Castor, un environnement C++ pour le calcul matriciel

Matthieu AUSSAL, CMAP, DEFI, Ecole polytechnique - Palaiseau

Marc BAKRY, CEA List - Gif-sur-Yvette

Laurent SERIES, CMAP, Ecole polytechnique - Palaiseau

21 juin 2021 - Congrès Biennale de la SMAI

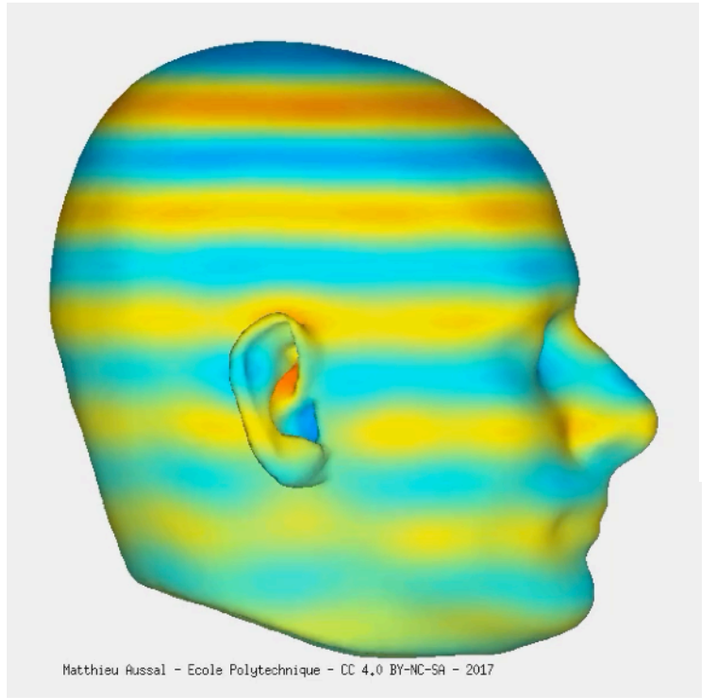
Prequel : Matlab

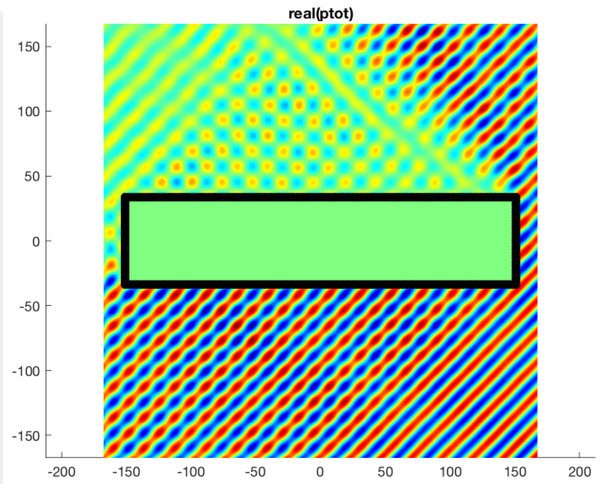
- Elements finis (FEM)
- Equations intégrales (BEM)
- Lancer de rayons
- Couplage multi-physique
- Problèmes inverses
- etc.

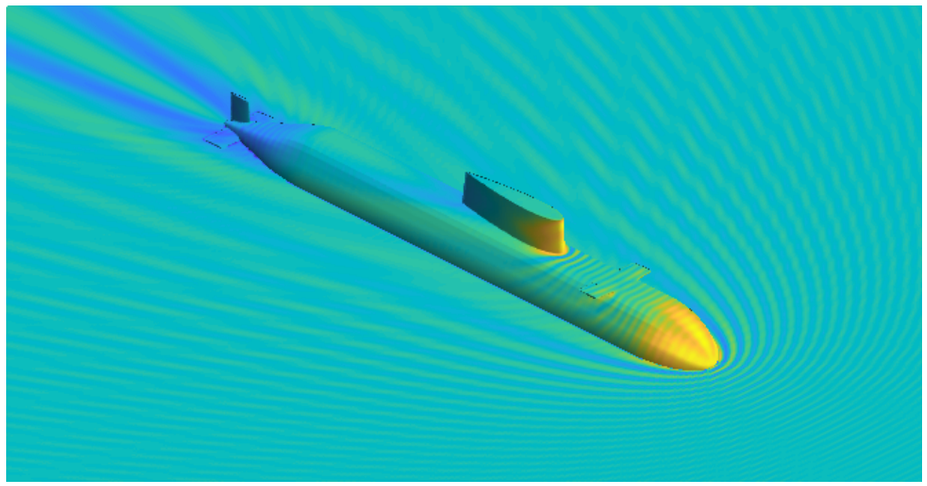
Protypage et performances !

