

The `mc2d` package.

R. POUILLOT, M.-L. DELIGNETTE-MULLER, D.L. KELLY & J.-B. DENIS

February 2, 2010

This documentation is intended for readers with:

- A medium level of experience in R. Please refer to the Manual An Introduction to R available with R distribution if needed;
- Some knowledge about Monte-Carlo simulation (its basic principles and its utility) and about Quantitative Risk Assessment (QRA).

This documentation will not describe all arguments of the functions. The definitive reference remains the documentation associated with the package.

Contents

1	Introduction	2
1.1	What is <code>mc2d</code> ?	2
1.2	What is Two-Dimensional Monte-Carlo Simulation (briefly)?	2
1.3	A basic example	5
1.3.1	One Dimensional Monte-Carlo Simulation	6
1.3.2	Two dimensional Monte-Carlo Simulation	7
2	Basic Principles and Functions	10
2.1	Preliminary Step	10
2.2	The <code>mcnode</code> Object as an Elementary Object.	10
2.2.1	<code>mcnode</code> Object Structure	10
2.2.2	The <code>mcstoc</code> function	11
2.2.3	The <code>mcdata</code> function	14
2.2.4	Operations on an <code>mcnode</code>	14
2.2.5	The <code>mcprobtrees</code> function	15
2.2.6	Other functions for constructing an <code>mcnode</code>	16
2.2.7	Specifying a correlation between <code>mcnodes</code>	16
2.3	The <code>mc</code> Object	17
2.3.1	The <code>mc</code> Function	17
2.3.2	The <code>mcmodel</code> and the <code>evalmcmmod</code> Functions	17
2.3.3	The <code>mcmodelcut</code> and the <code>evalmccut</code> Functions	18
2.4	Analysing an <code>mc</code> Object	19

2.4.1	The <code>summary</code> Function	19
2.4.2	The <code>hist</code> Function	20
2.4.3	The <code>plot</code> function	20
2.4.4	The <code>tornado</code> function	21
2.4.5	The <code>tornadounc</code> function	23
2.5	Other Functions and <code>mc</code> Objects	23
3	Multivariate Nodes	24
3.1	Multivariate Nodes for Multivariate Distributions	24
3.2	Multivariate Nodes as a Third Dimension for Multiple Options in a Model	27
3.3	Multivariate Nodes as a Third Dimension for Multiple Vectors/Contaminants	29
4	Another Example: A QRA of Waterborne Cryptosporidiosis in France	32
4.1	Tap Water Consumption Model	32
4.2	The Dose-Response Model	36
4.3	The Model	37

1 Introduction

1.1 What is mc2d?

`mc2d` means Two-Dimensional Monte-Carlo (*Monte-Carlo à Deux Dimensions*).
This package :

- provides additional probability distributions;
- provides tools to construct One-Dimensional and Two-Dimensional Monte-Carlo Simulations;
- provides tools to analyse One-Dimensional and Two-Dimensional Monte-Carlo Simulations.

In a previous version, some tools to fit parametric distributions to data were included. Because these functions are useful for other purposes, they have been moved to a separate package called `fitdistrplus`.

`mc2d` was built for QRA in the Food Safety domain but it can be used in other QRA domains.

1.2 What is Two-Dimensional Monte-Carlo Simulation (briefly)?

The following text and Figure 1 are adapted from [4] and [5] where this method was used. The principal reference for Two-Dimensional Monte-Carlo simulation remains [2].

According to international recommendations, a QRA should reflect the variability in the risk and calculate the uncertainty associated with the risk estimate.

The variability represents temporal, geographical and/or individual heterogeneity of the risk for a given population. The uncertainty is understood as stemming from a lack of perfect knowledge about the QRA model structure and associated parameters¹.

In order to estimate the natural variability of the risk, a Monte-Carlo simulation approach may be useful: the empirical distribution of the risk within the population may be estimated from the mathematical combination of distributions reflecting the variability of parameters across the population.

A two-dimensional (or second-order) Monte-Carlo simulation was proposed to estimate the uncertainty in the risk estimates stemming from parameter uncertainty [2]. A two-dimensional Monte-Carlo simulation is a Monte-Carlo simulation where the distributions reflecting variability and the distributions representing uncertainty are sampled separately in the simulation, so that variability and uncertainty in the output may be estimated separately. It may be described as following (see Figure 1):

1. The parameters of the model should be divided into three categories: the parameters whose distributions reflect variability only, hereinafter denoted as variable parameters, the parameters whose distributions reflect uncertainty only, denoted as uncertain parameters and the parameters whose distributions reflect both uncertainty and variability. For this latter category, a hierarchical structure, using hyper-parameters, should be specified: if a parameter is both uncertain and variable, one should be able to specify an empirical or parametric distribution representing variability. This distribution is conditional upon other parameters for which there is some associated uncertainty. As an example, one should be able to specify a relationship such as

$$X|a,b \sim N(a,b)$$

where the specified normal distribution represents variability in x conditional upon parameters a and b . Hyperdistributions, such as

$$a \sim Unif(l_a, u_a)$$

and

$$b \sim Unif(l_b, u_b)$$

represent the uncertainty in the parameters a and b ;

2. A set of uncertain parameters are randomly sampled from their respective distributions;
3. The QRA is performed using a classical (one-dimensional) Monte-Carlo simulation of size N_v , treating the uncertain parameters as fixed. This QRA takes into account the variability in all variable parameters, and

¹In the engineering risk community, these concepts are referred as aleatoric uncertainty for variability and epistemic uncertainty for uncertainty.

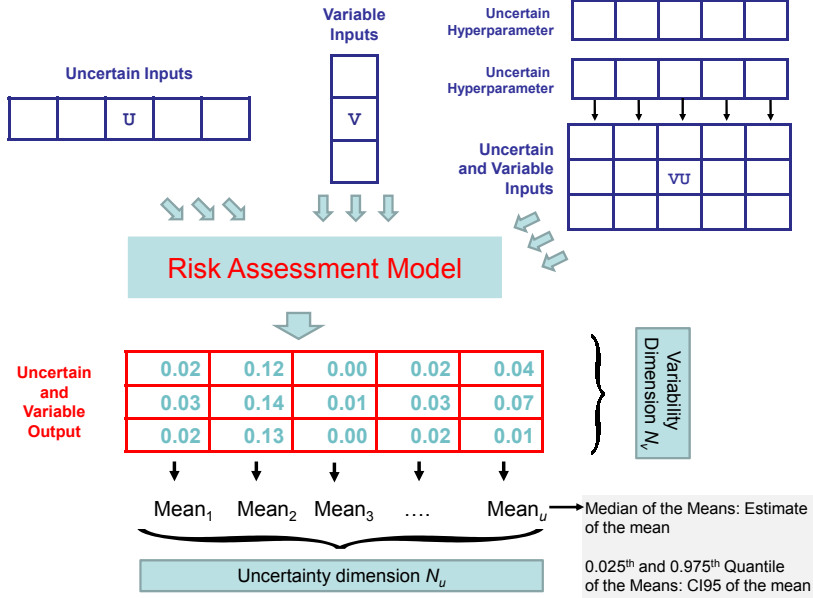


Figure 1: Schematic Representation of a Two-Dimensional Monte-Carlo Simulation.

leads to an empirical density function reflecting the variability of exposure/risk across the population, conditional upon the uncertain parameters. Various statistics (*e.g.* the mean, the standard deviation, percentiles) of the resulting empirical density function are evaluated and stored;

4. Steps 2) and 3) are performed a large number (N_u) of times, leading to N_u sets of statistics;
5. As output, the 50th percentile (median) of each statistic is used as a point estimate of this statistic; the 2.5th and 97.5th percentiles of each statistic are used to establish a 95% credible interval (CI95) of this statistic. The median of the N_u estimated values for each of the 101 estimated percentiles allows us to display a variability cumulative distribution via a graph. This curve is surrounded by the 2.5th and 97.5th percentiles obtained from the N_u estimates of each of the 101 percentiles.

`mc2d` is a set of R functions that will help to develop such two-dimensional Monte-Carlo simulations. The main difference from the procedure described above is that `mc2d` uses arrays of (at least) two dimensions to derive the results: the first dimension will reflect variability, the second will reflect uncertainty.

This document will not develop the method further, but will illustrate the practical application of `mc2d`, using a fictitious example.

1.3 A basic example

Quantitative Risk Assessment: *Escherichia coli* O157:H7 infection linked to the consumption of frozen ground beef in <3 year old children.

- We assume that, in a given batch of ground beef, *E. coli* O157:H7 are randomly distributed with a mean concentration of $c = 10$ bacteria (cfu) per gram of product;
- We assume that no bacterial growth occurs in storage, since the product is kept frozen until it is cooked, just before consumption;
- 2.7% of consumers cook their beef rare, 37.3% medium and 60.0% well done;
- The following bacterial inactivation i is associated with these cooking practices:
 - No inactivation for rare cooking;
 - 1/5 surviving bacteria for a medium cooking;
 - 1/50 surviving bacteria for a well done cooking.
- The variability in steak serving sizes s for <3 year children was estimated in a consumption survey. The best fit to the data was a gamma distribution with parameters: $shape = 3.93$, $rate = 0.0806$.
- The dose-response relationship, describing the probability of illness, P , according to the dose is a one-hit model. The probability of illness per hit r is assumed to be constant with $r = 0.001$.

The question is: *What is the distribution of the risk of illness in the population that consumed the contaminated lot?*

This distribution will be estimated using Monte-Carlo simulations performed with R via the `mc2d` package. First, the model will be developed in a one dimensional framework. Then, in order to include some uncertainties in the model, it will be derived in a two dimensional framework.

