

SOFTWARE LCC_SIMULATOR (Version 1.0)

Universidad Nacional de San Agustín - Arequipa

20 de diciembre de 2019

1. Introducción

El software **LCC_Simulator**(Version 1.0), es un programa de computador que realiza simulaciones numéricas sobre domíñios uni, bi y tri dimensionales de modelos matemáticos que utiliza la ecuación diferencial parcial conocida como la Ecuación de Poisson. Desarrollado a pedido del Prof. Mg. Ricardo Hancco coordinador del Laboratorio de Computación Científica de la Universidad Nacional de San Agustín.

De acuerdo con su solicitud desarrollamos el software para dos dimensiones sobre un cuadrado con centro en el origen y de lados con dos unidades de medida, $\Omega = [-1, 1] \times [-1, 1]$. El pedido original fue para simulaciones en dos dimensiones, sin embargo se puede realizar simulaciones en una dimensión sobre el intervalo $\Omega = [-1, 1]$ y sobre el cubo $\Omega = [-1, 1] \times [-1, 1] \times [-1, 1]$. Domíñios mas generales pueden ser adaptados dando un archivo de datos de la malla generado en el formato .dump a partir del software GID (<https://www.gidhome.com/>). Un requisito principal fue desarrollar el código en lenguaje de programación orientado a objetos C++ (ANSY), lo que fue cuidadosamente atendido. El código es multiplataforma, es decir, puede ser compilado en ambiente linux (sugerimos utilizar el compilador g++, para el sistema operacional Ubuntu), en ambiente windows (sugerimos utilizar el ambiente Visual Studio Express), y en ambiente macOS (sugerimos utilizar el ambiente XCode). De cualquier forma el código considera los archivos CMakeLists.txt, que son la base para rodar el programa CMake (<https://cmake.org/>) que generará el ejecutable adecuado al ambiente del sistema operacional del usuario, por lo que el código es multiplataforma y puede utilizar una gran variedad de compiladores.

Otra solicitud importante, elaborar el código utilizando las herramientas del ambiente computacional científico NeoPZ (<http://www.labmec.org.br/wiki/neopz/start>). En esa página puede encontrarse la documentación del ambiente y bajar la última versión del código. El NeoPZ es un ambiente computacional orientado a objetos para el desarrollo de simulaciones en elementos finitos, desarrollado por el Dr. Philippe Devloo en el laboratorio de investigación que él coordina: Labmec (Laboratório de Mecánica Computacional), en la UNICAMP (Universidade Estadual de Campinas). El objetivo de este ambiente computacional es proveer al usuario con un sistema modular coherente de herramientas de las tecnologías avanzadas en elementos finitos. Permite al usuario desarrollar simulaciones sofisticadas en un corto tiempo, preocupándose principalmente del modelo y sus problemas en análisis.

En este ambiente es posible utilizar diferentes espacios de aproximación para obtener simulaciones numéricas de modelos matemáticos en ecuaciones diferenciales parciales. El NeoPZ no es propiedad del autor del presente software, por lo tanto, no existe ningun derecho sobre esa biblioteca, ni del autor, ni del solicitante. La propiedad del software LCC_Simulator no da ningún derecho sobre la biblioteca NeoPZ, ni sobre las otras bibliotecas que utiliza el NeoPZ, como la biblioteca pthread. El software desarrollado considera una copia de las bibliotecas en modo Debug y Release de la biblioteca NeoPZ, tanto para el sistema operacional MacOs como para Windows.

2. Instalación y ejecución

La distribución considera un directorio LCC_UNSA contenido en cuatro directorios: bin, LCC_SimulatorIO, LCC_Simulator_Code y MANUAL_LCCSimulator.

En el directorio bin existen ejecutables del simulador que pueden ser copiados a la pasta de preferencia del usuario. Como están siendo anexados los códigos fuente, por practicidad colocamos un ejecutable LCC_Simulator.exe para la utilizar en sistema operacional Windows y el LCC_Simulator para utilizar en sistema operacional MacOs.

El segundo directorio LCC_SimulatorIO es importante para la entrada y salida de datos. En este directorio existe un subdirectorio InputData (/LCC_SimulatorIO/InputData) en el cual debe existir un archivo llamado «userdata.txt». En la próxima sección se especifica los campos en este archivo para que el ejecutable reconozca correctamente las solicitudes del usuario. Para el funcionamiento del simulador es necesario copiar este directorio (/LCC_SimulatorIO/InputData/). En el ambiente Windows este directorio debe estar en el mismo drive (C:/, D:/, etc) en el cual se ha copiado el ejecutable LCC_Simulator. En el ambiente MacOS deberá estar en el directorio Documents (/Users/.../Documents/LCC_Simulator/InputData). Conforme fue informado, el ejecutable puede ser copiado para cualquier directorio del usuario. Apenas el directorio /LCC_Simulator/InputData es necesario copiar en la posición referenciada.

En este segundo directorio el ejecutable creará un directorio llamado /LCC_Simulator/SimulationResults para almacenar los resultados de las soluciones aproximadas que serán generadas (solo será creado si él no existe). En el interior de ese directorio serán creados los archivos de las simulaciones en directorios diferentes, para evitar confusión. Cada simulación creará un directorio diferente con un nombre iniciado por una 'R' y considerando el segundo, minuto, hora, dia, mes y año de la simulación realizada. Conforme solicitado, los resultados serán almacenados en el formato .vtk (Visualization Tool Kit, que pueden ser visualizados con el software Paraview - <https://www.paraview.org/>). El segundo directorio también contiene un subdirectorio Dominios_GID con archivos ejemplos de diversas mallas de dominios generados por el software GID (existe una versión trial del GID). Los archivos que consideran los dominios solicitados están en el subdirectorio Cuadrado (Cubo que puede ser utilizado para realizar simulaciones en tres dimensiones). Ese subdirectorio es opcional.

El tercer directorio contiene las bibliotecas necesarias para el simulador y el código fuente del mismo, por lo tanto el usuario puede generar nuevos ejecutables del LCC_Simulator o para realizar modificaciones y actualizaciones de este simulador en el sistema operacional del usuario. Este código contiene los archivos CMakeLists.txt que permiten el uso del software libre CMake (www.cmake.org) con el que podemos compilar en multiplos sistemas operacionales - multiplataforma.

El directorio también contiene las bibliotecas externas: pthread (/LCC_Simulator/externallibs/pthread) y pzlib (/LCC_Simulator/externallibs/pzlib). En esos directorios están los archivos .h ((/LCC_Simulator/externallibs/pthread/include y (/LCC_Simulator/externallibs/pzlib/include) de esas dos bibliotecas y los archivos binarios de esas dos bibliotecas (/LCC_Simulator/externallibs/pthread/lib y (/LCC_Simulator/externallibs/pzlib/lib)). La biblioteca pthread funciona tanto para la versión de depuración (Debug) quanto para la versión final (Release) y también es independiente del sistema operacional. La biblioteca NeoPZ es dependiente del sistema operacional, por lo tanto, en el directorio correspondiente existe la versión para MacOS, para Windows y tanto para la versión Debug y para la versión Release. Esos archivos binarios son anexados, apenas por comodidad del usuario, pues pueden ser bajados y compilados segun la necesidad del usuario y para el sistema operacional (macOS, Ubuntu y Windows) y el compilador de preferencia. Estas copias binarias son las mas utilizadas por la comunidad científica. Según su necesidad puede copiar la versión mas adecuada para el directorio /externallibs/pzlib/lib .

Los códigos fuente del simulador desarrollado son de propiedad total y absoluta del solicitante y pagador del desarrollo. En este software se han desarrollado las simulaciones de los modelos de la Ecuación de Poisson para tres funciones fuente (source) para obtener las tres soluciones solicitadas. Estas simulaciones son descritas en la sección de Resultados. La función mas importante del software es que consigue realizar simulaciones utilizando espacios de funciones aproximantes continuas *H*1 (conocido com el método de elementos finitos clásico), funciones aproximantes discontinuas *GD* (conocido como el método de Galerkin discontinuo), y funciones aproximantes de la variable primal y de la variable flujo *HdivH*1 (conocido com el método de elementos finitos mixto).

El último directorio contiene el manual del mismo.

2.1. Guia de uso do userdata.txt

El archivo userdata.txt considera los datos del usuario siempre a continuación de un símbolo '@' (arroba). Esto significa que el dato que sigue al símbolo arroba es un dato del usuario. Los siguientes datos son esperados por el ejecutable (existen valores por default)

1. Para informar sobre la malla geométrica se utiliza el campo: FROM GID FILE. Si la malla está en un archivo *.dump (Gid generator) deve informarse a continuación del símbolo arroba el 1 y a seguir el nombre del archivo

.dump, que debe estar junto al archivo userdata.txt . Si la malla debe ser construida de forma simple utilice el 0 (zero). En el archivo ejemplo tiene algunas informaciones adicionales para facilitar su uso.

FROM GID FILE? /0 whether is simple mesh, /1 whether the mesh is from gid file, then give the gid filename
@1 Triangles.dump

En esta línea se informa que la malla está en el archivo Triangles.dump

Para el caso de una malla simple deberá expresarse:

FROM GID FILE? /0 whether is simple mesh, /1 whether the mesh is from gid file, then give the gid filename
@0 Triangles.dump

El nombre del archivo puede o no estar presente, para el caso de @0, este será desconsiderado. El executable buscará el próximo arroba.

