

Una introducción a la caja de herramientas DUNE Numerics para la solución de modelos matemáticos



Webinar 13 de Julio de 2021

Elaborado por:
John Jairo Leal Gómez
Universidad Nacional de Colombia
Carlos Alonso Aznarán Laos
Universidad Nacional de Ingeniería, Perú

Presentación del libro



Las matemáticas en la vida real Introducción básica al modelamiento matemático

John Jairo Leal Gómez / Juan Pablo Cardona Guío



Dirección de Investigación y Extensión
Vicerrectoría
Sede Palmira



UNIVERSIDAD
NACIONAL
DE COLOMBIA

CAPÍTULOS:

1. Introducción a los números reales \mathbb{R} .
2. Introducción a las funciones.
3. La derivada.
4. Modelamiento matemático.
5. Anexos.

Serie CIENCIAS BÁSICAS

Presentación del libro

4.3 Situaciones cotidianas

En primer lugar, se muestran “expresiones” de situaciones cotidianas con sus respectivas representaciones como funciones y sus derivadas.

4.3.1 Encender la luz



Figura 4.3.
Encender la luz

La acción de encender la luz, como en la figura 4.3, se puede escribir matemáticamente como el cambio en la posición del *switch* P como variable independiente o causa del fenómeno, y el efecto se puede ver en el cambio de la intensidad lumínica I . Esto quiere decir que la intensidad lumínica es una función de la posición del *switch* $I(P)$. La variación se puede escribir como:

$$\frac{dI}{dP}$$

4.1



DUNE Numerics Project

Distributed and Unified Numerics Environment (DUNE)

- ▶ Software de **código abierto** bajo la licencia GNU General Public Licence 2  Free as in Freedom.
- ▶ Disponible en macOS, Debian , Ubuntu , openSUSE , **Arch Linux**  y FreeBSD .
- ▶ Conjunto de bibliotecas **C++** con enlaces a **Python**.
- ▶ Utilizado en la resolución de **ecuaciones diferenciales parciales** e implementación de métodos basados en mallas, por ejemplo, *diferencias finitas*, *elementos finitos* o *volúmenes finitos*.



Origen: <https://dune-project.org/about/dune>.



Figura: Los binarios están disponible en el repositorio **Arch Linux for Education** (Jingbei Li, Carlos Azarán y otros, octubre 2022).

DUNE Numerics Project

Proyectos que emplean DUNE

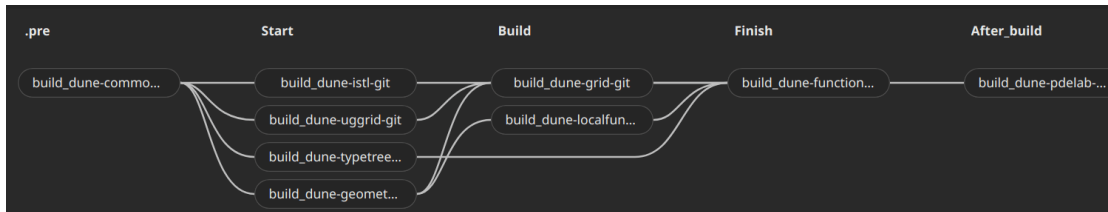
- ▶ <https://dumux.org>
- ▶ <https://opm-project.org>
- ▶ <https://precice.org>
- ▶ <https://amdis.readthedocs.io>
- ▶ <https://github.com/parafields>
- ▶ <https://www.zib.de/projects/kaskade7-finite-element-toolbox>



Origen: <https://dune-project.org/gallery>.

El DUNE verso: módulos

<https://dune-project.org/groups/core>



Origen: <https://gitlab.com/dune-archiso/repository/dune-archiso-repository-pdelab-git/-/pipelines>.

dune-common Clases fundamentales e infraestructura para la construcción del sistema.

dune-geometry Elementos de referencia, métodos de cuadraturas y transformaciones geométricas.

dune-grid Interfaces con las mallas (ALUGrid, UGGrid, AlbertaGrid, YaspGrid).

dune-istl Biblioteca de solucionadores iterativos de plantillas, clases genéricas de matrices/vectores dispersos.

dune-localfunctions Interface genérica para funciones de elementos finitos.

