FreeFEM-cs Tutorial

Antoine Le Hyaric

Laboratoire Jacques-Louis Lions, Sorbonne Université, Paris, France

2025 FreeFEM Days

Goal

Use FreeFEM-cs to create one finite-element simulation from scratch.

In this presentation

- A working finite-element example.
- Required physics and maths details.
- Links to course notes and books (online when available) for further information.

No computer science knowledge necessary

What does FreeFEM-cs bring to FreeFEM?

FreeFEM-cs uses FreeFEM and adds the following features :

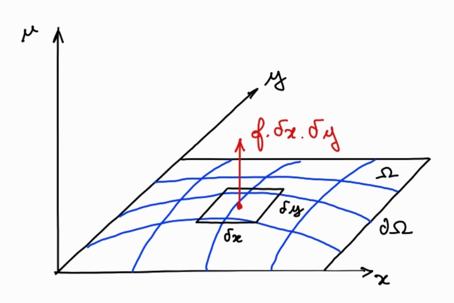
- Point-and-click app. Standard interface, one-click install.
- All available options and examples listed in menus.
- Always possible to switch back and forth with FreeFEM.

Typical uses

- Anyone willing to investigate maths and simulation but with no desire to deal with computing details.
- May be followed by potential developments in FreeFEM, Python or C++.

Membrane model

- Membrane fixed on a rigid frame
- Frame delimits a domain $\Omega \subset \mathbb{R}^2$
- Rigid frame : $u(x, y) = 0$ for $(x, y) \in \partial\Omega$
- Small vertical movements $u(x, y)$



Partial Differential Equation (PDE) [PDE]

If we apply Newton's Law of Motion [Newton] to a small patch of size $\delta x\ \delta y$, and if $\delta x \to 0$ and $\delta y \to 0$, then we obtain

Poisson's equation [Poisson]

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + f = 0 \ in\ \Omega \text{ with } u = 0 \ on\ \partial\Omega$$

- We need $u \in C^2(\Omega)$, $f \in C^1(\Omega)$ [ClassCk]
- For more details on how to obtain this equation see [LeDret1]
- $u = 0 \ on\ \partial\Omega$ is an homogeneous Dirichlet boundary condition. For other types of boundary conditions see [BVP].

Mathematical problem definition

"Strong form" or "boundary value problem" [BVP]

Hypotheses

- Ω is a Lipschitz domain [Lipschitz]
- $u \in H_0^1(\Omega)$ [Hilbert]
- $f \in L_2(\Omega)$ [LpSpace]

Mathematical analysis

- Is the problem is well-posed [WPP]?
  Existence and uniqueness of solutions in
  EN:[LeDret3], FR:[Cohen], EN/FR:[Allaire]
- Following steps
  [WeakForm], [Galerkin] method, Finite Element choice [FFfe]

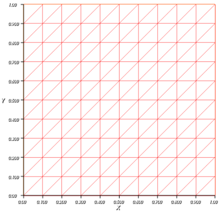
Discrete problem :

- Let h be the largest or average triangle size in a triangulation T_h of Ω
- The related Finite Element space is V_h
- We are looking for u_h, a solution for the same problem as u but restricted to V_h

FreeFEM is able to create 2D and 3D meshes [FFmesh], and to build basis functions [FFfe]. With a $[0,1]x[0,1]$ square and $h \simeq 1/10$:

FreeFEM

```
mesh Th=square(10,10);
plot(Th);
```



Hardware requirements

- Small 2D or 3D mesh ($\simeq 10^3$ elements) : any machine or smartphone

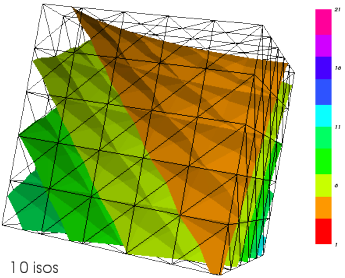


Figure: This 3D mesh of 6^3 nodes works on any machine

- Fine 2D mesh ($\simeq 10^6$ elements) : fast laptop or desktop
- Fine 3D mesh ($\simeq 10^9$ elements) : dedicated server

Download and install

`https://www.ljll.fr/lehyaric/ffcs`
(or type "freefem cs" in a search engine)

Windows

- Run the executable installer
- click on the FreeFEM-cs icon

Linux

- Untar the package to any location (e.g. the desktop)
- double-click on "FreeFEM-cs" in the package directory.

NB1 No administrator privilege required
NB2 Anyone willing to help with MacOS is welcome

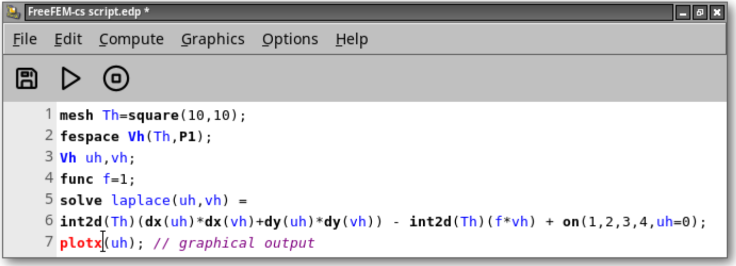
FreeFEM script

Ω is replaced by the triangulated surface T_h. We choose P1 Lagrangian finite elements (for other choices see [FFfe]). Solution u_h and test function v_h belong to the discretised space V_h.

