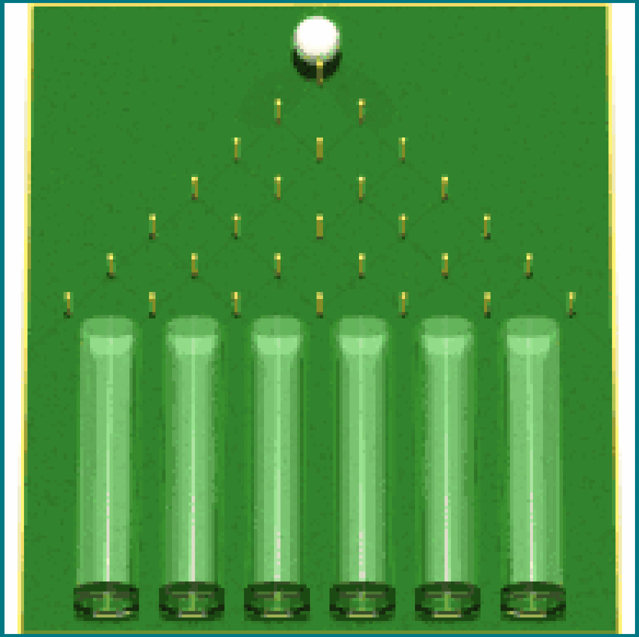
OpenTURNS Users' Day #11



Friday, the 15 th, June 2018

❖The Galton Board (1822 - 1911) : british explorer, geograph, inventor, meteorologist, proto-geneticist, psychometrician,statistician

OpenTURNS User's Day # 11

9h - 10h →

- ➢ **Reception at EDF Lab Saclay**

10h – 11h

- ➢ **Welcome speech**: **E. Duceau (Scientific Advisor Airbus)**
- ➢ **Open TURNS**
  - ✓ General features
  - ✓ New features of the 1.10 & 1.11 releases
  - ✓ On going & future developments

11h – 12h →

- ➢ **«Graph approach for high dimensional probabilistic model – otagrum module»: PH. Wuillemin (Sorbonne University, LIP6) – R. Lebrun (Airbus)**

Lunch (12h – 13h30)

13h30 – 15h30

- ➢ **Studies using Open TURNS** (1/2)
  - ✓ **« Anomaly detection in aircraft sensor data »: N. Huet (IMACS), J. Sen Gupta (Airbus)**
  - ✓ **« ANATOLE: 3D tolerancing and sensitivity analysis »: B. Chaigne (IMACS)**
  - ✓ **« New features of the GUI of OpenTURNS: system modelisations, dependence, validation»: M. Baudin (EDF), J. Schueller (Phimeca), A. Ladier (Phimeca)**
  - ✓ **« Marges for the conception of complex systems »: A. Touboul (IRT)**

Break (15h30 – 16h)

16h – 17h

- ➢ **Studies using Open TURNS** (2/2)
  - ✓ **« Estimation of penstock pipes reliability using system reliability methods»: E. Ardillon (EDF), A. Dumas (Phimeca)**
  - ✓ **« Modelisation of the maximum annual magnitude of earthquakes with extreme value distributions»: A. Dutfoy (EDF)**

Welcome speech

Eric Duceau

(Scientific Advisor Airbus)

What is OpenTURNS?

OpenTURNS

An Open source initiative for the Treatment of Uncertainties, Risks'N Statistics

- Multivariate probabilistic modeling including dependence
- Numerical tools dedicated to the treatment of uncertainties
- Generic coupling to any type of physical model
- Open source, LGPL licensed, C++/Python library

www.openturns.org

- **Linux, Windows**
- **First version launched in 2007**

- **4 full time software developers**
- **Project size: 720 classes**

- **Easy to use in Python world (Numpy, Scipy, Matplotlib)**
- **Algorithm parallelisation (TBB)**

eDF AIRBUS IMACS PHIMECA

OpenTURNS : content

Data analysis

Visual analysis: QQ-Plot, Cobweb
Fitting tests: Kolmogorov, Chi2
Multivariate distribution: kernel smoothing (KDE), maximum likelihood
Process: covariance models, Welch and Whittle estimators
Bayesian calibration: Metropolis-Hastings, conditional distribution

Probabilistic modeling

Dependence modelling: elliptical, archimedian copulas.
Univariate distribution: Normal, Weibull
Multivariate distribution: Student, Dirichlet, Multinomial, User-defined
Process: Gaussian, ARMA, Random walk.
Covariance models: Matern, Exponential, User-defined

Meta modeling

Functional basis methods: orthogonal basis (polynomials, Fourier, Haar, Soize Ghanem)
Gaussian process regression: General linear model (GLM), Kriging
Spectral methods: functional chaos (PCE), Karhunen-Loeve, low-rank tensors

Reliability, sensitivity

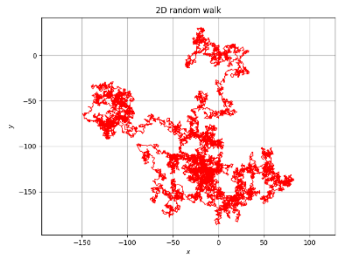
Sampling methods: Monte Carlo, LHS, low discrepancy sequences
Variance reduction methods: importance sampling, subset sampling
Approximation methods: FORM, SORM
Indices: Spearman, Sobol, ANCOVA
Importance factors: perturbation method, FORM, Monte Carlo

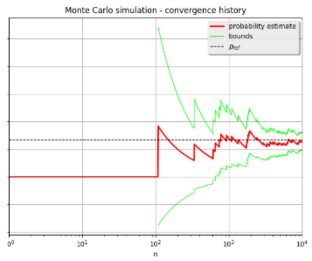
Functional modeling

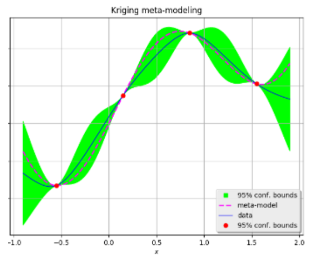
Numerical functions: symbolic, Python-defined, user-defined
Function operators: addition, product, composition, gradients
Function transformation: linear combination, aggregation, parametrization
Polynomials: orthogonal polynomial, algebra