1.3.1 One Dimensional Monte-Carlo Simulation

As a first step, we assume that no uncertainty exists in our model. All distributions represent variability only. The model may be written as:

$$\begin{aligned} c &= 10. \\ i &\sim emp(\{1, 1/5, 1/50\}, \{0.027, 0.373, 0.600\}) \\ s &\sim gamma(3.93, 0.0806) \\ n &\sim Poisson(c \times i \times s) \\ P &= 1 - (1 - 0.001)^n \end{aligned}$$

where $emp(X, P)$ is an empirical distribution wherein each value X_i is associated with a probability P_i . We will use a classical one dimensional Monte-Carlo simulation, with 1,000 iterations. Using the `mc2d` package, the model may be written as:

```
> library(mc2d)
> ndvar(1000)

[1] 1000

> conc <- 10
> cook <- mcstoc(rempiricalD, values = c(1, 1/5, 1/50), prob = c(0.027,
+ 0.373, 0.6))
> serving <- mcstoc(rgamma, shape = 3.93, rate = 0.0806)
> expo <- conc * cook * serving
> dose <- mcstoc(rpois, lambda = expo)
> r <- 0.001
> risk <- 1 - (1 - r)^dose
> EC1 <- mc(cook, serving, expo, dose, risk)
> print(EC1)
```

	node	mode	nsv	nsu	nva	variate	min	mean	median	max	Nas	type
1	cook	numeric	1000	1	1	1	0.02	0.1165	0.0200	1.000	0	V
2	serving	numeric	1000	1	1	1	5.17	48.4451	44.0195	219.976	0	V
3	expo	numeric	1000	1	1	1	1.03	56.2452	14.1530	935.189	0	V
4	dose	numeric	1000	1	1	1	0.00	56.0520	15.0000	938.000	0	V
5	risk	numeric	1000	1	1	1	0.00	0.0507	0.0149	0.609	0	V

```
outm
1 each
2 each
3 each
4 each
5 each

> summary(EC1)
```

```

cook :
      mean      sd  Min 2.5%  25%  50%  75% 97.5% Max  nsv Na's
NoUnc 0.116 0.176 0.02 0.02 0.02 0.02 0.2      1   1 1000    0

serving :
      mean      sd  Min 2.5%  25%  50%  75% 97.5% Max  nsv Na's
NoUnc 48.4 24.3 5.17 14.5 29.8 44 62.6 103 220 1000    0

expo :
      mean      sd  Min 2.5%  25%  50%  75% 97.5% Max  nsv Na's
NoUnc 56.2 96.8 1.03 3.5 8.11 14.2 79.1 229 935 1000    0

dose :
      mean      sd  Min 2.5%  25%  50%  75% 97.5% Max  nsv Na's
NoUnc 56 96.3 0 2 7 15 79 226 938 1000    0

risk :
      mean      sd  Min 2.5%  25%  50%  75% 97.5% Max  nsv Na's
NoUnc 0.0507 0.0755 0 0.002 0.00698 0.0149 0.076 0.203 0.609 1000    0

```

This One-Dimensional Monte-Carlo simulation provides an estimate of the mean risk (approximately 5%), as well as some quantiles of the risk distribution (2.5% of the population has a risk of illness greater than 20.3%).

1.3.2 Two dimensional Monte-Carlo Simulation

Assume now that:

- The mean concentration of bacteria in the batch is not known with certainty, but was only a point estimate. Microbiologists think that the uncertainty around this estimate can be represented via a normal distribution with parameters $\mu = 10$ and $\sigma = 2$;
- Epidemiological studies suggest that the r parameter is also uncertain. The uncertainty around the mean value of 0.001 can be represented with a uniform distribution between 0.0005 and 0.0015.

The model could then be written as:

$$\begin{aligned}
 c &\sim N(10, 2) \\
 i &\sim emp(\{1, 1/5, 1/50\}, \{0.027, 0.373, 0.600\}) \\
 s &\sim gamma(3.93, 0.0806) \\
 n &\sim Poisson(c \times i \times s) \\
 r &\sim Unif(0.0005, 0.0015) \\
 P &= 1 - (1 - r)^n
 \end{aligned}$$

Note that the distributions of r and c represent uncertainty, while the distributions of i and s represent variability. n , which is a function of c , i and s , will be both variable *and* uncertain.

We will use a two-dimensional Monte-Carlo simulation, with 1,000 iterations in the variability dimension and 100 iterations in the uncertainty dimension. Using the `mc2d` package, the model may be written as:

```
> ndunc(100)

[1] 100

> conc <- mcstoc(rnorm, type = "U", mean = 10, sd = 2)
> cook <- mcstoc(rempiricalD, type = "V", values = c(1, 1/5, 1/50),
+   prob = c(0.027, 0.373, 0.6))
> serving <- mcstoc(rgamma, type = "V", shape = 3.93, rate = 0.0806)
> expo <- conc * cook * serving
> dose <- mcstoc(rpois, type = "VU", lambda = expo)
> r <- mcstoc(runif, type = "U", min = 5e-04, max = 0.0015)
> risk <- 1 - (1 - r)^dose
> EC2 <- mc(conc, cook, serving, expo, dose, r, risk)
> print(EC2, digits = 2)
```

	node	mode	nsv	nsu	nva	variate	min	mean	median	max	Nas	type
1	conc	numeric	1	100	1	1	5.55771	9.9e+00	9.7214	1.7e+01	0	U
2	cook	numeric	1000	1	1	1	0.02000	1.1e-01	0.0200	1.0e+00	0	V
3	serving	numeric	1000	1	1	1	2.66586	5.0e+01	45.0430	1.6e+02	0	V
4	expo	numeric	1000	100	1	1	0.70535	5.3e+01	13.7118	1.7e+03	0	VU
5	dose	numeric	1000	100	1	1	0.00000	5.3e+01	14.0000	1.7e+03	0	VU
6	r	numeric	1	100	1	1	0.00051	9.6e-04	0.0009	1.5e-03	0	U
7	risk	numeric	1000	100	1	1	0.00000	4.6e-02	0.0136	8.4e-01	0	VU

```
  outm
1 each
2 each
3 each
4 each
5 each
6 each
7 each

> summary(EC2)

conc :
      NoVar
median  9.72
mean    9.94
2.5%    5.96
97.5%   14.46
```



```

cook :
      mean      sd  Min 2.5% 25% 50% 75% 97.5% Max  nsv Na's
NoUnc 0.107 0.166 0.02 0.02 0.02 0.02 0.2  0.22   1 1000   0

serving :
      mean      sd  Min 2.5% 25% 50% 75% 97.5% Max  nsv Na's
NoUnc 49.7 24.9 2.67 13.6 31 45 64.2 110 161 1000   0

expo :
      mean      sd  Min 2.5% 25% 50% 75% 97.5% Max  nsv Na's
median 51.9 94.2 1.234 3.06 7.87 13.58 71.8 240 938 1000   0
mean 53.1 96.3 1.261 3.12 8.04 13.89 73.4 245 959 1000   0
2.5% 31.8 57.8 0.756 1.87 4.82 8.33 44.0 147 575 1000   0
97.5% 77.2 140.2 1.836 4.55 11.71 20.21 106.8 357 1396 1000   0

dose :
      mean      sd  Min 2.5% 25% 50% 75% 97.5% Max  nsv Na's
median 51.9 94.7 0.00 2 7.00 14.0 71.8 242 958 1000   0
mean 53.1 96.7 0.04 2 7.53 14.2 73.4 245 964 1000   0
2.5% 31.7 57.8 0.00 1 4.47 9.0 43.5 146 573 1000   0
97.5% 77.6 140.7 1.00 3 11.00 20.5 107.8 355 1379 1000   0

r :
      NoVar
median 0.000902
mean 0.000962
2.5% 0.000525
97.5% 0.001459

risk :
      mean      sd      Min      2.5%      25%      50%      75% 97.5% Max  nsv
median 0.0445 0.0703 0.00e+00 0.001713 0.00687 0.01298 0.0645 0.2027 0.589 1000
mean 0.0455 0.0706 3.88e-05 0.001902 0.00717 0.01347 0.0674 0.2061 0.582 1000
2.5% 0.0191 0.0324 0.00e+00 0.000583 0.00282 0.00538 0.0271 0.0841 0.290 1000
97.5% 0.0730 0.1057 7.08e-04 0.004115 0.01226 0.02242 0.1116 0.3259 0.788 1000
      Na's
median 0
mean 0
2.5% 0
97.5% 0

```

Note that the syntax is similar to the earlier model. However, a `type` argument is provided for each distribution, indicating whether the parameter distribution represents uncertainty (`type=U`), variability (`type=V`), or both (`type=VU`).

The summary provides estimates of the variability distributions (in rows) but with a measure of their uncertainty, linked to the uncertainty around `conc` and `r`. The estimate of the mean risk is now uncertain. The median of the 100 simulations leads to a best estimate of 0.0445, with a 95% credible interval of [0.191, 0.0730].

2 Basic Principles and Functions

A typical session of R using `mc2d` is as follows:

- From data, expert knowledge, *etc.* an empirical or parametric distribution is chosen for each parent parameter. For developing an empirical distribution from data, the `fitdistrplus` package is recommended;
- For each parameter, an `mcnode` object is constructed (key functions: `mcdata`, `mcstoc`);
- Various `mcnode` objects are grouped into an `mc` object (key function: `mc`).
- The `mc` object is studied through summaries, graphs, and sensitivity analysis (key functions: `summary.mc`, `plot.mc`, `tornado`, `tornadounc`).

2.1 Preliminary Step

The `mc2d` library should be loaded at the beginning of your R session (`library(mc2d)`).

The default size of the Monte-Carlo Simulation should be defined using the `ndvar()` function (dimension of variability) and the `ndunc()` function (dimension of uncertainty).

