HYDROLOGICAL PROCESSES
Hydrol. Process. **22**, 4600–4608 (2008)
Published online 3 June 2008 in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/hyp.7064

Fast segmentation algorithms for long hydrometeorological time series

Hafzullah Aksoy,[1]* Abdullah Gedikli,[2] N. Erdem Unal[1] and Athanasios Kehagias[3]

[1] *Istanbul Technical University, Department of Civil Engineering, Hydraulics Division 34469 Maslak, Istanbul, Turkey*
[2] *Istanbul Technical University, Department of Civil Engineering, Applied Mechanics Division 34469 Maslak, Istanbul, Turkey*
[3] *Aristotle University of Thessaloniki, School of Engineering, GR 541 24 Thessaloniki, Greece*

Abstract:

A time series with natural or artificially created inhomogeneities can be segmented into parts with different statistical characteristics. In this study, three algorithms are presented for time series segmentation; the first is based on dynamic programming and the second and the third—the latter being an improved version of the former—are based on the branch-and-bound approach. The algorithms divide the time series into segments using the first order statistical moment (average). Tested on real world time series of several hundred or even over a thousand terms the algorithms perform segmentation satisfactorily and fast. Copyright © 2008 John Wiley & Sons, Ltd.

KEY WORDS time series; segmentation; change point; dynamic programming; branch-and-bound approach

Received 31 August 2007; Accepted 24 March 2008

INTRODUCTION

Hydrometeorological time series are among the basic data used to study earth-related phenomena. Records of river streamflow, precipitation, temperature, etc. are scrutinized to detect regularities and trends, which can be used to predict their future behaviour (Aksoy *et al*., 2008). For example, the records of global temperature through the centuries can be used to test the hypothesis that a greenhouse effect is currently taking place which can result in environmental disaster. Equally important is the analysis of hydrological records and ozone concentration.

The detection of irregularities, jumps and changes is of the highest importance (Aksoy, 2007; Dahamsheh and Aksoy, 2007). Various natural or manmade actions can result in the changing behaviour of a hydrometeorological time series and such changes must be taken into account when extrapolating the past into the future. The time points where changes take place are called *change points*; the interval included between two change points is a *segment* (of the time series); and the procedure by which the segments of a time series are determined is called *time series segmentation*. Time series segmentation is an important problem in hydrometeorology (as well as in many other applied sciences, e.g. climatology and environmetrics) for which the development of fast and efficient segmentation algorithms emerges as a practically significant problem.

The problem of time series segmentation has received considerable attention in the hydrological literature. An early landmark study on hydrological segmentation was made by Buishand (1982). Many segmentation methods and a very extensive bibliography are presented in Basseville and Nikiforov (1993) where the emphasis is on the *online* segmentation.

In this study, *offline* segmentation algorithms, based on dynamic programming (DP) and branch-and-bound (BB) approaches are presented and evaluated on several hydrometeorological time series. A very broad dichotomy can be made between *offline* and *online* segmentation methods. In the offline segmentation, an entire time series $x_1, x_2, \ldots, x_T$ is given which is required to be divided into segments. In the online segmentation, on the other hand, the data points $x_1, x_2, \ldots, x_t, \ldots$ arrive one at a time and, at every time step t, it must be decided whether x_t belongs to the previous segment or should be assigned to a new segment which starts at t (Dobigeon and Tourneret, 2007).

The main inspiration of the current study has been Hubert's (2000) work on *multiple segments, offline* segmentation. A version of the DP-based algorithm (denoted as DP) has been presented in Kehagias *et al*. (2006); similarly, early versions of the BB-based algorithm (denoted as AUG) appear in Aksoy *et al*. (2007) and Gedikli *et al*. (2008). The DP and AUG algorithms yield optimal segmentations (in terms of a well-defined segmentation criterion) and can segment long time series of over one thousand terms in seconds. Both algorithms are evaluated on several real world hydrometeorological time series. All the studies above contain fairly extensive references to the segmentation literature, which was therefore omitted from the current study. However, some alternative approaches, for instance the Bayesian Markov Chain Monte Carlo (MCMC) approach of Fortin *et al*. (2004a,b)

* Correspondence to: Hafzullah Aksoy, Istanbul Technical University, Department of Civil Engineering, Hydraulics Division 34469 Maslak, Istanbul, Turkey. E-mail: haksoy@itu.edu.tr

and Kehagias and Fortin (2006) and the hidden Markov model (HMM) approach of Kehagias (2004) and Kehagias and Fortin (2006) should be noted.

The study is organized as follows. In the following section, a general formulation of the segmentation problem is presented, which will be used in subsequent sections. The DP and AUG segmentation algorithms are then presented and it is detailed how the AUG algorithm was improved. After the issue of determining the number of segments, use of the Scheffe (1959) criterion is discussed. The performance of the algorithms on several natural hydrometeorological time series are evaluated and compared in separate subsequent sections. Conclusions are finally drawn and directions for future research are presented.

FORMULATION OF THE PROBLEM

The formulation and notation of Hubert (2000) are followed with slight changes. While the final goal is the segmentation of a time series, what is really segmented is the set of integers $\{1, 2, \ldots, T\}$ (the time series gives the information by which the various segmentations are evaluated and compared). The segmentation can be described by a sequence $\mathbf{t} = (t_0, t_1, \ldots, t_K)$ satisfying $0 = t_0 < t_1 < \ldots < t_{K-1} < t_K = T$. The intervals of integers $[t_0 + 1, t_1], [t_1 + 1, t_2], \ldots, [t_{K-1} + 1, t_K]$ are called *segments*, the times $t_0, t_1, \ldots, t_K$ are called *segment boundaries* and K, the number of segments, is called the *order of the segmentation*.

