Cluster Analysis of Discussions Change Dynamics on Twitter about War in Ukraine

Sofiia Mainych [1], Alina Bulhakova [1] and Victoria Vysotska [1,2]

[1] *Lviv Polytechnic National University, S. Bandera Street, 12, Lviv, 79013, Ukraine*
[2] *Osnabrück University, Friedrich-Janssen-Str. 1, Osnabrück, 49076, Germany*

Abstract

The article analyzes the dynamics of changes in the pace and directions of discussion of the russian-Ukrainian war on Twitter based on a set of thematic hashtags from February 23, 2022. A cluster analysis of popular tweets was conducted. The object of the study is the analysis of the discussion of the russian-Ukrainian war on Twitter. Data on the number of tweets and hashtags related to this topic of discussion from the beginning of the war were used for the analysis. The largest share of all hashtags was #Ukraine, followed by #NATO and #UkrainerussiaWar, respectively. At the beginning of the war (the first week), information about the situation in Ukraine became a kind of explosion in the media space of the world. The number of comments, posts, "tweets" on this topic reached a record high of 4,000,000. After a week, this trend changed dramatically - the number of discussions decreased significantly. This is due in particular to the fact that the crisis stage has passed (there has been a lot of information, it is no longer so "interesting" for the world). Although it is also worth saying that this decrease continued to a specific mark. For 50 days from the start of the invasion, the number of publications decreased from 4,000 thousand to 500 thousand. At the level of 500 thousand, the number is maintained until now (with insignificant deviations during certain military events). We can assume that under constant circumstances, without significant changes, this trend will take on a downward trend. At the same time, if the circumstances change, it is impossible to make objective assumptions. Another interesting conclusion can be that during the cluster analysis, it was found that the world focuses more on Ukraine, as a victim of the war, than on russia, as the aggressor. Also, a large share of foreigners supports the initiative to help Ukraine (in particular, the issue of joining NATO). On the contrary, it should not be forgotten that the assumptions "support" - "do not support" are not objective, since people tagging the NATO hashtag can also write negative posts. A fair statement would be the following: foreigners are concerned about the situation in Ukraine and its request for protection from NATO.

Keywords

Хештег, пост, Twitter, #Ukraine, #NATO, #UkrainerussiaWar, кластерний аналіз

1. Introduction

Continuous consumption of information content from the Internet over the last year has already become a new daily habit for most Ukrainians, especially from YouTube and various social networks such as Telegram, Twitter, Facebook and others [1].

The Kyiv International Institute of Sociology conducted a survey commissioned by the OPORA Civic Network from May 19 to 24, 2022, among 2,009 adult citizens of Ukraine who at that time were in the territory of Ukraine controlled by the Ukrainian authorities [2]. The results showed the following distribution of popularity of sources of operational news among surveyed Ukrainians: 76.6% - social networks (77.9% of men and 75.5% of women), 66.7% – television (predominance among women - 70.4%), 61.2% – Internet sources without social networks (predominance among men – 63.5%), 28.4%

COLINS-2023: 7th International Conference on Computational Linguistics and Intelligent Systems, April 20–21, 2023, Kharkiv, Ukraine
EMAIL: sofiia.mainych.sa.2020@lpnu.ua (S. Mainych); alina.bulhakova.sa.2020@lpnu.ua (A. Bulhakova); victoria.a.vysotska@lpnu.ua (V. Vysotska)
ORCID: 0000-0002-6449-5386 (S. Mainych); 0000-0002-6726-8859 (A. Bulhakova); 0000-0001-6417-3689 (V. Vysotska)
© 2023 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-3396/paper39.pdf

– radio and 15.7% – print mass media. Young people generally get their news from social media (92%), compared to the older, middle-aged generation (64%). The latter also prefer the Internet without social networks (60%). Older people prefer television (78%), radio (36%) and sometimes print media (20%).

Rural residents are more likely to watch television (71.3% versus 64.4% in cities) and read print media (21.5% versus 12.7%). On the other hand, townspeople more often consume news via the Internet (64% vs. 55.6% in the village), social networks (79.2% vs. 71.4%), and radio (28.5% vs. 28.2%) [3].

