Package 'autothresholdr'

June 12, 2019

Type Package

Title An R Port of the 'ImageJ' Plugin 'Auto Threshold'

Version 1.3.3

Maintainer Rory Nolan <rorynoolan@gmail.com>

Description Algorithms for automatically finding appropriate thresholds for numerical data, with special functions for thresholding images. Provides the 'ImageJ' 'Auto Threshold' plugin functionality to R users. See <http://imagej.net/Auto_Threshold> and Landini et al. (2017) <DOI:10.1111/jmi.12474>.

License GPL-3

LazyData TRUE

Depends R (>= 3.1)

Imports stats, magrittr (>= 1.5), Rcpp (>= 1.0.1), checkmate (>= 1.9.3), ijtiff (>= 2.0.0), filesstrings (>= 3.1.3), glue (>= 1.3.0), purrr, rlang (>= 0.3.3)

SystemRequirements C++11

RoxygenNote 6.1.1

URL https://rorynolan.github.io/autothresholdr/, https://www.github.com/rorynolan/autothresholdr

BugReports https://www.github.com/rorynolan/autothresholdr/issues

Suggests testthat, covr, knitr, rmarkdown, utils, ggplot2, dplyr, spelling

VignetteBuilder knitr

LinkingTo Rcpp (>= 1.0.1)

Encoding UTF-8

Language en-US

NeedsCompilation yes

Author Rory Nolan [aut, cre, trl] (<https://orcid.org/0000-0002-5239-4043>),
Luis Alvarez [ctb] (<https://orcid.org/0000-0003-1316-1906>),
Sergi Padilla-Parra [ctb, ths]
(<https://orcid.org/0000-0002-8010-9481>),
Gabriel Landini [ctb, cph] (<https://orcid.org/0000-0002-9689-0989>)

Repository CRAN

Date/Publication 2019-06-12 14:40:03 UTC

R topics documented:

`arr_mask` *Array mask class.*

Description

A *mask* of an array with respect to a given threshold is found by taking the original array and setting all elements falling below the threshold to `FALSE` and the others to `TRUE`. An object of class arr_mask has the attribute `thresh` detailing the threshold value that was applied.

Usage

```
arr_mask(arr, thresh)
```

Arguments

`arr`	An array of logicals (the mask).
`thresh`	The threshold. Either a scalar or an object of class th.

Value

An object of class `masked_arr`.

auto_thresh *Automatically threshold an array of non-negative integers.*

Description

These functions apply the ImageJ "Auto Threshold" plugin's image thresholding methods. The available methods are "IJDefault", "Huang", "Huang2", "Intermodes", "IsoData", "Li", "MaxEntropy", "Mean", "MinErrorI", "Minimum", "Moments", "Otsu", "Percentile", "RenyiEntropy", "Shanbhag", "Triangle" and "Yen". Read about them at http://imagej.net/Auto_Threshold.

Usage

```
auto_thresh(int_arr, method, ignore_black = FALSE,
  ignore_white = FALSE, ignore_na = FALSE)

auto_thresh_mask(int_arr, method, ignore_black = FALSE,
  ignore_white = FALSE, ignore_na = FALSE)

auto_thresh_apply_mask(int_arr, method, fail = NA,
  ignore_black = FALSE, ignore_white = FALSE, ignore_na = FALSE)

mask(int_arr, method, ignore_black = FALSE, ignore_white = FALSE,
  ignore_na = FALSE)

apply_mask(int_arr, method, fail = NA, ignore_black = FALSE,
  ignore_white = FALSE, ignore_na = FALSE)
```

Arguments

int_arr	An array (or vector) of non-negative *integers*.
method	The name of the thresholding method you wish to use. The available methods are "IJDefault", "Huang", "Huang2", "Intermodes", "IsoData", "Li", "MaxEntropy", "Mean", "MinErrorI", "Minimum", "Moments", "Otsu", "Percentile", "RenyiEntropy", "Shanbhag", "Triangle" and "Yen". Partial matching is performed i.e. method = "h" is enough to get you "Huang" and method = "in" is enough to get you "Intermodes". To perform *manual* thresholding (where you set the threshold yourself), supply the threshold here as a number e.g. method = 3; so note that this would *not* select the third method in the above list of methods.
ignore_black	Ignore black pixels/elements (zeros) when performing the thresholding?
ignore_white	Ignore white pixels when performing the thresholding? If set to TRUE, the function makes a good guess as to what the white (saturated) value would be (see 'Details'). If this is set to a number, all pixels with value greater than or equal to that number are ignored.
ignore_na	This should be TRUE if NAs in int_arr should be ignored or FALSE if you want the presence of NAs in int_arr to throw an error.

