Detection of multiple changes

J. Antoch
Univerzita Karlova v Praze

Němčičky 11. září 2012

PROBLEM

- Decision whether a sequence of random variables $X_1, \ldots, X_n$ observed sequentially in time is stationary.
- An alternative stochastic model assumes that there exist unknown time points (change points) such that the series is stationary in intervals between the change points while in neighboring intervals follows different stochastic models.

A change point analysis has usually two steps

- Decision whether the series is (hypotheses testing)
- Change points are estimated

We suggest applying a maximum type test derived in a natural way

1. Test statistic for the case with known and fixed change points is suggested
2. Values of this test statistic are calculated for all possible positions of the change points and the null hypothesis is rejected if at least one of these values, i.e. their maximum, is larger than a chosen critical value

BASIC ASSUMPTIONS

We assume :

- $X_1, \ldots, X_n$ are iid rv's
- X_i's are observed in time points $1, \ldots, n$
- maximal number of breaks (change points) d is fixed apriori
- changes consist in shifts of mean values of observed sequence ooccur in time points $0 < n_1 < \ldots < n_d < n$ ($n_0 = 0$, $n_{d+1} = n$)

MODEL AND TESTING PROCEDURE

Independent rv's $X_1, \ldots, X_n$ are observed.

$$H_0: \quad X_i = \mu + e_i, \quad i = 1, \ldots, n, \tag{1}$$

$$\begin{aligned} A_d: \quad & \text{there exist} \quad 1 \le n_1 < n_2 < \ldots < n_d < n \quad \text{satisfying} \\ & n_1 \ge \epsilon n, n_2 - n_1 \ge \epsilon n, \ldots, n - n_d \ge \epsilon n \quad \text{such that} \\ & X_i = \mu_1 + e_i, \qquad i = 1, \ldots, n_1, \\ & X_i = \mu_2 + e_i, \qquad i = n_1 + 1, \ldots, n_2, \\ & \vdots \\ & X_i = \mu_{d+1} + e_i, \quad i = n_d + 1, \ldots, n, \\ & \mu_j \ne \mu_{j+1} \text{ for at least one } j, \ 1 \le j \le d. \end{aligned} \tag{2}$$

- n_j and μ_j are unknown
- d is known and fixed in advance
- $\{e_i\}$ are iid, $E\, e_i = 0$, $E\, e_i^2 = 1$ and $E\, |e_i|^{2+\Delta} < \infty$ for some $\Delta > 0$
- neighboring breaks are situated in a distance $\ge \epsilon n$

NOTATION AND REMARKS

- $S(0) = 0$, $S(j) = \sum_{i=1}^{j} X_i$ for $j = 1, \ldots, n$.
- $\overline{X}(j, j') = \big(S(j') - S(j)\big)/(j' - j)$ for $j < j'$.
- $U(n_i, n_{i+1}) = \frac{1}{\sqrt{n_i n_{i+1}(n_{i+1} - n_i)}} \big(n_i\, S(n_{i+1}) - n_{i+1}\, S(n_i)\big), \; i = 1, \ldots, d$

Least squares estimates of $\mu_1, \ldots, \mu_{d+1}$ are

- $\widehat{\mu}_1 = \overline{X}(0, n_1)$
- $\widehat{\mu}_2 = \overline{X}(n_1, n_2)$
- $\ldots$
- $\widehat{\mu}_{d+1} = \overline{X}(n_d, n)$

Presence of gap parameter ϵ ensures that all segments contain enough observations to get "good" estimates of $\mu_1, \ldots, \mu_{d+1}$.

Assertion

Under H_0 and for fixed values $1 \leq n_1 < \ldots < n_d < n$ statistics $U(n_1, n_2), U(n_2, n_3), \ldots, U(n_d, n)$ are uncorrelated and asymptotically $N(0, 1)$ distributed rv's. Moreover,

$$\mathcal{L}\Big(U^2(n_1, n_2) + \ldots + U^2(n_d, n)\Big) \sim \chi^2_d \quad \text{as} \quad n \to \infty$$

MODEL AND TEST STATISTIC

Model : Independent rv's $X_1, \ldots, X_n$ are observed.