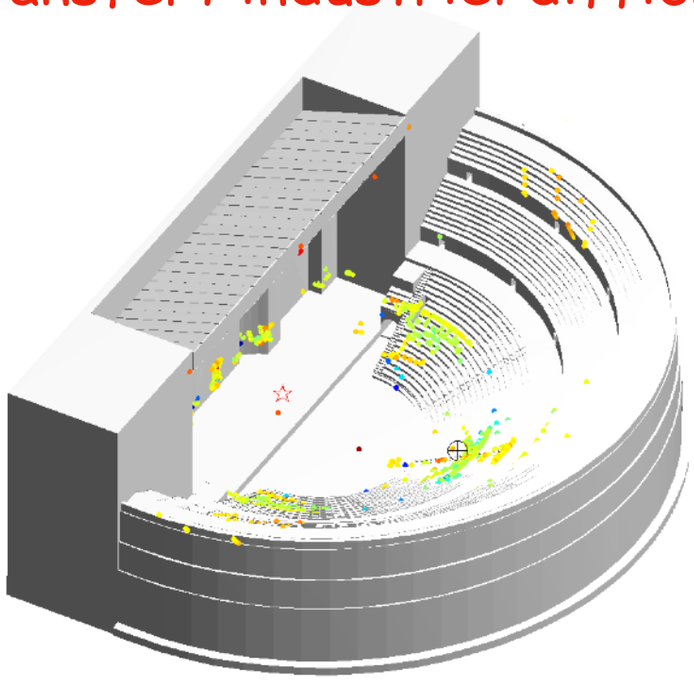
Mais :

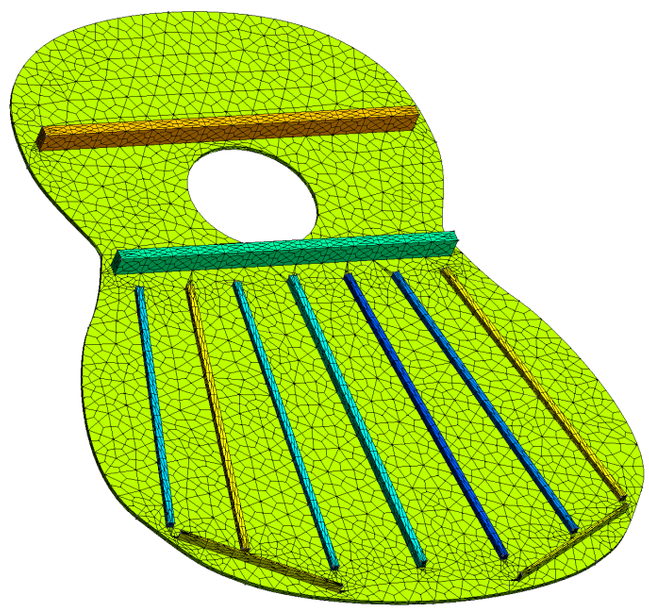
- Licence chère et contraignante
- Transfert industriel difficile…