In Western Ukraine, more people watch television (73.5%), read print media (23.2%) and listen to the radio (34.6%). On the other hand, in eastern Ukraine, internet media are read the most (63.2%). Social networks as a source of news are most actively used in southern Ukraine (77.8%) [1-3].

The main question to be investigated is the verification of the trend of interest regarding the situation in Ukraine in the world (by time). First of all, we can put forward the following hypothesis: "At the beginning of the war (the first few days-weeks), the number of publications (tweets) on the Internet increased rapidly, after that it remained at approximately the same level for a certain time, later (let's say a month later) interest is still higher, than it was before the reference point (the beginning of the war), and yet it has already significantly decreased compared to the maximum value." The presented hypothesis is based on completely logical ideas about the state of the media when riots, wars, and coups suddenly begin. The news, which originated in one media space, spreads around the world with extraordinary speed. People are interested in knowing details and predictions, both true and fake. Later, when the state of affect passes and the world is already better informed about the situation, this interest gradually subsides. But since this hypothesis is still purely empirical, a study was conducted, the results of which are presented in this report. The purpose of this work is to verify the proposed hypothesis using data analysis from the selected dataset on the topic "Discussion of the russian-Ukrainian war on Twitter".

2. Related works

After the escalation of russia's war with Ukraine on February 24, 2022, the top 3 social networks, where Ukrainians get information, changed somewhat [1-2]. Yes, YouTube remains the most stable in the lives of Ukrainians - year after year, starting in 2019, it ranks second in Ukraine as a source of news. In 2019-2021, the ranking of social networks remained almost unchanged, with the exception of 2019, when the third most popular messenger was Viber. However, after the start of the full-scale invasion, Facebook for the first time gave up its positions to Telegram and moved all the way to third place. Thus, among 76.6% of citizens who use social networks as a source of information, 66% choose Telegram, 61% choose YouTube, and another 58% choose Facebook. Less popular news sources for Ukrainians during the war were Viber (48%), Instagram (29%), TikTok (19.5%) and Twitter (8.9%).

The war exposed russian propaganda and changed the perception of Ukraine in the world, says Nina Yankovich, vice-president of the "Center for Information Resilience" project [3]. russian disinformation has played a crucial role in the war against Ukraine since 2014, although this role has changed somewhat. Now we have become more aware of russia's intentions in Ukraine. Starting from 2022, the russian information war in Ukraine is actually failing. Ukraine has shown the world its extraordinary resilience, amazing creativity and authenticity in communications. The aggressor, russia, has become less sophisticated in disseminating its disinformation, as social networks have learned to counter bots and trolls. Therefore, there is less of such an artificial increase in informational emissions now. When the full-scale invasion first began, people had a lot of questions about Ukraine. American publications covered this conflict qualitatively: almost all of them now have correspondents in Ukraine. Ukraine's constant presence in the international media has been incredibly important in filling the gaps that many Americans have. There is a false impression that fighting disinformation means stifling freedom of speech, as imposing restrictions on expression. This is not always true.

Social media is an important public platform for many democracies and autocracies around the world, and without it, people resisting authoritarian regimes would have no voice [3]. Therefore, it is very important that these multi-billion dollar corporations invest in people who actually understand the context of what is happening in the country and develop regional offices. If Twitter, for example, claims to be a voice for free speech and democracy, then it should support the democratic side of this conflict, which is fighting for freedom. Also, if we're talking about content sharing and freedom of speech, social

networks should definitely allow users to share and document what they experience in their daily lives. And any platform that doesn't, in my opinion, violates freedom of expression.

There is a huge interest in what is happening in Ukraine, and authentic content from "real people" is one of the most effective ways to fight disinformation [1-3].

Many Ukrainians are rapidly gaining hundreds of thousands of followers by telling stories about Ukraine. There is a huge interest in what is happening in Ukraine, and authentic content from "real people" is one of the most effective ways to combat disinformation. Every well-told story is the perfect antidote to propaganda. Because it's not just fact-checking or myth-busting, it's the real experience of ordinary people. And it allows people living in the West with their simple, predictable lives to put themselves in the shoes of people living in Ukraine, and it's extremely powerful. It is now much more difficult to create a fake account and impersonate someone. Social networks have found ways to recognize such accounts. And Americans have also become more savvy: they show this behavior more often. It is still an actual threat. Aggressor russia finds ways to manipulate our discourse on the Internet. But this threat is not as straightforward as before.

