

# Functional modules integrating essential cellular functions are predictive of the response of leukaemia cells to DNA damage

## R *Documentation*

---

Katrin Sameith<sup>1\*</sup>, Philipp Antczak<sup>1</sup>, Elliot Marston<sup>2</sup>, Nil Turan<sup>1</sup>,  
Dieter Maier<sup>3</sup>, Tanja Stankovic<sup>2</sup> and Francesco Falciani<sup>1†</sup>

<sup>1</sup>School of Biosciences and Institute of Biomedical Research (IBR),  
University of Birmingham, Birmingham, B152TT, UK

<sup>2</sup>CRUK Institute for Cancer Studies,  
University of Birmingham, Birmingham, B152TT, UK

<sup>3</sup>Biomax Informatics AG, Lochhamer Str. 9,  
82152 Martinsried, Germany

## Contents

1	<code>get.expr.dataset</code>	3
2	<code>read.adj.list</code>	4
3	<code>get.dataset.Symbol</code>	5
4	<code>get.network</code>	6
5	<code>network.subclusters</code>	7
6	<code>simplify.adj.list</code>	8
7	<code>reduce.dataset.acc.to.adj.list</code>	9
8	<code>get.ARACNE.MI.table</code>	11
9	<code>get.RANDOM.subset.MI</code>	12
10	<code>get.Random.Mean.Sigma</code>	13
11	<code>get.Random.Mean.Sigma.Smoothed</code>	14

---

\*Present address: Holstege Group, UMC Utrecht, Utrecht, 3508 AB, The Netherlands

†to whom correspondence should be addressed



---

## 1 `get.expr.dataset`

---

### Description

Reads in a numerical matrix of gene expression data.

### Usage

```
get.expr.dataset(f.dataset)
```

### Arguments

`f.dataset` character string naming a file which contains gene expression data ( $g \times m$ ) in a special format  $((g + 2) \times (m + 1))$ . The first line in `f.dataset` corresponds to  $m$  sample names of the dataset, the second line gives the class memberships of the  $m$  samples.  $g$  gene names are provided in column one.

### Values

`expr.matrix` numerical matrix of gene expression data ( $g \times m$ ) with  $g$  genes in rows and  $m$  samples in columns. Rownames correspond to the appropriate gene symbols. Column names correspond to the appropriate sample names.

`expr.classes` vector of class memberships of the  $m$  samples.

### Authors

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

### Example

```
source("FuMoSearch/RCode/FuMoSearch.R")
d <- get.expr.datasets("FuMoSearch/Documentation/ALL.symbol.txt")
dataset.Symbol <- d$expr.matrix
dataset.Symbol.classes <- d$expr.classes
```

---

## 2 read.adj.list

---

### Description

Reads in an adjacent list.

### Usage

```
read.adj.list(f.adj.list)
```

### Arguments

<code>f.adj.list</code>	character string naming a file which contains the adjacent list of interest. In each line, the first column depicts the name of a gene which is adjacent to genes whose corresponding row number is listed within this line.
-------------------------	--

### Values

<code>adj.list</code>	corresponding adjacent list.
-----------------------	------------------------------

### Authors

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

### Example

```
source("FuMoSearch/RCode/FuMoSearch.R")
a <- read.adj.list("FuMoSearch/Documentation/ALL.symbol.adj.list.txt")
adj.list <- a$adj.list
```

---

## 3 `get.dataset.Symbol`

---

### Description

`get.dataset.Symbol` transforms a dataset containing expression values for Affymetrix probesets into a dataset containing expression values for genes. Several Affy-IDs may correspond to the same gene symbol and therefore, their expression profiles are averaged in each sample.

### Usage

```
get.dataset.Symbol(dataset, dataset.classes, array, f.out)
```

### Arguments

<code>dataset</code>	numerical matrix of gene expression data ( $g_a \times m$ ) with $g_a$ Affymetrix probesets in rows and $m$ samples in columns. Rownames correspond to the appropriate Affy-IDs.
<code>dataset.classes</code>	vector of class memberships of the $m$ samples.
<code>array</code>	characters "hgu133a" or "hgu95av2" dependent on the source of the gene expression data.
<code>f.out</code>	character string naming the output file. The transformed dataset will be written into "f.out".

