 10^{-06}$, Final residual $= 7{,}539 \cdot 10^{-07}$, No Iterations $= 6$.

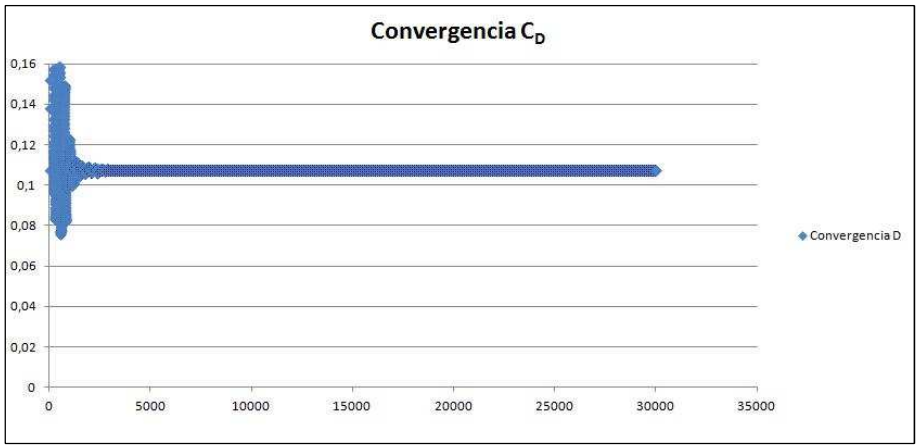
ExecutionTime: 10581,40 s. ClockTime: 10609 s.

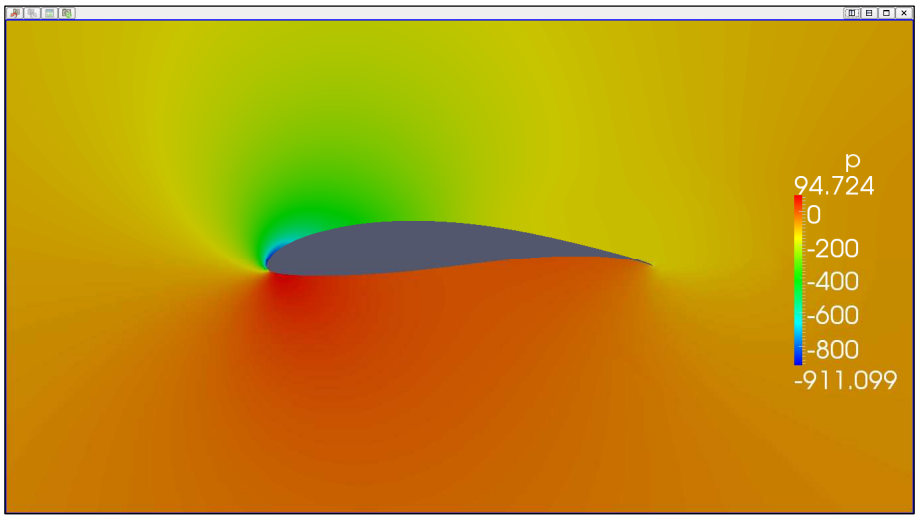
ForceCoeffs output:

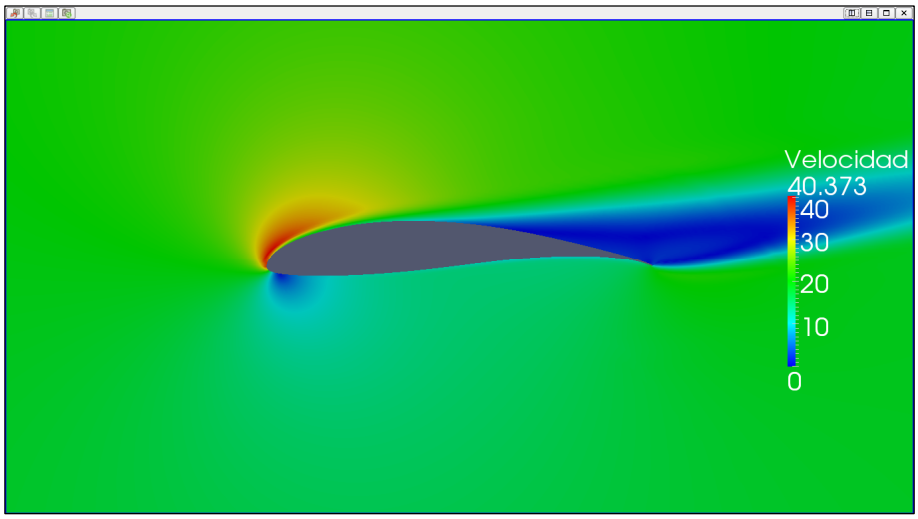
$C_L = 1{,}7476$

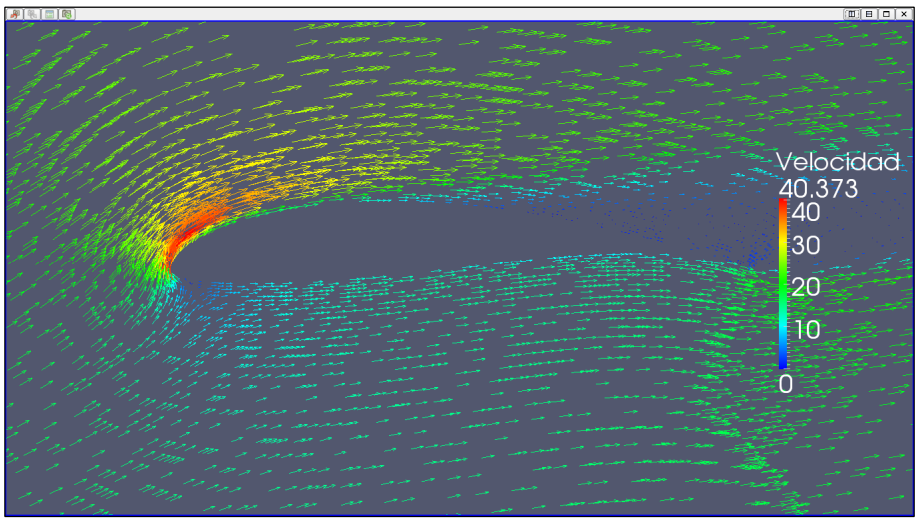
$C_D = 0{,}1074$






p
94.724
0
-200
-400
-600
-800
-911.099


Velocidad
40.373
40
30
20
10
0


Velocidad
40.373
40
30
20
10
0

II.3.15. Resultados ángulo de ataque 18º

II.3.15.1. Caso 1 (Ángulo de ataque 18º)

Time: 20000

Solving U_X: Initial residual $= 7{,}945 \cdot 10^{-03}$, Final residual $= 4{,}129 \cdot 10^{-04}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 1{,}739 \cdot 10^{-02}$, Final residual $= 8{,}361 \cdot 10^{-04}$, No Iterations $= 4$.

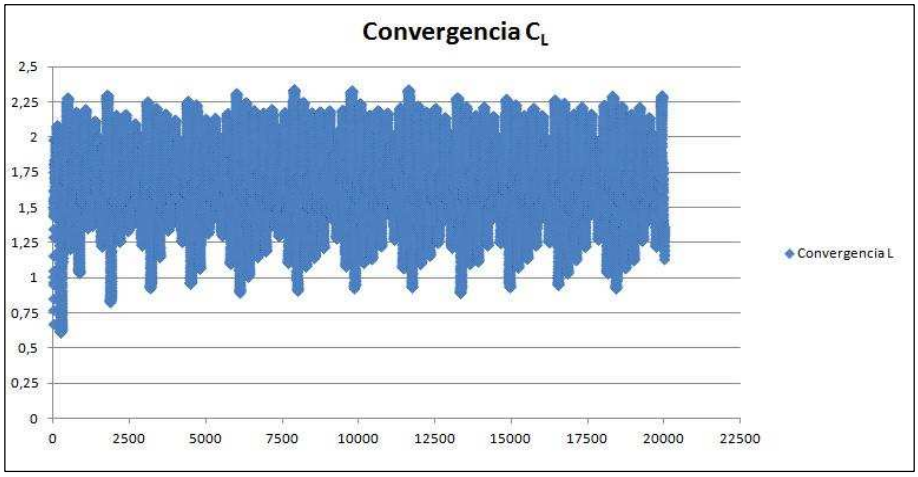
Solving p: Initial residual $= 1{,}264 \cdot 10^{-01}$, Final residual $= 9{,}175 \cdot 10^{-03}$, No Iterations $= 4$.

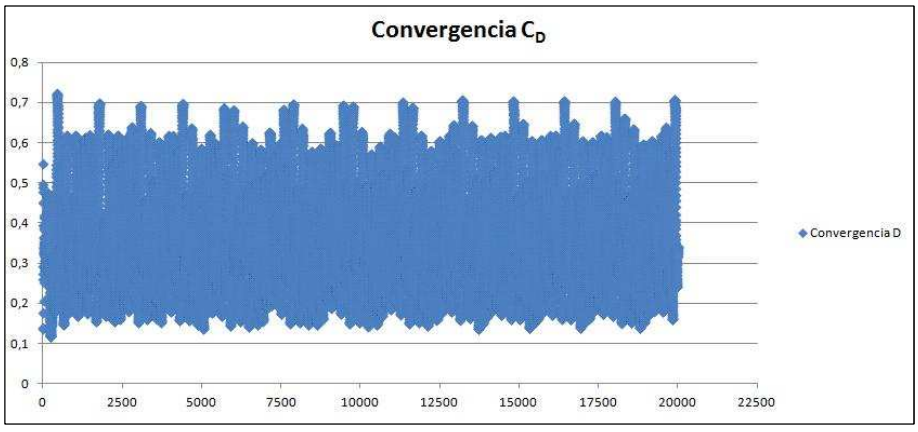
ExecutionTime: 6903,80 s. ClockTime: 6946 s.

ForceCoeffs output:

$C_L = 1{,}1326$

$C_D = 0{,}3160$





II.3.15.2. Caso 2 (Ángulo de ataque 18º)

Time: 20000

Solving U_X: Initial residual $= 9{,}648 \cdot 10^{-04}$, Final residual $= 7{,}517 \cdot 10^{-06}$, No Iterations $= 2$.