American businessman, founder of SpaceX and CEO of Tesla, Elon Musk, reported on May 18 that, according to his calculations, 50% of Twitter accounts are bots [4-7]. Twitter's new identity verification system has led to a boom in fakes. Perhaps Elon Musk is all too aware of how Twitter can influence politics, politicians and political debates around the world, and is taking advantage of it. Previously, Twitter said that only 5% of accounts on the social network are fake. After that, Musk suspended the deal to buy the company. He also stated that "the deal cannot move forward" unless Twitter provides evidence that the number of bots does not exceed 5% of daily active users per month during the quarter [8]. At a conference in Miami, the businessman said that fake users make up at least 20% of all Twitter accounts, and suggested that the figure could be as high as 90%. He also revealed that his team will conduct its own audit of 100 random Twitter accounts to identify the number of fake ones. After these statements, according to Musk, Twitter lawyers accused him of violating the rules of non-disclosure of information (NDA) [9]. The head of Twitter's security department noted that the new policy is designed to "help protect discussions on Twitter, starting with a focus on the russian invasion of Ukraine." On May 19, the Twitter company introduced a new policy "to combat misinformation during a crisis", which provides that false content about the war in Ukraine will be marked with special marks [10].

Twitter may add a warning to a post if it contains:

- false information that incorrectly characterizes the conditions on the ground during the development of the conflict;
- false statements about the use of force, encroachment on territorial sovereignty or the use of weapons;
- knowingly false or misleading statements about war crimes or mass atrocities against certain population groups;
- false information about the reaction of the international community, sanctions, defensive actions or humanitarian operations.

Under the new policy, the social network will not recommend or share tweets that have been identified as false. Twitter will also limit the spread of such information: users will not be able to like, retweet or share content that violates the new rules. This change is part of a broader promotion of truthful information during a conflict or crisis following the Soviet invasion of Ukraine.

Twitter's head of security and corporate ethics, Yoel Roth, noted that the new policy is designed to help protect discussions on Twitter, starting with a focus on the russian invasion of Ukraine [11]. These rules will focus on potentially dangerous misinformation about alleged war crimes, armed conflicts, humanitarian crises, etc.

We will remind, on February 26, the Twitter social network blocked the possibility of registering accounts in russia [12]. For users from Ukraine and russia, the social network has suspended advertising and some recommendations for tweets from people they are not following. This was done to reduce the spread of offensive content. On February 28, Twitter began flagging russian state media. Then, in four days, the social network recorded more than 45,000 tweets per day with links to russian state media, but as of March 11, Twitter saw a drop in impressions. On March 11, the social network began flagging the accounts of state media in Belarus [13]. On March 17, the social network deleted or marked as unreliable more than 50,000 messages containing false information about russia's war against Ukraine

[14-21]. That is why this social network itself was chosen to analyse the discussion of the russian-Ukrainian war on Twitter (the number of hashtags related to this topic from February 23, 2022 - from the beginning of the full-scale war of russia against Ukraine).

3. Methods and materials

The following datasets were used for this work:

- general statistics of the daily number of tweets during 23.02-22.09: https://github.com/alexdrk14/RussiaUkraineWar/blob/main/data/daily_stats.csv (Fig. 1);
- statistics of the use of various hashtags related to the russian-Ukrainian war of 2022: https://github.com/alexdrk14/RussiaUkraineWar/blob/main/data/daily_hashtags.csv.

We will process the data in several stages, in particular: graphic representation of trends, calculations of correlations (relationships between data) and cluster analysis. First of all, in any work with data, it is necessary to properly organize them for further processing. The first type of presentation of the received information is a tabular presentation. In addition to tables, graphs are a widely applicable way of presenting data. With their help, you can visually see certain patterns, establish the type of connection, and even roughly predict the behavior of a particular process. The main difference from a table is that in a graph we have to clearly specify some relationship that we want to investigate, unlike a table, which helps to quickly navigate a large volume of data.