2.2 The `mcnode` Object as an Elementary Object.

2.2.1 `mcnode` Object Structure

An `mcnode` object is the basic element of an `mc` object. It is an array of dimension `nsv * nsu * nvariables` where `nsv` is the dimension of variability, `nsu` is the dimension of uncertainty and `nvariables` is the number of variates of the `mcnode`². Four types of `mcnode` exist:

- **V `mcnode`**, for *Variability*, is an array of dimension $(nsv \times 1 \times nvariables)$. The distribution represents variability in the parameter;
- **U `mcnode`**, for *Uncertainty*, is an array of dimension $(1 \times nsu \times nvariables)$. The distribution represents uncertainty in the parameter.
- **VU `mcnode`**, for *Variability and Uncertainty*, is an array of dimension $(nsv \times nsu \times nvariables)$. The distribution represents both variability (in the first dimension) and uncertainty (in the second dimension) in the parameter.

²In this section, we will only consider `mcnodes` with `nvariables` = 1.

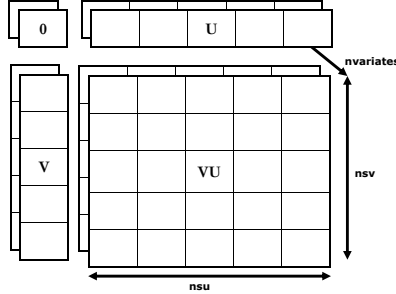


Figure 2: Structure of the various `mcnode` objects.

- Additionally, a `0 mcnnode` is also defined. `0` stands for Neither Variability or Uncertainty. Such nodes are arrays of dimension $(1 \times 1 \times nvariables)$. No uncertainty or variability is considered for these nodes. A `0 mcnnode` is not necessary in the univariate context (use a scalar instead) but is useful in constructing multivariate nodes (See section 3).

There are 5 ways to construct an `mcnode` object:

1. The `mcstoc` function constructs an `mcnode` from random number generating functions;
2. The `mcdata` function constructs an `mcnode` from data sets;
3. An `mcnode` can be constructed directly from operations on `mcnode` objects;
4. `mcprobtrees` is a special function that constructs an `mcnode` from other `mcnodes` using a probability tree;
5. Some functions, such as `==` or `>`, `is.na`, `is.finite` generate a new `mcnode` when applied to an existing `mcnode`.

2.2.2 The `mcstoc` function

The `mcstoc` function is written as³:

```
mcstoc(func=runif, type=c(V, U, VU, 0), ..., nsv=ndvar(),
nsu=ndunc(), nvariables=1, outm=each, nsample=n, seed=NULL,
rtrunc=FALSE, linf=-Inf, lsup=Inf, lhs=FALSE)
```

- `func` is a function providing random data or its name as a character. The table 1 provides available distributions from the `stats` and the `mc2d` libraries that can be used in `mcstoc`;

³as is standard in R, most arguments have logical default values and will be infrequently modified.

- **type** is the type of requested **mcnode**. By default, **mcstoc** constructs a **V mcnode**;
- ... are the arguments to be passed to the function **func**, with the exception of the argument providing the size of the sample. This latter is calculated by the function according to **func**, **type**, **nsv**, **nsu** and **nvariables**. If the name of the argument specifying the size of the sample is not **n** (e.g. functions **rhyper** and **rwilcox**, see table 1), the name of this parameter should be provided in the **nsample** argument. *Note that all of the following arguments should be named*;
- **nsv** and **nsu** are the number of samples needed in the variability and uncertainty dimension, respectively. By default, these values are the ones provided by **ndvar()** and **ndunc()**, respectively;
- **nvariables** is the desired number of variates in the **mcnode**;
- **outm** is the default output for multivariate nodes;
- **seed** optionally specifies a seed for the random number generator;
- **rtrunc** allows truncation of a distribution between **linf** and **lsup**. This function is not valid for every distribution (see table 1). See the **rtrunc** function help for further details;
- **lhs** allows Latin hypercube sampling of the node . This function is not valid for every distribution (see table 1). See the **lhs** function help for further details.

In our basic example, **mcstoc** was used to specify **conc** (a normal distribution), **cook** (an empirical discrete distribution), **serving** (a gamma distribution), and **dose** (a Poisson distribution). Note that the argument **lambda** of the Poisson distribution (node **dose**) is itself an **mcnode**.

```
> conc <- mcstoc(rnorm, type = "U", mean = 10, sd = 2)
> cook <- mcstoc(rempricalD, type = "V", values = c(1, 1/5, 1/50),
+   prob = c(0.027, 0.373, 0.6))
> serving <- mcstoc(rgamma, type = "V", shape = 3.93, rate = 0.0806)
> ...
> dose <- mcstoc(rpois, type = "VU", lambda = expo)
> r <- mcstoc(runif, type = "U", min = 5e-04, max = 0.0015)
> ...
```

A normal distribution with parameters $mean = 2$, $sd = 3$, truncated on the interval $[1.5, 2]$, with samples generated via Latin hypercube sampling could be written⁴:

⁴Note that the mean and the standard deviation of the untruncated normal distribution are not preserved in the truncated distribution.

Table 1: Available distributions

Package	Distribution	function	Parameter n	Other Parameters	trunc	ll
stats	beta	rbeta	n	shape1, shape2, ncp	Y	Y
	binomial	rbinom	n	size, prob	Y	Y
	Cauchy	rcauchy	n	location, scale	Y	Y
	chi-squared	rchisq	n	df, ncp	Y	Y
	exponential	rexp	n	rate	Y	Y
	F	rf	n	df1, df2, ncp	Y	Y
	gamma	rgamma	n	shape, rate (or scale)	Y	Y
	geometric	rgeom	n	prob	Y	Y
	hypergeometric	rhyper	nn	m, n, k	Y	Y
	lognormal	rlnorm	n	meanlog, sdlog	Y	Y
	logistic	rlogis	n	location, scale	Y	Y
	negative binomial	rnbinom	n	size, prob (or mu)	Y	Y
	normal	rnorm	n	mean, sd	Y	Y
	Poisson	rpois	n	lambda	Y	Y
	Student's t	rt	n	df, ncp	Y	Y
	uniform	runif	n	min, max	Y	Y
	Weibull	rweibull	n	shape, scale	Y	Y
	Wilcoxon	rwilcox	nn	m,n	Y	Y
mc2d	Bernoulli	rbern	n	prob	Y	Y
	empirical discrete	rempiricalD	n	values, prob	Y	Y
	empirical continuous	rempiricalC	n	min, max, values, prob	Y	Y
	PERT	rpert	n	min, mode, max, shape	Y	Y
	triangular	rtriang	n	min, mode, max	Y	Y
	generalised beta	rbetagen	n	shape1,shape2,min,max,ncp	Y	Y
	multinomial	rmultinomial	n	n, size, prob	N	N
	Dirichlet	rdirichlet	n	alpha	N	N
	multivariate normal	rmultinormal	n	mean, sigma	N	N

```
> x <- mcstoc(rnorm, mean = 2, sd = 3, rtrunc = TRUE, linf = 1.5,
+           lsup = 2, lhs = TRUE)
> summary(x)
```

```
node :
```

```
      mean      sd Min 2.5% 25% 50% 75% 97.5% Max  nsv Na's
NoUnc 1.75 0.144 1.5 1.51 1.63 1.75 1.88 1.99   2 1000   0
```

For convenience in using `mcstoc`, the following additional distributions have been implemented: the Bernoulli distribution (`rbern`), the empirical discrete distribution (`rempiricalD`), the PERT distribution (`rpert`) [6], the triangular distribution (`rtriang`), the Dirichlet distribution (`rdirichlet`) and the multivariate normal distribution (`rmultinormal`). The multinomial distribution has been adapted (vectorized): `rmultinomial` (library `mc2d`) should be used in place of `rmultinom` (library `stats`). The empirical discrete (*e.g.* for bootstrap), the Dirichlet, the multinomial and the multivariate normal may be used with uncertain and/or variable parameters by specifying multivariate nodes. See section 3.

2.2.3 The `mcddata` function

Another way to construct an `mcnode` object is *via* the `mcddata` function, when data are available.

```
mcddata(data, type=c(V, U, VU, 0), nsv=ndvar(), nsu=ndunc(),
nvariates=1, outm=each)
```

See the documentation associated with this function to see the size/type of data that can be used to construct an `mcnode`. The following example places a `TRUE` value in a `U` node in half of the simulations:

```
> nu <- ndunc()
> tmp <- (1:nu) > (nu/2)
> mcddata(tmp, type = "U")
```

```
 node      mode nsv nsu nva variate min mean median max Nas type outm
1    x logical   1 100   1         1  0  0.5   0.5   1  0    U each
```

2.2.4 Operations on an `mcnode`

`mcnodes` can be automatically constructed using operations on other `mcnodes`. Rules are used to transfer uncertainty and variability coherently within the model. Logically, the rules are as follows (illustrated here with a $+$)⁵:

- $0 + 0 = 0$;
- $0 + V = V$

⁵These rules are not the standard R rules for recycling.

- $0 + U = U$;
- $0 + VU = VU$;
- $V + V = V$;
- $V + U = VU$: the U mcnode is recycled by row, the V mcnode is recycled in the standard manner by column;
- $V + VU = VU$: the V mcnode is recycled in the standard manner by column;
- $U + U = U$;
- $U + VU = VU$: the U mcnode is recycled by row;
- $VU + VU = VU$

Thus, in our example:

```
> ...
> expo <- conc * cook * serving
> ...
> risk <- 1 - (1 - r)^dose
```

`expo` is a function of a U and two V mcnodes: it is a VU mcnode with variability in the row dimension and uncertainty in the column dimension. `risk` is a function of a U and a VU mcnode: it is therefore a VU mcnode.