The set of all segmentations of $\{1, 2, \ldots, T\}$ is denoted $\mathbf{T}$ and the set of all segmentations of the order K by $\mathbf{T}_K$. Clearly, $\mathbf{T} = \cup_{K=1}^{T} \mathbf{T}_K$. The number of all possible segmentations of $\{1, 2, \ldots, T\}$ is 2^{T-1}.

A time series $\mathbf{x} = \{x_1, x_2, \ldots, x_T\}$ is given and a segmentation of $\{1, 2, \ldots, T\}$ is sought which corresponds to changes in the behaviour of $\mathbf{x}$. This can be formulated as an optimization problem. In other words the optimal segmentation depends on $\mathbf{x}$. The so-called normalized segmentation cost $J(\mathbf{t})$ is defined

$$J(\mathbf{t}) = \sum_{k=1}^{K} d_{t_{k-1}+1, t_k} \tag{1}$$

where $d_{s,t}$ (for $0 \leq s < t \leq T$) is the normalized segment error corresponding to segment $[s, t]$. The segment error depends on the data vector $\mathbf{x} = (x_s, x_{s+1}, \ldots, x_t)$. A variety of $d_{s,t}$ functions can be used. The current study uses

$$\hat{d}_{s,t} = \sum_{\tau=s}^{t} (\chi_\tau - \mu_{s,t})^2 \tag{2}$$

where the segment-mean is given by

$$\mu_{s,t} = \frac{\sum_{\tau=s}^{t} \chi_\tau}{t - s + 1} \tag{3}$$

The normalized cost in Equation (1) is obtained by dividing the cost calculated in Equation (2) by $\hat{d}_{1,T}$ (the highest cost) such that the cost matrix is bound with zero at the lower limit and with 1 at the upper limit. Extensions using other regression functions are immediate (Kehagias *et al*., 2006).

The optimal segmentation, denoted by $\hat{\mathbf{t}} = (\hat{t}_0, \hat{t}_1, \ldots, \hat{t}_K)$ is defined as $\hat{\mathbf{t}} = \arg \min_{\mathbf{t} \in \mathbf{T}} J(\mathbf{t})$ and the optimal segmentation of order K, denoted by $\hat{\mathbf{t}}^{(K)} = \left(\hat{t}_0^{(K)}, \hat{t}_1^{(K)}, \ldots, \hat{t}_K^{(K)}\right)$, is defined as $\hat{\mathbf{t}}^{(K)} = \arg \min_{\mathbf{t} \in \mathbf{T}_K} J(\mathbf{t})$. The optimal segmentation can be found by exhaustive enumeration of all possible segmentations (and computation of the corresponding $d_{s,t}$). However, this is computationally infeasible, because the total number of segmentations increases exponentially in T. Hubert (2000) used a BB approach to search efficiently the set of all possible segmentations and stated that this approach can segment time series with several tens of terms but was not able "... to tackle series of much more than a hundred terms..." because of the combinatorial increase of computational burden. In Sections 3 and 4 in this study, algorithms which can segment time series with hundreds of terms in a few seconds are presented.

In order to obtain these fast algorithms, it will be useful to develop a fast method for computing the costs $d_{s,t}$. The recursive formulation of

$$d_{s,t+1} = d_{s,t} + (t - s + 1)(\mu_{s,t} - \mu_{s,t+1})^2 + (x_{t+1} - \mu_{s,t+1})^2 \tag{4}$$

is easily proven where

$$\mu_{s,t+1} = \frac{(t - s + 1)\mu_{s,t} + x_{t+1}}{t - s + 2}. \tag{5}$$

THE DP ALGORITHM

In this section, the DP segmentation algorithm is presented; it efficiently computes the optimal segmentation of order k for $k = 1, 2, \ldots, K$.

Consider the optimal segmentation of $x_1, x_2, \ldots, x_t$ which contains k segments; suppose its last segment is $[s + 1, t]$. Then the first $k - 1$ segments form an optimal segmentation of $x_1, x_2, \ldots, x_s$. More specifically, if c_t^k is the minimum segmentation cost of $x_1, x_2, \ldots, x_t$ into k segments then

$$c_t^k = c_s^{k-1} + d_{s+1,t} \tag{6}$$

is satisfied. Equation (6) allows the use of a typical dynamic programming approach to efficiently compute the optimal costs. Details of the DP algorithm can be found in Kehagias *et al*. (2006).

THE AUG ALGORITHM

As stated before, the AUG segmentation algorithm is based on the BB-type technique. The *branches* are

DOI: 10.1002/hyp

the possible segments of a kth order segmentation. As suggested by Hubert (2000), the upper bound u of the kth segment in a Kth order segmentation can trivially be given as

$$t_k \leq u = T - K + k \tag{7}$$

In the AUG algorithm, the term upper bound should not be understand as a constraint on the objective function of the optimization, but the criteria on the possible maximum value of t_k.

The easiest but most time-consuming formulation to determine the optimal segmentations of any order from $K = 2$ to $T - 1$ is presented in the following pseudocode referred to as the *primitive code*.

The Primitive code

Initialization
For $K = 1, \ldots, T$
$c_T^K = d_{1,T}$
Next K
Minimization
For $K = 2, \ldots, T-1$
For $t_1 = 1, \ldots, T - K + 1$
$C_1 = d_{1,t_1}$
For $t_2 = t_1 + 1, \ldots, T - K + 2$
$C_2 = C_1 + d_{t_1+1,t_2}$
...
For $t_k = t_{k-1} + 1, \ldots, T - K + k$
$C_k = C_{k-1} + d_{t_{k-1}+1,t_k}$
...
For $t_{K-1} = t_{K-2} + 1, \ldots, T - 1$
$C_{K-1} = C_{K-2} + d_{t_{K-2}+1,t_{K-1}}$
$C_K = C_{K-1} + d_{t_{K-1}+1,T}$
If $c_T^K > C_K$ then $c_T^K = C_K$ and $\hat{\mathbf{t}}^{(K)} = \{0, t_1, \ldots, t_{K-1}, T\}$
Next $t_{K-1}, \ldots, t_k, \ldots, t_2, t_1$
Next K

It is clear from the primitive code that it requires 2^{T-1} computational loops and is therefore not effective in obtaining all optimal segmentations. Loops in the primitive code are always completed from $K = 2$ to $T - 1$ and then a comparison and an update is made to minimize the cost which initially is taken equal to $d_{1,T}$. This also means that the cost of any kth order segmentation of the first t_k elements $c_{t_k}^k$ where $k < T$ is not considered in the primitive code. When this cost is considered, a more efficient way, referred to as the intermediate code, can be written as follows.