### Values

<code>dataset.Symbol</code>	numerical matrix of gene expression data ( $g_s \times m$ ) with $g_s$ genes in rows and $m$ samples in columns. Rownames correspond to the appropriate gene symbols. Also written into <code>f.out</code> .
<code>dataset.Symbol.classes</code>	vector of class memberships of the $m$ samples. Also written into <code>f.out</code> .

### Author(s)

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

### Example

```
source("FuMoSearch/RCode/FuMoSearch.R")
library(galgo)
data(ALL)
data(ALL.classes)
ALL.classes <- as.vector(ALL.classes)

d <- get.dataset.Symbol(ALL, ALL.classes, "hgu95av2", "ALL.symbol.txt")
dataset.Symbol <- d$dataset.Symbol
dataset.Symbol.classes <- d$dataset.Symbol.classes
```

---

## 4 `get.network`

---

### Description

`get.network` extracts molecular interactions from KEGG, DIP, BIND, BioGRID and HPRD to a given dataset and stores appropriate connections in an adjacent list.

### Usage

```
get.network(dataset.Symbol, f.out.adj.list)
```

### Arguments

<code>dataset.Symbol</code>	numerical matrix of gene expression data ( $g_s \times m$ ) with $g_s$ genes in rows and $m$ samples in columns. Rownames correspond to the appropriate gene symbols.
<code>f.out.adj.list</code>	character string naming the output file. The extracted adjacent list will be written into " <code>f.out.adj.list</code> ".

### Details

The output file contains an adjacent list. Hereby, the first column in each line corresponds to one gene which is connected to all other genes whose row number is listed in this line.

### Values

<code>adj.list</code>	extracted adjacent list of length $g_s$ . Also written into <code>f.out.adj.list</code> .
-----------------------	---

### Author(s)

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

### Example

```
source("FuMoSearch/RCode/FuMoSearch.R")
d <- get.expr.datasets("FuMoSearch/Documentation/ALL.symbol.txt")
dataset.Symbol <- d$expr.matrix

a <- get.network(dataset.Symbol, "ALL.symbol.adj.list.txt")
adj.list <- a$adj.list
```

---

## 5 network.subclusters

---

### Description

`network.subclusters` studies a network and determines subnetworks which are not connected to each other.

### Usage

```
network.subclusters(adj.list)
```

### Arguments

`adj.list` adjacent list of interest.

### Details

Looking at `G.sep`, one can manually decide whether subnetworks containing less than a certain number of genes should be removed from the adjacent list.

### Values

`G.sep` List of gene indices describing subnetworks which are not connected to each other.

### Author(s)

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

### Example

```
source("FuMoSearch/RCode/FuMoSearch.R")
a <- read.adj.list(
  "FuMoSearch/Documentation/ALL.symbol.adj.list.txt")
adj.list <- a$adj.list

G.sep <- network.subclusters(adj.list)$G.sep
G.sep.length <- sapply(G.sep, function(x) length(x))
print(G.sep.length)
ind.delete <- unlist(G.sep[-1])

# Only keep the biggest network containing 1106 nodes
adj.list.BIG.nw <- adj.list
for(i in 1:length(ind.delete)) adj.list.BIG.nw[[ind.delete[i]]] <- integer(0)
write.adj.list(adj.list.BIG.nw, "ALL.symbol.adj.list.big.nw.txt")
```

---

## 6 `simplify.adj.list`

---

### Description

`simplify.adj.list` removes genes with no connections from the adjacent list and renews indices of the remaining genes.