2.2.5 The `mcprobtree` function

The `mcprobtree` function can be used if a probability tree is needed to construct an mcnode. Assume that the distribution representing the uncertainty on `conc` was not itself certain, and that the microbiologists suggest that they are 75% sure that $conc \sim N(10, 2)$ but that they are 25% sure that $conc \sim U(8, 12)$. This could be written using `mcprobtree` as⁶:

```
> conc1 <- mcstoc(rnorm, type = "U", mean = 10, sd = 2)
> conc2 <- mcstoc(runif, type = "U", min = 8, max = 12)
> whichdist <- c(0.75, 0.25)
> concbis <- mcprobtree(whichdist, list(`0` = conc1, `1` = conc2),
+   type = "U")
```

`mcprobtree` can also be used to generate samples from a mixture distribution for variability.

⁶two alternatives for `whichdist` are `whichdist <- mcstoc(rempiricalD, type=U, values=c(0,1), prob=c(75,25))` or `whichdist <- mcstoc(rbern,type=U,prob=0.25)`

2.2.6 Other functions for constructing an mcnode

The functions `==`, `<`, `<=`, `>=`, `>`, generate an `mcnode` when applied to another `mcnode`.

Special functions `is.na(x)`, `is.nan(x)`, `is.finite(x)`, `is.infinite(x)` are implemented to test if any values are NA (missing data), NaN (*Not A Number*), or finite .

```
> cook < 1

node   mode  nsv nsu nva variate min  mean median max Nas type outm
1    x logical 1000  1  1      1  0 0.975      1  1  0    V each

> tmp <- log(mcstoc(runif, min = -1, max = 1))
> tmp

node   mode  nsv nsu nva variate  min  mean median      max Nas type outm
1    x numeric 1000  1  1      1 -8.19 -1.03 -0.699 -0.00167 512    V each

> is.na(tmp)

node   mode  nsv nsu nva variate min  mean median max Nas type outm
1    x logical 1000  1  1      1  0 0.512      1  1  0    V each
```

2.2.7 Specifying a correlation between mcnodes

Structural links between sets of parameters may be very important in QRA. In `mc2d`, a Spearman rank correlation structure for 2 or more nodes may be specified with the `cornode` function. This function uses the method of Iman & Conover to generate correlated samples [3]. Assume that a study suggests that people who consume rare ground beef also consume larger serving sizes. We could specify this relation using:

```
> cornode(cook, serving, target = 0.5, result = TRUE)

output Rank Correlation per variates
variates: 1
[1] 1.0000000 0.3796997 0.3796997 1.0000000
$cook
node   mode  nsv nsu nva variate min  mean median max Nas type outm
1    x numeric 1000  1  1      1 0.02 0.107  0.02  1  0    V each

$serving
node   mode  nsv nsu nva variate min mean median max Nas type outm
1    x numeric 1000  1  1      1 2.67 49.7   45 161  0    V each
```

Note that the resulting correlation (around 0.4) is obviously an approximation to the desired value of 0.5, because a discrete distribution (`cook`: 3 categories) is correlated with a continuous distribution (`serving`).

It is possible to create such correlations between V nodes, between U nodes, between VU nodes, or between one V node and multiple VU nodes.

The use of a multivariate normal distribution (`rmultinormal`) is another way to specify correlations among nodes, assuming that the individual nodes are normally distributed.

2.3 The mc Object

Once the `mcnode` objects are constructed, one should group them into a single object in order to analyse the Monte-Carlo results. The `mc` object is a list of `mcnodes`. There are three ways to construct an `mc` object: using the `mc` function, using the `evalmcmmod` function, or within the `evalmccut` function.

2.3.1 The mc Function

`mc(..., name=NULL, devname=FALSE)`
`...` are `mcnodes` or `mc` objects to be gathered into an `mc` object. `mc` value is an `mc` object with specific methods, e.g. `print` or `summary`. In our example, we used:

```
> ...
> EC2 <- mc(conc, cook, serving, expo, dose, r, risk)
> print(EC2)
> summary(EC2)
```

2.3.2 The mcmodel and the evalmcmmod Functions

A model may be written in one step using `mcmodel` (just a wrapper of your model in a function), and then evaluated using `evalmcmmod`. These functions may be used once your model is correct and has been tested using a small number of iterations. For our example:

```
> modelEC3 <- mcmodel({
+   conc <- mcstoc(rnorm, type = "U", mean = 10, sd = 2)
+   cook <- mcstoc(rempricalD, type = "V", values = c(1, 1/5,
+     1/50), prob = c(0.027, 0.373, 0.6))
+   serving <- mcstoc(rgamma, type = "V", shape = 3.93, rate = 0.0806)
+   r <- mcstoc(runif, type = "U", min = 5e-04, max = 0.0015)
+   expo <- conc * cook * serving
+   dose <- mcstoc(rpois, type = "VU", lambda = expo)
+   risk <- 1 - (1 - r)^dose
+   mc(conc, cook, serving, expo, dose, r, risk)
+ })
> modelEC3

expression({
  conc <- mcstoc(rnorm, type = "U", mean = 10, sd = 2)
```

```

    cook <- mcstoc(rempiricalD, type = "V", values = c(1, 1/5,
      1/50), prob = c(0.027, 0.373, 0.6))
    serving <- mcstoc(rgamma, type = "V", shape = 3.93, rate = 0.0806)
    r <- mcstoc(runif, type = "U", min = 5e-04, max = 0.0015)
    expo <- conc * cook * serving
    dose <- mcstoc(rpois, type = "VU", lambda = expo)
    risk <- 1 - (1 - r)^dose
    mc(conc, cook, serving, expo, dose, r, risk)
  })
  attr("class")
[1] "mcmodel"

```

Note that:

- the model is wrapped between { and };
- any (valid) R code may be placed in the model⁷;
- The model should end with an `mc()` function.

The model is then evaluated using the `evalmcmmod` function:

```
evalmcmmod(expr, nsv=ndvar(), nsu=ndunc(), seed=NULL)
```

One can re-run the model with various dimensions or random seeds in one line:

```

> EC3 <- evalmcmmod(modelEC3, nsv = 100, nsu = 10, seed = 666)
> EC4 <- evalmcmmod(modelEC3, nsv = 100, nsu = 1000, seed = 666)

```

2.3.3 The `mcmmodelcut` and the `evalmccut` Functions

If evaluating a high-dimensional model, R may exceed its memory limit. `evalmccut` evaluates a 2-dimensional Monte-Carlo model (written with the `mcmmodelcut` function) using a loop, and calculates and stores statistics in the uncertainty dimension for further analysis. Readers should refer to the corresponding documentation for further details. Our example would be written as:

```

> modEC4 <- mcmmodelcut({
+   {
+     cook <- mcstoc(rempiricalD, type = "V", values = c(0,
+       1/5, 1/50), prob = c(0.027, 0.373, 0.6))
+     serving <- mcstoc(rgamma, type = "V", shape = 3.93, rate = 0.0806)
+     conc <- mcstoc(rnorm, type = "U", mean = 10, sd = 2)
+     r <- mcstoc(runif, type = "U", min = 5e-04, max = 0.0015)
+   }
+   {

```

⁷If needed, it is possible to make reference to the simulation dimensions using `ndvar()` and/or `ndunc()`.

```

+      expo <- conc * cook * serving
+      dose <- mcstoc(rpois, type = "VU", lambda = expo)
+      risk <- 1 - (1 - r)^dose
+      res <- mc(zero, conc, cook, serving, expo, dose, r, risk)
+    }
+    {
+      list(sum = summary(res), plot = plot(res, draw = FALSE),
+           minmax = lapply(res, range), tor = tornado(res),
+           et = sapply(res, sd))
+    }
+  })
> evalmccut(modEC4, nsu = 10001, nsu = 101, seed = 666)

```

Note that the use of a `tornado` function in the model should be avoided as it slows the `evalmccut` function considerably. The `tornado` function will be rewritten in the near future to improve its performance.

2.4 Analysing an mc Object

As a reminder, the `print` function provides a very basic summary of the `mc` object. It has a `digits` argument (default: 3). Obviously, other more informative functions are provided in the `mc2d` package.

2.4.1 The summary Function

The `summary` function provides statistics on an `mc` object:

```
summary(object, probs=c(0,0.025,0.25,0.5,0.75,0.975,1),
lim=c(0.025,0.975), ...)
```

The mean, the standard deviation and the quantiles provided in the `probs` arguments are evaluated on the variability dimension. Then, the median and the quantiles provided in the `lim` argument are evaluated on these statistics. Of course, these arguments should be changed if other quantiles are needed.

```

> tmp <- summary(EC2, probs = c(0.995, 0.999), digits = 12)
> tmp$risk

```

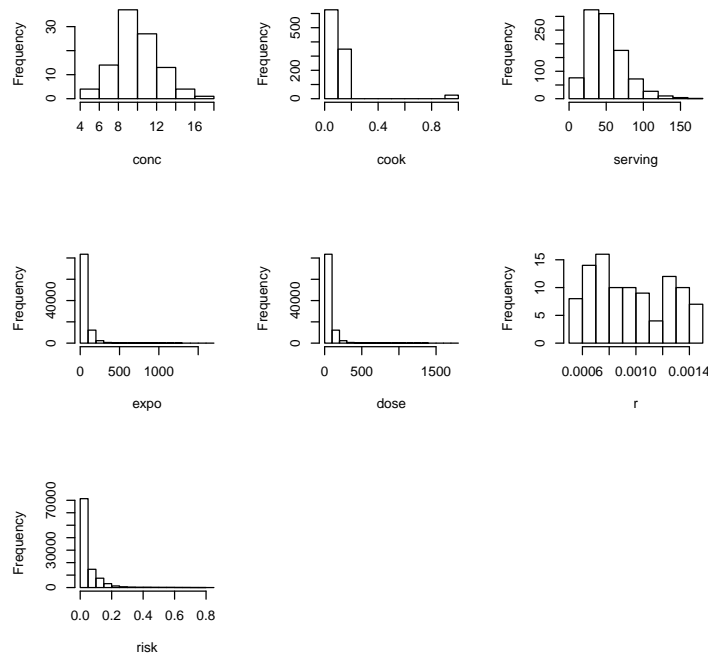
	mean	sd	99.5%	99.9%	nsu	Na's
median	0.04446930	0.07028198	0.5016035	0.5573356	1000	0
mean	0.04554518	0.07058057	0.4979376	0.5522955	1000	0
2.5%	0.01914973	0.03243724	0.2380268	0.2771793	1000	0
97.5%	0.07297994	0.10573336	0.7113664	0.7565299	1000	0

```

attr(,"type")
[1] "VU"

```

Figure 3: Function `hist`.