2. El segundo campo informa el tipo de elemento geométrico en que ha sido dividida la malla. Esta información es útil principalmente para la generación de las mallas simples. En el caso de dar un archivo .dump, el tipo de elemento debe ser coherente con el tipo de elementos finitos de la malla desde el archivo. En el archivo ejemplo tenemos la siguiente información:

GEOMETRICAL INFORMATION Type of geometrical element:

```
// 1 - linear(lin), 3 - triangle(lin), 5 - quadrilateral(lin),  
// 8 - prism(lin), 10 - hexahedrom(lin), 12 - tetrahedrom(lin), 14 - pyramid(lin). 0 - point  
// 40 - All elements (1, 3, 5, 8, 10,12, 14) --> 20 - All elements no 3D (1, 3, 5) --> 30 - All elements 3D (8,  
10, 12, 14)  
@3
```

En el ejemplo el @3 informa que debe utilizarse elementos finitos triangulares, lo que es coherente con el archivo de datos Triangles.dump . Vide la Figura 1.

Observar que en el caso de crear mallas simples, pueden realizarse simulaciones para diferentes tipos de elementos finitos. En el caso de escoger @40 serán realizadas todas las simulaciones para los elementos lineales, triangulares, cuadriláteros, hexaedros, tetraedros, prismas y pirámides. Para @30 utilizará todos los elementos tri-dimensionales (hexaedro, tetraedro, prisma y pirámide). Para @20 utilizará los elementos lineales (1D), triangulares y cuadriláteros (2D).

Observación: la expresión (lin) representa que son los elementos finitos lineales, es decir que los segmentos frontera, por ejemplo, del triángulo son segmentos lineales y no cuadráticos (curvos). Por otro lado, fueron dejados valores para los casos de considerar segmentos lineales cuadráticos, triángulos cuadráticos, cuadriláteros cuadráticos y bicuadráticos y así todos los casos cuadráticos para los elementos tridimensionales también. Caso en un futuro se precise de esos elementos finitos, que al momento de la solicitud no fueron considerados ni especificados.

3. Si el usuario desea verificar la malla geométrica sobre la cual se va a construir el espacio de aproximación adecuado, existe el campo: PRINTING GEOMETRIC MESH. En el archivo ejemplo se tiene:

PRINTING GEOMETRIC MESH: /0 no print, /1 print @1

Lo que significa que el usuario solicita imprimir la malla geométrica. Esta será exportada en un archivo del tipo .vtk (paraview).

4. El campo REFINEMENTS es utilizado para el usuario requisitar el nivel de refinamiento que debe ser realizado sobre la malla geométrica dada. Cada nivel es un refinamiento uniforme del anterior. En el archivo ejemplo tenemos:

REFINEMENTS: Number of refinements to apply into the original geometrical mesh: nref @1

Como ejemplo, presentamos la malla desde el archivo Triangles.dump después de un refinamiento uniforme:

5. El campo MODEL PROBLEM detendrá la selección del modelo por el usuario. En el archivo ejemplo fueron implementados los tres modelos solicitados: El primero, @0, que tiene como solución una función producto de senos. El segundo, @1, considera un modelo cuya solución tiene un fuerte gradiente sobre un círculo concéntrico en el origen de radio no nulo. El tercero modelo, @2, considera una solución fuertemente oscilatoria. Caso el usuario quiera simular todos los modelos, puede escoger la opción @3. De manera similar pueden ser

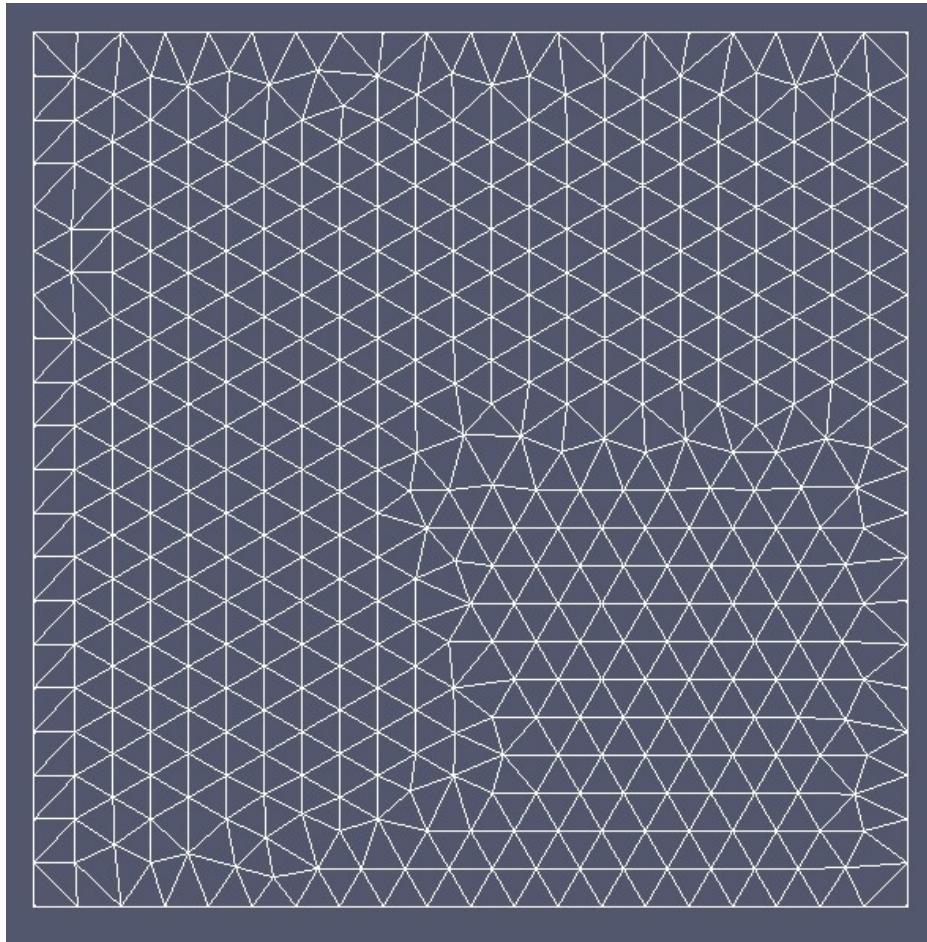


Figura 1: Malla triangular en el archivo Triangles.dump

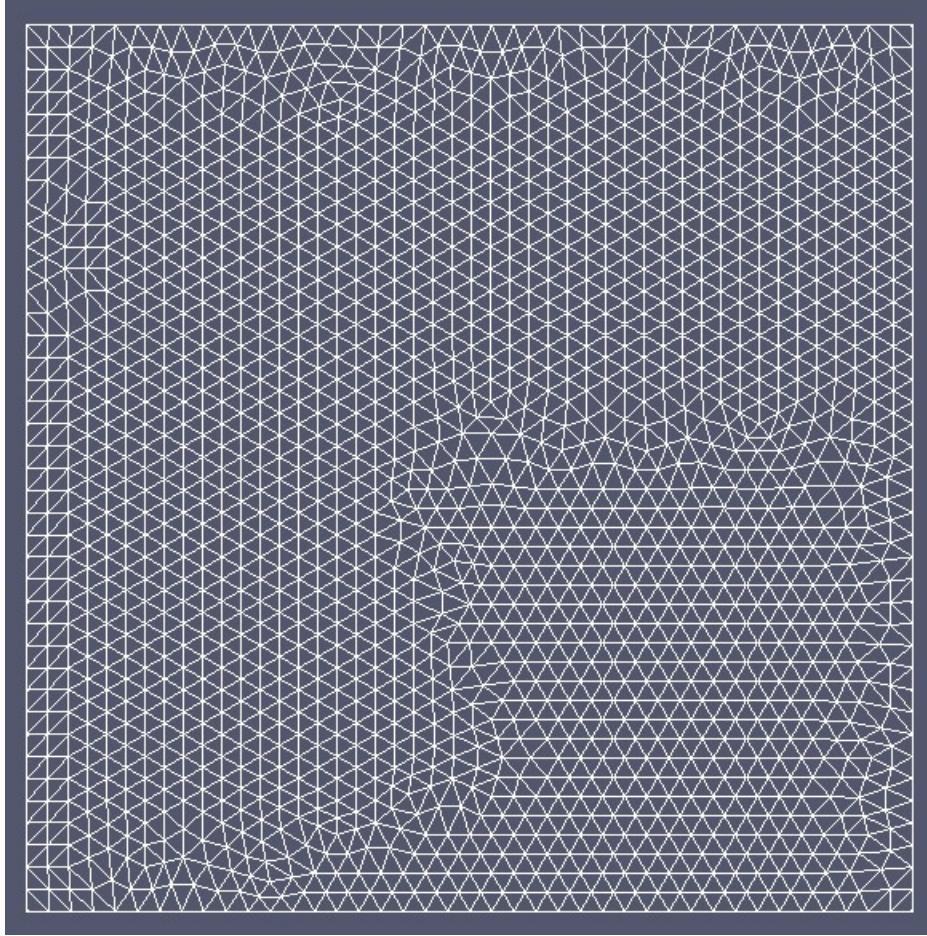


Figura 2: Malla triangular desde el archivo Triangles.dump con mas un nivel de refinamiento uniforme.

incrementados otros modelos (escribiendo la función fuente del modelo de Poisson y si posible la solución exacta esperada - esto no es necesario, porque muchas veces no conocemos la solución exacta). En el archivo ejemplo fue escogida la opción de simular los tres modelos:

MODEL PROBLEM OR EQUATION

```
/0 u=SIN(PI*x)SIN(PI*y)SIN(PI*z)
/1 u=5*(Pi/2 + ArcTg2*(1 - 2*r*r)) Function with strong gradient
/2 SIN*COS*ARCTG Function is oscillatory (strong)
/3 (All models)
@3
```

6. El campo APPROXIMATION SPACES, permite al usuario determinar el espacio de aproximación a utilizar en la simulación numérica. La opción @0 representa utilizar el espacio $H1$ (o método de elementos finitos clásico), la opción @1 utiliza el espacio de funciones discontinuas GD (o método de Galerkin discontinuo), y la opción @2 utilizará la formulación de elementos finitos mixto, es decir utilizará el espacio de aproximación $Hdiv$ (implementado en la tesis [6], conforme solicitud de desarrollo, y para la solución de la variable primal utilizará el espacio $H1$. La opción @3 permitirá seleccionar simulaciones que utilizarán todos los espacios aproximantes, un a un. En el archivo ejemplo utiliza la opción de simular con todos los espacios aproximantes:

APPROXIMATION SPACES:

```
/0 (H1)
/1 (GD)
/2 (HdivH1)
/3 (All)
@3
```

7. El campo P_ORDER permite escoger el orden de interpolación de los polinomios aproximantes en cada uno de los espacios aproximantes de elementos finitos. En el archivo ejemplo se ha solicitado utilizar polinomios cuadráticos:

P_ORDER: Order to piecewise polynomials: pOrder @2

8. En el campo LINEAR SYSTEM SOLVER se puede escoger resolver el sistema de ecuaciones linear (matricial) por métodos directos, o iterativos. En el archivo ejemplo se presenta las opciones, método directo por descomposición LU (@0), método iterativo utilizando el método GMRES para el caso de matrices no simétricas (@1), y el método iterativo utilizando el método del gradiente conjugado para el caso de matrices simétricas (@2). En el ejemplo se ha escogido la opción @0.

