

# Package: capushe (via r-universe)

August 24, 2024

**Type** Package

**Title** CALibrating Penalties Using Slope HEuristics

**Version** 1.1.2

**Date** 2011-07-13

**Author** Sylvain Arlot, Vincent Brault, Jean-Patrick Baudry, Cathy Maugis and Bertrand Michel

**Maintainer** Vincent Brault <vincent.brault@agroparistech.fr>

**Description** Calibration of penalized criteria for model selection. The calibration methods available are based on the slope heuristics.

**License** GPL (>= 2.0)

**LazyLoad** yes

**Depends** methods, graphics, MASS

**Collate** prog.R DDSE.R Djump.R capushe.R

**URL** <http://www.math.u-psud.fr/~brault/capushe.html>

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2023-11-27 13:25:41 UTC

## Contents

capushe-package . . . . .	2
BICAICcapushe . . . . .	3
capushe . . . . .	4
datacapushe . . . . .	6
DDSE . . . . .	7
Djump . . . . .	9
plot-methods . . . . .	11
validation . . . . .	12

<b>Index</b>	<b>14</b>
--------------	-----------

---

capushe-package

*Capushe*

---

## Description

This package includes functions for model selection via penalization. The model selection criterion has the following form:  $\gamma_n(\hat{s}_m) + scoef \times \kappa \times pen_{shape}(m)$ . Two algorithms based on the slope heuristics are proposed to calibrate the parameter  $\kappa$  in the penalty: the [data-driven slope estimation algorithm \(DDSE\)](#) and the [dimension jump algorithm \(Djump\)](#).

## Details

The data-driven slope estimation algorithm and the dimension jump algorithm are respectively implemented into the [DDSE](#) function and the [Djump](#) function. Some classes are defined for the outputs of [DDSE](#) and [Djump](#) and a [graphical](#) display is available for each one of these two classes. [DDSE](#) and [Djump](#) are both included in the [capushe](#) function which is the main function of the package.

## Author(s)

Sylvain Arlot, Vincent Brault, Jean-Patrick Baudry, Cathy Maugis and Bertrand Michel.

Maintainer: Vincent Brault <vincent.brault@math.u-psud.fr>

## References

<http://www.math.univ-toulouse.fr/~maugis/CAPUSHE.html>

<http://www.math.u-psud.fr/~brault/capushe.html>

Article: Baudry, J.-P., Maugis, C. and Michel, B. (2011) Slope heuristics: overview and implementation. *Statistics and Computing*, to appear. doi: 10.1007/s11222-011-9236-1

## See Also

[Djump](#) and [DDSE](#) for model selection algorithms based on the slope heuristics. [plot](#) for a graphical display of the two algorithms. [validation](#) to check that the slope heuristics can be applied confidently.

## Examples

```
data(datacapushe)
## capushe returns the same model with DDSE and Djump:
capushe(datacapushe)
## capushe also returns the model selected by AIC and BIC
capushe(datacapushe,n=1000)
## Djump only
Djump(datacapushe)
## DDSE only
DDSE(datacapushe)
## Graphical representations
```

```

plot(Djump(datacapushe))
plot(DDSE(datacapushe))
plot(capushe(datacapushe))
## Validation procedure
data(datapartialcapushe)
capushepartial=capushe(datapartialcapushe)
plot(capushepartial)
## Additional data
data(datavalidcapushe)
validation(capushepartial,datavalidcapushe) ## The slope heuristics should not
## be applied for datapartialcapushe.

```

---

BICAICcapushe

*AICcapushe and BICcapushe*


---

## Description

These functions return the model selected by the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC).

## Usage

```

AICcapushe(data, n)
BICcapushe(data, n)

```

## Arguments

data	data is a matrix or a data.frame with four columns of the same length and each line corresponds to a model: <ol style="list-style-type: none"> <li>1. The first column contains the model names.</li> <li>2. The second column contains the penalty shape values.</li> <li>3. The third column contains the model complexity values.</li> <li>4. The fourth column contains the minimum contrast value for each model.</li> </ol>
n	n is the sample size.