2.4.2 The `hist` Function

The `hist` provides a histogram of the different `mcnodes` making up the `mc` object (cf. Figure 3).

```
hist (x, griddim = NULL, xlab = names(x), ylab = Frequency, main = , ...)
```

In the current version, uncertainty and variability distributions are collapsed. Thus, the resulting histogram may be meaningless.

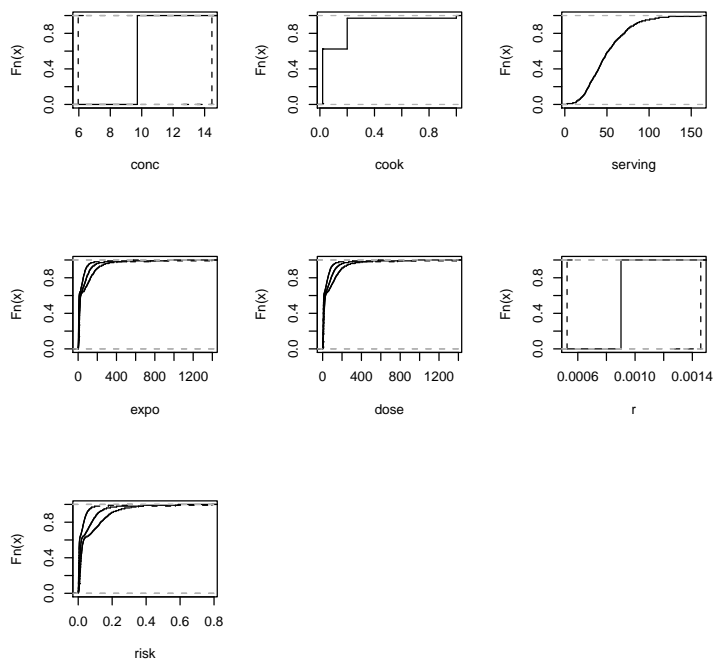
```
> hist(EC2)
```

2.4.3 The `plot` function

The `plot` function provides a graph of the empirical distribution function of the estimate (mean or median) of the quantiles.

```
plot (x, prec = 0.01, stat = c(median, mean), lim = c(0.025,0.975), na.rm = TRUE, griddim = NULL, xlab = NULL, ylab = Fn(x), main = , draw = TRUE, ...)
```

Figure 4: plot Function .



For our example, see Figure 4, a default graph.

```
> plot(EC2)
```

Note that `mcnode` objects have the same methods `print`, `summary`, `plot`, and `hist`.

2.4.4 The tornado function

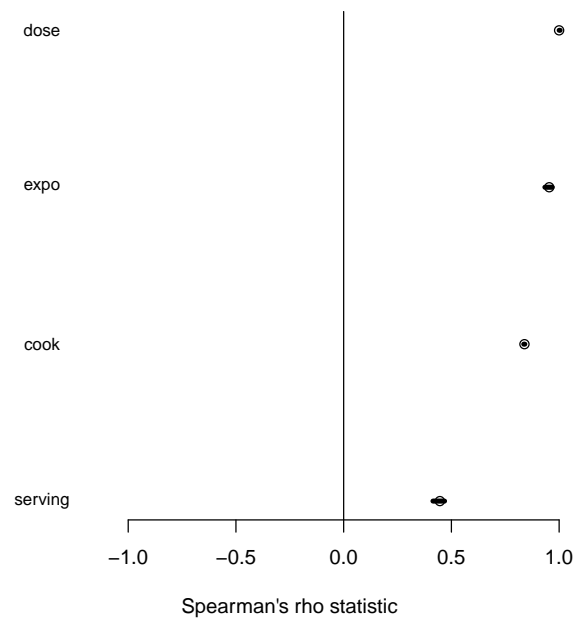
The `tornado` function calculates the Spearman (default) rank correlation between nodes of the `mc` object.

```
tornado(x, output=length(x), use=all.obs, method=c(spearman,
kendall,pearson), lim=c(0.025, 0.975))
```

where `output` is the `mcnode` (name or rank) of the output (default: the last `mcnode`). Missing data are treated using the `use` arguments (see the reference documentation). `tornado` creates a `tornado` object with a `plot` method (*cf.* Figure 5).

```
> torEC2 <- tornado(EC2)
> plot(torEC2)
```

Figure 5: `plot.tornado` Function .



2.4.5 The tornadounc function

The `tornadounc` function examines the impact of the uncertainty on the estimate of an output. It calculates the Spearman (default) rank correlation between statistics of the `mc` object in the variability dimension.

```
tornadounc(mc,output = length(mc), quant=c(0.5,0.75,0.975), use =  
all.obs, method=c(spearman,kendall,pearson), ...)
```

The `quant` argument indicates which quantiles should be used in the variability dimension. `tornadounc` creates a `tornadounc` object with a `plot` method

```
> tornadounc(EC2, output = "risk", quant = 0.99)
```

```
Tornado on uncertainty  
Spearman's rho statistic  
Output: risk  
$risk
```

	conc	mean expo	sd expo	99% expo	mean dose	sd dose	99% dose
mean risk	0.5244044	0.5244044	0.5244044	0.5244044	0.5220402	0.5235644	0.4966712
sd risk	0.5233123	0.5233123	0.5233123	0.5233123	0.5210321	0.5229043	0.4958971
99% risk	0.5233603	0.5233603	0.5233603	0.5233603	0.5214761	0.5233003	0.5088284

r

mean risk	0.7727933
sd risk	0.7728413
99% risk	0.7607921

The output shows the impact of the uncertain nodes (type U nodes) and some statistics (mean, median and, here, the 99th percentile) calculated on the variability dimension (type V and type VU nodes) of some output statistics .

2.5 Other Functions and mc Objects

`mc` objects are simply lists of three dimensional arrays; within each array, values in a given column represent variability in the parameter.

Knowing the structure of the `mc` and the structure of the `mcnode` objects, it is straightforward to apply any R function to these objects. The `$` function is helpful for extracting an `mcnode` from an `mc` object. The `unmc` function removes all attributes, classes, and dimensions equal to one, providing a list of vectors, matrices and/or arrays.

Here is a (silly) example building a linear model (in fact `ndunc()` linear models) between the `risk` and the `dose` within each uncertainty dimension and estimating some statistics for the coefficients. This example is here only to illustrate that the entire spectrum of R functionality is available for your analysis.

```
> tmp <- unmc(EC2, drop = TRUE)  
> dimu <- ncol(tmp$risk)
```

```
> coef <- sapply(1:dimu, function(x) lm(tmp$risk[, x] ~ tmp$dose[,
+   x])$coef)
> apply(coef, 1, summary)
```

```
      (Intercept) tmp$dose[, x]
Min.      0.0007991      0.0004028
1st Qu.    0.0038060      0.0005948
Median     0.0064130      0.0007084
Mean       0.0072600      0.0007334
3rd Qu.    0.0092290      0.0008837
Max.       0.0206100      0.0011200
```

3 Multivariate Nodes

The dimension `nvariables` is the third dimension of the `mcnode`. One can ignore it while using `mc2d`. Nevertheless, its use is mandatory to handle some multivariate distributions, and it may be useful in other circumstances. Constructing multivariate nodes is straightforward. We note that the following code:

```
> mcstoc(runif, nvariables = 3, min = c(1, 2, 3), max = 4)
```

	node	mode	nsv	nsu	nva	variate	min	mean	median	max	Nas	type	outm
1	x	numeric	1000	1	3	1	1.00	3.03	3.22	4	0	V	each
2	x	numeric	1000	1	3	2	1.00	3.00	3.18	4	0	V	each
3	x	numeric	1000	1	3	3	1.03	3.04	3.23	4	0	V	each

will logically not provide a node with 3 variates, each having a different limit. The recycling rule says that `c(1, 2, 3)` will be used in the first dimension, i.e. the variability dimension. Use instead:

```
> lim <- mcdata(c(1, 2, 3), type = "0", nvariables = 3)
> mcstoc(runif, nvariables = 3, min = lim, max = 4)
```

	node	mode	nsv	nsu	nva	variate	min	mean	median	max	Nas	type	outm
1	x	numeric	1000	1	3	1	1.00	2.48	2.47	4	0	V	each
2	x	numeric	1000	1	3	2	2.00	3.00	3.00	4	0	V	each
3	x	numeric	1000	1	3	3	3.00	3.50	3.50	4	0	V	each

3.1 Multivariate Nodes for Multivariate Distributions

The basic usage of multivariate nodes (and the reason why they have been implemented) is for multivariate distributions such as the Dirichlet distribution, the multinomial distribution, the multivariate normal distribution and, possibly, the empirical distribution

As an example, assume that 3-member families buy 500 g of ground beef. The proportions of steak eaten by the baby, his older brother and his mother follow a Dirichlet (uncertainty) distribution with (vector) parameter $\alpha = (2, 3, 5)$.