day	daily_tweets	Ukraine	Russia	StandWithUkraine	Putin	UkraineRussiaWar	NATO	StopRussia	Russian	StopPutin	ukraine
Feb-23	22740	3300	3058	440	2224	0	470	2	318	15	485
Feb-24	2352003	812373	314461	179946	239680	1253	59184	6612	40905	13797	39731
Feb-25	3997584	2206770	659808	647388	338973	4846	151971	32196	65886	123700	100248
Feb-26	4027485	2345772	585295	660822	310528	89957	105232	173115	119131	163692	131250
Feb-27	3762020	2195563	549506	560680	314656	257406	86367	121821	154006	120799	160405
Feb-28	3167455	1645522	423865	329716	226289	487234	69978	217143	71293	210134	96440
Mar-1	3050607	1556274	402347	346922	214579	510488	67966	236950	57560	236860	75501
Mar-2	2628486	1304224	415034	326594	199956	457085	49740	82810	58795	86955	80677
Mar-3	2892043	1126578	356885	237376	247517	276078	48362	53799	55198	122478	72268
Mar-4	2396854	1023753	287073	156840	406355	163207	75941	116743	52163	304824	73001
Mar-5	1934266	876501	275012	164737	191713	103417	65639	109180	45254	113872	63313
Mar-6	1982330	924149	297090	138199	240882	96683	54535	93995	56304	128026	49593
Mar-7	1838330	943847	284740	116011	129206	60545	40650	81573	60399	49389	44639
Mar-8	1466125	741920	242076	142642	107718	60269	45967	36716	34165	37992	38481
Mar-9	1453713	675299	214942	159348	87089	51512	52386	84527	36490	42955	34941
Mar-10	1067689	507188	201750	125502	77234	43790	26118	52743	26670	29857	27724
Mar-11	1107978	527414	189002	138131	78878	64617	35548	62269	36429	42088	32531
Mar-12	1085582	533857	168328	131937	68742	76859	29852	55559	30708	28334	29453
Mar-13	1138400	597592	166629	135565	69043	76419	37863	58987	31563	55003	24215
Mar-14	995092	512837	167263	136253	62583	72517	32911	27779	23443	32772	26284
Mar-15	995767	536553	178862	135061	90619	59864	39224	34303	31141	30608	24526

Figure 1: Dataset loading result

Data for table presentation: number of tweets per day (9 subtables in Fig. 2).