$$H_0 : \quad X_i = \mu + e_i, \quad i = 1, \ldots, n, \tag{3}$$

$$\begin{aligned} A_d : \quad & \text{there exist} \quad 1 \le n_1 < n_2 < \ldots < n_d < n \quad \text{satisfying} \\ & n_1 \ge \epsilon\, n, n_2 - n_1 \ge \epsilon\, n, \ldots, n - n_d \ge \epsilon\, n \quad \text{such that} \\ & X_i = \mu_1 + e_i, \qquad i = 1, \ldots, n_1, \\ & X_i = \mu_2 + e_i, \qquad i = n_1 + 1, \ldots, n_2, \\ & \quad \vdots \\ & X_i = \mu_{d+1} + e_i, \quad i = n_d + 1, \ldots, n, \\ & \mu_j \ne \mu_{j+1} \text{ for at least one } j,\ 1 \le j \le d. \end{aligned} \tag{4}$$

Test statistic

$$\chi^2_n(\epsilon) = \max_{\substack{1 \le n_1 < n_2 < \ldots < n_d < n \\ n_1 \ge \epsilon n, n_2 - n_1 \ge \epsilon n, \ldots, n - n_d \ge \epsilon n}} \left\{ U^2(n_1, n_2) + \ldots + U^2(n_d, n) \right\} \tag{5}$$

Remark : Suggested test statistic is equivalent with the log-likelihood ratio based test statistic under the assumption that $\{X_i\}$ are normally distributed.

DISASTER: COMPUTATIONAL COMPLEXITY

Attention please, despite test statistic (5) looks relatively simple, maximum is taken over enormously large number of terms if n increases:

ϵ	0		0.05	
n	$d = 2$	$d = 3$	$d = 2$	$d = 3$
1 000	501 501	167 668 501	362 526	85 974 801
3 000	4 504 501	4 509 005 501	3 255 076	2 309 764 401
5 000	12 507 501	20 858 342 501	9 037 626	10 682 674 001

Table 1. Numbers of different positions of change points for $n = 1\,000,\ 3\,000,\ 5\,000$, $d = 2,\ 3$ and $\epsilon = 0,\ 0.05$.

DISASTER: COMPUTATIONAL COMPLEXITY

Attention please, despite test statistic (5) looks relatively simple, maximum is taken over enormously large number of terms if n increases:

ϵ	0		0.05	
n	$d = 2$	$d = 3$	$d = 2$	$d = 3$
1 000	501 501	167 668 501	362 526	85 974 801
3 000	4 504 501	4 509 005 501	3 255 076	2 309 764 401
5 000	12 507 501	20 858 342 501	9 037 626	10 682 674 001

Table 2. Numbers of different positions of change points for $n = 1\,000,\ 3\,000,\ 5\,000$, $d = 2,\ 3$ and $\epsilon = 0,\ 0.05$.

Basic task revisited

How to obtain desired critical values when:

- Sample sizes n is large and d is small
- Sample sizes n is large and d is moderate or large

Algorithm for direct simulations

Algorithm

Input

Number of observations n, number of change points d, gap parameter ϵ, number of simulations NoS and a procedure enabling simulation of random variables from distribution $F(x)$.

Main loop

for $i = 1 : NoS$ **do**

Simulate iid rv's $X_1, \ldots, X_n$ with the distribution function $F(x)$.

Calculate value of the test statistic (5) and store it as TS_i.

end for

Output

Empirical distribution function calculated from $\{TS_i\}_{i=1}^{NoS}$.

Remarks

- Simplicity is main advantage.
- Computational complexity is major drawback
  Number of different positions of change points grows exponentially (their number is of the order n^d for $\epsilon \approx 0$)

IDEA

Solution of finding segments $[1, n_1]$, $[n_1+1, n_2]$,..., $[n_d+1, n]$ such that

$$Q^2 = \sum_{i=1}^{n_1} \left(X_i - \overline{X}(0, n_1)\right)^2 + \sum_{i=n_1+1}^{n_2} \left(X_i - \overline{X}(n_1, n_2)\right)^2 + \ldots + \sum_{i=n_d+1}^{n} \left(X_i - \overline{X}(n_d, n)\right)^2$$

is minimal for all possible

$0 = n_0 < n_1 < \ldots < n_d < n_{d+1} = n,\ n_i - n_{i-1} \geq n\epsilon, i = 1, \ldots, d+1,$
leads to the same solution as primary segmentation task we started with $\{n_i\}$ corresponding to the minimization of Q^2 are exactly the same as $\{n_i\}$ leading to the maximum of $\chi_n^2(\epsilon)$.