We want to derive the distribution (variability) of steak eaten by 500 babies sampled from these 500 families.

```
> (p <- mcstoc(rdirichlet, type = "U", nsu = 100, nvariates = 3,
+   alpha = c(2, 3, 5)))
```

	node	mode	nsv	nsu	nva	variate	min	mean	median	max	Nas	type	outm
1	x	numeric	1	100	3	1	0.00904	0.191	0.165	0.748	0	U	each
2	x	numeric	1	100	3	2	0.05607	0.297	0.280	0.686	0	U	each
3	x	numeric	1	100	3	3	0.14634	0.512	0.533	0.777	0	U	each

```
> s <- mcstoc(rmultinomial, type = "VU", nsv = 500, nsu = 100,
+   nvariates = 3, size = 500, prob = p)
> summary(s)
```

node :

[[1]]

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
median	82.3	8.30	60	67.00	76.5	82.0	88.0	98.0	110.0	500	0
mean	95.4	8.04	72	80.24	89.9	95.3	100.7	111.4	120.4	500	0
2.5%	13.0	3.51	2	6.95	10.4	12.9	15.4	20.4	22.9	500	0
97.5%	254.6	11.04	216	233.45	246.9	254.5	262.5	275.8	286.7	500	0

[[2]]

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
median	140.1	9.89	107.5	121.0	133.0	140.0	146.5	161.8	172.5	500	0
mean	148.6	9.56	119.8	130.3	142.2	148.6	155.0	167.3	178.5	500	0
2.5%	46.8	6.29	27.5	35.2	42.4	46.4	50.9	59.4	68.5	500	0
97.5%	326.2	11.51	296.4	306.3	319.4	326.4	332.9	344.9	357.8	500	0

[[3]]

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
median	267	10.71	232.5	247	260	267	274	288	302	500	0
mean	256	10.62	222.9	235	249	256	263	276	288	500	0
2.5%	123	9.13	97.4	104	116	123	130	143	153	500	0
97.5%	377	11.61	347.6	359	370	376	383	395	402	500	0

Assume that each member of these families eats a normal distribution (variability) of steak with mean 100, 150 and 250 g. There is a positive correlation between the servings of the children, and a negative one with the serving of the mother. We want to derive the distribution (variability) of steak eaten by 500 babies.

```
> (x <- mcstoc(rmultinormal, type = "V", nvariates = 3, mean = c(100,
+   150, 250), sigma = c(10, 2, -5, 2, 10, -5, -5, -5, 10)))
```

	node	mode	nsv	nsu	nva	variate	min	mean	median	max	Nas	type	outm
1	x	numeric	1000	1	3	1	89.9	100	100	111	0	V	each

```

2    x numeric 1000    1    3        2 140.3  150    150 160    0    V each
3    x numeric 1000    1    3        3 241.1  250    250 261    0    V each

> cor(x[, 1, ])

      [,1]      [,2]      [,3]
[1,] 1.0000000  0.2324175 -0.5065176
[2,] 0.2324175  1.0000000 -0.5136596
[3,] -0.5065176 -0.5136596  1.0000000

```

In this example, `mean` could be variable or uncertain, as well as `sigma`⁸. You could have used, for an uncertain mean:

```

> m <- mcdata(c(100, 150, 250), type = "0", nvariates = 3)
> mun <- mcstoc(rnorm, type = "U", nvariates = 3, mean = m, sd = 20)
> x <- mcstoc(rmultinormal, type = "VU", nvariates = 3, mean = mun,
+   sigma = c(10, 2, -5, 2, 10, -5, -5, -5, 10))
> cor(x[, 1, ])

      [,1]      [,2]      [,3]
[1,] 1.0000000  0.1629061 -0.4729414
[2,] 0.1629061  1.0000000 -0.4966917
[3,] -0.4729414 -0.4966917  1.0000000

```

The correlation is preserved, but the mean of each category is uncertain.

Finally, multivariate nodes may be useful to derive a nonparametric bootstrap. Assume that, based on a study, you obtained 6 individuals who eat 100 g, 12 individuals who eat 150 g, 6 individuals who eat 170 g and 6 individuals who eat 200 g of ground beef. You want to use a nonparametric bootstrap to derive uncertainty [2], and then select samples from the empirical distribution.

```

> (x <- mcstoc(rempiricalD, type = "U", outm = c("min", "mean",
+   "max"), nvariates = 30, values = c(100, 150, 170, 200), prob = c(6,
+   12, 6, 6)))

  node    mode nsv nsu nva variate min mean median max Nas type outm
1    x numeric   1 100  30      NA 100  100    100 100   0    U  min
2    x numeric   1 100  30      NA 138  153    153 168   0    U  mean
3    x numeric   1 100  30      NA 200  200    200 200   0    U  max

> mcstoc(rempiricalD, type = "VU", values = x)

  node    mode nsv nsu nva variate min mean median max Nas type outm
1    x numeric 1000 100   1      1 100  153    150 200   0   VU  each

```

Printing the statistics of the 30 variates of `x` is of no interest. Instead, we use the `outm` option, which allows us to specify which output we want (`none` for none, `each`, the default, for a series of statistics for each variate, or, as in the example, a vector of functions that are applied over all the 30 variates).

⁸Caution: the use of a varying `sigma` can make the analysis very slow.

3.2 Multivariate Nodes as a Third Dimension for Multiple Options in a Model

The recycling rules in `mc2d` regarding the `nvariate` dimension are as follows: the recycling will be done from `nvariables=1` to `nvariables=n` with $n > 1$. This allows you to use multivariate nodes as a third dimension, in case you want to test various alternatives.

Assume, as in section 2.2.5, that the distribution representing uncertainty in `conc` was not certain, and that the microbiologists suggest that $conc \sim N(10, 2)$ is possible, but that $conc \sim U(8, 12)$ is also possible. We can *i*) build a bivariate node reflecting these two options; ii) transfer these options into the final risk estimate. We obtain a bivariate node for the risk, one using the first hypothesis, the second the second hypothesis.

```
> conc1 <- mcstoc(rnorm, type = "U", mean = 10, sd = 2)
> conc2 <- mcstoc(runif, type = "U", min = 8, max = 12)
> conc <- mcdata(c(conc1, conc2), type = "U", nvariables = 2)
> cook <- mcstoc(rempiricalD, type = "V", values = c(1, 1/5, 1/50),
+   prob = c(0.027, 0.373, 0.6))
> serving <- mcstoc(rgamma, type = "V", shape = 3.93, rate = 0.0806)
> expo <- conc * cook * serving
> dose <- mcstoc(rpois, type = "VU", nvariables = 2, lambda = expo)
> r <- mcstoc(runif, type = "U", min = 5e-04, max = 0.0015)
> risk <- 1 - (1 - r)^dose
> EC5 <- mc(conc, cook, serving, expo, dose, r, risk)
> summary(EC5)
```

```
conc :
[[1]]
      NoVar
median 9.43
mean   9.70
2.5%   5.40
97.5%  13.69
```

```
[[2]]
      NoVar
median 9.81
mean   9.96
2.5%   8.21
97.5%  11.86
```

```
cook :
      mean    sd  Min 2.5%  25%  50% 75% 97.5% Max  nsv Na's
NoUnc 0.108 0.164 0.02 0.02 0.02 0.02 0.2  0.2  1 1000  0
```

serving :

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
NoUnc	48.3	24.2	3.85	13.3	30.5	44.1	61.9	111	141	1000	0

expo :

[[1]]

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
median	48.0	81.1	0.995	2.97	7.60	13.22	66.9	231	697	1000	0
mean	49.4	83.4	1.024	3.05	7.82	13.59	68.8	237	717	1000	0
2.5%	27.5	46.4	0.570	1.70	4.35	7.57	38.3	132	399	1000	0
97.5%	69.6	117.7	1.444	4.30	11.03	19.18	97.0	335	1011	1000	0

[[2]]

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
median	49.9	84.3	1.035	3.08	7.91	13.7	69.5	240	725	1000	0
mean	50.7	85.6	1.051	3.13	8.02	14.0	70.6	244	736	1000	0
2.5%	41.8	70.6	0.866	2.58	6.62	11.5	58.2	201	606	1000	0
97.5%	60.3	102.0	1.251	3.73	9.56	16.6	84.1	290	876	1000	0

dose :

[[1]]

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
median	47.9	81.4	0.00	2.00	7.00	14.0	67.0	233	717	1000	0
mean	49.4	83.7	0.01	1.87	7.35	14.2	69.2	238	733	1000	0
2.5%	27.4	46.5	0.00	1.00	4.00	8.0	38.9	134	409	1000	0
97.5%	69.6	117.1	0.00	3.00	10.52	20.0	99.2	339	1030	1000	0

[[2]]

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
median	49.9	85.1	0	2.00	8.0	14.5	70.0	244	753	1000	0
mean	50.7	86.1	0	1.84	7.6	14.5	70.9	244	756	1000	0
2.5%	41.8	71.1	0	1.00	6.0	12.0	58.5	201	619	1000	0
97.5%	60.2	102.3	0	2.99	9.0	17.0	84.2	294	909	1000	0

r :

	NoVar
median	0.000996
mean	0.001001
2.5%	0.000523
97.5%	0.001495

risk :

[[1]]

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv
--	------	----	-----	------	-----	-----	-----	-------	-----	-----

```

median 0.0454 0.0682 0.00e+00 0.001756 0.00742 0.01423 0.0674 0.2073 0.521 1000
mean    0.0448 0.0665 8.27e-06 0.001861 0.00732 0.01405 0.0666 0.2088 0.504 1000
2.5%    0.0205 0.0329 0.00e+00 0.000604 0.00326 0.00649 0.0286 0.0993 0.270 1000
97.5%   0.0737 0.1029 0.00e+00 0.003901 0.01239 0.02412 0.1115 0.3402 0.710 1000
Na's
median   0
mean     0
2.5%     0
97.5%    0

```

```
[[2]]
```

```

      mean      sd Min      2.5%      25%      50%      75% 97.5%      Max      nsv Na's
median 0.0461 0.0689   0 0.00179 0.00739 0.01405 0.0673 0.211 0.524 1000    0
mean    0.0461 0.0683   0 0.00184 0.00758 0.01444 0.0684 0.215 0.518 1000    0
2.5%    0.0231 0.0368   0 0.00067 0.00366 0.00667 0.0330 0.108 0.308 1000    0
97.5%   0.0701 0.0993   0 0.00299 0.01183 0.02222 0.1082 0.323 0.699 1000    0

```

(Do not forget to transfer the number of variates you want in `mcstoc...` (see the definition of `dose`). `mc2d` cannot guess...)