### Usage

```
simplify.adj.list(adj.list, f.out.adj.list.reduced,  
                 f.out.index.genes.to.keep)
```

### Arguments

<code>adj.list</code>	adjacent list of interest.
<code>f.out.adj.list.reduced</code>	character string naming an output file. The simplified adjacent list will be written into " <code>f.out.adj.list.reduced</code> ".
<code>f.out.index.genes.to.keep</code>	character string naming an output file. Original indices of genes kept in the simplified adjacent list will be written into " <code>f.out.index.genes.to.keep</code> ".

### Values

<code>adj.list.simpl</code>	simplified adjacent list of length $g_{simpl}$ . Also written into <code>f.out.adj.list.reduced</code> .
-----------------------------	--

### Authors

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

### Example

```
source("FuMoSearch/RCode/FuMoSearch.R")  
a <- read.adj.list(  
  "FuMoSearch/Documentation/ALL.symbol.adj.list.big.nw.txt")  
adj.list <- a$adj.list  
  
a.simpl <- simplify.adj.list(adj.list, "ALL.symbol.adj.list.simpl.txt",  
                             "ALL.symbol.adj.list.simpl.i.g.t.k.txt")  
adj.list.simpl <- a.simpl$adj.list.simpl
```

---

## 7 `reduce.dataset.acc.to.adj.list`

---

### Description

According to the simplified adjacent list, `reduce.dataset.acc.to.adj.list` removes genes with no connections from the dataset.

### Usage

```
reduce.dataset.acc.to.adj.list(dataset.Symbol, dataset.Symbol.classes  
                               f.index.genes.to.keep,  
                               f.out.dataset.Symbol.reduced)
```

### Arguments

<code>dataset.Symbol</code>	numerical matrix of gene expression data ( $g_s \times m$ ) with $g_s$ genes in rows and $m$ samples in columns. Rownames correspond to the appropriate gene symbols.
<code>dataset.Symbol.classes</code> <code>f.index.genes.to.keep</code>	vector of class memberships of the $m$ samples. character string naming a file which contains the original indices of genes kept in the simplified adjacent list.
<code>f.out.dataset.Symbol.reduced</code>	character string naming an output file. The simplified dataset will be written into " <code>f.out.dataset.Symbol.reduced</code> ".

### Values

<code>dataset.Symbol.simpl</code>	numerical matrix of gene expression data ( $g_{s_{simpl}} \times m$ ) with $g_{s_{simpl}}$ genes in rows and $m$ samples in columns. Rownames correspond to the appropriate gene symbols. Also written into <code>f.out.dataset.Symbol.reduced</code> .
<code>dataset.Symbol.simpl.classes</code>	vector of class memberships of the $m$ samples. Also written into <code>f.out.dataset.Symbol.reduced</code> .

### Authors

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

### Example

```
source("FuMoSearch/RCode/FuMoSearch.R")  
d <- get.expr.datasets("FuMoSearch/Documentation/ALL.symbol.txt")  
dataset.Symbol <- d$expr.matrix  
dataset.Symbol.classes <- d$expr.classes  
  
f.in <- "FuMoSearch/Documentation/ALL.symbol.adj.list.simpl.i.g.t.k.txt"  
dataset.Symbol.simpl <- reduce.dataset.acc.to.adj.list(dataset.Symbol,
```

```
dataset.Symbol.classes, f.in,  
"ALL.symbol.simpl.txt")
```

---

## 8 `get.ARACNE.MI.table`

---

### Description

`get.ARACNE.MI.table` transforms a dataset into an ARACNe compatible format, calls ARACNe and reformat ARACNe's output into a table of pairwise mutual information values.

### Usage

```
get.ARACNE.MI.table(dataset, f.ARACNE.input, f.ARACNE.output,  
                    ARACNE.home.dir, f.MI.table)
```

### Arguments

<code>dataset</code>	numerical matrix of gene expression data ( $g \times m$ ) with $g$ genes in rows and $m$ samples in columns. Rownames correspond to the appropriate gene symbols.
<code>f.ARACNE.input</code>	character string naming a file into which the ARACNe compatible format of <code>dataset</code> will be written.
<code>f.ARACNE.output</code>	character string naming a file into which the ARACNe output will be written.
<code>ARACNE.home.dir</code>	character string depicting the directory of the executable <code>aracne</code> program.
<code>f.MI.table</code>	character string naming an output file. The calculated table of mutual information values will be written into <code>f.MI.table</code>

### Details

This function is meant to be run on LINUX. ARACNe can be downloaded from [http://amdec-bioinfo.cu-genome.org/html/caWorkBench/upload/aracne\\_source.zip](http://amdec-bioinfo.cu-genome.org/html/caWorkBench/upload/aracne_source.zip) and must be copied into your home directory. Appropriate permissions of the executable "aracne" program might need to be reset.