`fail` When using `auto_thresh_apply_mask()`, to what value do you wish to set the pixels which fail to exceed the threshold? `fail = 'saturate'` sets them to saturated value (see "Details"). `fail = 'zero'` sets them to zero. You can also specify directly here a natural number (must be between `0` and `2^16 - 1`) to use.

Details

- Values greater than or equal to the found threshold *pass* the thresholding and values less than the threshold *fail* the thresholding.
- For `ignore_white = TRUE`, if the maximum value in the array is one of `2^8-1`, `2^12-1`, `2^16-1` or `2^32-1`, then those max values are ignored. That's because they're the white values in 8, 12, 16 and 32-bit images respectively (and these are the common image bit sizes to work with). This guesswork has to be done because `R` does not know how many bits the image was on disk. This guess is very unlikely to be wrong, and if it is, the consequences are negligible anyway. If you're very concerned, then just specify the white value as an integer in this `ignore_white` argument.
- If you have set `ignore_black = TRUE` and/or `ignore_white = TRUE` but you are still getting error/warning messages telling you to try them, then your chosen method is not working for the given array, so you should try a different method.
- For a given array, if all values are less than `2^8`, saturated value is `2^8 - 1`, otherwise, saturated value is `2^16 - 1`.

Value

`auto_thresh()` returns an object of class th containing the threshold value. Pixels exceeding this threshold pass the thresholding, pixels at or below this level fail.

`auto_thresh_mask()` returns an object of class arr_mask which is a binarized version of the input, with a value of `TRUE` at points which exceed the threshold and `FALSE` at those which do not.

`auto_thresh_apply_mask()` returns and object of class threshed_arr which is the original input masked by the threshold, i.e. all points not exceeding the threshold are set to a user-defined value (default `NA`).

`mask()` is the same as `auto_thresh_mask()` and `apply_mask()` is the same as `auto_thresh_apply_mask()`.

Acknowledgements

Gabriel Landini coded all of these functions in Java. These java functions were then translated to C++.

References

- Huang, L-K & Wang, M-J J (1995), "Image thresholding by minimizing the measure of fuzziness", Pattern Recognition 28(1): 41-51
- Prewitt, JMS & Mendelsohn, ML (1966), "The analysis of cell images", Annals of the New York Academy of Sciences 128: 1035-1053
- Ridler, TW & Calvard, S (1978), "Picture thresholding using an iterative selection method", IEEE Transactions on Systems, Man and Cybernetics 8: 630-632

- Li, CH & Lee, CK (1993), "Minimum Cross Entropy Thresholding", Pattern Recognition 26(4): 617-625
- Li, CH & Tam, PKS (1998), "An Iterative Algorithm for Minimum Cross Entropy Thresholding", Pattern Recognition Letters 18(8): 771-776
- Sezgin, M & Sankur, B (2004), "Survey over Image Thresholding Techniques and Quantitative Performance Evaluation", Journal of Electronic Imaging 13(1): 146-165
- Kapur, JN; Sahoo, PK & Wong, ACK (1985), "A New Method for Gray-Level Picture Thresholding Using the Entropy of the Histogram", Graphical Models and Image Processing 29(3): 273-285
- Glasbey, CA (1993), "An analysis of histogram-based thresholding algorithms", CVGIP: Graphical Models and Image Processing 55: 532-537
- Kittler, J & Illingworth, J (1986), "Minimum error thresholding", Pattern Recognition 19: 41-47
- Prewitt, JMS & Mendelsohn, ML (1966), "The analysis of cell images", Annals of the New York Academy of Sciences 128: 1035-1053
- Tsai, W (1985), "Moment-preserving thresholding: a new approach", Computer Vision, Graphics, and Image Processing 29: 377-393
- Otsu, N (1979), "A threshold selection method from gray-level histograms", IEEE Trans. Sys., Man., Cyber. 9: 62-66, doi:10.1109/TSMC.1979.4310076
- Doyle, W (1962), "Operation useful for similarity-invariant pattern recognition", Journal of the Association for Computing Machinery 9: 259-267, doi:10.1145/321119.321123
- Kapur, JN; Sahoo, PK & Wong, ACK (1985), "A New Method for Gray-Level Picture Thresholding Using the Entropy of the Histogram", Graphical Models and Image Processing 29(3): 273-285
- Shanbhag, Abhijit G. (1994), "Utilization of information measure as a means of image thresholding", Graph. Models Image Process. (Academic Press, Inc.) 56 (5): 414–419, ISSN 1049-9652
- Zack GW, Rogers WE, Latt SA (1977), "Automatic measurement of sister chromatid exchange frequency", J. Histochem. Cytochem. 25 (7): 74153, PMID 70454
- Yen JC, Chang FJ, Chang S (1995), "A New Criterion for Automatic Multilevel Thresholding", IEEE Trans. on Image Processing 4 (3): 370-378, ISSN 1057-7149, doi:10.1109/83.366472
- Sezgin, M & Sankur, B (2004), "Survey over Image Thresholding Techniques and Quantitative Performance Evaluation", Journal of Electronic Imaging 13(1): 146-165

