A Multiscale Test of Spatial Stationarity for Textured Images in R

by Matthew A. Nunes, Sarah L. Taylor and Idris A. Eckley

Abstract The ability to automatically identify areas of homogeneous texture present within a greyscale image is an important feature of image processing algorithms. This article describes the R package **LS2Wstat** which employs a recent wavelet-based test of stationarity for locally stationary random fields to assess such spatial homogeneity. By embedding this test within a quadtree image segmentation procedure we are also able to identify texture regions within an image.

Introduction

This paper provides an introduction to the **LS2Wstat** package (Taylor and Nunes, 2014), developed to implement recent statistical methodology for the analysis of (greyscale) textured images. Texture analysis is a branch of image processing concerned with studying the variation in an image surface; this variation describes the physical properties of an object of interest. The key applications in this field, namely discrimination, classification and segmentation, are often dependent on assumptions relating to the second-order structure (variance properties) of an image. In particular many techniques commonly assume that images possess the property of spatial stationarity (Gonzalez and Woods, 2001). However, for images arising in practice this assumption is often not realistic, i.e. typically the second-order structure of an image varies across location. It is thus important to test this assumption of stationarity before performing further image analysis. See Figure 1 for examples of textured images. For a comprehensive introduction to texture analysis, see Bishop and Nasrabadi (2006) or Petrou and Sevilla (2006).

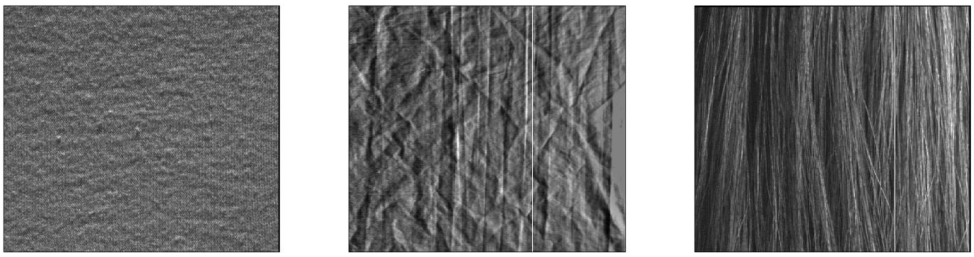


Figure 1: Examples of textured images: fabric, creased material and hair (available from the R package **LS2W**; Eckley and Nason, 2013).

Recently, Taylor et al. (in press) proposed a test of spatial stationarity founded on the *locally stationary two-dimensional wavelet* (LS2W) modelling approach of Eckley et al. (2010). The LS2W modelling approach provides a location-based decomposition of the spectral structure of an image. The $\text{Bootstat}_{\text{LS2W}}$ test proposed by Taylor et al. (in press) uses a statistic based on an estimate of the process variance within a hypothesis testing framework, employing bootstrap resampling under the null hypothesis assumption of stationarity to assess its significance.

Given a test of spatial stationarity for random fields, it is natural to consider how this might be usefully applied within a problem such as texture segmentation. The ability to determine non-stationarity and the presence of localised textured regions within images is important in a broad range of scientific and industrial applications, including product evaluation or quality control purposes. Possible areas of use for the methods described in this article include identifying uneven wear in fabrics (Chan and Pang, 2000; Taylor et al., in press) and defect detection on the surface of industrial components (Wiltschi et al., 2000; Bradley and Wong, 2001) or natural products (Funck et al., 2003; Pölzleitner, 2003). For a review of texture segmentation, see Pal and Pal (1993).

Readily available implementations for stationarity assessment have, up until now, been restricted to the time series setting; examples of such software in this context include the R add-on packages **urca** (Pfaff, 2008; Pfaff and Stigler, 2013), **CADFtest** (Lupi, 2009) and **locits** (Nason, 2013a,b).

Below we describe the package **LS2Wstat** which implements the spatial test of stationarity proposed by Taylor et al. (in press). The package has been developed in R and makes use of several functions within the **LS2W** package (Eckley and Nason, 2011, 2013). The article is structured as

follows. We begin by describing details of simulation of LS2W and other processes. An overview of the $\text{Bootstat}_{\text{LS2W}}$ test of stationarity is then given, focussing in particular on the function `TOS2D`. We then illustrate the application of the test on both simulated and real texture images. Finally, the article concludes by describing how the algorithm might be embedded within a quadtree image splitting procedure to identify regions of texture homogeneity within a multi-textured image.

