Package 'DRIP'

May 12, 2024

License GPL (>= 2)

Version 2.0

Date 2024-04-24

Description A collection of functions that perform jump regression and image analysis such as denoising, deblurring and jump detection. The implemented methods are based on the following research: Qiu, P. (1998) <doi:10.1214/aos/1024691468>, Qiu, P. and Yandell, B. (1997) <doi:10.1080/10618600.1997.10474746>, Qiu, P. (2009) <doi:10.1007/s10463-007-0166-9>, Kang, Y. and Qiu, P. (2014) <doi:10.1080/00401706.2013.844732>, Qiu, P. and Kang, Y. (2015) <doi:10.5705/ss.2014.054>, Kang, Y., Mukherjee, P.S. and Qiu, P. (2018) <doi:10.1080/00401706.2017.1415975>, Kang, Y. (2020) <doi:10.1080/10618600.2019.1665536>.

Title Discontinuous Regression and Image Processing

Author Yicheng Kang [aut, cre], Peihua Qiu [aut, ctb]

Maintainer Yicheng Kang <kangyicheng0527@gmail.com>

Depends R (>= 3.5.0), parallel, graphics, stats

LazyData true

NeedsCompilation yes

Repository CRAN

Date/Publication 2024-05-12 21:03:35 UTC

R topics documented:

`brain` *Brain Image*

Description

This file contains data of a brain image. It has 217x217 pixels. Gray levels are in the range [0, 255]. In the data file, observations are listed as a 217 x217 matrix. This image has blur involved.

Usage

```
brain
```

Format

A matrix containing 217x217 pixels.

References

Kang, Y., Mukherjee, P.S. and Qiu, P. (2018) "Efficient Blind Image Deblurring Using Nonparametric Regression and Local Pixel Clustering", *Technometrics*, **60**(4), 522 – 531, doi:10.1080/00401706.2017.1415975.

circles *Image of Circles*

Description

This file contains the circles image. It has 256x256 pixels. Gray levels are in the range [0, 255]. In the data file, observations are listed as a 256x256 matrix.

Usage

```
circles
```

Format

A matrix of 256x256 pixels.

References

Qiu, P. (2005) *Image Processing and Jump Regression Analysis*. New Jersey: Wiley.

cv.jpex *Bandwidth Selection and Noise Level Estimation*

Description

Select the leave-one-out cross validation bandwidth for local linear kernel smoothing and estimates the noise level in the input image. Both the bandwidth parameter and the noise level are required inputs for the blind image deblurring procedure `jpex`.

Usage

```
cv.jpex(image, bandwidths, ncpus = 1)
```

Arguments

`image`	A blurry input image.
`bandwidths`	A vector of positive integers that specify the size of the neighborhood for local smoothing.
`ncpus`	The number of CPUs allocated for parallel computing.

Value

LLK	The estimated surface by local linear kernel (LLK) smoothing, using the CV selected bandwidth.
sigma	The estimated noise level, defined as the square root of the mean squared error (MSE) between LLK and the input image.
cv	A vector of the same length as that of the input bandwidths. Each element in the vector is the leave-one-out CV score associated with the corresponding bandwidth value.
bandwidth	The bandwidth parameters input by user.
band.min	The bandwidth parameter that results in the smallest CV score.

Author(s)

Yicheng Kang

References

Kang, Y. (2020) "Consistent Blind Image Deblurring Using Jump-Preserving Extrapolation", *Journal of Computational and Graphical Statistics*, **29**(2), 372 – 382, doi:10.1080/10618600.2019.1665536.

See Also

jpex

Examples

```
library(DRIP)
data(stopsign)
out <- cv.jpex(stopsign, c(2,3))
```

---

dKQ *Performance Measure of Edge Detector*

Description

Compute the dissimilarity measure between two sets of edge pixels. It is used as a performance measure for step or roof edge detectors.

Usage

```
dKQ(edge1, edge2)
```

Arguments

edge1	One set of pixels.
edge2	The other set of pixels.