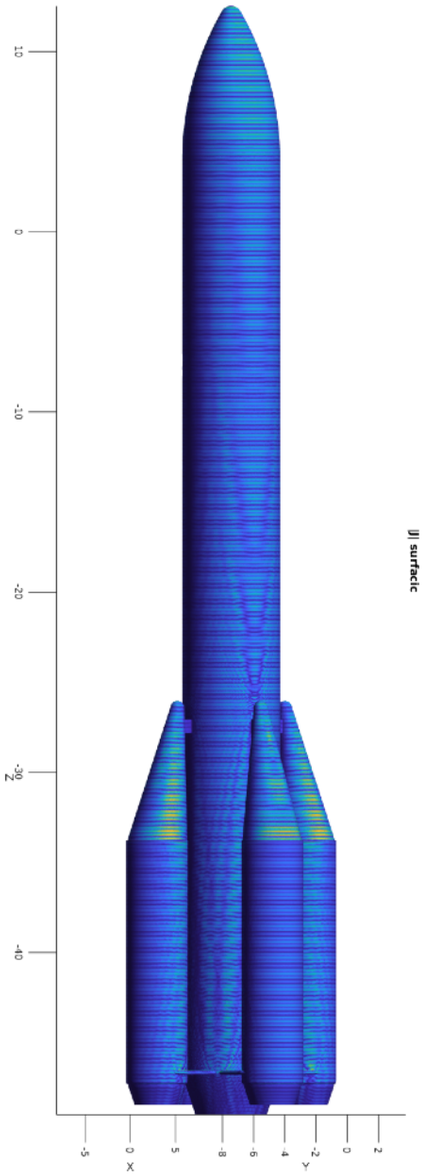




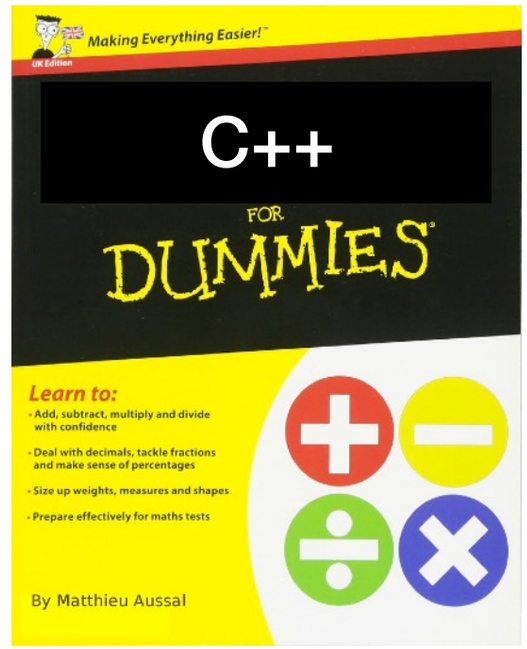








Le projet Castor



Objectif :
Interface simple pour rendre **plus accessible la librairie standard C++.**

Contenu :
- Format de stockage plein, creux et hiérarchique,
- Algèbre linéaire (BLAS/LAPACK),
- Visualisation graphique (VTK),
- Fast Fourier Transform (FFTW/KISSFFT)

Licence :
Open source (LGPL3.0).

Services :
Librairies, documentation, exemples et codes de démonstrations en ligne.
Tests unitaires et intégration continue.

Philosophie :
Sémantique « à la matlab », pour le **prototypage ET l'industrialisation** de codes de calcul et logiciels.

Positionnement : (vs Matlab, Python, Julia, R ou encore Xtensor, Eigen, etc.)
Aucun. A utiliser au besoin, ou pas.

Exemple 1

```cpp
#include "castor/matrix.hpp"

using namespace castor;

int main (int argc, char* argv[])
{
    // Calcul vectorise (a la Matlab)
    matrix<> X,Y;
    X = linspace(-1,1,1e5);
    tic();
    Y = X + 1 + 1 + 1 + 1 + 1;
    toc();

    // Calcul devectorise (a la C)
    tic();
    Y = zeros(size(X));
    for (std::size_t i=0; i<numel(X); ++i)
    {
        Y(i) = X(i) + 1 + 1 + 1 + 1 + 1;
    }
    toc();

    disp("done !");
    return 0;
}
```

```
$ ./test
Elapsed time is 0.00174831 seconds.
Elapsed time is 0.00015171 seconds.
done !
```

Castor est 10x plus lent...

6 copies VS 2 copies

Exemple 1 (correction)

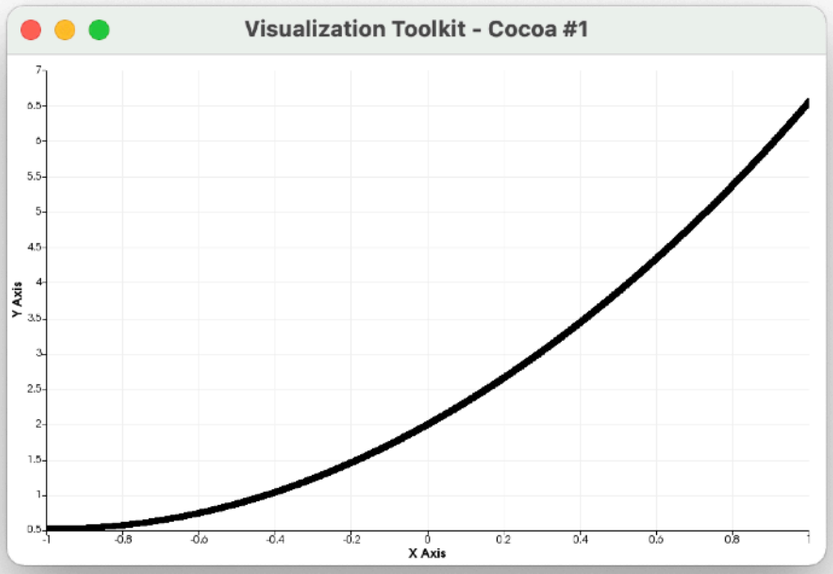
```
#include "castor/matrix.hpp"

using namespace castor;

int main (int argc, char* argv[])
{
    // Calcul vectorise (a la Matlab)
    matrix<> X,Y;
    X = linspace(-1,1,1e5);
    tic();
    Y = X + 5;
    toc();

    // Calcul devectorise (a la C)
    tic();
    Y = zeros(size(X));
    for (std::size_t i=0; i<numel(X); ++i)
    {
        Y(i) = X(i) + 5;
    }
    toc();

    disp("done !");
    return 0;
}
```

```
$ ./test
Elapsed time is 0.000386489 seconds.
Elapsed time is 0.000374775 seconds.
done !
```

Egalité...