For this optimal split

$$\chi_n^2(\epsilon) = \sum_{i=1}^{n} \left(X_i - \overline{X}(0, n)\right)^2 - Q^2$$

Main advantage of reformulation (9) is that it offers a way how to proceed in evaluation of (9), and therefore also of (5), very effectively when both n and especially d are large.

REMEDY

One possible remedy consists in using Bellman's principle of optimality (better known as dynamic programming principle)

Assume

- We wish to split $X_1, \ldots, X_n$ so that sum of losses over $d+1$ segments is minimal
- Loss $q^1_{l,m}$ of a segment $X_l, \ldots, X_m$ is sum of squares of residuals[1]

$$q^1_{l,m} = \sum_{k=l}^{m} \left(X_k - \frac{1}{m-l+1} \sum_{h=l}^{m} X_h \right)^2 \tag{6}$$

- $q^j_{1,i}$ denotes minimal loss obtained by optimal partitioning of $X_1, \ldots, X_i$ into j segments

[1] As concerns calculation of losses $q^1_{l,m}$, many efficient and fast algorithms allowing both sequential and nonsequential calculation has been suggested in the literature.

Remedy

One possible remedy consists in using Bellman's principle of optimality (better known as dynamic programming principle)

Assume

- We wish to split $X_1, \ldots, X_n$ so that sum of losses over $d+1$ segments is minimal
- Loss $q^1_{l,m}$ of a segment $X_l, \ldots, X_m$ is sum of squares of residuals[1]

$$q^1_{l,m} = \sum_{k=l}^{m} \left(X_k - \frac{1}{m-l+1} \sum_{h=l}^{m} X_h \right)^2 \tag{6}$$

- $q^j_{1,i}$ denotes minimal loss obtained by optimal partitioning of $X_1, \ldots, X_i$ into j segments
- **Then**

$$q^j_{1,i} = \min_{j-1 \le k \le i-1} \left[q^{j-1}_{1,k} + q^1_{k+1,i} \right], \quad 2 \le j \le d+1,\ j \le i \le n \tag{7}$$

[1] As concerns calculation of losses $q^1_{l,m}$, many efficient and fast algorithms allowing both sequential and nonsequential calculation has been suggested in the literature.

Remedy cont.

Recall
Let $q^j_{1,i}$ denotes minimal loss obtained by optimal partitioning of $X_1, \ldots, X_i$ into j segments, then

$$q^j_{1,i} = \min_{j-1 \le k \le i-1} \left[q^{j-1}_{1,k} + q^1_{k+1,i}\right], \quad 2 \le j \le d+1,\ j \le i \le n$$

Modification If we, moreover, impose the condition that each segment contains at least $\lceil n\,\epsilon \rceil \ge 1$ observations, then

$$q^j_{1,i} = \min_{(j-1)\cdot\lceil n\,\epsilon\rceil \le k \le i - \lceil n\,\epsilon \rceil} \left[q^{j-1}_{1,k} + q^1_{k+1,i}\right], \quad 2 \le j \le d+1,\ j \cdot \lceil n\,\epsilon \rceil \le i \le n$$

- Usually all $n(n-1)/2$ values $q^1_{l,m},\ 1 \le l \le m \le n$, are calculated at the setup phase and kept throughout the calculations. When setup phase is complete then recursion (7) is applied.
- Keeping the values $q^1_{l,m}$ in the memory during all calculations requires $n(n-1)/2$ unique storage places, so that with n increasing the internal RAM of the computer is very soon exhausted
- If external peripherals are used then computations slow down dramatically and memory of the computer becomes the greatest barrier.

