




kcpRS: An R Package for performing kernel change point detection on the running statistics of multivariate time series

Jedelyn Cabrieto¹ · Kristof Meers¹ · Evelien Schat¹ · Janne Adolf¹  · Peter Kuppens¹ · Francis Tuerlinckx¹ · Eva Ceulemans¹

Accepted: 22 April 2021 / Published online: 24 September 2021
© The Psychonomic Society, Inc. 2021

Abstract

In many scientific disciplines, researchers are interested in discovering when complex systems such as stock markets, the weather or the human body display abrupt changes. Essentially, this often comes down to detecting whether a multivariate time series contains abrupt changes in one or more statistics, such as means, variances or pairwise correlations. To assist researchers in this endeavor, this paper presents the **kcpRS** package for performing kernel change point (KCP) detection on user-selected running statistics of multivariate time series. The running statistics are extracted by sliding a window across the time series and computing the value of the statistic(s) of interest in each window. Next, the similarities of the running values are assessed using a Gaussian kernel, and change points that segment the time series into maximally homogeneous phases are located by minimizing a within-phase variance criterion. To decide on the number of change points, a combination of a permutation-based significance test and a grid search is provided. **kcpRS** stands out among the variety of change point detection packages available in **R** because it can be easily adapted to uncover changes in any user-selected statistic without imposing any distribution on the data. To exhibit the usefulness of the package, two empirical examples are provided pertaining to two types of physiological data.

Keywords change point · running statistics · kernel · permutation test

Introduction

Significant psychological events are often characterized by abrupt changes in statistical features. This is evidenced by sudden mean level, variance, correlation and/or autocorrelation shifts that researchers notice in the time series they monitor while studying such events. For instance, at the onset of physically (De Roover et al., 2014), mentally (Barnett & Onnela, 2016; Grassmann et al., 2016) or emotionally (Bulteel et al., 2014) demanding events, human physiology exhibits such sudden changes. Also nonhuman complex systems can show similar abrupt changes in reaction to critical events, such as computer network attacks (Tartakovsky et al., 2006), geological disasters such as volcanic eruptions (Tárraga et al., 2014), epidemic breakouts (Texier et al., 2016) and financial crises (Galeano & Wied, 2017).

Change point detection methods were developed to retrospectively capture such sudden changes in statistical features (Brodsky & Darkhovsky, 1993; Chen & Gupta, 2012). These methods are employed to address two crucial questions (Bulteel et al., 2014; Cabrieto et al., 2019; De Roover et al., 2014; Galeano & Wied, 2017) that researchers have. The first question pertains to the exact timing of the changes and how proximal they are to the onset of an event if this is known. The second question focuses on which types of change transpired. While in some applications, researchers already know which type of change is expected (i.e., mean, variance, correlation), making it straightforward to specify which statistic to track, in many other cases, no prior information is available and several statistics have to be screened for changes.

A number of methods and software packages have been proposed to aid researchers in capturing and characterizing unknown changes in time series data. However, they often fall short in practice because they are restricted to univariate data (Erdman & Emerson, 2007; Killick & Eckley, 2014; Ross, 2015), restricted to a specific statistic or statistical model (Barnett & Onnela, 2016; Dürre et al., 2015; Galeano & Wied, 2017; Zeileis et al., 2002) or impose other stringent assumptions on the data (Chen & Gupta, 2012; Davis et al.,

✉ Eva Ceulemans
eva.ceulemans@kuleuven.be

¹ Research Group of Quantitative Psychology and Individual Differences, KU Leuven - University of Leuven, Tiensestraat 102, Box 3713, 3000 Leuven, Belgium

2006; Lavielle, 2005). There are multivariate, nonparametric methods available (Arlot et al., 2012; Bulteel et al., 2014; Lung-Yut-Fong et al., 2012; Matteson & James, 2014) that can potentially detect all types of changes, but they often lack power to detect changes in statistics that do not pertain to the mean level (e.g., correlation; Cabrieto, Adolf, Tuerlinckx, Kuppens, & Ceulemans, 2018a) and do not uncover which statistics are exactly involved in the change. To bridge this gap, we present **kcpRS**, a flexible software package that implements the multivariate, nonparametric KCP-RS analysis (Cabrieto, Tuerlinckx, Kuppens, Wilhelm, et al., 2018c) that screens user-specified running statistics (RS) for changes. Because the tool screens the running statistics rather than the raw data, once a change point is flagged, the user knows what type of statistical change occurred (e.g., a mean or a variance change) and is therefore able to better understand the mechanisms that underlie an event.

KCP-RS is based on the generic kernel change point (KCP) approach proposed by Arlot et al. (2012) for flagging changes in the raw data. The key steps of KCP-RS are to first convert the original time series to running statistics, reflecting the fluctuations in the statistics of interest. This is carried out by sliding a window one time point at a time across the time series, and each time computing the statistics' values. Next, KCP is implemented on the running statistics to search for candidate change point locations, dividing the time series into distinct homogeneous time phases. Note that we prefer the term 'phase' over the term 'regime' to indicate that all time points within a phase are adjacent (for a comparison of KCP-RS to regime switching approaches, see Cabrieto, Adolf, Tuerlinckx, Kuppens, & Ceulemans, 2018a). Finally, the presence and the number of change points are determined via a permutation test and a grid search. We also discuss how to streamline more complex settings where researchers aim to screen (a subset of) four popular running statistics simultaneously. The performance and power of KCP-RS was thoroughly checked in multiple simulation studies (Cabrieto, Adolf, Tuerlinckx, Kuppens, & Ceulemans, 2018a; Cabrieto, Tuerlinckx, Kuppens, Wilhelm, et al., 2018c) and their usefulness was validated through analysis of several existing data sets containing known change points (Cabrieto et al., 2019).

The **kcpRS** package features two main functions. First, the **kcpRS** function is provided, allowing the user to flag change points in any statistic of interest. The user only has to supply a function that derives the running statistic of choice. Currently, four built-in functions are available for four popular statistics: mean, variance, autocorrelation and correlation. Second, the package includes the **kcpRS_workflow** function that further assists users in simultaneously screening (a subset of) the four statistics mentioned. Although the results of the **kcpRS** and **kcpRS_workflow** will almost always lead to the same conclusions, combining them can help to further avoid false

positive and false negative change points. On top of the change point analysis functions, visualization tools are available to generate graphs of the running statistics and the obtained change points.

The next sections in this paper are structured as follows. In Section 2, we present the steps of the KCP-RS analysis to flag changes in a single statistic of interest and provide recommendations for using KCP-RS. In Section 3, we discuss how to monitor multiple statistics. Both sections include toy examples with step-by-step illustrations of how to use the package. At the end, these sections conclude with real data examples on mental load data and CO₂-inhalation data.

KCP on the running statistics (KCP-RS)

In this section we first explain the steps of the KCP-RS analysis, followed by recommendations on how to optimize data characteristics and on how to set the KCP-RS analysis parameters. Finally, we illustrate how the analysis can be run using the **kcpRS** package.

The different steps of KCP-RS analysis and their statistical rationale

Step 1: Deriving the running statistics

The main goal of the KCP-RS analysis is to detect changes in a specific statistic. Thus, the first analysis step boils down to using a multivariate time series of observed data, \mathbf{X}_t ($t = 1, 2, \dots, n$), to generate a multivariate time series of running statistics, \mathbf{RS}_i ($i = 1, 2, \dots, w$), that reflects the fluctuations in the statistic of interest. Subscript t thereby denotes time running in unit steps from 1 to the total number of time points n . For each time point t , \mathbf{X}_t is thus a $v \times 1$ vector of scores on v variables observed at that time point. Hence, for the first time point, $t = 1$, we have vector $\mathbf{X}'_1 = [X_{11} \ X_{12} \ \dots \ X_{1v}]$, for the second time point, $t = 2$, we have vector $\mathbf{X}'_2 = [X_{21} \ X_{22} \ \dots \ X_{2v}]$ and so forth. The running statistics are then obtained by sliding a window of a selected size, $wsize$, one time point at a time, across the original time series, \mathbf{X}_t . In each window, we compute the statistic under study, yielding the derived time series of running statistics \mathbf{RS}_i . Subscript i thus denotes the window in which the running statistics are calculated and runs in unit steps from 1 to the total number of windows w . For each window i , \mathbf{RS}_i is a $d \times 1$ vector of running statistics. The number of running statistics d equals the number of original variables v when one calculates a univariate type of statistic such as the mean, the variance or the autocorrelation. However, when one tracks a statistic that involves multiple variables, then d can be larger or smaller than v . An example of the former is given by all pairwise correlations, with $d = \frac{v(v-1)}{2}$, whereas the intra-class

correlation of all the variables is an instance of the latter, with $d = 1$, since one obtains a single value (Erbaş et al., 2018). Also note that the total number of windows w will be smaller than the total number of time points n , because a window encompasses multiple adjacent time points (i.e., $wsize$ is larger than one).

Step 2: Locating the change points

The kernel function to assess the similarity of running statistics To locate the change points and divide the derived time series into distinct phases, KCP computes a kernel-based pairwise similarity between each pair of running statistic vectors RS_i and RS_j . The idea is straightforward: running statistics that belong to the same phase will have high similarities, while those that are separated by change points will have low similarities. As proposed by Arlot et al. (2012), any positive semi-definite kernel (Shawe-Taylor & Cristianini, 2004) can be plugged into KCP. In our previous studies (Cabrieto, Adolf, Tuerlinckx, Kuppens, & Ceulemans, 2018a; Cabrieto, Tuerlinckx, Kuppens, Hunyadi, & Ceulemans, 2018b; Cabrieto, Tuerlinckx, Kuppens, Wilhelm, et al., 2018c), we have shown that the Gaussian kernel works well in detecting change points in several running statistics (e.g., mean, variance, correlations and autocorrelations). Thus, in this package, we also use this Gaussian kernel function denoted by $Gk(\cdot)$ to measure the similarity of the running statistic vectors RS_i and RS_j :

$$Gk(RS_i, RS_j) = \exp\left(-\frac{\|RS_i - RS_j\|^2}{2h_{RS}^2}\right), \quad (1)$$

where h_{RS} is the median of the Euclidean distances of all pairs of running statistics. If RS_i and RS_j are extremely dissimilar, the kernel-based similarity approaches 0. If they are very similar, the similarity will be close to 1.