Intermediate code

Initialization
For $t = 1, .., T$
For $k = 1, .., t$
$c_t^k = d_{k,t}$
Next k,t
Minimization
For $K = 2, \ldots, T-1$
For $t_1 = 1, \ldots, T - K + 1$
$C_1 = d_{1,t_1}$
For $t_2 = t_1 + 1, \ldots, T - K + 2$
$C_2 = C_1 + d_{t_1+1,t_2}$
If $c_{t_2}^2 > C_2$ then $c_{t_2}^2 = C_2$ else Next t_2.
...
For $t_k = t_{k-1} + 1, \ldots, T - K + k$
$C_k = C_{k-1} + d_{t_{k-1}+1,t_k}$
If $c_{t_k}^k > C_k$ then $c_{t_k}^k = C_k$ else Next t_k.
...
For $t_{K-1} = t_{K-2} + 1, \ldots, T - 1$
$C_{K-1} = C_{K-2} + d_{t_{K-2}+1,t_{K-1}}$
If $c_{t_{K-1}}^{K-1} > C_{K-1}$ then $c_{t_{K-1}}^{K-1} = C_{K-1}$ else Next t_{K-1}.
$C_K = C_{K-1} + d_{t_{K-1}+1,T}$
If $c_{t_K}^K > C_K$ then $c_{t_K}^K = C_K$ and $\hat{\mathbf{t}}^{(K)} = \{0, t_1, \ldots, t_K\}$
Next $t_{K-1}, \ldots, t_k, \ldots, t_2, t_1$
Next K

When the cost of any kth order segmentation of the first t_k elements, $c_{t_k}^k$ for $k < T$ is considered, the process becomes much faster than that of the primitive code. It is also observed in the intermediate code that a comparison is made to the most recent updated cost and a new cost update takes place when applicable. Therefore, in this code, some of the segmentations with a cost higher than the updated cost are automatically eliminated to finally reduce required execution time.

The basic idea of the AUG algorithm (and, more generally, of the BB-type technique) is to enumerate (branch into) the possible solutions of the segmentation problem but to avoid exhaustive enumeration (which would require $O(2^{T-1})$ computation time) by eliminating clearly suboptimal solutions (bounds). Hence, before presenting the AUG algorithm, it is worth discussing upper and lower bounds of the segmentation, more specifically of the boundaries t_k (for $k = 1, 2, \ldots, K$).

In addition to those eliminated in the intermediate code, it is possible to further eliminate segmentations by reducing the upper bound of segments as defined in Equation (7). It is also easy to check that

$$c_{t+1}^k \geq c_t^k \geq (c_t^{k+1} \text{ and } c_{t+1}^{k+1}) \tag{8}$$

is valid for $t = 2, \ldots, T-1$ and $k = 1, \ldots, t$. The inequality (8) is rather obvious; a detailed derivation of it can be found in Gedikli *et al.* (2008) where four lemmas are given. In addition to Equation (8), it is also known that any k sequential segments extracted from the optimal segmentation are also optimal; i.e. if the cost of the optimal segmentation is $J(\hat{\mathbf{t}})$, then the cost $J(\hat{\mathbf{t}}_k)$ with change points $\mathbf{t}_k = \{t_0, t_1, \ldots, t_k\}$ also satisfies the optimality condition. It then becomes clear that a kth order segmentation of $x_1, \ldots, x_t$ with cost $c_t^{k-1} > c_T^K$ cannot be optimal (Gedikli *et al.*, 2008).

In order to reduce the upper bound u in this way, the *remaining cost* concept is defined as

$$R_{T,t}^{K,k} = c_T^K - c_t^k \tag{9}$$

where $k \leq K$ and $t \leq T$. Considering Equation (8), the reduced upper bound of the kth segment e can be obtained

DOI: 10.1002/hyp

as the largest integer satisfying

$$s \leq e \leq T - K + k \tag{10}$$

and

$$d_{s,e} \leq R_{T,s-1}^{K,k-1} \tag{11}$$

where s is the starting point of the kth segment. Based upon Equation (11), it is seen that the cost of the kth segment must be less than or equal to the remaining cost. When Equation (9) and inequality (11) are combined, it is noted for $k = 1$ that inequality (11) takes the form

$$d_{1,e} \leq c_T^K \tag{12}$$

since it is already known that

$$R_{T,0}^{K,0} = c_T^K \tag{13}$$

is valid. Considering the kth order segmentation of the subseries made of the first r items, and using inequality (11),

$$d_{s,r} \leq R_{e,s-1}^{k,k-1} \tag{14}$$

can be written and hence a new upper bound r satisfying

$$s \leq r \leq e \tag{15}$$

can be obtained. From our observations, it was noted that although additional computational efforts are required, locating r does not, for short series in particular, always result in a net gain in the execution time. However, it is worth emphasizing that the reduction in the upper bound of the segments is the unique feature of the AUG algorithm and it drastically reduces the number of possible segmentations evaluated by the algorithm in case of long time series in particular. Utilizing the above ideas, the AUG algorithm can now be described in the following pseudocode. This is an improved version of the AUG algorithm given in Gedikli *et al*. (2008), to be denoted as iAUG.