Simulating textured images with LS2Wstat

Before describing the implementation of the work proposed in Taylor et al. (in press), we first explain how to generate candidate textured images using the **LS2Wstat** package. Several different spatially stationary and non-stationary random fields can be generated with the `simTexture` function. See the package help file for full details of the processes available.

To demonstrate the **LS2Wstat** implementation, throughout this article we consider a realisation of a white noise process with a subregion of random Normal deviates in its centre with a standard deviation of 1.6. This simulated texture type is called `NS5`, and is one of several textures which can be simulated from the package. In particular, we consider an image of dimension 512×512 with a central region that is a quarter of the image, i.e. a dimension size of 128×128. This can be obtained as follows:

```
> library("LS2Wstat")
> set.seed(1)
> X <- simTexture(n = 512, K = 1, imtype = "NS5", sd = 1.6, prop = 0.25)[[1]]
> image(plotmtx(X), col = grey(255:0/256))
```

The `simTexture` function returns a list of length `K` with each list entry being a matrix representing an image of dimension `n` $\times$ `n` with the chosen spectral structure. In this case, since `K = 1`, a list of length 1 is returned. The simulated image `X` is shown in Figure 2. Note in particular that visually, one can discern that the image consists of two subtly different texture types. Firstly, the centre of the image has one particular form of second order structure. The second texture structure can be seen in the remainder of the image. Throughout the rest of this article we shall apply the approach of Taylor et al. (in press) to this image.

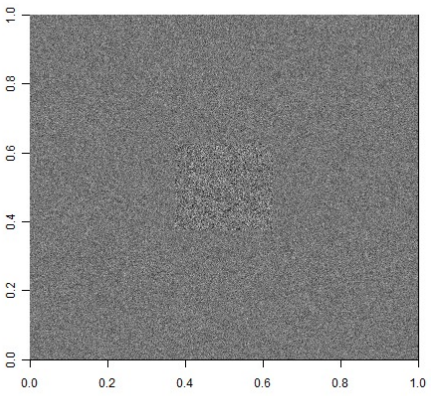


Figure 2: An example of a textured image (`NS5`) simulated with the `simTexture` function.

Testing the spatial stationarity of images

We now briefly introduce the LS2W random field model of Eckley et al. (2010) together with some associated notation, before describing the implementation of the test of stationarity proposed in Taylor et al. (in press). For an accessible introduction to wavelets, please see Prasad and Iyengar (1997), Vidakovic (1999) or Nason (2008).

The LS2W process model is defined by

$$X_{\boldsymbol{r}} = \sum_{l} \sum_{j=1}^{\infty} \sum_{\boldsymbol{u}} w_{j,\boldsymbol{u}}^{l} \psi_{j,\boldsymbol{u}}^{l}(\boldsymbol{r}) \xi_{j,\boldsymbol{u}}^{l}, \tag{1}$$

for directions $l = h, v$ or d and spatial locations $\boldsymbol{r}$, where $\{\xi_{j,\boldsymbol{u}}^{l}\}$ is a zero-mean random orthonormal increment sequence; $\{\psi_{j,\boldsymbol{u}}^{l}\}$ is a set of discrete nondecimated wavelets and $\{w_{j,\boldsymbol{u}}^{l}\}$ is a collection of

amplitudes, constrained to vary slowly over locations of an image (Eckley et al., 2010). In the above definition, we assume the image is of dyadic dimension, i.e. we have $\boldsymbol{r} = (r, s)$ with $r, s \in \{1, \ldots, 2^J\}$ and where J is the coarsest observed scale.

Eckley et al. (2010) also define the *local wavelet spectrum* (LWS) associated with an LS2W process. The LWS for a given location $\boldsymbol{z} = \left(\frac{r}{2^J}, \frac{s}{2^J}\right) \in (0,1)^2$, at scale j in direction l is $S_j^l(\boldsymbol{z}) \approx w_j^l(\boldsymbol{u}/\boldsymbol{R})^2$. The LWS provides a decomposition of the process variance at (rescaled) locations $\boldsymbol{z}$, directions l, and wavelet scales j. In practice the LWS is usually unknown and so needs to be estimated (see Eckley et al., 2010, for details). Spectral estimation using the LS2W model is implemented in R in the add-on package **LS2W** (Eckley and Nason, 2013). The **LS2Wstat** routines described below thus have a dependence on some functions from the **LS2W** package.