Details

The mathematical definition of d_{KQ} is as follows. $d_{KQ}(S_1, S_2) = \frac{0.5}{|S_1|}\sum_{p_1 \in S_1} d_E(p_1, S_2) + \frac{0.5}{|S_2|}\sum_{p_2 \in S_2} d_E(p_2, S_1)$, where S_1 and S_2 are two point sets, and d_E denotes the Euclidean distance.

Value

Value of the d_{KQ}

References

Kang, Y. and Qiu, P., "Jump Detection in Blurred Regression Surfaces," *Technometrics*, **56**, 2014, 539-550, doi:10.1080/00401706.2013.844732.

Examples

```
mat1 <- matrix(c(1, rep(0, 3)), ncol = 2)
mat2 <- matrix(c(rep(0, 3), 1), ncol = 2)
dKQ(mat1, mat2)
```

`jpex` *Blind Image Deblurring*

Description

Take in any square matrix (noisy blurry image) and deblur it.

Usage

```
jpex(image, bandwidth, alpha, sigma)
```

Arguments

`image`	A square matrix representing a blurry image.
`bandwidth`	A positive integer that specifies the size of the neighborhood for local smoothing.
`alpha`	A numeric between 0 and 1. This is the significance level for the Chi-square hypothesis test. The null hypothesis is that a given pixel is in a continuity region and not affected by the blur.
`sigma`	A positive numeric value for the noise level in the blurred image. It is used in the Chi-square test.

Value

`deblurred`	A square matrix representing the deblurred image.
`edge`	A square matrix, the element of which is the value of the Chi-square test statistic at a pixel location. One can classify a given pixel as a blurry pixel if `edge[i, j] > qchisq(1 - alpha, 2)`.

Author(s)

Yicheng Kang

References

Kang, Y. (2020) "Consistent Blind Image Deblurring Using Jump-Preserving Extrapolation", *Journal of Computational and Graphical Statistics*, **29**(2), 372 – 382, doi:10.1080/10618600.2019.1665536.

See Also

`cv.jpex`

Examples

```
library(DRIP)
data(stopsign)
out <- jpex(image = stopsign, bandwidth = as.integer(2), sigma =
0.00623, alpha = 0.001)
```

`JPLLK_surface` *Jump-Preserving Local Linear Kernel Smoothing*

Description

Estimate surface using piecewise local linear kernel smoothing. The bandwidth is chosen by leave-one-out cross validation.

Usage

```
JPLLK_surface(image, bandwidth, plot = FALSE)
```

Arguments

`image`	A square matrix, no missing value allowed.
`bandwidth`	A numeric vector of positive integers, which specifies the number of pixels used in the local smoothing. The final fitted surface uses the optimal bandwidth chosen from those provided by users.
`plot`	If plot = TRUE, the image of the fitted surface is plotted.

Details

At each pixel, the gradient is estimated by a local linear kernel smoothing procedure. Next, the local neighborhood is divided into two halves along the direction perpendicular to $(\widehat{f}'_x, \widehat{f}'_y)$. Then the one-sided local linear kernel (LLK) estimates are obtained in the two half neighborhoods respectively. Among these two one-sided estimates, the one with smaller weighted mean square error is chosen to be the final estimate of the regression surface at the pixel.

Value

A list of fitted values, residuals, chosen bandwidth and estimated sigma.

References

Qiu, P. (2009) "Jump-Preserving Surface Reconstruction from Noisy Data", *Annals of the Institute of Statistical Mathematics*, **61**(3), 715 – 751, doi:10.1007/s1046300701669.

See Also

threeStage, surfaceCluster

Examples

```
data(sar) # SAR image is bundled with the package and it is a
          # standard test image in statistics literature.
fit <- JPLLK_surface(image=sar, bandwidth=c(3, 4))
```

---

kid *Image of a Kid*

Description

This file contains the kid image. The image has 387x387 pixels. Gray levels are in the range [0, 255]. In the data file, observations are listed as a 387x387 matrix. This image has spatially variant blur involved.

Usage

```
kid
```

Format

A matrix of 387x387 pixels.

References

Kang, Y. and Qiu, P. (2014) "Jump Detection in Blurred Regression Surfaces," *Technometrics*, **56**(4), 539 – 550, doi:10.1080/00401706.2013.844732.

lena *Lena Image*

Description

This file contains the Lena image. It has 512x512 pixels. Gray levels are in the range [0, 255]. In the data file, observations are listed as a 512x512 matrix.