Examples

```
img_location <- system.file("extdata", "eg.tif", package = "autothresholdr")
img <- ijtiff::read_tif(img_location)
auto_thresh(img, "huang")
auto_thresh(img, "tri")
auto_thresh(img, "Otsu")
auto_thresh(img, 9)
mask <- auto_thresh_mask(img, "huang")
ijtiff::display(mask[, , 1, 1])
masked <- auto_thresh_apply_mask(img, "huang")
```

```
ijtiff::display(masked[, , 1, 1])
masked <- auto_thresh_apply_mask(img, 25)
ijtiff::display(masked[, , 1, 1])
```

`mean_stack_thresh` *Threshold every image frame in an image stack based on their mean.*

Description

An ijtiff_img is a 4-dimensional array indexed by `img[y, x, channel, frame]`. For each channel (which consists of a stack of frames), this function finds a threshold based on the sum all of the frames, uses this to create a mask and then applies this mask to every frame in the stack (so for a given pillar in the image stack, either all the pixels therein are thresholded away or all are untouched, where pillar `x,y` of channel `ch` is `img[y, x, ch, ]`).

Usage

```
mean_stack_thresh(img, method, fail = NA, ignore_black = FALSE,
  ignore_white = FALSE, ignore_na = FALSE)
```

Arguments

`img`	A 4-dimensional array in the style of an ijtiff_img (indexed by `img[y, x, channel, frame]`) or a 3-dimensional array which is a single channel of an ijtiff_img (indexed by `img[y, x, frame]`).
`method`	The name of the thresholding method you wish to use. The available methods are `"IJDefault"`, `"Huang"`, `"Huang2"`, `"Intermodes"`, `"IsoData"`, `"Li"`, `"MaxEntropy"`, `"Mean"`, `"MinErrorI"`, `"Minimum"`, `"Moments"`, `"Otsu"`, `"Percentile"`, `"RenyiEntropy"`, `"Shanbhag"`, `"Triangle"` and `"Yen"`. Partial matching is performed i.e. `method = "h"` is enough to get you `"Huang"` and `method = "in"` is enough to get you `"Intermodes"`. To perform *manual* thresholding (where you set the threshold yourself), supply the threshold here as a number e.g. `method = 3.8` (so note that this would *not* select the third method in the above list of methods). This manual threshold will then be used to threshold the sum stack to create a 2D mask and then this mask will be applied to all frames in the stack. If you want a different method for each channel, specify this parameter as a vector or list, one element per channel.
`fail`	When using `auto_thresh_apply_mask()`, to what value do you wish to set the pixels which fail to exceed the threshold? `fail = 'saturate'` sets them to saturated value (see 'Details'). `fail = 'zero'` sets them to zero. You can also specify directly here a natural number (must be between `0` and `2^16 - 1`) to use.
`ignore_black`	Ignore black pixels/elements (zeros) when performing the thresholding?
`ignore_white`	Ignore white pixels when performing the thresholding? If set to `TRUE`, the function makes a good guess as to what the white (saturated) value would be (see 'Details'). If this is set to a number, all pixels with value greater than or equal to that number are ignored.

ignore_na — This should be TRUE if NAs in int_arr should be ignored or FALSE if you want the presence of NAs in int_arr to throw an error.