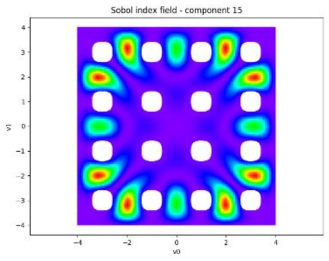
Numerical methods

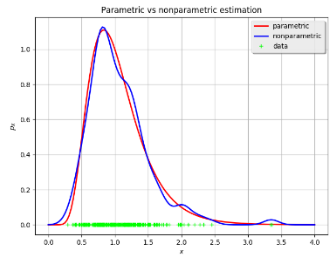
Integration: Gauss-Kronrod
Optimization: NLopt, Cobyla, TNC
Root finding: Brent, Bisection
Linear algrebra: Matrix, HMat
Interpolation: piecewise linear, piecewise Hermite
Least squares: SVD, QR, Cholesky



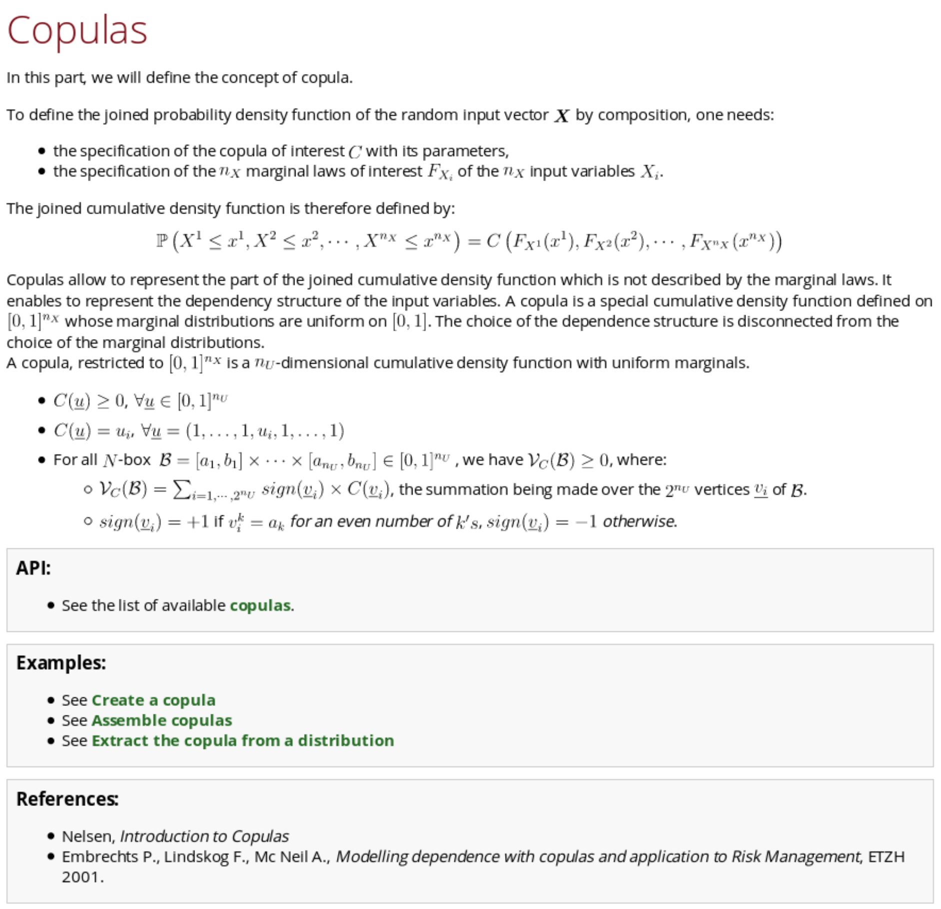








OpenTURNS : Documentation



Copulas

In this part, we will define the concept of copula.

To define the joined probability density function of the random input vector $\boldsymbol{X}$ by composition, one needs:

- the specification of the copula of interest C with its parameters,
- the specification of the n_X marginal laws of interest F_{X_i} of the n_X input variables X_i.

The joined cumulative density function is therefore defined by:

$$\mathbb{P}\left(X^1 \leq x^1, X^2 \leq x^2, \cdots, X^{n_X} \leq x^{n_X}\right) = C\left(F_{X^1}(x^1), F_{X^2}(x^2), \cdots, F_{X^{n_X}}(x^{n_X})\right)$$

Copulas allow to represent the part of the joined cumulative density function which is not described by the marginal laws. It enables to represent the dependency structure of the input variables. A copula is a special cumulative density function defined on $[0,1]^{n_X}$ whose marginal distributions are uniform on $[0,1]$. The choice of the dependence structure is disconnected from the choice of the marginal distributions.
A copula, restricted to $[0,1]^{n_X}$ is a n_U-dimensional cumulative density function with uniform marginals.

- $C(\underline{u}) \geq 0$, $\forall \underline{u} \in [0,1]^{n_U}$
- $C(\underline{u}) = u_i$, $\forall \underline{u} = (1, \ldots, 1, u_i, 1, \ldots, 1)$
- For all N-box $\mathcal{B} = [a_1, b_1] \times \cdots \times [a_{n_U}, b_{n_U}] \in [0,1]^{n_U}$, we have $\mathcal{V}_C(\mathcal{B}) \geq 0$, where:
  - $\mathcal{V}_C(\mathcal{B}) = \sum_{i=1,\cdots,2^{n_U}} sign(\underline{v}_i) \times C(\underline{v}_i)$, the summation being made over the 2^{n_U} vertices $\underline{v_i}$ of $\mathcal{B}$.
  - $sign(\underline{v}_i) = +1$ if $v_i^k = a_k$ *for an even number of* $k's$, $sign(\underline{v}_i) = -1$ *otherwise.*

API:

- See the list of available **copulas**.

