NISP Toolbox Manual

Version 0.1
July 2009

Michaël Baudin
Jean-Marc Martinez

Contents

Introduction

The goal of this toolbox is to provide a tool to manage uncertainties in simulated models. This toolbox is based on the NISP library, where NISP stands for "Non-Intrusive Spectral Projection". This work has been realized in the context of the OPUS project, "Open-Source Platform for Uncertainty treatments in Simulation", funded by ANR, the french "Agence Nationale pour la Recherche".

The NISP library is based on a set of 3 C++ classes so that it provides an object-oriented framework for uncertainty analysis. The Scilab toolbox provides a pseudo-object oriented interface to this library, so that the two approaches are consistent.

The NISP library provides three tools, which are detailed below.

- The "randvar" class allows to manage random variables, specified by their distribution law and their parameters. Once a random variable is created, one can generate random numbers from the associated law.

- The "setrandvar" class allows to manage a collection of random variables. This collection is associated with a sampling method, such as MonteCarlo, Sobol, Quadrature, etc... It is possible to build the sample and to get it back so that the experiments can be performed.

- The "polychaos" class allows to manage a polynomial representation of the simulated model. One such object must be associated with a set of experiments which have been performed. This set may be read from a data file. The object is linked with a collection of random variables. Then the coefficients of the polynomial can be computed by integration (quadrature). Once done, the mean, the variance and the Sobol indices can be directly computed from the coefficients.

Chapter 1

Overview

In this section, we present the main commands of the NISP toolbox as well as an example of use.

Chapter 2

Installation

In this section, we present the installation process for the toolbox. We present the steps which are required to have a running version of the toolbox and presents the several checks which can be performed before using the toolbox.

2.1 Architecture of the directories

We suppose that the archive has been unpacked in the "tbxnisp" directory. The following is a short list of the steps which are required to setup the toolbox.

1. build the toolbox : run the *tbxnisp/builder.sce* script to create the binaries of the library, create the binaries for the gateway, generate the documentation
2. load the toolbox : run the *tbxnisp/load.sce* script to load all commands and setup the documentation
3. setup the startup configuration file of your Scilab system so that the toolbox is known at startup (see below for details),
4. run the unit tests : run the *tbxnisp/runtests.sce* script to perform all unit tests and check that the toolbox is OK
5. run the demos : run the *tbxnisp/rundemos.sce* script to run all demonstration scripts and get a quick interactive overview of its features

The easiest way to setup your Scilab system is to configure the startup configuration file so that the toolboxes are known immediately at startup. The directory where this file is located is stored in the Scilab variable *SCIHOME*. On my Linux system, the Scilab 5.1 startup file is located in */home/myname/.Scilab/scilab-5.1/.scilab*. On my Windows system, the Scilab 5.1 startup file is located in *C:/Users/myname/AppData/Roaming/Scilab/scilab-5.1/.scilab*. This file is a regular Scilab script which is automatically loaded at Scilab's startup. If that file does not already exist,

create it. Copy the following lines into the *.scilab* file and configure the path to the toolboxes, stored in the *SCILABTBX* variable.

```
ilib(0);
SCILABTBX="/home/myname/mytoolboxes";
exec(SCILABTBX + filesep() + 'tbxnisp'+ filesep() + 'loader.sce');
```

The figure 2.1 presents the messages which are generated when the builder of the toolbox is launched.

```
-->exec D:\Baudin\ProjetScilab\toolboxes\tbxnisp\builder.sce
...
```

Fig. 2.1 : Launch of the builder

The figure 2.2 presents the messages which are generated when the loader of the toolbox is launched.

```
-->exec D:\Baudin\ProjetScilab\toolboxes\tbxnisp\loader.sce
...
```

Fig. 2.2 : Launch of the loader

The figure 2.3 and 2.4 presents the messages which are generated when the unit tests script of the toolbox is launched.

```
-->exec D:\Baudin\ProjetScilab\toolboxes\tbxnisp\runtests.sce
...
```

Fig. 2.3 : Launch of the unit tests script (part 1/2)

2.2 Configuration

The directories which are provided in the toolbox are presented in figure 2.5.