Details

It's called mean_stack_thresh() and not sum_stack_thresh() because its easier for people to visualize the mean of an image series than to visualize the sum, but for the sake of this procedure, both are equivalent, except for the fact that the thresholding routine invoked inside this function prefers integers, which we get by using a sum but not by using a mean.

- Values greater than or equal to the found threshold *pass* the thresholding and values less than the threshold *fail* the thresholding.
- For ignore_white = TRUE, if the maximum value in the array is one of 2^8-1, 2^16-1 or 2^32-1, then those max values are ignored. That's because they're the white values in 8, 16 and 32-bit images respectively (and these are the common image bit sizes to work with). This guesswork has to be done because R does not know how many bits the image was on disk. This guess is very unlikely to be wrong, and if it is, the consequences are negligible anyway. If you're very concerned, then just specify the white value as an integer in this ignore_white argument.
- If you have set ignore_black = TRUE and/or ignore_white = TRUE but you are still getting error/warning messages telling you to try them, then your chosen method is not working for the given array, so you should try a different method.
- For a given array, if all values are less than 2^8, saturated value is 2^8 - 1, otherwise, saturated value is 2^16 - 1.

Value

An object of class stack_threshed_img which is the thresholded image (an array in the style of an ijtiff_img). Pillars not exceeding the threshold are set to the fail value (default NA).

Examples

```
img <- ijtiff::read_tif(system.file("extdata", "50.tif",
  package = "autothresholdr"
))
ijtiff::display(img[, , 1, 1])
img_thresh_mask <- mean_stack_thresh(img, "Otsu")
ijtiff::display(img_thresh_mask[, , 1, 1])
ijtiff::display(img[, , 1, 1])
img_thresh_mask <- mean_stack_thresh(img, "Huang")
ijtiff::display(img_thresh_mask[, , 1, 1])
```

`med_stack_thresh` *Threshold every image frame in a stack based on their median.*

Description

An ijtiff_img is a 4-dimensional array indexed by `img[y, x, channel, frame]`. For each channel (which consists of a stack of frames), this function finds a threshold based on all of the frames, then takes the median of all the frames in the stack image, uses this to create a mask with the found threshold and then applies this mask to every frame in the stack (so for a given pillar in the image stack, either all the pixels therein are thresholded away or all are untouched, where pillar `x,y` of channel `ch` is `img[y, x, ch, ]`).

Usage

```
med_stack_thresh(img, method, fail = NA, ignore_black = FALSE,
  ignore_white = FALSE, ignore_na = FALSE)
```

Arguments

`img`	A 3-dimensional array (the image stack, possibly a time-series of images) where the nth slice is the nth image in the stack.
`method`	The name of the thresholding method you wish to use. The available methods are `"IJDefault"`, `"Huang"`, `"Huang2"`, `"Intermodes"`, `"IsoData"`, `"Li"`, `"MaxEntropy"`, `"Mean"`, `"MinErrorI"`, `"Minimum"`, `"Moments"`, `"Otsu"`, `"Percentile"`, `"RenyiEntropy"`, `"Shanbhag"`, `"Triangle"` and `"Yen"`. Partial matching is performed i.e. `method = "h"` is enough to get you `"Huang"` and `method = "in"` is enough to get you `"Intermodes"`. To perform *manual* thresholding (where you set the threshold yourself), supply the threshold here as a number e.g. `method = 3` (so note that this would *not* select the third method in the above list of methods). This manual threshold will then be used to threshold the median stack to create a 2D mask and then this mask will be applied to all frames in the stack. If you want a different method for each channel, specify this parameter as a vector or list, one element per channel.
`fail`	When using `auto_thresh_apply_mask()`, to what value do you wish to set the pixels which fail to exceed the threshold? `fail = 'saturate'` sets them to saturated value (see 'Details'). `fail = 'zero'` sets them to zero. You can also specify directly here a natural number (must be between `0` and `2^32 - 1`) to use.
`ignore_black`	Ignore black pixels/elements (zeros) when performing the thresholding?
`ignore_white`	Ignore white pixels when performing the thresholding? If set to `TRUE`, the function makes a good guess as to what the white (saturated) value would be (see 'Details').
`ignore_na`	This should be `TRUE` if `NA`s in `int_arr` should be ignored or `FALSE` if you want the presence of `NA`s in `int_arr` to throw an error.