The variance criterion to optimize the change point locations

Based on the computed similarities, the KCP algorithm finds the most optimal change point locations given a specified number of change points K and thus delineates the $K+1$ phases. These locations are found by minimizing the within-phase scatter over all possible change point location patterns and associated phases. For a given change point location pattern and a specific phase m , which starts after the $m-1$ -th change point occurred, the within-phase scatter is calculated as

$$\hat{V}_{\tau_1, \tau_2, \dots, \tau_K, m} = (\tau_m - \tau_{m-1})^{-1} \times \sum_{i=(\tau_{m-1}+1)}^{\tau_m} \sum_{j=(\tau_{m-1}+1)}^{\tau_m} Gk(RS_i, RS_j). \quad (2)$$

Here, τ_m and τ_{m-1} represent the boundaries, more specifically the last window of phase m and the previous phase $m-1$. These boundaries are part of the full set of boundaries, $\tau_1, \tau_2, \dots, \tau_K$, that are associated with a considered change point location pattern. Note that for the last (i.e., $K+1$ -th) phase, the boundary τ_{K+1} equals the total number of windows $= w$. Given a specific change point location pattern the within-phase scatter for phase m thus boils down to calculating the average kernel-based pairwise similarity between all running statistic vectors RS_i and RS_j within phase m , and subtracting this average from the number of time points in phase m . The more homogeneous a phase m , the larger the average similarity and the smaller the corresponding within-phase scatter $\hat{V}_{\tau_1, \tau_2, \dots, \tau_K, m}$. Thus, to find the optimal change point location pattern, featuring the optimal boundaries $\hat{\tau}_1, \hat{\tau}_2, \dots, \hat{\tau}_K$, KCP minimizes the variance criterion $\hat{R}(\tau_1, \tau_2, \dots, \tau_K)$ over all possible change point location patterns:

$$\hat{R}(\tau_1, \tau_2, \dots, \tau_K) = \frac{1}{w} \sum_{m=1}^{K+1} \hat{V}_{\tau_1, \tau_2, \dots, \tau_K, m}. \quad (3)$$

This variance criterion sums the within-phase scatters over all phases and divides it by the total number of windows, w , a division due to which the variance criterion takes on values between 0 and 1. The optimal change point location pattern is thus given by

$$\hat{\tau}_1, \hat{\tau}_2, \dots, \hat{\tau}_K = \arg \min \hat{R}(\tau_1, \tau_2, \dots, \tau_m, \dots, \tau_K). \quad (4)$$

To illustrate, in the simplest scenario where only one change point is requested, all possible phase boundaries $\tau_1 \in 1, 2, \dots, w-1$ are tested and the optimal value $\hat{\tau}_1$ will be determined by the $\hat{\tau}_1$ that yields the minimal $\hat{R}(\tau_1) = \frac{1}{w} (\hat{V}_{\tau_1, 1} + \hat{V}_{\tau_1, 2})$. To simplify the notation for the following steps, we will denote the minimized variance criterion for a given K as $\hat{R}_{min,K}$. Note that $\hat{R}_{min,K}$ decreases with increasing values of K . Furthermore, in line with the previous papers on the KCP method, we set the final change point locations by shifting the optimal boundary values one window forward, hence $\hat{\tau}_1 + 1, \hat{\tau}_2 + 1, \dots, \hat{\tau}_K + 1$. This way, the change point locations coincide with the start of the next phase, yielding a more intuitive interpretation.

Once the change point location based on the running statistics is known, it is converted back to the original time scale. Specifically, the change point location is set to the time point t that constitutes the midpoint of the window in which the change point was located (or to the time point before the midpoint if the window size is even; Cabrieto, Tuerlinckx, Kuppens, Wilhelm, et al., 2018c). This conversion implies that the estimated change point location comes with some uncertainty, as any other time point in this window interval could also be the change point (i.e., the interval that runs from

the first to the last time point of the window).¹ We therefore consider an estimated change point to be proximal to the true or expected one if the estimated change point is located less than half of the window size from the true or expected one.

Step 3: Choosing the number of change points

In practice, the user often does not know how many change points are expected; thus K should also be inferred from the data. To this end, the user chooses the maximum number of change points, K_{max} (see Arlot et al., 2012 for more details) that have to be considered, based on what is reasonable for the data at hand. Arlot et al. (2012) proposed estimating $K \in \{1, 2, 3, \dots, K_{max}\}$ by balancing the fit (value of the variance criterion, $\widehat{R}_{min,K}$) and complexity (number of change points, K) of the change point solution by penalizing the variance criterion as follows:

$$\widehat{K} = \arg \min \widehat{R}(\tau_1, \tau_2, \tau_3, \dots, \tau_K) + C \frac{V_{max}(K+1)}{w} \left[1 + \log\left(\frac{w}{K+1}\right) \right] \quad (5)$$

where C is a penalty coefficient that weighs the influence of the penalty term and V_{max} is set to the maximum value of the trace of the covariance matrix of the running statistics in the first and last 5% windows (Arlot et al., 2012).

When applied to the running statistics, this approach has two limitations however. First, since Arlot et al. (2012) assumed independent observations, the type 1 error rate is excessively high when this approach is implemented on the running statistics, which are serially dependent because they are extracted from overlapping windows. Second, the user has to tune the penalty coefficient C . We therefore proposed to determine the number of change points via significance testing and a grid search, which are described in detail below.

Step 3a: The permutation test to establish whether there is at least one change point To ensure that the type 1 error rate is under control, we first conduct a significance test to decide whether the running statistics contain at least one change point (Cabrieto, Tuerlinckx, Kuppens, Hunyadi, & Ceulemans, 2018b). The proposed significance test is permutation based in that it compares the obtained results, based on the original data, with a distribution of reference results that one gets for a large number of permutations of the original data. The purpose of permuting the data is to simulate the test's sampling

¹ Note that the above interval and its interpretation should thus be distinguished from a confidence interval in classical statistical inference, in that it is not based on sampling distributions.

distribution under the null hypothesis of no change points. In the present case, the permutations are versions of the original time series for which the time indices, and thus the original time ordering of the vectors X_t , have been reshuffled.² Two such tests have been proposed: the variance and the variance drop test. Combining these tests optimizes power for finding correlation changes (Cabrieto, Tuerlinckx, Kuppens, Hunyadi, & Ceulemans, 2018b). However, for data that contain a background autocorrelation larger than zero, the variance test may yield an inflated type 1 error rate for running statistics influenced by such serial dependency (e.g., running autocorrelation (Cabrieto, Adolf, Tuerlinckx, Kuppens, & Ceulemans, 2018a), running variance, running means). We therefore focus on the variance drop test here.

The variance drop test focuses on the amount of improvement in the variance criterion when allowing for an additional change point: $\widehat{R}_{min,K} - \widehat{R}_{min,K-1}$. The test exploits the idea that if there is at least one change point present in the running statistics, this variance drop measure will be large for at least one K -value. For the time permuted data sets, we expect these variance drop measures to be significantly smaller. We therefore compute the p -value for the variance drop test as follows:

$$P_{\text{variance drop test}} = \frac{\#(\text{maxvariance drop}_p > \text{maxvariance drop})}{n_{\text{perm}}} \quad (6)$$

where n_{perm} is the total number of permuted data sets. As in the usual significance testing framework, the variance drop test is declared significant if the resulting p -value is smaller than the significance level set by the user.

Step 3b: The grid search to choose the exact number of change points (conditional on a yes in Step 3a) If the significance test discussed above suggests that the running statistics contain at least one change point, the next step is to assess the exact number of change points K . Building on the penalization strategy proposed by Arlot et al. (2012) and the work of Lavielle (2005), we proposed to tune the penalty coefficient in Eq. 5 via a grid search (Cabrieto et al., 2017; Cabrieto, Tuerlinckx, Kuppens, Wilhelm, et al., 2018c). This search starts by setting $C = 1$, which corresponds to selecting K_{max} change points. Next, we linearly increase C until the penalty term becomes so large that we retain zero change points. Each considered C -value corresponds to a K -value. The most stable K -value, that is the K -value that is most frequently returned

² We also considered other ways of permuting the data. Specifically, we also tried permuting the time indices of the running statistics and permuting blocks of indices of adjacent time points, based on recent results (Bodner et al., in press; Bulteel et al., 2018; Moulder et al., 2018). However, both approaches yielded inflated type 1 error rates and were therefore discarded.

across all C 's considered, is the chosen K . The final change point solution is then given by the optimal change point locations for this selected K .

Recommendations

Data characteristics: number of time points and number of variables

A natural first question is how many time points are needed for applying KCP-RS. The answer should be formulated in terms of the phase length. Obviously, the length of the phases should not be shorter than the window size used, as the changes will otherwise not be detected. However, it is better to have clearly longer phase lengths. In KCP-RS simulation studies, for instance, the phase length was often 100 time points and never less than 50 time points. Moreover, the empirical examples also had relatively long phase lengths. Hence, for now, we recommend aiming for 100 time points per phase.

Furthermore, we advise users to think carefully about which variables to include. A larger number of variables showing a change makes it easier for KCP-RS to detect this change. However, the presence of noise variables (i.e., variables that do not change) makes it harder for KCP-RS to detect changes.

KCP-RS analysis

The user has to set the number of permutations, the maximum number of change points, the window size, the significance tests used, the significance level of these tests and the number of CPU cores used.

The *number of permutations* to test the presence of at least one change point (see Step 3a) should be large (i.e., at least 1000) in order to adequately approximate the null distribution of the variance drop test.

Regarding the *maximum number of change points*, we recommend that the user set K_{max} to 10, to allow for unexpected change points.