LINEAR SYSTEM SOLVER

```
/0 Direct - LU
/1 Iterative - Using GMRES (no symmetric)
/2 Iterative - Using CG (symmetric case)
Solver to System equation @0
```

9. En el campo TO POST PROCESS el usuario puede informar la cantidad de variables escalares de la solución aproximada que desea que sean exportadas en los archivos .vtk y los nombres de esas variables. Posteriormente solicita el número de variables vectoriales y sus respectivos nombres, también para ser exportados en el mismo archivo .vtk. En el archivo ejemplo, se solicita dos variables escalares que serán la presión y el valor exacto de la presión (esto porque conocemos la solución exacta esperada). También se solicita exportar los valores de dos variables vectoriales, el flujo (calculado - aproximado) y el flujo exacto conocido de la solución.

TO POST PROCESS - Output of solutions

Number of scalar variables to output solutions, and the names of the scalar variables: @2 Pressure ExactPressure

Number of vector variables to output solutions, and the names of the vector variables: @2 Flux ExactFlux

En el campo TO POST PROCESS, los nombres de las variables pueden ser encontradas en la documentación del NeoPZ para la clase TPZMatPoisson3d y de la clase TPZMixedPoisson.

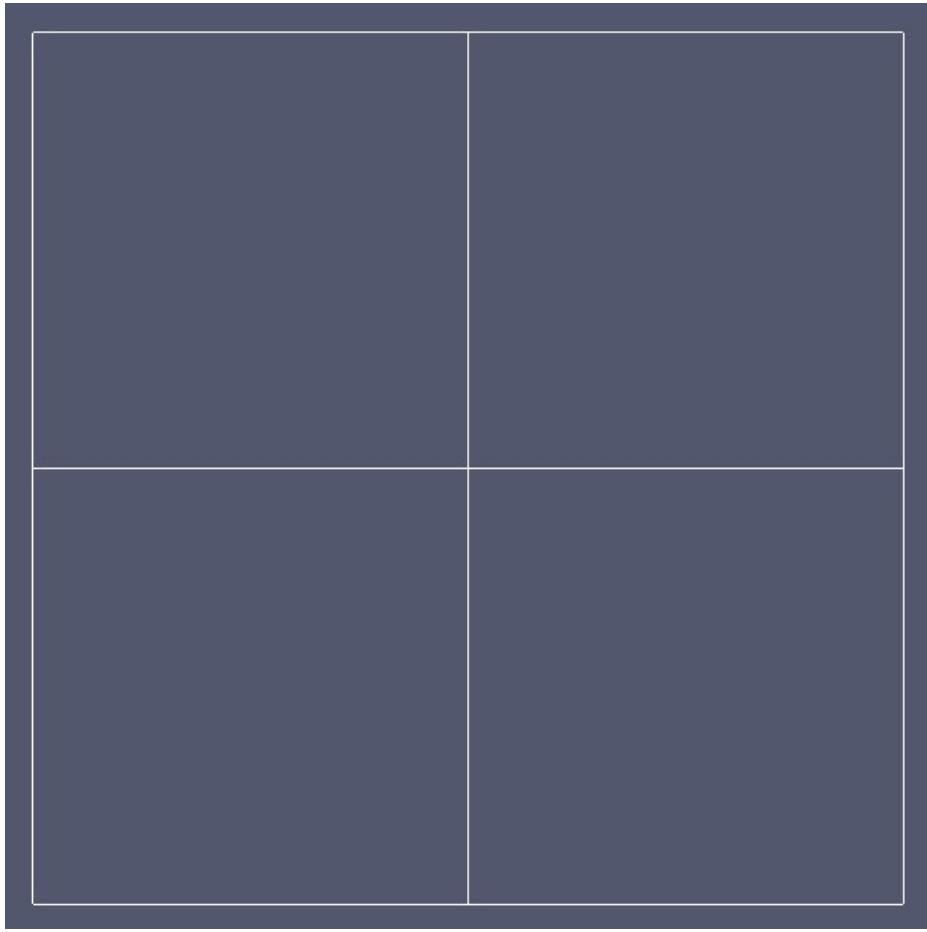


Figura 3: Domínio quadrado de lado 2.

3. Guia de uso do Simulator Results

Después de ejecutar el software LCC_Simulator, los resultados estarán almacenados en /LCC_SimulatorIO/SimulationResults.

El usuario puede encontrar los resultados obtenidos de las simulaciones por data y hora de la ejecución.

Para poder visualizar los datos de las soluciones aproximadas, basta tener instalado el software libre Paraview (www.paraview.org).

Para cada ejecución es creado un archivo con las informaciones de la simulación: tamaño de los sistemas lineales montados, tiempos de ejecución e inclusive los errores cometidos con la simulación (solo cuando la solución exacta es conocida). El archivo es el «InfoRun_Erros.txt». Visualizelo y obtenga muchas informaciones para verificar si la simulación genera tantas ecuaciones que su sistema no soporta.

4. Modelo Matemático

Sea el domínio bidimensional $\Omega = [-1, 1] \times [-1, 1]$.

Considerar el modelo del problema de Poisson definido en una región poligonal $\Omega \subset \mathbb{R}^2$, vea la Figura [3], y escrito en la forma

$$\nabla \cdot (\mathbf{K} \nabla u) = f, \text{ en } \Omega \quad (1)$$

$$u = 0, \text{ sobre } \partial\Omega, \quad (2)$$

donde \mathbf{K} es un tensor definido positivo, simétrico y limitado, y $f \in L^2(\Omega)$.

A partir de las ecuaciones 1 y 2 se realiza la formulación débil del sistema de ecuaciones diferenciales parciales aplicando dos espacios de aproximación diferentes, $H1$ para aplicar el método de elementos finitos clásico [1, 8, 2]. Una segunda alternativa es utilizar una formulación débil similar que utiliza un espacio de funciones discontinuas llamado de método de Galerkin discontinuo [5, 4, 3]. El software implementa los dos métodos ($H1$ y GD).

Este problema puede ser expresado como

$$\begin{aligned} \nabla \cdot \boldsymbol{\sigma} &= f && \text{en } \Omega, \\ \boldsymbol{\sigma} &= -\mathbf{K} \nabla u && \text{en } \Omega, \\ u &= 0 && \text{sobre } \partial\Omega, \end{aligned}$$

La formulación débil para el problema (1)-(2) es: hallar $u \in H_0^1(\Omega)$ tal que

$$(\mathbf{K} \nabla u, \nabla v)_\Omega = (f, v), \forall v \in H_0^1(\Omega). \quad (3)$$

A partir de la formulación 3, conocida como formulación mixta de elementos finitos, se resuelve con el método mixto de elementos finitos [10, 11, 9, 7]. El software implementa el método de elementos finitos ($HdivH1$).

Con las implementaciones de estos tres métodos en el software LCC_Simulator, fueron realizados experimentos numéricos para el modelo de Poisson indicado. Los resultados son expuestos en la siguiente sección.

5. Simulaciones de los tres modelos

5.1. Ejemplo Seno-Seno

Considerar un problema de Poisson definido sobre el dominio $\Omega = (0, 1)^2$ con función de carga f , tensor $\mathbf{K} = \mathbb{I}$, y solución exacta $u(x, y) = \sin(\pi x) \sin(\pi y)$. En la literatura este particular ejemplo, ha presentado comportamiento de superconvergencia para la variable flujo.

5.1.1. Espacio de funciones continuas $H1$

Realizando la simulación con el método de elementos finitos clásico ($H1$) presentamos la visualización de la solución numérica aproximada para la variable **presión**:

Observar los valores de los flujo (en módulo) para este espacio de aproximación:

5.1.2. Espacio de funciones discontinuas GD

Realizando la simulación con el método de funciones discontinuas GD presentamos la visualización de la solución numérica aproximada para la variable **presión**

Comentario: en esta figura se puede observar claramente que fueron usadas funciones discontinuas, porque los bordes están desconectados de los elementos.