A test of stationarity for LS2W processes

Next we turn to describe the implementation of a test of stationarity within the **LS2Wstat** package. We focus on describing the $\text{Bootstat}_{\text{LS2W}}$ approach implemented in the **LS2Wstat** package, referring the interested reader to Taylor et al. (in press) for details of other tests which might be employed. Throughout this section let us assume that we have some image $X_{\mathbf{r}}$ (as in Figure 2), whose second-order structure we wish to test for spatial stationarity. We assume that X is an LS2W process with associated unknown spectrum, S_j^ℓ for $j = 1, \ldots, J$ and $\ell = v, h$ or d. Since the model in (1) assumes the process has zero mean, if necessary the image can be detrended. This can be done in R, for example, by using the core **stats** package function `medpolish`, which implements Tukey's median polish technique (Tukey, 1977).

Under the null hypothesis of stationarity, the wavelet spectrum will be constant across location for each scale and direction. Motivated by this fact Taylor et al. (in press) formulate a hypothesis test for the stationarity of the image $X_{\mathbf{r}}$ with

$$H_0 : S_j^\ell(\mathbf{z}) \text{ is constant across } \mathbf{z} \text{ for all } j \text{ and } \ell,$$
$$H_A : S_j^\ell(\mathbf{z}) \text{ is not constant across } \mathbf{z} \text{ for some } j \text{ or } \ell.$$

Hence, a test statistic for the hypothesis should measure how much the wavelet spectrum for an observed image differs from constancy. Taylor et al. (in press) suggest using the average scale-direction spectral variance as a test statistic to measure the degree of non-stationary within an image, where the variance is taken over pixel locations, that is:

$$T\left\{\hat{S}_j^\ell(\mathbf{z})\right\} = \frac{1}{3J}\sum_{\ell}\sum_{j=1}^{J} \text{var}_{\mathbf{u}}\left(\hat{S}_{j,\mathbf{u}}^\ell\right). \tag{2}$$

In practice this statistic is computed based on an (unbiased) estimate of the local wavelet spectrum, produced by the **LS2W** function `cddews` (see the documentation in **LS2W** for details on optional arguments to this function). For the (square) image `X`, the test statistic is calculated using the function `avespecvar` as follows:

```
> TSvalue <- avespecvar(cddews(X, smooth = FALSE))
> TSvalue
[1] 0.2044345
```

Since the spectrum characterises the second-order structure of the observed random field (and hence its stationarity properties), Taylor et al. (in press) suggest determining the p-value of the hypothesis test by performing parametric bootstrapping. This corresponds to sampling LS2W processes assuming stationarity under the null hypothesis, and comparing the observed test statistic to that of the simulated LS2W processes under stationarity. For pseudo-code of this algorithm, please see Algorithm 1.

This bootstrap algorithm is performed with the **LS2Wstat** function `TOS2D`. The function has arguments:

`image`: The image you want to analyse.

`detrend`: A binary value indicating whether the image should be detrended before applying the bootstrap test. If set to `TRUE`, the image is detrended using Tukey's median polish method.

`nsamples`: The number of bootstrap simulations to carry out. This is the value B in the pseudocode given above. By default this takes the value 100.

`theTS`: This specifies the test statistic function to be used within the testing procedure to measure non-stationarity. The test statistic should be based on the local wavelet spectrum and by default is the function `avespecvar` representing the statistic (2).

$\texttt{Bootstat}_{\texttt{LS2W}}$:

1. Compute the estimate of the LWS for the observed image, $\hat{S}_j^l(\mathbf{z})$.
2. Evaluate T (Equation (2)) on the observed image, call this value T^{obs}.
3. Compute the pixel average stationary spectrum $\widetilde{S}_j^l$ by taking the average of spectrum values for each scale and direction.
4. **Iterate** for i in 1 to B bootstraps:
   (a) Simulate $X_{\mathbf{r}}^{(i)}$ from the stationary LS2W model using squared amplitudes given by $\widetilde{S}_j^l$ and Gaussian process innovations.
   (b) Compute the test statistic T on the simulated realisation, call this value $T^{(i)}$.
5. Compute the p-value for the test as $p = \frac{1+\#\{T^{obs} \leq T^{(i)}\}}{B+1}$.

