FreeFem++-cs, the FreeFem++ Graphical Interface

Antoine Le Hyaric

Laboratoire Jacques-Louis Lions
Université Pierre et Marie Curie

Antoine.Le_Hyaric@upmc.fr

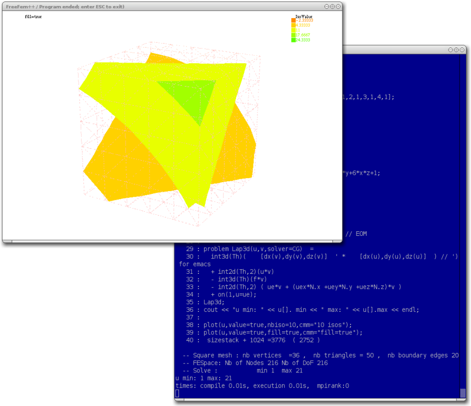
December 10, 2014

How to start using FF?

- The mathematical bases for the FF language are very well documented (PDF, HTML and Wiki available online)
- But there is more to know to run a simulation successfully
- Minimize the FF learning curve through the use of the FFCS interface.

FreeFem++ or FreeFem++-cs?

FF is a full-featured standalone application. FFCS is a shell around it. This is FF :

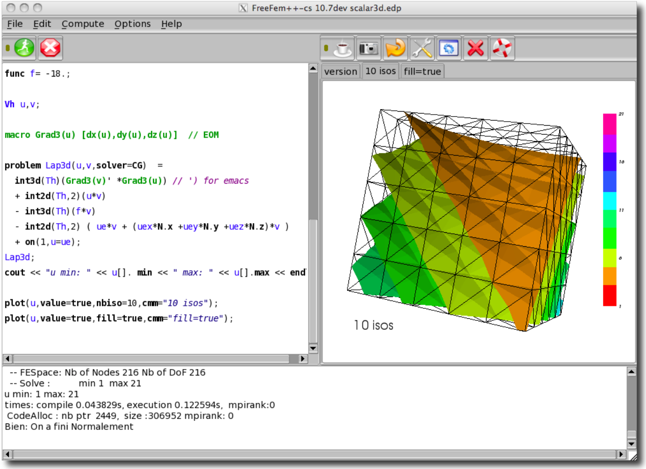


```
1,2,1,3,1,4,1];

*y+6*x*z+1;

// EOM
  29 : problem Lap3d(u,v,solver=CG)  =
  30 :   int3d(Th)(    [dx(v),dy(v),dz(v)]  ' *    [dx(u),dy(u),dz(u)]  ) // ')
 for emacs
  31 :   + int2d(Th,2)(u*v)
  32 :   - int3d(Th)(f*v)
  33 :   - int2d(Th,2) ( ue*v + (uex*N.x +uey*N.y +uez*N.z)*v )
  34 :   + on(1,u=ue);
  35 : Lap3d;
  36 : cout << "u min: " << u[]. min << " max: " << u[].max << endl;
  37 :
  38 : plot(u,value=true,nbiso=10,cmm="10 isos");
  39 : plot(u,value=true,fill=true,cmm="fill=true");
  40 :  sizestack + 1024 =3776  ( 2752 )

  -- Square mesh : nb vertices  =36 ,  nb triangles = 50 ,  nb boundary edges 20
  -- FESpace: Nb of Nodes 216 Nb of DoF 216
  -- Solve :           min 1  max 21
u min: 1 max: 21
times: compile 0.01s, execution 0.01s,  mpirank:0
```

FFCS brings the following extra features :

- graphical shell (more comfortable for first-time users)
- inline documentation (manual, examples, reference card)
- Minimizing the learning curve: available options explicitely listed in menus.
- always possible to switch back and forth between FFCS and FF at any time
- Users should be able to start from scratch (includes editor!)
- Create a simple interface for most uses

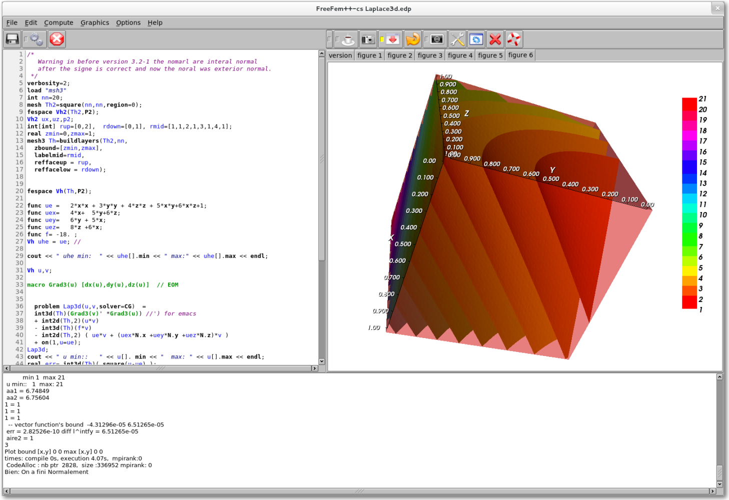
This is FFCS


FreeFem++-cs 10.7dev scalar3d.edp
File Edit Compute Options Help
version
10 isos
fill=true
21
16
11
6
1
10 isos

```
func f= -18.;

Vh u,v;

macro Grad3(u) [dx(u),dy(u),dz(u)]  // EOM

problem Lap3d(u,v,solver=CG)  =
  int3d(Th)(Grad3(v)' *Grad3(u)) // ') for emacs
  + int2d(Th,2)(u*v)
  - int3d(Th)(f*v)
  - int2d(Th,2) ( ue*v + (uex*N.x +uey*N.y +uez*N.z)*v )
  + on(1,u=ue);
Lap3d;
cout << "u min: " << u[]. min << " max: " << u[].max << end

plot(u,value=true,nbiso=10,cmm="10 isos");
plot(u,value=true,fill=true,cmm="fill=true");
```

```
  -- FESpace: Nb of Nodes 216 Nb of DoF 216
  -- Solve :          min 1  max 21
u min: 1 max: 21
times: compile 0.043829s, execution 0.122594s,  mpirank:0
 CodeAlloc : nb ptr  2449,  size :306952 mpirank: 0
Bien: On a fini Normalement
```

Graphical User Interface

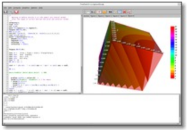

FreeFem++-cs Laplace3d.edp
File Edit Compute Graphics Options Help
version figure 1 figure 2 figure 3 figure 4 figure 5 figure 6
X
Y
Z
1.00 0.900 0.800 0.700 0.600 0.500 0.400 0.300 0.200 0.100 0.00
21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

```
/*
   Warning in before version 3.2-1 the nomarl are interal normal
   after the signe is correct and now the noral was exterior normal.
 */
verbosity=2;
load "msh3"
int nn=20;
mesh Th2=square(nn,nn,region=0);
fespace Vh2(Th2,P2);
Vh2 ux,uz,p2;
int[int] rup=[0,2],  rdown=[0,1], rmid=[1,1,2,1,3,1,4,1];
real zmin=0,zmax=1;
mesh3 Th=buildlayers(Th2,nn,
  zbound=[zmin,zmax],
  labelmid=rmid,
  reffaceup = rup,
  reffacelow = rdown);


fespace Vh(Th,P2);

func ue =   2*x*x + 3*y*y + 4*z*z + 5*x*y+6*x*z+1;
func uex=   4*x+  5*y+6*z;
func uey=   6*y + 5*x;
func uez=   8*z +6*x;
func f= -18. ;
Vh uhe = ue; //

cout << " uhe min:  " << uhe[].min << " max:" << uhe[].max << endl;

Vh u,v;

macro Grad3(u) [dx(u),dy(u),dz(u)]  // EOM


  problem Lap3d(u,v,solver=CG)  =
  int3d(Th)(Grad3(v)' *Grad3(u)) //') for emacs
  + int2d(Th,2)(u*v)
  - int3d(Th)(f*v)
  - int2d(Th,2) ( ue*v + (uex*N.x +uey*N.y +uez*N.z)*v )
  + on(1,u=ue);
Lap3d;
cout << " u min::   " << u[]. min << "  max: " << u[].max << endl;
real err= int3d(Th)( square(u-ue) );
```