Usage

```
lena
```

Format

A 512x512 matrix.

Source

November 1972 issue of Playboy magazine.

modify1 *Type-1 Modification Procedure*

Description

Modify detected edges to make them thin.

Usage

```
modify1(bandwidth, image, edge, plot)
```

Arguments

image	A matrix that represents the image.
bandwidth	A positive integer that specifies the number of pixels to use in the local smoothing.
edge	A matrix of 0 and 1 represents detected edge pixels.
plot	If plot = TRUE, images of detected edges before and after the modification are plotted.

Details

A local-smoothing based edge detection algorithm may flag deceptive edge pixel candidates. One kind of such candidates consists of those close to the real edges. They occur due to the nature of local smoothing. That is, if the point (x_i, y_j) is flagged, then its neighboring pixels will be flagged with high probability. This kind of deceptive candidates can make the detected edges thick. This modification procedure makes the detected edges thin.

Value

A matrix of zeros and ones of the same size as the input image.

References

Qiu, P. and Yandell, B. (1997) "Jump Detection in Regression Surfaces," *Journal of Computational and Graphical Statistics* **6(3)**, 332-354, doi:10.1080/10618600.1997.10474746.

See Also

modify2

Examples

```
edge <- stepEdge(sar, bandwidth = 4, thresh = 20, degree = 0)
out <- modify1(4, sar, edge)
```

---

modify2 *Type-2 Modification Procedure*

Description

Delete deceptive edge pixels that are scattered in the design space.

Usage

```
modify2(bandwidth, edge, plot)
```

Arguments

bandwidth	A positive integer that specifies the number of pixels to use in the local smoothing.
edge	A matrix of 0 and 1 representing the detected edge pixels.
plot	If plot = TRUE, images of the detected edges before and after the modification are plotted.

Details

A local-smoothing based edge detection algorithm may flag deceptive edge pixel candidates. One kind of such candidates existis due to the nature of hypothesis testing, on which the threshold value of the edge detection criterion is based. That is, a point (x_i, y_j) could be flagged as a edge pixel with certain probability, even if it is actually not a edge pixel. Deceptive candidates of this kind are scattered in the whole design space. This modification procedure deletes scattered edge pixel candidates.

Value

A matrix of zeros and ones of the same size as the input image.

References

Qiu, P. and Yandell, B. (1997) "Jump Detection in Regression Surfaces," *Journal of Computational and Graphical Statistics* **6(3)**, 332-354, doi:10.1080/10618600.1997.10474746.

See Also

modify1

Examples

```
edge <- stepEdge(sar, bandwidth = 4, thresh = 20, degree = 0)
out <- modify2(4, edge)
```

---

peppers *Image of Peppers*

Description

This file contains the peppers image. It has 512x512 pixels. Gray levels are in the range [0, 255]. In the data file, observations are listed as a 512x512 matrix.

Usage

```
peppers
```

Format

A matrix of 512x512 pixels.

References

Kang, Y. (2020) “Consistent Blind Image Deblurring Using Jump-Preserving Extrapolation”, *Journal of Computational and Graphical Statistics*, **29**(2), 372 – 382, doi:10.1080/10618600.2019.1665536.

---

roofDiff *Roof Edge Detection Statistics*

Description

Compute the difference between two one-sided gradient estimates.

Usage

```
roofDiff(image, bandwidth, blur)
```

Arguments

image	A square matrix, no missing value allowed.
bandwidth	A positive integer that specifies the number of pixels to use in the local smoothing.
blur	If blur = TRUE, besides the conventional 2-D kernel function, a univariate kernel function is used to address the issue of blur.

Details

At each pixel, the second-order derivarives (i.e., f''_{xx}, f''_{xy} and f''_{yy}) are estimated by a local quadratic kernel smoothing procedure. Next, the local neighborhood is first divided into two halves along the direction perpendicular to ($\widehat{f}''_{xx}$, $\widehat{f}''_{xy}$). Then the one-sided estimates of f'_{x+} and f'_{x-} are obtained respectively by local linear kernel smoothing. The estimates of f'_{y+} and f'_{y-} are obtained by the same procedure except that the neighborhood is divided along the direction perpendicular to ($\widehat{f}''_{xy}$, $\widehat{f}''_{yy}$).