Examples:

- See **Create a copula**
- See **Assemble copulas**
- See **Extract the copula from a distribution**

References:

- Nelsen, *Introduction to Copulas*
- Embrechts P., Lindskog F., Mc Neil A., *Modelling dependence with copulas and application to Risk Management*, ETZH 2001.

Reference Guide: Copula

➢ **Several documentations for several goals**

➔ Mathematics**: Reference Guide**

➔The way to use it**: API**

➔Some script examples: **Examples (basic & advanced)**

➔The library architecture: **Developer's Guide**

➢ **The documentation wants to be ….**:

➔ as exhaustive as possible

➔ illustrated : with access to the script used to produce the graph

➔ a link between different needs: crossed references between classes & ref doc

➔an help providing validated examples (part of the tests catalog)

Open TURNS – Trainings

- **EDF** ❖ **contact : Corine TRIPET: 01 78 19 40 32**

✓ « Uncertainty Management : Open TURNS »
  - ✓ 3 days to learn the use of Open TURNS : python TUI, wrapping aspects, application of the methodologie from step A to step C & C'
  - ✓ next session : **10-12/09/2018**, ITECH
  - ✓ EDF Lab Paris-Saclay

« Uncertainty Management : Methodology »
  - ✓ 3 days to learn the Uncertainty Methodology
  - ✓ next session : **3-5/09/2018**, ITECH
  - ✓ EDF Lab Chatou

- **PhiMECA : 2 sessions each year for each modulus**

www.phimeca.com/formations **contact : Antoine DUMAS : 04 43 86 88 95**

- ✓ Probability & Statistics: 2 days
- ✓ Response Surfaces Models : 2 days
- ✓ Uncertainty propagation methods for sensitivity & dispersion analysis : 2 * 1 day
- ✓ Uncertainty propagation methods for reliability evaluation : 2 days

- ✓ Introduction to the use of OpenTURNS : 2 days
- ✓ Modelica & Python: 1 day
- ✓ *Salome : 2 days*

- **PRACE**
  - ✓ 1 annual training at Maison de la Simulation (Saclay)
  - ✓ 3 days
  - ✓ Méthodology + Practice on OpenTURNS or Uranie (CEA)

Where to load OpenTURNS?

Sourceforge:

➔ **Sources**: for those who want to compile

➔ **Windows Python Module** (architectures 32 et 64 bit + 3 python versions)

➔ **Native Windows Library**

Sous Linux:

➔ distributions Debian, Ubuntu, CentoS, Fedor, openSUZE, Archlinux, Parabola

➔ compiled packages installed by the **package manager** (care: Administrative privilege)

Conda:

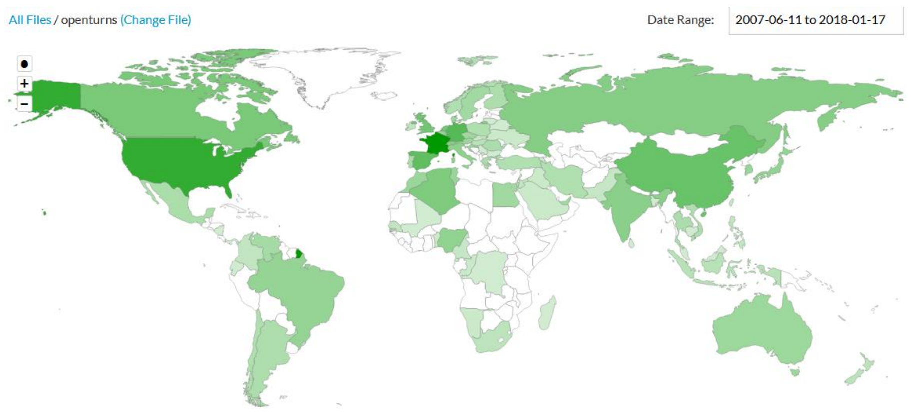
➔ installation of python without privilege (USERS)

➔ **Python Module python on anaconda** (https: //anaconda.org/openturns/openturns) since v1.7 (2016): windows / linux, several architectures, …

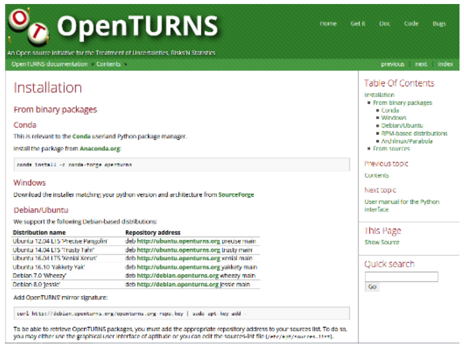
SALOME::

➔ OpenTURNS is now included in specific Salome version with a specific GUI

❖ https://www.salome-platform.org/contributions/copy_of_combs


All Files / openturns (Change File)
Date Range: 2007-06-11 to 2018-01-17

Téléchargements depuis SourceForge depuis 06/2007


OpenTURNS
Home Get it Doc Code Bugs
An Open source Initiative for the Treatment of Uncertainties, Risks'N Statistics
OpenTURNS documentation » Contents »
previous | next | index

Installation

From binary packages

Conda

This is relevant to the Conda userland Python package manager.

Install the package from Anaconda.org:

```
conda install -c conda-forge openturns
```

Windows

Download the installer matching your python version and architecture from SourceForge

Debian/Ubuntu

We support the following Debian-based distributions:

Distribution name	Repository address
Ubuntu 12.04 LTS 'Precise Pangolin'	deb http://ubuntu.openturns.org precise main
Ubuntu 14.04 LTS 'Trusty Tahr'	deb http://ubuntu.openturns.org trusty main
Ubuntu 16.04 LTS 'Xenial Xerus'	deb http://ubuntu.openturns.org xenial main
Ubuntu 16.10 'Yakkety Yak'	deb http://ubuntu.openturns.org yakkety main
Debian 7.0 'Wheezy'	deb http://debian.openturns.org wheezy main
Debian 8.0 'Jessie'	deb http://debian.openturns.org jessie main

Add OpenTURNS' mirror signature:

```
curl http://debian.openturns.org/openturns.org-repo.key | sudo apt-key add -
```

To be able to retrieve OpenTURNS packages, you must add the appropriate repository address to your sources list. To do so, you may either use the graphical user interface of aptitude or you can edit the sources-list file (/etc/apt/sources.list).