DPP : What we must calculate

# of segments	1	2	3	4	5
goal to find	$q^1_{1,1}$				
candidates	$q^1_{1,1}$				
goal to find	$q^1_{1,2}$	$q^2_{1,2}$			
candidates		$q^1_{1,1}+q^1_{2,2}$			
goal to find	$q^1_{1,3}$	$q^2_{1,3}$	$q^3_{1,3}$		
candidates	$q^1_{1,3}$	$q^1_{1,1}+q^1_{2,3}$ $q^1_{1,2}+q^1_{3,3}$	$q^2_{1,2}+q^1_{3,3}$		
goal to find	$q^1_{1,4}$	$q^2_{1,4}$	$q^3_{1,4}$	$q^4_{1,4}$	
candidates	$q^1_{1,4}$	$q^1_{1,1}+q^1_{2,4}$ $q^1_{1,2}+q^1_{3,4}$ $q^1_{1,3}+q^1_{4,4}$	$q^2_{1,2}+q^1_{3,4}$ $q^2_{1,3}+q^1_{4,4}$	$q^3_{1,3}+q^1_{4,4}$	
goal to find	$q^1_{1,5}$	$q^2_{1,5}$	$q^3_{1,5}$	$q^4_{1,5}$	$q^5_{1,5}$
candidates	$q^1_{1,5}$	$q^1_{1,1}+q^1_{2,5}$ $q^1_{1,2}+q^1_{3,5}$ $q^1_{1,3}+q^1_{4,5}$ $q^1_{1,4}+q^1_{5,5}$	$q^2_{1,2}+q^1_{3,5}$ $q^2_{1,3}+q^1_{4,5}$ $q^2_{1,4}+q^1_{5,5}$	$q^3_{1,3}+q^1_{4,5}$ $q^3_{1,4}+q^1_{5,5}$	$q^4_{1,4}+q^1_{5,5}$
goal to find	$q^1_{1,6}$	$q^2_{1,6}$	$q^3_{1,6}$	$q^4_{1,6}$	$q^5_{1,6}$
candidates	$q^1_{1,6}$	$q^1_{1,1}+q^1_{2,6}$ $q^1_{1,2}+q^1_{3,6}$ $q^1_{1,3}+q^1_{4,6}$ $q^1_{1,4}+\boxed{q^1_{5,6}}$ $q^1_{1,5}+q^1_{6,6}$	$q^2_{1,2}+q^1_{3,6}$ $q^2_{1,3}+q^1_{4,6}$ $q^2_{1,4}+\boxed{q^1_{5,6}}$ $q^2_{1,5}+q^1_{6,6}$	$q^3_{1,3}+q^1_{4,6}$ $q^3_{1,4}+\boxed{q^1_{5,6}}$ $q^3_{1,5}+q^1_{6,6}$	$q^4_{1,4}+\boxed{q^1_{5,6}}$ $q^4_{1,5}+q^1_{6,6}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
goal to find	$q^1_{1,n}$	$q^2_{1,n}$	$q^3_{1,n}$	$q^4_{1,n}$	$q^5_{1,n}$
candidates	$q^1_{1,n}$	$q^1_{1,1}+q^1_{2,n}$ $q^1_{1,2}+q^1_{3,n}$ $q^1_{1,3}+q^1_{4,n}$ $q^1_{1,4}+q^1_{5,n}$ $\vdots$ $q^1_{1,n-2}+q^1_{n-1,n}$ $q^1_{1,n-1}+q^1_{n,n}$	$q^2_{1,2}+q^1_{3,n}$ $q^2_{1,3}+q^1_{4,n}$ $q^2_{1,4}+q^1_{5,n}$ $\vdots$ $q^2_{1,n-2}+q^1_{n-1,n}$ $q^2_{1,n-1}+q^1_{n,n}$	$q^3_{1,3}+q^1_{4,n}$ $q^3_{1,4}+q^1_{5,n}$ $\vdots$ $q^3_{1,n-2}+q^1_{n-1,n}$ $q^3_{1,n-1}+q^1_{n,n}$	$q^4_{1,4}+q^1_{5,n}$ $\vdots$ $q^4_{1,n-2}+q^1_{n-1,n}$ $q^4_{1,n-1}+q^1_{n,n}$

Table 3: All candidates that must be evaluated when DPP is used for optimal splitting of $X_1, \ldots, X_n$ into five segments.