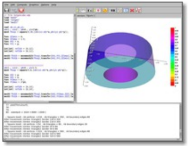
```
       min 1  max 21
 u min::  1  max: 21
 aa1 = 6.74849
 aa2 = 6.75604
1 = 1
1 = 1
1 = 1
 -- vector function's bound  -4.31296e-05 6.51265e-05
 err = 2.82526e-10 diff l^intfy = 6.51265e-05
 aire2 = 1
3
Plot bound [x,y] 0 0 max [x,y] 0 0
times: compile 0s, execution 4.07s,  mpirank:0
 CodeAlloc : nb ptr  2828,  size :336952 mpirank: 0
Bien: On a fini Normalement
```

More screenshots

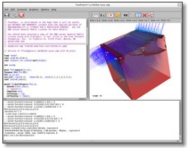
Version 13.6



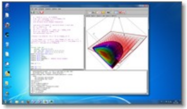
Version 12.12



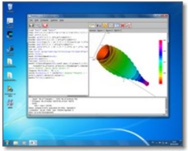
Version 12.8



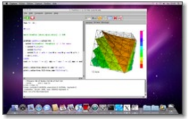
Version 11.13 on Windows 7



Version 10.14 on Windows 7



Version 10.7 on MacOS 10.6



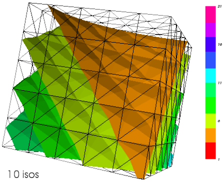
Which machine?

- **Coarse 2D or 3D meshes :** any machine (even a palmtop) will do.
- **Fine 2D meshes :** a fast laptop.
- **Fine 3D meshes :** HPC + MPI

The operating system can be Windows, Linux or MacOS. FF and FFCS work fine under all of these.

Which Machine? (2)

As an example, the following simulation runs fine on any machine :



Some examples of bigger test cases at
`http://www.ljll.math.upmc.fr/~gostaf/mfd2010`

Installing FreeFem++-cs

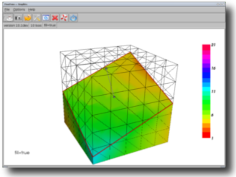
This is the easy part.

- automatic install procedure
- No other software needed
- No administrator privileges needed

FreeFem++-vtk

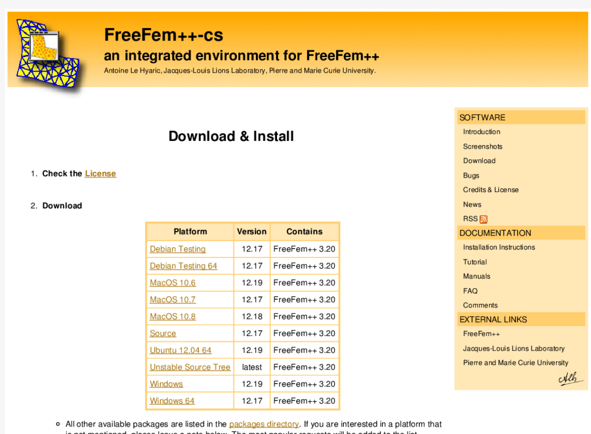
FreeFem++-vtk : Same graphics as FFCS, same command line as FF

- Install FF
- Install FFCS
- run FF with `-glut .../FreeFem++-vtk`



Download

`www.ljll.math.upmc.fr/lehyaric/ffcs/install.php`



FreeFem++-cs

an integrated environment for FreeFem++

Antoine Le Hyaric, Jacques-Louis Lions Laboratory, Pierre and Marie Curie University.

Download & Install

1. **Check the License**

2. **Download**

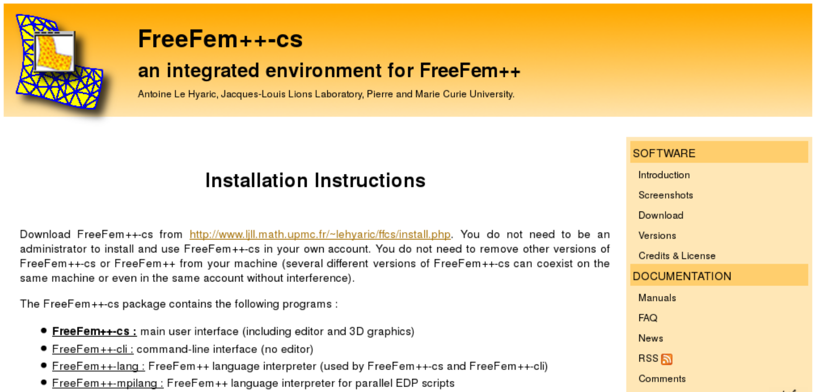
Platform	Version	Contains
Debian Testing	12.17	FreeFem++ 3.20
Debian Testing 64	12.17	FreeFem++ 3.20
MacOS 10.6	12.19	FreeFem++ 3.20
MacOS 10.7	12.17	FreeFem++ 3.20
MacOS 10.8	12.18	FreeFem++ 3.20
Source	12.17	FreeFem++ 3.20
Ubuntu 12.04 64	12.19	FreeFem++ 3.20
Unstable Source Tree	latest	FreeFem++ 3.20
Windows	12.19	FreeFem++ 3.20
Windows 64	12.17	FreeFem++ 3.20

- All other available packages are listed in the packages directory. If you are interested in a platform that is not mentioned, please leave a note below. The most popular requests will be added to the list.

Installation instructions

Installation instructions (from Source, on Windows, on MacOS, on Linux, etc)

`www.ljll.math.upmc.fr/lehyaric/ffcs/quickstart.php`



FreeFem++-cs

an integrated environment for FreeFem++

Antoine Le Hyaric, Jacques-Louis Lions Laboratory, Pierre and Marie Curie University.

Installation Instructions

Download FreeFem++-cs from http://www.ljll.math.upmc.fr/~lehyaric/ffcs/install.php. You do not need to be an administrator to install and use FreeFem++-cs in your own account. You do not need to remove other versions of FreeFem++-cs or FreeFem++ from your machine (several different versions of FreeFem++-cs can coexist on the same machine or even in the same account without interference).