This is an overview of the content of these directories :

- *tbxnisp/demos* : demonstration scripts
- *tbxnisp/doc* : the documentation
- *tbxnisp/doc/usermanual* : the LaTeXsources of this manual

...

Fig. 2.4 : Launch of the unit tests script (part 2/2)

Fig. 2.5 : Architecture of the toolbox

- *tbxnisp/etc* : startup and shutdow scripts for the toolbox
- *tbxnisp/help/en_US/scilab_en_US_help* : html pages of the help
- *tbxnisp/jar* : java archive for the help
- *tbxnisp/macros* : Scilab macros files *.sci
- *tbxnisp/tests* : tests
- *tbxnisp/tests/nonreg_tests* : tests after some bug has been identified
- *tbxnisp/tests/unit_tests* : unit tests

2.3 NISP Library

The current version is based on the NISP Library v2.1. As presented in figure 2.5, the NISP library must be installed in the directory *src/nisp*.

Chapter 3

The "randvar" class

In this section, we present the "randvar" class, which allows to define a random variable, and to generate random numbers from a given distribution function.

3.1 Introduction

In this section, we make a brief introduction to the statistical definitions of distribution function, expectation and variance. This section does not aim at giving a complete overview of statistics, but aims at setting the notations used in this document.

The density function and the cumulative distribution function of a continuous variable are defined as following.

Definition 3.1.1 *(*Density function*) Let X be a continuous random variable. A density function (or distribution function) for X is a function $f : \mathbb{R} \to \mathbb{R}$ which satisfies*

$$P(a \leq x \leq b) = \int_a^b f(x)dx, \tag{3.1}$$

for all $a, b \in \mathbb{R}$.

Definition 3.1.2 *(*Cumulative distribution function*) Let X be a continuous random variable. The cumulative distribution function F of X is defined by*

$$F(x) = P(X \leq x), \tag{3.2}$$

for all $x \in \mathbb{R}$.

The relationship between the density function and the cumulative distribution function is given by the following proposition, which is not prooved in this document.

Name	$f(x)$	Expectation	Variance
"Normale"	$\frac{1}{2\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$	μ	σ^2
"Uniforme"	$\begin{cases} \frac{1}{b-a}, & \text{if } x \in [a,b[\\ 0 & \text{if } x \notin [a,b[\end{cases}$	$\frac{b+a}{2}$	$\frac{(b-a)^2}{12}$
"Exponentielle"	$\begin{cases} \lambda \exp(-\lambda x), & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$
"LogNormale"	$\begin{cases} \frac{1}{\sigma x\sqrt{2\pi}} \exp\left(-\frac{1}{2}\frac{(\ln(x)-\mu)^2}{2\sigma^2}\right), & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$	$\mu' = \exp\left(\mu + \frac{1}{2}\sigma^2\right)$	$\left(\exp(\sigma^2) - 1\right) \exp\left(2\mu + \sigma^2\right)$
"LogUniforme"	$\begin{cases} \frac{1}{x}\frac{1}{\ln(b)-\ln(a)}, & \text{if } x \in [a,b[\\ 0 & \text{if } x \notin [a,b[\end{cases}$	$\frac{b-a}{\ln(b)-\ln(a)}$	$\frac{1}{2}\frac{b^2-a^2}{\ln(b)-\ln(a)} - E(x)$

Fig. 3.1 : Distributions functions of the "randvar" class

Proposition 3.1.3 *Let X be a continuous real-valued random variable with density function $f(x)$. Then the function F defined by*

$$F(x) = \int_{-\infty}^{x} f(t)dt, \tag{3.3}$$

is the cumulative distribution function of X. Furthermore, we have

$$F'(x) = f(x). \tag{3.4}$$

Definition 3.1.4 *(*Variance*) Let X be a real-valued random variable with density function $f(x)$ and expectation $mu = E(X)$. The variance $\sigma^2 = V(X)$ is defined by*

$$\sigma^2 = V(X) = E((X-\mu)^2). \tag{3.5}$$

In practice, it may be convenient to compute the variance with the equivalent formula

$$V(X) = \int_{x\in\mathbb{R}} (x-\mu)^2 f(x)dx. \tag{3.6}$$

One can proove that the variance can be computed from the formula

$$V(X) = E(X^2) - \mu^2. \tag{3.7}$$

3.2 The distribution functions

The table 3.1 gives the list of distribution functions which are available with the "randvar" class [2].