Table Of Contents
- Installation
  - From binary packages
    - Conda
    - Windows
    - Debian/Ubuntu
    - RPM-based distributions
    - Archlinux/Parabola
  - From sources

Previous topic

Contents

Next topic

User manual for the Python interface

This Page

Show Source

Quick search

Go

Téléchargements:

➔ **14 000** downloads from **SourceForge** since 2007 (1500 from 2017)

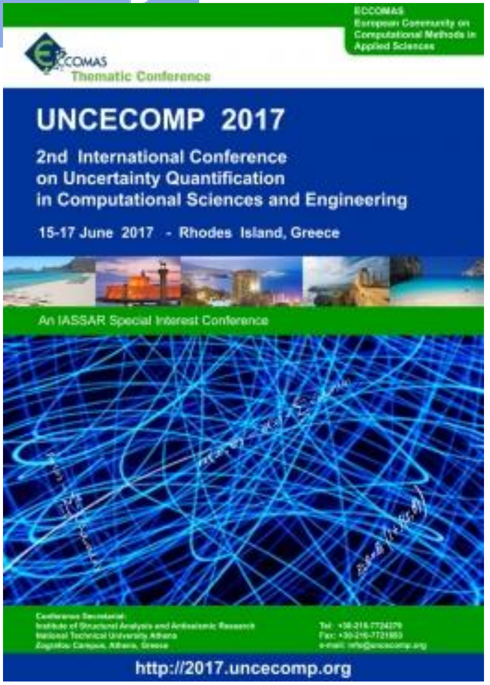
➔ **40 000** downloads **Conda** since the beginning of 2016

➔ + all the others!

Open TURNS publications

- ➢ **International conferences**
  - **EuroScipy (08/ 2017): python community**
    - ❑ **2-day conference in Europe**
    - ❑ **OpenTURNS presentation**
    - ❑ **Objective next year: OpenTURNS tutorial ➔ 1h30 live demonstration**
    - ❖ **Significative increase in OpenTURNS loads with conda**
  - **SIAMUQ SIAMUQ 2018 : uncertainty community**
  - **UNCECOMP 2017 : uncertainty community**

EuroSciPy



ECCOMAS
European Community on
Computational Methods in
Applied Sciences

ECCOMAS
Thematic Conference

UNCECOMP 2017

2nd International Conference
on Uncertainty Quantification
in Computational Sciences and Engineering

15-17 June 2017 - Rhodes Island, Greece

An IASSAR Special Interest Conference

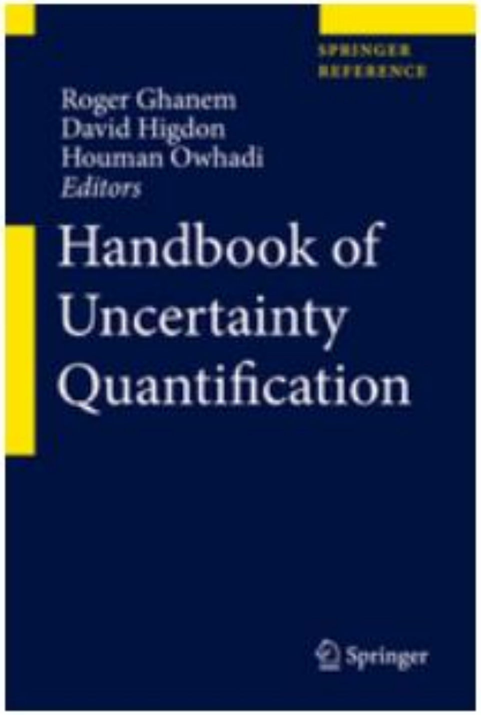
Conference Secretariat:
Institute of Structural Analysis and Antiseismic Research
National Technical University Athens
Zografou Campus, Athens, Greece

Tel: +30-210-7724279
Fax: +30-210-7721693
e-mail: info@uncecomp.org

http://2017.uncecomp.org

SIAM Conference on
Uncertainty
Quantification

April 16-19, 2018
Hyatt Regency Orange County,
Garden Grove, California, USA



SPRINGER
REFERENCE

Roger Ghanem
David Higdon
Houman Owhadi
Editors

Handbook of
Uncertainty
Quantification

Springer

- ➢ **Publication :**
  - **Springer Handbook of Uncertainty Quantification (2017): one chapter dedicated to OpenTURNS**

Open TURNS in the academic world

Master curse based on OpenTURNS (EDF /Phimeca)

Anthony Nouy : 3d year curse on meta models (low rank tensors)

Henning Bruske: Master 2 curse

Engineers trained to the Uncertainty Methodology and able to use OpenTURNS on the labor market very soon

Uncertainty Management – Robust Optimization (Airbus)

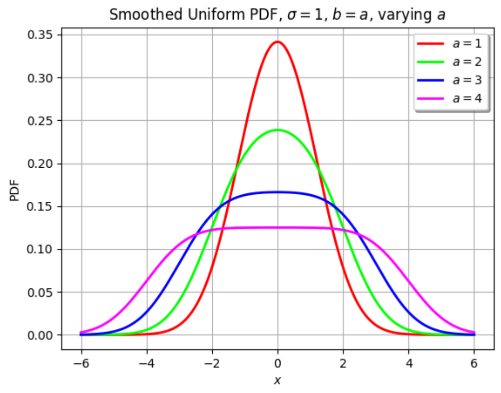
Master curse based on OpenTURNS(CEA, IMACS)