3.3 Multivariate Nodes as a Third Dimension for Multiple Vectors/Contaminants

The recycling rules in `mc2d` also allow you to use multivariate nodes as a third dimension for multiple vectors/Contaminants.

Assume in our ground beef example that we have two contaminants: one has a mean concentration that follows an uncertainty distribution $conc \sim N(10, 2)$, the second one follows $conc \sim N(14, 2)$. We can *i*) build a bivariate node reflecting these two concentrations⁹ ; *ii*) transfer these options into the final dose; *iii*) sum the dose over the variates (using `mcapply`). The behavior of contaminants is transferred in the model.

```

> mconc <- mcdata(c(10, 14), type = "0", nvariates = 2)
> conc <- mcstoc(rnorm, nvariates = 2, type = "U", mean = mconc,
+              sd = 2)
> cook <- mcstoc(rempiricalD, type = "V", values = c(1, 1/5, 1/50),
+              prob = c(0.027, 0.373, 0.6))
> serving <- mcstoc(rgamma, type = "V", shape = 3.93, rate = 0.0806)
> expo <- conc * cook * serving
> dose <- mcstoc(rpois, type = "VU", nvariates = 2, lambda = expo)
> dosetot <- mcapply(dose, margin = "variates", fun = sum)
> r <- mcstoc(runif, type = "U", min = 5e-04, max = 0.0015)
> risk <- 1 - (1 - r)^dosetot

```

⁹Note that we could simulate a correlation between both contaminants using a multivariate normal distribution.

```

> EC6 <- mc(conc, cook, serving, expo, dose, dosetot, r, risk)
> summary(EC6)

conc :
[[1]]
      NoVar
median 10.10
mean    9.96
2.5%    5.17
97.5%   13.48

[[2]]
      NoVar
median 14.0
mean   14.1
2.5%   10.4
97.5%  17.7

cook :
      mean    sd  Min 2.5%  25%  50%  75% 97.5% Max  nsv Na's
NoUnc 0.117 0.184 0.02 0.02 0.02 0.02 0.2    1   1 1000    0

serving :
      mean    sd  Min 2.5%  25%  50%  75% 97.5% Max  nsv Na's
NoUnc 49.2 24.1 3.57 14.1 31.9 45.2 63.7   105 189 1000    0

expo :
[[1]]
      mean    sd  Min 2.5%  25%  50%  75% 97.5% Max  nsv Na's
median 59.4 104.7 0.720 3.07  7.96 14.13 78.8   404 963 1000    0
mean   58.6 103.3 0.710 3.03  7.85 13.94 77.7   399 950 1000    0
2.5%   30.4  53.7 0.369 1.57  4.08  7.24 40.4   207 493 1000    0
97.5%  79.4 139.9 0.962 4.10 10.63 18.87 105.2   540 1286 1000    0

[[2]]
      mean    sd  Min 2.5%  25%  50%  75% 97.5% Max  nsv Na's
median 82.3 145 0.997 4.25 11.02 19.6 109.1   559 1333 1000    0
mean   82.9 146 1.004 4.28 11.10 19.7 109.9   564 1343 1000    0
2.5%   61.0 108 0.739 3.15  8.17 14.5  80.9   415  989 1000    0
97.5% 104.0 183 1.261 5.37 13.94 24.7 138.0   708 1686 1000    0

dose :
[[1]]
      mean    sd  Min 2.5%  25%  50%  75% 97.5% Max  nsv Na's

```

median	59.4	105.4	0.00	2.00	8.00	15.00	79.0	409	955	1000	0
mean	58.6	103.5	0.02	1.96	7.56	14.64	78.7	396	953	1000	0
2.5%	30.4	53.8	0.00	1.00	3.48	7.47	40.3	197	503	1000	0
97.5%	79.4	140.7	0.00	3.00	10.52	19.52	107.2	533	1311	1000	0

[[2]]

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
median	82.5	145	0.00	3.00	11.0	20.0	111	559	1340	1000	0
mean	82.9	146	0.21	3.14	10.8	20.5	111	560	1346	1000	0
2.5%	61.3	109	0.00	2.00	8.0	15.5	82	420	996	1000	0
97.5%	103.8	183	1.00	4.00	14.0	25.5	140	708	1679	1000	0

dosetot :

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
median	142	250	1.00	6.00	19.0	35.0	190	963	2288	1000	0
mean	141	250	0.96	6.13	18.8	34.5	190	961	2299	1000	0
2.5%	110	193	0.00	4.00	14.5	26.5	146	744	1794	1000	0
97.5%	175	310	3.00	8.00	23.0	43.0	235	1205	2848	1000	0

r :

	NoVar
median	0.001030
mean	0.001038
2.5%	0.000553
97.5%	0.001448

risk :

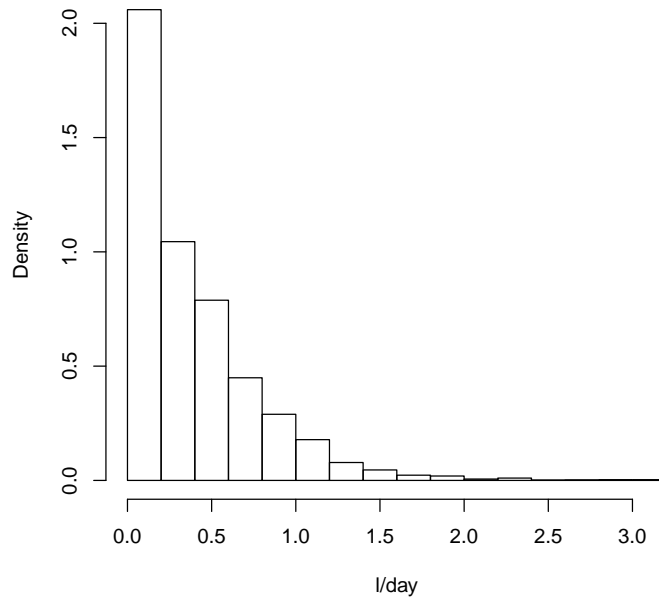
	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv
median	0.1174	0.1548	0.000903	0.00624	0.01952	0.0362	0.183	0.637	0.910	1000
mean	0.1136	0.1498	0.001028	0.00635	0.01930	0.0351	0.178	0.615	0.883	1000
2.5%	0.0641	0.0957	0.000000	0.00309	0.00989	0.0180	0.095	0.385	0.690	1000
97.5%	0.1603	0.1943	0.002938	0.01033	0.02925	0.0534	0.262	0.789	0.975	1000

Na's

median	0
mean	0
2.5%	0
97.5%	0

As a conclusion, this third dimension is highly flexible...

Figure 6: Histogram of daily tap water intake



4 Another Example: A QRA of Waterborne Cryptosporidiosis in France

This example is adapted from [4]. The aim is to evaluate the risk of infection with *Cryptosporidium parvum* from consumption of tap water, given that n oocysts /100 l. have been observed in a storage reservoir.

4.1 Tap Water Consumption Model

We have raw data of daily consumption of tap water from 1,180 tap water consumers (var `inca`, see Figure 6). We could choose to use this empirical distribution to evaluate the variability in the tap water consumption:

```
> ndvar(1001)
```

```
[1] 1001
```

```
> ndunc(1001)
```

```
[1] 1001
```



```
> mcstoc(rempiricalD, type = "V", values = inca)
```

```

      node    mode  nsv nsu nva variate min  mean median max Nas type outm
1      x numeric 1001   1   1         1  0 0.412   0.35 3.2   0   V each

```

but we will use the `fitdistrplus` library. `inca` includes a lot of 0 nodes, corresponding to days when individuals do not drink tap water (possibly they drink bottled water on those days). We could try a mixture of distributions, with 0 and non-0 data.

```

> library(fitdistrplus)
> pzero <- sum(inca == 0)/length(inca)
> inca_non_0 <- inca[inca != 0]
> descdist(inca_non_0)

```

```

summary statistics
-----
min:  0.02208   max:  3.2
median:  0.48
mean:  0.5658625
sample sd:  0.38546
sample skewness:  1.747461
sample kurtosis:  7.984648

```

Following the `descdist` function (See figure 7), let us try the lognormal distribution.

```

> Adj_water <- fitdist(inca_non_0, "lnorm", method = "mle")
> meanlog <- Adj_water$est[1]
> sdlog <- Adj_water$est[2]
> summary(Adj_water)

```

```
FITTING OF THE DISTRIBUTION ' lnorm ' BY MAXIMUM LIKELIHOOD
PARAMETERS
```

```

      estimate Std. Error
meanlog -0.7841836 0.008914641
sdlog    0.6740436 0.006303541
Loglikelihood: -1373.763   AIC:  2751.527   BIC:  2764.829
Correlation matrix:
      meanlog sdlog
meanlog      1      0
sdlog        0      1

```

```

-----
GOODNESS-OF-FIT STATISTICS

```

```
----- Chi-squared -----
```

Figure 7: Graph from the `descdist` function.

summary statistics

min: 0.02208 max: 3.2
 median: 0.48
 mean: 0.5658625
 sample sd: 0.38546
 sample skewness: 1.747461
 sample kurtosis: 7.984648