The iAUG algorithm

```
Input
The time series x_1, x_2, ..., x_T; the errors d_{s,t} (0 ≤ s < t ≤ T).
Initialization
c_0^1 = 0
For t = 1 to T
    For k = 1 to t
    c_t^k = d_{k,t}
    Next k
Next t

Main
K = 1
For K = 2 to T − 1
    Update(T, K, c)
    r = ReduceUB(T, K, 1, t_*, C_*, c_*^*)
    For t_1 = 1 to r
        C_1 = d_{1,t_1}
        c_{t_1}^1 = d_{1,t_1}
        r = ReduceUB(T, K, 2, t_*, C_*, c_*^*)
        For t_2 = t_1 + 1 to r
            C_2 = C_1 + d_{t_1+1,t_2}
        If c_{t_2}^2 > C_2 then c_{t_2}^2 = C_2 else Next t_2.
            ...
            r = ReduceUB(T, K, k, t_*, C_*, c_*^*)
            For t_k = t_{k-1} + 1 to T − K + k
                C_k = C_{k-1} + d_{t_{k-1}+1,t_k}
                If c_{t_k}^k > C_k then c_{t_k}^k = C_k else Next t_k.
                ...
            r = ReduceUB(T, K, K − 1, t_*, C_*, c_*^*)
                For t_{K-1} = t_{K-2} + 1 to T − 1
                    C_{K-1} = C_{K-2} + d_{t_{K-2}+1,t_{K-1}}
                      If c_{t_{K-1}}^{K-1} > C_{K-1} then c_{t_{K-1}}^{K-1} = C_{K-1}
else Next t_{K-1}.
                    C_K = C_{K-1} + d_{t_{K-1}+1,T}
                    If c_{t_K}^K > C_K then
                        c_{t_K}^K = C_K
                    t̂^(K) = {0, t_1, ..., t_K}
    Next t_{K-1}, ..., t_k, ..., t_2, t_1
    Next K

Function ReduceUB(T, K, k, t_*, C_*, c_*^*)
    e = T − K + k
    r = Reduce(T, K, k, t_*, C_*, c_*^*)
    While (e > r)
        e = r
        r = Reduce(e, k, k, t_*, C_*, c_*^*)
    End While
    Return r
Function Reduce(T, K, k, t_*, C_*, c_*^*)
        ! p and q are dummy variables as integer.
        p = s = t_{k-1} + 1
        u = T − K + k
        R = c_T^K − C_{k-1}      ! Remaining cost defined by Equation (9)
        While (u > p + 1)
        q = (u + p)/2      ! Round down to the nearest integer.
            If d_{s,j} > R then u = q else p = q
        End While
        Return u
Subroutine Update(T, K, c_*^*)
    For t = K + 1 to T
        If c_t^{K-1} < c_{t-1}^{K-1} then c_{t-1}^{K-1} = c_t^{K-1}
        If c_{t-1}^{K-1} < c_t^K then c_t^K = c_{t-1}^{K-1}
    Next k
```

As stated above, this is an improved version of the AUG algorithm (iAUG). The basic improvement is present in the global update phase of the algorithm. The pseudocode of the global update used in the AUG algorithm is provided below to demonstrate the difference between the iAUG algorithm as given above.

```
Subroutine Update(T, K, c_*^*)
    For k = K to T
```


DOI: 10.1002/hyp

For $t = k + 1$ to T
If $c_t^{k-1} < c_{t-1}^{k-1}$ then $c_{t-1}^{k-1} = c_t^{k-1}$
If $c_{t-1}^{k-1} < c_t^k$ then $c_t^k = c_{t-1}^{k-1}$
Next t
Next k

When the pseudocodes are compared, it is seen that the cost values are updated according to the lemmas for all segmentation orders, $K-1, K, K+1, \ldots, T$ in the AUG algorithm while an improvement is made in the iAUG algorithm by updating the cost for segmentation orders $K-1$ and K only. That is, the number of loops in the global update is reduced to 1 from 2 in the iAUG algorithm, i.e. the process runs in order O(T) not in $O(T^2)$ as before. The algorithms were all written and compiled (in the release mode) with Microsoft Visual C++ 6·0 and run on an Intel Pentium 4 CPU 3·00 GHz, 2·00 GB of RAM running Microsoft Windows XP. The algorithms are online available at http://www2.itu.edu.tr/~gedikliab/Segmentation.

DETERMINING THE NUMBER OF SEGMENTS

The algorithms compute a *sequence* of optimal segmentations $\hat{\mathbf{t}}_1, \hat{\mathbf{t}}_2, \ldots, \hat{\mathbf{t}}_K$ where $\hat{\mathbf{t}}_k$ is the kth order optimal segmentation. Determining the optimal *order* of segmentation i.e. selecting the number of segments, is however a subsequent step in the segmentation procedure to be performed for which the Scheffe test is employed.

The Scheffe test is based on the following idea. For a given segmentation ($\hat{\mathbf{t}}_k$ for instance), the hypothesis that the means of consecutive segments are significantly different is tested. This is done using Scheffe's hypothesis test (Scheffe, 1959) which, in short, is a very general multiple means comparison test. We run this test on the optimal segmentations $\hat{\mathbf{t}}^{(1)}, \hat{\mathbf{t}}^{(2)}, \ldots, \hat{\mathbf{t}}^{(K)}$. In the Hubert (2000) algorithm, $\hat{\mathbf{t}}^{(k)}$ is accepted as the optimal segmentation when $\hat{\mathbf{t}}^{(k+1)}$ is the first lowest order segmentation rejected by Scheffe's test (i.e. the first segmentation for which at least two consecutive segments do not show a statistically significant difference in their means). In this study, the *highest order* segmentation accepted by the Scheffe test is considered instead of the *first lowest*.

