



FEniCS Course

Lecture 1: Introduction to FEniCS

Contributors

Anders Logg

André Massing

What is FEniCS?

FEniCS is an automated programming environment for differential equations

- C++/Python library
- Initiated 2003 in Chicago
- 1000–2000 monthly downloads
- Part of Debian and Ubuntu
- Licensed under the GNU LGPL



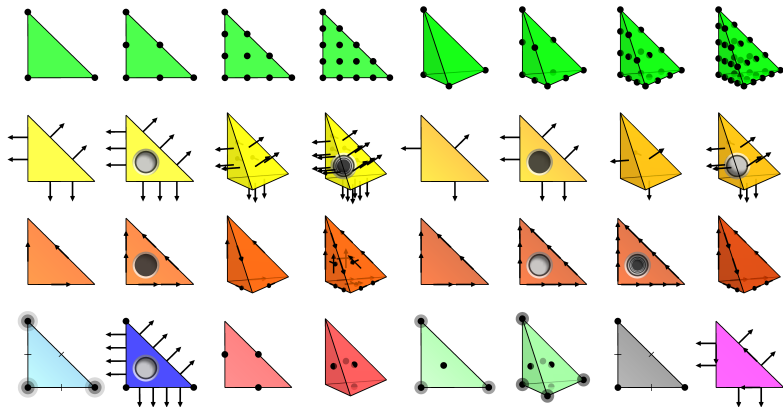
<http://fenicsproject.org/>

Collaborators

Simula Research Laboratory, University of Cambridge, University of Chicago, Texas Tech University, KTH Royal Institute of Technology, Chalmers University of Technology, Imperial College London, University of Oxford, Charles University in Prague, ...

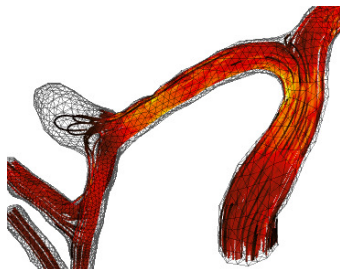
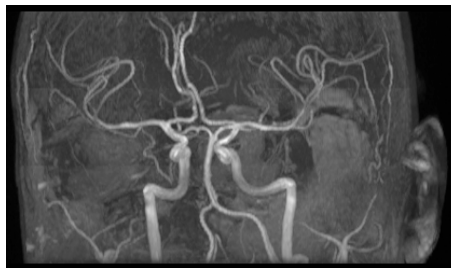
FEniCS is automated FEM

- Automated generation of basis functions
- Automated evaluation of variational forms
- Automated finite element assembly
- Automated adaptive error control



What has FEniCS been used for?

Computational hemodynamics



- Low wall shear stress may trigger aneurysm growth
- Solve the incompressible Navier–Stokes equations on patient-specific geometries

$$\begin{aligned}\dot{u} + u \cdot \nabla u - \nabla \cdot \sigma(u, p) &= f \\ \nabla \cdot u &= 0\end{aligned}$$

Computational hemodynamics (contd.)



Python code

```
# Define Cauchy stress tensor
def sigma(v,w):
    return 2.0*mu*0.5*(grad(v) + grad(v).T) -
        w*Identity(v.cell().d)

# Define symmetric gradient
def epsilon(v):
    return 0.5*(grad(v) + grad(v).T)

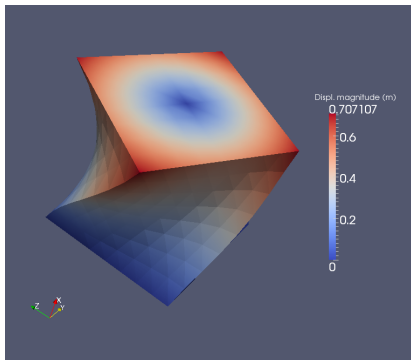
# Tentative velocity step (sigma formulation)
U = 0.5*(u0 + u)
F1 = rho*(1/k)*inner(v, u - u0)*dx + rho*inner(v,
    grad(u0)*(u0 - w))*dx \
    + inner(epsilon(v), sigma(U, p0))*dx \
    + inner(v, p0*n)*ds - mu*inner(grad(U).T*n,
        v)*ds \
    - inner(v, f)*dx
a1 = lhs(F1)
L1 = rhs(F1)

# Pressure correction
a2 = inner(grad(q), k*grad(p))*dx
L2 = inner(grad(q), k*grad(p0))*dx - q*div(u1)*dx

# Velocity correction
a3 = inner(v, u)*dx
L3 = inner(v, u1)*dx + inner(v, k*grad(p0 -
    p1))*dx
```