Furthermore, for the *window size*, we set the default value to 25 time points since this value proved to be optimal in our simulation studies (Cabrieto, Tuerlinckx, Kuppens, Wilhelm, et al., 2018c). However, the user can adapt this choice based on the characteristics of the event under study and check the stability of the results for different window sizes. Specifically, we recommend first trying to determine a window size by dividing the expected minimum length of the phase by two. Next, one assesses the robustness of the obtained change point(s) by running the analysis with a number of different window sizes around this computed window size. If one obtains similar change points with this range of possible window sizes, one can obviously be more confident about the presence

and location of the change points. Since simulation studies (Cabrieto, Tuerlinckx, Kuppens, Hunyadi, & Ceulemans, 2018b) have revealed that smaller window sizes generally yield more accurate locations, the final recommendation is to select the locations yielded by the smallest window size that resulted in similar findings as the subsequent larger window sizes.

Regarding which *significance test* to apply, we recommend users conduct the more robust test—the variance drop test—and set this as the default option. Since future studies may aim to investigate the behavior of the variance test or of the combination of both tests for other running statistics, the variance test was also made available in the `kcpRS` package (see Section 2.2). For the statistical principles underlying this additional test, we refer to Cabrieto, Tuerlinckx, Kuppens, Hunyadi, et al. (Cabrieto, Tuerlinckx, Kuppens, Hunyadi, & Ceulemans, 2018b).

Next, users need to set the significance level α , following usual statistical inference guidelines. Hence, when monitoring a single type of running statistic, α could be set to 0.05. When monitoring multiple types of statistics (see Section 3.1), a correction such as the Bonferroni correction can be applied. If the user implements the variance test alongside the default variance drop test, the overall type 1 error rate across these two tests is automatically controlled through Bonferroni correction: if the overall significance level is set to α , each subtest uses the adjusted level $\alpha/2$. The presence of at least one change point is declared if at least one of the subtests is significant (Cabrieto, Tuerlinckx, Kuppens, Hunyadi, & Ceulemans, 2018b). In addition to testing the presence of at least one change point, the software also returns the exact p -values associated with each test. Users can obviously use these p -values to implement alternative correction methods if they wish (for an overview see e.g., Bretz et al., 2016).

Finally, we have built the `kcpRS` software such that the user can exploit multiple *CPU cores*. This option is especially beneficial in case the time series is considerably long, and the permutation test step proves to be computationally intensive. The maximum number of cores in the user's computer can be determined by running `detectCores()` which is available from the package `parallel`. Table 1 gives an indication of the computation time using a single CPU core and using the setting `ncpu = detectCores()` for the applications presented in this paper.

Performing the KCP-RS steps by means of the `kcpRS` package

In this section, we illustrate through two toy examples how one can use the `kcpRS` package to carry out a KCP-RS analysis. A third real data example on stock returns data can be found at <https://osf.io/zrh4v/>.

Table 1 Computation times in seconds for the example data sets, using a single versus 12 cores (`ncpu = 1` versus `ncpu = detectCores()`).

Data set	<code>ncpu = 1</code>	<code>ncpu = detectCores()</code>
Toy example 1	5.09	3.48
Toy example 2	3.92	3.03
Toy example 3	22.49	13.15
Mental load data	410.82	89.29
CO ₂ inhalation data	13.48	9.21

Toy example 1: Correlation change

The first example is taken from Cabrieto, Tuerlinckx, Kuppens, Hunyadi, et al. (Cabrieto, Tuerlinckx, Kuppens, Hunyadi, & Ceulemans, 2018b) and features simulated data comprised of three multivariate normal variables, two of which exhibit a correlation change. All three variables have zero means and unit variances within each phase. Three phases are simulated: in the first phase, the variables are uncorrelated, in the second phase, the first two variables correlate highly ($\rho = .7$), and in the last phase, all variables become uncorrelated again. The data are generated using the `mvtnorm` package using the following code:

```
R> library(mvtnorm)
R> m <- rep(0, 3)
R> cr <- diag(3)
R> c1 <- diag(3)
R> c1[1, 2] <- c1[2, 1] <- .7

R> set.seed(04222017)
R> phase1 <- rmvnorm(100, m, cr)
R> phase2 <- rmvnorm(50, m, c1)
R> phase3 <- rmvnorm(100, m, cr)
R> X <- rbind(phase1, phase2, phase3)
```

(Barnett & Onnela, 2016) or CUSUM (Galeano & Wied, 2017) were able to detect the change points, whereas KCP-RS on the running correlations was (see <https://osf.io/zrh4v/> for R code and output of the E-divisive method. Code for the other methods is available upon request). To analyze the data with the `kcpRS` package, we first load the package:

```
R> library(kcpRS)
```

Next, we use the main function `kcpRS` as follows, taking to heart the provided recommendations on how to select the window size (`wsiz`), the number of permutations (`nperm`) and the test (`varTest`³) for the significance testing, the maximum number of change points (`Kmax`), and the significance level α (`alpha`): We note that X should be a data frame where the rows pertain to the time points and the columns pertain to the variables. X is automatically scaled such that all variables have a variance of 1 to ensure that each variable will have the same influence while optimizing the KCP criterion. The argument `RS_fun` indicates which function should be used to compute the running statistic of interest. As mentioned earlier, there are four built-in functions⁴ that compute four popular running statistics: `runMean` for the running mean, `runVar` for the running variance, `runAR` for the running autocorrelation (lag 1) and `runCorr` for the running correlation. For this example, we use the `runCorr` function as we are interested in flagging correlation changes. The `RS_name` argument specifies the la-

```
R> kcpRS(data = X, RS_fun = runCorr, RS_name = "correlation",
+ wsiz = 25, nperm = 1000, Kmax = 10, alpha = 0.05, varTest = FALSE,
+ ncpu = 1)
```

The simulated time series are shown in Fig. 1a. Detecting these two correlation change points is challenging for two reasons: first, the middle phase, in which the change occurs, is smaller in size (50 time points) than the other two phases (100 time points). Second, only one of the three pairwise correlations exhibits the change. Therefore, the signal may not be large enough to detect it. Indeed, neither the general purpose method E-divisive (Matteson & James, 2014) nor correlation-specific methods such as Frobenius norm

bel of the running statistic and this is passed on to `kcpRS` methods such as `summary` and `plot` so that outputs are properly labelled.

³ The `varTest` argument allows one to implement the variance test alongside the variance drop test during the significance testing. This is by default set to `FALSE`, and the user has to set it to `TRUE` to carry out the additional variance test if desired.

⁴ The four built-in functions to extract the running statistics for the `kcpRS` package were built on the `roll` package (Foster, 2019).

First, the number of change points, K , resulting from the grid search (Fig. 1d) and the corresponding locations (Fig. 1b) are displayed. There are two change points detected, which is indeed the case for the toy example. The change point locations are placed at $T = 106$ and $T = 144$, which are proximal to the real change points at $T = 101$ and $T = 151$. As mentioned before (see Section 2.1 Step 2), we consider a change point to be proximal if it is located less than half of the window size from the true or expected change point (i.e., from 12 time points before to 12 time points after). Similar change points were found when using different window sizes, ranging from 10 to 35 (with steps of 5). The first change point ranged from $T = 106$ to $T = 109$ and the second change point ranged from $T = 143$ and $T = 148$. From a window size of 40

onwards, the change points were no longer detected. Next, the significance level chosen by the user as well as the p -value from the permutation test are displayed. For the toy example, the variance drop test (p -value = .002) is highly significant (Fig. 1c), confirming the presence of at least one change point. The last part of the output is the KCP solution for all K 's considered, including the value of the variance criterion as well as the locations of the extracted change points. Note that if `nperm` is set to 0, the significance test will not be carried out, and thus, only this table will be displayed.

To visualize the `kcpRS` solution, the output can be stored as a `kcpRS` object. Applying the `plot` function on this object will display the running statistics and the detected change point locations:

```
R> result <- kcpRS(data = X, RS_fun = runCorr, RS_name =
+ "correlation",
+ wsize = 25, nperm = 1000, Kmax = 10, alpha = 0.05, varTest = FALSE,
+ ncpu = 1)
R> plot(result)
```

The package also provides a summary function for the `kcpRS` class. When called, the summary function will display the settings specified by the user (see below) and the output of the analysis, which was already discussed above.

```
R> summary(result)

SETTINGS:
Running statistics monitored: correlation
Number of running correlations monitored: 3
Selected window size: 25
Number of time windows: 226
Selected maximum number of change points: 10

Permutation test: Yes
Number of permuted data sets used: 1000

OUTPUT:
..
```

Toy example 2: A user-specified running statistic

The `kcpRS` function was built to accommodate change point detection in any user-specified running statistic. As mentioned before, four built-in functions are already available in `kcpRS` to enable detecting change points in running means, variances, autocorrelations and correlations. If the user is however

interested in flagging change points in a different running statistic, the `kcpRS` function can be easily adapted. The user only has to provide and compile a function that can compute the running statistic of interest and indicate its name in the `RS_fun` argument. This function should accept two arguments, the data and the window size, and should output a data frame containing the running statistics. The rows of this data frame pertain to the windows and the columns pertain to the variables (combination of variables) for which the running statistic is computed.