## Details

The penalty shape value should be increasing with respect to the complexity value (column 3). The complexity values have to be positive. n is necessary to compute AIC and BIC criteria. n is the size of sample used to compute the contrast values given in the data matrix. Do not confuse n with the size of the model collection which is the number of rows of the data matrix.

## Value

model	The model selected by AIC or BIC.
AIC	The corresponding value of AIC (for AICcapushe only).
BIC	The corresponding value of BIC (for BICcapushe only).

**Author(s)**

Vincent Brault

**References**

<http://www.math.univ-toulouse.fr/~maugis/CAPUSHE.html>

<http://www.math.u-psud.fr/~brault/capushe.html>

Article: Baudry, J.-P., Maugis, C. and Michel, B. (2011) Slope heuristics: overview and implementation. *Statistics and Computing*, to appear. doi: 10.1007/s11222-011-9236-1

**See Also**

[capushe](#) for a model selection function including AIC, BIC, the [DDSE](#) algorithm and the [Djump](#) algorithm.

**Examples**

```
data(datacapushe)
AICcapushe(datacapushe, n=1000)
BICcapushe(datacapushe, n=1000)
```

---

capushe

*Calibrating Penalties Using Slope HEuristics (CAPUSHE)*

---

**Description**

The capushe function proposes two algorithms based on the slope heuristics to calibrate penalties in the context of model selection via penalization.

**Usage**

```
capushe(data, n=0, pct=0.15, point=0, psi.rlm=psi.bisquare, coef=2,
        Careajump=0, Ctresh=0)
```

**Arguments**

data	data is a matrix or a data.frame with four columns of the same length and each line corresponds to a model: <ol style="list-style-type: none"> <li>1. The first column contains the model names.</li> <li>2. The second column contains the penalty shape values.</li> <li>3. The third column contains the model complexity values.</li> <li>4. The fourth column contains the minimum contrast value for each model.</li> </ol>
n	n is the sample size.

pct	Minimum percentage of points for the plateau selection. See <a href="#">DDSE</a> for more details.
point	Minimum number of point for the plateau selection (See <a href="#">DDSE</a> for more details). If point is different from 0, pct is obsolete.
psi.rlm	Weight function used by <a href="#">rlm</a> . See <a href="#">DDSE</a> for more details. <code>psi.rlm="lm"</code> for non robust linear regression.
scoef	Ratio parameter. Default value is 2.
Careajump	Constant of jump area (See <a href="#">Djump</a> for more details). Default value is 0 (no area).
Ctresh	Maximal treshold for the complexity associated to the penalty coefficient (See <a href="#">Djump</a> for more details). Default value is 0 (Maximal jump selected as the greater jump).

## Details

The model  $\hat{m}$  selected by the procedure fulfills

$$\hat{m} = \operatorname{argmin} \gamma_n(\hat{s}_m) + \text{scoef} \times \kappa \times \text{pen}_{\text{shape}}(m)$$

where

- $\kappa$  is the penalty coefficient.
- $\gamma_n$  is the empirical contrast.
- $\hat{s}_m$  is the estimator for the model  $m$ .
- *scoef* is the ratio parameter.
- *pen<sub>shape</sub>* is the penalty shape.

The *capushe* function calls the functions [DDSE](#) and [Djump](#) to calibrate  $\kappa$ , see the description of these functions for more details. In the case of equality between two penalty shape values, only the model with the smallest contrast is considered.

## Value

@DDSE	A list returned by the <a href="#">DDSE</a> function.
@DDSE@model	The model selected by the <a href="#">DDSE</a> function.
@DDSE@kappa	The vector of the successive slope values.
@DDSE@ModelHat	A list providing details about the model selected by the <a href="#">DDSE</a> function.
@DDSE@interval	A list about the "slope interval" corresponding to the plateau selected in <a href="#">DDSE</a> . See <a href="#">DDSE</a> for more details.
@DDSE@graph	A list computed for the <a href="#">plot</a> function.
@Djump	A list returned by the <a href="#">Djump</a> function.
@Djump@model	The model selected by the <a href="#">Djump</a> function.
@Djump@ModelHat	A list providing details about the model selected by the <a href="#">Djump</a> function.
@Djump@graph	A list computed for the <a href="#">plot</a> function.
@AIC_capushe	A list returned by the <a href="#">AICcapushe</a> function.
@BIC_capushe	A list returned by the <a href="#">BICcapushe</a> function.
@n	Sample size.