### Values

`MI table` symmetric table ( $g \times g$ ) of mutual information values.

### Author(s)

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

### Example

```
source("FuMoSearch/RCode/FuMoSearch.R")  
d <- get.expr.dataset("FuMoSearch/Documentation/ALL.Symbol.simpl.txt")  
dataset.Symbol.simpl <- d$expr.matrix  
# (ARACNe folder "aracne_source" is copied into your home directory)  
  
MI.table <- get.ARACNE.MI.table(dataset.Symbol.simpl,  
                               "ALL.Symbol.simpl.ARACNE.exp", "ALL.Symbol.simpl.ARACNE.adj",  
                               "aracne_source", "ALL.Symbol.simpl.MI.table.txt")$ARACNE.MI.table
```

---

## 9 `get.RANDOM.subset.MI`

---

### Description

`get.RANDOM.subset.MI` iteratively generates random subsets of genes and calculates their basic scores corresponding to the average mutual information between the genes in a subset.

### Usage

```
get.RANDOM.subset.MI(MI.table, RMAX, KMAX = nrow(MI.table), f.out)
```

### Arguments

<code>MI.table</code>	numerical matrix ( $g \times g$ ) of pairwise mutual information.
<code>RMAX</code>	maximal number of iterations, i.e. for each subset size $k$ , corresponding <code>RMAX</code> subsets are randomly generated.
<code>KMAX</code>	maximal number of subset size $k$ to be considered. By default, <code>KMAX</code> = $g$ . However, to accelerate the process, a lower <code>KMAX</code> can be chosen, e.g. <code>KMAX</code> = 500 as one would not expect the subsequent functional module search procedure to consider modules of size 501 or more (and hence, $\mu_{501}$ and $\sigma_{501}$ would not be needed!).

### Values

<code>score.RandSub</code>	numerical matrix ( <code>KMAX</code> $\times$ <code>RMAX</code> ) of scores calculated from randomly generated subsets.
----------------------------	---

### Author(s)

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

### Example

```
source("FuMoSearch/RCode/FuMoSearch.R")
MI.table <- read.table(
  "FuMoSearch/Documentation/ALL.symbol.simpl.MI.table.txt",
  row.names = 1, header = TRUE)

r <- get.RANDOM.subset.MI(MI.table, RMAX = 1000, KMAX = 500, f.out =
  "ALL.symbol.simpl.RANDOM.subsets.txt")
score.randomSubsets <- r$score.RandSub
```

---

## 10 `get.Random.Mean.Sigma`

---

### Description

`get.Random.Mean.Sigma` calculates the score mean and standard deviation of *a priori* generated and scored subsets of size  $k$ ,  $1 \leq k \leq \text{KMAX}$ .

### Usage

```
get.Random.Mean.Sigma(RANDOM.subsets, f.out.RANDOM.mean = "RANDOM.mean.txt",  
                      f.out.RANDOM.sigma = "RANDOM.sigma.txt")
```

### Arguments

<code>RANDOM.subsets</code>	numerical matrix ( $\text{KMAX} \times \text{RMAX}$ ) of scores calculated from randomly generated subsets. For every $k$ , $1 \leq k \leq \text{KMAX}$ , $\text{RMAX}$ random gene subsets of size $k$ are generated and subsequently assessed.
<code>f.out.RANDOM.mean</code>	character string naming an output file. Mean scores of random subsets are written into <code>f.out.RANDOM.mean</code> .
<code>f.out.RANDOM.sigma</code>	character string naming an output file. Score standard deviations of random subsets are written into <code>f.out.RANDOM.sigma</code> .