Value

A matrix where each entry is the maximum of the differences: $|\widehat{f}_{x+} - \widehat{f}_{x-}|$ and $|\widehat{f}_{y+} - \widehat{f}_{y-}|$ at each pixel location.

References

Qiu, P. and Kang, Y. (2015) "Blind Image Deblurring Using Jump Regression Analysis", *Statistica Sinica*, **25**, 879 – 899, doi:10.5705/ss.2014.054.

See Also

roofEdgeParSel, roofEdge

Examples

```
data(peppers)
diff <- roofDiff(image = peppers, bandwidth = 8) # Time consuming
```

---

roofEdge *Roof Edge Detector*

Description

Detect roof/valley edges in an image using piecewise local quadratic kernel smoothing.

Usage

```
roofEdge(image, bandwidth, thresh, edge1, blur, plot)
```

Arguments

image	A square matrix, no missing value allowed.
bandwidth	A positive integer that specifies the number of pixels to use in the local smoothing.
thresh	Threshold value to use in the edge detection criterion.
edge1	A square matrix representing the image's step edges. The function excludes step edges when detects roof edges.
blur	If blur = TRUE, besides the conventional 2-D kernel function, a univariate kernel function is used in the local smoothing to address the issue of blur.
plot	If plot = TRUE, an image of detected edges is plotted.

Details

At each pixel, the second-order derivarives (i.e., f''_{xx}, f''_{xy}, and f''_{yy}) are estimated by a local quadratic kernel smoothing procedure. Next, the local neighborhood is first divided into two halves along the direction perpendicular to ($\widehat{f}''_{xx}$, $\widehat{f}''_{xy}$). Then the one-sided estimates of f'_{x+} and f'_{x-} are obtained respectively by local linear kernel smoothing. The estimates of f'_{y+} and f'_{y-} are obtained by the same procedure except that the neighborhood is divided along the direction perpendicular to ($\widehat{f}''_{xy}$, $\widehat{f}''_{yy}$). The pixel is flagged as a roof/valley edge pixel if $max(|\widehat{f}_{x+} - \widehat{f}_{x-}|, |\widehat{f}_{y+} - \widehat{f}_{y-}|) >$ the specified threshold and there is no step edge pixels in the neighborhood.

Value

A matrix of zeros and ones of the same size as the input image.

References

Qiu, P. and Kang, Y. (2015) "Blind Image Deblurring Using Jump Regression Analysis", *Statistica Sinica*, **25**, 879 – 899, doi:10.5705/ss.2014.054.

See Also

roofEdgeParSel, roofDiff

Examples

```
data(peppers)
## Not run:
step.edges <- stepEdge(peppers, bandwidth = 6, thresh = 25, degree = 1)
roof.edges <- roofEdge(image = peppers, bandwidth = 9, thresh = 3000,
    edge1 = step.edges, blur = FALSE, plot = FALSE) # Time consuming
edges <- step.edges + roof.edges
par(mfrow = c(2, 2))
image(1 - step.edges, col = gray(0:1))
image(1 - roof.edges, col = gray(0:1))
image(1 - edges, col = gray(0:1))
image(peppers, col = gray(c(0:255)/255))

## End(Not run)
```

`roofEdgeParSel` *Parameter Selection in Roof Edge Detection*

Description

Select bandwidth and threshold value for the roof/valley edge detector using bootstrap.

Usage

```
roofEdgeParSel(image, bandwidth, thresh, nboot, edge1, blur = FALSE)
```

Arguments

`image`	A square matrix object of size n by n, no missing value allowed.
`bandwidth`	Positive integers to specify the number of pixels used in the local smoothing. These are the bandwidth parameters to be chosen from.
`thresh`	Threshold values to be chosen from.
`nboot`	Number of bootstrap samples.
`edge1`	Step edges. The function excludes step edges when detect roof/valley edges.
`blur`	TRUE if the image contains blur, FALSE otherwise.