**Author(s)**

Vincent Brault

**References**

<http://www.math.univ-toulouse.fr/~maugis/CAPUSHE.html>

<http://www.math.u-psud.fr/~brault/capushe.html>

Article: Baudry, J.-P., Maugis, C. and Michel, B. (2011) Slope heuristics: overview and implementation. *Statistics and Computing*, to appear. doi: 10.1007/s11222-011-9236-1

**See Also**

[Djump](#), [DDSE](#), [AIC](#) or [BIC](#) to use only one of these model selection functions. [plot](#) for graphical displays of DDSE and Djump.

**Examples**

```
data(datacapushe)
capushe(datacapushe)
capushe(datacapushe, 1000)
```

---

datacapushe

*datacapushe*

---

**Description**

A dataframe example for the [capushe package](#) based on a simulated Gaussian mixture dataset in  $\mathbb{R}^3$ .

**Usage**

```
data(datacapushe)
```

**Format**

A data frame with 50 rows (models) and the following 4 variables:

`model` a character vector: model names.

`pen` a numeric vector: model penalty shape values.

`complexity` a numeric vector: model complexity values.

`contrast` a numeric vector: model contrast values.

## Details

The simulated dataset is composed of  $n = 1000$  observations in  $\mathbf{R}^3$ . It consists of an equiprobable mixture of three large "bubble" groups centered at  $\nu_1 = (0, 0, 0)$ ,  $\nu_2 = (6, 0, 0)$  and  $\nu_3 = (0, 6, 0)$  respectively. Each bubble group  $j$  is simulated from a mixture of seven components according to the following density distribution:

$$x \in \mathbf{R}^3 \rightarrow 0.4\Phi(x|\mu_1 + \nu_j, I_3) + \sum_{k=2}^7 0.1\Phi(x|\mu_k + \nu_j, 0.1I_3)$$

with  $\mu_1 = (0, 0, 0)$ ,  $\mu_2 = (0, 0, 1.5)$ ,  $\mu_3 = (0, 1.5, 0)$ ,  $\mu_4 = (1.5, 0, 0)$ ,  $\mu_5 = (0, 0, -1.5)$ ,  $\mu_6 = (0, -1.5, 0)$  and  $\mu_7 = (-1.5, 0, 0)$ . Thus the distribution of the dataset is actually a 21-component Gaussian mixture.

A model collection of spherical Gaussian mixtures is considered and the dataframe `datacapushe` contains the maximum likelihood estimations for each of these models. The number of free parameters of each model is used for the complexity values and  $pen_{shape}$  is defined by this complexity divided by  $n$ .

`datapartialcapushe` and `datavalidcapushe` can be used to run the `validation` function. `datapartialcapushe` only contains the models with less than 21 components. `datavalidcapushe` contains three models with 30, 40 and 50 components respectively.

## Source

<http://www.math.univ-toulouse.fr/~maugis/CAPUSHE.html>

## References

Article: Baudry, J.-P., Maugis, C. and Michel, B. (2011) Slope heuristics: overview and implementation. *Statistics and Computing*, to appear. doi: 10.1007/s11222-011-9236-1

## Examples

```
data(datacapushe)
capushe(datacapushe, n=1000)
## BIC, DDSE and Djump all three select the true model
plot(capushe(datacapushe))
## Validation:
data(datapartialcapushe)
capushepartial=capushe(datapartialcapushe)
data(datavalidcapushe)
validation(capushepartial, datavalidcapushe) ## The slope heuristics should not
## be applied for datapartialcapushe.
```

## Description

DDSE is a model selection function based on the slope heuristics.