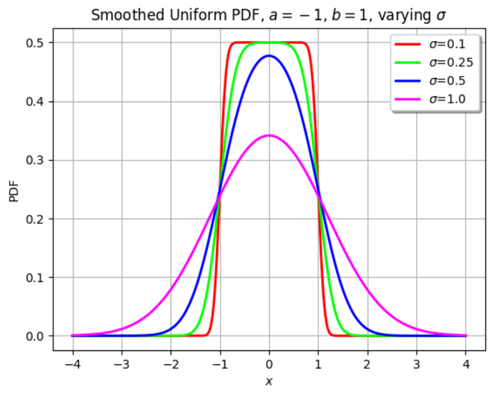
Master curse on Data Science

Uncertainty Management curse (participation EDF and Airbus)

New features of the 1.10 & 1.11 releases

- **Uncertainty modelling & quantification**
  - ✓ **SmoothedUniform(a, b, σ) distribution of X+Y where X~U(a, b), Y~N(0,σ), X indep. Y**





This distribution allows for further simplifications in RandomMixture (speed & accuracy gains)

  - ✓ **FrechetFactory based on maximum likelihood estimates. Used eg in extreme values quantification.**

New features of the 1.10 & 1.11 releases

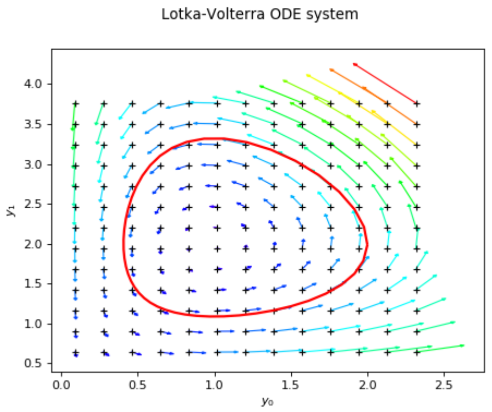
- **Numerical methods**
  - ✓ **Fehlberg to integrate systems of ordinary differential equations (explicit, adaptive)**

Nested Runge Kutta schemes

Dynamic simulation grid by local error estimation

Results interpolated on the user grid.

Four schemes available (1/2 to 4/5)

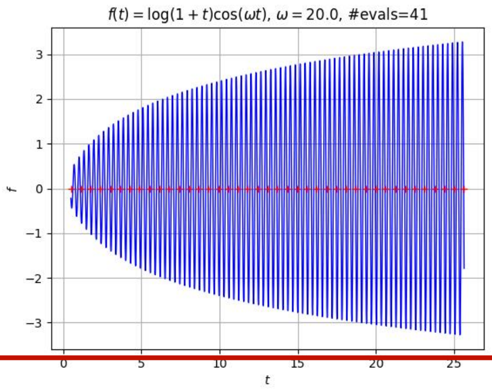


  - ✓ **FilonQuadrature**

Filon: 41 calls, 0.08% error

Gauss Legendre: 41 calls, >100000% error

Gauss Kronrod: 60 calls, >100000% error



New features of the 1.10 & 1.11 releases

➢ **New models with functional input or output**

- ✓ **OT 1.9:**
  - **Fonction to map R^n into R^q**
  - **FieldFunction to map $F(R^m, R^n)$ into $F(R^p, R^q)$**
- ✓ **OT 1.10 adds**
  - **PointToFieldFunction to map R^n into $F(R^p, R^q)$**
  - **FieldToPointFunction to map $F(R^m, R^n)$ into R^q**

+ all the classes to compose these models in any way: PointToPointConnection, PointToFieldConnection, FieldToPointConnection, FieldToFieldConnection

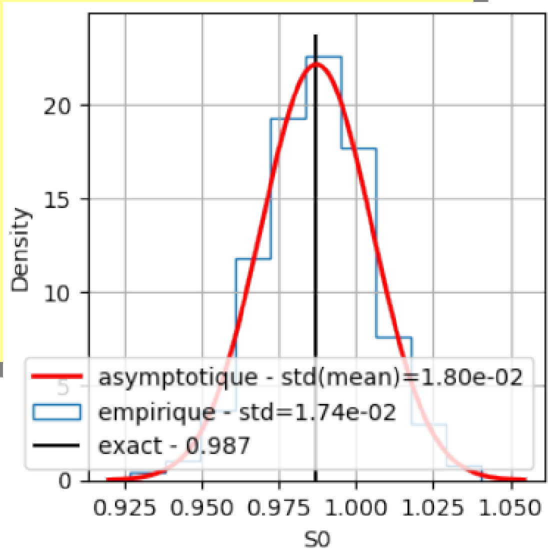
❖ **You still have to present all the inputs as a unique field if at least one of them is a field, and all the inputs must share the same mesh. Same constraint for the output.**

❖ **The input and output meshes will be part of the model definition in the future version, only the sample of values will transit through the evaluation operator**

New features of the 1.10 & 1.11 releases

- **Sobol' indices asymptotic variance estimators**
  - ✓ Previously only bootstrap estimator was available (asymptotic only for Martinez)
  - ✓ Asymptotic variance estimators using the "delta" method (Janon & al, 2014)
  - ✓ Variance estimators are declined for 4 all Sobol' estimators (Jansen, MauntzK, ...)
  - ✓ Access to the confidence interval as well as the distribution of the indices
  - ✓ Some corrections of the raw estimators of the indices (Saltelli, ...)