The FreeFem++-cs package contains the following programs :

- **FreeFem++-cs :** main user interface (including editor and 3D graphics)
- FreeFem++-cli : command-line interface (no editor)
- FreeFem++-lang : FreeFem++ language interpreter (used by FreeFem++-cs and FreeFem++-cli)
- FreeFem++-mpilang : FreeFem++ language interpreter for parallel EDP scripts

Prerequisites

- the Finite Element Method, variational formulation `www.um.es/freefemv3/ff++/pmwiki.php?n=FFDoc.GettingStarted`
- C language syntax

Detailed examples

The FreeFem++ documentation contains some very detailed tutorial examples, described from their mathematical definition to the resulting FF program in chapter 3 :
`www.um.es/freefemv3/ff++/pmwiki.php?n=FFDoc.LearningByExamples`
Membranes, Heat Exchanger, Acoustics, Thermal Conduction, Irrotational Fan Blade Flow and Thermal, effects, Pure Convection : The Rotating Hill, A Projection Algorithm for the Navier-Stokes equations, The System of elasticity, The System of Stokes for Fluids, A Large Fluid Problem, An Example with Complex Numbers, Optimal Control, A Flow with Shocks

The list of all FreeFem++ examples

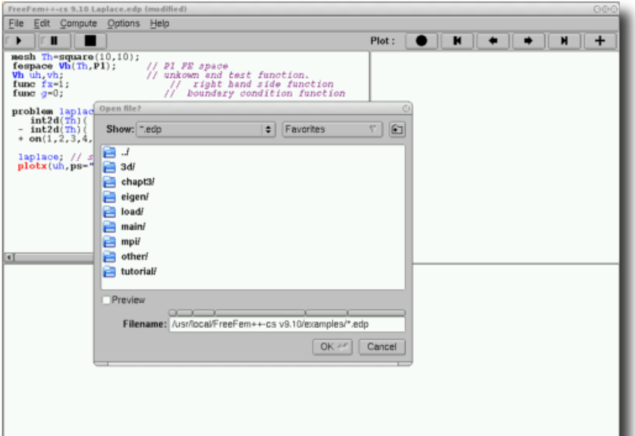
Starting from an existing example makes it easier to try something easily avoid the blank page syndrom. All FF examples (including shared libraries and 3D) are available through :

`www.ljll.math.upmc.fr/lehyaric/ffcs/examples`

tutorial, main, 3d, eigenvalues, load modules, MPI, etc

Viewing examples from FFCS

The "File/Open Example..." menu option in FFCS :



```
mesh Th=square(10,10);
fespace Vh(Th,P1);        // P1 FE space
Vh uh,vh;                 // unkown and test function.
func fx=1;                   //  right hand side function
func g=0;                   //  boundary condition function

problem laplac
    int2d(Th)(
 -  int2d(Th)(
 +  on(1,2,3,4,

 laplace; // s
 plotx(uh,ps="
```

Show: *.edp

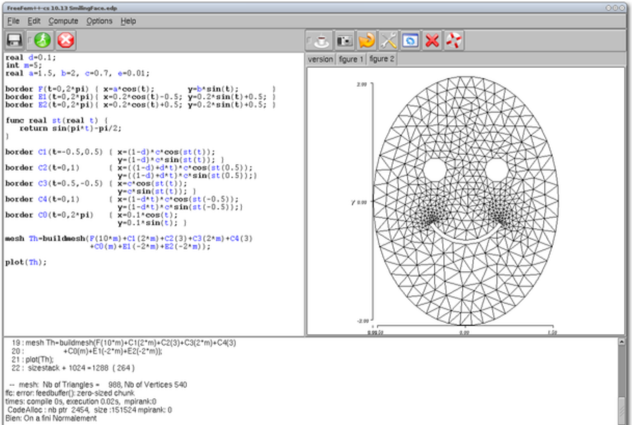
Favorites

- ../
- 3d/
- chapt3/
- eigen/
- load/
- main/
- mpi/
- other/
- tutorial/

Filename: /usr/local/FreeFem++-cs v9.10/examples/*.edp

Writing a program

All 3 steps of The Development Cycle (edit,run,visualize) in one window.



```
real d=0.1;
int m=5;
real a=1.5, b=2, c=0.7, e=0.01;

border F(t=0,2*pi) { x=a*cos(t);        y=b*sin(t);         }
border E1(t=0,2*pi){ x=0.2*cos(t)-0.5; y=0.2*sin(t)+0.5; }
border E2(t=0,2*pi){ x=0.2*cos(t)+0.5; y=0.2*sin(t)+0.5; }

func real st(real t) {
   return sin(pi*t)-pi/2;
}

border C1(t=-0.5,0.5) { x=(1-d)*c*cos(st(t));
                        y=(1-d)*c*sin(st(t)); }
border C2(t=0,1)      { x=((1-d)+d*t)*c*cos(st(0.5));
                        y=((1-d)+d*t)*c*sin(st(0.5));}
border C3(t=0.5,-0.5) { x=c*cos(st(t));
                        y=c*sin(st(t)); }
border C4(t=0,1)      { x=(1-d*t)*c*cos(st(-0.5));
                        y=(1-d*t)*c*sin(st(-0.5));}
border C0(t=0,2*pi)   { x=0.1*cos(t);
                        y=0.1*sin(t); }

mesh Th=buildmesh(F(10*m)+C1(2*m)+C2(3)+C3(2*m)+C4(3)
                  +C0(m)+E1(-2*m)+E2(-2*m));

plot(Th);
```

```
  19 : mesh Th=buildmesh(F(10*m)+C1(2*m)+C2(3)+C3(2*m)+C4(3)
  20 :                  +C0(m)+E1(-2*m)+E2(-2*m));
  21 : plot(Th);
  22 :  sizestack + 1024 =1288  ( 264 )

 --  mesh:  Nb of Triangles =     988, Nb of Vertices 540
fc: error: feedbuffer(): zero-sized chunk
times: compile 0s, execution 0.02s,  mpirank:0
 CodeAlloc : nb ptr  2454,  size :151524 mpirank: 0
Bien: On a fini Normalement
```

2d live demo

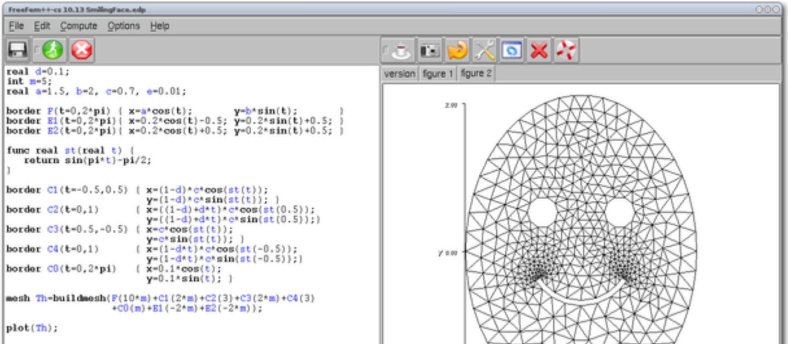
```
mesh Th=square(10,10);
fespace Vh(Th,P1);
Vh uh,vh;
func f=1;
func g=0;
solve Laplace(uh,vh)=int2d(Th)(dx(uh)*dx(vh)+dy(uh)*dy(vh))
-int2d(Th)(f*vh)
+on(1,2,3,4,uh=g);
plot(uh);
```

3d live demo

```
load ”msh3”
mesh Th2=square(10,10);
int[int] rup=[0,1],rlow=[0,1],rmid=[1,1,2,1,3,1,4,1];
mesh3
Th=buildlayers(Th2,10,reffaceup=rup,reffacelow=rlow,reffacemid=rmid);
fespace Vh(Th,P1);
Vh uh,vh;
func f=1;
func g=0;
solve
Laplace(uh,vh)=int3d(Th)(dx(uh)*dx(vh)+dy(uh)*dy(vh)+dz(uh)*dz(vh))
-int3d(Th)(f*vh)
+on(1,uh=g);
plot(uh);
```