EXPERIMENTS

In this section, the performance of the DP, AUG and iAUG algorithms using several data sets was studied. In all of the experiments presented here, the three algorithms have obtained identical segmentations of all orders. This is not surprising, since the algorithms minimize (in an exact manner, without approximation) the same segmentation cost. Hence, in every one of the following experiments, results are presented in a single table which contains the segmentations (obtained by either DP, AUG or iAUG) of all orders up to the highest order accepted by the Scheffe test.

Given below are results of the experiments performed using hydrometeorological time series of streamflow, precipitation and temperature, with a short analysis. The analysis does not aim to concentrate on hydrology, meteorology or climatology but only on the performance of the DP, AUG and iAUG algorithms on time series segmentation as a computational problem. If desired, the results can be submitted to hydrologists, meteorologists and climatologists for further physical interpretation.

Experiment 1

This time series has a length of 131 years and consists of the annual total precipitation data (in mm) at Fortaleza, Brazil, for the period 1849–1979. The time series has been presented in Morettin *et al*. (1987). The algorithms have been applied to the time series. The segmentations obtained are listed in Table I up to the 4th order, which is the highest order segmentation accepted by the Scheffe test. In Figure 1, both the original time series and segment

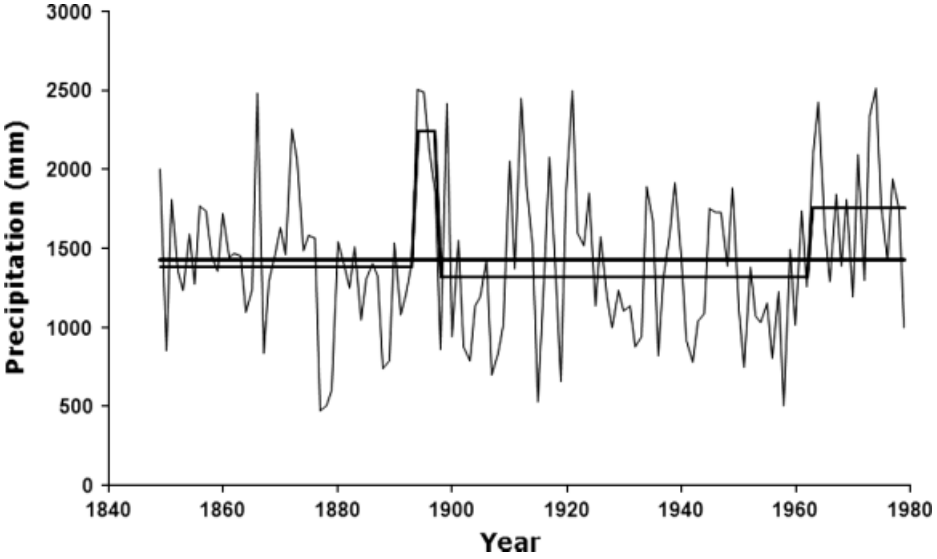

Figure 1. Segmentation of the Fortaleza total annual precipitation data (1849–1979) for $K = 4$

DOI: 10.1002/hyp

Table I. Change points in the optimal segmentations of the Fortaleza annual precipitation data, for orders $K = 2, 3, 4$

K	Change points				
2	1848	1962	1979		
3	1848	1949	1960	1979	
4	1848	1893	1897	1962	1979

means corresponding to the 4th order segmentation are plotted together with long-term average.

With the exception of the 4 year segment during the period 1893–1896, the annual precipitation in Fortaleza can be considered stable for more than a century until 1962, after which an increase is observed up to the end of the observation period, 1979. Figure 1 demonstrates that the annual precipitation ranges approximately from 500 mm to 2500 mm except for the last segment in which the minimum annual precipitation remains higher than 1000 mm; this is considered the reason for the upward shift in the time series. Having consecutive years with high precipitation reaching about 2500 mm at annual scale resulted in the 4 year segment mentioned above.

Experiment 2

The second time series used in the study has 581 years and consists of the 'hockey-stick' data, the northern hemisphere mean temperature for the period 1400–1980 (McIntyre and McKitrick, 2005). The data set was reconstructed by Mann *et al*. (1999, 2000) using proxy data networks.

The algorithms were applied to the time series. The segmentations obtained are listed in Table II up to the 10th order, which is the highest order segmentation accepted by the Scheffe test. In Figure 2, the original time series was plotted together with segment means corresponding to the 10th order segmentation as well as its long-term average. The most dominant observation made in Figure 2 is the rapid increase taking place in early 20th century to stabilize after the second half of

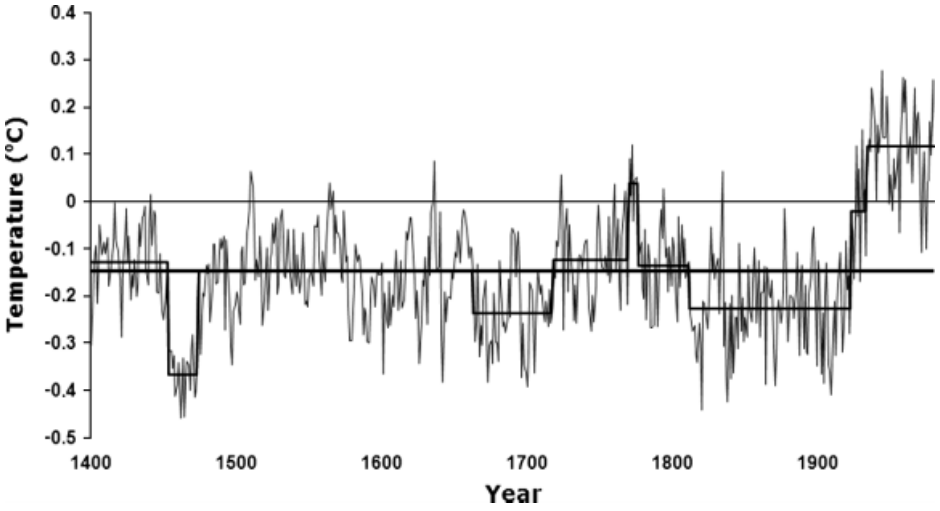

Figure 2. Segmentation of the northern hemisphere mean temperature (1400–1980) for $K = 10$