For the "LogNormale" law, the parameters given when creating such a variable are μ', the expected value of the LogNormale law and σ, the variance of the underlying Normale law. The mean μ associated with the Normal law is computed from the equation

$$\mu = \ln(\mu') - \sigma^2. \tag{3.8}$$

Name	Parameter #1	Parameter #1	Conditions
"Normale"	$\mu = 0.$	$\sigma = 1.$	$\sigma > 0$
"Uniforme"	$a = 0.$	$b = 1.$	$a < b$
"Exponentielle"	$\lambda = 1.$	-	-
"LogNormale"	$\mu' = 0.1$	$\sigma = 1.0$	$\mu', \sigma > 0$
"LogUniforme"	$a = 0.1$	$b = 1.0$	$a, b > 0$, $a < b$

Fig. 3.2 : Default parameters for distributions functions

One random variable can be specified by giving explicitely its parameters or by using default parameters. The parameters for all distribution function are presented in figure 3.2, which also presents the conditions which must be satisfied by the parameters.

3.2.1 Derivation of the "LogNormale" and "LogUniforme" laws

The LogNormale distribution function is presented in many textbooks. Since the LogUniforme law, which is not so common, is based on the same principles, we choosed to present the derivation of this type of laws.

Let X be a continuous random variable with cumulative distribution function F_X and density function f_X. Let Y be the random variable computed from $Y = \exp(X)$. We are going to compute the cumulative distribution function F_Y and the density function f_Y of the variable Y and define it as the Log-f law.

The cumulative distribution function F_Y is defined by

$$\begin{aligned} F_Y(x) &= P(Y \leq x) && (3.9)\\ &= P(\exp(X) \leq x) && (3.10)\\ &= P(X \leq \ln(x)) && (3.11)\\ &= F_X(\ln(x)). && (3.12) \end{aligned}$$

Therefore, the cumulative distribution function F_Y is

$$F_Y(x) = \int_{-\infty}^{\ln(x)} f_X(t)dt. \quad (3.13)$$

We now make the change of variable $s = \exp(t)$. This implies $t = \ln(s)$ and therefore $dt = \frac{1}{s}ds$. We have

$$F_Y(x) = \int_0^x \frac{1}{s} f_X(\ln(s))ds, \quad (3.14)$$

which implies that the distribution function f_Y satisfies

$$f_Y(x) = \frac{1}{x} f_X(\ln(x)). \quad (3.15)$$

We can now apply equation 3.15 in order to compute the LogNormale and LogUniforme distribution functions.

The Normale distribution function is

$$f_X(x) = \frac{1}{2\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right). \tag{3.16}$$

We apply 3.15 and we get the LogNormale distribution function

$$f_Y(x) = \frac{1}{2x\sigma\sqrt{2\pi}} \exp\left(-\frac{(\ln(x)-\mu)^2}{2\sigma^2}\right). \tag{3.17}$$

The Uniforme distribution function is

$$f_X(x) = \frac{1}{b'-a'}, \qquad x \in [a', b'], \tag{3.18}$$

for $a', b' \in \mathbb{R}$. We apply 3.15 and we get the LogNormale distribution function

$$f_Y(x) = \frac{1}{x}\frac{1}{b'-a'}, \qquad \ln(x) \in [a', b']. \tag{3.19}$$

To get the final form of the LogNormale distribution function, we define the parameters

$$a = \exp(a'), \quad b = \exp(b'), \tag{3.20}$$

where $a, b > 0$. These equalities can be reversed so that

$$a' = \ln(a), \quad b' = \ln(b). \tag{3.21}$$

If we plug the previous equality into 3.19, we get the LogUniforme distribution function

$$f_Y(x) = \frac{1}{x}\frac{1}{\ln(b)-\ln(a)}, \qquad x \in [a, b], \tag{3.22}$$

where $a, b > 0$.

3.2.2 Methods

In this section, we give an overview of the methods which are available in the "randvar" class.

The figure 3.3 presents the methods available in the "randvar" class. The detailed analysis of each method is presented in figure 3.4.