Algorithm 1: The bootstrap algorithm for testing the stationarity of locally stationary images.

`verbose`: A binary value indicating whether informative messages should be printed.

`...`: Any optional arguments to be passed to the **LS2W** function `cddews`. See the documentation for the `cddews` function for more details.

Note that `TOS2D` uses the **LS2W** process simulation function `LS2Wsim` from the **LS2W** R package to simulate bootstrap realizations under the null hypothesis. The output of `TOS2D` is a list object of class `"TOS2D"`, which describes pertinent quantities associated with the bootstrap test of stationarity. The object contains the following components:

`data.name`: The name of the image tested for stationarity.

`samples`: A vector of length `nsamples + 1` containing each of the test statistics calculated in the bootstrap test. The first element of the vector is the value of the test statistic calculated for the original image itself.

`statistic`: The statistic used in the test.

`p.value`: The bootstrap p-value associated with the test.

In particular, the object returns the measure of spectral constancy in the entry `statistic`, together with the p-value associated with the stationarity test (in the `p.value` component).

An example of the function call is

```
> Xbstest <- TOS2D(X, nsamples = 100)
```

Note that the p-value returned within the `"TOS2D"` object is computed using the utility function `getpval`, which returns the parametric bootstrap p-value for the test from the bootstrap test statistics provided by counting those test statistic values less than T^{obs} (see Davison et al., 1999, for more details). In other words, the `p.value` component is obtained by the following call:

```
> pval <- getpval(Xbstest$samples)
Observed bootstrap is  0.204
p-value is  0.00990099
```

This p-value can then be used to assess the stationarity of a textured image region.

Information on the `"TOS2D"` class object can be obtained using the `print` or `summary` S3 methods for this class. For example, using the summary method, one would obtain

```
> summary(Xbstest)

2D bootstrap test of stationarity
      object of class TOS2D
---------------------------------

summary
=======
data: X
Observed test statistic: 0.204
bootstrap p-value: 0.01
```

Alternatively, the `print` method for the `"TOS2D"` class prints more information about the `Xbstest` object. Note that the function internally calls the `summary` method for `"TOS2D"` objects:

```
2D bootstrap test of stationarity
      object of class TOS2D
---------------------------------

summary
=======
data: X
Observed test statistic: 0.204
bootstrap p-value: 0.01

Number of bootstrap realizations: 100
spectral statistic used: avespecvar
```

Other textured images

To demonstrate the test of stationarity further, we now provide some other textured image examples. Firstly, we consider a *Haar wavelet random field* with a diagonal texture, an example of a LS2W process as described in Eckley et al. (2010). The realisation of the process (shown in Figure 2) is simulated using the `simTexture` function with the command:

```
> Haarimage <- simTexture(512, K = 1, imtype = "S5")[[1]]
```

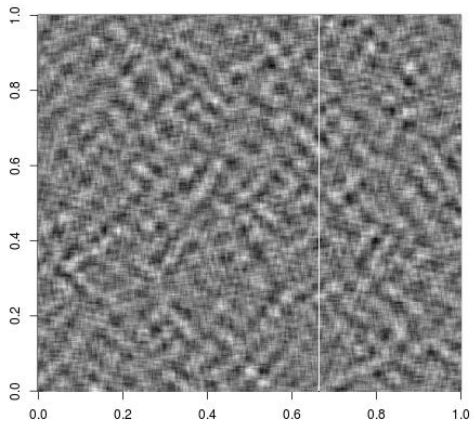


Figure 2: A realisation of a stationary LS2W process, `Haarimage`, with a diagonal texture.

The test of stationarity of Taylor et al. (in press) performed on the image `Haarimage` with the function `TOS2D` reveals that the image is spatially stationary as expected, with a high p-value associated to the test.

```
> Haarimtest <- TOS2D(Haarimage, smooth = FALSE, nsamples = 100)
> summary(Haarimtest)

2D bootstrap test of stationarity
      object of class TOS2D
---------------------------------

summary
=======
data: Haarimage
Observed test statistic: 0.631
bootstrap p-value: 0.673

Number of bootstrap realizations: 100
spectral statistic used: avespecvar
```

As another example of a textured image, we construct an image montage using two of the textures shown in Figure 1 from the package **LS2W**. The montage, `montage1`, is shown in Figure 3.