Date	Amount of tweets	Date	Amount of tweets	Date	Amount of tweets	Date	Amount of tweets	Date	Amount of tweets	Date	Amount of tweets	Date	Amount of tweets	Date	Amount of tweets	Date	Amount of tweets
Feb-23	22740	Mar-19	825088	Apr-12	392054	May-6	273280	May-30	206093	Jun-23	179676	Jul-17	126528	Aug-10	105378	Sep-3	133991
Feb-24	2352003	Mar-20	875569	Apr-13	440644	May-7	279232	May-31	186994	Jun-24	162005	Jul-18	126866	Aug-11	110189	Sep-4	139605
Feb-25	3997584	Mar-21	884823	Apr-14	409549	May-8	351429	Jun-1	194058	Jun-25	137684	Jul-19	120272	Aug-12	89380	Sep-5	156857
Feb-26	4027485	Mar-22	748688	Apr-15	377294	May-9	371957	Jun-2	166972	Jun-26	182568	Jul-20	135559	Aug-13	94308	Sep-6	160933
Feb-27	3762020	Mar-23	698314	Apr-16	344037	May-10	272020	Jun-3	188766	Jun-27	216178	Jul-21	131626	Aug-14	93538	Sep-7	163553
Feb-28	3167455	Mar-24	770915	Apr-17	359673	May-11	254718	Jun-4	168607	Jun-28	246091	Jul-22	124005	Aug-15	113157	Sep-8	168072
Mar-1	3050607	Mar-25	612233	Apr-18	396723	May-12	267200	Jun-5	200731	Jun-29	240913	Jul-23	119036	Aug-16	114333	Sep-9	155944
Mar-2	2628486	Mar-26	608478	Apr-19	454303	May-13	250351	Jun-6	170903	Jun-30	169089	Jul-24	106318	Aug-17	115054	Sep-10	199259
Mar-3	2892043	Mar-27	529504	Apr-20	355061	May-14	259501	Jun-7	175930	Jul-1	163650	Jul-25	110939	Aug-18	123191	Sep-11	253597
Mar-4	2396854	Mar-28	487217	Apr-21	331034	May-15	299676	Jun-8	183332	Jul-2	145615	Jul-26	108822	Aug-19	127020	Sep-12	215892
Mar-5	1934266	Mar-29	419553	Apr-22	305095	May-16	284562	Jun-9	182514	Jul-3	135203	Jul-27	112776	Aug-20	129536	Sep-13	196044
Mar-6	1982330	Mar-30	421240	Apr-23	334252	May-17	247941	Jun-10	176658	Jul-4	139255	Jul-28	147314	Aug-21	133841	Sep-14	209552
Mar-7	1838330	Mar-31	442421	Apr-24	321408	May-18	238620	Jun-11	176318	Jul-5	141492	Jul-29	179954	Aug-22	130967	Sep-15	178638
Mar-8	1466125	Apr-1	412778	Apr-25	303711	May-19	163192	Jun-12	175696	Jul-6	125128	Jul-30	141515	Aug-23	131378	Sep-16	179858
Mar-9	1453713	Apr-2	454576	Apr-26	343281	May-20	226020	Jun-13	164759	Jul-7	127728	Jul-31	135196	Aug-24	247779	Sep-17	188446
Mar-10	1067689	Apr-3	721685	Apr-27	356886	May-21	201747	Jun-14	128390	Jul-8	133029	Aug-1	122608	Aug-25	157018	Sep-18	167805
Mar-11	1107978	Apr-4	618069	Apr-28	355179	May-22	189271	Jun-15	169412	Jul-9	119984	Aug-2	131833	Aug-26	137166	Sep-19	157851
Mar-12	1085582	Apr-5	613541	Apr-29	305894	May-23	212091	Jun-16	200478	Jul-10	135665	Aug-3	121159	Aug-27	108005	Sep-20	178182
Mar-13	1138400	Apr-6	506522	Apr-30	296992	May-24	199731	Jun-17	178415	Jul-11	145558	Aug-4	121857	Aug-28	102978	Sep-21	398435
Mar-14	995092	Apr-7	481950	May-1	333819	May-25	179317	Jun-18	151051	Jul-12	135234	Aug-5	120998	Aug-29	136040	Sep-22	304057
Mar-15	995767	Apr-8	483729	May-2	314230	May-26	207915	Jun-19	145590	Jul-13	119758	Aug-6	132187	Aug-30	143758	NA	NA

Figure 2: Presentation of data in the form of a report table

Data for graphic representation: number of tweets per day since the start of the war:

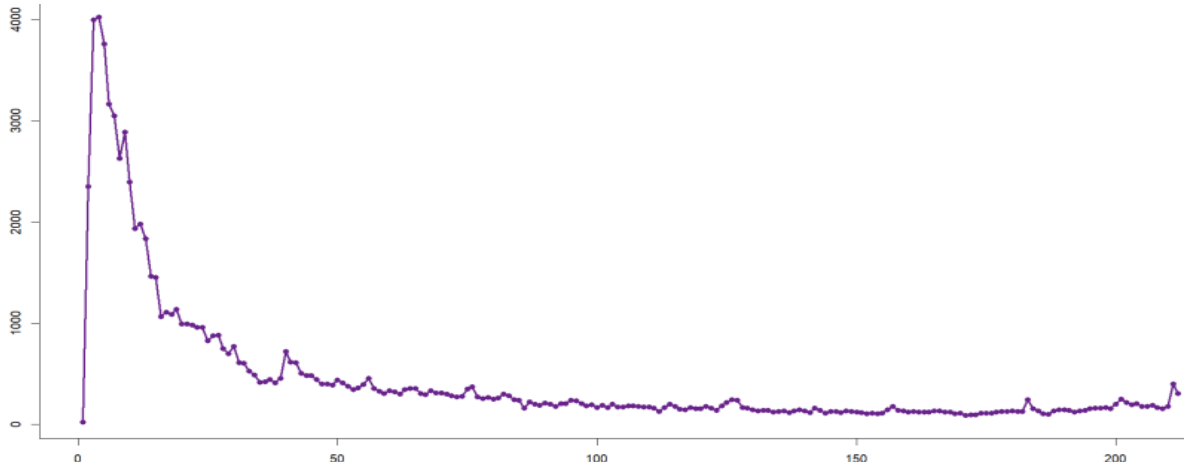

Figure 3: Plotting in the Cartesian coordinate system for dynamics of discussion about russian-Ukrainian war (x is day of war, y is the amount of tweets inthousands