DPP : What we must calculate – Detail

# of segments	1	2	3	4	5
goal to find	$q^1_{1,5}$	$q^2_{1,5}$	$q^3_{1,5}$	$q^4_{1,5}$	$q^5_{1,5}$
candidates	$q^1_{1,5}$	$q^1_{1,1}+q^1_{2,5}$ $q^1_{1,2}+q^1_{3,5}$ $q^1_{1,3}+q^1_{4,5}$ $q^1_{1,4}+q^1_{5,5}$	$q^2_{1,2}+q^1_{3,5}$ $q^2_{1,3}+q^1_{4,5}$ $q^2_{1,4}+q^1_{5,5}$	$q^3_{1,3}+q^1_{4,5}$ $q^3_{1,4}+q^1_{5,5}$	$q^4_{1,4}+q^1_{5,5}$
goal to find	$q^1_{1,6}$	$q^2_{1,6}$	$q^3_{1,6}$	$q^4_{1,6}$	$q^5_{1,6}$
candidates	$q^1_{1,6}$	$q^1_{1,1}+q^1_{2,6}$ $q^1_{1,2}+q^1_{3,6}$ $q^1_{1,3}+q^1_{4,6}$ $q^1_{1,4}+\boxed{q^1_{5,6}}$ $q^1_{1,5}+q^1_{6,6}$	$q^2_{1,2}+q^1_{3,6}$ $q^2_{1,3}+q^1_{4,6}$ $q^2_{1,4}+\boxed{q^1_{5,6}}$ $q^2_{1,5}+q^1_{6,6}$	$q^3_{1,3}+q^1_{4,6}$ $q^3_{1,4}+\boxed{q^1_{5,6}}$ $q^3_{1,5}+q^1_{6,6}$	$q^4_{1,4}+\boxed{q^1_{5,6}}$ $q^4_{1,5}+q^1_{6,6}$

Table 4: Selected candidates that must be evaluated when DPP is used for optimal splitting of $X_1, \ldots, X_n$ into five segments.

ORDER OF EVALUATION

- It is not necessary to keep all $q^1_{l,m}$ in memory of computer all over the time.
- Instead, necessary values are:
  - Calculated only once.
  - Kept in the operational memory for a short time only.
  - Order of their evaluation must be changed.

[2] This question has a parallel with multiplication of matrices.

ORDER OF EVALUATION

- It is not necessary to keep all $q^1_{l,m}$ in memory of computer all over the time.
- Instead, necessary values are:
  - Calculated only once.
  - Kept in the operational memory for a short time only.
  - Order of their evaluation must be changed.

First important question is order of evaluation when searching for optimal splits[2].

One possible order :
Calculate, step by step, $q^1_{1,1}, q^1_{1,2}, q^2_{1,2}, q^1_{1,3}, q^2_{1,3}, q^3_{1,3}, \ldots, q^1_{1,n}, \ldots, q^{d+1}_{1,n}$.

Advantages :

- Speed.
- Memory saving.
- Optimal splits of all subseries $X_1, \ldots, X_k,\ 1 \le k \le i$, into $1, \ldots, d+1$ subsegments, when splitting of a sequence $X_1, \ldots, X_i$ has been finished are available.

[2] This question has a parallel with multiplication of matrices.

ORDER OF AUXILIARY CALCULATIONS

Second key question is an order of auxiliary calculations connected with the evaluation of $q^1_{k,l}$.

- Terms $q^1_{k,i}$, $1 \leq k \leq i$, when both i and k were fixed, are needed only when evaluating $q^2_{1,i}, \ldots, q^{d+1}_{1,i}$.
- It is not necessary to calculate the values $q^1_{k,i}$ always from the scratch, because knowledge of $q^1_{k,i-1}$ can be effectively used.
- When searching for $q^j_{1,i}$, $1 \leq j \leq \min\{i, d+1\}$, we must know $q^1_{1,1}$, $q^1_{1,2}$, $q^2_{1,2}$, $q^1_{1,3}$, $q^2_{1,3}$, $q^3_{1,3}$, $\ldots$, $q^1_{1,i-1}$, $\ldots$, $q^{\min\{i-1,d+1\}}_{1,i-1}$.

goal to find	$q^1_{1,6}$	$q^2_{1,6}$	$q^3_{1,6}$	$q^4_{1,6}$	$q^5_{1,6}$
candidates	$q^1_{1,6}$	$q^1_{1,1}+q^1_{2,6}$			
		$q^1_{1,2}+q^1_{3,6}$	$q^2_{1,2}+q^1_{3,6}$		
		$q^1_{1,3}+q^1_{4,6}$	$q^2_{1,3}+q^1_{4,6}$	$q^3_{1,3}+q^1_{4,6}$	
		$q^1_{1,4}+\boxed{q^1_{5,6}}$	$q^2_{1,4}+\boxed{q^1_{5,6}}$	$q^3_{1,4}+\boxed{q^1_{5,6}}$	$q^4_{1,4}+\boxed{q^1_{5,6}}$
		$q^1_{1,5}+q^1_{6,6}$	$q^2_{1,5}+q^1_{6,6}$	$q^3_{1,5}+q^1_{6,6}$	$q^4_{1,5}+q^1_{6,6}$