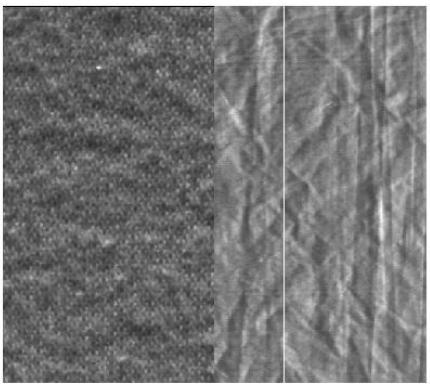


Figure 3: An example of an image montage, `montage1`, using two of the textures from Figure 1.

Note that since this image may not have zero mean as assumed by the LS2W model (1), we detrend the montage first using the `medpolish` function in the **stats** package.

```
> data(textures)
> montage1 <- cbind(A[1:512, 1:256], B[, 1:256])
> montage1zm <- medpolish(montage1)$residuals
```

The `TOS2D` test indicates that the texture montage is non-stationary:

```
> montage1zmtest <- TOS2D(montage1zm, smooth = FALSE, nsamples = 100)
> summary(montage1zmtest)

2D bootstrap test of stationarity
      object of class TOS2D
---------------------------------

summary
=======
data: montage1zm
Observed test statistic: 0
bootstrap p-value: 0.01

Number of bootstrap realizations: 100
spectral statistic used: avespecvar
```

Identifying areas of homogeneous texture using the bootstrap test of stationarity

In this section we describe embedding a test of stationarity into a quadtree algorithm to identify regions of spatial homogeneity within a textured image. This segmentation approach is similar in spirit to, e.g., Spann and Wilson (1985) or Pal and Pal (1987) which use homogeneity measures within a quadtree structure. We first give details of the quadtree implementation, and subsequently describe functions to aid illustration of quadtree decompositions.

A quadtree algorithm implementation

In essence, a region splitting algorithm recursively subdivides an input image into smaller regions, with the subdivision decisions being based on some statistical criterion. More specifically, in a *quadtree representation*, at each stage a (sub)image is divided into its four subquadrants if the criterion is not satisfied (see e.g., Sonka et al., 1999). The statistical criterion we use is spatial homogeneity, that is, a quadrant is further divided if it is considered as non-stationary by the $\text{Bootstat}_{\text{LS2W}}$ test. In practice, the quadtree implementation in **LS2Wstat** continues until all subregions are considered as stationary, or until the subregions reach a particular minimal dimension. The motivation for this is to ensure that we obtain statistically meaningful decisions using the stationarity test by not allowing too small a

`Quadtree decomposition:`

For an input image X:

Use the $\text{Bootstat}_{\text{LS2W}}$ test to assess whether X is second-order stationary. If stationary, stop. If not,

1. Divide the image into four quadrants.
2. For each quadrant, assess its stationarity with the $\text{Bootstat}_{\text{LS2W}}$ test.
3. For each quadrant assessed as non-stationary, recursively repeat steps 1–2, until the minimum testing region is reached or until all sub-images are judged to be stationary.

Algorithm 2: The quadtree algorithm for segmenting an image into regions of spatial stationarity.

testing sub-image. This procedure segments an image into regions of spatial stationarity. The quadtree algorithm is summarised in Algorithm 2.

Each image is further split if deemed non-stationary, which is determined by a test of stationarity such as `TOS2D`. After the first subdivison of an image, each sub-image is of size $n/2 \times n/2$. The sizes of the regions halve in size at each progressive division but increase in number. The R function in **LS2Wstat** which creates the quadtree structure described in Algorithm 2 is `imageQT`. The function has inputs:

`image`: The image to be decomposed with the quadtree algorithm.

`test`: A function for assessing regions of spatial homogeneity, for example `TOS2D`.

`minsize`: The testing size of sub-images below which we should not apply the function `test`.

`alpha`: The significance level of the $\text{Bootstat}_{\text{LS2W}}$ test, with which to assess spatial stationarity of textured regions.

`...`: Any other optional arguments to `test`.

As an illustration of using the `imageQT` function, consider the code below to decompose the (non-stationary) input image `X`. We use the function `TOS2D` to assess the regions of spatial homogeneity although the `imageQT` function allows other functions to be supplied.

```
> QTdecX <- imageQT(X, test = TOS2D, nsamples = 100)
```

The output of the `imageQT` function is a list object of class `"imageQT"` with components:

`indl`: The index representation of the non-stationary images in the quadtree decomposition.