2 copies VS 2 copies

Exemple 2



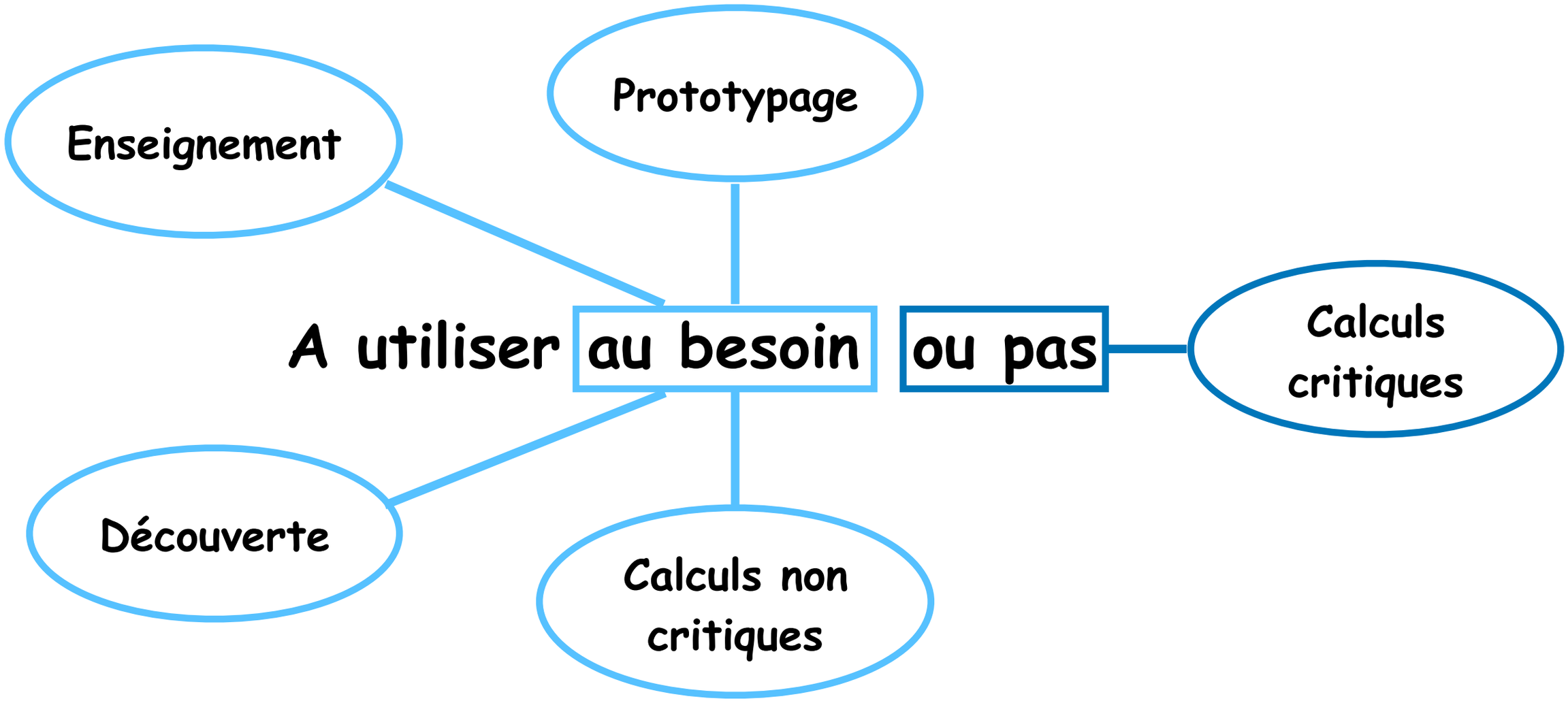

```cpp
#include "castor/matrix.hpp"
#include "castor/graphics.hpp"

using namespace castor;

int main (int argc, char* argv[])
{
    // Calcul vectorise (a la Matlab)
    matrix<> X,Y;
    X = linspace(-1,1,1e5);
    tic();
    Y = 1 + X + X*X + sin(X) + exp(X);
    toc();

    // Calcul devectorise (a la C)
    tic();
    Y = zeros(size(X));
    for (std::size_t i=0; i<numel(X); ++i)
    {
        Y(i) = 1 + X(i) + X(i)*X(i) + std::sin(X(i)) + std::exp(X(i));
    }
    toc();

    // Rendu graphique
    figure fig;
    plot(fig,X,Y);
    drawnow(fig);

    disp("done !");
    return 0;
}
```

```
$ ./test
Elapsed time is 0.00573435 seconds.
Elapsed time is 0.00350536 seconds.
done !
```

Castor est 2x plus lent...