Describing the mesh programmatically

- very handy in 2d (not so much in 3d)
- easily parameterized once written
- `www.um.es/freefemv3/ff++/pmwiki.php?n=FFDoc.MeshGeneration`

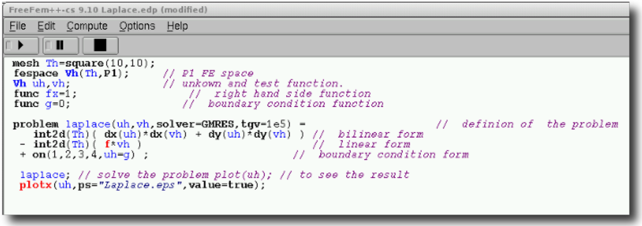


```
real d=0.1;
int m=5;
real a=1.5, b=2, c=0.7, e=0.01;

border F(t=0,2*pi) { x=a*cos(t);         y=b*sin(t);         }
border E1(t=0,2*pi){ x=0.2*cos(t)-0.5; y=0.2*sin(t)+0.5; }
border E2(t=0,2*pi){ x=0.2*cos(t)+0.5; y=0.2*sin(t)+0.5; }

func real st(real t) {
   return sin(pi*t)-pi/2;
}

border C1(t=-0.5,0.5) { x=(1-d)*c*cos(st(t));
                        y=(1-d)*c*sin(st(t)); }
border C2(t=0,1)      { x=((1-d)+d*t)*c*cos(st(0.5));
                        y=((1-d)+d*t)*c*sin(st(0.5));}
border C3(t=0.5,-0.5) { x=c*cos(st(t));
                        y=c*sin(st(t)); }
border C4(t=0,1)      { x=(1-d*t)*c*cos(st(-0.5));
                        y=(1-d*t)*c*sin(st(-0.5));}
border C0(t=0,2*pi)   { x=0.1*cos(t);
                        y=0.1*sin(t); }

mesh Th=buildmesh(F(10*m)+C1(2*m)+C2(3)+C3(2*m)+C4(3)
                  +C0(m)+E1(-2*m)+E2(-2*m));

plot(Th);
```

External CAD tools

- In 3D, tetgen can be called from an FF program, but programming the intersections is cumbersome
- For basic shapes (cube, sphere, ...) see `examples/examples++-3d`
- This is where it is useful to use a CAD tool like GMSH :
  - Draw geometry and build the mesh in GMSH. Then export it to the MEDIT format.
  - Find the numeric labels of the boundaries which will need a boundary condition (e.g. by loading the mesh in MEDIT and displaying the material properties of hand-picked points on the boundaries).
  - Load the mesh into the FF program with readmesh3().

Editing and running

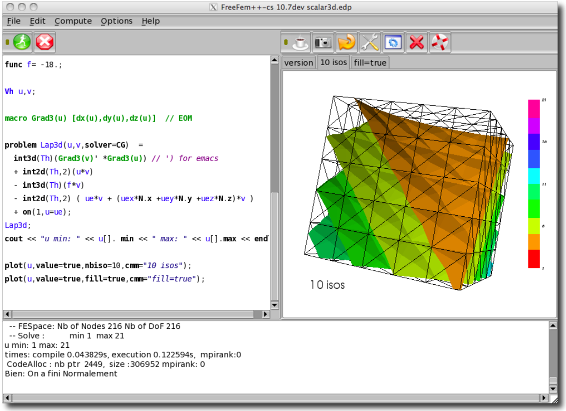


```
mesh Th=square(10,10);
fespace Vh(Th,P1);       // P1 FE space
Vh uh,vh;                // unkown and test function.
func fx=1;                  //  right hand side function
func g=0;                  //  boundary condition function

problem laplace(uh,vh,solver=GMRES,tgv=1e5) =              //  definion of  the problem
    int2d(Th)( dx(uh)*dx(vh) + dy(uh)*dy(vh) ) //  bilinear form
  - int2d(Th)( f*vh )                          //  linear form
  + on(1,2,3,4,uh=g) ;                      //  boundary condition form

 laplace; // solve the problem plot(uh); // to see the result
 plotx(uh,ps="Laplace.eps",value=true);
```

bold black : all standard FreeFem++ keywords. **blue** : user-defined objects User-defined types are in **bold face**. Character strings are in *"dark blue"*. **green** : macros and include files. **bold red** : spelling mistakes, syntax errors, undefined objects, macros and include files containing errors. **purple** : comments.

Visualizing

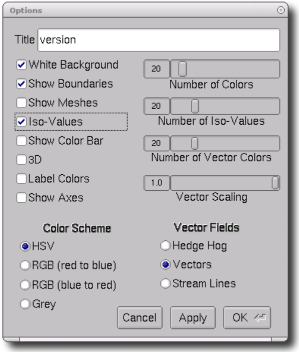
Simply use the plot() command with 3D data :



```
func f= -18.;

Vh u,v;

macro Grad3(u) [dx(u),dy(u),dz(u)]  // EOM

problem Lap3d(u,v,solver=CG)  =
  int3d(Th)(Grad3(v)' *Grad3(u)) // ') for emacs
  + int2d(Th,2)(u*v)
  - int3d(Th)(f*v)
  - int2d(Th,2) ( ue*v + (uex*N.x +uey*N.y +uez*N.z)*v )
  + on(1,u=ue);
Lap3d;
cout << "u min: " << u[]. min << " max: " << u[].max << end

plot(u,value=true,nbiso=10,cmm="10 isos");
plot(u,value=true,fill=true,cmm="fill=true");
```

```
  -- FESpace: Nb of Nodes 216 Nb of DoF 216
  -- Solve :          min 1  max 21
u min: 1 max: 21
times: compile 0.043829s, execution 0.122594s,  mpirank:0
 CodeAlloc : nb ptr  2449,  size :306952 mpirank: 0
Bien: On a fini Normalement
```

- Graphical options are modifiable from the FF program itself
- All VTK classes are potentially usable
- We are not reimplementing Paraview inside FFCS



Options

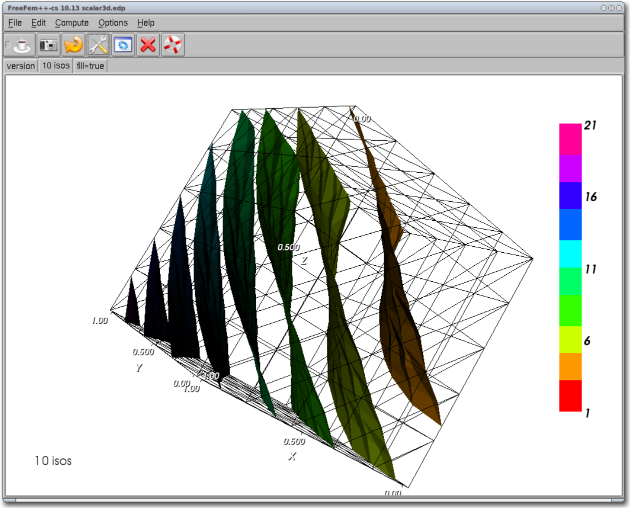
Title: version

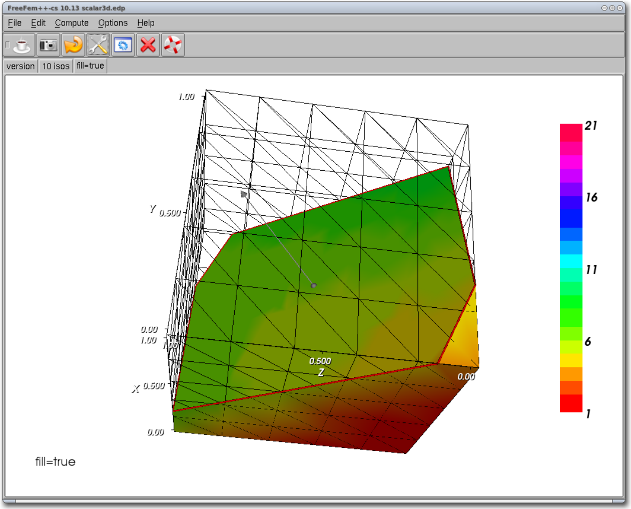
☑ White Background
☑ Show Boundaries
□ Show Meshes
☑ Iso-Values
□ Show Color Bar
□ 3D
□ Label Colors
□ Show Axes