EI DUNE verso: módulos

Dependencias de algunos módulos

dune-fem

- dune-alugrid
- dune-istl
- dune-localfunctions
- python-fenics-ufl
- python-matplotlib
- python-scipy
- dune-polygongrid (opcional)
- dune-spgrid (opcional)
- eigen (opcional)
- papi (opcional)

opm-models

- dune-alugrid
- dune-localfunctions
- opm-grid
 - opm-common
 - suitesparse
 - zoltan
- dune-fem (opcional)

dumux


- dune-grid
- dune-istl
- dune-localfunctions
- dune-alugrid (opcional)
- dune-foamgrid (opcional)
- dune-functions (opcional)
- dune-mmesh (opcional)
- dune-spgrid (opcional)
- dune-subgrid (opcional)
- opm-grid (opcional)

dune-pdelab

- arpack++
- dune-alugrid
- dune-functions
- suitesparse
- superlu
- dune-multidomaingrid (opcional)

Curso de DUNE/PDELab 2021

<https://dune-pdelab-course.readthedocs.io>

 DUNE/PDELab Course Material

latest


Search docs

Introduction


Lectures

Questions and Answers

Licensing and Copyright

 Read the Docs

v: latest

 » Dune/PDELab Course

[Edit on GitHub](#)

Dune/PDELab Course

- [Introduction](#)
 - [About Dune](#)
 - [About this Course](#)
 - [How to study with the Material](#)
 - [Setting up the exercise environment](#)
- [Lectures](#)
 - [C++ for Scientific Computing](#)
 - [Introduction to Finite Elements](#)
 - [The Dune Grid Interface](#)
 - [Simulation Workflow](#)
 - [Elliptic Problems](#)
 - [Instationary Problems](#)
 - [Finite Volumes](#)
 - [Systems of PDEs](#)
 - [Adaptivity](#)
 - [Parallelization](#)
 - [Code Generation with Python](#)
- [Questions and Answers](#)
- [Licensing and Copyright](#)

Next

© Copyright 2021, Dune Course Team. Revision 73f0edae.
Built with [Sphinx](#) using a [theme](#) provided by [Read the Docs](#).

Snippet en C++

Listado: Programa dune-basics.cc.

```
#ifdef HAVE_CONFIG_H
#include "config.h"
#endif
#include <iostream>
#include <dune/common/parallel/mpihelper.hh> // An initializer of MPI
#include <dune/common/exceptions.hh>        // We use exceptions

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-basics." << 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;
    }
}
```

Pipeline Browser

builtin:
solution_navier_stokes-00+

Properties

Information

Properties

Apply Reset Delete ?

Search ... (use Esc to clear text)

Properties ()

- ☒ Cell/Point Array Status
- ☒ cg2_dirichlet_gfs_0_0_pow2gf
- ☒ cg2_dirichlet_gfs_0_0_pow2gf
- ☒ cg1_gfs_1

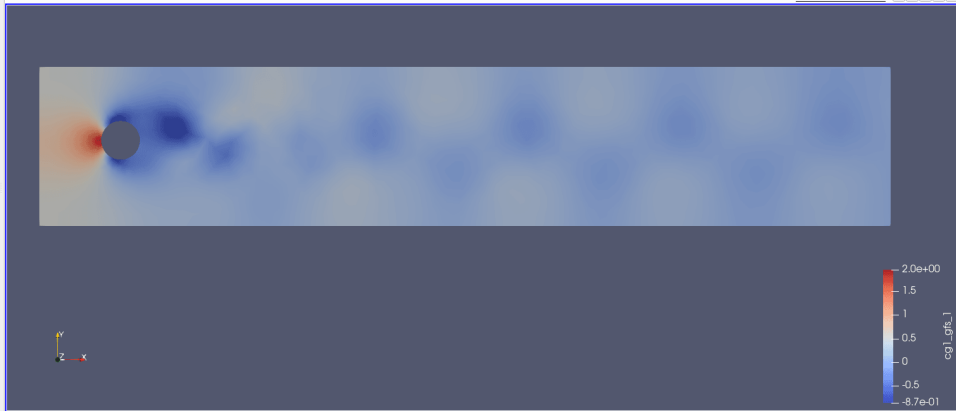
Display (Un) () () ()

Representation Surface

Coloring

Layout #1

RenderView1



Snippet en Python

<https://dune-project.org/sphinx/content/sphinx/dune-fem>

Eigenvalue problems

FURTHER TOPICS

Grid Views: Adaptivity and Moving Domains

Overview and some basic grid views (level and filtered)