Observemos la simulación numérica aproximada para la variable flujo GD

5.1.3. Espacios de funciones $HdivH1$

Realizando la simulación con el método $HdivH1$ presentamos la visualización de la solución numérica aproximada para la variable **presión**:

Observemos la simulación numérica aproximada para la variable flujo $HdivH1$

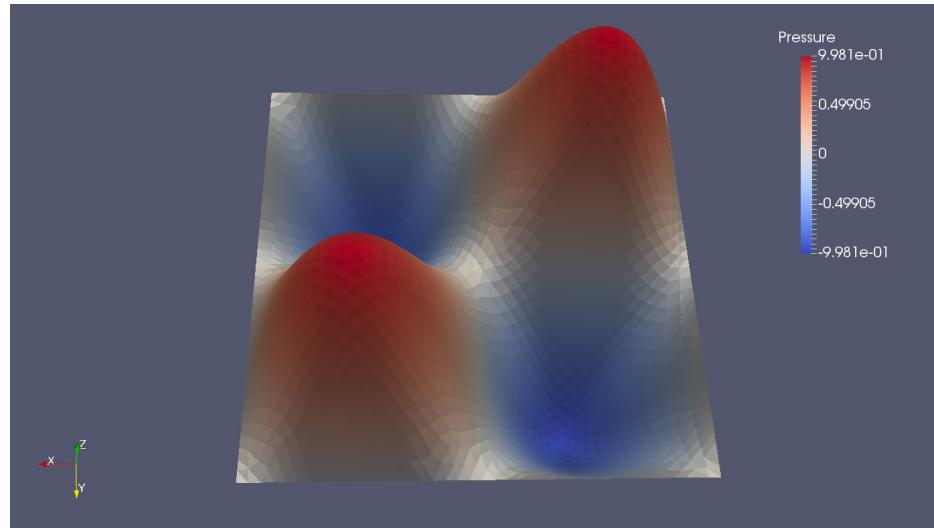


Figura 4: Presión en funciones continuas

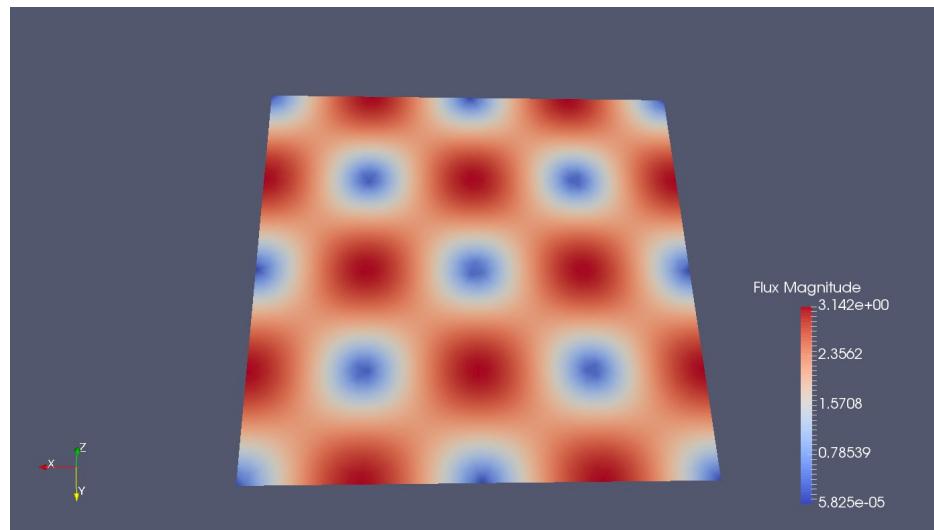


Figura 5: Módulo del flujo al utilizar el espacio $H1$

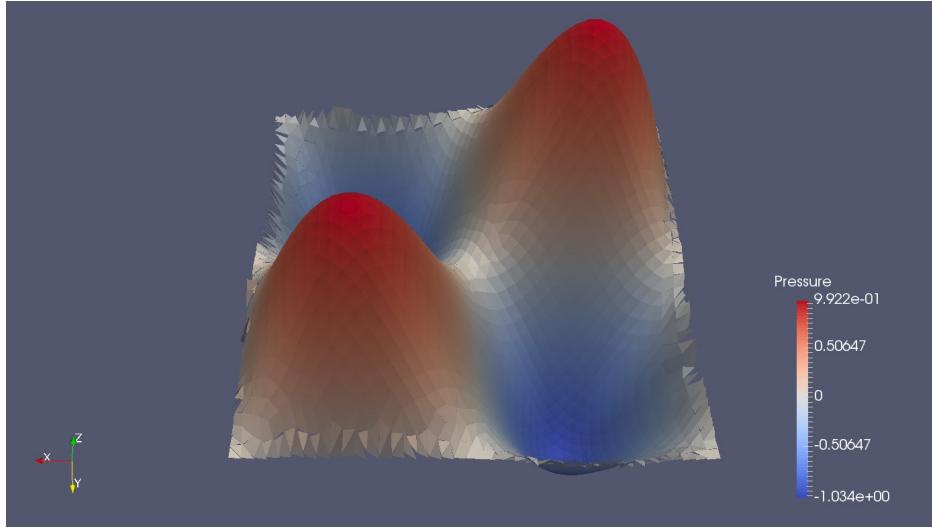


Figura 6: Presión en funciones discontinuas

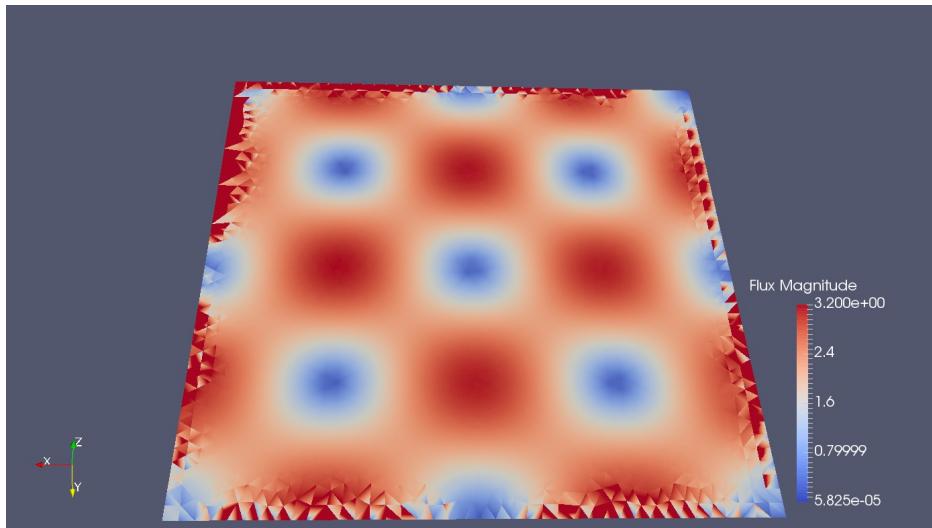


Figura 7: Modo de flujo en el espacio GD

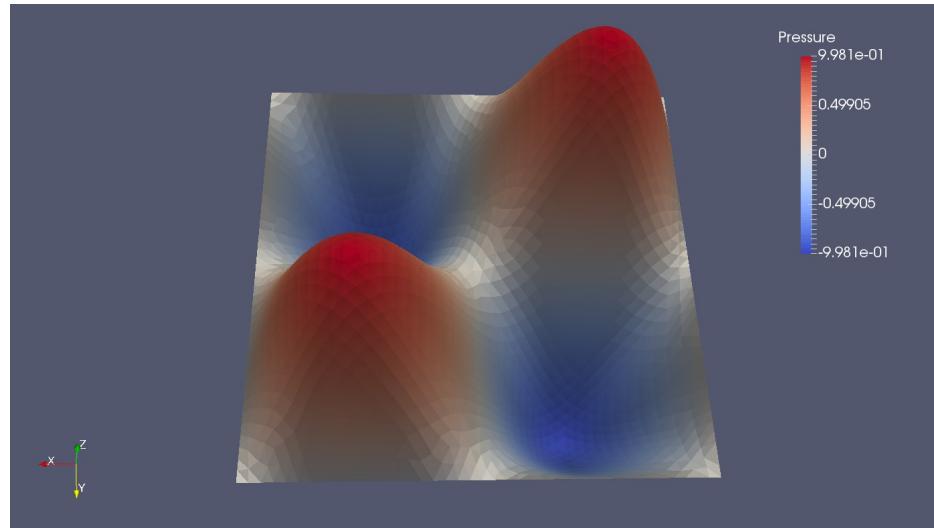


Figura 8: Presión en funciones HdivH1

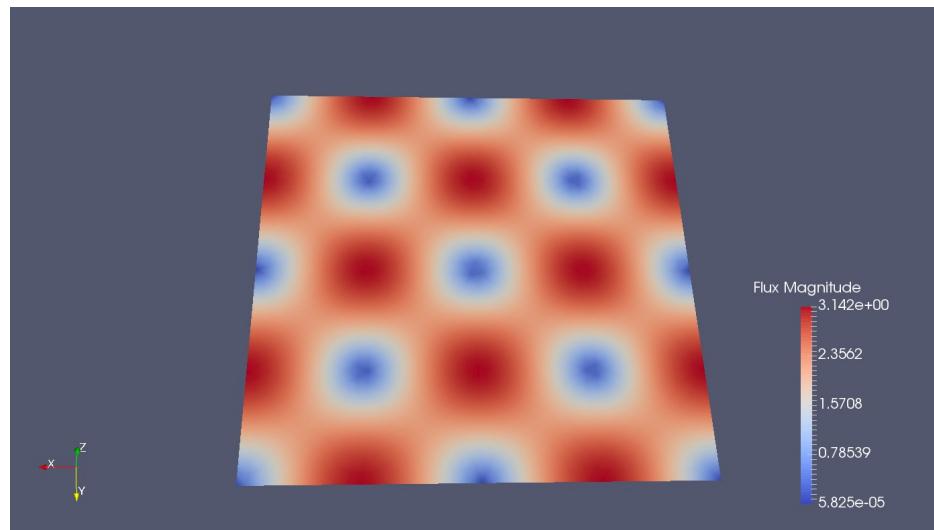


Figura 9: Modo de flujo en el espacio HdivH1

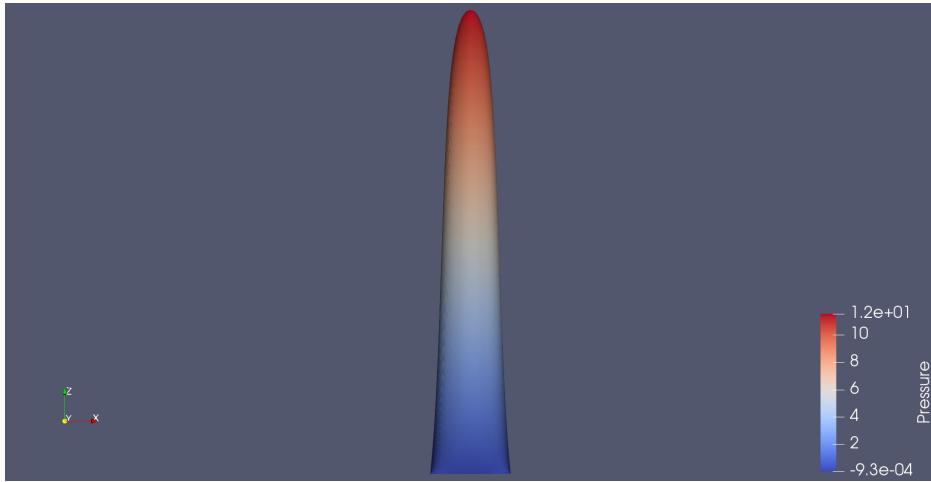


Figura 10: Presión en funciones continuas

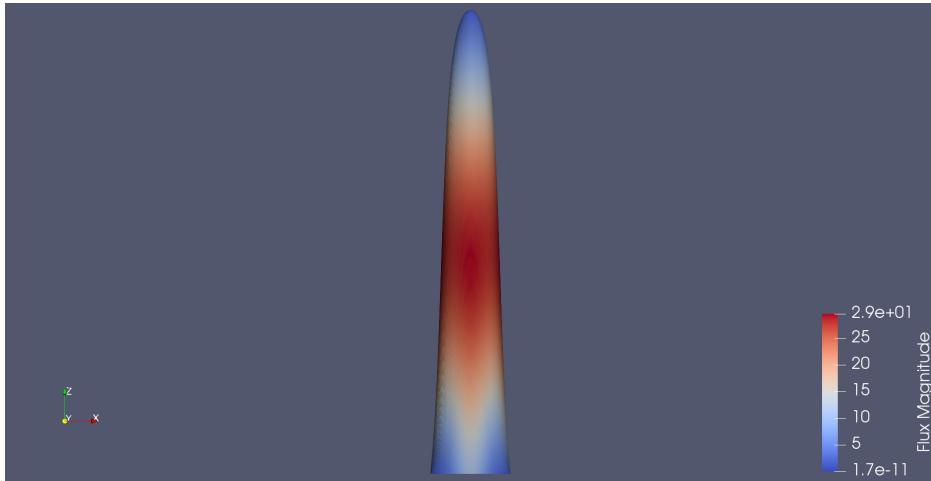


Figura 11: Módulo del flujo al utilizar el espacio $H1$

5.2. Ejemplo de Alto Gradiente

Considerar un problema de Poisson definido sobre el dominio $\Omega = (0, 1)^2$ con función de carga f , tensor $\mathbf{K} = \mathbb{I}$, y solución exacta $u(x, y) = 5 * \left(\frac{\pi}{2} + \text{ArcTg} \left(2 \left(1 - 2\sqrt{x^2 + y^2} \right) \right) \right)$. En la literatura este particular ejemplo, presenta un gradiente muy alto.

5.2.1. Espacio de funciones continuas $H1$