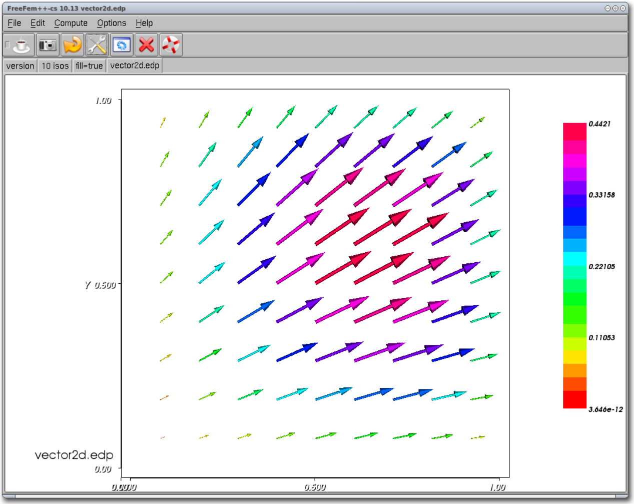
Number of Colors: 20
Number of Iso-Values: 20
Number of Vector Colors: 20
Vector Scaling: 1.0

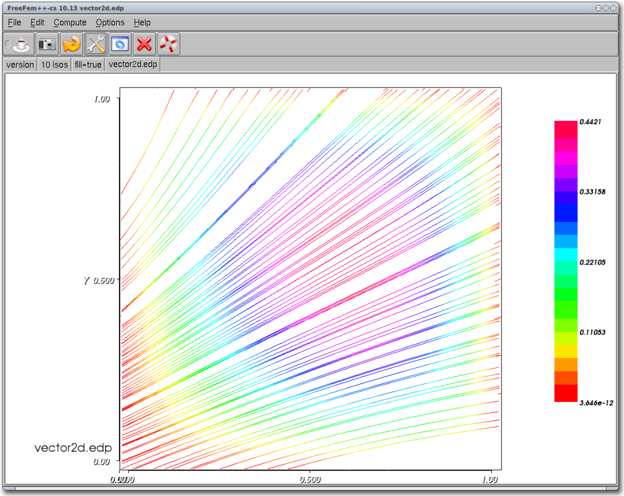
Color Scheme
● HSV
○ RGB (red to blue)
○ RGB (blue to red)
○ Grey

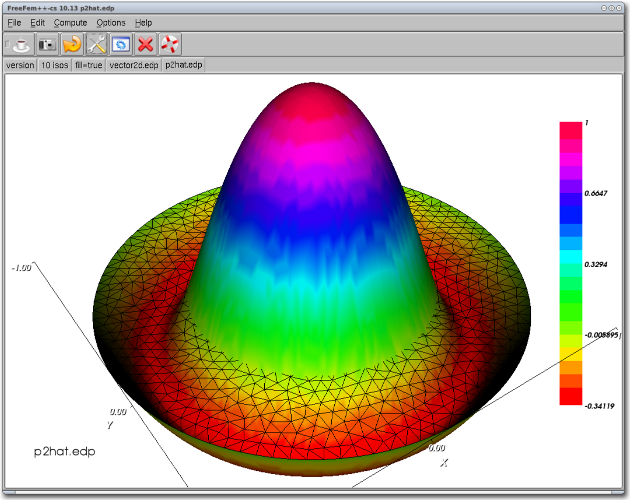
Vector Fields
○ Hedge Hog
● Vectors
○ Stream Lines


FreeFem++-cs 10.13 scalar3d.edp
File
Edit
Compute
Options
Help
version
10 isos
fill=true
0.00
0.500
Z
1.00
0.500
Y
0.00
1.00
1.00
0.500
X
0.00
21
16
11
6
1
10 isos


FreeFem++-cs 10.13 scalar3d.edp
File
Edit
Compute
Options
Help
version
10 isos
fill=true
1.00
Y 0.500
0.00
1.00
1.00
X 0.500
0.00
0.500
Z
0.00
21
16
11
6
1
fill=true


FreeFem++-cs 10.13 vector2d.edp
File
Edit
Compute
Options
Help
version
10 isos
fill=true
vector2d.edp
1.00
Y 0.500
0.00
0.000
0.500
1.00
vector2d.edp
0.4421
0.33158
0.22105
0.11053
3.646e-12


FreeFem++-cs 10.13 vector2d.edp
File
Edit
Compute
Options
Help
version
10 isos
fill=true
vector2d.edp
1.00
Y 0.500
0.00
0.0000
0.500
1.00
vector2d.edp
0.4421
0.33158
0.22105
0.11053
3.646e-12

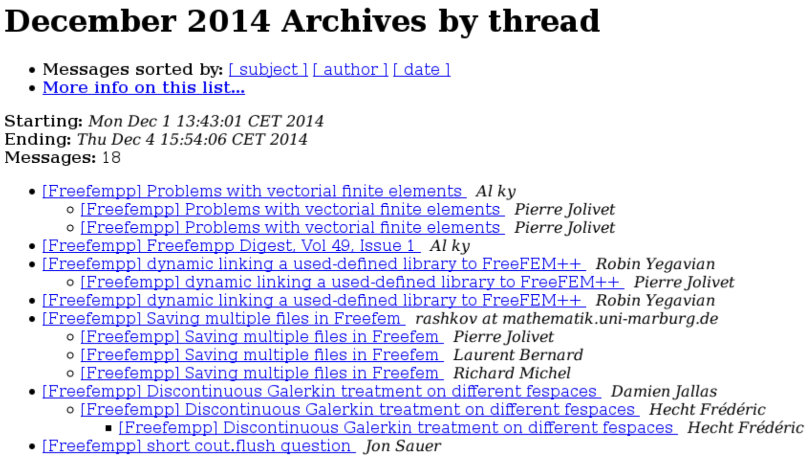

FreeFem++-cs 10.13 p2hat.edp
File
Edit
Compute
Options
Help
version
10 isos
fill=true
vector2d.edp
p2hat.edp
-1.00
0.00
Y
0.00
X
p2hat.edp
1
0.6647
0.3294
-0.005895
-0.34119

The FF documentation (which doubles as the FF book) can be accessed in different ways :

- As a PDF : `www.freefem.org/ff++/ftp/freefem++doc.pdf`
- As a Wiki : `www.um.es/freefemv3/ff++`
- More documents at `www.ljll.math.upmc.fr/lehyaric/ffcs/doc.php`