3.2.3 A sample session

We present a sample Scilab session, where the "randvar" class is used to generate samples from the Normale law.

In the following Scilab session, we create a Normale random variable and compute samples from this law. The *nisp_initseed* function is used to initialize the seed for the uniform random

rvlist = randvar_tokens ()
nbrv = randvar_size ()
rv = randvar_new (type [, options])
value = randvar_getvalue (rv [, options])
randvar_destroy (rv)
randvar_getlog (rv)

Fig. 3.3 : Outline of the methods of the "randvar" class

variable generator. Then we use the *randvar_new* function to create a new random variable from the Normale law with mean 1. and standard deviation 0.5. The main loop allows to compute 1000 samples from this law, based on calls to the *randvar_getvalue* function. Once the samples are computed, we use the Scilab function *mean* to check that the mean is close to 1 (which is the expected value of the Normale law, when the number of samples is infinite). Finally, we use the *randvar_destroy* function to destroy our random variable.

```
1  nisp_initseed ( 0 );
2  rv = randvar_new("Normale" , 1.0 , 0.5 );
3  nbshots = 1000;
4  values = zeros(nbshots);
5  for i=1:nbshots
6    values(i) = randvar_getvalue(rv);
7  end
8  computed = mean (values); // Exact value : 1.0
9  computed = variance (values); // Exact value 0.5^2
10 randvar_destroy(rv);
```

3.2.4 Variable transformations

In this section, we give some additionnal explanations for the function *randvar_getvalue (rv , rv2 , value2)*. In short, this method allows to transform a random variable sample from one law to another.

More specifically, this method allows to compute a new sample from the law associated with the current random variable *rv*. This sample is based on a transformation of the value *value2*, which is expected to be computed from the law associated to *rv2*.

In the following session, we transform a uniform random variable sample into a LogUniform variable sample. We begin to create a random variable *rv* from a LogUniform law and parameters $a = 10, b = 20$. Then we create a second random variable *rv2* from a Uniforme law and parameters $a = 2, b = 3$. The main loop is based on the transformation of a sample computed from *rv2* into a sample from *rv*. The *mean* allows to check that the transformed samples have an mean value which corresponds to the random variable *rv*.

rvlist = randvar_tokens () returns the current list of random variables
nbrv = randvar_size () returns the number of random variables currently in use
rv = randvar_new ("Normale") returns a Normale random variable with default parameters
rv = randvar_new ("Normale" , mu , sigma) returns a Normale random variable with parameters *mu* and *sigma*
rv = randvar_new ("Uniforme") returns a Uniforme random variable with default parameters
rv = randvar_new ("Uniforme" , a , b) returns a Uniforme random variable with parameters *a* and *b*
rv = randvar_new ("Exponentielle") returns a Exponentielle random variable with default parameters
rv = randvar_new ("Exponentielle" , lambda) returns a Exponentielle random variable with parameter *lambda*
rv = randvar_new ("LogNormale") returns a LogNormale random variable with default parameters
rv = randvar_new ("LogNormale" , mu , sigma) returns a LogNormale random variable with parameters *mu* and *sigma*
rv = randvar_new ("LogUniforme") returns a LogUniforme random variable with default parameters
rv = randvar_new ("LogUniforme" , a , b) returns a LogUniforme random variable with parameters *a* and *b*
value = randvar_getvalue (rv) returns a random value from the distribution function of the current random variable
value = randvar_getvalue (rv , rv2 , value2) returns a random value from the distribution function of the random variable *rv* by transformation of *value2* from the distribution function of random variable *rv2*
randvar_destroy (rv) destroys the current random variable
randvar_getlog (rv) prints a log for the current random variable

Fig. 3.4 : Methods of the "randvar" class

```
nisp_initseed ( 0 );
a = 10.0;
b = 20.0;
rv = randvar_new ( "LogUniforme" , a , b );
rv2 = randvar_new ( "Uniforme" , 2 , 3 );
nbshots = 1000;
values = zeros(nbshots);
for i=1:nbshots
  value2 = randvar_getvalue( rv2 );
  values(i) = randvar_getvalue( rv , rv2 , value2 );
end
computed = mean (values);
mu = (b-a)/(log(b)-log(a))
expected = mu; // "computed" should be close to "expected"
randvar_destroy(rv);
randvar_destroy(rv2);
```

The transformation depends on the *mother* random variable *rv1* and on the *daughter* random variable *rv*. Specific transformations are provided for all many combinations of the two distribution functions. These transformations will be analysed in the next sections.