Solving U_Y: Initial residual $= 1{,}732 \cdot 10^{-03}$, Final residual $= 1{,}323 \cdot 10^{-05}$, No Iterations $= 2$.

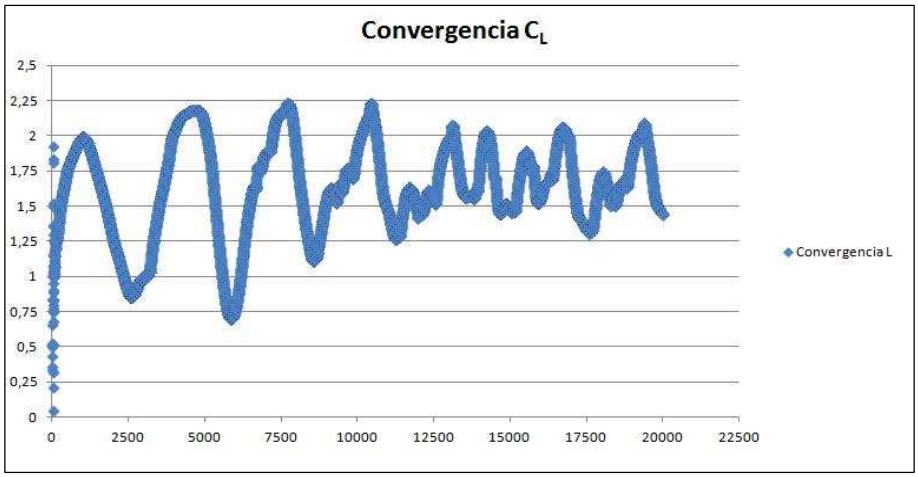
Solving p: Initial residual $= 2{,}424 \cdot 10^{-02}$, Final residual $= 1{,}687 \cdot 10^{-03}$, No Iterations $= 3$.

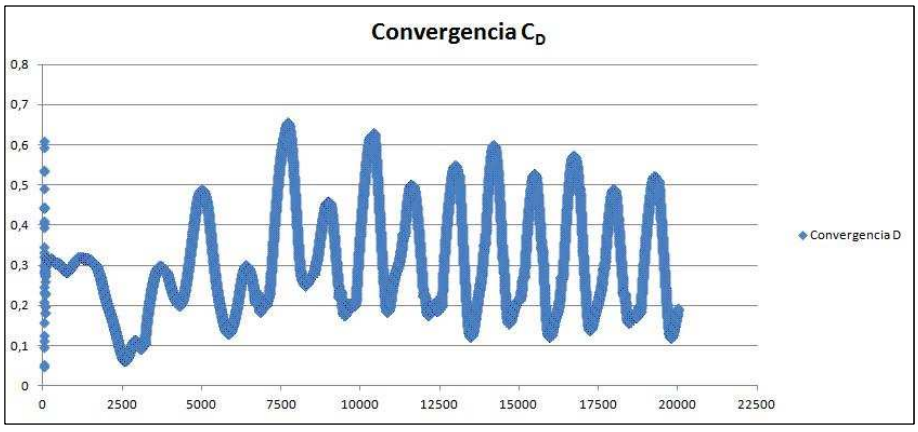
ExecutionTime: 6434,36 s. ClockTime: 6462 s.

ForceCoeffs output:

$C_L = 1{,}4411$

$C_D = 0{,}1924$





II.3.15.3. Caso 3 (Ángulo de ataque 18º)

Time: 30000

Solving U_X: Initial residual $= 6{,}410 \cdot 10^{-03}$, Final residual $= 3{,}701 \cdot 10^{-04}$, No Iterations $= 4$.

Solving U_Y: Initial residual $= 1{,}275 \cdot 10^{-02}$, Final residual $= 6{,}668 \cdot 10^{-04}$, No Iterations $= 4$.

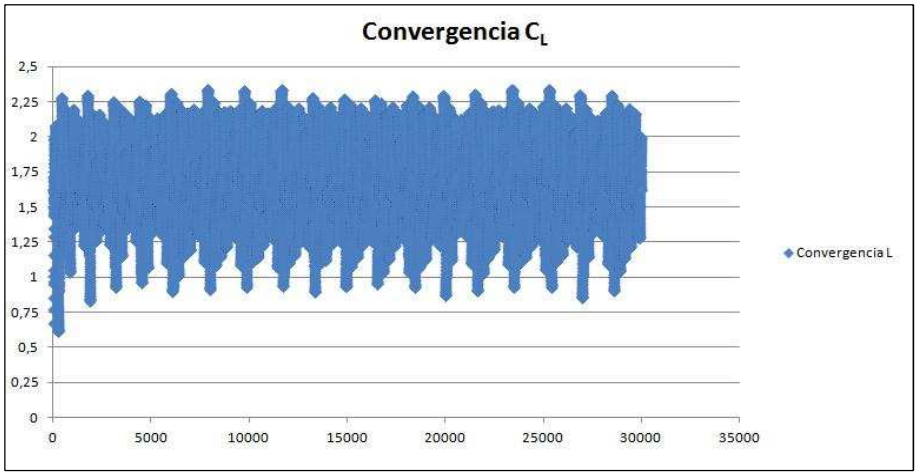
Solving p: Initial residual $= 8{,}350 \cdot 10^{-02}$, Final residual $= 6{,}760 \cdot 10^{-03}$, No Iterations $= 4$.

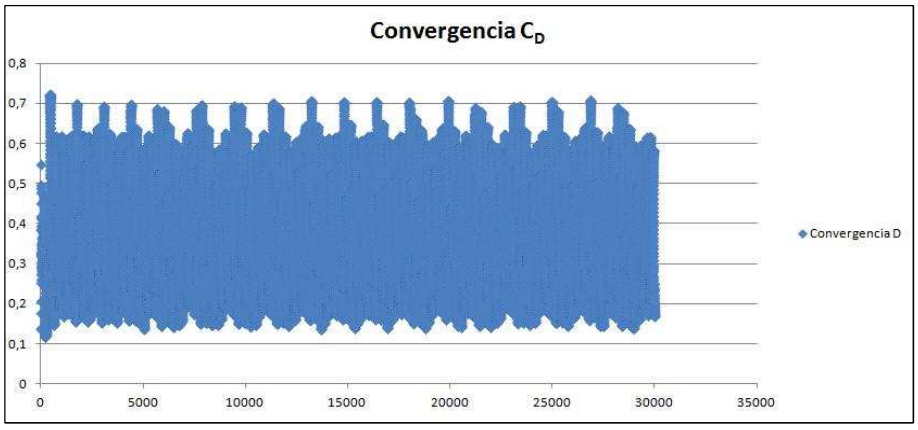
ExecutionTime: 10360,80 s. ClockTime: 10387 s.

ForceCoeffs output:

$C_L = 1{,}6226$

$C_D = 0{,}2006$

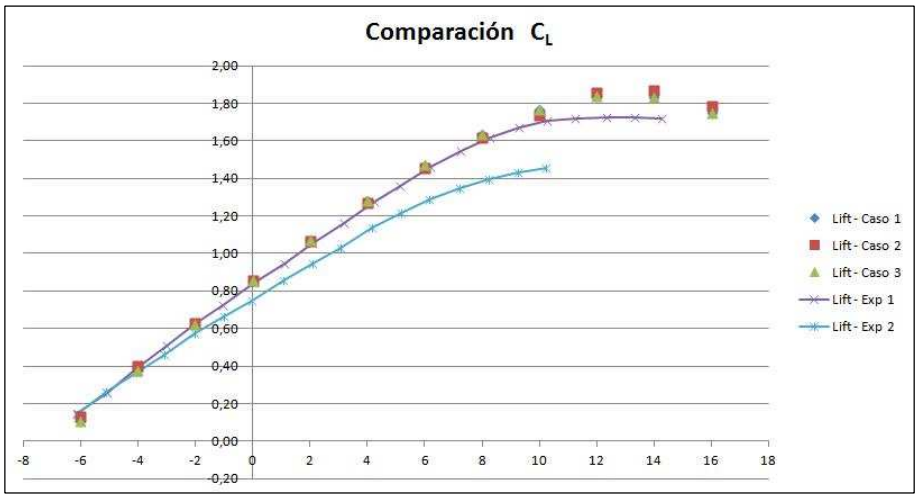


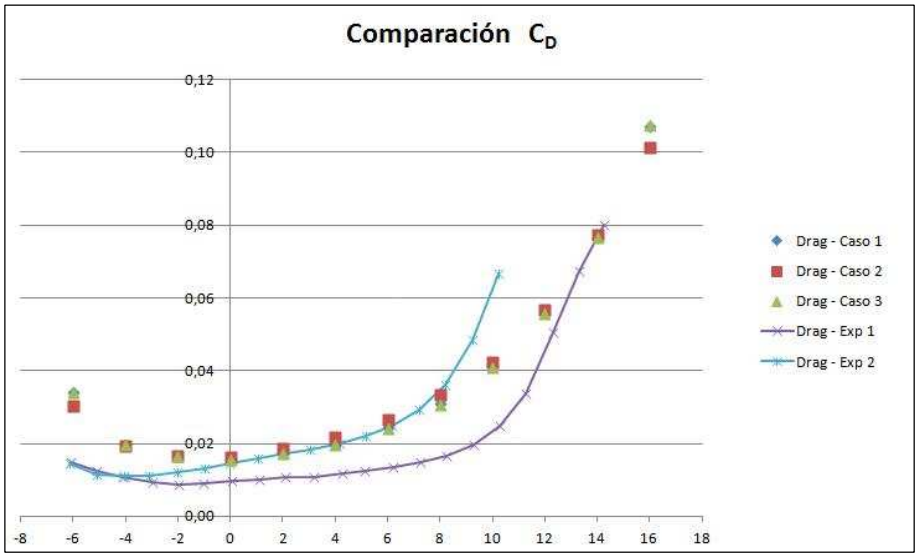


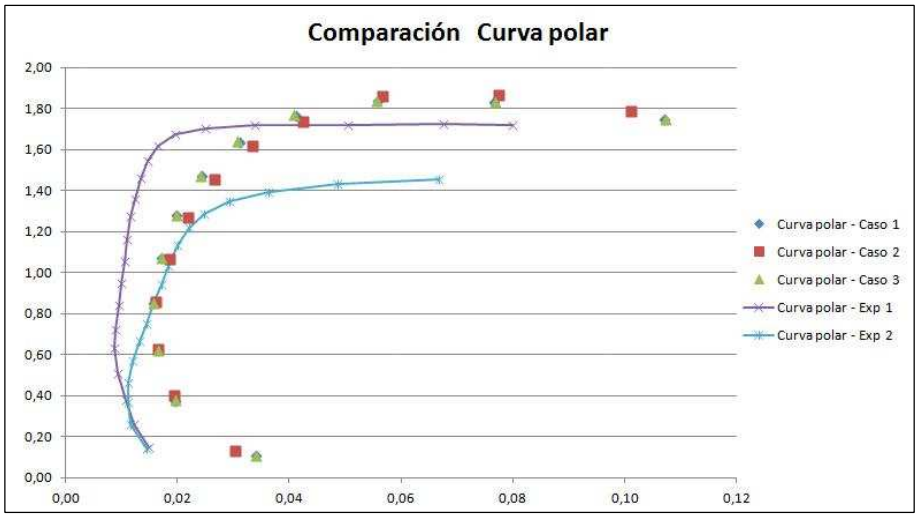
II.2.16. Resumen de resultados y comparación

Ángulo	C_L - Caso1	C_L - Caso2	C_L - Caso3	C_D - Caso1	C_D - Caso2	C_D - Caso3
-6º	0,1068	0,1307	0,1063	0,0340	0,0304	0,0340
-4º	0,3760	0,3994	0,3766	0,0196	0,0194	0,0196
-2º	0,6194	0,6283	0,6200	0,0166	0,0165	0,0166
0º	0,8537	0,8567	0,8537	0,0157	0,0162	0,0157
2º	1,0718	1,0639	1,0718	0,0171	0,0186	0,0171
4º	1,2796	1,2668	1,2797	0,0198	0,0218	0,0198
6º	1,4693	1,4548	1,4713	0,0243	0,0266	0,0241
8º	1,6349	1,6159	1,6378	0,0311	0,0335	0,0308
10º	1,7658	1,7390	1,7673	0,0412	0,0424	0,0409
12º	1,8384	1,8587	1,8391	0,0559	0,0566	0,0556
14º	1,8314	1,8680	1,8324	0,0766	0,0774	0,0768
16º	1,7465	1,7873	1,7476	0,1072	0,1012	0,1074

Ángulo - Exp1	C_L - Exp1	C_D - Exp1	Ángulo - Exp 2	C_L - Exp 2	C_D - Exp2
-6,11º	0,1470	0,0148	-6,13º	0,1420	0,0144
-5,08º	0,2590	0,0123	-5,09º	0,2600	0,0115
-4,09º	0,3810	0,0108	-4,07º	0,3660	0,0111
-2,98º	0,5070	0,0094	-3,09º	0,4630	0,0112
-1,96º	0,6300	0,0087	-2,02º	0,5720	0,0121
-1,05º	0,7250	0,0090	-0,99º	0,6670	0,0133
0,07º	0,8410	0,0096	-0,02º	0,7490	0,0145
1,11º	0,9480	0,0100	1,07º	0,8530	0,0158
2,10º	1,0520	0,0106	2,08º	0,9430	0,0171
3,17º	1,1620	0,0109	3,06º	1,0320	0,0183
4,25º	1,2740	0,0116	4,16º	1,1350	0,0201
5,14º	1,3600	0,0124	5,18º	1,2180	0,0221
6,20º	1,4590	0,0135	6,16º	1,2870	0,0248
7,26º	1,5470	0,0147	7,21º	1,3500	0,0294
8,26º	1,6170	0,0166	8,22º	1,3950	0,0362
9,27º	1,6730	0,0197	9,23º	1,4310	0,0486
10,28º	1,7040	0,0249	10,22º	1,4560	0,0667
11,26º	1,7170	0,0338	-	-	-
12,32º	1,7220	0,0505	-	-	-
13,31º	1,7230	0,0675	-	-	-
14,24º	1,7210	0,0800	-	-	-


Comparación C_L
2,00
1,80
1,60
1,40
1,20
1,00
0,80
0,60
0,40
0,20
0,00
-0,20
-8 -6 -4 -2 0 2 4 6 8 10 12 14 16 18
Lift - Caso 1
Lift - Caso 2
Lift - Caso 3
Lift - Exp 1
Lift - Exp 2


Comparación C_D
0,12
0,10
0,08
0,06
0,04
0,02
0,00
-8 -6 -4 -2 0 2 4 6 8 10 12 14 16 18
Drag - Caso 1
Drag - Caso 2
Drag - Caso 3
Drag - Exp 1
Drag - Exp 2


Comparación Curva polar
2,00
1,80
1,60
1,40
1,20
1,00
0,80
0,60
0,40
0,20
0,00
0,00 0,02 0,04 0,06 0,08 0,10 0,12
Curva polar - Caso 1
Curva polar - Caso 2
Curva polar - Caso 3
Curva polar - Exp 1
Curva polar - Exp 2

ANEXO III

SOFTWARE UTILIZADO

III.1. Introducción

El objetivo del *Anexo III* es describir las principales características del software utilizado en el presente proyecto, así como las posibilidades que ofrece.

Tal y como se ha comentado con anterioridad en la *Memoria*, durante el desarrollo del proyecto se han utilizado principalmente tres programas: *Salome* para la obtención del modelo 3D y el mallado del mismo, *OpenFOAM* como programa encargado de realizar el cálculo computacional y *Paraview* para el postprocesado (representación gráfica de los resultados).

Un aspecto a tener en cuenta es que tanto *Salome*, como *OpenFOAM*, como *Paraview* son open – source software, es decir, se trata de software distribuido y desarrollado libremente. Esto implica que los usuarios son capaces de leer, redistribuir y modificar el código fuente de estos programas. Este aspecto es realmente interesante en el caso de *OpenFOAM*, ya que se pueden modificar los *solvers* que vienen por defecto con el programa, añadiéndoles otras funciones de interés en el código fuente.

Sin embargo, mientras que para *Salome* y *Paraview* existen versiones compatibles con *Windows*, *OpenFOAM* solo está disponible para el sistema operativo *Gnu/Linux*. Este hecho puede suponer inicialmente una valoración negativa, pero se pueden encontrar versiones de *Ubuntu* (distribución *Gnu/Linux*) de rápida instalación y que están enfocadas para facilitar al usuario su utilización.

De este modo, en los siguientes capítulos se explica detalladamente el software utilizado en este proyecto, destacando las utilidades aprovechadas, pero también comentando las posibilidades que estos programas ofrecen y que no han sido empleadas.

III.2. Salome

III.2.1. Introducción a Salome

Salome es un open – source software que proporciona una plataforma genérica para el pre & post procesado en el campo de la simulación numérica. En la *Figura III.2.1* se muestra su página web oficial (www.salome-platform.org), desde la cual es posible descargar tanto los ejecutables como el código fuente.

Figura III.2.1: Página web oficial de *Salome*

Los principales módulos que ofrece *Salome* son: *Kernel*, *GUI* (graphical user interface), *Geometry*, *Mesh*, *Med*, *Post-processor*, *Supervisor* y *YACS*.

Los módulos más importantes y con una mayor relación con la temática de este proyecto son *GUI*, *Geometry*, *Mesh* y *Post-processor*. La utilización del resto de módulos disponibles se limita a unos fines muy específicos que no conciernen en el desarrollo de este proyecto y, por lo tanto, no se comentan posteriormente.

En este punto es importante destacar que *Salome* facilita al usuario una guía explicativa de todos y cada uno de los módulos. Se puede acceder a ella a través de la página web (www.salome-platform.org/user-section/documentation/current-release) o desde el mismo programa. En ella, se pueden encontrar todas las posibilidades y herramientas que ofrece cada uno de los módulos, así como una explicación clara y detallada de cómo utilizarlas.

Retomando el discurso sobre los módulos más relevantes (*GUI*, *Geometry*, *Mesh* y *Post-processor*), en apartados posteriores se realiza una descripción más amplia y detallada de sus principales características y de sus posibles utilidades.

Por otro lado, para un usuario familiarizado y con cierta experiencia en el manejo de programas CAD – CAE no debería suponer un gran esfuerzo el aprender a utilizar *Salome*. Presenta una interface genérica, 'amigable' y eficiente, lo que conlleva que el uso de cualquier herramienta o comando sea bastante intuitivo, evitando de esta forma perder tiempo en aspectos secundarios.

Además, en su página web oficial se pueden encontrar una serie de tutoriales donde se muestran con ejemplos prácticos las posibilidades que ofrece este software. Estos tutoriales resultan de gran ayuda a la hora de aprender a realizar modelos CAD en el módulo *Geometry*; también muestran la multitud de opciones que ofrece el módulo *Mesh* en el proceso de mallado; e incluso explican los principales estilos de visualización de resultados que presenta el módulo *Post-processor*.

Como complemento a estos tutoriales, se puede consultar el manual de usuario citado anteriormente.

Por último comentar que Salome dispone de un foro propio para usuarios registrados en el que se puede interactuar con el resto de usuarios preguntando dudas, compartiendo experiencias, ayudando a otras personas, etc. De este modo, personas con más experiencia o que han llevado a cabo trabajos similares pueden ser de gran ayuda en los problemas que puedan surgir a lo largo del estudio.

III.2.2. Módulo GUI

El módulo *GUI* (graphical user interface) es la herramienta gráfica que proporciona al usuario la capacidad de interactuar con los distintos comandos, herramientas, editores, visualizadores y procesadores de información disponibles en *Salome*.