Asking questions to the community

`www.ljll.math.upmc.fr/cgi-bin/mailman/listinfo/freefempp`



December 2014 Archives by thread

- **Messages sorted by:** [subject] [author] [date]
- **More info on this list...**

Starting: *Mon Dec 1 13:43:01 CET 2014*
Ending: *Thu Dec 4 15:54:06 CET 2014*
Messages: 18

- [Freefempp] Problems with vectorial finite elements *Al ky*
  - [Freefempp] Problems with vectorial finite elements *Pierre Jolivet*
  - [Freefempp] Problems with vectorial finite elements *Pierre Jolivet*
- [Freefempp] Freefempp Digest, Vol 49, Issue 1 *Al ky*
- [Freefempp] dynamic linking a used-defined library to FreeFEM++ *Robin Yegavian*
  - [Freefempp] dynamic linking a used-defined library to FreeFEM++ *Pierre Jolivet*
- [Freefempp] dynamic linking a used-defined library to FreeFEM++ *Robin Yegavian*
- [Freefempp] Saving multiple files in Freefem *rashkov at mathematik.uni-marburg.de*
  - [Freefempp] Saving multiple files in Freefem *Pierre Jolivet*
  - [Freefempp] Saving multiple files in Freefem *Laurent Bernard*
  - [Freefempp] Saving multiple files in Freefem *Richard Michel*
- [Freefempp] Discontinuous Galerkin treatment on different fespaces *Damien Jallas*
  - [Freefempp] Discontinuous Galerkin treatment on different fespaces *Hecht Frédéric*
    - [Freefempp] Discontinuous Galerkin treatment on different fespaces *Hecht Frédéric*
- [Freefempp] short cout.flush question *Jon Sauer*

Here are some 3D graphical tools that may become useful when developing more complex simulations with FF.

- **MEDIT** : `www.um.es/freefemv3/ff++/pmwiki.php?n=FFDoc.LinkWithMedit`
- **VTK output** : very useful to connect to e.g. Paraview. see `examples/examples++-load/iovtk.cpp` or use "File/Save Image As.../VTK" in FFCS.
- **GMSH** : See `www.geuz.org/pipermail/gmsh/2010/005844.html`

The FFCS editor has one big advantage for new users : it highlights mistyped variables in red. Apart from that, it will never have as many features as a complete development editor (Emacs anyone?). FFCS can deal with a program text edited in an external editor :

- "File/Auto Reload" to reload a file into FFCS automatically when it has been changed externally.
- "Compute/Auto Run" to run a newly reloaded program text automatically.
- "Edit/Hide Editor Panel" to put the unused internal editor window out of sight and make more space for the graphics window.

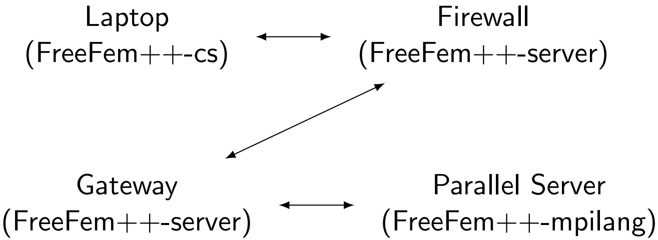
- "load modules" = dynamic library written in C++
- Because of the universal nature of C++, load modules can be used to implement any functionality missing from FF
- (Downside : may become very complex to implement)
- Just download both packages (FF and FFCS) to use the "ff-c++" script

- **Multicores** : supported automatically throught the use of numerical libraries capable of Shared Memory Parallelism (e.g. multithreaded ATLAS/OpenBLAS/MKL/Veclib/...).
- **Distributed Memory Parallelism** (ie several distinct machines working together) : also possible but requires MPI programming.

FFCS does its best to hide this complexity from the user, but often needs to be recompiled to take the user's choice of libraries into account.

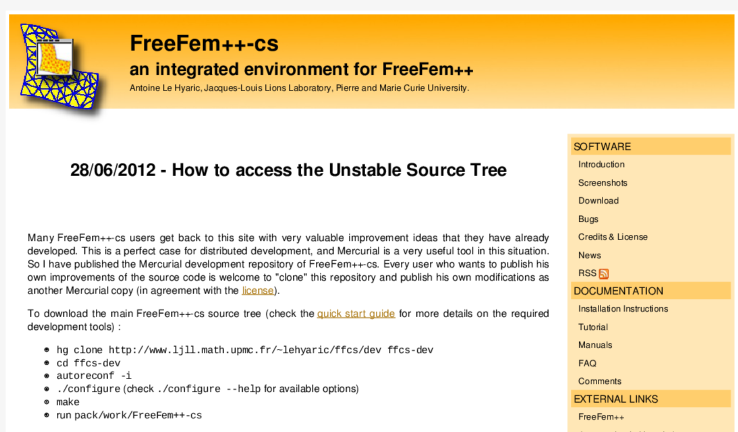
For more details about parallel programming :
`www.um.es/freefemv3/ff++/pmwiki.php?n=FFDoc.MPIParallelVersion`

- created in cooperation with the EJS project `fem.um.es/Ejs`
- good for big runs on parallel machines and local visualisation



Mercurial

`www.ljll.math.upmc.fr/lehyaric/ffcs/news/20120628-1722.php`



FreeFem++-cs

an integrated environment for FreeFem++

Antoine Le Hyaric, Jacques-Louis Lions Laboratory, Pierre and Marie Curie University.

28/06/2012 - How to access the Unstable Source Tree

Many FreeFem++-cs users get back to this site with very valuable improvement ideas that they have already developed. This is a perfect case for distributed development, and Mercurial is a very useful tool in this situation. So I have published the Mercurial development repository of FreeFem++-cs. Every user who wants to publish his own improvements of the source code is welcome to "clone" this repository and publish his own modifications as another Mercurial copy (in agreement with the license).

To download the main FreeFem++-cs source tree (check the quick start guide for more details on the required development tools) :

- `hg clone http://www.ljll.math.upmc.fr/~lehyaric/ffcs/dev ffcs-dev`
- `cd ffcs-dev`
- `autoreconf -i`
- `./configure` (check `./configure --help` for available options)
- `make`
- run `pack/work/FreeFem++-cs`

Mercurial Structure

`ffcs/ff/upstream`
is now ~~cloned~~ copied from
`www.freefem.org/ff++/ff++`

+ sources are still coherent
+ less synchronization work

Developers can still participate in the same way :

- via patches
- via Mercurial clones, published on the web

Mercurial Commands

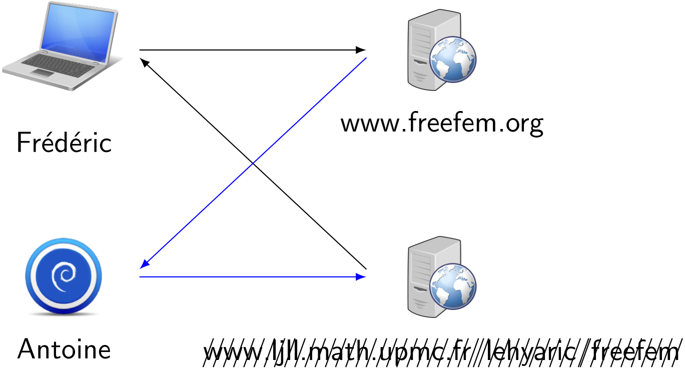
To download the different versions of the source :

- `hg clone http://www.freefem.org/ff++/ff++`
- ~~hg clone https://www.ann.jussieu.fr/˜lehyaric/freefem~~
- `hg clone https://www.ann.jussieu.fr/˜lehyaric/ffcs/dev`