Le temps de calcul masque
une partie du temps de copie

Le projet Castor



En détails

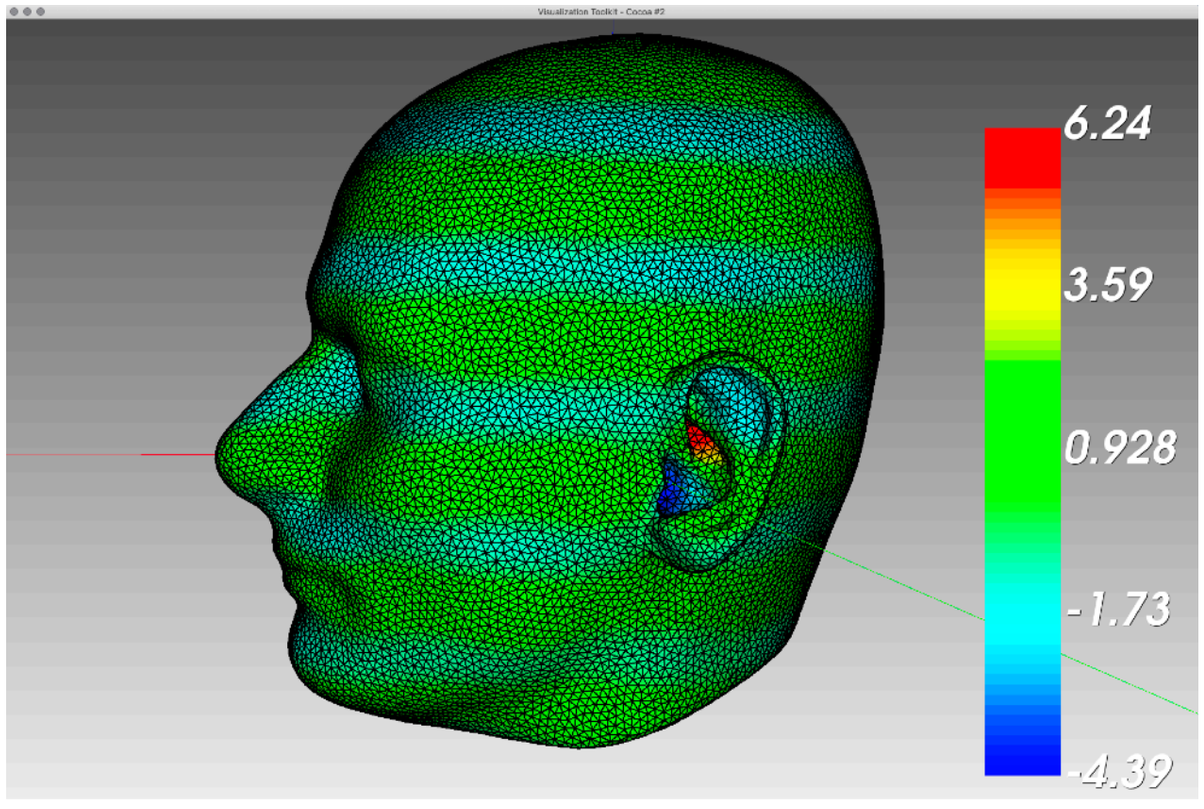
Noyau librairie Castor :

matrix.hpp	Matrice avec stockage plein
linalg.hpp	Interface BLAS/LAPACK pour l'algèbre linéaire
graphics.hpp	Visualisation graphique 2D/3D (VTK)
smatrix.hpp	Matrice avec stockage creux (sparse)
hmatrix.hpp	Matrice avec stockage hiérarchique et algèbre linéaire
kissfft.hpp	Transformée de Fourier rapide (1D, single precision)