Basic idea revisited

Basic idea is simple. We do not evaluate all values $q^1_{k,i}$ at once but during a search for $q^1_{1,i},\dots,q^{d+1}_{1,i}$ we compute and store only the values of the loss function for every subsequence ending at the ith place of the sequence $X_1,\dots,X_n$ and compare succesively new candidates with temporary optimal solution.

As a result we do not have the final values all at once but have a lot of running values during the calculation.

Basic idea revisited

Basic idea is simple. We do not evaluate all values $q^1_{k,i}$ at once but during a search for $q^1_{1,i},\ldots,q^{d+1}_{1,i}$ we compute and store only the values of the loss function for every subsequence ending at the ith place of the sequence $X_1,\ldots,X_n$ and compare succesively new candidates with temporary optimal solution.
As a result we do not have the final values all at once but have a lot of running values during the calculation.

Memory requirements

1. For storing values of losses $q^1_{1,1}$, $q^1_{1,2}$, $q^2_{1,2}$, $q^1_{1,3}$, $q^2_{1,3}$, $q^3_{1,3}$, $\ldots$, $q^1_{1,n}$, $\ldots$, $q^{d+1}_{1,n}$ matrix $n \times (d+1)$ is needed.
2. Matrix of the same size is needed for storing positions of optimal change points. This can be kept on the hard disk and updated time by time.
   On the ij^{th} place is beginning of j^{th} segment when $X_1,\ldots,X_i$ is optimally split into j segments $\Longrightarrow$ backtracking.
3. Several vectors of different sizes are needed for keeping $X_1,\ldots,X_n$ and temporary values from auxiliary calculations.

	χ^2_n		$\chi^2_n n, \epsilon,\ \epsilon = 0.05$		$\chi^2_n n, \epsilon,\ \epsilon = 0.10$	
n	$d = 2$	$d = 3$	$d = 2$	$d = 3$	$d = 2$	$d = 3$
1 000	0.16	11.55	0.04	5.90	0.03	2.39
3 000	0.59	438.59	0.43	203.22	0.29	79.01
5 000	1.68	2 301.07	1.20	1 166.46	0.83	454.38
10 000	6.67	22 743.85	4.84	11 478.48	3.27	4 734.51

Table 5: CPU times (in hours) needed to calculate critical values for statistics χ^2_n and $\chi^2_{n,\epsilon}$ using direct simulations and 10^4 repetitions.

			d					
n	statistic	ϵ	1	2	3	5	7	9
1 000	χ^2_n		7.86	8.72	10.34	11.94	13.60	14.15
	$\chi^2_{n,\epsilon}$	0.05	7.25	7.95	9.01	9.78	10.15	10.39
	$\chi^2_{n,\epsilon}$	0.10	6.45	6.95	7.50	7.76	7.84	7.84
3 000	χ^2_n		106.95	114.71	128.30	141.87	154.66	161.26
	$\chi^2_{n,\epsilon}$	0.05	99.42	105.55	114.01	121.22	125.95	128.00
	$\chi^2_{n,\epsilon}$	0.10	92.58	96.74	101.46	103.96	104.37	104.47
5 000	χ^2_n		412.85	433.95	472.18	508.09	543.89	564.90
	$\chi^2_{n,\epsilon}$	0.05	392.09	409.10	432.37	450.76	465.17	470.92
	$\chi^2_{n,\epsilon}$	0.10	369.49	381.42	394.92	400.48	403.13	403.21
10 000	χ^2_n		2 780.37	2 871.46	3 028.09	3 175.18	3 320.66	3 395.20
	$\chi^2_{n,\epsilon}$	0.05	2 687.64	2 756.48	2 853.49	2 934.31	2 999.30	3 007.64
	$\chi^2_{n,\epsilon}$	0.10	2 586.57	2 643.44	2 684.88	2 710.86	2 714.95	2 722.94

Table 6. CPU times (in hours) needed to calculate critical values for statistics χ^2_n and $\chi^2_{n,\epsilon}$ using modified dynamic programming principle and 10^4 repetitions.