### Values

<code>RANDOM.mean</code>	vector of length $\text{KMAX}$ depicting mean scores of random subsets of size $k$ , $1 \leq k \leq \text{KMAX}$ . Also written into <code>f.out.RANDOM.mean</code> .
<code>RANDOM.sigma</code>	vector of length $\text{KMAX}$ depicting score standard deviations of random subsets of size $k$ , $1 \leq k \leq \text{KMAX}$ . Also written into <code>f.out.RANDOM.sigma</code> .

### Author(s)

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

### Example

```
source("FuMoSearch/RCode/FuMoSearch.R")  
RANDOM.subsets <- as.matrix(read.table(  
"FuMoSearch/Documentation/ALL.symbol.simpl.RANDOM.subsets.txt", header = FALSE))  
  
r <- get.Random.Mean.Sigma(RANDOM.subsets,  
"ALL.symbol.simpl.RANDOM.mean.txt", "ALL.symbol.simpl.RANDOM.sigma.txt")  
  
# Mean scores of random subsets for each k, 1 <= k <= KMAX  
RANDOM.mean <- r$RANDOM.mean  
  
# Score standard deviations of random subsets for each k, 1 <= k <= KMAX  
RANDOM.sigma <- r$RANDOM.sigma
```

---

## 11 `get.Random.Mean.Sigma.Smoothed`

---

### Description

`get.Random.Mean.Sigma.Smoothed` smoothes distributions of `RANDOM.mean` and `RANDOM.sigma`.

### Usage

```
get.Random.Mean.Sigma.Smoothed(RANDOM.mean, RANDOM.sigma,  
                                f.out.RANDOM.sm.mean = "RANDOM.sm.mean.txt",  
                                f.out.RANDOM.sm.sigma = "RANDOM.sm.sigma.txt",  
                                thresh = 100)
```

### Arguments

<code>RANDOM.mean</code>	vector of length <code>KMAX</code> depicting mean scores of random subsets of size $k$ , $1 \leq k \leq KMAX$ .
<code>RANDOM.sigma</code>	vector of length <code>KMAX</code> depicting score standard deviations of random subsets of size $k$ , $1 \leq k \leq KMAX$ .
<code>f.out.RANDOM.sm.mean</code>	character string naming an output file. Smoothed values of <code>RANDOM.mean</code> are written into <code>f.out.RANDOM.sm.mean</code> .
<code>f.out.RANDOM.sm.sigma</code>	character string naming an output file. Smoothed values of <code>RANDOM.sigma</code> are written into <code>f.out.RANDOM.sm.sigma</code> .
<code>thresh</code>	numerical value defining the maximal number of random gene subset size $k$ to be considered during the smoothing process.

### Details

`RANDOM.mean` and `RANDOM.sigma` are smoothed using the `pspline` R-package. To this end, `pspline` must be installed. Within the functional module search procedure a particular subnetwork  $A$  is assessed by a  $z$ -score

$$zscore_A = \frac{score_A - RANDOM.mean[k]}{RANDOM.sigma[k]}$$

where  $k$  is the size of  $A$ , i.e. the number of nodes in the subnetwork  $A$ . We expect the  $z$ -score to be a monotone increasing function of  $k$ . However, several peaks can be observed. Therefore, the function `get.Random.Mean.Sigma.Smoothed` can be applied to remove peaks up to a  $k = \text{thresh}$ .

### Values

<code>RANDOM.sm.mean</code>	vector of length <code>KMAX</code> depicting smoothed mean scores of random subsets of size $k$ , $1 \leq k \leq KMAX$ .
<code>RANDOM.sm.sigma</code>	vector of length <code>KMAX</code> depicting smoothed score standard deviations of random subsets of size $k$ , $1 \leq k \leq KMAX$ .