To select the right branch (“default”,”alh-dev”,”draft”) :

- `hg update [branch]`

Physical location of FreeFem++ source code



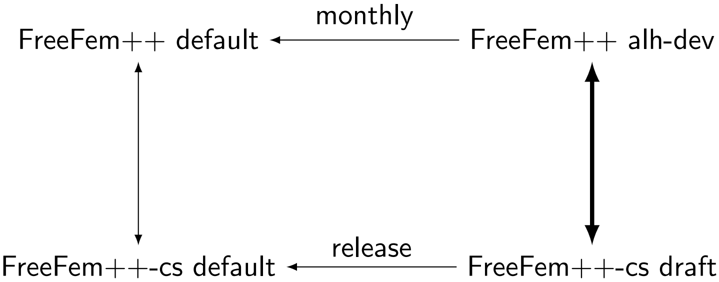
Frédéric

www.freefem.org

Antoine

~~www.ljll.math.upmc.fr/lehyaric/freefem~~

Compatibility between Mercurial branches



FreeFem++ default

monthly

FreeFem++ alh-dev

FreeFem++-cs default

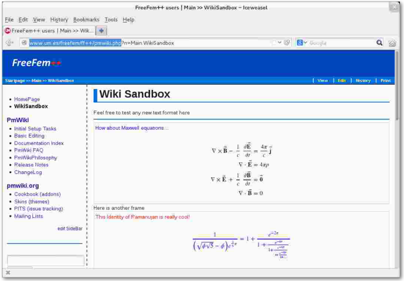
release

FreeFem++-cs draft

Other FFCS changes in 2014

- FFCS
  - VTK 6.1
  - Variable tab size in editor
  - 7 releases this year so far
  - Keeping up with MacOS
- Common FF source
  - cross-references (file index.org)
  - GSL module load

- FreeFem++ full documentation on Eliseo Chacon Vera's Wiki : DONE



Wiki Sandbox

Feel free to test any new text format here

How about Maxwell equations ...

$$\nabla \times \vec{\mathbf{B}} - \frac{1}{c}\frac{\partial \vec{\mathbf{E}}}{\partial t} = \frac{4\pi}{c}\vec{\mathbf{j}}$$

$$\nabla \cdot \vec{\mathbf{E}} = 4\pi\rho$$

$$\nabla \times \vec{\mathbf{E}} + \frac{1}{c}\frac{\partial \vec{\mathbf{B}}}{\partial t} = \vec{\mathbf{0}}$$

$$\nabla \cdot \vec{\mathbf{B}} = 0$$

Here is another frame

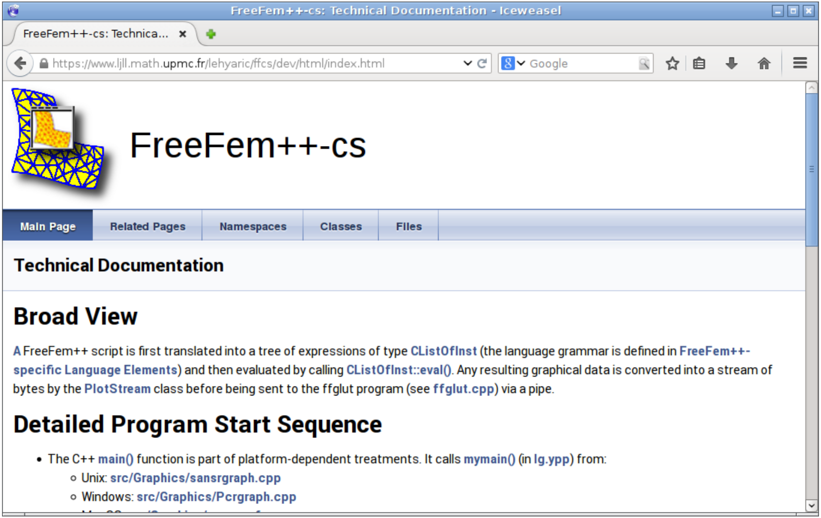
This Identity of Ramanujan is really cool!

$$\frac{1}{\left(\sqrt{\phi\sqrt{5}} - \phi\right)e^{\frac{2}{5}\pi}} = 1 + \frac{e^{-2\pi}}{1 + \frac{e^{-4\pi}}{1 + \frac{e^{-6\pi}}{1 + \frac{e^{-8\pi}}{1 + \cdots}}}}$$

`www.um.es/freefem/ff++/pmwiki.php`

- Debugger
  - opening up the C++ classes through Doxygen documentation : structure DONE
    `www.ljll.math.upmc.fr/lehyaric/freefem/html`
    `www.ljll.math.upmc.fr/lehyaric/ffcs/dev/html`
  - Emacs “org-mode” documentation : index DONE
    `(require 'org-install)`
    `(global-set-key "\C-co" 'org-open-at-point-global)`
    links : `[[file:/a/b/c/d.cpp::name]]` targets : `<<name>>`
  - connection through GUI
  - connection through other languages

www.ljll.math.upmc.fr/lehyaric/ffcs/dev/html/index.html


FreeFem++-cs: Technical Documentation - Iceweasel
FreeFem++-cs: Technica...
https://www.ljll.math.upmc.fr/lehyaric/ffcs/dev/html/index.html
Google
FreeFem++-cs
Main Page
Related Pages
Namespaces
Classes
Files

Technical Documentation

Broad View

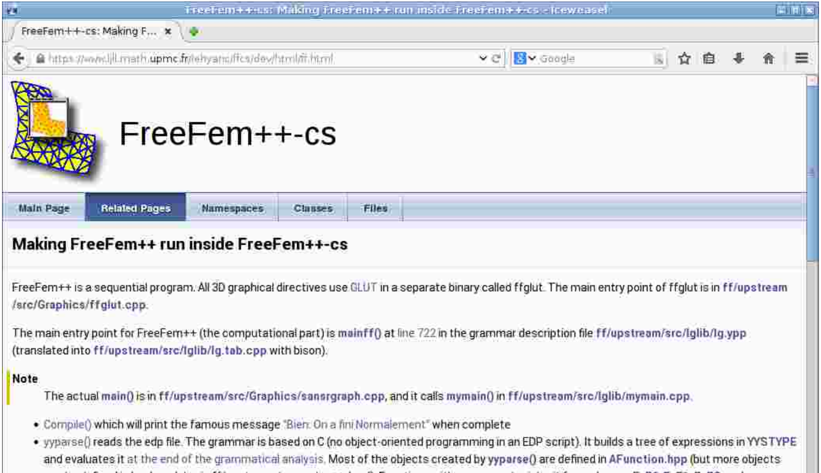
A FreeFem++ script is first translated into a tree of expressions of type CListOfInst (the language grammar is defined in FreeFem++-specific Language Elements) and then evaluated by calling CListOfInst::eval(). Any resulting graphical data is converted into a stream of bytes by the PlotStream class before being sent to the ffglut program (see ffglut.cpp) via a pipe.