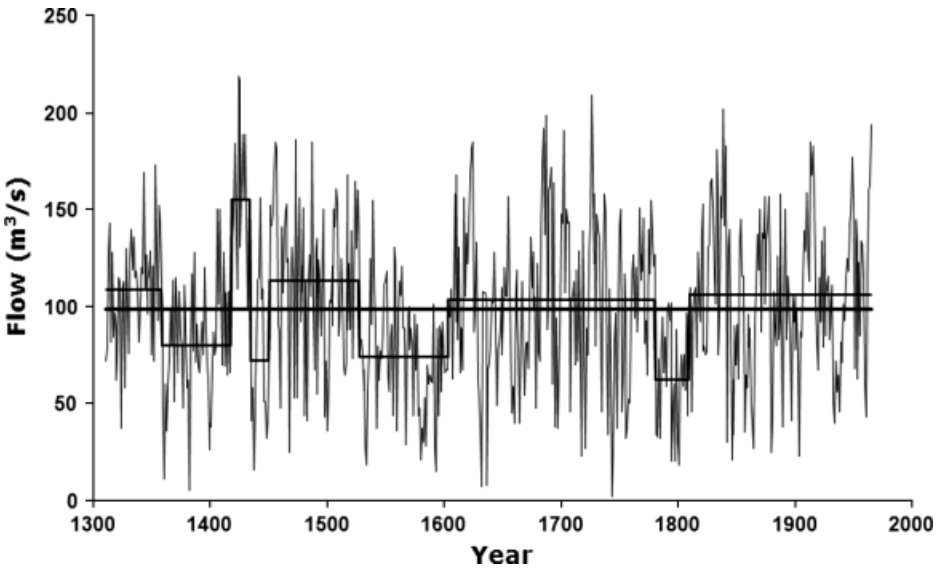

Figure 3. Segmentation of the annual streamflow data at Limber Pine Dell, Montana, US (1311–1965) for $K = 9$

DOI: 10.1002/hyp

Table II. Change points in the optimal segmentations of the northern hemisphere mean temperature (1400–1980) for orders $K = 2, 3, \ldots, 10$

K	Change points									
2	1399	1925	1980							
3	1399	1811	1925	1980						
4	1399	1453	1473	1925	1980					
5	1399	1453	1473	1811	1925	1980				
6	1399	1453	1473	1747	1811	1925	1980			
7	1399	1453	1473	1662	1717	1811	1925	1980		
8	1399	1453	1473	1662	1717	1811	1922	1933	1980	
9	1399	1453	1473	1662	1717	1769	1776	1811	1925	1980
10	1399	1453	1473	1662	1717	1769	1776	1811	1922	1933	1980

Table III. Change points in the optimal segmentations of the Limber Pine, Dell, Montana, US for orders $K = 2, 3, \ldots, 9$

K	Change points									
2	1310	1962	1965							
3	1310	1417	1434	1965						
4	1310	1358	1417	1434	1965					
5	1310	1417	1431	1527	1603	1965				
6	1310	1417	1434	1450	1527	1603	1965			
7	1310	1417	1431	1527	1603	1780	1809	1965		
8	1310	1417	1434	1450	1527	1603	1780	1809	1965	
9	1310	1358	1417	1434	1450	1527	1603	1780	1809	1965

the century. It is this increase which has led to the data set being referred to as the 'hockey-stick' data/graph (McIntyre and McKitrick, 2005). Another observation made from the time series is the departure of the mentioned increase from the overall mean value of the time series, recorded as the maximum departure through the time series.

Experiment 3

The third time series consists of streamflow data from the Limber Pine, Dell, Montana, US for the period 1311–1965. The time series was used in Hipel and McLeod (1994) and has a length of 655 years.

Again the three algorithms were applied. The segmentations obtained are listed in Table III up to the 9th order, which is the highest order segmentation accepted by the Scheffe test. In Figure 3, the time series, the ninth-order segmentation and the long-term average are plotted. It is seen that the first six segments fluctuate around the overall mean value of the time series and then a steady regime is noted for the data set when the 29 year 8th segment is considered null.

Experiment 4

Finally, the algorithms were applied to the time series of minimum water level data of the River Nile for the years 622–1918. These data can also be found in Hipel and McLeod (1994). It has previously been used in Kehagias (2004), Kehagias *et al*. (2006, 2007), Aksoy *et al*. (2007) and Gedikli *et al*. (2008).

While the time series purportedly covers the years 622–1921 and should have length 1300, its actual length is 1297. We assume that the actual years corresponding to these 1297 data points are 622–1918. With 1297 points, it is the longest of all data sets used in this study.

The segmentations obtained are listed in Table IV up to order 16, which is again the highest order segmentation accepted by the Scheffe test. (In previous applications of this data set (Aksoy *et al*., 2007; Gedikli *et al*., 2008), the optimal segmentation was mistakenly printed as 14 instead of 16). In Figure 4, the 16th order segmentation of the time series is plotted together with the original time series and its long-term average. As previously observed by Gedikli *et al*. (2008), a very long stable period was located for a period of 294 years beginning very early in the 9th century. A segment of constant values was discovered by both algorithms starting with 1528, which can be considered as verification that the algorithms work properly. This data set can also be considered a 'hockey-stick' graph as an increase trend is observed in last decades of the time series.