Final program

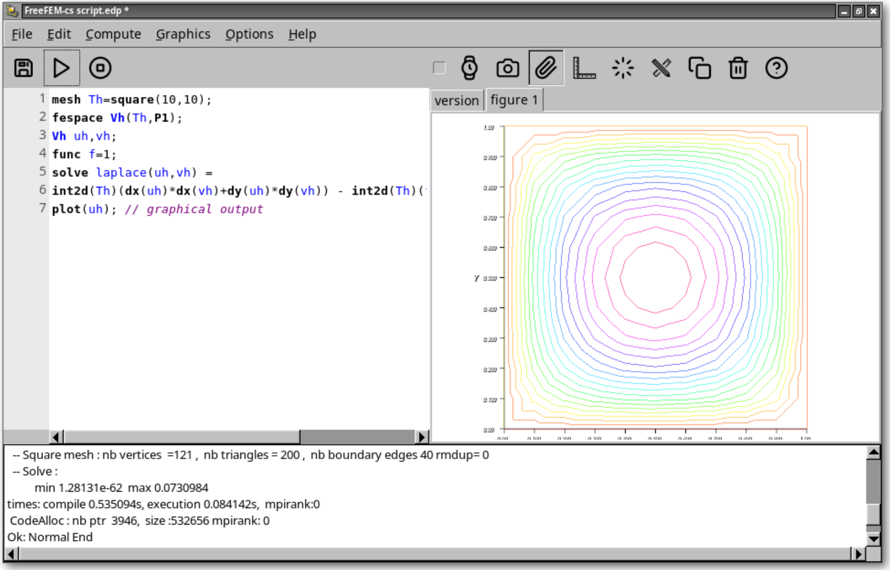
```
mesh Th=square(10,10);
fespace Vh(Th,P1);
Vh uh,vh;
func f=1;
solve laplace(uh,vh) =
  int2d(Th)(dx(uh)*dx(vh)+dy(uh)*dy(vh))
  - int2d(Th)(f*vh)
  + on(1,2,3,4,uh=0);
plot(uh); // graphical output
```

Writing the script



keyword	bold black [FFlang]
user variable	blue
string	dark blue
macro	green
comment	purple
error	bold red

Numerical result



More graphical options

All graphical options can be changed on this panel (and in the script).

All [VTK] classes could potentially be implemented here on request. [ParaView] works too.

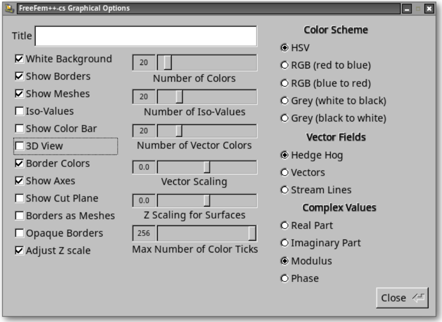
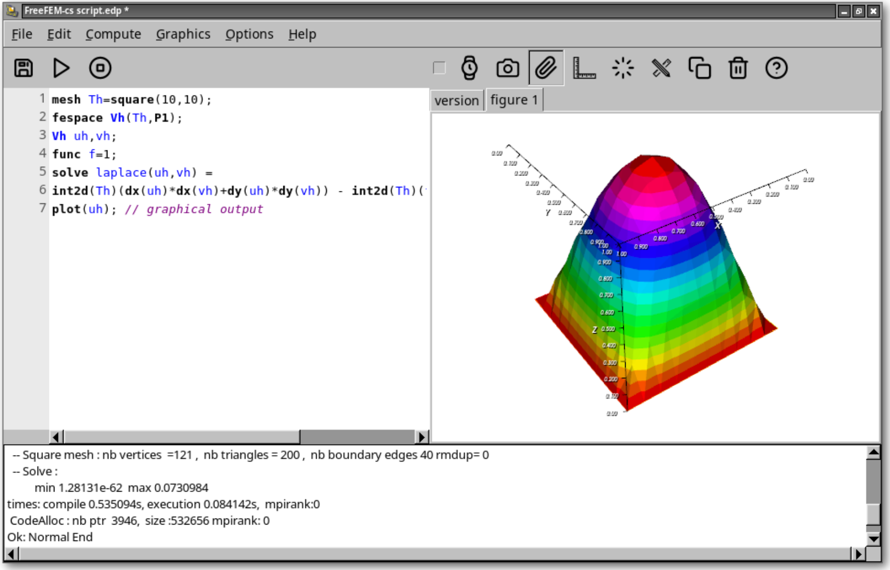


Figure: FreeFEM-cs graphical options panel

Another view of the solution



What next?