Realizando la simulación con el método de elementos finitos clásico ($H1$) presentamos la visualización de la solución numérica aproximada para la variable **presión**:

Observar los valores de los flujo (en módulo) para este espacio de aproximación:

5.2.2. Espacio de funciones discontinuas GD

Realizando la simulación con el método de funciones discontinuas GD presentamos la visualización de la solución numérica aproximada para la variable **presión**

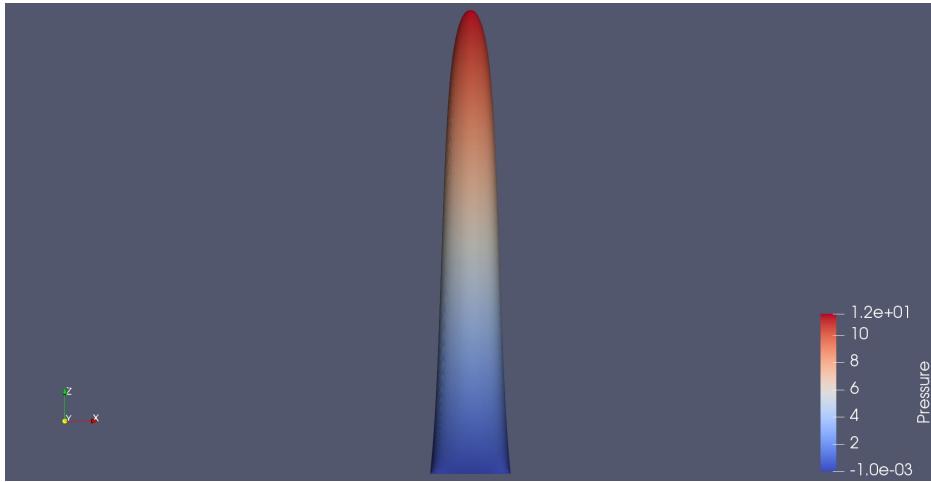


Figura 12: Presión en funciones discontinuas

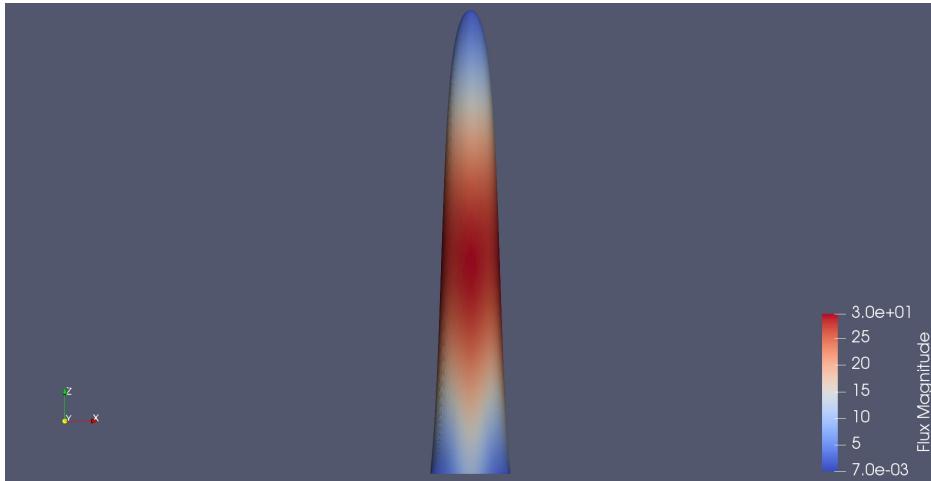


Figura 13: Modo de flujo en el espacio GD

Comentario: en esta figura se puede observar claramente que fueron usadas funciones discontinuas, porque los bordes están desconectados de los elementos.

Observemos la simulación numérica aproximada para la variable flujo GD

5.2.3. Espacios de funciones HdivH1

Realizando la simulación con el método HdivH1 presentamos la visualización de la solución numérica aproximada para la variable **presión**:

Observemos la simulación numérica aproximada para la variable flujo HdivH1

5.3. Ejemplo de Oscilación

Considerar un problema de Poisson definido sobre el dominio $\Omega = (0, 1)^2$ con función de carga f , tensor $\mathbf{K} = \mathbb{I}$, y solución exacta $u(x, y) = \text{Sin}(5\pi x)(1 + \text{Cos}(5\pi y)) \left(\frac{\pi}{2} + \text{ArcTg} \left(2 \left(1 - 2\sqrt{x^2 + y^2} \right) \right) \right)$. En la literatura este particular ejemplo, presenta un comportamiento muy oscilatorio.