- The Navier–Stokes solver is implemented in Python/FEniCS
- FEniCS allows solvers to be implemented in a minimal amount of code

Hyperelasticity



Python code

```
from fenics import *

mesh = UnitCubeMesh(24, 16, 16)
V = VectorFunctionSpace(mesh, "Lagrange", 1)

left = CompiledSubDomain(("std::abs(x[0]) < DOLFIN_EPS" && on_boundary"))
right = CompiledSubDomain(("std::abs(x[0] - 1.0) < DOLFIN_EPS" && on_boundary"))

c = Expression(("0.0", "0.0", "0.0"))
r = Expression(("0.0",
"0.5*(y0+(x[1]-y0)*cos(t)-(x[2]-z0)*sin(t)-x[1])",
"0.5*(z0+(x[1]-y0)*sin(t)+(x[2]-z0)*cos(t)-x[2])"),
y0=0.5, z0=0.5, t=pi/3)
bcl = DirichletBC(V, c, left)
bcr = DirichletBC(V, r, right)
bcs = [bcl, bcr]
v = TestFunction(V)
u = Function(V)
B = Constant((0.0, -0.5, 0.0))
T = Constant((0.1, 0.0, 0.0))
I = Identity(V.cell().d)
F = I + grad(u)
Ic = tr(F.T*F)
J = det(F)
E, nu = 10.0, 0.3
mu, lambda = Constant(E/(2*(1 + nu))),
Constant(E*nu/((1 + nu)*(1 - 2*nu)))
psi = (mu/2)*(Ic - 3) - mu*ln(J) +
(lambda/2)*(ln(J))**2
Pi = psi*dx - dot(B, u)*dx - dot(T, u)*ds
F = derivative(Pi, u, v)

solve(F == 0, u, bcs)
plot(u, interactive=True, mode="displacement")
```


How to use FEniCS?

Hello World in FEniCS: problem formulation

Poisson's equation

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

Finite element formulation

Find $u \in V$ such that

$$\underbrace{\int_{\Omega} \nabla u \cdot \nabla v \, dx}_{a(u,v)} = \underbrace{\int_{\Omega} f v \, dx}_{L(v)} \quad \forall v \in V$$

Hello World in FEniCS: implementation

Python code

```
from fenics import *

mesh = UnitSquareMesh(32, 32)

V = FunctionSpace(mesh, "Lagrange", 1)
u = TrialFunction(V)
v = TestFunction(V)
f = Expression("x[0]*x[1]")

a = dot(grad(u), grad(v))*dx
L = f*v*dx

bc = DirichletBC(V, 0.0, DomainBoundary())

u = Function(V)
solve(a == L, u, bc)
plot(u)
```

Basic API

- Mesh, Vertex, Edge, Face, Facet, Cell
 - FiniteElement, FunctionSpace
 - TrialFunction, TestFunction, Function
 - grad(), curl(), div(), ...
 - Matrix, Vector, KrylovSolver, LUSolver
 - assemble(), solve(), plot()
-
- Python interface generated semi-automatically by SWIG
 - C++ and Python interfaces almost identical

Sounds great, but how do I find my way through the jungle?



Three survival advices



Use the right Python
tools



Explore the
documentation



Ask, report and
request

Use the right Python tools!

Python tools

Doc tools

- Standard terminal:
> pydoc dolfin
> pydoc dolfin.Mesh
- Python console
>>> help(dolfin)
>>> help(dolfin.Mesh)

Sophisticated Python environments

- **IDLE** the official (but rather limited) Python IDE
- **IPython** <http://ipython.org/> provides a Python shell and notebook including syntax highlighting, tab-completion, object inspection, debug assisting, history ...
- **Eclipse** plugin <http://pydev.org/> includes syntax highlighting, code completion, unit-testing, refactoring, debugger ...

IPython notebook

IP[y]: Notebook

solving-poisson

Save QuickHelp

Actions

New Open

Download (ipynb)

Print

Cell

Actions

Delete

Format Code Markdown

Output Toggle ClearAll

Insert Above Below

Move Up Down

Run Selected All

Autindent:

Kernel

Actions

Interrupt Restart

Kill kernel upon exit:

Help

Links

Python IPython

NumPy SciPy

MPL SymPy

Shift-Enter: run selected cell

Ctrl-Enter: run selected cell in-place

Ctrl-m h: show keyboard shortcuts

Configuration

Tooltip on tab:

Smart compiler:

Time before tooltip: 1200 milliseconds

Let's solve numerically the following variational problem: Find $u \in H_0^1(\Omega)$ such that $a(u, v) = L(v) \forall v \in H_0^1(\Omega)$ where $a(u, v) = \int_{\Omega} \nabla u \nabla v \, dx$ and $L(v) := \int_{\Omega} f v \, dx$. To do that in FEniCS we start by defining a mesh:

```
In [26]: from dolfin import *
m = UnitSquare(10, 10)
print m

<Mesh of topological dimension 2 (triangles) with 121 vertices and 200 cells, ordered>
```

Now we need some function space. Let's take P1 elements:

```
In [27]: V = FunctionSpace(m, "CG", 1)
```

It's time to define some test and trial functions.

```
In [28]: u = TrialFunction(V)
v = TestFunction(V)
```

And finally we can define the variational problem:

```
In [29]: a = inner(grad(u), grad(v)) * dx
f = Constant(1)
L = f * v * dx
def boundary(x, on_boundary):
    return on_boundary
u = Function(V)
zero = Constant(0)
bc = DirichletBC(V, zero, boundary)
solve(a=L, u, bc)
```

```
In [30]: plot(u)
```

```
Out[30]: <viper.viper_dolfin.Viper at 0x4b76890>
```

```
In [31]: interactive()
```

Eclipse plugin Pydev

The screenshot shows the Eclipse IDE with the PyDev plugin. The main editor window displays the following Python code:

```
import unittest

class TestCase(unittest.TestCase):

    def testRobots(self):
        from robots.core import Robot
        robot = Robot()
        robot.Walk()
        robot.SayHello()

if __name__ == '__main__':
    unittest
```

The PyDev Package Explorer on the left shows the project structure:

- Robots
 - src
 - robots
 - tests
 - test_robot.py (selected)
 - __init__.py
 - core.py
 - my.py
 - .project
 - .pydevproject
 - python (D:\bin\python265\python.exe)

The Outline view at the bottom left shows the class hierarchy:

- unittest
 - TestCase
 - testRobots
 - Robot (robots.core)
 - __main__

A tooltip for the `sys` module is visible over the code, listing various attributes and methods:

- sys
 - SystemError
 - SystemExit
 - sys_executable - setuptools.command.easy_install
 - sys_maxsize - win32com.test.testShell
 - sys_version - django.core.servers.basehttp
 - sys_version - platform
 - sys_version - wsgiref.simple_server
 - SysconfigTestCase - distutils.tests.test_sysconfig
 - sysdate - sqlalchemy.sql.functions
 - sysfile - numpy.distutils.system_info
 - sysid - xml.sax.saxutils

Press Ctrl+Space for templates.

Explore the FEniCS documentation!

Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment *DOLFIN* and the form language *UFL*.

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

The FEniCS Tutorial

A good starting point for new users is the *FEniCS Tutorial*. The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

The FEniCS Book



The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method, is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software developed as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter

of the FEniCS Book.

The FEniCS Manual

The FEniCS Manual is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

Additional Documentation

Mixing software with FEniCS is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

Documented DOLFIN demos (Python)

Documented DOLFIN demos (C++)

The demos are *already installed on your system* or can be found in the demo directory of the DOLFIN source tree.

Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

Basic classes and functions in DOLFIN (Python)

Basic classes and functions in DOLFIN (C++)

Complete Programmer's References

All classes and functions in DOLFIN (Python)

All classes and functions in DOLFIN (C++)

All classes and functions in UFL

Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment *DOLFIN* and the form language *UFL*.

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

The FEniCS Tutorial

A good starting point for new users is the *FEniCS Tutorial*. The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

The FEniCS Book



The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method, is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software developed as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter of the FEniCS Book.

The FEniCS Manual

The FEniCS Manual is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

Additional Documentation

Mixing software with FEniCS is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

Documented DOLFIN demos (Python)

Documented DOLFIN demos (C++)

The demos are *already installed on your system* or can be found in the demo directory of the DOLFIN source tree.

Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

Basic classes and functions in DOLFIN (Python)

Basic classes and functions in DOLFIN (C++)

Complete Programmer's References

All classes and functions in DOLFIN (Python)

All classes and functions in DOLFIN (C++)

All classes and functions in UFL

Documentation for FEniCS 1.3.0


Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment *DOLFIN* and the form language *UFL*.

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

The FEniCS Tutorial

A good starting point for new users is the *FEniCS Tutorial*. The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

The FEniCS Book



The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method, is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software developed as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter of the FEniCS Book.

The FEniCS Manual

The *FEniCS Manual* is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

Additional Documentation

Mixing software with FEniCS is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

[Documented DOLFIN demos \(Python\)](#)

[Documented DOLFIN demos \(C++\)](#)

The demos are *already installed on your system* or can be found in the demo directory of the DOLFIN source tree.

Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

[Basic classes and functions in DOLFIN \(Python\)](#)

[Basic classes and functions in DOLFIN \(C++\)](#)

Complete Programmer's References

[All classes and functions in DOLFIN \(Python\)](#)

[All classes and functions in DOLFIN \(C++\)](#)

[All classes and functions in UFL](#)

<http://fenicsproject.org/documentation/>

Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment *DOLFIN* and the form language *UFL*.

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

The FEniCS Tutorial

A good starting point for new users is the *FEniCS Tutorial*. The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

The FEniCS Book



The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method, is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software developed as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter of the FEniCS Book.

The FEniCS Manual

The *FEniCS Manual* is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

Additional Documentation

Mixing software with FEniCS is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

Documented DOLFIN demos (Python)

Documented DOLFIN demos (C++)

The demos are *already installed on your system* or can be found in the demo directory of the DOLFIN source tree.

Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

Basic classes and functions in DOLFIN (Python)

Basic classes and functions in DOLFIN (C++)

Complete Programmer's References

All classes and functions in DOLFIN (Python)

All classes and functions in DOLFIN (C++)

All classes and functions in UFL

<http://fenicsproject.org/documentation/>

Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment *DOLFIN* and the form language *UFL*.

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

The FEniCS Tutorial

A good starting point for new users is the *FEniCS Tutorial*. The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

The FEniCS Book



The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method, is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software developed as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter

of the FEniCS Book.

The FEniCS Manual

The FEniCS Manual is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

Additional Documentation

Mixing software with FEniCS is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

[Documented DOLFIN demos \(Python\)](#)

[Documented DOLFIN demos \(C++\)](#)

The demos are *already installed on your system* or can be found in the demo directory of the DOLFIN source tree.

Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

[Basic classes and functions in DOLFIN \(Python\)](#)

[Basic classes and functions in DOLFIN \(C++\)](#)

Complete Programmer's References

[All classes and functions in DOLFIN \(Python\)](#)

[All classes and functions in DOLFIN \(C++\)](#)

[All classes and functions in UFL](#)

<http://fenicsproject.org/documentation/>

Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment *DOLFIN* and the form language *UFL*.

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

The FEniCS Tutorial

A good starting point for new users is the *FEniCS Tutorial*. The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

The FEniCS Book



The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method, is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software developed as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter of the FEniCS Book.

The FEniCS Manual

The *FEniCS Manual* is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

Additional Documentation

Mixing software with FEniCS is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

[Documented DOLFIN demos \(Python\)](#)

[Documented DOLFIN demos \(C++\)](#)

The demos are *already installed on your system* or can be found in the demo directory of the DOLFIN source tree.

Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

[Basic classes and functions in DOLFIN \(Python\)](#)

[Basic classes and functions in DOLFIN \(C++\)](#)

Complete Programmer's References

[All classes and functions in DOLFIN \(Python\)](#)

[All classes and functions in DOLFIN \(C++\)](#)

[All classes and functions in UFL](#)

Ask questions, report bugs and request new features!

Development community is organized via bitbucket.org

The screenshot shows the Bitbucket web interface for the repository 'fenics-project/DOLFIN'. The browser address bar shows the URL 'https://bitbucket.org/fenics-project/dolfin'. The page layout includes a left-hand navigation sidebar with sections for 'ACTIONS' (Clone, Create branch, Create pull request, Compare, Fork) and 'NAVIGATION' (Overview, Source, Commits, Branches, Pull requests, Issues, Wiki, Downloads, Settings). The main content area is titled 'Overview' and displays repository statistics: '99+ Branches', '3 Tags', '48 Forks', and '73 Watchers'. It also features an 'Invite users to this repo' box with a 'Send invitation' button and a 'Recent activity' section listing three recent commits with their authors and timestamps. Below the overview, there are sections for 'DOLFIN' (description), 'Installation' (with a code block for build instructions), and 'License' (GNU Lesser General Public License).

fenics-project / DOLFIN — Bitbucket - Mozilla Firefox

Atlassian, Inc. (US) | https://bitbucket.org/fenics-project/dolfin

Bitbucket Dashboard Teams Repositories Snippets Create Find a repository...

fenics-project DOLFIN

ACTIONS

- Clone
- Create branch
- Create pull request
- Compare
- Fork

NAVIGATION

- Overview
- Source
- Commits
- Branches
- Pull requests
- Issues
- Wiki
- Downloads
- Settings

Overview

SSH git@bitbucket.org:fenics-project/dolfin Share

Last updated	6 minutes ago	99+ Branches	3 Tags
Language	C++	48 Forks	73 Watchers
Access level	Admin (revoke)		