**Usage**

```
DDSE(data, pct = 0.15, point = 0, psi.rlm = psi.bisquare, scoef = 2)
```

**Arguments**

<code>data</code>	data is a matrix or a data.frame with four columns of the same length and each line corresponds to a model: <ol style="list-style-type: none"> <li>1. The first column contains the model names.</li> <li>2. The second column contains the penalty shape values.</li> <li>3. The third column contains the model complexity values.</li> <li>4. The fourth column contains the minimum contrast value for each model.</li> </ol>
<code>pct</code>	Minimum percentage of points for the plateau selection. It must be between 0 and 1. Default value is 0.15.
<code>point</code>	Minimum number of point for the plateau selection. If point is different from 0, pct is obsolete.
<code>psi.rlm</code>	Weight function used by <code>rlm</code> . <code>psi.rlm="lm"</code> for non robust linear regression.
<code>scoef</code>	Ratio parameter. Default value is 2.

**Details**

Let  $M$  be the model collection and  $P = \{pen_{shape}(m), m \in M\}$ . The DDSE algorithm proceeds in four steps:

1. If several models in the collection have the same penalty shape value (column 2), only the model having the smallest contrast value  $\gamma_n(\hat{s}_m)$  (column 4) is considered.
2. For any  $p \in P$ , the slope  $\hat{\kappa}(p)$  (argument `@kappa`) of the linear regression (argument `psi.rlm`) on the couples of points  $\{(pen_{shape}(m), -\gamma_n(\hat{s}_m)); pen_{shape}(m) \geq p\}$  is computed.
3. For any  $p \in P$ , the model fulfilling the following condition is selected:
$$\hat{m}(p) = \operatorname{argmin} \gamma_n(\hat{s}_m) + scoef \times \hat{\kappa}(p) \times pen_{shape}(m).$$
This gives an increasing sequence of change-points  $(p_i)_{1 \leq i \leq I+1}$  (output `@ModelHat$point_breaking`). Let  $(N_i)_{1 \leq i \leq I}$  (output `@ModelHat$number_plateau`) be the lengths of each "plateau".
4. If point is different from 0, let  $\hat{i} = \max \{1 \leq i \leq I; N_i \geq point\}$  else let  $\hat{i} = \max \{1 \leq i \leq I; N_i \geq pct \sum_{l=1}^I N_l\}$  (output `@ModelHat$imax`). The model  $\hat{m}(p_{\hat{i}})$  (output `@model`) is finally returned.

The "slope interval" is the interval  $[a, b]$  where  $a = \inf\{\hat{\kappa}(p), p \in [p_{\hat{i}}, p_{\hat{i}+1}[\cap P\}$  and  $b = \sup\{\hat{\kappa}(p), p \in [p_{\hat{i}}, p_{\hat{i}+1}[\cap P\}$ .

**Value**

<code>@model</code>	The model selected by the DDSE algorithm.
<code>@kappa</code>	The vector of the successive slope values.
<code>@ModelHat</code>	A list describing the algorithm.
<code>@ModelHat\$model_hat</code>	The vector of preselected models $\hat{m}(p)$ .



@ModelHat\$point\_breaking  
The vector of the breaking points  $(p_i)_{1 \leq i \leq I+1}$ .

@ModelHat\$number\_plateau  
The vector of the lengths  $(N_i)_{1 \leq i \leq I}$ .

@ModelHat\$imax The rank  $\hat{i}$  of the selected plateau.

@interval A list about the "slope interval".

@interval\$interval  
The slope interval.

@interval\$percent\_of\_points  
The proportion  $N_{\hat{i}} / \sum_{l=1}^I N_l$ .

@graph A list computed for the `plot` method.

**Author(s)**

Vincent Brault

**References**

<http://www.math.univ-toulouse.fr/~maugis/CAPUSHE.html>

<http://www.math.u-psud.fr/~brault/capushe.html>

Article: Baudry, J.-P., Maugis, C. and Michel, B. (2011) Slope heuristics: overview and implementation. *Statistics and Computing*, to appear. doi: 10.1007/s11222-011-9236-1

**See Also**

`capushe` for a model selection function including `AIC`, `BIC`, the DDSE algorithm and the `Djump` algorithm. `plot` for graphical displays of the DDSE algorithm and the `Djump` algorithm.