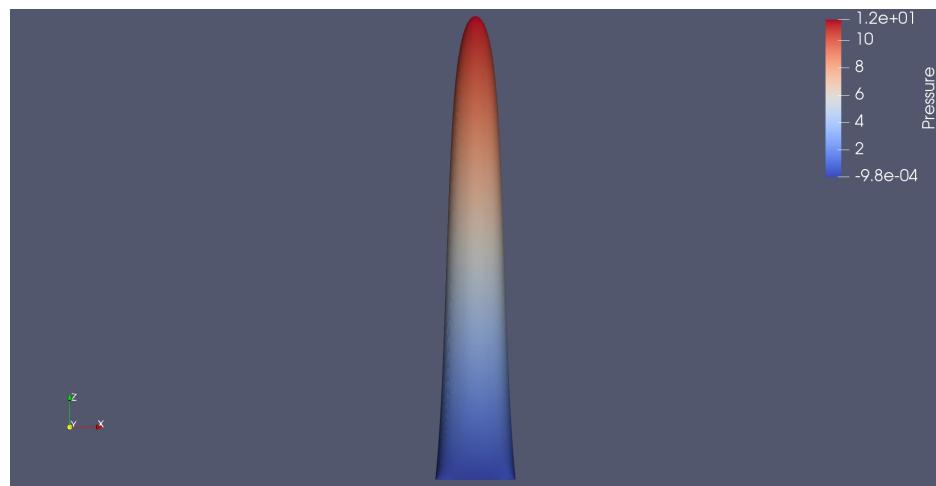


Figura 14: Presión en funciones HdivH1

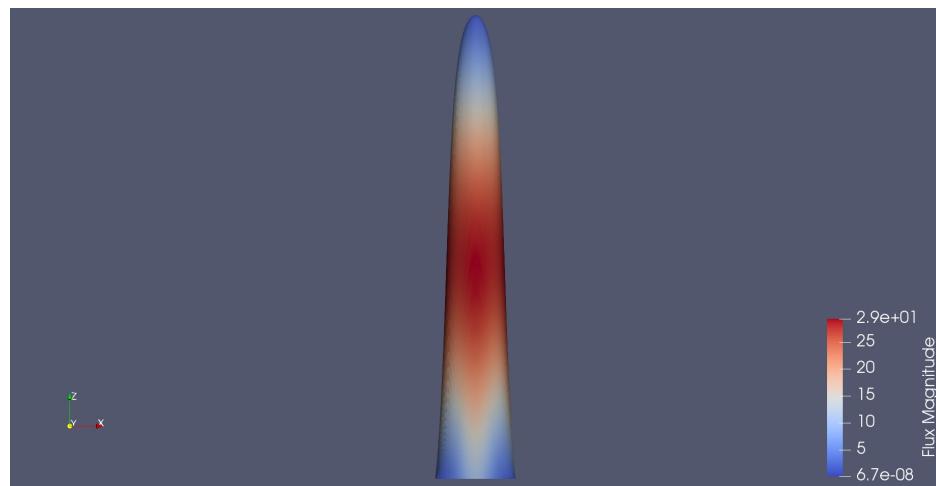


Figura 15: Modo de flujo en el espacio HdivH1

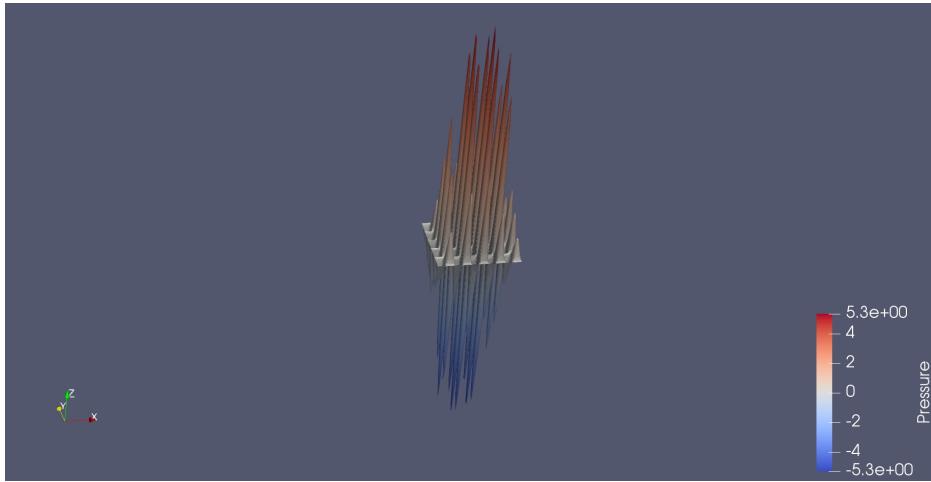


Figura 16: Presión en funciones continuas

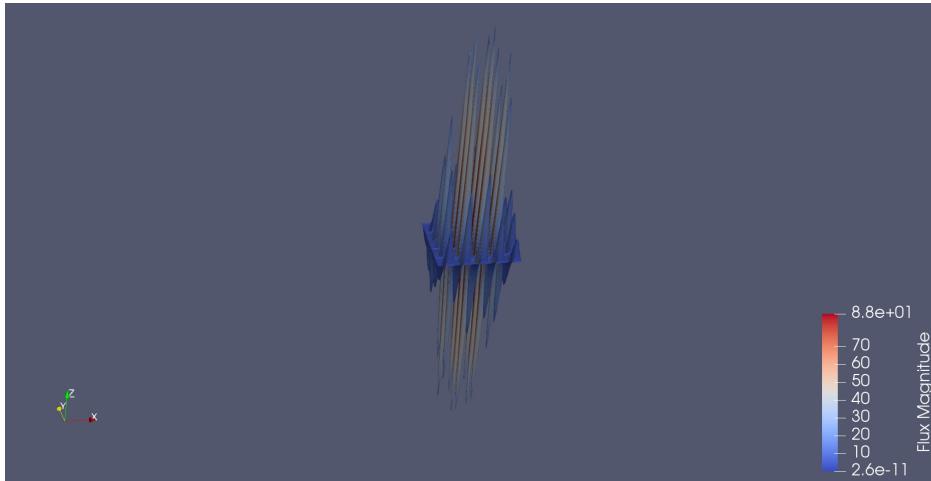


Figura 17: Módulo del flujo al utilizar el espacio $H1$

5.3.1. Espacio de funciones continuas $H1$

Realizando la simulación con el método de elementos finitos clásico ($H1$) presentamos la visualización de la solución numérica aproximada para la variable **presión**:

Observar los valores de los flujo (en módulo) para este espacio de aproximación:

5.3.2. Espacio de funciones discontinuas GD

Realizando la simulación con el método de funciones discontinuas GD presentamos la visualización de la solución numérica aproximada para la variable **presión**

Comentario: en esta figura se puede observar claramente que fueron usadas funciones discontinuas, porque los bordes están desconectados de los elementos.