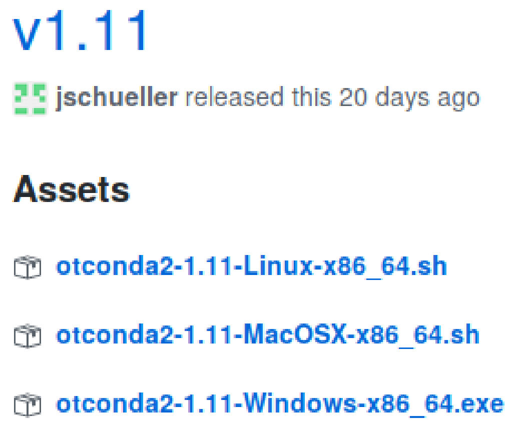
```
import openturns as ot
X = ot.SobolIndicesExperiment(distribution, size).generate()
Y = f(X)
algo = ot.SaltelliSensitivityAlgorithm(X, Y)
algo.setUseAsymptoticDistribution(True)

fo = algo.getFirstOrderIndices()
to = algo.getTotalOrderIndices()

interval_fo = algo.getFirstOrderIndicesInterval()
interval_to = algo.getTotalOrderIndicesInterval()

dist_fo = algo.getFirstOrderIndicesDistribution()
dist_to = algo.getTotalOrderIndicesDistribution()
```



New features of the 1.10 & 1.11 releases

- **New installation media**
  - ✓ Python package index (PyPI)
  - ✓ otconda bundle : Anaconda-like offline installer of OpenTURNS and its modules

```
pip install openturns [--user]
```



v1.11

jschueller released this 20 days ago

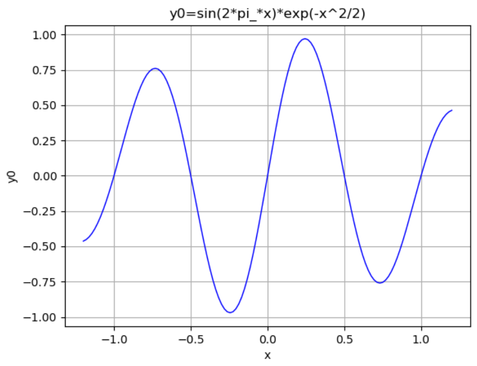
Assets

- otconda2-1.11-Linux-x86_64.sh
- otconda2-1.11-MacOSX-x86_64.sh
- otconda2-1.11-Windows-x86_64.exe

New features of the 1.10 & 1.11 releases

- **ExprTk**
  - ✓ Like muParser, allows to parse analytical formulas from C++:
    - + no external dependency, this library is a single C++ header file
    - + slightly faster evaluation than muParser
    - + supports multiple outputs
    - + full programming language (see its documentation)
    - - requires more memory at run time
  - ✓ Both muParser and ExprTk are currently supported
  - ✓ ResourceMap `SymbolicParser-Backend` entry set to `ExprTk` (default) or `MuParser`

```
>>> import openturns as ot
>>> from openturns.viewer import View

>>> formula = "sin(2*pi_*x)*exp(-x^2/2)"
>>> f = ot.SymbolicFunction("x", formula)
>>> graph = f.draw(-1.2, 1.2)
>>> graph.setTitle("y0="+formula)
>>> View(graph).show()
```



```
>>> import openturns as ot
>>> from math import sqrt, atan2
>>> formula = "r := sqrt(x * x + y * y); theta := atan2(y, x)"
>>> polar = ot.SymbolicFunction(["x", "y"], ["r", "theta"], formula)
>>> print(polar([1.0, 1.0]))
[1.41421,0.785398]
```

New features of the 1.10 & 1.11 releases

- **Improve objects conversion Python ↔ OpenTURNS**
  - ✓ **Avoid copies when converting OT containers (Point, Sample, Matrix) into Numpy arrays**
  - ✓ **In general, there are still copies when converting Numpy arrays into OT containers**
  - ✓ **Functions defined in Python now receive an argument which is not a Sample but a view on a Sample, which only supports indexing ; should work everywhere, call Sample(X) if you need to call Sample methods on X**
  - ✓ **Functions defined in Python can be as fast as a SymbolicFunction (defined in Python),**

  but not trivial ; implement evaluation on a Sample with Numpy arrays to avoid Python for-loops

```
import openturns as ot
from math import cos, sin, tan, exp
import numpy as np
def f_point(X):
    x = X[0]
    return [exp(cos(sin(tan(x)))) \
            + cos(exp(sin(tan(x))))]
def f_sample(X):
    x = np.array(X)
    return np.exp(np.cos(np.sin(np.tan(x)))) \
           + np.cos(np.exp(np.sin(np.tan(x))))

f0 = ot.PythonFunction(1, 1, f_point)
f1 = ot.PythonFunction(1, 1, func_sample=f_sample)
f2 = ot.SymbolicFunction("x",
     "exp(cos(sin(tan(x))))+cos(exp(sin(tan(x))))")
```

	OT 1.9 & 1.10	OT 1.11
f0	5.85s	1.15s
f1	0.60s	0.17s
f2	0.35s	0.22s

Evaluation on a Sample of size 1E6

New features of the 1.10 & 1.11 releases

- **Changes in handling of cache and history of functions**
  - ✓ **Cache and history was defined in lowest-level class, and derived classes had to take care of implement them if needed**
  - ✓ **Now these methods have been removed, and a single class MemoizeFunction is added**

OT < 1.11

```
import openturns as ot
…

heavyFunction.enableCache()
heavyFunction.enableHistory()
outSample = heavyFunction(inputSample)
nrCalls = heavyFunction.getCallsNumber()
```

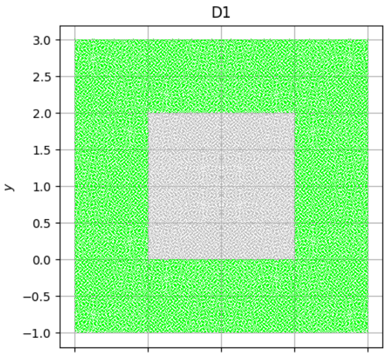
OT ≥ 1.11

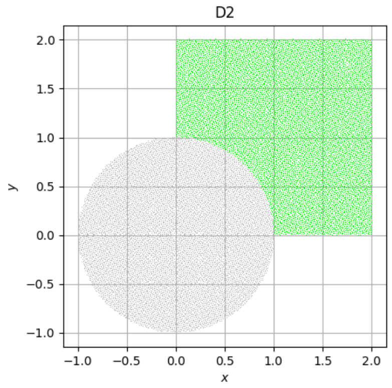
```
import openturns as ot
…

myFunc = ot.MemoizeFunction(heavyFunction)
myFunc.enableCache()
myFunc.enableHistory()
outSample = myFunc(inputSample)
nrCalls = myFunc.getCallsNumber()
```