As an illustration, let us return to the simulated data in the previous toy example and investigate whether the data contain changes in level. However, instead of the mean, we want to use a more robust measure of location: the median. The first step then is to write a function, which we call `runMed`, and which computes the running median:

```
R> library(roll)
R> runMed <- function(data, wsize = 25){
+ data <- as.data.frame(data)
+ labs <- colnames(data)
+ data <- as.matrix(data)
+
+ N <- nrow(data)
+ windows <- floor((N - wsize)) + 1
+
+ RunMed_TS <- roll_median(data, width = wsize)
+ RunMed_TS <- RunMed_TS[seq(wsize, wsize + windows - 1, 1), ]
+
+ RunMed_TS <- as.data.frame(RunMed_TS)
+ colnames(RunMed_TS) <- labs
+
+ return(RunMed_TS)
+ }
```

```
R> result <- kcpRS(X, RS_fun = runMed, RS_name = "median",
+ wsize = 25, nperm = 1000, Kmax = 10, alpha = 0.05, varTest = FALSE,
+ ncpu = 1)
R> summary(result)
```

After compiling this new function so that it is available in the current **R** environment, we can simply run the analysis using **kcpRS** by indicating `RS_fun = runMed` and supplying the label `RS_name = "median"`:

```
SETTINGS:
Running statistics monitored: median
Number of running medians monitored: 3
Selected window size: 25
Number of time windows: 226
Selected maximum number of change points: 10

Permutation test: Yes
Number of permuted data sets used: 1000

OUTPUT:
Number of change points detected: 0

The KCP permutation test is NOT significant:
Significance level: 0.05
P-value of the variance drop test: 0.783

...
```

The summary shows the settings of our tailored analysis, in which we monitored the running medians. The analysis output shows that no change point was detected, which makes sense as no changes in level were introduced in this toy example. This result was consistent across different window sizes, ranging from 10 to 50. Through this example, we have shown how flexible the **kcpRS** function is, and how easy it is to switch to another statistic of interest.

Monitoring multiple running statistics simultaneously

Statistical rationale of the simultaneous approach

In many substantive applications, the user wants to monitor a single type of user-specified statistic only. The proposed **kcpRS** function is perfectly suited for these situations. However, in other applications the user wants to track changes in multiple types of statistics, for instance because a crucial event occurred and they want to explore which types of changes transpired. To address these more complex questions, the **kcpRS** function is applied separately to all statistics of interest, while controlling for the type 1 error rate by applying a correction to **alpha**. A straightforward and simple method is the Bonferroni correction. This means that when the user wants to flag changes in four types of running statistics using an overall significance level α , each running statistic will be allotted a significance level, $\alpha/4$. Obviously, other less conservative corrections, such as the Holm procedure (for an overview see e.g., Bretz et al., 2016), can also be applied. To facilitate this, the **kcpRS** function outputs not only the (non) significance of an obtained solution, but also all exact *p*-values.

There is also an alternative, namely the KCP-RS workflow, which scans the time series for changes in means, variance, autocorrelation and correlation, while immediately taking the possible but rare effects of detected mean changes on other

running statistics into account.⁵ We will not go into detail, as the workflow is merely meant as a check that will in most cases yield the same results. In the case of contradicting findings, possible causes of the differences obviously should be carefully considered. The KCP-RS workflow is implemented in the function `kcpRS_workflow`. The output of the KCP-RS workflow for the examples presented in the following sections can be found at <https://osf.io/zrh4v/>.

Implementing the simultaneous approach using the `kcpRS` package

Toy example 3: Mean and correlation change

In this section, we will show how to simultaneously monitor multiple statistics using the `kcpRS` function by reanalyzing a simulated data example from Cabrieto, Tuerlinckx, Kuppens, Wilhelm, et al. (Cabrieto, Tuerlinckx, Kuppens, Wilhelm, et al., 2018c), in which both mean and correlation changes were introduced. Three multivariate normal variables with unit variances within each phase are simulated. Again, the

time series comprises three phases: in the first phase, all variables have zero means and are uncorrelated. In the second phase, a mean change of two standard deviations is introduced on the first two variables. Finally, in the third phase, a correlation change of .9 is applied to the same first two variables. The simulation code is as follows:

```
R> m <- c(0, 0, 0)
R> m1 <- c(2, 2, 0)
R> cr <- diag(3)
R> c1 <- diag(3)
R> c1[1, 2] <- c1[2, 1] <- .9

R> set.seed(072016)
R> phase1 <- rmvnorm(100, m, cr)
R> phase2 <- rmvnorm(100, m1, cr)
R> phase3 <- rmvnorm(100, m1, c1)
R> X <- rbind(phase1, phase2, phase3)
```

We apply the `kcpRS` function separately for all the statistics of interest. This is done as described in Section 2.2, but we now also control the overall type 1 error rate by applying the Bonferroni correction:

```
R> kcp_mean <- kcpRS(data = X, RS_fun = runMean, RS_name = 'Mean',
+ wsize = 25, nperm = 1000, Kmax = 10, alpha = (0.05 / 4),
+ varTest = FALSE, ncpu = detectCores())

R> kcp_var <- kcpRS(data = X, RS_fun = runVar, RS_name = 'Variance',
+ wsize = 25, nperm = 1000, Kmax = 10, alpha = (0.05 / 4),
+ varTest = FALSE, ncpu = detectCores())

R> kcp_corr <- kcpRS(data = X, RS_fun = runCorr,
+ RS_name = 'Correlation', wsize = 25, nperm = 1000, Kmax = 10,
+ alpha = (0.05 / 4), varTest = FALSE, ncpu = detectCores())

R> kcp_AR <- kcpRS(data = X, RS_fun = runAR,
+ RS_name = 'Autocorrelation', wsize = 25, nperm = 1000, Kmax = 10,
+ alpha = (0.05 / 4), varTest = FALSE, ncpu = detectCores())
```

The output of the `kcpRS` analyses can again be obtained using the summary function. For example, to inspect possible mean change points, we use:

```
R> summary(kcp_mean)
```

⁵ Behind the scenes, the KCP-RS workflow starts by implementing KCP on the running means to flag changes in level. If significant mean change points are found, the data will be centered using the phase-means implied by the change points, to filter out possible local influences on other running statistics. For example, a simultaneous mean change across several variables can artificially increase the variance in a window, and thus influence the results of KCP on the running variances. Next, the function automatically applies KCP on the running variances, correlations and autocorrelations of the phase-centered data. The function also controls the type 1 error rate by means of Bonferroni correction, but we again facilitate the application of other correction procedures by returning all exact *p*-values.

As the output of KCP-RS is presented in the same way as in toy examples 1 and 2, a summary is given in Table 2 (see the Appendix for the complete output). The main results shown are the number of change points detected for each running statistic, their locations and the significance level and *p*-values of the tests. In this toy example, Bonferroni correction was applied, since we set `alpha = (0.05/4)`. Thus, as shown in the output, the overall significance level for each statistic was adjusted to .0125, which is the overall significance level

Table 2 KCP-RS results for toy example 3. * indicates p -values lower than $\alpha = .0125$

Running statistic	p -value	Change point locations
Mean	0.000*	100
Variance	0.483	-
Correlation	0.000*	207
Autocorrelation	0.457	-

(.05) divided by 4, the number of running statistics monitored. Only the running means and the running correlations exhibited significant change points, and this is what we expect given the simulation code. The location of the mean change point is $T = 100$, while that of the correlation change point is $T = 207$. These KCP change points are proximal to the real mean and correlation change points which were set at $T = 101$ and $T = 201$, respectively. Moreover, **kcpRS** yielded similar change points across different window sizes, ranging from 10 to 50. Specifically, the detected mean change points ranged from $T = 100$ to $T = 107$, and the correlation change points from $T = 205$ to $T = 209$. The plots for the KCP-RS analyses

and the output for the KCP-RS workflow can be found at <https://osf.io/zrh4v/>.

Real data analysis

Mental load data

We will apply KCP-RS to second-by-second data on the mental load of aviation pilots, collected by Grassmann et al. (2016). Three physiological measures are monitored (i.e., heart rate, respiration rate and petCO_2) throughout an experiment to assess the pilots' mental load as they carry out multiple highly demanding tasks. There are four experimental phases: the resting baseline, the vanilla baseline, multiple tasks and recovery. Each phase lasted for approximately 6 minutes; however, the first and last 30 seconds of each phase were removed to exclude artifacts caused by speech or movement during experimental phase transitions. The four phases consisted of 332, 341, 380 and 340 seconds (i.e., time points), respectively. For a detailed description of each phase, see Grassmann et al. (2016). Cabrieto et al. (2017) analyzed data