### Author(s)

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

## Example

```
source("FuMoSearch/RCode/FuMoSearch.R")
RANDOM.mean <- read.table(
  "FuMoSearch/Documentation/ALL.symbol.simpl.RANDOM.mean.txt")[,1]
RANDOM.sigma <- read.table(
  "FuMoSearch/Documentation/ALL.symbol.simpl.RANDOM.sigma.txt")[,1]

r.sm50 <- get.Random.Mean.Sigma.Smoothed(RANDOM.mean, RANDOM.sigma,
  "ALL.symbol.simpl.RANDOM.mean.sm50.txt",
  "ALL.symbol.simpl.RANDOM.sigma.sm50.txt", 50)

# Smoothed mean scores of random subsets for each k, 1 <= k <= KMAX
RANDOM.sm.mean <- r.sm50$RANDOM.sm.mean

# Smoothed score standard deviations of random subsets for each k,
# 1 <= k <= KMAX
RANDOM.sm.sigma <- r.sm50$RANDOM.sm.sigma
```

---

## 12 get.FUMO

---

### Description

get.FUMO searches functional modules.

### Usage

```
get.FUMO <- function( f.MI.table = NULL, f.adj.list = NULL,
  f.RANDOM.sm.mean = NULL, f.RANDOM.sm.sigma = NULL, M = 1, G =
  NULL, Temp.function = NULL, term.cond.max.silence = NULL,
  term.cond.max.steps = 100000, temp.shift = FALSE,
  set.G.random.after.temp.shift = FALSE,
  set.G.tolastnode.after.temp.shift = FALSE, plotting = FALSE,
  history.window = 100, speed = 10, f.OUT.history.modules =
  "History.modules.txt", f.OUT.temp.shift = "Temp.shift.txt",
  f.OUT.image = ".RDATA")
```

### Arguments

<code>f.MI.table</code>	character string naming a file which contains the table of pairwise mutual information values.
<code>f.adj.list</code>	character string naming a file which contains the adjacent list.
<code>f.RANDOM.sm.mean</code>	character string naming a file which contains score means of random gene subsets.
<code>f.RANDOM.sm.sigma</code>	character string naming a file which contains score standard deviations of random gene subsets.
<code>M</code>	number of <i>seed genes</i> .
<code>G</code>	optional: list of starting modules.
<code>Temp.function</code>	a function which returns a "Temperature" value for a given cycle number. $T.0.1 = e^{-0.1 \cdot cycle.nr}$ and $T.0.01 = e^{-0.01 \cdot cycle.nr}$ are predefined. However, any function implemented and used in the search.
<code>term.cond.max.silence</code>	maximal number of cycles without improvement. If <code>term.cond.max.silence</code> is reached, a new simulated annealing procedure is started.
<code>term.cond.max.steps</code>	maximal number of cycles.

<code>set.G.random.after.temp.shift</code>	logical. If true, $M$ <i>seed genes</i> are randomly selected at the beginning of each simulated annealing procedure.
<code>set.G.tolastnode.after.temp.shift</code>	logical. If true, a new simulated annealing procedure is started from the very last node changed during the prior procedure. If both <code>set.G.random.after.temp.shift</code> and <code>set.G.tolastnode.after.temp.shift</code> are false, a new simulated annealing procedure is started from the modules selected at the termination of the prior procedure.
<code>plotting</code>	logical. If true, a real-time plot visualises the progress of the search procedure.
<code>history.window</code>	number of search cycles in the working memory. For example, if <code>history.window = 10</code> , results of <code>cycle = 1</code> are written into a file and hence, not plotted anymore at <code>cycle = 11</code> .
<code>speed</code>	number depicting the span of cycles to consider in the "speed" plot. For example, if <code>speed = 3</code> , the "speed" plot depicts the absolute overlap of modules from cycle $t$ and cycle $t - 3$ .
<code>f.OUT.history.modules</code>	character string naming an output file. Modules from the complete search are written into <code>f.OUT.history.modules</code> .
<code>f.OUT.temp.shift</code>	character string naming an output file. Each cycle number at which a temperature shift occurred is written into <code>f.OUT.temp.shift</code> .
<code>f.OUT.image</code>	character string naming an R object output file. Selected and quenced modules as well as their sizes, scores and number of occurrences are saved in <code>f.OUT.image</code> .