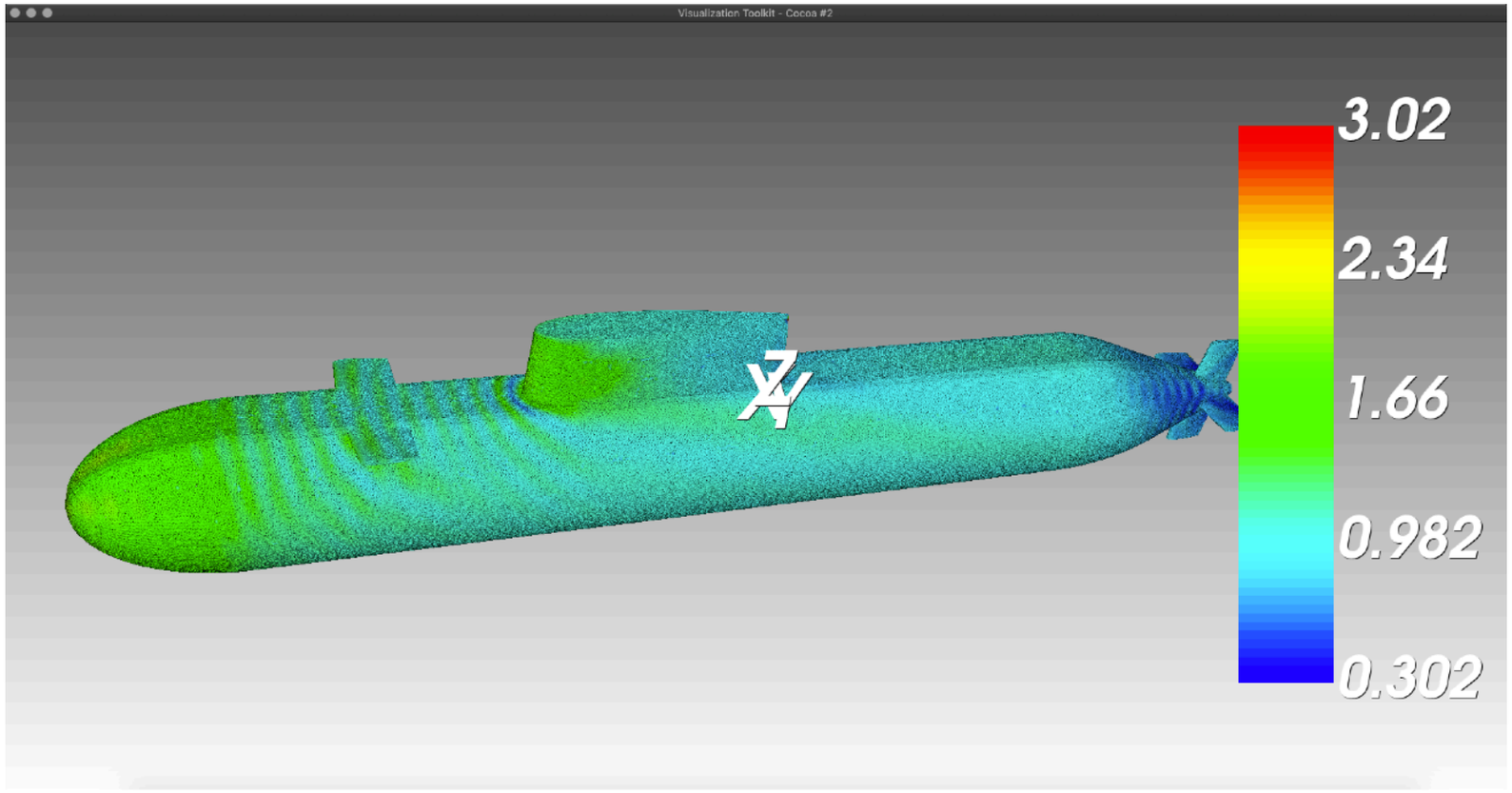
Extension du projet Castor :

FEM / BEM	Element finis (P0/P1 2D) et équations intégrales (Helmholtz 3D)
Analytical Scattering	Solutions analytique pour problèmes de diffraction
FFTW	Interface castor pour la librairie FFTW

Equations Intégrales

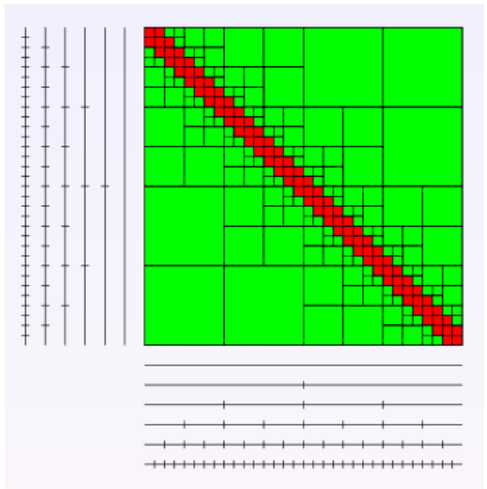


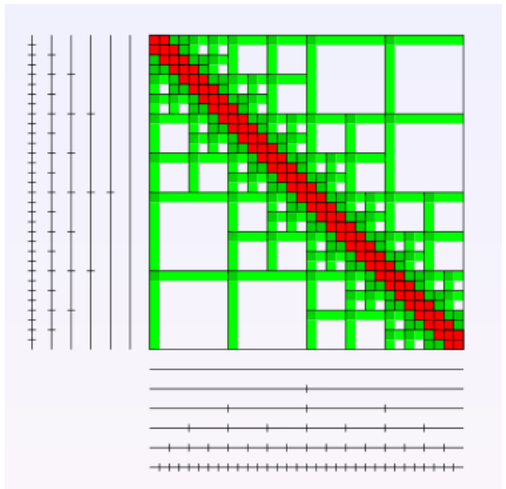
Résolution d'un problème de diffraction acoustique par méthodes intégrales