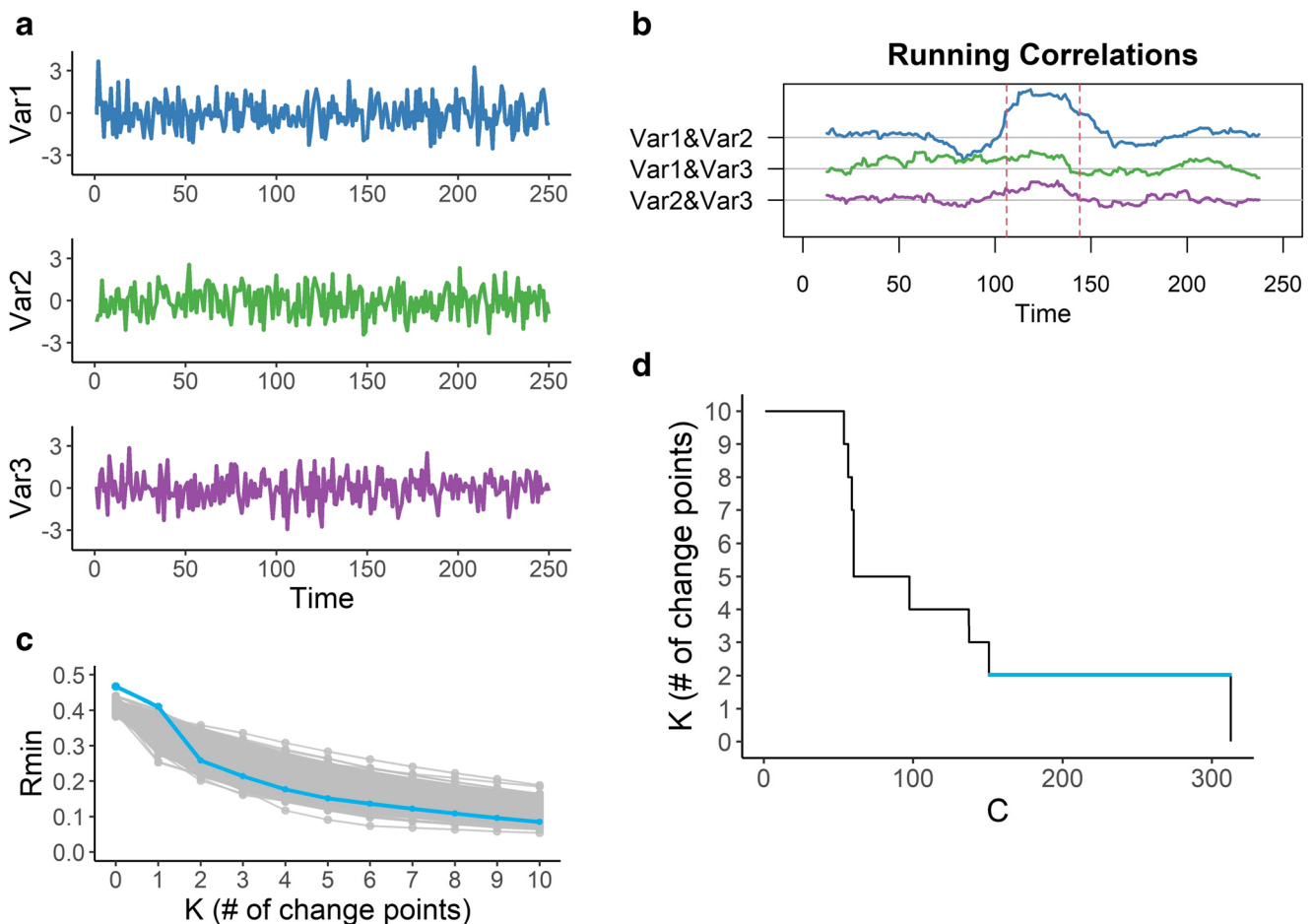


Fig. 1 Time series data for toy example 1. **a** The raw trivariate normal data are shown, with correlation change points at $T = 101$ and $T = 151$ for the first two variables. **b** The running correlations, where the red vertical lines indicate the detected change points. **c** Variance criterion \hat{R}_{min} values for the running

correlations, plotted against the number of change points K . The blue line indicates the values for the observed data whereas the grey lines denote the values of the permuted data sets. **d** Results of the grid search for the running correlation, where K is plotted against the penalty coefficient

from one of these pilots using general purpose methods, namely, E-divisive (Matteson & James, 2014), MultiRank (Lung-Yut-Fong et al., 2012), KCP (Arlot et al., 2012) and DeCon (Bulteel et al., 2014) and detected common change points at the boundaries of the multiple tasks period, which is a highly demanding phase. A post hoc analysis was carried out to test if changes occurred in the statistics of interest, revealing that both mean and correlation changes transpired.

The data set is available in the **kcpRS** package and can be loaded using:

```
R> data(MentalLoad)
```

To inspect whether there are changes in the means, variances, autocorrelations and correlations of the physiological measures, we apply the **kcpRS** function with Bonferroni correction to all four statistics. A window size of 20 was used, as this window size provided consistent results (see lower):

```
R> kcp_mean <- kcpRS(data = MentalLoad[, 2:4], RS_fun = runMean,
+ RS_name = 'Mean', wsize = 20, nperm = 1000, Kmax = 10,
+ alpha = (0.05 / 4), varTest = FALSE, ncpu = detectCores())

R> kcp_var <- kcpRS(data = MentalLoad[, 2:4], RS_fun = runVar,
+ RS_name = 'Variance', + wsize = 20, nperm = 1000, Kmax = 10,
+ alpha = (0.05 / 4), varTest = FALSE, ncpu = detectCores())

R> kcp_corr <- kcpRS(data = MentalLoad[, 2:4], RS_fun = runCorr,
+ RS_name = 'Correlation', wsize = 20, nperm = 1000, Kmax = 10,
+ alpha = (0.05 / 4), varTest = FALSE, ncpu = detectCores())

R> kcp_AR <- kcpRS(data = MentalLoad[, 2:4], RS_fun = runAR,
+ RS_name = 'Autocorrelation', wsize = 20, nperm = 1000, Kmax = 10,
+ alpha = (0.05 / 4), varTest = FALSE, ncpu = detectCores())
```

The output of KCP-RS can be found in the [Appendix](#). Change points are detected in the running means at $T = 673$ and $T = 1056$ (see also Fig. 2a, first panel). These locations are very close to the beginning and end of the multiple task phase at $T = 674$ and $T = 1054$ and correspond to the results of the general purpose methods. Note that these change points were consistently found across the KCP-RS analysis, with different window sizes ranging from 15 to 50, in which the first mean change ranged from $T = 670$ to $T = 673$ and the second mean change from $T = 1055$ to $T = 1057$. Furthermore, autocorrelation change points were found (Fig. 2a, third panel), but only for smaller window sizes (15 and 20). The last two autocorrelation change points are proximal to the running mean change points, implying that we can consider them common change points. No evidence was found for changes in variance⁶ nor correlation, contrary to the post hoc test results by Cabrieto et al. (2017), which yielded significant correlation changes. This inconsistency in conclusions about whether or not correlation changes are present is likely attributable to the very different nature of the statistical analyses that led to these conclusions. The post hoc analysis tested for correlation changes by entering the change points obtained with the general purpose approach as a predictor in a regression model, and thus considers these change points as known to test whether they might involve correlation changes. These tests were run for each variable pair separately. In the present analysis we rather aim to estimate the location and significance of correlation change points from the data. This single test also combines the information of all variable pairs. As holds for statistical analyses in general, changing test features can lead to different results.

We can thus conclude that, for the mental load data, three change points were found; the first one occurs quite early in the experiment and is an autocorrelation change, and the next two mark the beginning and the end of the multiple task phase where the mean and autocorrelation change. When interpreting these change points, one has to keep in mind that the test only indicates that at least one variable changes

significantly. It does not reveal either which or how many variables change, or the direction of changes. Hence, to interpret the obtained change points, we turn to the raw data and the running statistics as displayed in Fig. 2b. Regarding the mean change points around the multiple task, we see that during the multiple task, the mean levels of at least one of the physiological measures increased, likely reflecting the needed physiological arousal to fulfil the task at hand. Although harder to discern visually, autocorrelation levels (seem to) decrease at the start of the multiple task, possibly

⁶ Note that some variance change points were found when using the KCP-RS workflow, but only when using smaller window sizes.

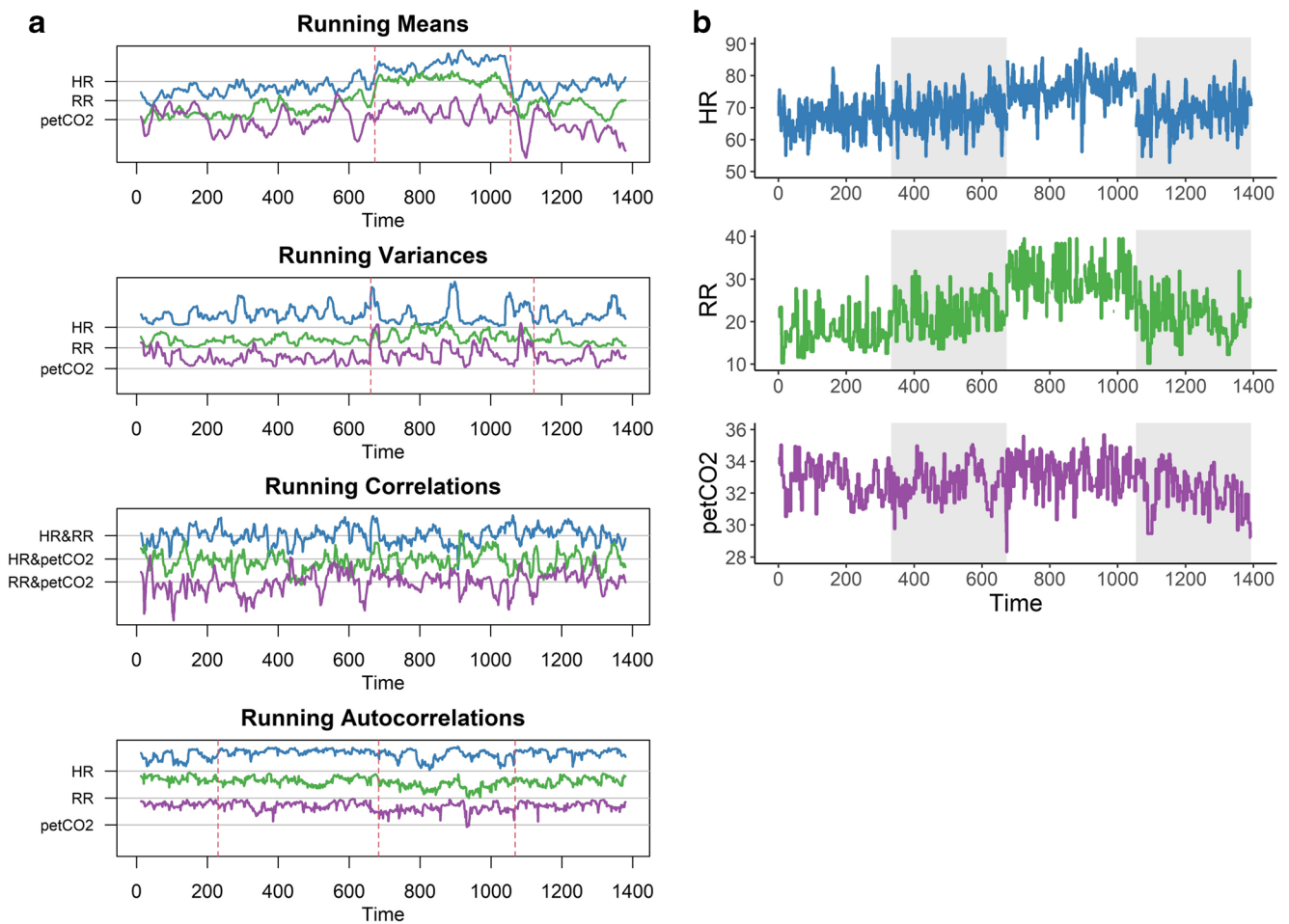


Fig. 2 **a** Running statistics and the obtained change points for the mental load data, using the KCP-RS function. **b** Raw data of the mental load measures, with the different background shading indicating the phases: resting baseline, vanilla baseline, multiple tasks and recovery