Observemos la simulación numérica aproximada para la variable flujo GD

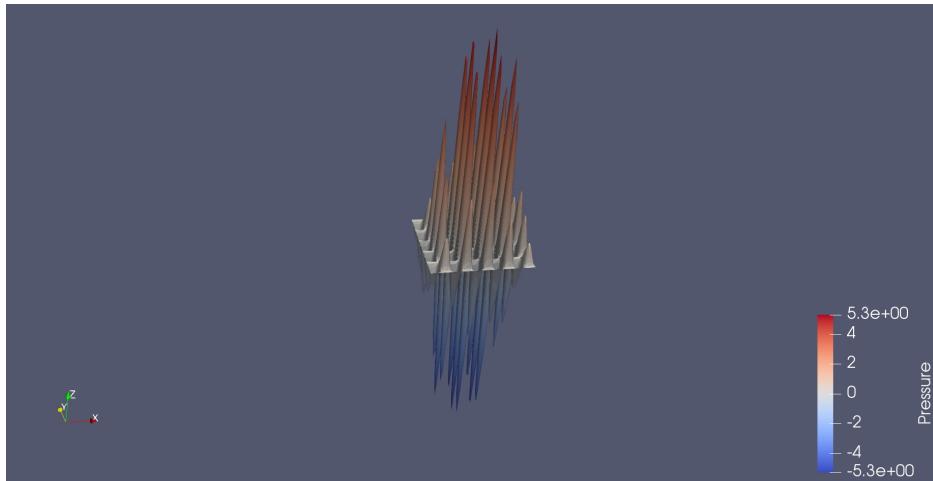


Figura 18: Presión en funciones discontinuas

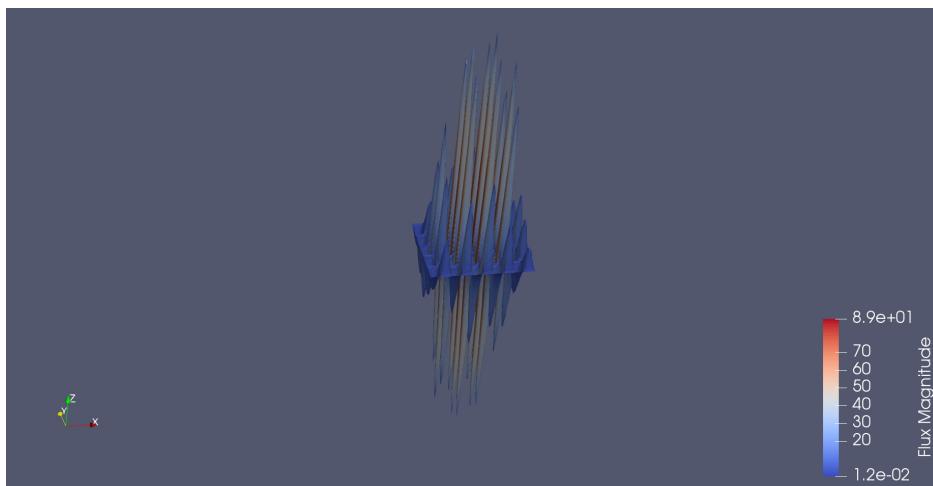


Figura 19: Modo de flujo en el espacio GD

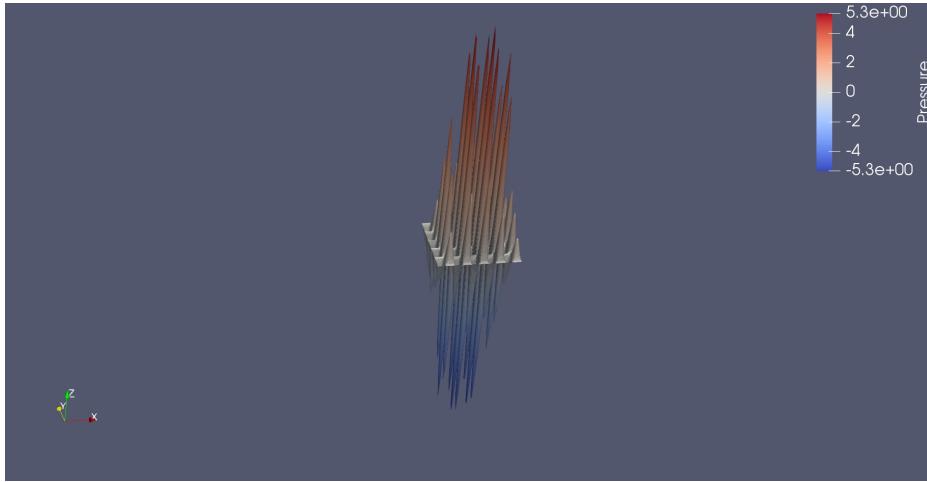


Figura 20: Presión en funciones HdivH1

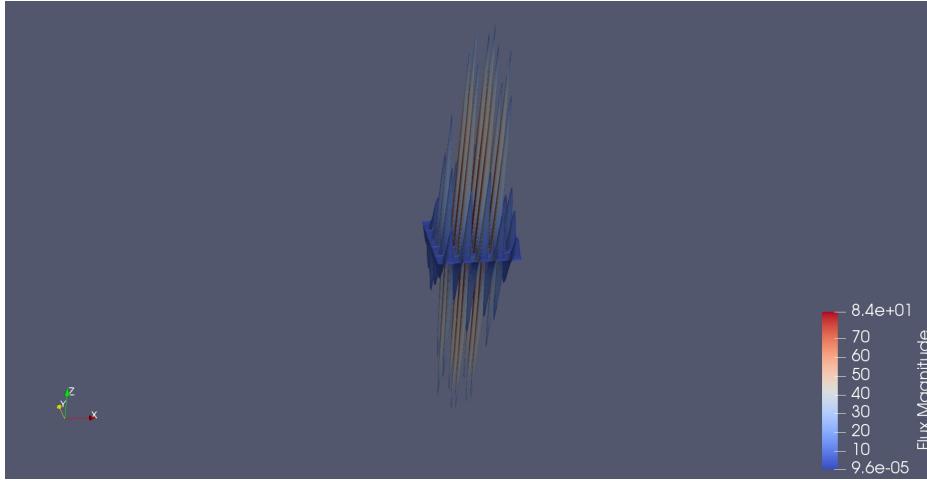


Figura 21: Modo de flujo en el espacio HdivH1

5.3.3. Espacios de funciones HdivH1

Realizando la simulación con el método HdivH1 presentamos la visualización de la solución numérica aproximada para la variable **presión**:

Observemos la simulación numérica aproximada para la variable flujo HdivH1

6. Modelos en una y tres dimensiones

El software por naturaleza realiza simulaciones en una y tres dimensiones también. Colocamos algunos ejemplos a seguir.

6.1. Ejemplo Seno-Seno

Considerar un problema de Poisson definido sobre el dominio $\Omega = (0, 1)^3$ con función de carga f , tensor $\mathbf{K} = \mathbb{I}$, y solución exacta $u(x, y, z) = \sin(\pi x) \sin(\pi y) \sin(\pi z)$. En la literatura este particular ejemplo, ha presentado

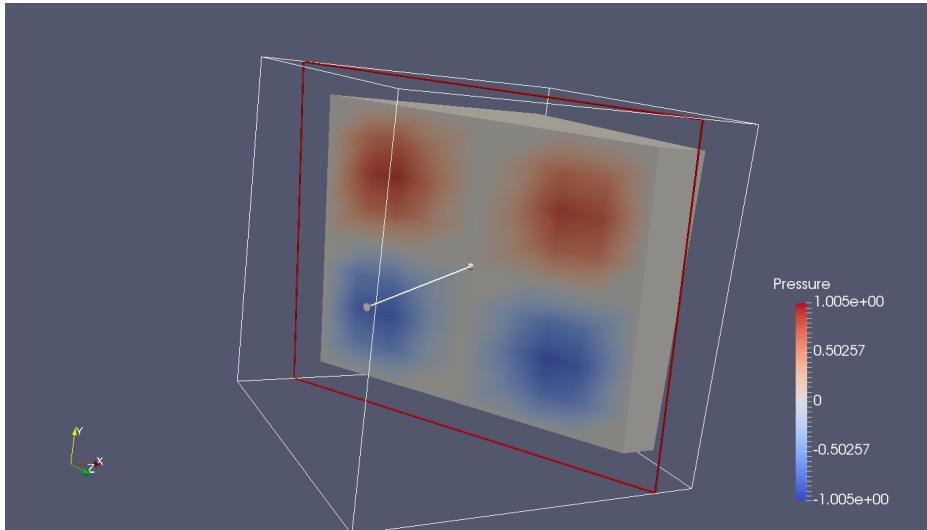


Figura 22: Presión en funciones continuas

comportamiento de superconvergencia para la variable flujo.

6.2. Ejemplo de Alto Gradiente

Considerar un problema de Poisson definido sobre el dominio $\Omega = (0, 1)^3$ con función de carga f , tensor $\mathbf{K} = \mathbb{I}$, y solución exacta $u(x, y, z) = 5 * \left(\frac{\pi}{2} + \text{ArcTg} \left(2 \left(1 - 2\sqrt{x^2 + y^2 + z^2} \right) \right) \right)$. En la literatura este particular ejemplo, presenta un gradiente muy alto.

6.3. Ejemplo de Oscilación

Considerar un problema de Poisson definido sobre el dominio $\Omega = (0, 1)^3$ con función de carga f , tensor $\mathbf{K} = \mathbb{I}$, y solución exacta $u(x, y, z) = \text{Sin}(5\pi x)(1 + \text{Cos}(5\pi y))(1 + \text{Cos}(5\pi z)) \left(\frac{\pi}{2} + \text{ArcTg} \left(2 \left(1 - 2\sqrt{x^2 + y^2 + z^2} \right) \right) \right)$. En la literatura este particular ejemplo, presenta un comportamiento muy oscilatorio.

6.3.1. Resultados informativos

Tres imágenes de las simulaciones para cada uno de los modelos:

6.4. Mallas en elementos tridimensionales

6.5. Algo en 1D

7. Glosario de términos

1. H1 Espacio de funciones continuas, polinómicas por partes, del método de Elementos Finitos clásicos.
2. GD Espacio de funciones discontinuas, polinómicas por partes definidas de forma independiente sobre cada elemento finito, del método de Galerkin Discontinuo.
3. HdivH1 utiliza dos espacios de aproximación, H1 y un espacio de funciones con derivadas normales sobre las fronteras de cada elemento finito, Hdiv, y son funciones vectoriales.

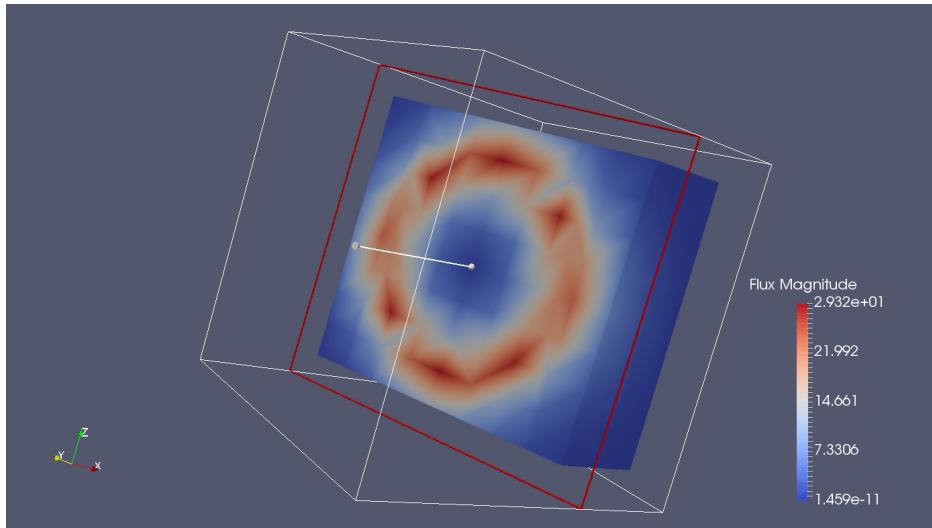


Figura 23: Flujo para el modelo de fuerte gradiente

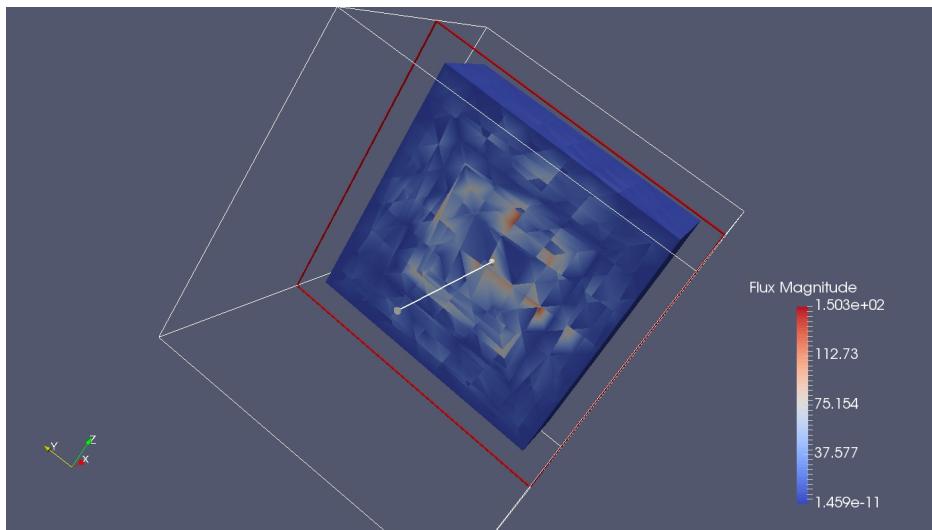


Figura 24: Flujo en HdivH1

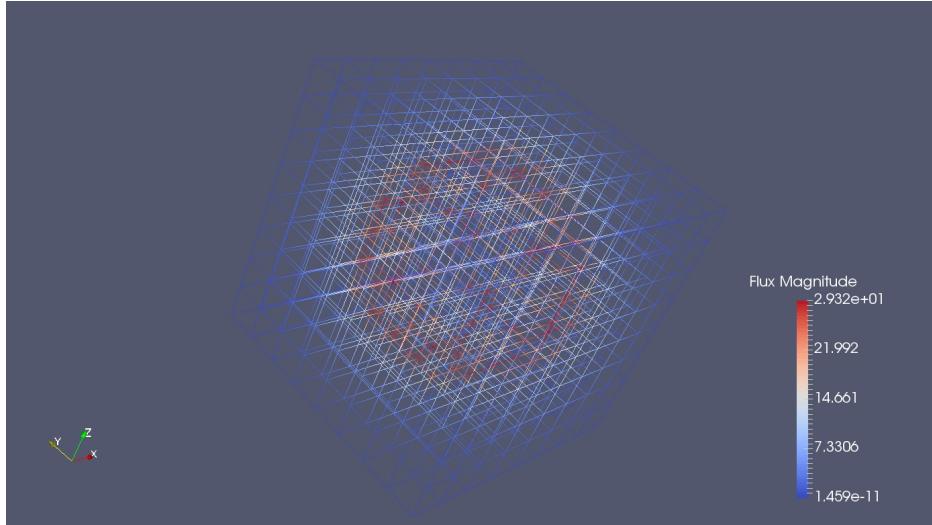


Figura 25: Malla en cubos (hexaedros)