3.3 Analysis of the distributions

In this section, we analyse the algorithms which are used in the "randvar" class. For each distribution, we present the random number generator principles. We also present the transformations which allow to transform a given outcome from one distribution to another distribution.

3.3.1 Uniforme random number generation

In this section, we present the generation of uniform random numbers.

The Uniforme law is associated with the parameters $a, b \in \mathbb{R}$ with $a < b$. It produces real values uniform in the interval $[a, b]$.

To compute the uniform random number X in the interval $[a, b]$, a uniform random number in the interval $[0, 1]$ is generated and then scaled with

$$X = a + (b - a)\overline{X}. \tag{3.23}$$

Let us now analyse how the uniform random number $\overline{X} \in [0, 1]$ is computed. The uniform random generator is based on the C function *rand*, which returns an integer n in the interval $[0, RAND\_MAX[$. The value of the $RAND\_MAX$ variable is defined in the file stdlib.h and is compiler-dependent. For example, with the Visual Studio C++ 2008 compiler, the value is

$$RAND\_MAX = 2^{15} - 1 = 32767. \tag{3.24}$$

Source	Target
Normale	
	Normale
	Uniforme
	Exponentielle
	LogNormale
	LogUniforme

Source	Target
LogNormale	
	Normale
	Uniforme
	Exponentielle
	LogNormale
	LogUniforme

Source	Target
Uniforme	
	Uniforme
	Normale
	Exponentielle
	LogNormale
	LogUniforme

Source	Target
LogUniforme	
	Uniforme
	Normale
	Exponentielle
	LogNormale
	LogUniforme

Source	Target
Exponentielle	
	Exponentielle

Fig. 3.5 : Variable transformations available in the "randvar" class

A uniform value $\overline{X}$ in the range $[0, 1[$ is computed from

$$\overline{X} = \frac{n}{N}, \tag{3.25}$$

where $N = RAND\_MAX$ and $n \in [0, RAND\_MAX[$.

3.3.2 Normale law random number generation

3.4 Variable transformations

In this section, we present the transformation of uniform random variables into other types of variables. We begin the analysis by a presentation of the theory required to perform transformations. Then we present the transformations which are provided by the library.

The transformations which are available in the "randvar" class are presented in figure 3.5.

3.4.1 Functions of a random variable

In this section, we present a theorem which allows to compute the c.d.f. of a transformed variable.

Proposition 3.4.1 *Let X be a continuous random variable associated with the cumulative density function F_X. Assume that $\phi : \mathbb{R} \to \mathbb{R}$ is a strictly increasing function on the range of X. Let us define the random variable Y by the equation*

$$Y = \phi(X). \tag{3.26}$$

Therefore, the cumulative density function F_Y of the variable Y satisfies

$$F_Y(y) = F_X(\phi^{-1}(y)). \tag{3.27}$$

If ϕ is strictly decreasing on the range of X, therefore the cumulative density function F_Y satisfies

$$F_Y(y) = 1 - F_X(\phi^{-1}(y)). \tag{3.28}$$

Proof Assume that ϕ is an increasing function. By definition of the cumulative density function F_Y satisfies

$$\begin{aligned} F_Y(y) &= P(Y \leq y) && (3.29)\\ &= P(\phi(X) \leq y) && (3.30)\\ &= P(X \leq \phi^{-1}(y)) && (3.31)\\ &= F_X(\phi^{-1}(y)), && (3.32) \end{aligned}$$

which concludes the first part of the proof. Assume now that ϕ is a decreasing function. By definition of the cumulative density function F_Y satisfies

$$\begin{aligned} F_Y(y) &= P(Y \leq y) && (3.33)\\ &= P(\phi(X) \leq y) && (3.34)\\ &= P(X \geq \phi^{-1}(y)) && (3.35)\\ &= 1 - P(X < \phi^{-1}(y)) && (3.36)\\ &= 1 - F_X(\phi^{-1}(y)), && (3.37) \end{aligned}$$

which concludes the proof. ∎

3.4.2 Transformation from Uniform to other c.d.f.

The previous theorem leads to an important application, which allows to transform any uniform random variable into a target variable.