Las principales utilidades que se pueden encontrar en la pantalla de inicio de *Salome* son las siguientes:

- Menú principal
- Barra de herramientas estándar
- Barra de herramientas de componentes
- Barra de herramientas de módulo
- Barra de herramientas de visualizador
- Listado de objetos
- Visualizador
- Consola *Phyton*

En el menú principal (*Main menu*) se puede encontrar una serie de barras de herramientas, algunas comunes y otras específicas del módulo en ese instante activado. Desde este menú y a través de las barras de herramientas es posible realizar cualquier operación o acción, aunque es cierto que en la mayoría de los casos es más rápido y cómodo utilizar las barras de herramientas presentes en la pantalla de inicio.

La barra de herramientas estándar (*Standard toolbar*) contiene los iconos que permiten una gestión rápida de los estudios (creación, operaciones de guardado, copiado y pegado de objetos).

La barra de herramientas de componentes (*Components toolbar*) permite el paso de un módulo de trabajo a otro, lo que a su vez conlleva el cambio de las barras de herramientas específicas de cada módulo.

En la barra de herramientas del módulo (*Module toolbar*) se encuentran los iconos específicos del correspondiente módulo. Visualmente aparecen los más importantes y de mayor uso, aunque también se puede acceder a los menos utilizados. Además, como en la mayoría de programas, el usuario tiene total libertad a la hora de configurar las barras de herramientas.

La barra de herramientas de visualizador (*Viewer toolbar*) permite el paso de un visualizador a otro en la pantalla principal de visualización. Esto se debe a que no todos los módulos de trabajo utilizan el mismo visualizador; por ejemplo, el módulo *Geometry* y el módulo *Mesh* utilizan visualizadores distintos.

En el listado de objetos (*Object browser*) se puede gestionar los objetos creados o importados a *Salome*.

El visualizador (*Viewer*) es el encargado de la visualización de los objetos.

La consola *Python* (*Python console*) funciona del mismo modo que la ejecución de un archivo en este lenguaje de programación. De hecho, existe la posibilidad de importar un script (archivo de texto) programado en *Python*, con idéntico resultado. La utilización de esta consola permite efectuar las operaciones mediante la introducción de un determinado comando. Es posible consultar los códigos y su estructura lógica en el manual de usuario, lo que debe de animar al uso de esta herramienta ya que puede resultar muy útil.

En la *Figura III.2.2* se puede observar la distribución de estas utilidades en la pantalla de inicio de *Salome*:

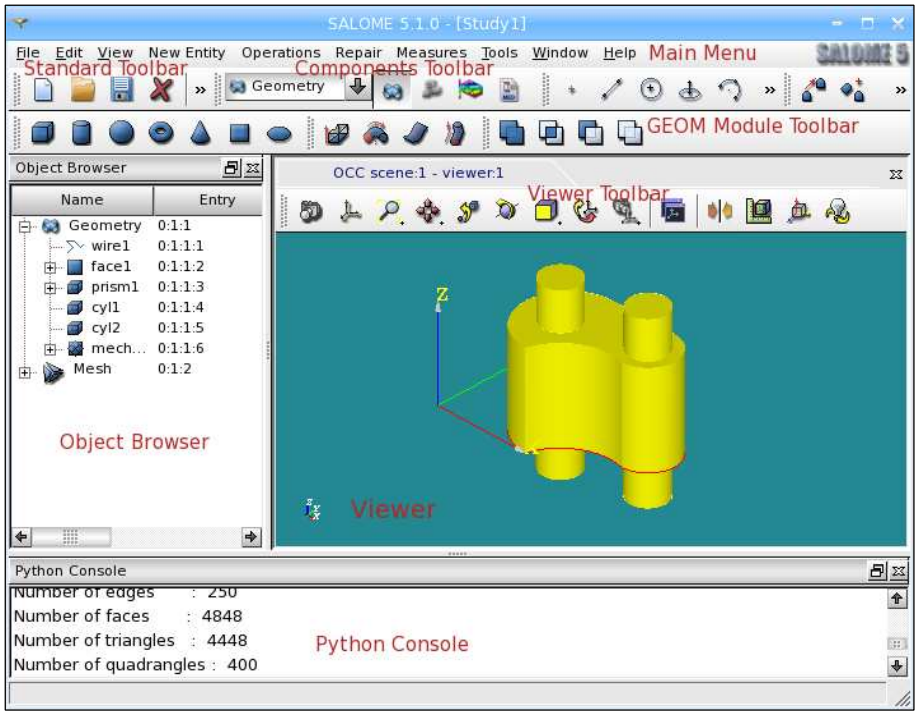


Figura III.2.2: Pantalla de inicio de *Salome*

III.2.3. Módulo Geometry

El módulo *Geometry* es la herramienta que se encarga de la gestión de los modelos CAD. Esta gestión incluye la importación – exportación de modelos con otros programas de la misma temática; la construcción, visualización, transformación y optimización de objetos geométricos; así como la observación de las principales propiedades geométricas de los objetos y otra información de interés a través de las herramientas de medición.

De hecho, una de las características que puede ser decisiva a la hora de escoger *Salome* como programa a utilizar es su interoperabilidad con otros programas de diseño CAD y análisis computacional. Así, *Salome* permite importar los modelos realizados en otro programa CAD, lo que evita que usuarios acostumbrados al empleo de un determinado software tengan que invertir tiempo en su aprendizaje. Para este tipo de usuario, *Salome* puede ser visto como una herramienta específica para el mallado de diseños CAD.

Sin embargo, lo comentado anteriormente no debe dar una mala imagen acerca de las posibilidades y la capacidad que ofrece *Salome* en su módulo *Geometry*. Si bien es cierto que no ofrece tantas herramientas específicas como las disponibles en otros programas orientados al diseño, es posible realizar cualquier modelo que se desee mediante una interface bastante intuitiva.

A continuación se muestra en la *Figura III.2.3* el aspecto general que presenta *Salome* cuando se activa el módulo *Geometry*. Se puede observar la gran cantidad de herramientas específicas de modelado disponibles:

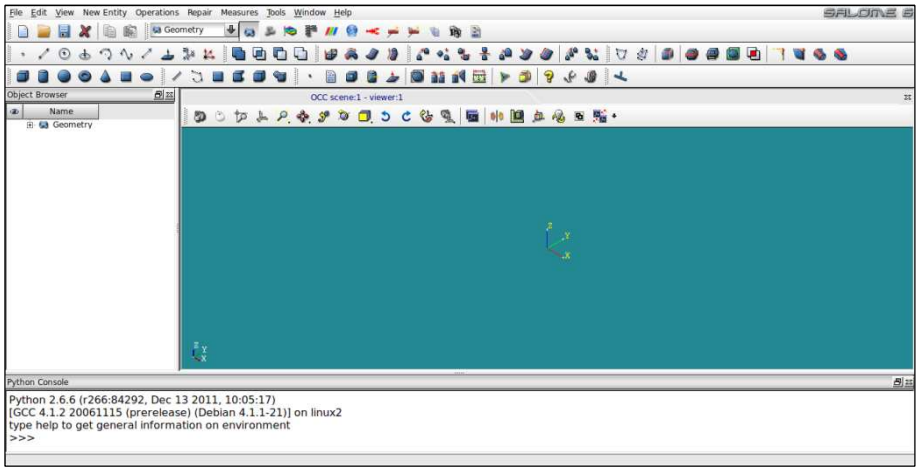


Figura III.2.3: Módulo *Geometry*

Como se ha comentado previamente, *Salome* dispone de una guía de usuario en la que, concretamente para el módulo *Geometry*, se puede consultar el funcionamiento de cualquier herramienta de modelado. En la guía aparece una breve descripción de cómo han de utilizarse acompañada de un ejemplo gráfico, tal y como se observa en la *Figura III.2.4*:

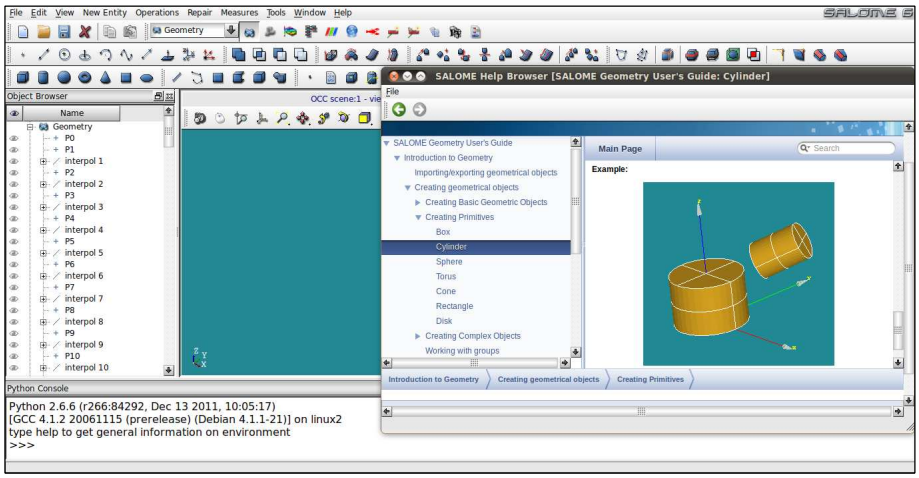


Figura III.2.4: Guía de usuario para el módulo *Geometry*

Debido a la gran relevancia que ha tenido en el modelado la posibilidad de trabajar con archivos programados en *Python*, se vuelve a remarcar su gran utilidad.

Así pues, en la *Memoria* y en el *Anexo I* se ha explicado que el diseño geométrico de los perfiles aerodinámicos se ha llevado a cabo mediante la importación de un script previamente programado en *Python*. Esto ha evitado la pérdida de una gran cantidad de tiempo en un proceso que, de haberlo realizado 'manualmente', habría supuesto un trabajo muy repetitivo y tedioso.