Comparison

Based upon the experimental results detailed above, the algorithms are seen to minimize the segmentation cost defined by Equations (1–3). Since the minimization is exact, the algorithms naturally give the same results. The only comparison possible between the algorithms is therefore in terms of the execution time, which can be seen in Table V from which it is observed that the DP algorithm is much faster than the AUG and its improved version. It is also seen that with the improvement in the AUG algorithm, the process is completed faster. However, it should be mentioned that the three algorithms can produce segmentations of time series on which the Hubert (2000) algorithm does not even terminate.

DOI: 10.1002/hyp

Table IV. Change points in the optimal segmentations of the Nile minimum level, for orders $K = 2, 3, \ldots, 16$

K	Change points																
2	621	1857	1918														
3	621	1527	1583	1918													
4	621	1527	1583	1857	1918												
5	621	1426	1527	1583	1857	1918											
6	621	1017	1428	1527	1583	1857	1918										
7	621	1081	1196	1426	1527	1583	1857	1918									
8	621	1081	1196	1426	1527	1583	1836	1887	1918								
9	621	731	804	1081	1196	1426	1527	1583	1857	1918							
10	621	731	804	1081	1196	1426	1527	1583	1836	1887	1918						
11	621	731	804	1098	1131	1196	1426	1527	1583	1836	1887	1918					
12	621	731	804	1098	1131	1196	1426	1527	1583	1619	1836	1887	1918				
13	621	731	804	1098	1131	1196	1353	1396	1426	1527	1583	1836	1887	1918			
14	621	731	804	1098	1131	1196	1353	1396	1426	1527	1583	1619	1836	1887	1918		
15*	621	731	804	1098	1131	1196	1356	1357	1396	1426	1527	1583	1619	1836	1887	1918	
16	621	731	804	1098	1131	1196	1353	1396	1426	1527	1583	1619	1798	1822	1857	1889	1918

* Rejected by the Scheffe test

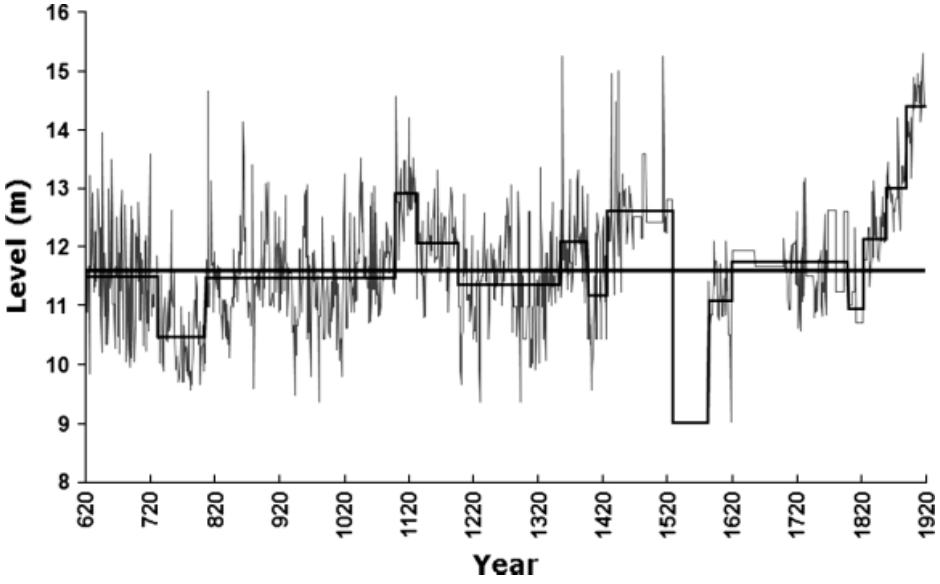

Figure 4. Segmentation of the minimum water level data of the River Nile (622–1918) for $K = 16$

CONCLUSIONS AND FUTURE RESEARCH

The following conclusions can be drawn from the study.

(1) *Execution time*: Using the pioneering work of Hubert (2000) as the starting point, three segmentation algorithms are obtained. The algorithms use recursive computation and are able to segment time series of over a thousand samples in a few seconds (DP) or just a minute or two (AUG and iAUG). Hence, they are all fast in terms of computer time and can handle much longer time series than the Senegal River annual flow time series, which was shorter than 100 years at the time it was used by Hubert (2000). Because of the speed of execution, the DP algorithm, the AUG algorithm and its improved version can be used not only as standalone segmentation algorithms but also as exploratory tools, used by a human operator to quickly and interactively explore the set of candidate segmentations.

(2) *Variation in the time series*: One important point observed during the study is that the performance of the AUG and iAUG algorithms depends not only on the length of the time series but also on its variation. For example, the execution time required for a time series may differ from the execution time required when the time series is reversed as $\mathbf{x} = (x_T, x_{T-1}, \ldots, x_1)$. Observations on that issue show that a longer execution time is required when the time series is more variable at the beginning than its end.

Table V. Execution time (s) for the DP, AUG and iAUG algorithms

Experiment	Length	DP	AUG	iAUG
1	131	0·016	0·110	0·079
2	581	0·625	12·109	10·981
3	655	0·875	47·750	45·500
4	1297	7·140	87·234	67·640

DOI: 10.1002/hyp

(3) *Upper bound reduction*: It is also noted from our observations that, although additional computational efforts are required, the upper bound reduction does not (for short time series in particular) always result in a net gain in the execution time. The AUG and iAUG algorithms strictly depend on the intermediate-code presented in the study. The only way to arrive at faster algorithms is the reduction of upper bounds of each branch/segment. By using Equations (9–15), this reduction is performed. If a more effective reduction method is obtained in the future, then faster versions of the AUG and iAUG algorithms can be written. The DP algorithm can also be made faster than its present version if the upper bound reduction is incorporated.