Dynamic Local Grid Refinement and Coarsening

Evolving Domains

Mean Curvature Flow

Using C++ Code Snippets

EXTENSION MODULES

Discontinuous Galerkin Methods: the DUNE-FEM-DG Module

Virtual Element Methods: the DUNE-VEM module

USER PROJECTS

HP adaptive DG scheme for twophase flow problem

Mixed-dimensional PDEs: the Dune-MMesh module

INFORMATION AND RESOURCES

Information for C++ Developers

```
[1]: from ufl import *
      from dune.ufl import Constant, DirichletBC
      import dune.ufl
      import dune.geometry as geometry
      import dune.fem as fem
      from dune.fem.plotting import plotPointData as plot
      import matplotlib.pyplot as pyplot
```

set up polynomial order and radius of reference surface

```
[2]: order = 2
      R0 = 2.
```

We begin by setting up reference domain Γ_0 (`grid`), and the space on Γ_0 that describes $\Gamma(t)$ (`space`). From this we interpolate the non-spherical initial surface `positions` , and, then reconstruct `space` for the discrete solution on $\Gamma(t)$.

```
[3]: from dune.fem.view import geometryGridView
      from dune.fem.space import lagrange as solutionSpace
      from dune.alugrid import aluConformGrid as leafGridView
      gridView = leafGridView("sphere.dgf", dimgrid=2, dimworld=3)
      space = solutionSpace(gridView, dimRange=gridView.dimWorld, order=order)
      u = TrialFunction(space)
      v = TestFunction(space)
      x = SpatialCoordinate(space)
      # positions = space.interpolate(x * (1 + 0.5*sin(2*pi*x[0]*x[1])*cos(pi*x[2])), name="positions")
      positions = space.interpolate(x * (1 + 0.5*sin(2*pi*(x[0]+x[1]))*cos(0.25*pi*x[2])), name="positions")
      surface = geometryGridView(positions)
      space = solutionSpace(surface, dimRange=surface.dimWorld, order=order)
      solution = space.interpolate(x, name="solution")
```

```
GridParameterBlock: Parameter 'bisectioncompatibility' not specified, defaulting to '0' (fa
```

Finite Elements

As another example, we solve the Poisson equation

$$\begin{aligned} -\Delta u &= f && \text{in } \Omega \\ u &= 0 && \text{in } \partial\Omega \end{aligned}$$

in Python based on a simplicial Dune grid: `ALUConformGrid`.

```
[1]: import numpy as np
      from dune.grid import cartesianDomain, gridFunction
      from dune.alugrid import aluConformGrid

[2]: vertices = np.array([(0, 0), (1, 0), (1, 1), (0, 1),
                          (-1, 1), (-1, 0), (-1, -1), (0, -1)])
      triangles = np.array([(2, 0, 1), (0, 2, 3), (4, 0, 3),
                           (0, 4, 5), (6, 0, 5), (0, 6, 7)])

[3]: aluView = aluConformGrid({"vertices": vertices, "simplices": triangles})
      aluView.hierarchicalGrid.globalRefine(2)
```

```
DUNE-INFO: Generating dune-py module in /home/carlosal1015/.cache/dune-py
DUNE-INFO: Compiling HierarchicalGrid (new)
DUNE-INFO: Compiling ReferenceElements (new)
DUNE-INFO: Compiling ReferenceElements (new)
DUNE-INFO: Compiling ReferenceElements (new)
```

[Pull requests](#) [Issues](#) [Trending](#) [Explore](#)

C++ review DUNE

Una organización donde compartir notas acerca de C++ con pdfs escritos en LaTeX.

[America](#) stackoverflow.com/c/cpp-review-dune

Repositories 21 **Packages** **People** 10 **Teams** 1 **Settings**

Pinned repositories

Customize pinned repositories

introductory-review ⋮

Un repositorio donde compartir notas acerca de C++ con pdfs escritos en LaTeX.

Dockerfile 1

hdnum Template ⋮

C++

dune-basics Template ⋮

An example module that says Hello World.

TeX

github-starter-course Template ⋮

github-starter-course created by GitHub Classroom

cpp-examples Template ⋮

Forked from igormcoelho-learning/autograding-example-cpp-catch

Example of C/C++ autograding with Catch2 library - GitHub Classroom

C++

sandbox Template ⋮

Forked from corneliusludmann/gitpod-playground

This repository intentionally left empty. It merely serves as an entry point for personal Gitpod experiments.