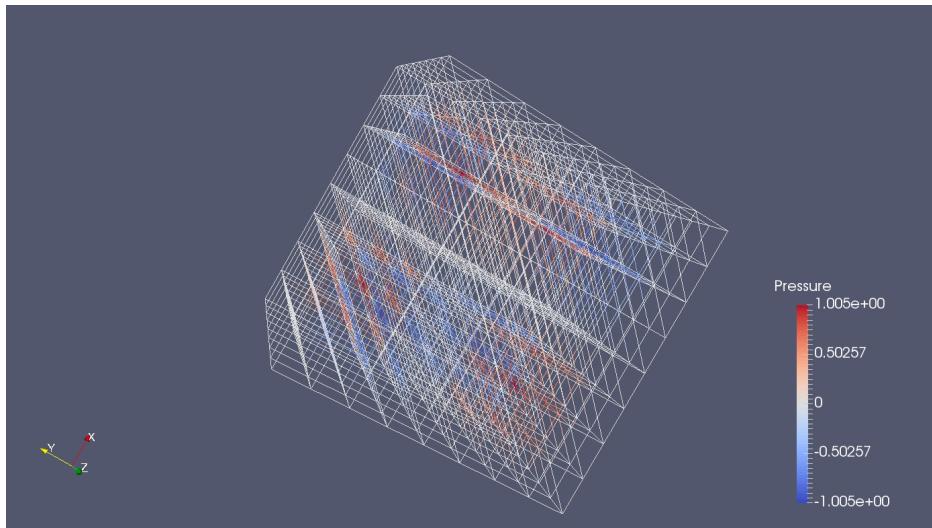


Figura 26: Malla en prismas

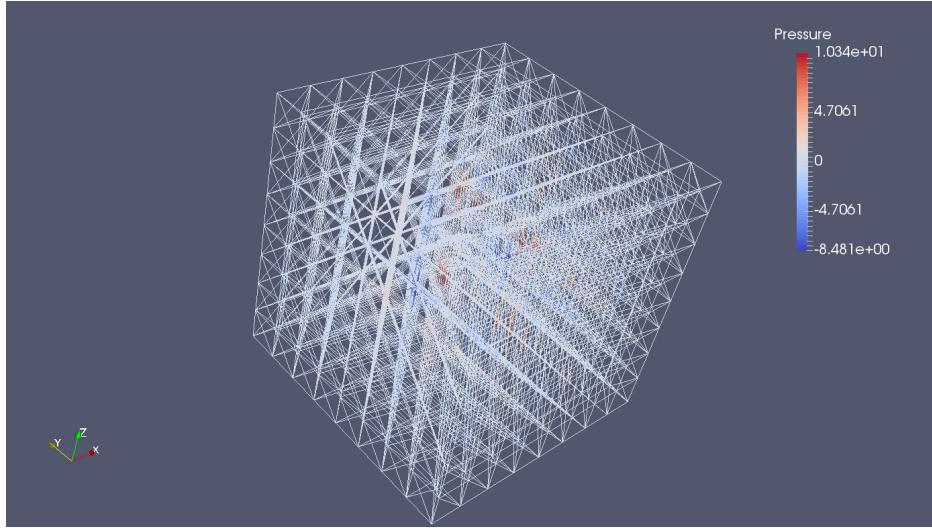


Figura 27: Malla en tetraedros

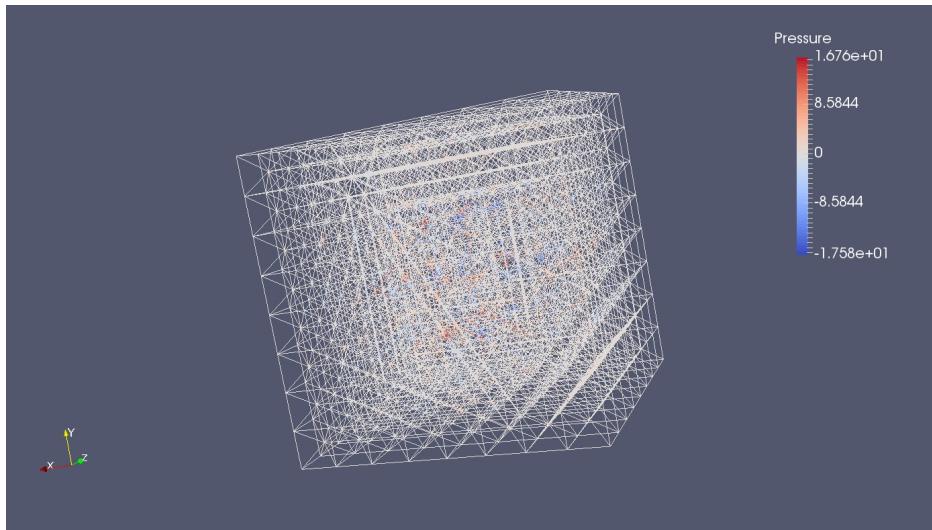


Figura 28: Malla en piramides

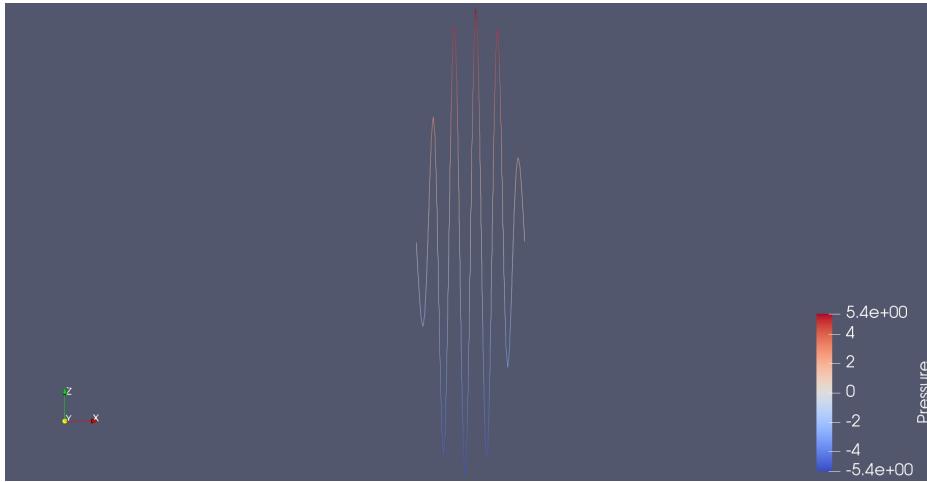


Figura 29: Modelo oscilatorio en 1D

4. NeoPZ Ambiente de Computación Científica del laboratorio de mecánica computacional de la Unicamp (LAB-MEC: www.labmec.org.br).

Referencias

- [1] Ivo Babuska, John Whiteman, and Theofanis Strouboulis. *Finite elements, An introduction to the method and error estimation.* Oxford, 2011.
- [2] Eric B. Becker, Graham F. Carey, and J. Tinsley Oden. *Finite Elements, An Introduction. Volume I.* Prentice-Hall, 1981.
- [3] B. Cockburn. Discontinuous galerkin method for convection-dominated problems. in high-order methods for computational physics. *Lectures Notes in Computational Science and Engineering*, 9:69–224, 1999.
- [4] B. Cockburn. Discontinuous galerkin methods. Technical report, School of Mathematics - University of Minnesota, 2003.
- [5] B. Cockburn, G. Karniadakis, and C. Shu. *Discontinuous Galerkin Methods. Theory, computation and applications.* Number 11 in Lecture Notes in Computational Science and Engineering. Springer, Berlin, 2000.
- [6] Denise de Siqueira. *Construção de espaços de elementos finitos do tipo HDiv.* PhD thesis, Unicamp, 2012.
- [7] Philippe R. B. Devloo, Omar Duran, Agnaldo M. Farias, and Sonia M. Gomes Paulo C. A. Lucci. Advanced finite element techniques for multiphysics and multiscale simulations. Technical report, IACM Expressions, 2018.
- [8] Ronald H. W. Hoppe. *Finite Element Methods*, chapter 1 Foundations of elliptic boundary value problems, pages 1 – 11. www.math.uh.edu, 2016.
- [9] Weifeng Qiu and Leszek Demkowicz. Mixed hp-finite element method for linear elasticity with weakly imposed symmetry. *Computer Methods in Applied Mechanics and Engineering*, 2009.
- [10] Weifeng Qiu and Leszek Demkowicz. Mixed hp-finite element method for linear elasticity with weakly imposed symmetry. ii. curvilinear elements in 2d. Technical report, ICES Report 10-06, 2010.
- [11] Weifeng Qiu and Leszek Demkowicz. Mixed hp-finite element method for linear elasticity with weakly imposed symmetry iii. stability analysis in 3d. Technical report, ICES Report 10-20, 2010.