Contents

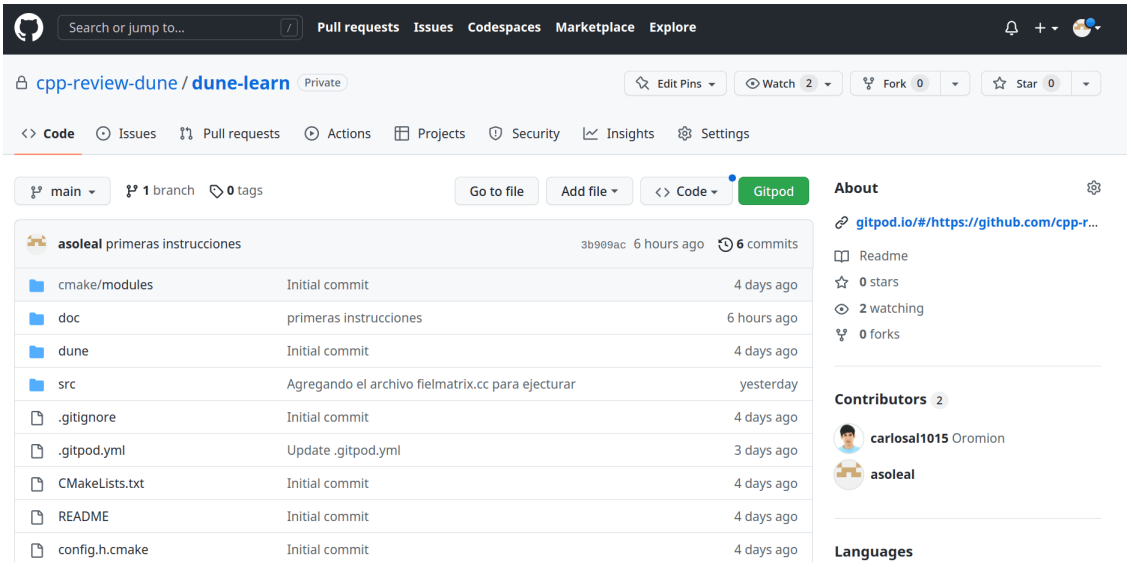
1 Introduccin

Este manual pretende introducir a los usuarios en el manejo del software de simulacin numrica Distributed and Unified Numerics Environment (Dune), que se puede ubicar en la direccin https://dune-project.org, proyecto de cdigo abierto que proporciona una infraestructura flexible y modular para la solucin de ecuaciones diferenciales parciales.

1.1 Instalacin local

1.2 Configuracin en la nube

Para correr el software Dune, vamos a utilizar el repositorio en GitHub https://github.com/cpp-review-dune/dune-learn, diseado para funcionar en la nube, utilizando las ventajas que nos ofrece https://www.gitpod.io, en l se utiliza una imagen docker prediseada basada en la distribucin GNU/Linux Arch Linux, que contiene las dependencias y repositorios necesarios para su funcionamiento. [1] Ver la figura 1.2:



[1] La extensin de gitpod puede ser instalada dependiendo del navegador que se utilice

1.3 Instalacin

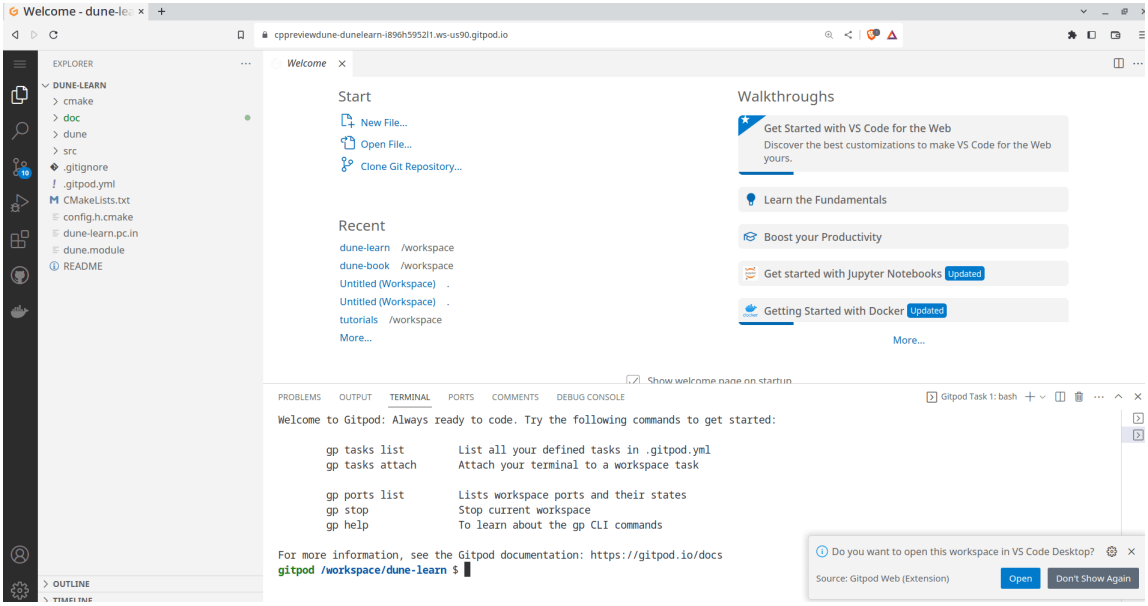
Use el repositorio Arch Linux for Education. Para un mayor detalle vea.