`resl`: The results of the stationarity testing (from the `test` argument) during the quadtree decomposition. The results giving `FALSE` correspond to those non-stationary sub-images contained in the `indl` component and the results giving `TRUE` correspond to the stationary sub-images, i.e. those contained in the `indS` component.

`indS`: The index representation of the stationary images in the quadtree decomposition.

This particular way of splitting an image has a convenient indexing representation to identify the position of subregions within an image. If a (sub)image is subdivided into quadrants, we assign it a base 4 label as follows: 0 – top-left quadrant; 1 – bottom-left quadrant; 2 – top-right quadrant; 3 – bottom-right quadrant. By continuing in this manner, we can assign an index to each tested subregion, with the number of digits in the index indicating how many times its parent images have been subdivided from the "root" of the tree (the original image). This indexing is illustrated for the quadtree decomposition given in the example in Figure 3.

Examining the quadtree decomposition of the image `X` using the `print` S3 method for the `"imageQT"` class, we have

```
> print(QTdecX)

2D quadtree decomposition
 object of class imageQT
------------------------

summary
=======
data: X
```

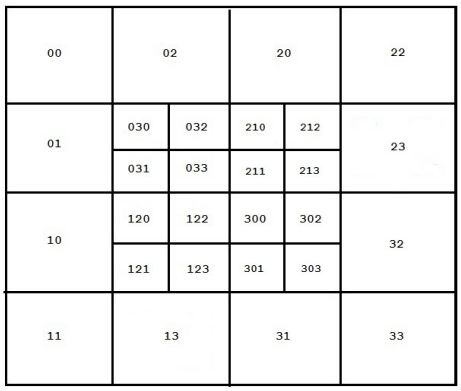


Figure 3: An example of a quadtree decomposition. The location of the sub-images in the decomposition are described by the indexing system described in the text.

```
Indices of non-stationary sub-images:
"0" "1" "2" "3" "03" "12" "21" "30"

Indices of stationary sub-images:
"00" "01" "02" "10" "11" "13" "20" "22" "23" "31" "32" "33" "030" "031" "032" "033"
"120" "121" "122" "123" "210" "211" "212" "213" "300" "301" "302" "303"

minimum testing region: 64
```

The `resl` component gives the results of the test of stationarity for all sub-images tested during the quadtree procedure, reporting `FALSE` for the non-stationary sub-images and `TRUE` for the stationary ones:

```
> QTdecX$resl
[1] FALSE
[[2]]
[1] FALSE FALSE FALSE FALSE
[[3]]
 [1] TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE
[13] FALSE  TRUE  TRUE  TRUE
[[4]]
 [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
[16] TRUE
```

Plotting a quadtree decomposition

By performing the quadtree algorithm given in Algorithm 2, it is possible to decompose images into regions indicating regional stationarity. Note that if a texture discrimination procedure is used to classify the output from the stationarity quadtree algorithm, the image segmentation method can be seen as a *split and merge* technique.

Suppose we have performed the quadtree decomposition. The **LS2Wstat** package includes an S3 `plot` method for `"imageQT"` objects to plot the output for the `"imageQT"` class and optionally a classification of those textured regions. If the classification output is plotted (`class = TRUE`), each textured region is uniquely coloured according to its texture group. The function has arguments:

`x`: A quadtree decomposition object, such as output from `imageQT`.

`cires`: Vector of class labels associated to the subimages produced by the quadtree decomposition.

`unclassval`: A value for any unclassified values in a quadtree decomposition.

`class`: A Boolean value indicating whether to plot the classification of the quadtree subimages.

`QT`: A Boolean value indicating whether to plot the quadtree decomposition.

