# Package 'spm'

November 13, 2019

**Title** SPM extract package

**Version** 1.1

**Date** 2019-11-13

**Author** A. Dunn

**Maintainer** A Dunn <Alistair.Dunn@OceanEnvironmental.co.nz>

**Description** A set of R functions for extracting from SPM output files.

**License** CPL v1.0. See the SPM User Manual for license details.

**URL** http://www.niwa.co.nz



## R topics documented:

---

`spm-package` *SPM extract package*

---

## Description

A set of R functions for extracting and processing SPM output.

## Details

| | |
|---|---|
| Package: | spm |
| Date: | 2013-02-12 |
| License: | See the SPM manual, or use 'spm -l' at the command line to view the SPM license. |
| URL: | http://www.niwa.co.nz |
| Copyright: | National Institute of Water & Atmospheric Research (NIWA). |

Index:

| | |
|---|---|
| spm-package | SPM package |

## Author(s)

A. Dunn

Maintainer: A Dunn <a.dunn@niwa.co.nz>

---

`calc.abundance` *Generate a multivariate uniform distribution based on the bounds for the extimated parameters*

---

## Description

Generate a multivariate uniform distribution based on the bounds for the extimated parameters

## Usage

```
calc.abundance(data, categories = NULL, ages = NULL, total = T)
```

## Arguments

| | |
|---|---|
| data | data representing a partition |
| categories | Optional subset of categories |
| ages | Optional subset of ages |
| total | If true, then sum over selected categories and ages. Default = TRUE |

## Author(s)

Alistair Dunn

---

`extract` *Extract SPM output into R*

---

## Description

Extract SPM output into R

## Usage

```
extract(file, path = "", ignore.unknown = FALSE)
```

## Arguments

| | |
|---|---|
| file | the name of the input file containing model output to extract |
| path | Optionally, the path to the file |
| ignore.unknown | Ignore unknown objects in the output file |

## Author(s)

Alistair Dunn

## extract.ageingerror *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.ageingerror(lines)
```

### Author(s)

Alistair Dunn

## extract.agesize *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.agesize(lines)
```

### Author(s)

Alistair Dunn

## extract.ageweight *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.ageweight(lines)
```

### Author(s)

Alistair Dunn

## extract.covariance *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.covariance(lines)
```

### Author(s)

Alistair Dunn

## extract.derivedquantity *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.derivedquantity(lines)
```

### Author(s)

Alistair Dunn

## extract.derivedquantitybycell *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.derivedquantitybycell(lines)
```

### Author(s)

Alistair Dunn

## extract.estimatesummary *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.estimatesummary(lines)
```

### Author(s)

Alistair Dunn

## extract.estimatevalue *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.estimatevalue(lines)
```

### Author(s)

Alistair Dunn

## extract.initialisationphase *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.initialisationphase(lines)
```

### Arguments

| | |
|---|---|
| lines | a file of scanned lines |

### Author(s)

Alistair Dunn

## extract.layer *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.layer(lines)
```

### Author(s)

Alistair Dunn

## extract.layerderivedview *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.layerderivedview(lines)
```

### Author(s)

Alistair Dunn

## extract.MCMC *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.MCMC(lines)
```

### Author(s)

Alistair Dunn

## extract.objectivefunction *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.objectivefunction(lines)
```

### Author(s)

Alistair Dunn

## extract.observation *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.observation(lines)
```

### Author(s)

Alistair Dunn

## extract.partition *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.partition(lines)
```

### Author(s)

Alistair Dunn

## extract.partitionbiomass *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.partitionbiomass(lines)
```

### Author(s)

Alistair Dunn

## extract.randomnumberseed *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.randomnumberseed(lines)
```

### Author(s)

Alistair Dunn

## extract.selectivity *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.selectivity(lines)
```

### Author(s)

Alistair Dunn

## extract.simulations *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.simulations(file, path = "")
```

### Author(s)

Alistair Dunn

## extract.sizeweight *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.sizeweight(lines)
```

### Author(s)

Alistair Dunn

## extract.spatialmap *Utility extract function*

### Description

Utility extract function

### Usage

```
extract.spatialmap(lines)
```

### Author(s)

Alistair Dunn

## generate.MVU — *Generate a multivariate uniform distribution based on the bounds for the extimated parameters*

### Description

Generate a multivariate uniform distribution based on the bounds for the extimated parameters