Type ▾

Language ▾

Sort ▾

New

6 results for repositories written in C++ sorted by last updated

Clear filter

study-scientific-programming

Study of book Scientific Programming Advanced Concepts of Christian Engwer

Top languages

C++ TeX Python
 Jupyter Notebook Dockerfile



dune-archiso

Archiso profile based on CyberOS with DUNE Numerics

Status: **Beta** Brought to you by: [carlosal1015](#)

[Add a Review](#)**Downloads: 11 This Week****Last Update: 2021-06-15****Download**[Get Updates](#)[Share This](#)

Linux

[Summary](#)[Files](#)[Reviews](#)[Support](#)[Blog](#)[Discussion](#)[Admin](#)[Add New...](#)

This is a live USB containing a full operating system that can be booted, this means that you can use a USB stick to burn this image or virtualize it to Linux-KVM, QEMU, Virtualbox, VMWare, Hyper-V. We included the following repositories:

- Arch Linux Core [Official]
- Arch Linux Extra [Official]
- Arch Linux Community [Official]
- Arch Linux Multilib [Official]
- Arch4Edu [Third-party]
- Cyber [Third-party]
- Dune-archiso-repository-core [Third-party]
- Dune-archiso-repository-extra [Third-party]

In addition, we provide the packages of some modules of DUNE Numerics (version 2.7.1), DuMux (version 3.4) and the Open Porous Media (version 2021.04). The full list of packages is described in <https://dune-archiso.gitlab.io/packages>

Enjoy. I don't belong to dune-project. All the blame falls on me (github.com/carlosal1015).

Recommended Projects



Arm Mbed OS
Platform operating system designed for the Internet of...



Apache OpenOffice
The free and Open Source productivity suite



KeePass
A lightweight and easy-to-use password manager



Clonezilla
A partition and disk imaging/cloning program



7-Zip
A free file archiver for extremely high compression

Top Searches

[cyberos](#)[cyber os](#)[linux security](#)

Referencias

► Libros



Oliver Sander. *DUNE — The Distributed and Unified Numerics Environment*. First. Lecture Notes in Computational Science and Engineering 140. Springer International Publishing, 2020. ISBN: 978-3-030-59701-6. DOI: 10.1007/978-3-030-59702-3.

► Artículos



Andreas Dedner, Robert Klöforn y Martin Nolte. “The DUNE-ALUGrid Module”. En: *CoRR* abs/1407.6954 (2014). URL: <http://arxiv.org/abs/1407.6954>.



Andreas Dedner y Martin Nolte. “The Dune Python Module”. En: *CoRR* abs/1807.05252 (2018). eprint: 1807.05252. URL: <http://arxiv.org/abs/1807.05252>.



Peter Bastian et al. “The Dune framework: Basic concepts and recent developments”. En: *Computers & Mathematics with Applications* 81.1 (1 de ene. de 2021). Development and Application of Open-source Software for Problems with Numerical PDEs, págs. 75-112. ISSN: 0898-1221. DOI: <https://doi.org/10.1016/j.camwa.2020.06.007>.

Referencias

► Sitios web



Oliver Sander. *The Distributed and Unified Numerics Environment (DUNE)*. 12 de abr. de 2016. URL: <http://congress.cimne.com/icme2016/admin/files/filepaper/p72.pdf> (visitado 12-07-2021).



Alexander Jaust. *Coupling fluid flows with DuMuX, preCICE workshop 2020*. 19 de feb. de 2020. URL: <https://precice.org/precice-workshop-2020.html> (visitado 12-07-2021).



Simon Praetorius. *AMDiS Workshop 2021*. 12 de jul. de 2020. URL: <http://wwwpub.zih.tu-dresden.de/~praetori/amdis/workshop2021> (visitado 12-07-2021).



Dune Course Team. *Dune/PDELab Course*. 22 de oct. de 2020. URL: <https://dune-pdelab-course.readthedocs.io> (visitado 26-06-2021).

Agradecimientos

¡Muchas gracias!



Presentación disponible en:

[https://cpp-review-dune.github.io/webinar/
slides.pdf](https://cpp-review-dune.github.io/webinar/slides.pdf)

Grabación disponible en:

<https://player.vimeo.com/video/572717824>

Dudas, sugerencias o preguntas a:

jlealgom@unal.edu.co
caznaranl@uni.pe