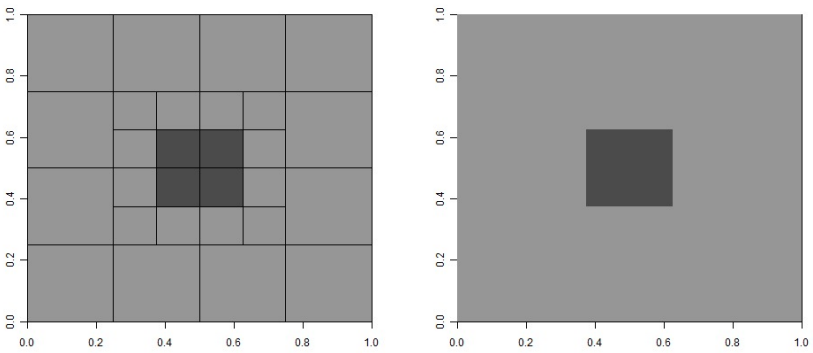
We now illustrate the use of this function with the example given in Figure 2. Suppose the textured regions identified by the quadtree algorithm in the `QTdecX` object have been classified according to some texture discrimination procedure. For the purposes of this example, we suppose that the 28 regions of stationarity in `QTdecX` (see Figure 3) have been classified as coming from two groups according to the labels

```
> texclass <- c(rep(1, times = 15), rep(c(2, 1, 1), times = 4), 1)
> texclass
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 1 2 1 1 2 1 1 1
```

Using the output from the quadtree technique (`QTdecX`) and the texture classification vector `texclass`, we can use the quadtree plotting function for `"imageQT"` objects as follows:

```
> plot(QTdecX, texclass, class = TRUE, QT = TRUE)
> plot(QTdecX, texclass, class = TRUE, QT = FALSE)
```

The quadtree decomposition from this example is shown in Figure 4a; the same decomposition is shown together with the texture classification in Figure 4b.



(a) **(b)**

Figure 4: An example of a quad-tree decomposition using `imageQT`, together with an assumed sub-image texture classification.

We also consider an image montage using the textures from the package **LS2W**. The montage `Y` is shown in Figure 5. Prior to performing the quadtree decomposition, we detrend the image.

```
> data(textures)
> Y <- cbind(A[1:512, 1:256], rbind(B[1:256, 1:256], C[1:256, 1:256]))
> Yzm <- medpolish(Y)$residuals
```

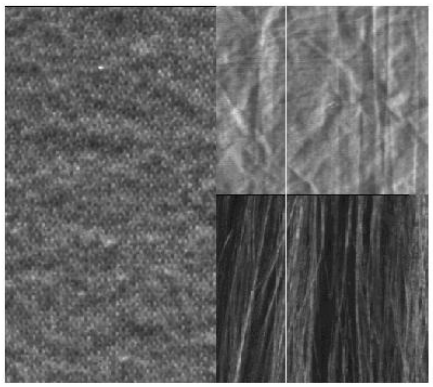


Figure 5: An example of an image montage, `Y`, using the textures from Figure 1.

Similarly to above, we can now perform a quadtree decomposition of the image `Y`:

```
> QTdecYzm <- imageQT(Yzm, test = TOS2D, nsamples = 100)
> print(QTdecYzm)

2D quadtree decomposition
 object of class imageQT
-------------------------
```

```
summary
=======
data: Yzm
Indices of non-stationary sub-images:

Indices of stationary sub-images:
"0" "1" "2" "3"

minimum testing region: 64
```

The function `imageQT` initially assesses that the image is indeed non-stationary, and then proceeds to analyse sub-images of the montage. The algorithm stops the quadtree decomposition after the first decomposition level, since it judges all quadrants of the image to be stationary, described by the indices "0", "1", "2", and "3".

Summary

In this article we have described the **LS2Wstat** package, which implements some recent methodology for image stationarity testing (Taylor et al., in press). Our algorithm is most useful as a test of homogeneity in textures which are visually difficult to assess. We have also extended its potential use by embedding it within a quadtree implementation, allowing assessment of potentially multi-textured images. The implementation is demonstrated using simulated and real textures throughout the paper.

Acknowledgements

We thank Aimée Gott for suggestions on an early version of the package. We would also like to thank Heather Turner, two anonymous referees and the Editor for helpful comments which have resulted in an improved manuscript and package.