### Usage

```
generate.MVU(file, path = "", output.file, sample.size = 1)
```

### Arguments

| | |
|---|---|
| file | the name of the input file containing the estimated fits |
| path | Optionally, the path to the file |
| output.file | The name of the output file to write randomly generated values |
| sample.size | The number f samples to generate |

### Author(s)

Sophie Mormede

## PF — *Evaluate a preference function*

### Description

Evaluate a preference function

### Usage

```
PF(type = "none", x, alpha, ..., rescale = T)
```

### Arguments

| | |
|---|---|
| type | A type of preference function. Valid values include constant", "doublenormal", "exponential", "inverselogistic", "logistic", "normal", and "threshold" |
| x | x-values over which to evaluate. Default = 1 |
| alpha | The value of the alpha parameter. Default = 1 |
| ... | the parameters of the preference functions |
| rescale | Rescale the function to have value 1. Default = TRUE |

### Author(s)

Alistair Dunn

## PFconstant *Evaluate the constant preference function*

### Description

Evaluate the constant preference function

### Usage

```
PFconstant(x, alpha, rescale = T)
```

### Arguments

| | |
|---|---|
| x | x-values over which to evaluate. Default = 1 |
| alpha | The value of the alpha parameter. Default = 1 |
| rescale | Rescale the function to have value 1?. Default = TRUE |

### Author(s)

Alistair Dunn

## PFdoublenormal *Evaluate the double normal preference function*

### Description

Evaluate the double normal preference function

### Usage

```
PFdoublenormal(x, alpha, mu, sigmaL, sigmaR, rescale = T)
```

### Arguments

| | |
|---|---|
| x | x-values over which to evaluate. Default = 1 |
| alpha | The value of the alpha parameter. Default = 1 |
| mu | the mean of the double normal |
| sigmaL | the standard deviation of the left-hand side of the double normal |
| sigmaR | the standard deviation of the right-hand side of the double normal |
| rescale | Rescale the function to have value 1?. Default = TRUE |

### Author(s)

Alistair Dunn

## PFexponential *Evaluate the exponential preference function*

### Description

Evaluate the exponential preference function

### Usage

```
PFexponential(x, alpha, lambda, rescale = T)
```

### Arguments

| | |
|---|---|
| x | x-values over which to evaluate. Default = 1 |
| alpha | The value of the alpha parameter. Default = 1 |
| lambda | the rate of the exponential |
| rescale | Rescale the function to have value 1?. Default = TRUE |

### Author(s)

Alistair Dunn

## PFinverselogistic *Evaluate the inverse-logistic preference function*

### Description

Evaluate the inverse-logistic preference function

### Usage

```
PFinverselogistic(x, alpha, a50, ato95, rescale = T)
```

### Arguments

| | |
|---|---|
| x | x-values over which to evaluate. Default = 1 |
| alpha | The value of the alpha parameter. Default = 1 |
| a50 | the a50 value of the inverse logisitic |
| ato95 | the ato95 value of the inverse logisitic |
| rescale | Rescale the function to have value 1?. Default = TRUE |

### Author(s)

Alistair Dunn

## PFlogistic *Evaluate the logistic preference function*

### Description

Evaluate the logistic preference function

### Usage

```
PFlogistic(x, alpha, a50, ato95, rescale = T)
```

### Arguments

| | |
|---|---|
| x | x-values over which to evaluate. Default = 1 |
| alpha | The value of the alpha parameter. Default = 1 |
| a50 | the a50 value of the logisitic |
| ato95 | the ato95 value of the logisitic |
| rescale | Rescale the function to have value 1?. Default = TRUE |

### Author(s)

Alistair Dunn

## PFnormal *Evaluate the normal preference function*

### Description

Evaluate the normal preference function

### Usage

```
PFnormal(x, alpha, mu, sigma, rescale = T)
```

### Arguments

| | |
|---|---|
| x | x-values over which to evaluate. Default = 1 |
| alpha | The value of the alpha parameter. Default = 1 |
| mu | the mean of the normal |
| sigma | the standard deviation of the normal |
| rescale | Rescale the function to have value 1?. Default = TRUE |