(4) *Linear or quadratic segmentation for future studies*: Various issues remain open and we plan to address them in future research. As presented in this study, the algorithms are based on a 'mean-inhomogeneity' segmentation cost. But they can be modified to incorporate other segmentation costs based on various autoregressive models, for example linear (Kehagias, 2004), quadratic, exponential or logarithmic. Or even, a segmentation algorithm can be formulated combining all these approximations. In particular, the linear segmentation among these approximations can serve as an exploratory tool for trend analysis, a recent popular topic in environmetrics and climatology to detect climate change or climate variability, for instance.

(5) *Multiple optimal segmentation*: Another worthwhile research topic is to address the issue of non-uniqueness of optimal segmentation (i.e. the case when two or more segmentations yield the same cost). In such cases, the current AUG and iAUG codes (in C++) report the minimum number of non-unique solutions alternating each other. The DP algorithm can be modified to track such multiple optimal segmentations.

(6) *Dominant change points*: When the change points are analysed, it is seen that certain points in the time series always appear as change points of the optimal segmentations after a critical segmentation order is reached. Such change points can be referred to as *dominant change points* and their properties merit further analysis in the future. The dominant change points can help in determining the optimal segmentation with considerably reduced execution time provided that the time series is divided into sub-series using the dominant change points.

(7) *Evaluation of Type I and Type II errors*: There are two types of error which a time series segmentation algorithm can commit. It can produce segments of a data sequence which was actually produced by a stationary process (Type I error) or ignore the changes of a non-stationary time series (Type II error). These two types of errors are somewhat complimentary, i.e. an algorithm can be prone to either over-segment or under-segment. To study the interplay of the two error types the true segmentation must be known, which will generally be the case only for *artificially created* time series. The developed algorithms are planned for further applications in the future for this type of study.

ACKNOWLEDGEMENTS

The authors thank Dr Pierre Hubert of Université P. & M. Curie, Paris, France for sharing his software of automatic segmentation algorithm online.

REFERENCES

Aksoy H. 2007. Hydrological variability of the European part of Turkey. *Iranian Journal of Science and Technology, Transaction B—Engineering* **31**: 225–236.

Aksoy H, Unal NE, Gedikli A. 2007. Letter to the editor. *Stochastic Environmental Research and Risk Assessment* **21**: 447–449.

Aksoy H, Unal NE, Alexandrov V, Dakova S, Yoon J. 2008. Hydrometeorological analysis of northwestern Turkey with links to climate change. *International Journal of Climatology* (in press) DOI: 10·1002/joc.1599.

Baseville M, Nikiforov IV. 1993. *Detection of Abrupt Changes: Theory and Application.* PRT Prentice Hall: Englewood Cliffs, N.J.

Buishand TA. 1982. Some methods for testing the homogeneity of rainfall records. *Journal of Hydrology* **58**: 11–27.

Dahamsheh A, Aksoy H. 2007. Structural characteristics of annual precipitation data in Jordan. *Theoretical and Applied Climatology* **88**: 201–212.

Dobigeon N, Tourneret JY. 2007. Joint segmentation of wind speed and direction using a hierarchical model. *Computational Statistics & Data Analysis* **51**: 5603–5621.

Fortin V, Perreault L, Salas JD. 2004a. Restropective analysis and forecasting of streamflows using a shifting level models. *Journal of Hydrology* **296**: 135–163.

Fortin V, Perreault L, Salas JD. 2004b. Analyse retrospective et prevision des debits en presence de changements de regime. *In Proceedings of 57th Annual meeting of the Canadian Water Resources Association*, Montreal, June 16–18, 2004.

Gedikli A, Aksoy H, Unal NE. 2008. Segmentation algorithm for long time series analysis. *Stochastic Environmental Research and Risk Assessment* **22**: 291–302.

Hipel KW, McLeod AI. 1994. *Time Series Modelling of Water Resources and Environmental Systems*. Elsevier: Amsterdam.

Hubert P. 2000. The segmentation procedure as a tool for discrete modeling of hydrometeorological regimes. *Stochastic Environmental Research and Risk Assessment* **14**: 297–304.

Kehagias A. 2004. A hidden Markov model segmentation procedure for hydrological and environmental time series. *Stochastic Environmental Research and Risk Assessment* **18**: 117–130.

Kehagias A, Fortin V. 2006. Time series segmentation with shifting means hidden Markov models. *Nonlinear Processes in Geophysics* **13**: 339–352.

Kehagias A, Nidelkou E, Petridis V. 2006. A dynamic programming segmentation procedure for hydrological and environmental time series. *Stochastic Environmental Research and Risk Assessment* **20**: 77–94.

Kehagias A, Petridis V, Nidelkou E. 2007. Reply by the authors to the letter by Aksoy, *et al*., *Stochastic Environmental Research and Risk Assessment* **21**: 451–455.

Mann ME, Bradley RS, Hughes MK. 1999. Northern hemisphere temperatures during the past millennium: inferences, uncertainties, and limitations. *Geophysical Research Letters* **26**: 759–762.

Mann ME, Gille E, Bradley RS, Hughes MK, Overpeck J, Keimig FT, Gross W. 2000. Global temperature patterns in past centuries: an interactive presentation. *Earth Interactions* **4**: pp.1–29.

McIntyre S, McKitrich R. 2005. Hockey sticks, principal components, and spurious significance. *Geophysical Research Letters* **32**: L03710, doi: 10·1029/2004GL021750.

Morettin PA, Mesquita AR, Rocha JGC. 1987. Rainfall at Fortaleza in Brazil revisited. *Time Series Analysis, Theory and Practice* **6**: 67–85.

Scheffe M. 1959. *The analysis of variance*. Wiley: New York.