Bibliography

C. M. Bishop and N. M. Nasrabadi. *Pattern Recognition and Machine Learning*, volume 1. Springer-Verlag, New York, 2006. [p20]

C. Bradley and Y. S. Wong. Surface texture indicators of tool wear – a machine vision approach. *The International Journal of Advanced Manufacturing Technology*, 17(6):435–443, 2001. [p20]

C. Chan and G. K. H. Pang. Fabric defect detection by Fourier analysis. *IEEE Transactions on Industry Applications*, 36(5):1267–1276, 2000. [p20]

A. Davison, D. Hinkley, and A. J. Canty. *Bootstrap Methods and their Application*. Cambridge University Press, 1999. [p23]

I. A. Eckley and G. P. Nason. LS2W: Locally stationary wavelet fields in R. *Journal of Statistical Software*, 43(3):1–23, 2011. URL http://www.jstatsoft.org/v43/i03. [p20]

I. A. Eckley and G. P. Nason. *LS2W: Locally Stationary Two-Dimensional Wavelet Process Estimation Scheme*, 2013. URL http://CRAN.R-project.org/package=LS2W. R package version 1.3-3. [p20, 22]

I. A. Eckley, G. P. Nason, and R. L. Treloar. Locally stationary wavelet fields with application to the modelling and analysis of image texture. *Journal of the Royal Statistical Society C*, 59(4):595–616, 2010. [p20, 21, 22, 24]

J. W. Funck, Y. Zhong, D. A. Butler, C. C. Brunner, and J. B. Forrer. Image segmentation algorithms applied to wood defect detection. *Computers and Electronics in Agriculture*, 41(1):157–179, 2003. [p20]

R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice Hall, 2nd edition, 2001. [p20]

C. Lupi. Unit root CADF testing with R. *Journal of Statistical Software*, 32(2):1–19, 2009. URL http://www.jstatsoft.org/v32/i02. [p20]

G. P. Nason. *Wavelet Methods in Statistics with R*. Springer-Verlag, 2008. [p21]

G. P. Nason. *locits: Test of Stationarity and Localized Autocovariance*, 2013a. URL http://CRAN.R-project.org/package=locits. R package version 1.4. [p20]

G. P. Nason. A test for second-order stationarity and approximate confidence intervals for localized autocovariances for locally stationary time series. *Journal of the Royal Statistical Society B*, 75(5): 879–904, 2013b. [p20]

N. R. Pal and S. K. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9): 1277–1294, 1993. [p20]

S. K. Pal and N. R. Pal. Segmentation using contrast and homogeneity measures. *Pattern Recognition*, 5 (4):293–304, 1987. [p25]

M. Petrou and P. G. Sevilla. *Image Processing: Dealing with Texture*. John Wiley & Sons, 2006. [p20]

B. Pfaff. *Analysis of Integrated and Cointegrated Time Series with R*. Springer-Verlag, New York, 2nd edition, 2008. [p20]

B. Pfaff and M. Stigler. *urca: Unit Root and Cointegration Tests for Time Series Data*, 2013. URL http://CRAN.R-project.org/package=urca. R package version 1.2-8. [p20]

W. Pölzleitner. Quality classification of wooden surfaces using Gabor filters and genetic feature optimisation. In *Machine Vision for the Inspection of Natural Products*, pages 259–277. Springer-Verlag, 2003. [p20]

L. Prasad and S. S. Iyengar. *Wavelet Analysis with Applications to Image Processing*. CRC Press, 1997. [p21]

M. Sonka, R. Boyle, and V. Hlavac. *Image Processing, Analysis, and Machine Vision*. PWS Publishing, 2nd edition, 1999. [p25]

M. Spann and R. Wilson. A quad-tree approach to image segmentation which combines statistical and spatial information. *Pattern Recognition*, 18(3/4):257–269, 1985. [p25]

S. Taylor and M. A. Nunes. *LS2Wstat: A Multiscale Test of Spatial Stationarity for LS2W Processes*, 2014. URL http://CRAN.R-project.org/package=LS2Wstat. R package version 2.0-3. [p20]

S. L. Taylor, I. A. Eckley, and M. A. Nunes. A test of stationarity for textured images. *Technometrics*, in press. doi: 10.1080/00401706.2013.823890. [p20, 21, 22, 24, 29]

J. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977. [p22]

B. Vidakovic. *Statistical Modelling by Wavelets*. John Wiley & Sons, New York, 1999. [p21]

K. Wiltschi, A. Pinz, and T. Lindeberg. An automatic assessment scheme for steel quality inspection. *Machine Vision and Applications*, 12(3):113–128, 2000. [p20]

Matthew A. Nunes
Lancaster University
Lancaster
UK
m.nunes@lancaster.ac.uk

Sarah L. Taylor
Lancaster University
Lancaster
UK
tsarah8624@yahoo.co.uk

Idris A. Eckley
Lancaster University
Lancaster
UK
i.eckley@lancaster.ac.uk