Invite users to this repo

Send invitation

Recent activity

- 1 commit Pushed to fenics-project/dolfin | 8bc2b98 Merge branch 'garth/replace-boo... Garth Wells · 7 minutes ago
- 1 commit Pushed to fenics-project/dolfin | 8db4a67 Merge branch 'garth/replace-lexi... Garth Wells · 10 minutes ago
- 1 commit Pushed to fenics-project/dolfin | ab59d44 Replace Boost lexical_cast with ... Garth Wells · 10 minutes ago
- 1 commit Pushed to fenics-project/dolfin | be6345d Merge branch 'garth/replace-boo... Garth Wells · 35 minutes ago

DOLFIN

DOLFIN is the C++/Python interface of FEniCS, providing a consistent PSE (Problem Solving Environment) for ordinary and partial differential equations.

Installation

To build DOLFIN, run:

```
mkdir build
cd build
cmake ..
make install
```

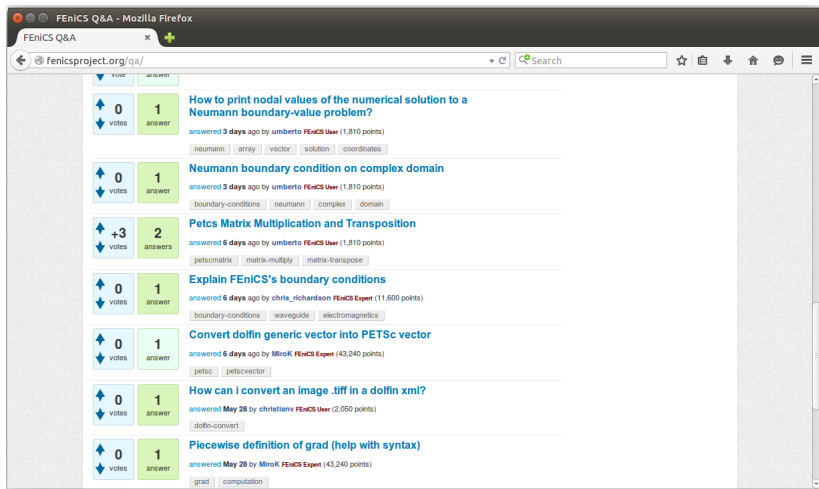
For detailed instructions, see the file INSTALL.

License

DOLFIN is free software: you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

<http://bitbucket.org/fenics-project/>

Community help is available via QA forum



The screenshot shows a web browser window titled "FEniCS Q&A - Mozilla Firefox" with the URL "fenicsproject.org/qa/". The page displays a list of questions and answers. Each entry includes a question title, the number of votes and answers, the user who answered, the time since the answer was posted, and the user's reputation points. Tags are also present for each question.

Question Title	Votes	Answers	Answered By	Points	Time Ago	Tags
How to print nodal values of the numerical solution to a Neumann boundary-value problem?	0	1	umberto FEniCS User	1,810	3 days ago	neumann, array, vector, solution, coordinates
Neumann boundary condition on complex domain	0	1	umberto FEniCS User	1,810	3 days ago	boundary-conditions, neumann, complex, domain
Petsc Matrix Multiplication and Transposition	+3	2	umberto FEniCS User	1,810	6 days ago	petscmatrix, matrix-multiply, matrix-transpose
Explain FEniCS's boundary conditions	0	1	chris_richardson FEniCS Expert	11,600	6 days ago	boundary-conditions, waveguide, electromagnetics
Convert dolfin generic vector into PETSc vector	0	1	MiroK FEniCS Expert	43,240	6 days ago	petsc, petscvector
How can i convert an image .tiff in a dolfin xml?	0	1	christianv FEniCS User	2,050	May 28	dolfin-convert
Piecewise definition of grad (help with syntax)	0	1	MiroK FEniCS Expert	43,240	May 28	grad, computation

<https://fenicsproject.org/qa>

Installation



☞ Use official packages for Debian and Ubuntu



☞ Use drag and drop installation on Mac OS X



☞ Use VirtualBox + official FEniCS image



☞ Build from source (fenics-install.sh)



☞ Other options: Docker, Conda packages

<http://fenicsproject.org/download/>

The FEniCS challenge!

- 1 Install FEniCS on your laptop!

<http://fenicsproject.org/download/>

- 2 Find and execute `demo_cahn-hilliard.py`, try to visualize the results with Paraview.
- 3 What are the main packages of the `dolfin` module?
- 4 Which elements are supported in `dolfin`?
- 5 Plot at least two finite elements from each row on page 4 and identify those elements you are most curious about!