Proposition 3.4.2 *Assume that X is a uniform variable in the interval $[0, 1]$. Assume that F_Y is a given cumulative distribution function. It implies that F_Y is strictly increasing, so that the variable $Y = \phi(X) = F_Y^{-1}(X)$ is defined. Therefore, the variable Y is associated with the distribution function F_Y.*

Proof Let us denote by G_Y the distribution function of the variable Y. Recall that the uniform cumulative distribution function is $F_X(x) = x$. By proposition 3.4.1, we have

$$\begin{aligned} G_Y(y) &= F_X(\phi^{-1}(y)) && (3.38)\\ &= \phi^{-1}(y) && (3.39)\\ &= F_Y(y), && (3.40) \end{aligned}$$

which concludes the proof. ∎

3.4.3 Inverse of the standard Normale c.d.f.

In this section, we present the standard Normale cumulative distribution function and analyse how the inverse of this function is computed in the library. Indeed, the inverse of the standard Normale c.d.f. is required in many variable transformations.

Consider the Normale cumulative distribution function with parameters μ and $\sigma > 0$, denoted by $F_{\mu,\sigma}$ and defined by

$$F_{\mu,\sigma}(x) = \int_{-\infty,x} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(t-\mu)^2}{2\sigma^2}\right) dt. \tag{3.41}$$

The *standard* Normale cumulative distribution function is denoted by $F_{0,1}(x)$ and is associated with $\mu = 0$ and $\sigma = 1$.

The standard Normale cumulative distribution function is computed from the error function erf with

$$F_{0,1}(x) = \frac{1}{2} + \frac{1}{2}\,\mathrm{erf}\left(\frac{x}{\sqrt{2}}\right). \tag{3.42}$$

The erf function is computed by a polynomial expansion which guarantees a minimum relative relative error on the value of the function.

The inverse function $F_{0,1}^{-1}$ is required in variable transformations, in particular to transform a uniform random variable into a standard Normale variable. Indeed, assume that X is a uniform variable in the interval $[0, 1]$. Therefore, the variable

$$Y = F_{0,1}^{-1}(X) \tag{3.43}$$

is standard Normale variable.

We now present the Newton method which allow to compute the inversion of the function $F_{0,1}^{-1}$. Given the variable $Y \in [0, 1]$, we want to find $X \in \mathbb{R}$ as the solution of the equation

$$F_{0,1}(X) - Y = 0. \tag{3.44}$$

Assume that X^0 is an initial guess for the solution and consider the sequence

$$X^{n+1} = X^n + \Delta X, \tag{3.45}$$

where n is a positive integer and $\Delta X \in \mathbb{R}$ is unknown. Newton's method proceeds by considering a linear approximation of the function $F_{0,1}$ is the neighbourhood of X^n. We have

$$F_{0,1}(X^{n+1}) \approx F_{0,1}(X^n) + \Delta X F'_{0,1}(X^n). \tag{3.46}$$

The method computes ΔX so that the following equation is satisfied

$$F_{0,1}(X^n) + \Delta X F'_{0,1}(X^n) = Y, \tag{3.47}$$

which leads to

$$\Delta X = \frac{Y - F_{0,1}(X^n)}{F'_{0,1}(X^n)}. \tag{3.48}$$

Once ΔX is determined the iterate X^{n+1} is computed from

$$X^{n+1} = X^n + \Delta X. \tag{3.49}$$

The derivative of the function $F_{0,1}$ is

$$F'_{0,1}(X) = f_{0,1}(X) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{X^2}{2}\right), \tag{3.50}$$

where $f_{0,1}$ is the density function of the Normale law.