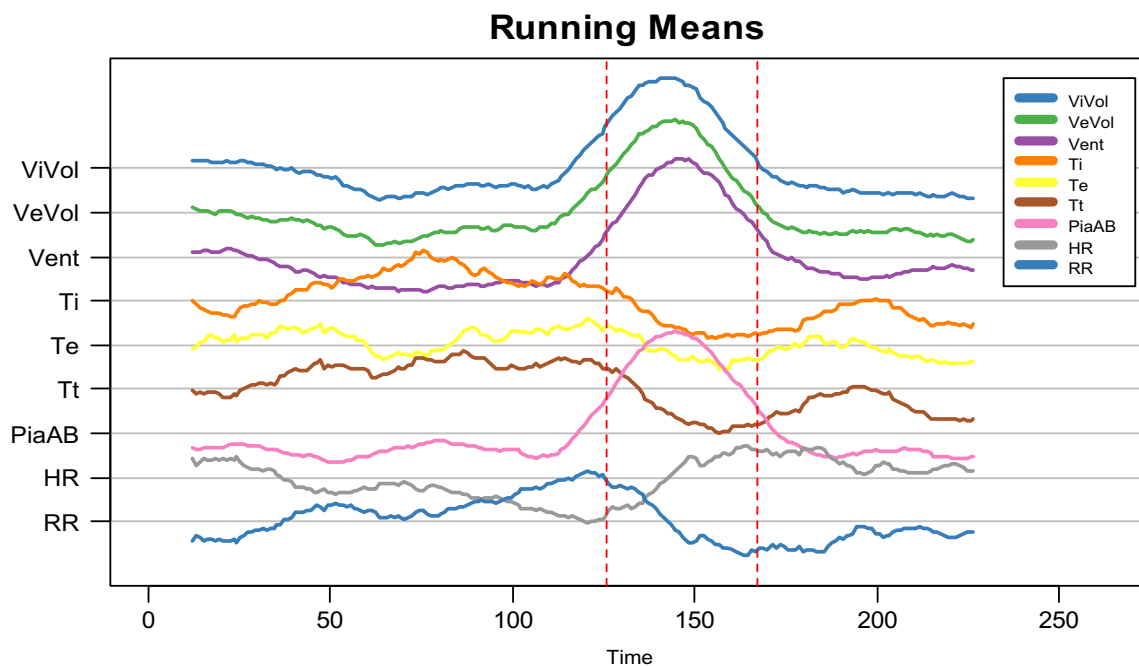


Fig. 3 Running means and the obtained change points for the CO₂-inhalation data

indicating a heightened flexibility to carry out the task. The autocorrelation levels (seem to) increase again near the end of the task.

(2014), we monitor the mean, variance and correlation using the `kcpRS` function:

```
R> kcp_mean <- kcpRS(data = CO2Inhalation[, 2:10], RS_fun = runMean,
+ RS_name = 'Mean', wsize = 25, nperm = 1000, Kmax = 10,
+ alpha = (0.05 / 3), varTest = FALSE, ncpu = detectCores())

R> kcp_var <- kcpRS(data = CO2Inhalation[, 2:10], RS_fun = runVar,
+ RS_name = 'Variance', wsize = 25, nperm = 1000, Kmax = 10,
+ alpha = (0.05 / 3), varTest = FALSE, ncpu = detectCores())

R> kcp_corr <- kcpRS(data = CO2Inhalation[, 2:10], RS_fun = runCorr,
+ RS_name = 'Correlation', wsize = 20, nperm = 1000, Kmax = 10,
+ alpha = (0.05 / 3), varTest = FALSE, ncpu = detectCores())
```

CO₂ inhalation data

The second empirical example is another physiological data set, resulting from a CO₂ inhalation experiment. These data were previously analyzed by De Roover et al. (2014) using the switching PCA technique to reveal mean and covariance changes in the nine monitored measures describing breathing volume (ViVol, VeVol, Vent and PiaAB), breathing duration (Ti, Te and Tt), heart rate (HR) and cardiac beat interval (RRi) (see Fig. 3). The experiment comprises three phases, namely, the baseline (115 time points), CO₂-inhalation (40 time points) and recovery phases (84 time points). While for a detailed description of the experimental setup, we refer the reader to De Roover et al. (2014), it is crucial to mention that in the baseline, the participant inhales room air through an inhalation tube; during the CO₂-inhalation phase the CO₂ level in this tube is increased by the experimenter, and during the recovery phase, the participant inhales room air again. The goal of the analysis is to uncover whether and which changes transpire during the CO₂-inhalation phase. The analysis of De Roover et al. (2014) reveals two change points that are proximal to the boundaries of the CO₂-inhalation phase. To make sense of these change points, a post hoc comparison between the recovered phases was carried out for three statistics: mean, variances and correlations⁷.

Using KCP-RS, the user specifies which statistics to monitor at the beginning of the analysis. Therefore, if a change point is detected, it is immediately clear which statistic exhibited the change. To load the data, the following code is run:

```
R> data(CO2Inhalation)
```

To focus on the same types of changes as De Roover et al.

The KCP-RS results show that only the running means changed significantly⁸, locating the change points at $T = 126$ and $T = 167$. Varying the window size from 10 to 30 further demonstrated the robustness of these findings, in that the location of the first mean change ranged from $T = 125$ to $T = 127$ and that of the second mean change from $T = 163$ to $T = 168$. No change points were found with larger window sizes. The two mean change points are close to the start ($T = 116$) and the end ($T = 156$) of the CO₂-inhalation phase, respectively, and to the locations found by De Roover et al. (2014). In Fig. 3, it can be seen that at the first change point the breathing volume variables (ViVol, VeVol, Vent and PiaAB) dramatically increased, whereas the breathing duration variables (Ti, Te and Tt) and RRi gradually dropped. At the second change point, it seems that only the breathing volume variables recovered quickly (i.e., returned to their baseline levels), while the rest of the variables still lingered at the CO₂ inhalation levels and recovered more slowly.

Conclusion

Through toy examples and real data analyses we have shown that the `kcpRS` package is a powerful and flexible change point detection tool to address two crucial questions: when exactly do change points occur, and what exactly changes at these change points. To the best of our knowledge, `kcpRS` is the only software package in **R** that can aid researchers in jointly tackling these questions via a multivariate and non-parametric approach. Our past studies (Cabrieto, Adolf, et al., 2018; Cabrieto, Tuerlinckx, Kuppens, Wilhelm, et al., 2018) have tested the performance of the KCP-RS approach for four popular statistics: mean, variance, autocorrelation and correlation, but the framework is not at all limited to these statistics. In fact, we have built a package that in principle

⁷ De Roover et al. (2014) explored the correlation structure by looking at the component structure.

⁸ The KCP-RS workflow yielded the same results.

allows us to track changes in any statistic of interest. With the help of the **kcpRS** package, future studies may therefore investigate how KCP-RS performs for other statistics and may shed further light on the mechanisms underlying crucial events.

Appendix

Toy example 3: Mean and correlation change

Output of the **kcpRS** function.

```
R> summary(kcp_mean)

SETTINGS:
Running statistics monitored: Mean
Number of running Means monitored: 3
Selected window size: 25
Number of time windows: 276
Selected maximum number of change points: 10

Permutation test: Yes
Number of permuted data sets used: 1000

OUTPUT:
Number of change points detected: 1
Change point location(s): 100

KCP permutation test is significant:
Significance level: 0.0125
P-value of the variance drop test: 0

Optimal change points given k:
k   Rmin CP1 CP2 CP3 CP4 CP5 CP6 CP7 CP8 CP9 CP10
0 0.5330 0 0 0 0 0 0 0 0 0 0
1 0.1865 100 0 0 0 0 0 0 0 0 0
2 0.1577 95 107 0 0 0 0 0 0 0 0
3 0.1295 99 176 255 0 0 0 0 0 0 0
4 0.1047 95 104 176 255 0 0 0 0 0 0
5 0.0910 34 96 106 176 255 0 0 0 0 0
6 0.0844 34 95 104 153 177 255 0 0 0 0
7 0.0763 34 95 104 125 153 177 255 0 0 0
8 0.0693 34 95 104 125 153 202 231 253 0 0
9 0.0624 34 95 104 125 153 176 202 231 253 0
10 0.0558 34 95 102 109 125 153 176 202 231 253
```

```
R> summary(kcp_var)

SETTINGS:
Running statistics monitored: Variance
Number of running Variances monitored: 3
Selected window size: 25
Number of time windows: 276
Selected maximum number of change points: 10

Permutation test: Yes
Number of permuted data sets used: 1000

OUTPUT:
```

Number of change points detected: 0

The KCP permutation test is NOT significant:

Significance level: 0.0125

P-value of the variance drop test: 0.483

Optimal change points given k:

k	Rmin	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
0	0.4445	0	0	0	0	0	0	0	0	0	0
1	0.4007	159	0	0	0	0	0	0	0	0	0
2	0.3402	80	144	0	0	0	0	0	0	0	0
3	0.3033	38	80	144	0	0	0	0	0	0	0
4	0.2679	38	80	107	144	0	0	0	0	0	0
5	0.2392	38	80	107	128	161	0	0	0	0	0
6	0.2125	38	71	90	107	128	161	0	0	0	0
7	0.1895	38	71	90	107	128	163	263	0	0	0
8	0.1688	38	71	90	107	128	159	244	261	0	0
9	0.1545	38	71	90	107	128	142	161	244	261	0
10	0.1413	38	71	90	107	128	144	165	210	235	261

R> summary(kcp_AR)

SETTINGS:

Running statistics monitored: Autocorrelation

Number of running Autocorrelations monitored: 3

Selected window size: 25

Number of time windows: 275

Selected maximum number of change points: 10

Permutation test: Yes

Number of permuted data sets used: 1000

OUTPUT:

Number of change points detected: 0

The KCP permutation test is NOT significant:

Significance level: 0.0125

P-value of the variance drop test: 0.457

Optimal change points given k:

k	Rmin	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
0	0.4085	0	0	0	0	0	0	0	0	0	0
1	0.3659	243	0	0	0	0	0	0	0	0	0
2	0.3050	178	240	0	0	0	0	0	0	0	0
3	0.2689	37	175	240	0	0	0	0	0	0	0
4	0.2292	92	111	175	240	0	0	0	0	0	0
5	0.1861	92	111	136	175	240	0	0	0	0	0
6	0.1615	35	92	111	136	175	240	0	0	0	0
7	0.1501	37	71	92	111	136	175	240	0	0	0
8	0.1392	37	71	92	111	136	175	234	244	0	0
9	0.1292	37	71	92	111	136	175	234	244	266	0
10	0.1192	37	71	92	111	136	175	198	213	233	244

R> **summary(kcp_corr)**

SETTINGS:

Running statistics monitored: Correlation
 Number of running Correlations monitored: 3
 Selected window size: 25
 Number of time windows: 276
 Selected maximum number of change points: 10

Permutation test: Yes
 Number of permuted data sets used: 1000

OUTPUT:

Number of change points detected: 1
 Change point location(s): 207

KCP permutation test is significant:
 Significance level: 0.0125
 P-value of the variance drop test: 0

Optimal change points given k:

k	Rmin	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
0	0.4581	0	0	0	0	0	0	0	0	0	0
1	0.2092	207	0	0	0	0	0	0	0	0	0
2	0.1787	66	207	0	0	0	0	0	0	0	0
3	0.1581	27	181	207	0	0	0	0	0	0	0
4	0.1415	26	75	111	207	0	0	0	0	0	0
5	0.1258	26	75	111	181	207	0	0	0	0	0
6	0.1127	26	75	111	181	196	208	0	0	0	0
7	0.0994	26	75	111	141	171	194	208	0	0	0
8	0.0886	26	75	111	141	171	194	208	238	0	0
9	0.0808	26	75	111	141	169	181	196	208	238	0
10	0.0720	26	75	111	141	171	194	208	238	249	277

Mental load data

Output of the kcpRS function.

R> **summary(kcp_mean)**

SETTINGS:

Running statistics monitored: Mean
 Number of running Means monitored: 3
 Selected window size: 20
 Number of time windows: 1374
 Selected maximum number of change points: 10

Permutation test: Yes
 Number of permuted data sets used: 1000

OUTPUT:

Number of change points detected: 2
 Change point location(s): 673 1056


```
KCP permutation test is significant:
Significance level: 0.0125
P-value of the variance drop test: 0
```

```
Optimal change points given k:
```

k	Rmin	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
0	0.4246	0	0	0	0	0	0	0	0	0	0
1	0.3399	600	0	0	0	0	0	0	0	0	0
2	0.2183	673	1056	0	0	0	0	0	0	0	0
3	0.2013	333	674	1056	0	0	0	0	0	0	0
4	0.1881	207	399	674	1056	0	0	0	0	0	0
5	0.1774	207	399	674	1055	1180	0	0	0	0	0
6	0.1674	207	556	604	677	1055	1180	0	0	0	0
7	0.1593	207	268	557	604	677	1055	1180	0	0	0
8	0.1517	207	556	604	677	1055	1084	1114	1178	0	0
9	0.1436	207	268	557	604	677	1055	1084	1114	1178	0
10	0.1364	44	207	268	557	604	677	1055	1084	1114	1178

```
R> summary(kcp_var)
```

```
SETTINGS:
```

```
Running statistics monitored: Variance
Number of running Variances monitored: 3
Selected window size: 20
Number of time windows: 1374
Selected maximum number of change points: 10
```

```
Permutation test: Yes
Number of permuted data sets used: 1000
```

```
OUTPUT:
```

```
Number of change points detected: 0
```

```
The KCP permutation test is NOT significant:
Significance level: 0.0125
P-value of the variance drop test: 0.036
```

```
Optimal change points given k:
```

k	Rmin	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
0	0.4344	0	0	0	0	0	0	0	0	0	0
1	0.4194	664	0	0	0	0	0	0	0	0	0
2	0.3979	664	1045	0	0	0	0	0	0	0	0
3	0.3827	664	684	1045	0	0	0	0	0	0	0
4	0.3691	664	684	1045	1118	0	0	0	0	0	0
5	0.3579	135	664	684	1045	1118	0	0	0	0	0
6	0.3469	664	684	886	907	1045	1118	0	0	0	0
7	0.3357	135	664	684	886	907	1045	1118	0	0	0
8	0.3257	135	664	684	863	886	907	1045	1118	0	0
9	0.3169	135	664	684	863	886	907	1045	1080	1118	0
10	0.3090	135	664	684	776	864	886	907	1045	1080	1118

```
R> summary(kcp_AR)
```

```
SETTINGS:
```

```

Running statistics monitored: Autocorrelation
Number of running Autocorrelations monitored: 3
Selected window size: 20
Number of time windows: 1373
Selected maximum number of change points: 10

Permutation test: Yes
  Number of permuted data sets used: 1000

```

OUTPUT:

```

Number of change points detected: 3
Change point location(s): 231 682 1066

```

```

KCP permutation test is significant:
Significance level: 0.0125
P-value of the variance drop test: 0.005

```

Optimal change points given k:

k	Rmin	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
0	0.4206	0	0	0	0	0	0	0	0	0	0
1	0.3950	230	0	0	0	0	0	0	0	0	0
2	0.3771	230	671	0	0	0	0	0	0	0	0
3	0.3526	230	681	1065	0	0	0	0	0	0	0
4	0.3408	230	664	792	866	0	0	0	0	0	0
5	0.3266	230	664	793	841	1064	0	0	0	0	0
6	0.3158	230	664	793	841	1020	1065	0	0	0	0
7	0.3068	230	664	792	867	921	1019	1065	0	0	0
8	0.2997	230	664	792	867	923	973	1020	1065	0	0
9	0.2935	230	481	542	664	792	867	921	1019	1065	0
10	0.2864	230	481	542	664	792	867	923	973	1020	1065

R> summary(kcp_corr)

SETTINGS:

```

Running statistics monitored: Correlation
Number of running Correlations monitored: 3
Selected window size: 20
Number of time windows: 1374
Selected maximum number of change points: 10

```

```

Permutation test: Yes
  Number of permuted data sets used: 1000

```

OUTPUT:

```

Number of change points detected: 0

```

```

The KCP permutation test is NOT significant:
Significance level: 0.0125
P-value of the variance drop test: 0.399

```

Optimal change points given k:

k	Rmin	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
0	0.4166	0	0	0	0	0	0	0	0	0	0
1	0.4015	419	0	0	0	0	0	0	0	0	0

```

2 0.3899 431 473 0 0 0 0 0 0 0 0
3 0.3839 46 426 473 0 0 0 0 0 0 0
4 0.3741 431 472 908 1027 0 0 0 0 0 0
5 0.3652 425 503 684 973 1027 0 0 0 0 0
6 0.3570 425 503 684 973 1048 1063 0 0 0 0
7 0.3495 431 473 666 684 973 1048 1063 0 0 0
8 0.3423 426 474 630 665 684 973 1048 1063 0 0
9 0.3362 46 425 498 630 665 684 973 1048 1063 0
10 0.3297 426 474 630 665 684 973 1021 1139 1186 1253

```

CO2 inhalation data

Output of the kcpRS function.

R> summary(kcp_mean)

SETTINGS:

```

Running statistics monitored: Mean
Number of running Means monitored: 9
Selected window size: 25
Number of time windows: 215
Selected maximum number of change points: 10

```

```

Permutation test: Yes
Number of permuted data sets used: 1000

```

OUTPUT:

```

Number of change points detected: 2
Change point location(s): 126 167

```

```

KCP permutation test is significant:
Significance level: 0.01666667
P-value of the variance drop test: 0.003

```

Optimal change points given k:

k	Rmin	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
0	0.4533	0	0	0	0	0	0	0	0	0	0
1	0.3498	126	0	0	0	0	0	0	0	0	0
2	0.2114	126	167	0	0	0	0	0	0	0	0
3	0.1628	41	126	167	0	0	0	0	0	0	0
4	0.1261	41	119	135	167	0	0	0	0	0	0
5	0.0971	41	119	134	157	170	0	0	0	0	0
6	0.0835	41	117	128	138	158	171	0	0	0	0
7	0.0737	41	99	119	130	140	158	171	0	0	0
8	0.0639	38	60	87	118	128	138	158	171	0	0
9	0.0548	38	60	87	118	128	138	157	168	180	0
10	0.0468	38	60	87	118	128	138	157	168	180	208

R> summary(kcp_var)

SETTINGS:

```

Running statistics monitored: Variance
Number of running Variances monitored: 9

```

Selected window size: 25
 Number of time windows: 215
 Selected maximum number of change points: 10

Permutation test: Yes
 Number of permuted data sets used: 1000

OUTPUT:

Number of change points detected: 0

The KCP permutation test is NOT significant:
 Significance level: 0.01666667
 P-value of the variance drop test: 0.497

Optimal change points given k:

k	Rmin	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
0	0.4160	0	0	0	0	0	0	0	0	0	0
1	0.3635	137	0	0	0	0	0	0	0	0	0
2	0.2982	117	135	0	0	0	0	0	0	0	0
3	0.2360	74	114	135	0	0	0	0	0	0	0
4	0.1918	74	113	137	172	0	0	0	0	0	0
5	0.1562	41	74	113	137	172	0	0	0	0	0
6	0.1268	41	74	114	135	152	172	0	0	0	0
7	0.1113	41	74	112	121	135	152	172	0	0	0
8	0.0975	41	74	112	121	135	152	172	205	0	0
9	0.0863	41	74	111	118	127	135	152	172	205	0
10	0.0753	41	74	111	118	127	135	152	170	180	205