- Full online version
  `https://www.ljll.fr/lehyaric/ffcs/doc.php`
- More examples
  - List of tutorial examples : [FFlearn]
  - List of mathematical models : [FFmodels]
  - Numerical results for the above examples : [ffref]
  - Even more examples : [FFmorex]
- Start your own script from the closest example

Long term

- FreeFEM capabilities : implementation [FFimpl], language [FFlang]
- Numerical Analysis theory : online:[LeDret], book:[Allaire]
- Using Python or C++ [ffapi] (work in progress)

FreeFEM	C++	Python

`load "msh3"`
`load "tetgen"`
	`#include "ff2cpp.hpp"`
	`using namespace ff2cpp;`
	`ff2cpp::init();`
		`from ff2py import *`

`mesh Th=square(10,10,`
`[xA+(xB-xA)*x,yA+(yB-yA)*y],`
`flags=1,label=labs,region=10);`
	`mesh Th=square(10,10,`
	`{xA+(xB-xA)*x,yA+(yB-yA)*y},`
	`flags=1,label=labs,region=10);`
		`Th=square(10,10,`
		`[xA+(xB-xA)*x,yA+(yB-yA)*y],`
		`flags=1,label=labs,region=10)`

`fespace Vh(Th,P1);`	`fespace Vh(Th,P1);`	`Vh=fespace(Th,P1)`
`Vh uh;`	`fefunc uh(Vh);`	`uh=fefunc(Vh)`

Wikipedia

- [BVP] Boundary value problem (link)
- [ClassCk] Smoothness of functions (link)
- [FEM] Finite element method (link)
- [Galerkin] Galerkin method (link)
- [Green] Green's identities (link)
- [Helmholtz] Helmholtz equation (link)
- [Hilbert] Hilbert space (link)
- [LaplaceEq] Laplace's equation (link)
- [LaplaceOp] Laplace operator (link)

Wikipedia 2

- [Lipschitz] Lipschitz domain (link)
- [LpSpace] L^p space (link)
- [Newton] Newton's laws of motion (link)
- [PDE] Partial Differential Equation (link)
- [Poisson] Poisson's equation (link)
- [VectorSpace] Vector space (link)
- [WeakForm] Weak formulation (link)
- [WPP] Well-posed problem (link)

Course notes and books

- (FR) (EN) [Allaire] Analyse numérique et optimisation, Grégoire Allaire, 2005 (link)
- (FR) [Allaire2] Approximation numérique et optimisation. Une introduction à la modélisation mathématique et à la simulation numérique, Grégoire Allaire, 2019, (link)
- (FR) [Cohen] Approximations variationnelles des EDP, Notes du Cours de M2, Albert Cohen (link)
- (EN) [ChenLong] Introduction to finite element methods, Long Chen (link)

Course notes and books 2

- (EN) [Frey] The finite element approximation, Pascal Frey (link)
- (EN) [ICL] Constructing finite elements, Finite Elements: numerical analysis and implementation, Imperial College London (link)
- (EN) [LeDret] Numerical Approximation of PDEs, Master 1 Lecture Notes 2011–2012, Hervé Le Dret
  - Chapter 1 - Mathematical Modeling and PDEs (link)
  - Chapter 2 - A review of analysis (link)
  - Chapter 3 - The variational formulation of elliptic PDEs (link)
  - Chapter 4 - Variational approximation methods for elliptic PDEs (link)
  - Chapter 5 - The finite element method in dimension two (link1, link2)
  - Chapter 6 - The heat equation (link1, link2)

FreeFEM documentation

All the links below refer to the (EN) FreeFEM documentation website.

- [FFstart] Getting Started (link)
- [FFlearn] Learning by examples (link)
  - [FFmembrane] Membrane (link)
  - [FFclasses] Classification of partial differential equations (link)
- [FFimpl] Implementation (link)
  - [FFmesh] Mesh generation (link)
  - [FFfe] List of available Finite Elements in FreeFEM (link)
- [FFlang] Language reference (link)
- [FFmodels] Mathematical models (link)
  - [FFeigen] Eigenvalue problems (link)
- [FFmorex] More examples (link)

Software

- [FreeFEM-cs] FreeFEM-cs (link)
- [FreeFEM] FreeFEM (link, doc)
- [VTK] VTK, The Visualization Toolkit (link)
- [ParaView] Open-source, multi-platform data analysis and visualization application (link)
- [VSCode] VSCode, The open source AI code editor (link)
- [ffapi] Application Programming Interface for FreeFEM (link)
- [ffref] Examples from the FreeFEM documentation, with reference values (link)
- [FEsoft] List of finite element software packages, Wikipedia (link)

Other online resources

- (EN) [PeriodicTable] Periodic Table of the Finite Elements (link)
- (EN) [StackEx1] Deriving the 2D Wave Equation for a Vibrating Membrane using Newton's Law of Motion and Taylor Series, StackExchange, 2023 (link)
- (EN) [StackEx2] Conceptual difference between strong and weak formulations, StackExchange, 2013 (link)