Para los usuarios con poca o nula experiencia en programación *Python* decir que en la guía se encuentra la estructura básica que ha de tener el archivo junto con todos los comandos necesarios. En la *Figura III.2.5* y en la *Figura III.2.6* se puede apreciar lo comentado:

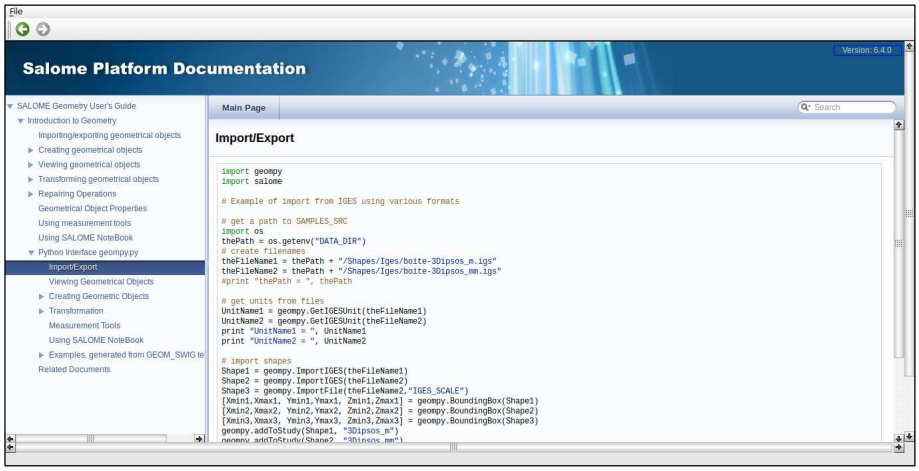


Figura III.2.5: Estructura básica en *Python*

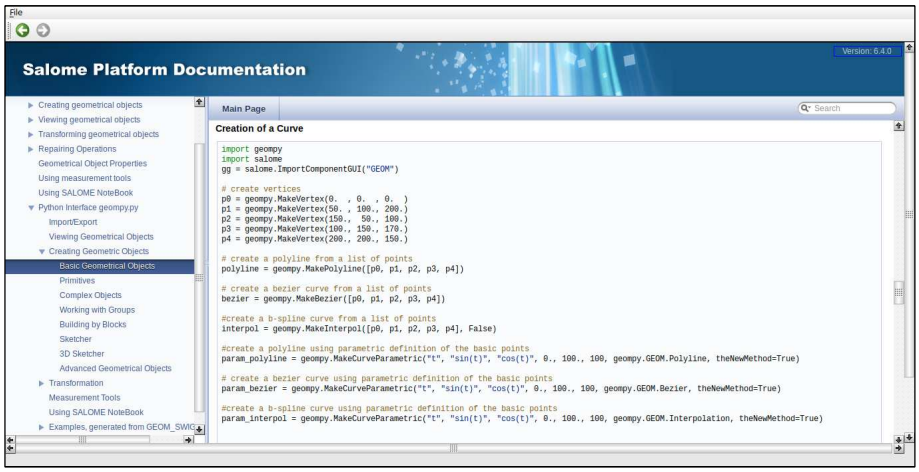


Figura III.2.6: Comandos específicos en *Python*

Además, por si esto fuera poco en la *User Section* de la página web oficial de *Salome* es posible descargar ejemplos de archivos (scripts) programados. También se puede encontrar fácilmente en la red páginas en las que se explican los principios básicos de programación *Python*.

III.2.4. Módulo Mesh

El módulo *Mesh* se ocupa de todas las operaciones que guardan relación con el mallado, como son la importación – exportación de mallados en distintos formatos; la creación y edición de la malla para modelos geométricos; la visualización de mallados y sus principales características; la creación y gestión de grupos de elementos de la malla; la medición de objetos mallados; etc. En la *Figura III.2.7* se observa el aspecto general del módulo *Mesh* en *Salome*:

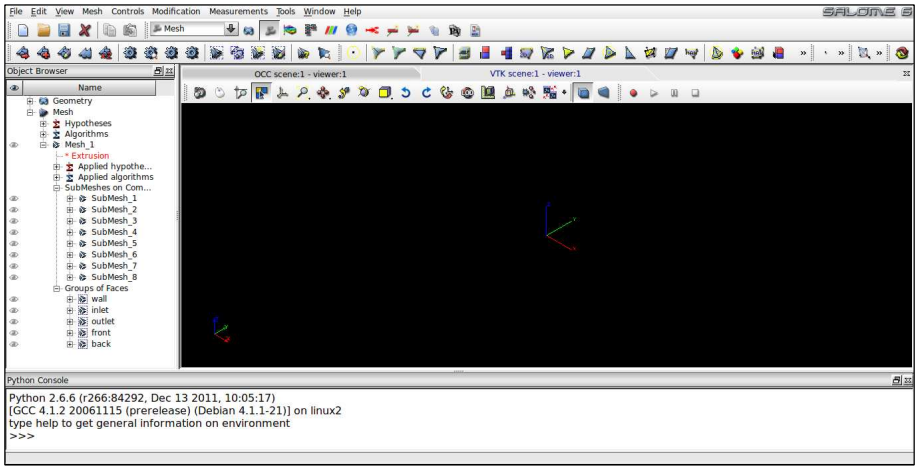


Figura III.2.7: Módulo *Mesh*

Así pues, no cabe duda de que otro de los puntos fuertes de *Salome* es su gran capacidad de mallado en elementos complejos. Hay disponible una cantidad considerable de algoritmos de mallado lo que, junto con la posibilidad de crear submallados en grupos de elementos geométricos específicos, permite obtener un mallado final tan complejo como sea necesario.

Esta afirmación ha quedado demostrada en el resultado final del mallado expuesto en la *Memoria* y en el *Anexo I*. Est3 mallado da una idea acerca de las posibilidades que *Salome* ofrece como 'mallador', aunque bien es cierto que en este proyecto no se ha aprovechado al máximo todo su potencial.

En este punto es necesario comentar la posibilidad de realizar el mallado a través del terminal *Python*. En este proyecto no se ha tenido en cuenta esta opción ya que este método supondría mayor tiempo y esfuerzo con casi total seguridad. No obstante, para los usuarios interesados decir que los comando específicos de mallado pueden encontrarse en la sección correspondiente de la guía de usuario.

Al igual que en el módulo anterior, la guía del usuario puede servir de apoyo para entender los principios básicos del módulo *Mesh*. En la *Figura III.2.8* y en la *Figura III.2.9* aparecen explicados los procesos de creación de mallados y submallados.

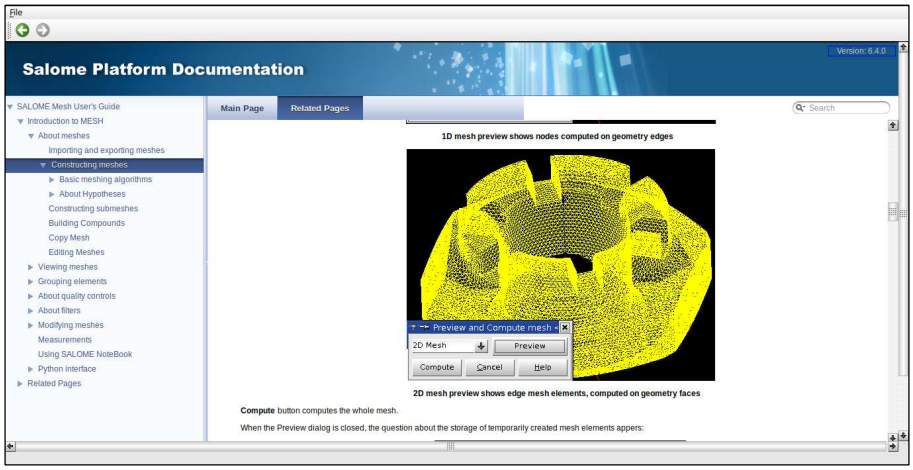


Figura III.2.8: Proceso de mallado

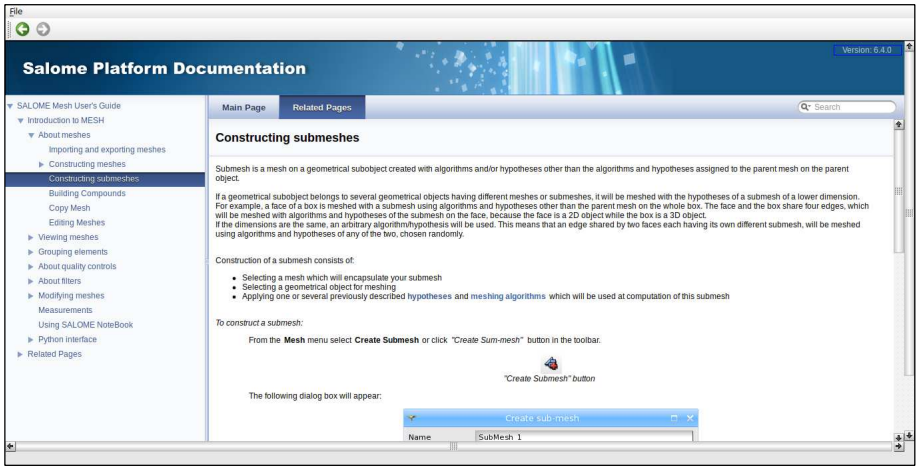


Figura III.2.9: Proceso de submallado

También se puede consultar los algoritmos de mallado disponibles y cómo se comporta cada uno de ellos, tal y como se muestra en *Figura III.2.10*:

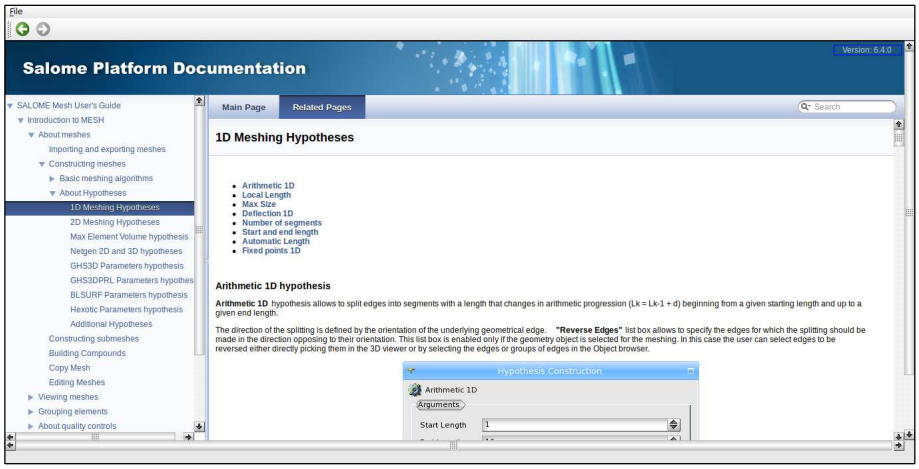


Figura III.2.10: Algoritmos de mallado

III.2.5. Módulo Post-processor

El módulo *Post-processor* (*Post-Pro*) es la herramienta que se encarga del análisis y post-procesado de los resultados obtenidos en las simulaciones numéricas. Este módulo trabaja con archivos en formato MED, los cuales incluyen una descripción geométrica del objeto, el mallado del mismo, así como un conjunto de valores de variables físicas correspondientes a las celdas del mallado.