**Examples**

```
data(datacapushe)
DDSE(datacapushe)
plot(DDSE(datacapushe))
## DDSE with "lm" for the regression
DDSE(datacapushe,psi.rlm="lm")
```

---

Djump

*Model selection by dimension jump*

---

**Description**

Djump is a model selection function based on the slope heuristics.

**Usage**

```
Djump(data,coef=2,Careajump=0,Ctresh=0)
```

**Arguments**

data	data is a matrix or a data.frame with four columns of the same length and each line corresponds to a model: <ol style="list-style-type: none"> <li>1. The first column contains the model names.</li> <li>2. The second column contains the penalty shape values.</li> <li>3. The third column contains the model complexity values.</li> <li>4. The fourth column contains the minimum contrast value for each model.</li> </ol>
scoef	Ratio parameter. Default value is 2.
Careajump	Constant of jump area. Default value is 0 (no area). In practice, it is advisable to take $Careajump = \sqrt{\frac{\log(n)}{n}}$ where $n$ is the number of observations.
Ctresh	Maximal treshold for the complexity associated to the penalty coefficient. Default value is 0 (Maximal jump selected as the greatest jump). In practice, it is advisable to take $Ctresh = \frac{n}{\log(n)}$ where $n$ is the number of observations.

**Details**

The Djump algorithm proceeds in three steps:

1. For all  $\kappa > 0$ , compute
$$m(\kappa) \in \operatorname{argmin}_{m \in M} \{\gamma_n(\hat{s}_m) + \kappa \times pen_{shape}(m)\}$$
This gives a decreasing step function  $\kappa \mapsto C_{m(\kappa)}$ .
2. Find  $\hat{\kappa}$  such that  $C_{m(\hat{\kappa})}$  corresponds to the greatest jump of complexity if  $C_{tresh} = 0$  else  $\hat{\kappa}$  such that
$$\hat{\kappa} = \inf\{\kappa > 0 : C_{m(\kappa)} \leq Ctresh\}.$$
3. Select  $\hat{m} = m(scoef \times \hat{\kappa})$  (output @model).

Arlot has proposed a jump area containing the maximal jump defined by :

$$[\kappa(1 - Careajump); \kappa(1 + Careajump)].$$

If  $Careajump > 0$ , Djump return the area with the greatest jump. In practice, it is advisable to take  $Careajump = \frac{\log(n)}{n}$  where  $n$  is the number of observations.

**Value**

@model	The model selected by the dimension jump method.
@ModelHat	A list describing the algorithm.
@ModelHat\$jump	The vector of jump heights.
@ModelHat\$kappa	The vector of the values of $\kappa$ at each jump.
@ModelHat\$model_hat	The vector of the selected models $m(\kappa)$ by the jump.
@ModelHat\$JumpMax	The location of the greatest jump.
@ModelHat\$Kopt	$\kappa_{opt} = scoef \hat{\kappa}$ .
@graph	A list computed for the <code>plot</code> method.

**Author(s)**

Vincent Brault

**References**<http://www.math.univ-toulouse.fr/~maugis/CAPUSHE.html><http://www.math.u-psud.fr/~brault/capushe.html>

Article: Baudry, J.-P., Maugis, C. and Michel, B. (2011) Slope heuristics: overview and implementation. *Statistics and Computing*, to appear. doi: 10.1007/s11222-011-9236-1

**See Also**

[capushe](#) for a model selection function including [AIC](#), [BIC](#), the [DDSE](#) algorithm and the [Djump](#) algorithm. [plot](#) for a graphical display of the [DDSE](#) algorithm and the [Djump](#) algorithm.