Matrices Hiérarchiques

Intégration en cours avec esi get it right®





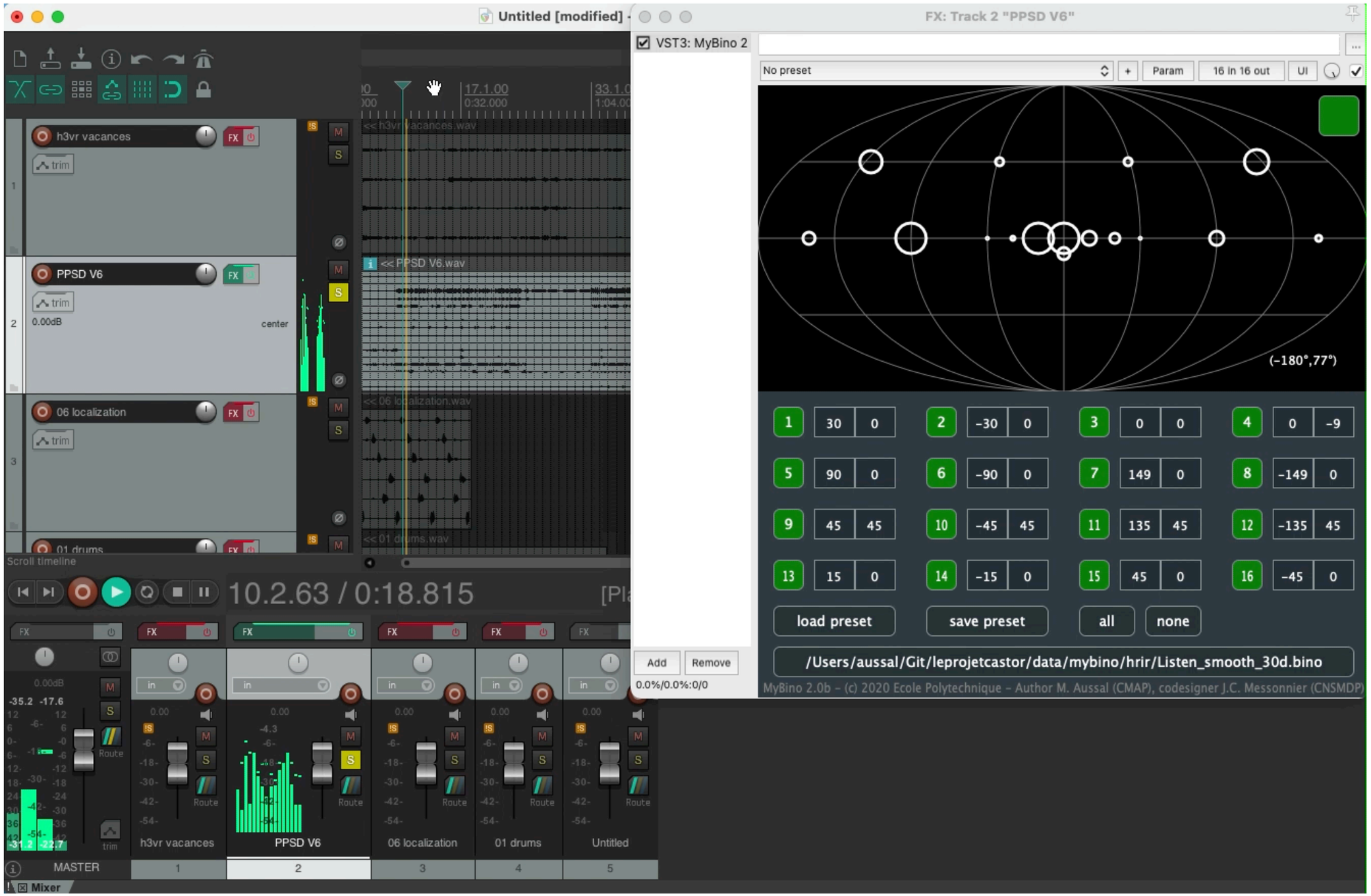
Compression de matrices pleines
Algèbre hiérarchique
Parallelisme OpenMP

Frequency: 100Hz , Formulation: \int \int v(x) G(x,y) u(y) dx dy = -\int pw(x) v(x) dx #THREADS: 36						
1E+03	0.14	70.70	1.79e-4	0.14	36.48	1.88e-4
3E+03	0.59	37.43	2.90e-4	0.86	18.08	2.94e-4
1E+04	2.63	12.78	5.29e-4	4.83	5.30	5.12e-4
3E+04	10.21	4.72	3.98e-4	22.55	1.71	4.01e-4
1E+05	44.34	1.95	4.27e-3	105.51	0.75	4.31e-4
3E+05	170.97	0.67	4.55e-4	428.61	0.23	4.72e-4
1E+06	724.94	0.22	3.86-4	1975.04	0.07	4.16e-4

Frequency: 100Hz , Formulation: \int \int v(x) G(x,y) u(y) dx dy = -\int pw(x) v(x) dx scalability							
Build time(s)	**#THREADS:1**	**#THREADS:2**	**#THREADS: 4**	**#THREADS: 8**	**#THREADS: 16**	**#THREADS: 32**	**#THREADS: 36**
1E+04	91.35	45.83	22.97	11.57	5.83	2.94	2.63
3E+04	357.44	178.92	89.99	45.45	22.80	11.44	10.21
1E+05	1541.92	763.40	383.39	192.62	97.01	49.30	44.34
3E+05	5870.76	2926.23	1473.91	744.295	376.04	190.26	170.97
1E+06	24563.9	12482.6	6304.52	3190.53	1617.66	810.96	724.94

Monitoring binaural (son 3D)