Details

If *blur=TRUE*, then a conventional local linear kernel smoothing is applied to estimate the blurred surface; Bootstrap samples are obtained by drawing with replacement from the residuals and the d_{KQ} is computed for the detected edges of the original sample and those of the bootstrap samples. If *blur=FALSE*, the procedure is the same as when *blur=TRUE* except that a jump-preserving kernel smoothing procedure is used to obtain residuals.

Value

Returns a list of the selected bandwdith, the selected threshold value, and a matrix of d_{KQ} values with each entry corresponding to each combination of bandwdith and threshold.

References

Qiu, P. and Kang, Y. (2015) "Blind Image Deblurring Using Jump Regression Analysis", *Statistica Sinica*, **25**, 879-899, doi:10.5705/ss.2014.054

See Also

`roofDiff`, `roofEdge`

Examples

```
## Not run:
step.edges <- stepEdge(peppers, bandwidth = 9, thresh = 17)
set.seed(24)
parSel <- roofEdgeParSel(image = peppers, bandwidth = 5, thresh = 5000,
    nboot = 1, edge1 = step.edges, blur = TRUE) # Time Consuming

## End(Not run)
```

---

sar *Synthetic Aperture Radar Image*

Description

This file contains the synthetic aperture radar (SAR) image of an area near Thetford forest, England. The image has 250x250 pixels. Gray levels are in the range [0, 255]. In the data file, observations are listed as a 250x250 matrix. This image is noisy.

Usage

```
sar
```

Format

A matrix of 250x250 pixels.

Source

https://users.phhp.ufl.edu/pqiu/research/book/data/index.html

References

Qiu, P. (2005) *Image Processing and Jump Regression Analysis*. New Jersey: Wiley.

---

stepDiff *Step Edge Detection Statistics*

Description

Compute difference between two one-sided local kernel estimates along the gradient direction.

Usage

```
stepDiff(image, bandwidth, degree = 1, blur = FALSE, plot = FALSE)
```

Arguments

image	A square matrix, no missing value allowed.
bandwidth	A positive integer that specifies the number of pixels to use in the local smoothing.
degree	An integer equal to 0 for local constant kernel smoothing or 1 for local linear kernel smoothing. The default value is 1.
blur	If blur = TRUE, in addition to a conventional 2-D kernel function, a 1-D kernel is used in local smoothing to address the issue of blur. The default value is FALSE.
plot	If plot = TRUE, an image of the detection statistics at each pixel is plotted.

Details

At each pixel, the gradient is estimated by a local linear kernel smoothing procedure. Next, the local neighborhood is divided into two halves along the direction perpendicular to $(\widehat{f}'_x, \widehat{f}'_y)$. Then the one- sided local kernel estimates are obtained in the two half neighborhoods respectively.

Value

A matrix of the estimated difference, $|\widehat{f}_+ - \widehat{f}_-|$, at each pixel.

References

Kang, Y. and Qiu, P. (2014) "Jump Detection in Blurred Regression Surfaces," *Technometrics*, **56**(4), 539 – 550, doi:10.1080/00401706.2013.844732.

See Also

roofDiff

Examples

```
data(sar) # SAR image is bundled with the package and it is a
          # standard test image in statistics literature.
diff <- stepDiff(image = sar, bandwidth = 4, degree = 0)
```

---

stepEdge *Step Edge Detector*

Description

Detect step edges in an image.

Usage

```
stepEdge(image, bandwidth, thresh, degree = 1, blur = FALSE,
plot = FALSE)
```

Arguments

image	A square matrix, no missing value allowed.
bandwidth	A positive integer that specifies the number of pixels to use in the local smoothing.
thresh	The threshold value to use in the edge detection criterion. Must be a positive value.
degree	An integer equal to 0 for local constant kernel smoothing or 1 for local linear kernel smoothing. The default value is 1.
blur	If blur = TRUE, in addition to a conventional 2-D kernel function, a 1-D kernel is used in local smoothing to address the issue of blur. The default value is FALSE.
plot	If plot = TRUE, an image of the detected edges is plotted.