En la seccin de C++, se presentar dune-pdelab, que fue estudiada en el curso presentado en el ao 2022, de manera remota y cuya link est en: https://dune-pdelab-course.readthedocs.io/en/latest/intro.html, y en la seccin de Python, utilizaremos JupyterLab para introducir el uso de dune-fem.

2 Dune en C++

2.1 Primer proyecto en Dune

Para iniciar la ejecucin, una vez hemos ingresado a la plataforma y estamos en la nube, debemos ver una imagen similar a la siguiente:



Como se puede apreciar, es muy similar al editor de cdigo basado en navegador, Monaco. Para hacer mi primer proyecto, en la terminal debo escribir:
gitpod/workspace$ duneproject

== Dune project/module generator ==

duneproject will assist you in the creation of a new Dune application.
During this process a new directory with the name of your project will be
created. This directory will hold all configuration and Makefiles and a
simple example application.

1) Name of your new Project? (e.g.: dune-grid):

En sta parte, debo completar la informacin de mi proyecto, el nombre vamos a poner por ejemplo dune-prueba, luego

2) Which modules should this module depend on?
The following modules have been found:
dune-common dune-geometry dune-uggrid dune-grid dune-localfunctions dune-istl dune-typetree dune-functions dune-alugrid dune-pdelab dune-fem dune-learn
Enter space-separated list:

En ste caso, vamos a usar dune-pdelab, luego pregunta:

3) Project/Module version?

Por ejemplo puede ser 01, luego nos pide el email:

4) Maintainer's email address? maintainer@unal.edu.co

Finalmente, nos pregunta si los datos estn correctos:

creating Project "dune-prueba", version 01
which depends on "dune-pdelab"
with maintainer "maintainer@unal.edu.co"
Is this information correct? [y/N] y

Si los datos son correctos, asignamos "y", de lo contrario "N", y finalmente enter para iniciar la configuracin. Una vez iniciada la configuracin, se crea una carpeta con el nombre dune-prueba, que fue el nombre del proyecto que asignamos. Una vez que el proyecto se ha construido, podemos utilizar el comando

gitpod/workspace $ ls

Dentro del listado debe aparecer la carpeta dune-prueba, que si explora dentro de ella, encontrar varios archivos y carpetas. Se puede dirigir a la siguiente direccin:

cd /workspace/dune-prueba/src $

Cuando liste, encontrar dos archivos

CMakeLists.txt dune-prueba.cc

En el primer archivo, usted encuentra lo siguiente:

add˙executable("dune-prueba" dune-prueba.cc)
target˙link˙dune˙default˙libraries("dune-prueba")

Significa que se ha creado un cdigo fuente, que se llama "dune-prueba.cc", tiene el mismo nombre del proyecto que creamos, y en el que est escrito nuestro primer programa, el "Hola mundo de DUNE". A continuacin se puede apreciar el contenido completo del primer programa:

```
#ifdef HAVE˙CONFIG˙H
#include "config.h"
#endif
#include ¡dune/common/exceptions.hh¿          // We use exceptions
#include ¡dune/common/parallel/mpihelper.hh¿ // An initializer of MPI
#include ¡dune/learn/learn.hh¿
#include ¡iostream¿

int main(int argc, char **argv)
–
  try –
    // Maybe initialize MPI
    Dune::MPIHelper &helper = Dune::MPIHelper::instance(argc, argv);
    std::cout ¡¡ "Hello World! This is dune-learn." ¡¡ std::endl;
    if (Dune::MPIHelper::isFake)
      std::cout ¡¡ "This is a sequential program." ¡¡ std::endl;
    else
      std::cout ¡¡ "I am rank " ¡¡ helper.rank() ¡¡ " of " ¡¡ helper.size()
                ¡¡ " processes!" ¡¡ std::endl;
    return 0;
  ”
  catch (Dune::Exception &e) –
    std::cerr ¡¡ "Dune reported error: " ¡¡ e ¡¡ std::endl;
  ”
  catch (...) –
    std::cerr ¡¡ "Unknown exception thrown!" ¡¡ std::endl;
  ”
”
```