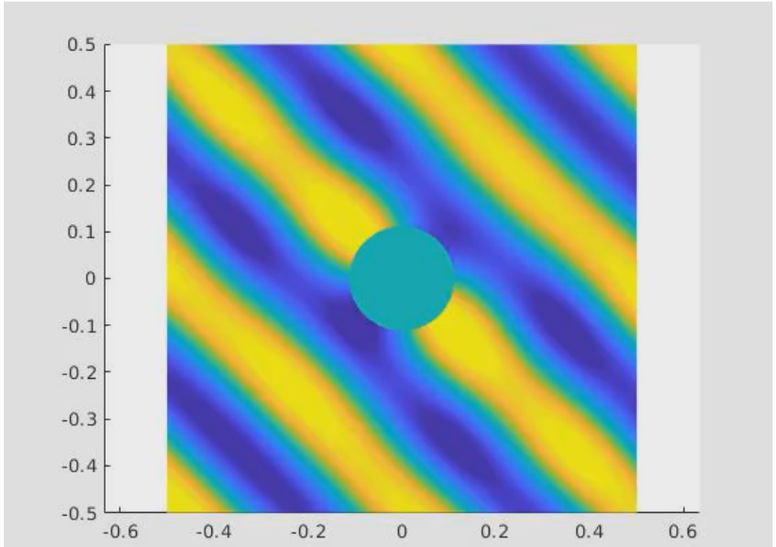
Collaboration et déploiement avec CONSERVATOIRE NATIONAL SUPÉRIEUR DE MUSIQUE ET DE DANSE DE PARIS



Ecole à distance

Collaboration et intégration avec

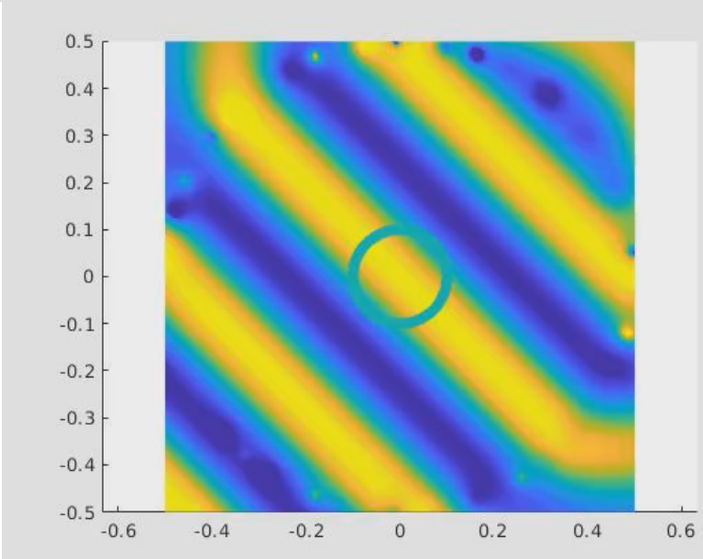
Captation



A-Format

Binaural 3D

Restitution



Entraînement de Jedi

Evaluation en cours avec

Venez l'essayer! :-)

Le projet Castor

http://leprojetcastor.gitlab.labos.polytechnique.fr/castor/

Welcome to the castor library documentation

The objective of the **castor** library is to propose **high-level semantics**, inspired by the Matlab language, allowing fast software prototyping in a low-level compiled language. It is nothing more than a **matrix management layer** using the tools of the standard **C++** library, in different storage formats (full, sparse and hierarchical). Indeed, the use of IDEs 1 such as Xcode, Visual studio, Eclipse, etc. allows today to execute compiled code (C, C++, fortran, etc.) with the **same flexibility as interpreted languages** (Matlab, Python, Julia, etc.).

A **header-only** template library for matrix management has been developed based on the standard C++ library, notably the std::vector class. Many tools and algorithms are provided to simplify the development of scientific computing programs. Particular attention has been paid to semantics, for a simplicity of use "à la matlab", but written in C++. This high-level semantic/low-level language coupling makes it possible to gain efficiency in the prototyping phase, while ensuring performance for applications. In addition, direct access to data allows users to optimize the most critical parts of their code in native C++. Finally, **complete documentation** is available, as well as continuous integration unit tests. All of this makes it possible to meet the needs of teaching, academic issues and industrial applications at the same time.

The **castor** library provides tools to :

- create and manipulate dense, sparse and hierarchical matrices
- make linear algebra computations based on optimized BLAS library
- make graphical representations based on VTK library

These tools are used by applicative projects :

- finite and boundary element method using Galerkin approximation
- analytical solutions for scattering problems

The source files of the library is available here : https://gitlab.labos.polytechnique.fr/leprojetcastor/castor.

As the semantics offered by **castor** library being voluntarily close to the Matlab environment, there are functions signature and their documentation inspired by it. You can refer to https://fr.mathworks.com/help/matlab/index.html.

Licensing

The **castor** library is provided in open source under LGPL 3.0.

This program is free software, distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY. Natively, you can use, redistribute and/or modify it under the terms of the GNU Lesser General Public License, as published by the Free Software Foundation (version 3 or later, http://www.gnu.org/licenses).

Gallery