**Examples**

```
data(datacapushe)
Djump(datacapushe)
plot(Djump(datacapushe))
Djump(datacapushe, Careajump=sqrt(log(1000)/1000))
plot(Djump(datacapushe, Careajump=sqrt(log(1000)/1000)))
Djump(datacapushe, Ctresh=1000/log(1000))
plot(Djump(datacapushe, Ctresh=1000/log(1000)))
```

---

plot-methods

*Plot for capushe*


---

**Description**

The plot methods allow the user to check that the slope heuristics can be applied confidently.

Usage

`plot(x,newwindow=TRUE,ask=TRUE)` for [capushe](#).

`plot(x,newwindow=TRUE)` for [DDSE](#) and [Djump](#).

**Arguments**

<code>x</code>	Output of <a href="#">DDSE</a> , <a href="#">Djump</a> or <a href="#">capushe</a> .
<code>newwindow</code>	If <code>newwindow=TRUE</code> , a new window is created for each plot.
<code>ask</code>	If <code>ask=TRUE</code> , <code>plot</code> waits for the user to press a key to display the next plot (only for the class <code>capushe</code> ).

## Details

The graphical window of DDSE is composed of three graphics (see [DDSE](#) for more details):

**left** The left plot shows  $-\gamma_n(\hat{s}_m)$  with respect to the penalty shape values.

**topright** Successive slope values  $\hat{\kappa}(p)$ .

**bottomright** The bottomright plot shows the selected models  $\hat{m}(p)$  with respect to the successive slope values. The plateau in blue is selected.

The graphical window of Djump shows the complexity  $C_{m(\kappa)}$  of the selected model with respect to  $\kappa$ .  $\hat{\kappa}^{dj}$  corresponds to the greatest jump.  $\kappa_{opt}$  is defined by  $\kappa_{opt} = slope \times \hat{\kappa}^{dj}$ . The red line represents the slope interval computed by the DDSE algorithm (only for capushe). See [Djump](#) for more details.

## Methods

`signature(x = "Capushe")` This graphical function displays the DDSE plot and the Djump plot.

`signature(x = "DDSE")` This graphical function displays the [DDSE](#) plot.

`signature(x = "Djump")` This graphical function displays the [Djump](#) plot.

## Note

Use `newwindow=FALSE` to produce a PDF files (for an object of class `capushe`, use moreover `ask=FALSE`).

---

validation

*validation*

---

## Description

`validation` checks that the slope heuristics can be applied confidently.

## Usage

```
validation(x, data2, ...)
```

## Arguments

- |                    |  |
|--------------------|--|
| <code>x</code>     | <code>x</code> must be an object of <a href="#">class capushe</a> or <a href="#">DDSE</a> , in practice an output of the <a href="#">capushe</a> function or the <a href="#">DDSE</a> function.  |
| <code>data2</code> | <code>data2</code> is a matrix or a <code>data.frame</code> with four columns of the same length and each line corresponds to a model: <ol style="list-style-type: none"> <li>1. The first column contains the model names.</li> <li>2. The second column contains the penalty shape values.</li> <li>3. The third column contains the model complexity values.</li> <li>4. The fourth column contains the minimum contrast value for each model.</li> </ol> |
| <code>...</code>   | <ul style="list-style-type: none"> <li>• If <code>newwindow==TRUE</code>, a new window is created for the plot.</li> </ul>   |

**Details**

The validation function plots the additional and more complex models data2 to check that the linear relation between the penalty shape values and the contrast values (which is recorded in  $x$ ) is valid for the more complex models.

**Author(s)**

Vincent Brault

**References**

<http://www.math.univ-toulouse.fr/~maugis/CAPUSHE.html>

<http://www.math.u-psud.fr/~brault/capushe.html>

Article: Baudry, J.-P., Maugis, C. and Michel, B. (2011) Slope heuristics: overview and implementation. *Statistics and Computing*, to appear. doi: 10.1007/s11222-011-9236-1

**See Also**

[capushe](#) for a more general model selection function including [AIC](#), [BIC](#), the [DDSE](#) algorithm and the [Djump](#) algorithm.