Details

At each pixel, the gradient is estimated by a local kernel smoothing procedure. Next, the local neighborhood is divided into two halves along the direction perpendicular to $(\widehat{f}'_x, \widehat{f}'_y)$. Then the one-sided local kernel estimates are obtained in the two half neighborhoods respectively. The pixel is flagged as a step edge pixel if $|\widehat{f}_+ - \widehat{f}_-| > u$, where u is the specified threshold value.

Value

A matrix of zeros and ones. Ones represent the detected edge pixels and zeros represent the non-edge pixels.

References

Kang, Y. and Qiu, P. (2014) "Jump Detection in Blurred Regression Surfaces," *Technometrics*, **56**(4), 539 – 550, doi:10.1080/00401706.2013.844732.

See Also

roofDiff, stepDiff, roofEdge

Examples

```
data(sar)
edges <- stepEdge(image = sar, bandwidth = 4, degree = 0,
thresh = 16)
```

stepEdgeParSel *Parameter Selection in Step Edge Detection*

Description

Select the bandwidth and threshold parameters for step edge detection.

Usage

```
stepEdgeParSel(image, bandwidth, thresh, nboot, degree = 1,
blur = FALSE)
```

Arguments

image	A square matrix, no missing value allowed.
bandwidth	A positive integer that specifies the number of pixels to use in the local smoothing.
thresh	The threshold value to use in the edge detection criterion. Must be a positive value.
nboot	Number of bootstrap samples to use in estimating d_{KQ}.
degree	An integer equal to 0 for local constant kernel smoothing or 1 for local linear kernel smoothing. The default value is 1.
blur	If blur = TRUE, in addition to a conventional 2-D kernel function, a 1-D kernel is used in local smoothing to address the issue of blur. The default value is FALSE.

Details

A jump-preserving local linear kernel smoothing is applied to estimate the discontinuous regression surface; Bootstrap samples are obtained by drawing with replacement from the residuals and the d_{KQ} is computed for the detected edges of the original sample and those of the bootstrap samples.

Value

A list of the selected bandwidth, the selected threshold value and a matrix of d_{KQ} values with each entry corresponding to each combination of bandwidth and threshold.

References

Kang, Y. and Qiu, P. (2014) "Jump Detection in Blurred Regression Surfaces," *Technometrics*, **56**(4), 539 – 550, doi:10.1080/00401706.2013.844732.

See Also

roofDiff, stepDiff, roofEdge

Examples

```
set.seed(24)
parSel <- stepEdgeParSel(image = sar, bandwidth = 5,
thresh = c(17, 21), nboot = 1)
```

`stopsign` *Stop Sign Image*

Description

This file contains the stop sign image. The image has 160x160 pixels. Gray levels are in the range [0, 255]. In the data file, observations are listed as a 160x160 matrix. This image has much blurring involved.

Usage

```
stopsign
```

Format

A matrix of 160x160 pixels.

References

Kang, Y. (2020) "Consistent Blind Image Deblurring Using Jump-Preserving Extrapolation", *Journal of Computational and Graphical Statistics*, **29**(2), 372 – 382, doi:10.1080/10618600.2019.1665536.

`surfaceCluster` *Jump-Preserving Surface Estimation Using Pixel Clustering*

Description

Estimate surface using local pixel clustering and kernel smoothing. The bandwidth parameter is specified by the user.

Usage

```
surfaceCluster(image, bandwidth, sig.level, sigma, phi0,
    mean_std_abs, cw=3, blur = FALSE, plot = FALSE)
```

Arguments

image	A square matrix, no missing value allowed.
bandwidth	A positive integer that specifies the number of pixels to use in the local smoothing.
sig.level	The significance level for the hypothesis test deciding whether to cluster pixels or not.
sigma	The noise level (i.e., standard deviation of the error distribution). It is used for computing the asymptotic threshold for residuals, which are defined to be the difference between the local linear kernel smoothing output and the center weighted median filter output. If not specified by the user, a jump-preserving local linear kernel smoothing surface estimation (Qiu 2009) is used to obtain an estimated sigma.
phi0	The density of the standardized error distribution at 0. It is used for computing the asymptotic threshold for residuals, whose definition is given above. If not specified by the user, a jump-preserving local linear kernel smoothing surface estimation (Qiu 2009) is used to obtain an estimated value.
mean_std_abs	The mean of absolute value of the standardized error. It is used for computing the asymptotic threshold for residuals, whose definition is given above. If not specified by the user, a jump-preserving local linear kernel smoothing surface estimation (Qiu 2009) is used to obtain an estimated value.
cw	The center weight for the center weighted median filter. It must be a positive integer.
blur	If blur = TRUE, besides a conventional 2-D kernel function, a univariate increasing kernel function is used in the local kernel smoothing to address the issue with blur.
plot	If plot = TRUE, the image of the fitted surface is plotted