Formulas for the transition to polar coordinates:

$$r^2 = y^2 + x^2$$

$$\varphi = \begin{cases} arctg\left(\frac{y}{x}\right) & x > 0,\ y \geq 0 \\ arctg\left(\frac{y}{x}\right) + 2\pi & x > 0,\ y < 0 \\ arctg\left(\frac{y}{x}\right) + \pi & x < 0 \\ \frac{\pi}{2} & x = 0,\ y > 0 \\ \frac{3\pi}{2} & x = 0,\ y < 0 \end{cases}$$

Data for graphical representation: ratio of hashtags #StopRussian and #StandWithUkraine.

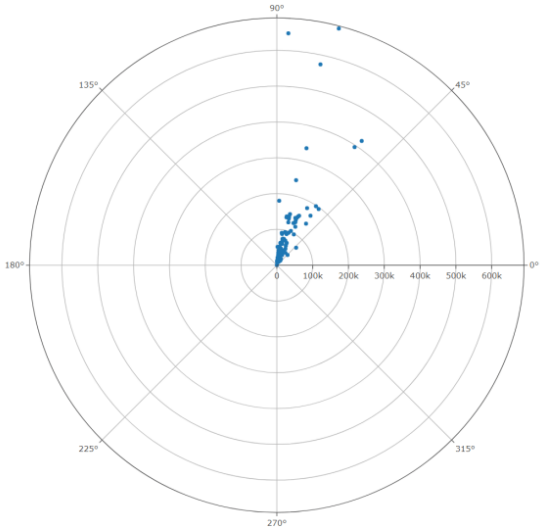

Figure 4: Graphing in the polar coordinate system

As we can already see from Fig. 1-4, the number of "references" to the war is much greater at the beginning of the war and significantly decreases over time. The graph of the Cartesian coordinate system clearly shows this rapid drop in quantity to a certain number, after which the quantity is maintained at a roughly constant level. The next important step in research is computation. As already mentioned, the hypothesis that was put forward has no "numerical" basis, and it is this problem that is solved by quantitative characteristics. We will use for work (Fig. 5):

- Mean – arithmetic mean (a measure of central tendency that reflects the most characteristic value for this sample): $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$;

- Sd – standard deviation (a measure of variation of a characteristic that reflects the amount of its spread relative to the arithmetic mean): $\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}}$;
- Median – median (the value that divides an ordered set of variables in half): $Me(n) = \frac{n+1}{2}$ (or odd n), $Me(n) = \frac{Me\left(\frac{n}{2}\right) + Me\left(\frac{n+2}{2}\right)}{2}$ (for even n);
- Trimmed (trimmed mean) – abbreviated average;
- Mad – median absolute deviation;
- Min – minimum value;
- Max – maximum value;
- Range – scope;
- Skew – asymmetry (reflects the skew of the distribution relative to the fashion to the left or right);
- Kurtosis – kurtosis (displays the height of the distribution);
- Se – standard error.

For calculations, we use the R studio environment and its built-in function for constructing descriptive statistics: `describe`.