En la *Figura III.2.11* que sigue se muestra el aspecto general que ofrece *Salome* al trabajar con el módulo *Post – Pro*:

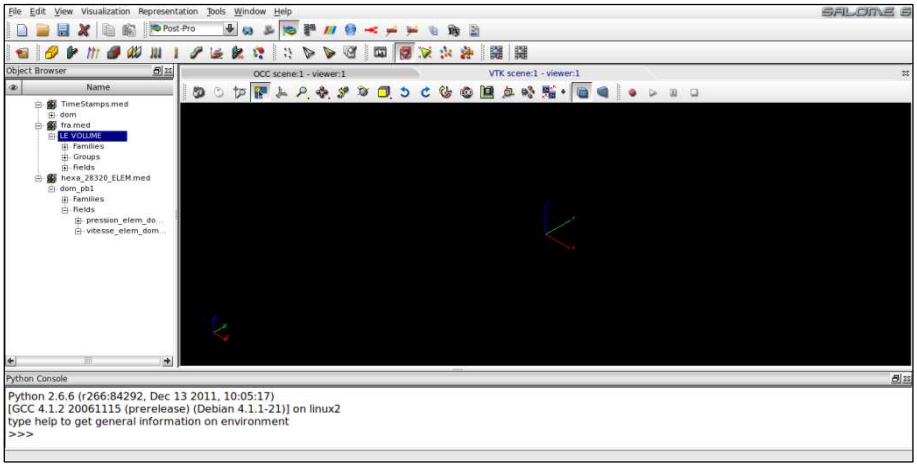


Figura III.2.11: Módulo *Post – Pro*

Como ya es sabido, durante la realización de este proyecto se ha utilizado *Salome* para obtener el modelo 3D del sistema a estudiar, así como su correspondiente mallado. Sin embargo, para el análisis del cálculo computacional se ha elegido otro programa distinto (*Paraview*). Esta decisión, como se detalla posteriormente, se debe principalmente a la rapidez y sencillez para visualizar los resultados con *Paraview* una vez que el programa de simulación numérica *OpenFOAM* ha terminado de calcular.

Sin embargo, en este punto se tiene en cuenta la posibilidad que ofrece *Salome* para el análisis de resultados procedentes del cálculo computacional. Con toda seguridad habrá usuarios que por unas razonas u otras decidan utilizar el módulo *Post – Pro* para el análisis de datos numéricos.

Así pues, se recomienda consultar la sección correspondiente en la guía de usuario, al igual que en módulos anteriores. Entre la información relacionada con el post-procesado que se puede encontrar destacan las múltiples opciones de visualización de datos, así como la estructura que deben de guardar las distintas carpetas para una lectura correcta de los datos.

Con el fin de reflejar de un modo más ilustrativo las opciones gráficas que *Salome* ofrece en su módulo *Post-Pro*, en la *Figura III.2.12* se numeran las diferentes formas de representación, y en la *Figura III.2.13* y en la *Figura III.2.14* se muestran los resultados gráficos que se pueden conseguir:

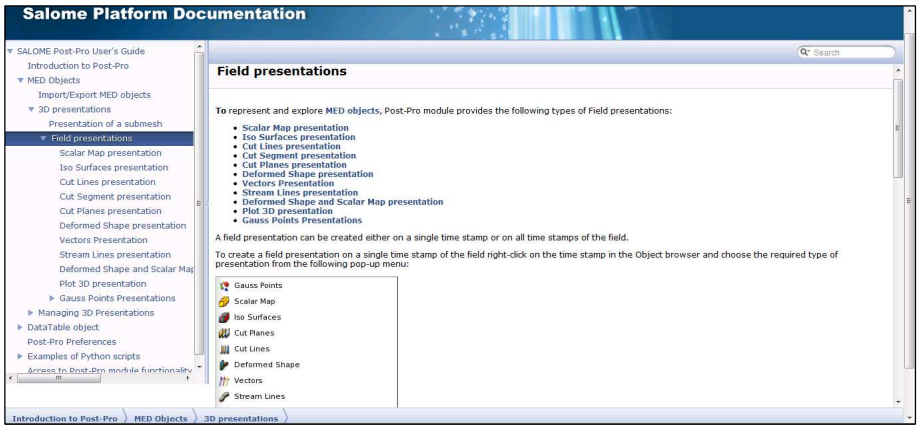


Figura III.2.12: Modos de representación

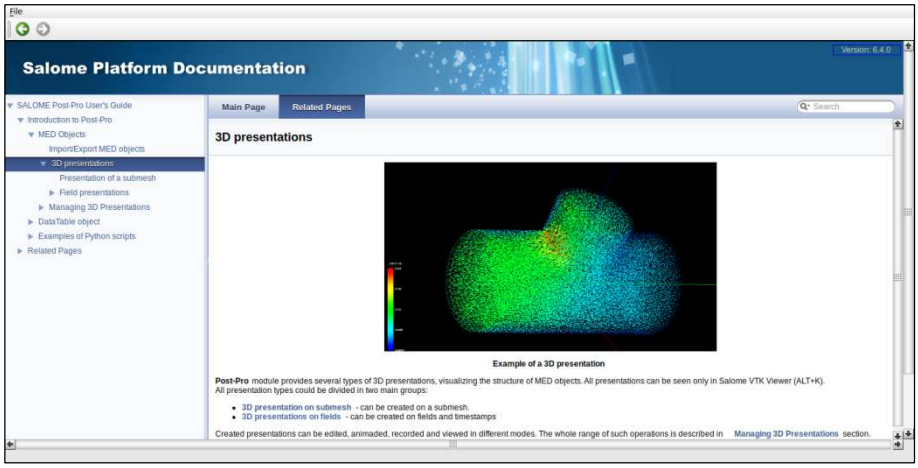


Figura III.2.13: Resultados gráficos I

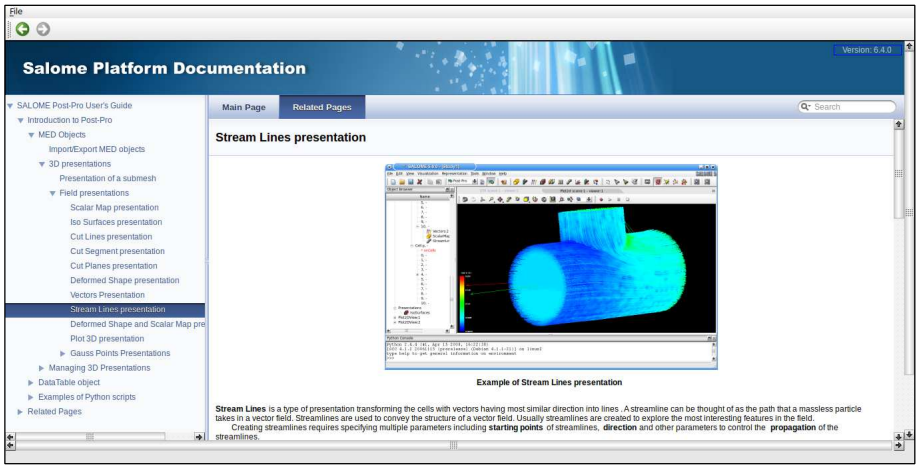


Figura III.2.14: Resultados gráficos II

III.3. OpenFOAM

III.3.1. Introducción a OpenFOAM

OpenFOAM (*Open Field Operation and Manipulation*) es un open – source software dedicado al cálculo CFD. Su empleo abarca gran parte de las distintas áreas de la ciencia y la ingeniería, con fines tanto académicos como comerciales.

OpenFOAM contiene una cantidad importante de solvers, donde el usuario tiene libertad para modificarlos, personalizarlos e incluso implementar los suyos propios. Estos solvers numéricos son capaces de simular cualquier tipo de flujo por complejo que sea, lo que incluye reacciones químicas, sistemas turbulentos, transferencia de calor, etc. También puede analizar sistemas relacionados con el campo del electromagnetismo y problemas estructurales. Además, dispone de herramientas específicas para el mallado y para el pre y post – procesado.

Como se ha comentado anteriormente, *OpenFOAM* solo está disponible para el sistema operativo *Gnu/Linux*. Las instrucciones para su instalación se pueden encontrar en su página web oficial (www.openfoam.com), la cual se muestra en la *Figura III.3.1*:

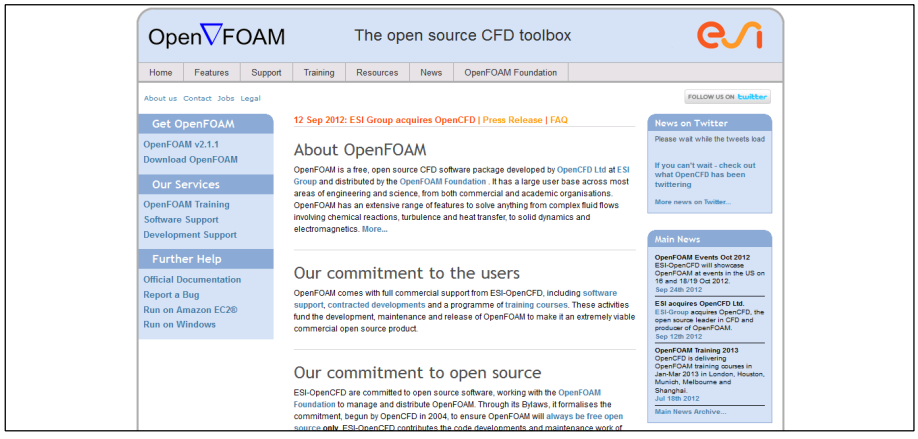


Figura III.3.1: Página web oficial de *OpenFOAM*

En esta página web se puede encontrar una gran cantidad de información relacionada con este software. Así, es posible consultar el manual de usuario (www.openfoam.org/docs/user/); ver los solvers disponibles junto con una breve descripción del fenómeno físico para el que han sido diseñados; acceder a los modelos físicos empleados a través de las librerías, etc.