R> summary(kcp_corr)

SETTINGS:

Running statistics monitored: Correlation
 Number of running Correlations monitored: 36
 Selected window size: 25
 Number of time windows: 215
 Selected maximum number of change points: 10

Permutation test: Yes
 Number of permuted data sets used: 1000

OUTPUT:

Number of change points detected: 0

The KCP permutation test is NOT significant:
 Significance level: 0.01666667
 P-value of the variance drop test: 0.466

Optimal change points given k:

k	Rmin	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
0	0.3988	0	0	0	0	0	0	0	0	0	0
1	0.3464	74	0	0	0	0	0	0	0	0	0
2	0.2887	111	174	0	0	0	0	0	0	0	0
3	0.2468	111	174	205	0	0	0	0	0	0	0
4	0.2063	74	115	174	205	0	0	0	0	0	0
5	0.1790	49	74	115	174	205	0	0	0	0	0
6	0.1596	49	74	111	154	174	205	0	0	0	0
7	0.1373	49	74	111	127	152	174	205	0	0	0
8	0.1206	49	74	111	127	138	152	174	205	0	0
9	0.1099	49	74	111	127	138	152	172	180	205	0
10	0.1000	49	74	89	111	127	138	152	172	180	205

Acknowledgements Jedelyn Cabrieto was supported by the Research Council of KU Leuven. In addition, the research leading to the results reported in this paper was sponsored by a research grant from the Fund for Scientific Research – Flanders (FWO, Project No. G074319N) awarded to Eva Ceulemans, by the Research Council of KU Leuven (C14/19/054), and by a research fellowship from the German Research Foundation (DFG, AD 637/1-1) awarded to Janne Adolf. The authors acknowledge Mariel Grassmann and Ilse Van Diest who shared the analyzed physiological data. These data are available as part of the **kcpRS** package.

References

- Arlot, S., Celisse, A., & Harchaoui, Z. (2012). *Kernel Multiple Change-point Detection*. <http://arxiv.org/abs/1202.3878v1>
- Barnett, I., & Onnela, J.-P. (2016). Change Point Detection in Correlation Networks. *Scientific Reports*, 6(1), 18893. <https://doi.org/10.1038/srep18893>
- Bodner, N., Tuerlinckx, F., Bosmans, G., & Ceulemans, E. (in press). Accounting for auto-dependency in binary dyadic time series data: A comparison of model- and permutation-based approaches for testing pairwise associations. *British Journal of Mathematical and Statistical Psychology*.
- Bretz, F., Hothorn, T., & Westfall, P. (2016). *Multiple Comparisons Using R* (0 ed.). Chapman and Hall/CRC. <https://doi.org/10.1201/9781420010909>
- Brodsky, B. E., & Darkhovsky, B. S. (1993). *Nonparametric Methods in Change-Point Problems*. Springer Netherlands <https://doi.org/10.1007/978-94-015-8163-9>
- Bulteel, K., Ceulemans, E., Thompson, R. J., Waugh, C. E., Gotlib, I. H., Tuerlinckx, F., & Kuppens, P. (2014). DeCon: A tool to detect emotional concordance in multivariate time series data of emotional responding. *Biological Psychology*, 98, 29–42. <https://doi.org/10.1016/j.biopsycho.2013.10.011>
- Bulteel, K., Mestdagh, M., Tuerlinckx, F., & Ceulemans, E. (2018). VAR(1) based models do not always outpredict AR(1) models in typical psychological applications. *Psychological Methods*, 23(4), 740–756. <https://doi.org/10.1037/met0000178>
- Cabrieto, J., Adolf, J., Tuerlinckx, F., Kuppens, P., & Ceulemans, E. (2018a). Detecting long-lived autodependency changes in a multivariate system via change point detection and regime switching models. *Scientific Reports*, 8(1), 15637. <https://doi.org/10.1038/s41598-018-33819-8>
- Cabrieto, J., Adolf, J., Tuerlinckx, F., Kuppens, P., & Ceulemans, E. (2019). An Objective, Comprehensive and Flexible Statistical Framework for Detecting Early Warning Signs of Mental Health Problems. *Psychotherapy and Psychosomatics*, 88(3), 184–186. <https://doi.org/10.1159/000494356>
- Cabrieto, J., Tuerlinckx, F., Kuppens, P., Grassmann, M., & Ceulemans, E. (2017). Detecting correlation changes in multivariate time series: A comparison of four non-parametric change point detection methods. *Behavior Research Methods*, 49(3), 988–1005. <https://doi.org/10.3758/s13428-016-0754-9>
- Cabrieto, J., Tuerlinckx, F., Kuppens, P., Hunyadi, B., & Ceulemans, E. (2018b). Testing for the Presence of Correlation Changes in a Multivariate Time Series: A Permutation Based Approach. *Scientific Reports*, 8(1). <https://doi.org/10.1038/s41598-017-19067-2>
- Cabrieto, J., Tuerlinckx, F., Kuppens, P., Wilhelm, F. H., Liedlgruber, M., & Ceulemans, E. (2018c). Capturing Correlation Changes by Applying Kernel Change Point Detection On the Running Correlations. *Information Sciences*. <https://doi.org/10.1016/j.ins.2018.03.010>
- Chen, J., & Gupta, A. K. (2012). *Parametric Statistical Change Point Analysis*. Birkhäuser Boston. <https://doi.org/10.1007/978-0-8176-4801-5>
- Davis, R. A., Lee, T. C. M., & Rodriguez-Yam, G. A. (2006). Structural Break Estimation for Nonstationary Time Series Models. *Journal of the American Statistical Association*, 101(473), 223–239. <https://doi.org/10.1198/016214505000000745>
- De Roover, K., Timmerman, M. E., Van Diest, I., Onghena, P., & Ceulemans, E. (2014). Switching principal component analysis for modeling means and covariance changes over time. *Psychological Methods*, 19(1), 113–132. <https://doi.org/10.1037/a0034525>
- Dürre, A., Fried, R., Liboschik, T., Rathjens, J., & R Core Team. (2015). *robts: Robust Time Series Analysis*. (R package version 0.3.0) [Computer software]. <https://rdrr.io/rforge/robts/>
- Erbas, Y., Ceulemans, E., Kalokerinos, E. K., Houben, M., Koval, P., Pe, M. L., & Kuppens, P. (2018). Why I don't always know what I'm feeling: The role of stress in within-person fluctuations in emotion differentiation. *Journal of Personality and Social Psychology*, 115(2), 179–191. <https://doi.org/10.1037/pspa0000126>
- Erdman, C., & Emerson, J. W. (2007). **hcp**: An R Package for Performing a Bayesian Analysis of Change Point Problems. *Journal of Statistical Software*, 23(3). <https://doi.org/10.18637/jss.v023.i03>
- Foster, J. (2019). *roll: Rolling Statistics* (R package version 1.1.2) [Computer software]. <https://cran.r-project.org/web/packages/roll/index.html>
- Galeano, P., & Wied, D. (2017). Dating multiple change points in the correlation matrix. *TEST*, 26(2), 331–352. <https://doi.org/10.1007/s11749-016-0513-3>
- Grassmann, M., Vlemincx, E., von Leupoldt, A., & Van den Bergh, O. (2016). The role of respiratory measures to assess mental load in pilot selection. *Ergonomics*, 59(6), 745–753. <https://doi.org/10.1080/00140139.2015.1090019>
- Killick, R., & Eckley, I. A. (2014). **changePoint**: An R Package for Change-point Analysis. *Journal of Statistical Software*, 58(3). <https://doi.org/10.18637/jss.v058.i03>
- Lavielle, M. (2005). Using penalized contrasts for the change-point problem. *Signal Processing*, 85(8), 1501–1510. <https://doi.org/10.1016/j.sigpro.2005.01.012>
- Lung-Yut-Fong, A., Lévy-Leduc, C., & Cappé, O. (2012). Homogeneity and change-point detection tests for multivariate data using rank statistics. *ArXiv:1107.1971 [Math, Stat]*. <http://arxiv.org/abs/1107.1971>
- Matteson, D. S., & James, N. A. (2014). A Nonparametric Approach for Multiple Change Point Analysis of Multivariate Data. *Journal of the American Statistical Association*, 109(505), 334–345. <https://doi.org/10.1080/01621459.2013.849605>
- Moulder, R. G., Boker, S. M., Ramseyer, F., & Tschacher, W. (2018). Determining synchrony between behavioral time series: An application of surrogate data generation for establishing falsifiable null-hypotheses. *Psychological Methods*, 23(4), 757–773. <https://doi.org/10.1037/met0000172>
- Ross, G. J. (2015). Parametric and nonparametric sequential change detection in R: the cpm package. *Journal of Statistical Software*, 66(3), 1–20.
- Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press <https://doi.org/10.1017/CBO9780511809682>
- Tárraga, M., Martí, J., Abella, R., Carniel, R., & López, C. (2014). Volcanic tremors: Good indicators of change in plumbing systems during volcanic eruptions. *Journal of Volcanology and Geothermal Research*, 273, 33–40. <https://doi.org/10.1016/j.jvolgeores.2014.01.003>
- Tartakovsky, A. G., Rozovskii, B. L., Blazek, R. B., & Hongjoong Kim. (2006). A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point

- detection methods. *IEEE Transactions on Signal Processing*, 54(9), 3372–3382. <https://doi.org/10.1109/TSP.2006.879308>
- Texier, G., Farouh, M., Pellegrin, L., Jackson, M. L., Meynard, J.-B., Deparis, X., & Chaudet, H. (2016). Outbreak definition by change point analysis: A tool for public health decision? *BMC Medical Informatics and Decision Making*, 16(1), 33. <https://doi.org/10.1186/s12911-016-0271-x>
- Zeileis, A., Leisch, F., Hornik, K., & Kleiber, C. (2002). **strucchange**: An R Package for Testing for Structural Change in Linear Regression Models. *Journal of Statistical Software*, 7(2). <https://doi.org/10.18637/jss.v007.i02>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.