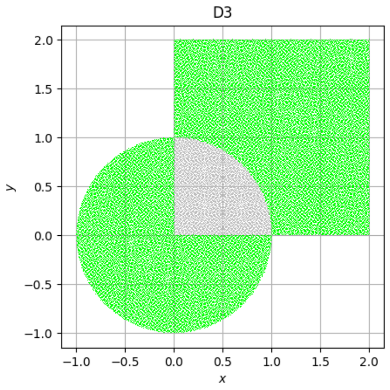
New features of the 1.10 & 1.11 releases

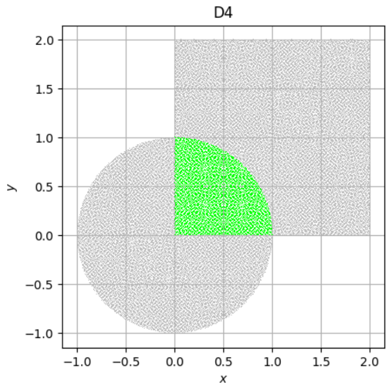
- **Geometry**
  - ✓ Domain algebra: eg allows to define complex failure events

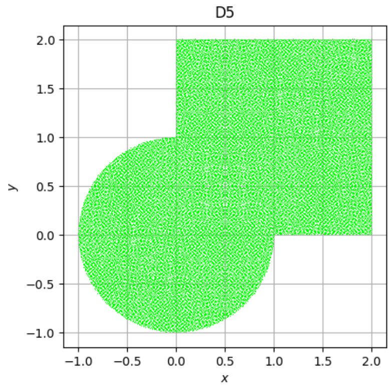
```
D1 = DomainComplement(cube)
D2 = DomainDifference(cube, sphere)
D3 = DomainDisjunctiveUnion(cube, sphere)
D4 = DomainIntersection(cube, sphere)
D5 = DomainUnion(cube, sphere)

X = RandomVector(Normal(2))
E1 = EventDomain(X, D1) # or D2, D3…
```











New features of the 1.10 & 1.11 releases

- **Geometry (cont'ed)**
  - ✓ Up to 1.10, Mesh contained a kd-tree to speed-up nearest neighbour searches
  - ✓ Mesh is now only a container of vertices and simplices
  - ✓ Services for spatial lookup are moved into new classes
  - ✓ Mesh no more derives from Domain
  - ✓ MeshDomain(mesh) converts a Mesh into a Domain

- **Spatial lookup**
  - ✓ New classes added in 1.11 :

    NearestNeighbourAlgorithm (interface class)

    NearestNeighbour1D, RegularGridNearestNeighbour (1-d meshes)

    KDTree, NaiveNearestNeighbour
  - ✓ These classes provide a `query` method to find the index of the nearest neighbour of a Point

```
>>> import openturns as ot
>>> sample = ot.Normal(2).getSample(10000)
>>> finder = ot.NearestNeighbourAlgorithm(sample)
>>> neighbour = sample[finder.query([0.1, 0.2])]
>>> print(neighbour)
[0.0967085,0.194408]
```

New features of the 1.10 & 1.11 releases

- **Spatial lookup (cont'ed)**
  - ✓ New classes to find the enclosing simplex of a Point in a Mesh :
    - EnclosingSimplexAlgorithm (interface class)
    - EnclosingSimplexMonotonic1D, RegularGridEnclosingSimplex (1-d meshes)
    - NaiveEnclosingSimplex, BoundingVolumeHierarchy
  - ✓ These classes provide a `query` method to find the index of the enclosing simplex

```
>>> import openturns as ot
>>> mesh = ot.IntervalMesher([100]*2).build(ot.Interval([0.0]*2, [10.0]*2))
>>> locator = ot.EnclosingSimplexAlgorithm(mesh.getVertices(), mesh.getSimplices())
>>> simplex = locator.query([0.1, 0.2])
>>> print(mesh.getVertices().select(mesh.getSimplex(simplex)))
    [ v0  v1  ]
0 : [ 0.1 0.2 ]
1 : [ 0.2 0.2 ]
2 : [ 0.1 0.3 ]
```

New features of the 1.10 & 1.11 releases

- **Spatial lookup (cont'ed)**
  - ✓ Used by P1LagrangeEvaluation (to create a Function) and new class P1LagrangeInterpolation (to create a FieldFunction)
  - ✓ Mesh interpolation is computed for each evaluation of P1LagrangeEvaluation
  - ✓ Mesh interpolation is computed only once in P1LagrangeInterpolation, this is preferred when interpolating several fields

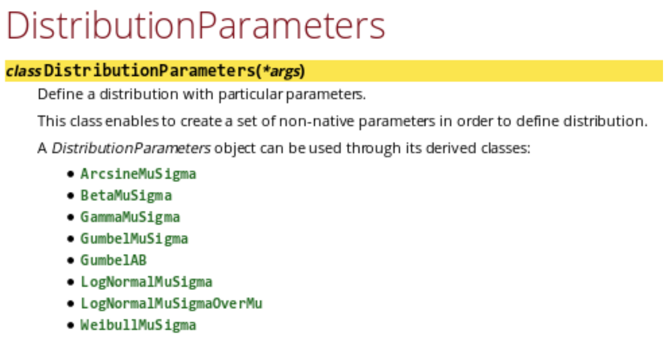
```
>>> import openturns as ot
>>> field = ot.Field(ot.RegularGrid(0.0, 1.0, 4), [[0.5], [1.5], [1.0], [-0.5]])
>>> evaluation = ot.P1LagrangeEvaluation(field)
>>> print(evaluation([2.3]))
[0.55]
```

```
>>> import openturns as ot
>>> inputMesh = ot.RegularGrid(0.0, 1.0, 4)
>>> outputMesh = ot.Mesh([[2.3]])
>>> interpolation = ot.P1LagrangeInterpolation(inputMesh, outputMesh, 1)
>>> field = ot.Field(inputMesh, [[0.5], [1.5], [1.0], [-0.5]])
>>> print(interpolation(field))
0 : [ 0.55 ]
```