Detailed Program Start Sequence

- The C++ main() function is part of platform-dependent treatments. It calls mymain() (in lg.ypp) from:
  - Unix: src/Graphics/sansrgraph.cpp
  - Windows: src/Graphics/Pcrgraph.cpp

Connecting FreeFem++-cs to FreeFem++

`www.ljll.math.upmc.fr/lehyaric/ffcs/dev/html/ff.html`


FreeFem++-cs: Making FreeFem++ run inside FreeFem++-cs - Iceweasel
FreeFem++-cs: Making F...
https://www.ljll.math.upmc.fr/lehyaric/ffcs/dev/html/ff.html
Google
FreeFem++-cs
Main Page
Related Pages
Namespaces
Classes
Files

Making FreeFem++ run inside FreeFem++-cs

FreeFem++ is a sequential program. All 3D graphical directives use GLUT in a separate binary called ffglut. The main entry point of ffglut is in ff/upstream/src/Graphics/ffglut.cpp.

The main entry point for FreeFem++ (the computational part) is mainff() at line 722 in the grammar description file ff/upstream/src/lglib/lg.ypp (translated into ff/upstream/src/lglib/lg.tab.cpp with bison).

Note

The actual main() is in ff/upstream/src/Graphics/sansrgraph.cpp, and it calls mymain() in ff/upstream/src/lglib/mymain.cpp.

- Compile() which will print the famous message "Bien: On a fini Normalement" when complete
- yyparse() reads the edp file. The grammar is based on C (no object-oriented programming in an EDP script). It builds a tree of expressions in YYSTYPE and evaluates it at the end of the grammatical analysis. Most of the objects created by yyparse() are defined in AFunction.hpp (but more objects

`www.ljll.math.upmc.fr/lehyaric/ffcs/dev/html/...`

- VTK
  - Visualization `.../visu.html`
  - Dashboard `.../classDashboard.html`
  - Visucontrol `.../classVisucontrol.html`
  - Visudata `.../classVisudata.html`
- Documentation `.../doc.html`
  - End-users `.../doc.html#end-users`
  - ~~Reference card~~ (replaced with wiki)
  - Developers `.../doc.html#developers`
  - Source tree `.../files.html`

More pointers to the doc

`www.ljll.math.upmc.fr/lehyaric/ffcs/dev/html/...`

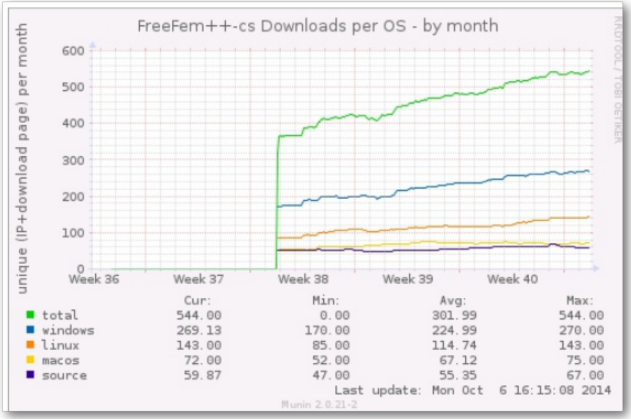
- Packages `.../porting.html`
- Testing `.../testing.html`
- Specifying reference values for testing `.../mkffref_source.html`
- Parallelism `.../parallelism.html`

Changing the GUI framework

FF and FFCS are mature codes, we now need to improve the looks and the code structure

- Optimize portability (Linux, MacOS, Windows)
- Long-term design choice : GTK? Qt? plain Java? Java with Emscripten? WxWindows?
- MacOS catch-up game
- Connect with course material (eg in Latex or HTML)
- Solve all licence issues

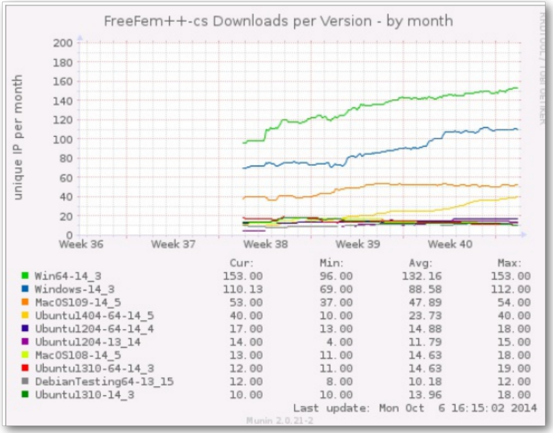
Stats per OS



	Cur:	Min:	Avg:	Max:
🟩 total	544.00	0.00	301.99	544.00
🟦 windows	269.13	170.00	224.99	270.00
🟧 linux	143.00	85.00	114.74	143.00
🟨 macos	72.00	52.00	67.12	75.00
🟪 source	59.87	47.00	55.35	67.00

Last update: Mon Oct 6 16:15:08 2014

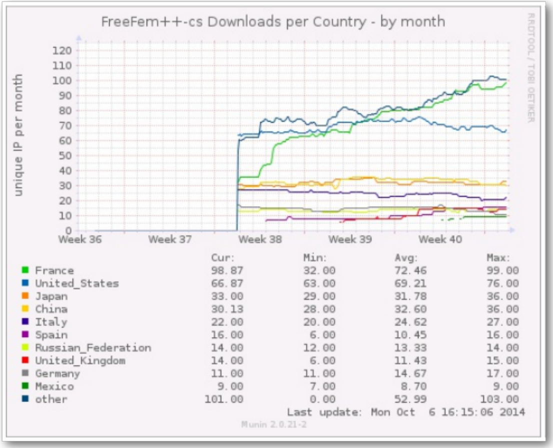
Stats per version



	Cur:	Min:	Avg:	Max:
🟩 Win64-14_3	153.00	96.00	132.16	153.00
🟦 Windows-14_3	110.13	69.00	88.58	112.00
🟧 MacOS109-14_5	53.00	37.00	47.89	54.00
🟨 Ubuntu1404-64-14_5	40.00	10.00	23.73	40.00
🟦 Ubuntu1204-64-14_4	17.00	13.00	14.88	18.00
🟪 Ubuntu1204-13_14	14.00	4.00	11.79	15.00
🟩 MacOS108-14_5	13.00	11.00	14.63	18.00
🟥 Ubuntu1310-64-14_3	12.00	11.00	14.63	19.00
■ DebianTesting64-13_15	12.00	8.00	10.18	12.00
🟩 Ubuntu1310-14_3	10.00	10.00	13.96	18.00

Last update: Mon Oct 6 16:15:02 2014

Stats per country



	Cur:	Min:	Avg:	Max:
🟩 France	98.87	32.00	72.46	99.00
🟦 United_States	66.87	63.00	69.21	76.00
🟧 Japan	33.00	29.00	31.78	36.00
🟨 China	30.13	28.00	32.60	36.00
🟪 Italy	22.00	20.00	24.62	27.00
🟪 Spain	16.00	6.00	10.45	16.00
🟨 Russian_Federation	14.00	12.00	13.33	14.00
🟥 United_Kingdom	14.00	6.00	11.43	15.00
⬛ Germany	11.00	11.00	14.67	17.00
🟩 Mexico	9.00	7.00	8.70	9.00
🟦 other	101.00	0.00	52.99	103.00

Last update: Mon Oct 6 16:15:06 2014

Munin 2.0.21-2

Through the years, the selling point of FFCS has been to let users focus on the mathematical questions. The single-window approach attracts new users to FF. We will try and keep this philosophy in the future.

- Do not hesitate to comment. The mailing list is at : `ljll.math.upmc.fr/cgi-bin/mailman/listinfo/freefempp`
- For the afternoon tutorials, download early to avoid network congestion

To find FreeFem++-cs on the net : just search for “freefem cs”

And thank you for all your messages.