Details

- Values greater than or equal to the found threshold *pass* the thresholding and values less than the threshold *fail* the thresholding.
- For `ignore_white = TRUE`, if the maximum value in the array is one of `2^8-1`, `2^16-1` or `2^32-1`, then those max values are ignored. That's because they're the white values in 8, 16 and 32-bit images respectively (and these are the common image bit sizes to work with). This guesswork has to be done because `R` does not know how many bits the image was on disk. This guess is very unlikely to be wrong, and if it is, the consequences are negligible anyway. If you're very concerned, then just specify the white value as an integer in this `ignore_white` argument.
- If you have set `ignore_black = TRUE` and/or `ignore_white = TRUE` but you are still getting error/warning messages telling you to try them, then your chosen method is not working for the given array, so you should try a different method.
- For a given array, if all values are less than `2^8`, saturated value is `2^8 - 1`, otherwise, saturated value is `2^16 - 1`.

Value

An object of class stack_threshed_img which is the thresholded image (an array in the style of an ijtiff_img). Pillars not exceeding the threshold are set to the `fail` value (default `NA`).

Examples

```
img <- ijtiff::read_tif(system.file("extdata", "50.tif",
  package = "autothresholdr"
))
ijtiff::display(img[, , 1, 1])
img_thresh_mask <- med_stack_thresh(img, "Otsu")
ijtiff::display(img_thresh_mask[, , 1, 1])
ijtiff::display(img[, , 1, 1])
img_thresh_mask <- med_stack_thresh(img, "Triangle")
ijtiff::display(img_thresh_mask[, , 1, 1])
```

---

stack_threshed_img *Stack-thresholded image class.*

Description

A stack-thresholded array is an array which has had stack-thresholding applied to it. See `mean_stack_thresh()`. It has 3 necessary attributes:

- `thresh` is the threshold that was applied. This is either a number or an object of class th. Values in the original array which were less than this value are deemed to have failed the thresholding.
- `fail_value` is the value to which elements of the array which failed the thresholding were set. This could be something like `0` or `NA`.
- `stack_thresh_method` details which stacked-thresholding method was employed; this is either `"mean"` or `"median"`.

Usage

```
stack_threshed_img(img, thresh, fail_value, stack_thresh_method)
```

Arguments

img	A 4-dimensional array in the style of an ijtiff_img (indexed by img[y, x, channel, frame]) or a 3-dimensional array which is a single channel of an ijtiff_img (indexed by img[y, x, frame]).
thresh	The threshold that was used. Either a number or an object of class th.
fail_value	The value to which elements of the array which failed the thresholding were set.
stack_thresh_method	This must be set to either "mean" or "median" to tell which stacked-thresholding method was employed.

Value

An object of class stack_threshed_img.

See Also

threshed_arr, mean_stack_thresh(), med_stack_thresh().

---

th *Automatically found threshold class.*

Description

A threshold found automatically via auto_thresh(). It is a number (the value of the threshold) with 4 attributes:

- ignore_black is TRUE if black values were ignored during the thresholding and FALSE otherwise.
- ignore_white is TRUE if white values were ignored during the thresholding and FALSE otherwise.
- ignore_na is TRUE if NAs were ignored during the thresholding and FALSE otherwise.
- autothresh_method details which automatic thresholding method was used.

Usage

```
th(thresh, ignore_black, ignore_white, ignore_na, autothresh_method)
```

Arguments

thresh	A scalar. The threshold.
ignore_black	TRUE if black values were ignored during the thresholding and FALSE otherwise.
ignore_white	TRUE if white values were ignored during the thresholding and FALSE otherwise.
ignore_na	TRUE if NA values were ignored during the thresholding and FALSE otherwise.
autothresh_method	The name of the automatic thresholding method used.

Value

An object of class th.

---

threshed_arr *Thresholded array class.*

Description

A thresholded array is an array which has had a threshold applied to it. It has an attribute thresh which is the threshold that was applied which can be a number or an object of class th.

Usage

```
threshed_arr(arr, thresh)
```

Arguments

arr	The thresholded array (*not* the original array).
thresh	The threshold that was used. Either a number or an object of class th.

Details

The term 'array' is used loosely here in that vectors and matrices qualify as arrays.

Value

An object of class threshed_arr.

See Also

stack_threshed_img, apply_mask().

Index