Acerca del modo de empleo de *OpenFOAM* por parte del usuario, decir que este software no presenta una interface genérica como a la que la mayoría de gente está acostumbrada. Su modo de trabajar está basado en la ejecución de una serie de comandos específicos en el terminal de *GNU/Linux*, así como en una disposición característica de carpetas y archivos de texto.

No obstante, aprender a utilizar este programa de cálculo no es tan difícil como pueda parecer. De hecho, a través de la guía de usuario es posible acceder a una serie de tutoriales con los que se puede asimilar los aspectos básicos de ejecución, simulación y post-procesado de resultados. Para ello se utilizan unos ejemplos prácticos incluidos en la instalación del programa.

En este punto, por ser *OpenFOAM* el software utilizado con un modo de empleo más complejo, se recomienda visitar una de las páginas web más importantes sobre el cálculo computacional, como es *CFD Online* (www.cfd-online.com/).

En la *Figura III.3.2* aparece la página principal del foro, donde se puede encontrar y consultar información sobre cualquier tema relacionado con el cálculo CFD. Esta sección está dividida en una serie de áreas específicas para la discusión de temas como cálculo CFD; hardware para cálculo numérico; software de modelado CAD; software para el mallado; programas de análisis computacional; software específico para la visualización y post-procesado; etc.

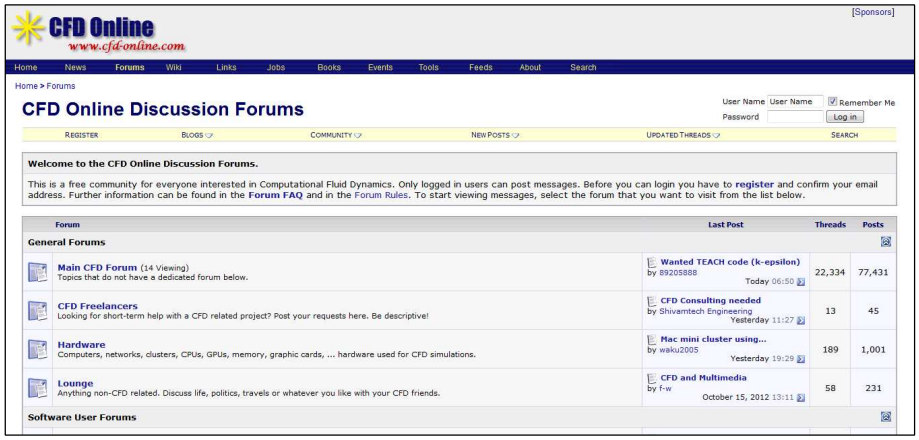


Figura III.3.2: Página web oficial de *CFD Online*

Al igual que en el foro propio de *Salome*, se sugiere al usuario su utilización en caso de duda o ante los problemas que puedan aparecer en el desarrollo de estudios de temática como la aquí discutida. De hecho, el foro de *CFD Online* es seguido y utilizado por una gran cantidad de personas a nivel internacional. En muchos casos es posible encontrar una conversación en la que se discute un problema similar al propio, y en caso contrario, se puede plantear la situación y valorar las opiniones que ofrecen otras personas con conocimiento e interés en el tema.

III.3.2. Estructura de archivos para la simulación

III.3.2.1. Estructura general

En este apartado se describe la estructura de archivos necesaria para que al ejecutar una aplicación o solver, éste funcione correctamente. A modo de resumen se presenta la *Figura III.3.3* obtenida en el correspondiente apartado de la guía de usuario:

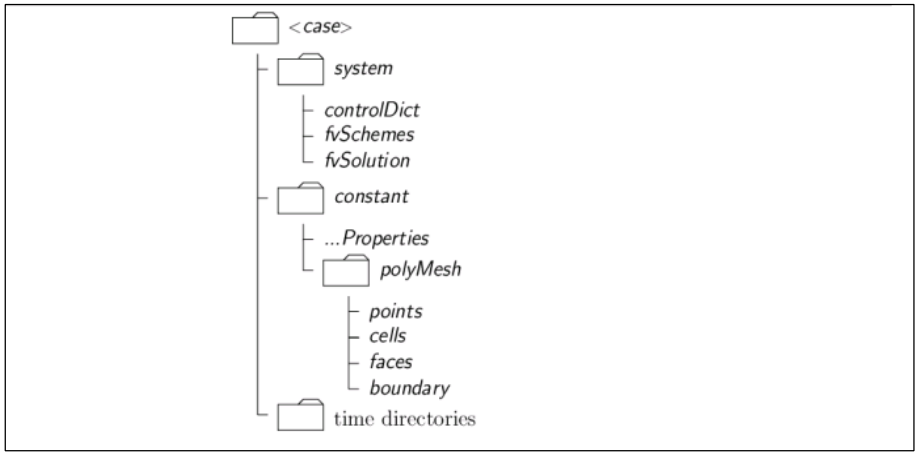


Figura III.3.3: Estructura de archivos

De este modo, sin perder de vista el esquema básico anterior, se procede a detallar de forma un poco más detallada cada uno de los archivos y subdirectorios que debe presentar el directorio principal del estudio *case*.

Recordar en este punto que el seguimiento de los tutoriales que facilita *OpenFOAM* puede resultar de gran ayuda para entender la estructura de archivos que guarda cada caso de estudio.

Por último, incidir en el hecho de que en los apartados que siguen se realiza una descripción básica del contenido necesario para poder efectuar el análisis numérico. La correcta codificación de cada uno de los archivos precisa de un estudio exhaustivo del manual de usuario; o en el caso de que el fenómeno físico a analizar coincida con alguno de los ejemplos disponibles en *OpenFOAM*, podría bastar con adaptar el código a las condiciones buscadas.

III.3.2.2. Directorio base (*case*)

Como se ha podido apreciar en la imagen, la carpeta denominada *case* es el directorio en el cual se encuentran todos los demás archivos y subdirectorios del estudio. Normalmente el usuario asigna un nombre relacionado con el estudio a realizar a este directorio *case*. Este directorio debe de disponer de los subdirectorios *system*, *constant* y *time directories*. Éstos al mismo tiempo contienen una serie de carpetas y archivos tal y como se comenta posteriormente.

A pesar de que no debería haber problemas relacionados con la ubicación de los directorios base, se recomienda situarlos dentro de una carpeta con nombre *run*. A su vez, esta carpeta *run* se dispone en el directorio propio del usuario de *OpenFOAM*, quedando finalmente: *$home/User/OpenFOAM/user-2.1.1/run/case*.

Antes de hablar sobre los subdirectorios citados con anterioridad, decir que la ejecución del solver se efectúa desde este directorio *case*. Para ello se abre el terminal de *GNU/Linux* y se introduce el nombre del solver a utilizar, lo que da lugar al comienzo del cálculo computacional.

III.3.2.3. Subdirectorio *system*

En el subdirectorio *system* se establecen los parámetros asociados al proceso de resolución numérica. Esta carpeta debe contener al menos los archivos *controlDict*, *fvSchemes* y *fvSolution*. El archivo *controlDict* se encarga de los parámetros de control de la ejecución, como son el tiempo de inicio y de fin de ejecución, el paso temporal y la monitorización de datos. Por otro lado, en el archivo *fvSchemes* se seleccionan los esquemas de discretización a utilizar en el proceso de cálculo. Por último, en *fvSolution* se determinan los solvers para cada una de las ecuaciones, las tolerancias, los coeficientes de relajación y otros controles de algoritmo.

Para mostrar el aspecto general de estos archivos, en la *Figura III.3.4* aparece parte del código de un archivo *controlDict*:

Figura III.3.4: Estructura de código

III.3.2.4. Subdirectorio *constant*

El subdirectorio *constant* contiene una descripción completa del mallado en una carpeta llamada *Polymesh*, así como un conjunto de archivos donde se especifican una serie de propiedades físicas relativas al análisis numérico a realizar.

Volviendo a la carpeta *Polymesh*, esta descripción del mallado se realiza a través de los archivos *points*, *cells*, *faces*, *neighbour*, *boundary*, *etc*. Estos archivos contienen una gran cantidad de elementos geométricos que forman parte del mallado, además de información que los sitúa en el espacio, ordena y relaciona entre ellos.

III.3.2.5. Subdirectorios *time*

Cada uno de los subdirectorios de tiempo (*time*) dispone de un conjunto de archivos con datos numéricos correspondientes a campos de velocidades, presiones y a cualquier otra variable física presente en el estudio. Estos datos numéricos pueden proceder por parte del usuario como valores iniciales y condiciones de contorno, o pueden haber sido creados durante la ejecución en *OpenFOAM* como resultado del cálculo computacional.

Así pues, en muchos casos es necesaria la imposición de unas condiciones iniciales para llevar a cabo las simulaciones numéricas. En este caso, todos los valores iniciales de las distintas variables que participan en la simulación son guardados en una serie de archivos dentro de una carpeta designada como *0*.

Los resultados numéricos obtenidos por *OpenFOAM* aparecen en carpetas cuyos nombres son números, como por ejemplo *1603* ó *2805*. El número guarda relación con el número de iteración en que finaliza el cálculo, ya sea porque se ha alcanzado cierta precisión o un determinado número de iteraciones; o en el caso de analizar un problema transitorio, el número corresponde al tiempo físico en el que se almacenan los datos.