**Examples**

```
data(datapartialcapushe)
capushepartial=capushe(datapartialcapushe)
data(datavalidcapushe)
validation(capushepartial,datavalidcapushe) ## The slope heuristics should not
## be applied for datapartialcapushe.
data(datacapushe)
plot(capushe(datacapushe))
```

# Index

- \* **datasets**
    - datacapushe, 6
  - \* **methods**
    - plot-methods, 11
  - \* **models**
    - BICAICcapushe, 3
    - capushe, 4
    - DDSE, 7
    - Djump, 9
    - validation, 12
  - \* **package**
    - capushe-package, 2
- AIC, 6, 9, 11, 13
- AICapushe (BICAICcapushe), 3
- Aicapushe (BICAICcapushe), 3
- aicapushe (BICAICcapushe), 3
- AICcapushe, 5
- AICcapushe (BICAICcapushe), 3
- Aiccapushe (BICAICcapushe), 3
- aiccapushe (BICAICcapushe), 3
- BIC, 6, 9, 11, 13
- BICAICcapushe, 3
- BICapushe (BICAICcapushe), 3
- Bicapushe (BICAICcapushe), 3
- bicapushe (BICAICcapushe), 3
- BICcapushe, 5
- BICcapushe (BICAICcapushe), 3
- Biccapushe (BICAICcapushe), 3
- biccapushe (BICAICcapushe), 3
- bubble (datacapushe), 6
- bubbles (datacapushe), 6
- CALibrating (capushe), 4
- Calibrating (capushe), 4
- CAPUSHE (capushe), 4
- Capushe (capushe), 4
- capushe, 2, 4, 4, 9, 11–13
- Capushe-class (capushe), 4
- capushe-package, 2
- capushe.package (capushe-package), 2
- capushepack (capushe-package), 2
- Data-Driven (DDSE), 7
- Data-driven (DDSE), 7
- data-driven (DDSE), 7
- data-driven slope estimation algorithm (DDSE), 2
- datacapushe, 6
- datapartialcapushe (datacapushe), 6
- datavalidcapushe (datacapushe), 6
- DDSE, 2, 4–6, 7, 11–13
- Ddse (DDSE), 7
- ddse (DDSE), 7
- DDSE-class (DDSE), 7
- dimension jump algorithm (Djump), 2
- Dimension\_Jump (Djump), 9
- Dimension\_jump (Djump), 9
- DimensionJump (Djump), 9
- Dimensionjump (Djump), 9
- DJUMP (Djump), 9
- Djump, 2, 4–6, 9, 9, 11–13
- djump (Djump), 9
- Djump-class (Djump), 9
- graphical, 2
- heuristic (capushe-package), 2
- plot, 2, 5, 6, 9–11
- plot, Capushe-method (capushe), 4
- plot, DDSE-method (DDSE), 7
- plot, Djump-method (Djump), 9
- plot-methods, 11
- plot.capushe (plot-methods), 11
- plot.DDSE (plot-methods), 11
- plot.Djump (plot-methods), 11
- plotcapushe (plot-methods), 11
- plotDDSE (plot-methods), 11

plotDjump (plot-methods), 11  
print, Capushe-method (capushe), 4  
print, DDSE-method (DDSE), 7  
print, Djump-method (Djump), 9  
print.capushe (capushe), 4  
print.DDSE (DDSE), 7  
print.Djump (Djump), 9

rlm, 5, 8

show, Capushe-method (capushe), 4  
show, DDSE-method (DDSE), 7  
show, Djump-method (Djump), 9  
show.capushe (capushe), 4  
show.DDSE (DDSE), 7  
show.Djump (Djump), 9  
slope (capushe-package), 2  
summary, Capushe-method (capushe), 4  
summary, DDSE-method (DDSE), 7  
summary, Djump-method (Djump), 9  
summary.capushe (capushe), 4  
summary.DDSE (DDSE), 7  
summary.Djump (Djump), 9

validation, 2, 7, 12  
validation, Capushe-method (capushe), 4  
validation, DDSE-method (DDSE), 7  
validation-methods (validation), 12  
validation.capushe (validation), 12  
validation.DDSE (validation), 12  
validationcapushe (validation), 12  
validationDDSE (validation), 12