Value

A list of 'estImg', the restored image, 'sigma', the estimated standard deviation of the random error, 'phi0', the estimated density of the error distribution at 0, and 'mean_std_abs', the estimated absolute mean of the error distribution.

References

Kang, Y., Mukherjee, P.S. and Qiu, P. (2017) "Efficient Blind Image Deblurring Using Nonparametric Regression and Local Pixel Clustering", *Technometrics*, **60**(4), 522 – 531, doi:10.1080/00401706.2017.1415975.

Qiu, P. (2009) "Jump-Preserving Surface Reconstruction from Noisy Data", *Annals of the Institute of Statistical Mathematics*, **61**, 715 – 751, doi:10.1007/s1046300701669.

See Also

JPLLK_surface, threeStage

Examples

```
data(brain)
fit <- surfaceCluster(image = brain, bandwidth = 4,
    sig.level = .9995, cw = 3, blur = TRUE)
```

`surfaceCluster_bandwidth` *Bandwidth Selection for Clustering-Based Surface Estimation*

Description

Select the bandwidth parameter for the function `surfaceCluster` using cross validation. In the cases when there is no blur involved (i.e., denoising only), leave-one-out cross validation is used. In the cases when there is blur involved, a modified cross validation is used.

Usage

```
surfaceCluster_bandwidth(image, bandwidths, sig.level, sigma,
 phi0, mean_std_abs, relwt = 0.5, cw = 3, blur = FALSE)
```

Arguments

`image`	A square matrix, no missing value allowed.
`bandwidths`	An array of positive integers that specifies the candidate bandwidth parameters. All the array elements must be positive integers because the bandwidth is specified in terms of number of pixels.
`sig.level`	The significance level for the hypothesis test deciding whether to cluster pixels or not.
`sigma`	The noise level (i.e., standard deviation of the error distribution). It is used for computing the asymptotic threshold for residuals, which are defined to be the difference between the local linear kernel smoothing output and the center weighted median filter output. If not specified by the user, a jump-preserving local linear kernel smoothing surface estimation (Qiu 2009) is used to obtain an estimated sigma.
`phi0`	The density of the standardized error distribution at 0. It is used for computing the asymptotic threshold for residuals, whose definition is given above. If not specified by the user, a jump-preserving local linear kernel smoothing surface estimation (Qiu 2009) is used to obtain an estimated value.
`mean_std_abs`	The mean of absolute value of the standardized error. It is used for computing the asymptotic threshold for residuals, whose definition is given above. If not specified by the user, a jump-preserving local linear kernel smoothing surface estimation (Qiu 2009) is used to obtain an estimated value.
`relwt`	The relative weight assigned to the cross validation score in the continuity region. That is, 1 - relwt is assigned to the cross validation score around the step edges. It is used only when there is blur involved.

cw	The center weight for the center weighted median filter. It must be a positive integer.
blur	If blur = TRUE, besides a conventional 2-D kernel function, a univariate increasing kernel function is used in the local kernel smoothing to address the issue with blur.

Value

A list: 'cv_dataframe', a data frame containing the cross validation scores corresponding to each candidate bandwidth, 'bandwidth_hat', the selected bandwidth, 'sigma', the estimated standard deviation of the random error, 'phi0', the estimated density of the error distribution at 0, and 'mean_std_abs', the estimated absolute mean of the error distribution.

References

Kang, Y., Mukherjee, P.S. and Qiu, P. (2017) "Efficient Blind Image Deblurring Using Nonparametric Regression and Local Pixel Clustering", *Technometrics*, **60**(4), 522 – 531, doi:10.1080/00401706.2017.1415975.