Perspectives 2018-2019: main ongoing works

User Comfort

- ✓ **Increase the coherence in the use of parameters**
  - ❑ **Modèles de covariance: notion de paramètres actifs**
  - ❑ **Fonctions: notion de ParametrizedFunctions**
  - ❑ **Distributions: notion de paramétrage alternatif.**



DistributionParameters

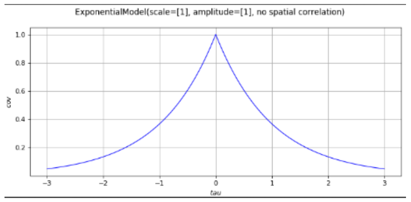
class **DistributionParameters(**args*)**

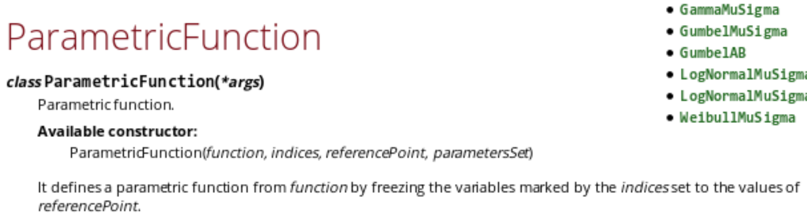
Define a distribution with particular parameters.

This class enables to create a set of non-native parameters in order to define distribution.

A *DistributionParameters* object can be used through its derived classes:

- ArcsineMuSigma
- BetaMuSigma
- GammaMuSigma
- GumbelMuSigma
- GumbelAB
- LogNormalMuSigma
- LogNormalMuSigmaOverMu
- WeibullMuSigma





ParametricFunction

class **ParametricFunction(**args*)**

Parametric function.

Available constructor:

ParametricFunction(*function, indices, referencePoint, parametersSet*)

It defines a parametric function from *function* by freezing the variables marked by the *indices* set to the values of *referencePoint*.

- ✓ **Complete Python Projection ?**
  - ❑ **Use of a dictionary to declare the arguments of a method**
  - ❑ **Integration in pypi (python packaging system, currently : conda)**
  - ❑ **Enhanced the classes Sample and Matrix with functionalities similar to numpy**

- ✓ **Install OpenTURNS under the windows environment without Administrator privilege nor Internet connection**

- ✓ **Performance**
  - ✓ **Parallel evaluations of functions**
  - ✓ **Gérer de manière globale les threads lancés par OpenTURNS provenant de TBB ou d'OpenBlas**
  - ✓ **Ongoing work on H-Mat représentation compressée de matrices)**

- ✓ **Documentation**
  - ✓ **Finalize the transformation to dynamic doc**
  - ✓ **More and more complete…**

Perspectives 2018-2019: main ongoing works

Meta models

- ✓ Validation tests
  - ❑ QQ-Plot, residuals graph, …
  - ❑ Mutualize as much as possible the validation methods among meta models
- ✓ « Easy chaos »
  - ❑ Simplify as much as possible the parametrization of the algorithms

Statistical inference

- ✓ Distribution of the estimators
  - ❑ Quantile, probabilities, SRC indices, Sobol indices, moments, …
  - ❑ Useful to establish confidence intervals

Sensitivity analysis

- ✓ Csiszar indices
  - ❑ Distance between (Y,X_i) copula and the independent copula

$$\boxed{d_i = \mathbb{E}\left[D_f(F_Y \| F_{Y|X_i})\right]} \Longrightarrow \boxed{d_i = \mathbb{E}\left[D_f(C_{Y,X_i} \| \Pi)\right]}$$

Perspectives 2018-2019: main ongoing works

System models

- ✓ Systems Events: union and intersection of elementary events
  - ❑ Probability estimation
- ✓ Modulus otfmi
  - ❑ Fmi: standard interface model
  - ❑ Enables to insert within a modelica description a connection to an external composant
  - ❑ Enhance the modulus with all the OpenTURNS connectors: vector / field

Maintenance - Validation

- ✓ Unit validation python test
  - ❑ Distance from a reference result
  - ❑ Infrastrusture of the environmment test
  - ❑ A good example: Scikit-Learn ….

Diffusion

- ✓ Python communities
  - ❑ EuroScipy
  - ❑ Scipy
- ✓ Uncertainty communities
  - ❑ SIAMUQ
- ✓ Publications
  - ❑ Dedicated to a particular method: multivariate Kriging, Karhunen Loeve, … + notebooks
  - ❑ Journal of Statistical Softwere

OpenTURNS: *otagrum*

« Graph approach for high dimensional probabilistic model – otagrum module »

PH. Wuillemin (Sorbonne University, LIP6)
R. Lebrun (Airbus)

Studies with Open TURNS

Study 1: **« Anomaly detection in aircraft sensor data »: N.Huet (IMACS), J. Sen Gupta (Airbus)**

Study 2: **« ANATOLE: 3D tolerancing and sensitivity analysis »: B. Chaigne (IMACS)**

Study 3: **« New features of the GUI of OpenTURNS: system modelisations, dependence, validation»: M. Baudin (EDF), J. Schueller (Phimeca), A. Ladier (Phimeca)**

Study 4: **« Marges for the conception of complex systems »: A. Touboul (IRT)**

Break

Study 5 :**« Estimation of penstock pipes reliability using system reliability methods»: E. Ardillon (EDF), A. Dumas (Phimeca)**

Study 6: **« Modelisation of the maximum annual magnitude of earthquakes with extreme value distributions»: A. Dutfoy (EDF)**

The end ....

Thanks for your participation ...

and see you next year !