Input
Series $X_1, \dots, X_n$ to be split.
Function *LOSS* calculating loss $q^1_{l,m}$ from the observations $\big(X_l, \dots, X_m\big)$.

Setup
for $i = 1 : n$ **do**
  for $j = 1 : n$ **do**
    if $j < i$ **then**
      $matQ(i,j) = +\infty$
    else
      $matQ(i,j) = LOSS\big(X_i, \dots, X_j\big)$
    end if
  end for
  $matQ(i,1) = matQ(1,i)$
  $matR(i,1) = 1$
end for

Main loop
for $i = 2 : n$ **do, for** $j = 2 : i$ **do, for** $k = j : i$ **do**
  $guess = matQ(k-1, j-1) + matQ(k,i)$
  if $guess <= matQ(i,j)$ **then**
    $matQ(i,j) = guess$
    $matR(i,j) = k$
  end if
end for, end for, end for

Output *matQ* and *matR*, where:
$matQ(i,j) = q^j_{1,i}$ if $1 \le j \le i \le n$ and $matQ(i,j) = q^1_{i,j}$ if $1 \le i < j \le n$
$matR(i,j)$ contains beginning of the j-th segment when sequence $X_1, \dots, X_i$ is optimally split into j segments

Classical dynamic programming principle applied to our segmentation problem.

1: **Main loop**
2: **for** $i = 2 : n$ **do, for** $j = 2 : i$ **do, for** $k = j : i$ **do**
3:     $guess = matQ(k-1, j-1) + matQ(k, i)$
4:     **if** $guess <= matQ(i, j)$ **then**
5:         $matQ(i, j) = guess$
6:         $matR(i, j) = k$
7:     **end if**
8: **end for, end for, end for**

9: **Output** *matQ* and *matR*, where:
10: $matQ(i, j) = q^{j}_{1,i}$ if $1 \leq j \leq i \leq n$ and $matQ(i, j) = q^{1}_{i,j}$ if $1 \leq i < j \leq n$
11: $matR(i, j)$ contains beginning of the j-th segment when sequence $X_1, \ldots, X_i$ is optimally split into j segments

Remark: Significant speed up is achieved if lines 9 and 10 of the setup (calculation of the $LOSS(X_i, \ldots, X_j)$ are omited and **Main loop** replaced by:

12: **Modified main loop**
13: **for** $i = 2 : n$ **do**
14:     **for** $k = i : -1 : 2$ **do**
15:         $jnk = LOSS(X_k, \ldots, X_i),\ matQ(k, i) = jnk$
16:         **for** $j = 2 : k$ **do**
17:             $guess = matQ(k-1, j-1) + jnk$
18:             **if** $guess <= matQ(i, j)$ **then**
19:                 $matQ(i, j) = guess$
20:                 $matR(i, j) = k$
21:             **end if**
22:         **end for**
23:     **end for**
24: **end for**

Modified dynamic programming principle applied to our segmentation problem.

Input

Series $X_1, \ldots, X_n$ to be split into $d+1$ segments.

Function *LOSS* calculating loss $q^1_{l,m}$ from the observations $X_l, \ldots, X_m$.

Setup

```
matQ = 0, matR = 0
for i = 1 : n do
   for j = 1 : min(i, d + 1) do
      if j < i then
         matQ(i, j) = +∞
      end if
   end for
   matQ(i, 1) = LOSS(X_1, ..., X_i)
   matR(i, 1) = 1
end for
```

Main loop

```
for i = 2 : n do
   for k = i : −1 : 2 do
      jnk = LOSS(X_k, ..., X_i)
      for j = 1 : min(k, d + 1) do
         guess = matQ(k − 1, j − 1) + jnk
         if guess <= matQ(i, j) then
            matQ(i, j) = guess
            matR(i, j) = k
         end if
      end for
   end for
end for
```

Output *matQ* and *matR*, where:

- $matQ(i,j) = q^j_{1,i},\ 1 \le i \le n \ \&\ 1 \le j \le \min(i, d+1)$
- $matR(i,j),\ 1 \le i \le n \ \&\ 1 \le j \le \min(i, d+1)$, contains beginning of the j-th segment when a series $X_1, \ldots, X_i$ is optimally split into j segments.