3.4.4 Transformations from Uniforme law

Assume that X is a uniform random variable in the interval $[a, b]$. The first step of the transformation is a *scaling*, which allows to remove the dependency from the parameters a, b of the original random variable X. To accomplish this, we compute a random number $\overline{X}$ uniform in the interval $[0, 1]$ by

$$\overline{X} = \frac{X - a}{b - a}. \tag{3.51}$$

Given a target distribution function, the goal is to compute a variable Y defined by

$$Y = g(\overline{X}), \tag{3.52}$$

where g is a function which is to be defined. The function g depends on the target distribution function and we will present one function g for each target distribution.

From Uniforme Into Uniforme Assume that the target distribution function is the LogUniforme law, with parameters $a', b' > 0$ and $a < b$. The variable

$$Y \quad = \quad a' + (b' - a')\overline{X} \tag{3.53}$$

is associated with a Uniforme distribution function with parameters a', b'.

From Uniforme Into LogUniforme Assume that the target distribution function is the LogUniforme law, with parameters $a', b' > 0$ and $a < b$. The variable

$$\tilde{X} = \ln(a') + (\ln(b') - \ln(a'))\overline{X} \tag{3.54}$$

is associated with a uniform distribution function in the interval $[\ln(a'), \ln(b')]$. The variable

$$\begin{aligned} Y &= \exp(\tilde{X}) && (3.55)\\ &= \exp(\ln(a') + (\ln(b') - \ln(a'))\overline{X}) && (3.56) \end{aligned}$$

is associated with a LogUniforme distribution function with parameters a', b'.

From Uniforme Into Normale Assume that the target distribution function is the Normale law, with parameters μ and $\sigma > 0$.

The variable

$$\tilde{X} = F_{0,1}^{-1}\left(\overline{X}\right) \tag{3.57}$$

is associated with a standard Normale cumulative distribution function. The inversion of the function $F_{0,1}^{-1}$ is based on the algorithm presented in section 3.4.3.

It can be prooved that the variable

$$\begin{aligned} Y &= \mu + \sigma\tilde{X} && (3.58)\\ &= \mu + \sigma F_{0,1}^{-1}\left(\overline{X}\right) && (3.59) \end{aligned}$$

is associated with a Normale c.d.f. with parameters μ and σ.

Indeed, consider the function ϕ defined by

$$\phi(\tilde{X}) = \mu + \sigma\tilde{X}. \tag{3.60}$$

The inverse function ϕ^{-1} satisfies the equality

$$\phi^{-1}(y) = \frac{y - \mu}{\sigma}. \tag{3.61}$$

By hypothesis, we have $\sigma > 0$ which implies that the function ϕ is increasing. The proposition 3.4.1 states that Y is associated with the c.d.f. defined by

$$F_Y(y) = F_{\tilde{X}}(\phi^{-1}(y)), \tag{3.62}$$

where $F_{\tilde{X}}$ is the standard Normale c.d.f.. We have

$$F_Y(y) = \int_{-\infty}^{\frac{y-\mu}{\sigma}} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{s^2}{2}\right) ds. \tag{3.63}$$

Let us consider the change of variable

$$t = \sigma s + \mu. \tag{3.64}$$

The previous equality can be transformed into

$$s = \frac{t - \mu}{\sigma}, \tag{3.65}$$

which leads to $ds = \frac{1}{\sigma}dt$. If we plug this equality into 3.63, we get

$$F_Y(y) = \int_{-\infty}^{y} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{t-\mu}{\sigma}\right)^2\right) dt. \tag{3.66}$$

The previous equality is the definition of the Normale c.d.f. with parameters μ and σ.

3.5 References and notes

The definitions and notations in section 3.1 are inspired from [1], chapter 2, section 2.2 "Continuous Density Functions". The proposition 3.4.1 and its proof are given in [1], chapter 5, section 5.2, "Functions of a Random Variable".

Chapter 4

The ”setrandvar” class

Chapter 5

The "polychaos" class

Chapter 6

Thanks

Many thanks to Allan Cornet and Bernard Hugueney, who helped me many times in the creation of this toolbox.

Bibliography

[1] M. Grinstead, Charles and Laurie Snell, J. *Introduction to probabilities, Second Edition.* American Mathematical Society, 1997.

[2] Didier Pelat. *Bases et méthodes pour le traitement des données (Bruits et Signaux).* Master M2 Recherche : Astronomie?astrophysique, 2006.

Index