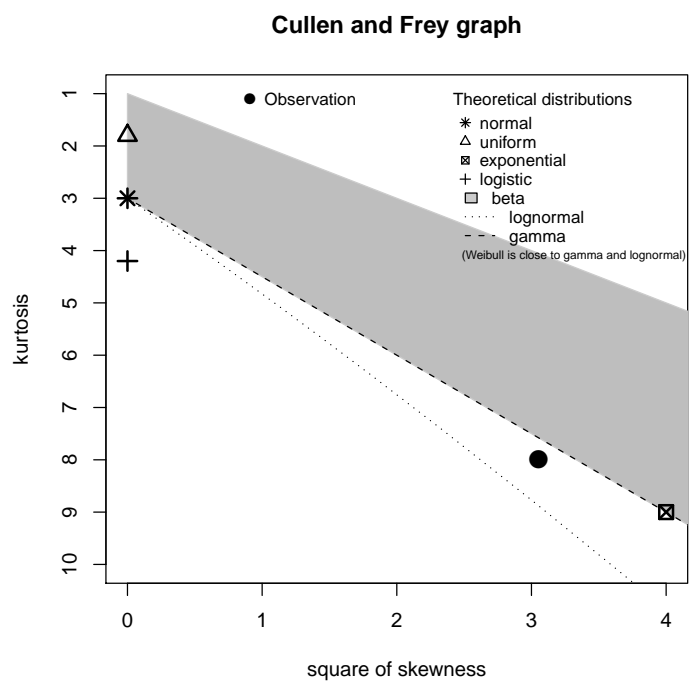
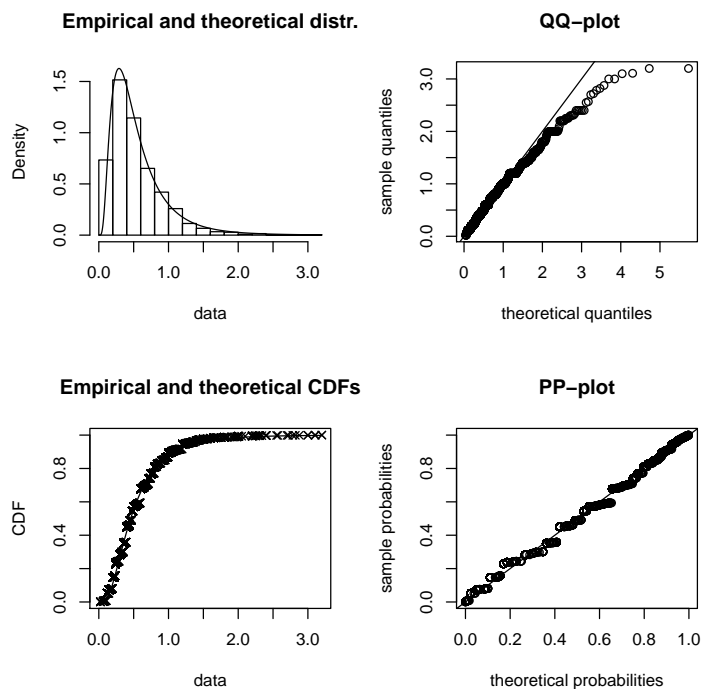


Figure 8: Graph from the `fitdist` function.



Chi-squared statistic: 3081.178
Degree of freedom of the Chi-squared distribution: 23
Chi-squared p-value: 0

!!! For continuous distributions, Kolmogorov-Smirnov and
Anderson-Darling statistics should be preferred !!!

----- Kolmogorov-Smirnov -----
Kolmogorov-Smirnov statistic: 0.06425329
Kolmogorov-Smirnov test: rejected
!!! The result of this test may be too conservative as it
assumes that the distribution parameters are known !!!

----- Anderson-Darling -----
Anderson-Darling statistic: 18.78726
Anderson-Darling test: rejected

```
> plot(Adj_water)
```

Not so bad (See Figure 8), and better than a gamma distribution (results not shown). We can now rebuild our mixture. We could consider uncertainty around the maximum likelihood estimates using the `bootdist` function of the `fitdistrplus` package, using something like:

```
> Boot <- bootdist(Ajust_lnorm, bootmethod = "param", niter = ndunc())
> Mean_conso <- mcddata(Boot$estim$meanlog, type = "U")
> Sd_conso <- mcddata(Boot$estim$sdlog, type = "U")
> conso1 <- mcstoc(rlnorm, type = "VU", meanlog = Mean_conso, sdlog = Sd_conso)
```

But for simplicity, we will not consider uncertainty around the estimates.

We will use the `mcprobtrees` function to construct a mixture of 0 and non-0 distributions:

```
> conso0 <- mcddata(0, type = "V")
> conso1 <- mcstoc(rlnorm, type = "V", meanlog = meanlog, sdlog = sdlog)
> v <- mcprobtrees(c(pzero, 1 - pzero), list(`0` = conso0, `1` = conso1),
+       type = "V")
> summary(v)
```

node :

	mean	sd	Min	2.5%	25%	50%	75%	97.5%	Max	nsv	Na's
NoUnc	0.403	0.474	0	0	0	0.306	0.588	1.69	4.28	1001	0

4.2 The Dose-Response Model

We propose a bootstrap from data (`datDR`) derived from [1]. We first define a function `DR` with an `n` argument for the size of the sample to draw. This function may then be used in a `mcstoc` function:

```
> datDR <- list(dose = c(30, 100, 300, 500, 1000, 10000, 1e+05,
+       1e+06), pi = c(2, 4, 2, 5, 2, 3, 1, 1), ni = c(5, 8, 3, 6,
+       2, 3, 1, 1))
> estDR <- function(pos, ref) {
+   -glm(cbind(ref$ni - pos, pos) ~ ref$dose + 0, binomial(link = "log"))$coefficients
+ }
> ml <- 1 - exp(-estDR(datDR$pi, datDR) * datDR$dose)
> DR <- function(n) {
+   boot <- matrix(rbinom(length(datDR$dose) * n, datDR$ni, ml),
+       nrow = length(datDR$dose))
+   apply(boot, 2, estDR, ref = datDR)
+ }
> r <- mcstoc(DR, type = "U")
> summary(r)
```

node :

NoVar

```

median 0.00547
mean   0.00578
2.5%   0.00286
97.5%  0.01094

```

4.3 The Model

Deriving the final model is straightforward. We construct the `mcnode` corresponding to the recovery rate (Uncertainty, `Rr`), the probability for an oocyst to be infective (Variability, `w`):

```

> Rr <- mcstoc(rbeta, type = "U", shape1 = 2.65, shape2 = 3.64)
> w <- mcstoc(rbeta, type = "V", shape1 = 2.6, shape2 = 3.4)

```

Given that $O_o = 2$ oocysts are observed in 100 l of water, the expected number of oocysts in the sample is 1:

```

> Oo <- 2
> l <- (Oo + mcstoc(rnbinom, type = "U", size = Oo + 1, prob = Rr))/100

```

The expected number of oocysts drunk by the individuals is `Or` and the risk ($\times 10000$) is estimated by:

```

> Or <- l * v * w
> P <- 10000 * (1 - exp(-r * Or))
> summary(P)

```

```

node :
      mean      sd Min 2.5% 25%   50%   75%  97.5%   Max  nsv Na's
median 0.544 0.709   0    0    0 0.362 0.755  2.485  6.60 1001    0
mean   0.810 1.056   0    0    0 0.538 1.124  3.700  9.81 1001    0
2.5%   0.146 0.190   0    0    0 0.097 0.203  0.667  1.77 1001    0
97.5%  3.239 4.220   0    0    0 2.153 4.493 14.790 39.20 1001    0

```

This result can be compared (roughly since there is some differences in the modelsvariability) to the results shown in Table 2 in [4].

Improvement: the results for $O_o = \{0, 1, 2, 5, 10, 20, 50, 100, 1000\}$ can be obtained in one step using:

```

> Oo <- mcdata(c(0, 1, 2, 5, 10, 20, 50, 100, 1000), type = "O",
+             nvariates = 9)

```

As a Conclusion

We think and hope that `mc2d` could help risk assessors to constuct and analyse their models, and that it may help in developing two-dimensional simulations. Nevertheless, `mc2d` is currently under development:

*CHECK YOUR MODEL CAREFULLY AND EXAMINE RESULTS TO
DETECT BUGS*

and, if you would like to improve it, join us at

<http://riskassessment.r-forge.r-project.org/>

Please refer any comments or bugs to rpouillot@yahoo.fr.

References

- [1] C. L. Chappell, P. C. Okhuysen, C. R. Sterling, and H. L. DuPont. Cryptosporidium parvum: intensity of infection and oocyst excretion patterns in healthy volunteers. *Journal of Infectious Diseases*, 173(1):232–6., 1996.
- [2] A.C. Cullen and H.C. Frey. *Probabilistic techniques in Exposure assessment*. Plenum Press, New York, 1999.
- [3] R. L. Iman and W. J. Conover. A distribution-free approach to inducing rank correlation among input variables. *Communication in Statistics*, B11(3):311–334, 1982.
- [4] R Pouillot, P Beaudeau, J.-B. Denis, F Derouin, and AFSSA Cryptosporidium Study Group. A quantitative risk assessment of waterborne cryptosporidiosis in france using second-order monte carlo simulation. *Risk Anal*, 24(1):1–17, 2004.
- [5] R Pouillot, N Miconnet, A.-L. Afchain, M.-L. Delignette-Muller, A Beaufort, L Rosso, J.-B. Denis, and M Cornu. Quantitative risk assessment of listeria monocytogenes in french cold-salmon : I. quantitative exposure assessment. *Risk Analysis*, 27(3):683–700, 2007.
- [6] D. Vose. *Risk Analysis, A quantitative guide, 2nd Edition*. Wiley and Sons, Chichester, 2nd edition, 2000.