Por último, avanzar que estos subdirectorios *time* son las carpetas a las que accede el programa de visualización de datos *ParaView* a la hora de representar gráficamente la evolución de las variables dentro del dominio de estudio.

III.4. Paraview

III.4.1. Introducción

Paraview es un open – source software que se ocupa del análisis y visualización de datos. Este programa permite realizar cualquier tipo de gráfico, así como renderizados 2D y 3D a partir de grandes paquetes de datos.

Para ello, al igual que otras muchas aplicaciones de visualización avanzadas, *Paraview* utiliza la librería *VTK* (*Visualization Tool Kit*) como procesador de datos e imágenes y motor de la renderización y visualización.

Los objetivos que se desean alcanzar con este software son:

- Desarrollar una aplicación open – source para la visualización multiplataforma
- Ayudar en la distribución de modelos computacionales para el procesamiento de grandes paquetes de datos
- Crear una interface de usuario abierta, flexible e intuitiva
- Desarrollar una arquitectura extensible basada en estándares abiertos

La instalación de *Paraview* se puede realizar de forma conjunta a *OpenFOAM*, por lo que en este punto se vuelve a remitir a su página web oficial. No obstante, como es de esperar, su instalación también puede gestionarse desde la página web oficial de *Paraview* (www.paraview.org), la cual se muestra en la *Figura III.4.1*:

Figura III.4.1: Página web oficial de *Paraview*

En cuanto al aspecto visual que presenta *Paraview*, decir que dispone de una interface genérica y bastante intuitiva, similar a la que ofrece *Salome*. Sin embargo, esto no significa que desde el primer momento el usuario vaya a ser capaz de manejar correctamente este software y aprovechar todo su potencial. No hay que perder de vista el gran número de herramientas para la visualización disponibles, así como la enorme cantidad de parámetros de diseño asociados.

Es posible consultar el funcionamiento de todas y cada una de las herramientas presentes en *Paraview*, así como cualquier otra información de interés en la guía de usuario (www.paraview.org/paraview/help/documentation.html). Por otro lado, los tutoriales disponibles de *OpenFOAM* pueden resultar de gran ayuda en el proceso de aprendizaje, ya que en éstos también se explica el funcionamiento de algunas de las herramientas gráficas de *Paraview*.

En el siguiente apartado se describe con mayor detalle las características más relevantes del programa, las principales herramientas de visualización y su distribución, así como otra información de interés.

III.4.2. Paraview

Como se ha comentado con anterioridad, mientras que *Paraview* está disponible para varios sistemas operativos (*Windows*, *Mac*, *GNU/Linux*), *OpenFOAM* es más restrictivo en este aspecto. De este modo, la necesidad de trabajar con *GNU/Linux* para realizar el cálculo numérico ha sido determinante en la decisión de utilizar *Paraview* en su versión disponible para *GNU/Linux*. A pesar de esto, lejos de suponer un problema, esta forma de trabajar resulta realmente cómoda y sencilla.

Así pues, en el momento en el que finaliza el análisis numérico es posible comprobar los resultados obtenidos rápidamente. Para ello, se introduce en el terminal de *GNU/Linux* el comando correspondiente que inicia *Paraview* y se seleccionan los datos a observar así como las opciones de visualizado. En ningún caso es necesario efectuar un tratamiento especial a los datos para su posterior visualización.

Anteriormente se ha adelantado que este software accede a las carpetas *time* del directorio base para el tratamiento y procesado de la gran cantidad de datos obtenidos en el cálculo numérico. Cada uno de los subdirectorios *time* dispone de tantos archivos como variables físicas intervienen en el cálculo, de modo que *Paraview* es capaz de representar gráficamente la evolución de cada una de estas variables en el dominio de estudio.

Asimismo, gracias a los archivos que contiene la carpeta *constant*, este programa dispone de la información relativa a la geometría característica del dominio, haciendo posible su representación.

Una vez explicado el procedimiento para iniciar *Paraview* y las condiciones que se deben cumplir para una lectura correcta de los datos a representar, se procede a la descripción del programa.

El aspecto general que presenta este software en su pantalla de inicio se puede observar en la *Figura III.4.2*:

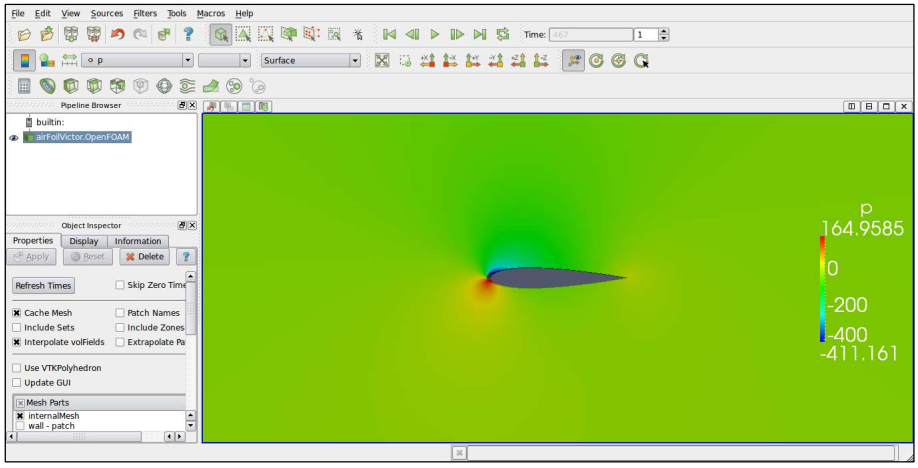


Figura III.4.2: Pantalla principal de *Paraview*

Las principales utilidades son el navegador de objetos (*Pipeline browser*); la ventana dedicada a la visualización de datos; las barras de herramientas de la parte superior; así como los paneles de propiedades, de visualización y de información (*Properties*, *Display* e *Information panels*, respectivamente).

El navegador de objetos (*Pipeline browser*) se encarga de la gestión de los objetos creados a partir de la geometría disponible en el directorio *constant* de *OpenFOAM*, además de las distintas manipulaciones realizadas a éstos objetos mediante las distintas herramientas y filtros que posee *Paraview*.

La ventana de visualización es el lugar donde el usuario puede ver representados los resultados obtenidos en el cálculo computacional. En esta ventana se puede mostrar la evolución en el dominio de cualquiera de las variables, con el estilo de visualización que se desee. Asimismo, es posible realizar una subdivisión de esta ventana, lo que puede resultar útil para observar simultáneamente la evolución de una variable junto con distintos gráficos de interés.

Como ha quedado reflejado en la imagen anterior, *Paraview* pone a disposición del usuario una gran cantidad de iconos en las barras de herramientas de la parte superior. A pesar de que no se pretende explicar la función que cada uno de estos iconos desempeña, en la *Figura III.4.3* se presenta una clasificación según su funcionalidad.

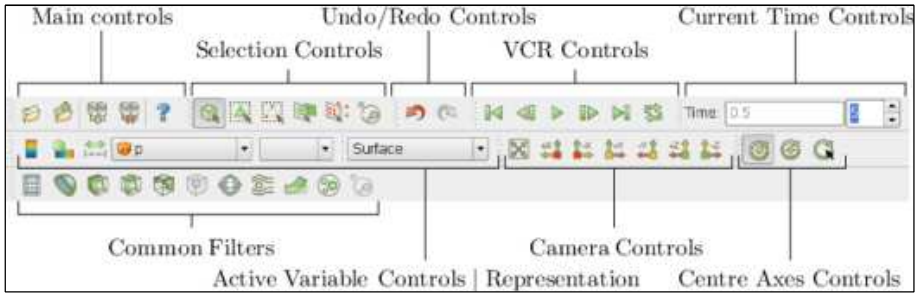


Figura III.4.3: Clasificación de controles

Destacar por encima del resto de herramientas los controles VCR, los cuales permiten el movimiento entre las distintas carpetas temporales *time*; los controles de activación de variables, con los que se puede seleccionar rápidamente la variable a visualizar, acompañándola de una leyenda con sus respectivos valores numéricos y; los filtros comunes, con los cuales se efectúan representaciones de líneas de corriente, campos vectoriales, isolíneas, planos de corte, etc.

Por otro lado, en el panel de propiedades (*Properties panel*) se seleccionan los datos de entrada que se quieren visualizar, como carpetas temporales *time*, regiones del mallado, variables físicas, etc.

El panel de visualización (*Display panel*) controla la representación visual del elemento u objeto activo. En esta pestaña es posible seleccionar y editar los distintos estilos de visualización de datos.

Por último, en el panel de información (*Information panel*) se puede encontrar información relacionada con la geometría del mallado y los valores numéricos de la variable representada.

III.5. Comentarios finales

Antes de dar por finalizado este *Anexo III* es necesario volver a incidir en una serie de aspectos comentados a lo largo de este documento.

En primer lugar, recordar que el objetivo perseguido con la elaboración de este anexo no es otro que describir las características más relevantes del software utilizado durante el proyecto, incluyendo aquellas herramientas o utilidades que no han sido empleadas.

En ningún momento se ha pretendido que este documento explique hasta el mínimo detalle el uso de *Salome*, *OpenFOAM* o *Paraview*. Tampoco ofrece soluciones para problemas concretos que puedan surgir en el desarrollo de un proyecto o estudio de características semejantes.

A los usuarios que puedan encontrarse en este tipo de situación, se insiste nuevamente en la consulta de las guías de usuario mencionadas anteriormente. Asimismo se les recuerda la existencia de tutoriales y foros específicos que pueden resultar de gran ayuda.

Por otra parte, dar una valoración más que positiva a todos los programas utilizados en este proyecto. Obviamente, al tratarse de software de temática técnica se requiere un cierto tiempo para aprender a usarlo correctamente, a pesar de que no se ha aprovechado al máximo todo su potencial. Se agradece el esfuerzo realizado por parte de los responsables de su desarrollo por todas las facilidades a la hora de su aprendizaje y utilización.

Finalmente, expresar el deseo de que este documento sirva de ayuda a los usuarios interesados en el cálculo CFD y que tengan intención o estén en este momento desarrollando trabajos similares.