## Details

`get.FUMO` performs repeated simulated annealing procedures, selects modules at each termination and locally optimises them. Important results are saved in `f.OUT.image`. Hereby, `qu.best.fumo.unique` represents the results of the selected modules which have been locally optimised.

## Values

The following results are saved in `f.OUT.image`.

<code>f.OUT.history.modules</code>	character string naming an output file. Modules from the complete search are written into <code>f.OUT.history.modules</code> .
------------------------------------	--

<code>f.OUT.temp.shift</code>	character string naming an output file. Each cycle number at which a temperature shift occurred is written into <code>f.OUT.temp.shift</code> .
<code>MI.table</code>	table of pairwise mutual information values used in the search procedure.
<code>RANDOM.sm.mean</code>	score means of random gene subsets used in the search procedure.
<code>RANDOM.sm.sigma</code>	score standard deviations of random gene subsets used in the search procedure.
<code>adj.list</code>	adjacency list used in the search procedure.
<code>temp.shift</code>	time points of temperature shifts occurred in the search procedure.
<code>best.fumo</code>	list of <code>best.modules</code> (list of modules selected at each temperature shift), <code>best.modules.fitness</code> (numerical table of scores of modules selected at each temperature shift), <code>best.modules.size</code> (numerical table of sizes of modules selected at each temperature shift), <code>best.modules.nr.modules</code> (number of modules count at each temperature shift), <code>best.modules.list.equal</code> (list of equal modules described as indices within <code>best.modules</code> ).
<code>best.fumo.unique</code>	list of <code>best.modules.unique</code> (list of unique selected modules containing at least three genes), <code>best.modules.unique.fitness</code> (numerical table of scores of unique modules, associated to the first time point of their selection), <code>best.modules.unique.size</code> (numerical table of sizes of unique modules, associated to the first time point of their selection), <code>best.modules.unique.nr.modules</code> (number of modules count at each temperature shift), <code>best.modules.unique.count</code> (number of occurrences of each selected module).
<code>qu.best.fumo.unique</code>	list of <code>best.modules.unique</code> (list of unique quenched modules), <code>best.modules.unique.fitness</code> (numerical table of scores of unique quenched modules, associated to the first time point of their selection), <code>best.modules.unique.size</code> (numerical table of sizes of unique quenched modules, associated to the first time point of their selection), <code>best.modules.unique.count</code> (number of occurrences of each unique quenched module).

## Author(s)

Katrin Sameith. Dr. Francesco Falciani Group. University of Birmingham.

## Example

```
source("FuMoSearch/RCode/FuMoSearch.R")

# Functional module search procedure
get.FUMO(
  f.MI.table =
    "FuMoSearch/Documentation/ALL.symbol.simpl.MI.table.txt",
  f.adj.list =
    "FuMoSearch/Documentation/ALL.symbol.adj.list.simpl.txt",
  f.RANDOM.sm.mean =
    "FuMoSearch/Documentation/ALL.symbol.simpl.RANDOM.mean.sm50.txt",
  f.RANDOM.sm.sigma =
    "FuMoSearch/Documentation/ALL.symbol.simpl.RANDOM.sigma.sm50.txt",
  M = 1,
```

```
Temp.function = Temp.0.1,  
term.cond.max.silence = 100,  
term.cond.max.steps = 10000,  
temp.shift = TRUE,  
set.G.random.after.temp.shift = TRUE,  
f.OUT.history.modules = "ALL.symbol.simpl.History.modules.txt",  
f.OUT.temp.shift = "ALL.symbol.simpl.Temp.shift.txt",  
f.OUT.image = "ALL.symbol.simpl.FuMoSearch.RDATA",  
plotting = TRUE)  
  
# Load the results  
load("ALL.symbol.simpl.FuMoSearch.RDATA")
```