### Author(s)

Alistair Dunn

## PFthreshold *Evaluate the threshold preference function*

### Description

Evaluate the threshold preference function

### Usage

```
PFthreshold(x, alpha, N, lambda, rescale = T)
```

### Arguments

| | |
|---|---|
| x | x-values over which to evaluate. Default = 1 |
| alpha | The value of the alpha parameter. Default = 1 |
| N | the threshhold value |
| rescale | Rescale the function to have value 1?. Default = TRUE |

### Author(s)

Alistair Dunn

## spm.area *utility function*

### Description

utility function

### Usage

```
spm.area(corners)
```

### Author(s)

Alistair Dunn

## spm.convert.to.lines *utility function*

### Description

utility function

### Usage

```
spm.convert.to.lines(filename)
```

### Author(s)

Alistair Dunn

## spm.dplot *utility function*

### Description

utility function

### Usage

```
spm.dplot(..., name = T, quantiles = c(0.5), plot.mean = F,
  main = "", xlab = "", ylab = "", ylim, srtx = 0, bw = "nrd0",
  adjust = 1/3, adj = 0, fill = F, gap = 0.2)
```

### Author(s)

Alistair Dunn

## spm.extract.simulated.observation *utility function*

### Description

utility function

### Usage

```
spm.extract.simulated.observation(lines)
```

### Author(s)

Alistair Dunn

## spm.get.lines *utility function*

### Description

utility function

### Usage

```
spm.get.lines(lines, from = -1, to = -1, contains = "",
  starts.with = "", clip.to = "", clip.from = "",
  clip.to.match = "", clip.from.match = "", ...)
```

### Author(s)

Alistair Dunn

## spm.is.whole.number *utility function*

### Description

utility function

### Usage

```
spm.is.whole.number(string)
```

### Author(s)

Alistair Dunn

## spm.isin *utility function*

### Description

utility function

### Usage

```
spm.isin(x, y)
```

### Author(s)

Alistair Dunn

## spm.make.filename *utility function*

### Description

utility function

### Usage

```
spm.make.filename(file, path = "")
```

### Author(s)

Alistair Dunn

## spm.make.list *utility function*

### Description

utility function

### Usage

```
spm.make.list(lines)
```

### Author(s)

Alistair Dunn

## spm.make.table *utility function*

### Description

utility function

### Usage

```
spm.make.table(lines)
```

### Author(s)

Alistair Dunn

## spm.pos *utility function*

### Description

utility function

### Usage

```
spm.pos(vector, x)
```

### Author(s)

Alistair Dunn

## spm.pos.match *utility function*

### Description

utility function

### Usage

```
spm.pos.match(vector, regexp)
```

### Author(s)

Alistair Dunn

## spm.recodevector *utility function*

### Description

utility function

### Usage

```
spm.recodevector(in.data, from.vals, to.vals)
```

### Author(s)

Alistair Dunn

## spm.remove.first.words *utility function*

### Description

utility function

### Usage

```
spm.remove.first.words(string, words = 1)
```

### Author(s)

Alistair Dunn

## spm.remove.last.words *utility function*

### Description

utility function

### Usage

```
spm.remove.last.words(string, words = 1)
```

### Author(s)

Alistair Dunn

## spm.report.types *utility function*

### Description

utility function

### Usage

```
spm.report.types()
```

### Author(s)

Alistair Dunn

## spm.string.to.vector.of.numbers *utility function*

### Description

utility function

### Usage

```
spm.string.to.vector.of.numbers(string, sep = " ")
```

### Author(s)

Alistair Dunn

## spm.string.to.vector.of.words *utility function*

### Description

utility function

### Usage

```
spm.string.to.vector.of.words(string, sep = " ")
```

### Author(s)

Alistair Dunn

## spm.unpaste *utility function*

### Description

utility function

### Usage

```
spm.unpaste(string, sep = " ")
```

### Author(s)

Alistair Dunn

## spm.zeroFun *utility function*

### Description

utility function

### Usage

```
spm.zeroFun(x, delta = 1e-11)
```

### Author(s)

Alistair Dunn

# Index