2.1.1 Compilacin

Para compilar un programa en C++ que emplee las cabeceras de DUNE, es necesario construir una carpeta temporal de construccin que llamaremos build, que se utiliza para construir all los aspectos necesarios para la compilacin del proyecto, y aparecer despus de compilar el archivo ejecutable. Una vez creada la carpeta, es necesario utilizar el comando cmake. En ste caso se indica a cmake que utilice como fuente ("S" de source) la carpeta dune-prueba y que la construccin de la compilacin se va a guardar en la carpeta build, por eso se escribe ("-B" build).

gitpod/workspace $ cmake -S dune-prueba -B build

En caso de que se quiera compilar en la misma carpeta se debe utilizar el comando:

gitpod/workspace $ cmake -S . -B build

Luego de hacer el proceso, si hacemos una lista utilizando el comando ls, obtendremos dos carpetas, a saber:

gitpod/workspace $ ls
gitpod/workspace $ build dune-prueba

Es necesario aclarar, que nuestro cdigo estar en la carpeta dune-prueba, mientras que el ejecutable y todos los archivos necesarios para la compilacin estn en la carpeta build. A continuacin, ingresamos a la carpeta build y usamos el comando make:

gitpod/workspace $ cd build
gitpod/workspace/build $ make

Con lo que se iniciar el proceso de generacin del archivo ejecutable, que estar ubicado en ste caso, en la direccin siguiente y se debe ejecutar de la siguiente forma:

gitpod/workspace/build/src $ ./dune-prueba

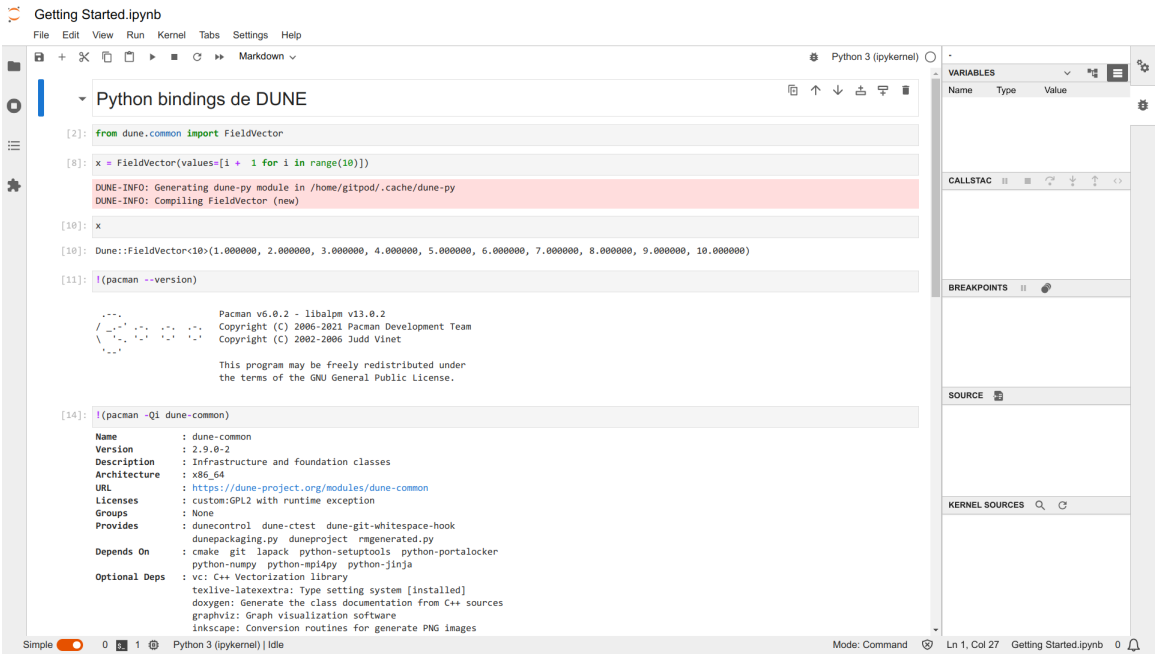
El programa y su salida se presentan a continuacin:

Hello World! This is dune-learn.
I am rank 0 of 1 processes!

https://man.archlinux.org/man/cmake.1

3 Dune en Python

3.1 Enlaces a Python



4 Dune en Julia

DuneIstlSolvers.jl

5

6