Qiu, P. (2009) "Jump-Preserving Surface Reconstruction from Noisy Data", *Annals of the Institute of Statistical Mathematics*, **61**, 715 – 751, doi:10.1007/s1046300701669.

See Also

JPLLK_surface, threeStage

Examples

```
data(brain)
bandwidth_select <- surfaceCluster_bandwidth(image = brain,
bandwidths = c(3:4), sig.level = .9995, blur = TRUE)
```

---

threeStage *Three-Stage Denoising and Deblurring*

Description

Estimate jump location curves using local principal component lines. One-sided kernel smoothing is then used for surface estimation. Bandwidth is specified by the user.

Usage

```
threeStage(image, bandwidth, edge1, edge2,
    blur = FALSE, plot = FALSE)
```

Arguments

image	A square matrix, no missing value allowed.
bandwidth	A positive integer that specifies the number of pixels to use in the local smoothing.
edge1	A matrix of 0 and 1 representing the detected step edge pixels
edge2	A matrix of 0 and 1 representing the detected roof/valley edge pixels
blur	If blur = TRUE, besides a conventional 2-D kernel function, a univariate increasing kernel function is used in the local kernel smoothing to address the issue with blur.
plot	If plot = TRUE, the image of the fitted surface is plotted

Details

At each pixel, if there are step edges detected in the local neighborhood, a principal component line is fitted through the detected edge pixels to approximate the step edge locally and then the regression surface is estimated by a local constant kernel smoothing procedure using only the pixels on one side of the principal component line. If there are no step edges but roof/valley edges detected in the local neighborhood, the same procedure is followed except that the principal component line to fitted through the detected roof/valley edge pixels. In cases when there is either no step edges or roof/valley edges detected in the neighborhood, the regression surface at the pixel is estimated by the conventional local linear kernel smoothing procedure.

Value

The restored image, which is represented by a matrix.

References

Qiu, P. and Kang, Y. (2015) "Blind Image Deblurring Using Jump Regression Analysis", *Statistica Sinica*, **25**, 879 – 899, doi:10.5705/ss.2014.054.

See Also

JPLLK_surface, surfaceCluster

Examples

```
step.edges <- stepEdge(sar, bandwidth = 4, thresh = 20, degree = 0)
stepEdge1 <- modify2(bandwidth = 4, step.edges)
fit <- threeStage(image = sar, bandwidth = 4, edge1 = stepEdge1,
    edge2 = array(0, rep(ncol(sar), 2)))
```

threeStageParSel *Parameter Selection in Three-Stage Image Restoration*

Description

Select the bandwidth value for the image restoration method implemented in the function threeStage

Usage

```
threeStageParSel(image, bandwidth, edge1, edge2, nboot,
    blur = FALSE)
```

Arguments

image	A square matrix, no missing value allowed.
bandwidth	Bandwidth values to be chosen from. Each of these values need to be an positive integer specifying the number of pixels to use in the local smoothing.
edge1	A matrix of 0 and 1 representing the detected step edge pixels.
edge2	A matrix of 0 and 1 representing the detected roof/valley edge pixels.
nboot	Required when blur is TRUE. Unused when blur is FALSE. It must be a positive integer specifying the number of bootstraps to perform. See Qiu and Kang (2015) for details.
blur	TRUE if the image contains blur, FALSE otherwise. If TRUE, the hybrid selection method proposed in Qiu and Kang (2015) is used. If FALSE, the leave-one-out cross validation is used.

Value

A list of the selected bandwidth, and a matrix of cross-validation scores with each entry corresponding to a choice of bandwidth.

References

Qiu, P. and Kang, Y. (2015) "Blind Image Deblurring Using Jump Regression Analysis", *Statistica Sinica*, **25**, 879 – 899, doi:10.5705/ss.2014.054.

Examples

```
## Not run:
step.edges <- stepEdge(peppers, bandwidth = 9, thresh = 17)
roof.edges <- roofEdge(peppers, bandwidth = 6, thresh = 3000,
    edge1 = step.edges)
set.seed(24)
# Time consuming
parSel <- threeStageParSel(image = peppers, edge1 = step.edges,
    edge2 = roof.edges, bandwidth = 4, nboot = 1, blur = TRUE)

## End(Not run)
```

Index