	vars	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
daily_tweets	1	212	435301.92	664039.55	195051.0	269080.918	107186.050	22740	4027485	4004745	3.510906	12.99922	45606.424
Ukraine	2	212	218002.11	343803.61	102657.0	135736.818	63797.019	3300	2345772	2342472	4.002991	18.06601	23612.529
Russia	3	212	76491.29	95887.61	41951.0	54255.594	24768.316	3058	659808	656750	3.374894	13.28812	6585.588
StandWithUkraine	4	212	49080.83	87354.85	19680.0	29995.200	14023.172	440	660822	660382	4.692934	26.13242	5999.556
Putin	5	212	36974.48	59639.86	16620.0	22337.424	10100.212	2224	406355	404131	3.716964	14.76966	4096.082
UkraineRussiaWar	6	212	28976.79	63444.17	11736.5	16160.788	6799.945	0	510488	510488	5.848403	37.27886	4357.364
NATO	7	212	17204.54	19334.69	10394.5	12834.865	5707.269	470	151971	151501	3.254004	13.87084	1327.912
StopRussia	8	212	15967.20	31107.25	5011.5	8530.429	4420.372	2	236950	236948	4.324306	22.49984	2136.455
Russian	9	212	13736.26	17464.54	7887.0	10019.671	5415.938	318	154006	153688	4.269705	25.08915	1199.470
StopPutin	10	212	12960.83	37107.50	1909.0	4033.300	1255.762	15	304824	304809	4.994643	28.21328	2548.554
ukraine	11	212	11570.32	20210.28	4560.5	6743.612	2245.398	485	160405	159920	4.356686	22.16035	1388.048

Figure 5: Descriptive data statistics

Another way of presenting data is a histogram. It is often used to visualize the difference in the number of certain substances or substances over time. In our case, we will look at the histogram of the number of #Ukraine hashtags during March. A histogram is used to display interval series. At the same time, feature intervals are plotted on the abscissa axis, and frequencies are plotted on the ordinate axis.

For construction, it is necessary to find the number of partition intervals. For this, we will use the Sturges formula $k = 1 + log_2 n,\ where\ k$ is number of intervals.

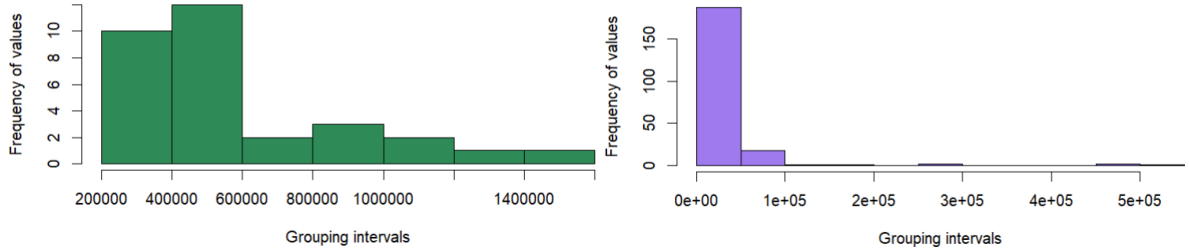

Figure 6: Histograms of hastag #Ukraine in March and of#UkraineRussiaWar hastags during all war

Cumulatives are also a way of presenting data. It has a certain similarity with a histogram, but for its construction it is necessary to calculate the accumulated frequencies. Accumulated frequencies show how many units of the population have a characteristic value no greater than that under consideration. You can build a cumulate in several ways, in particular, the number of #UkraineRussiaWar hashtags throughout the war (Fig.6). Number of split intervals: *Sturges formula* $k = 1 + log_2 n,\ k$ is number of intervals.

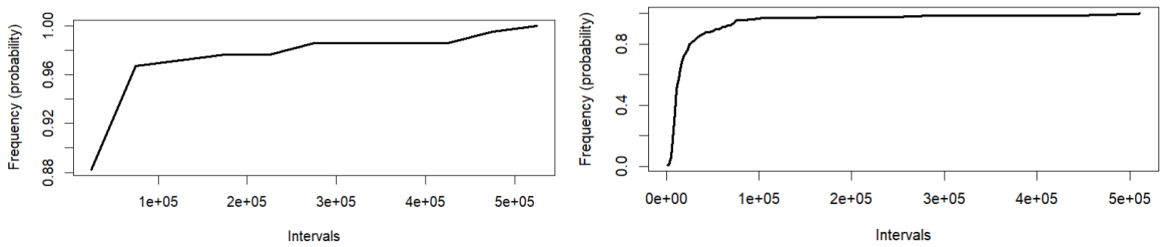

Figure 7: Cumulative plot by histogram and integral percentage

As can be seen in fig. 6-7, the built cumulates are similar, but the one built by integral percentage is smoother, so it gives a smaller error.

```
library(readr) # dataset loading
dataset1 <- read.csv("daily_stats.csv")
dataset2 <- read.csv("daily_hashtags.csv")
dataset <- cbind(dataset1[1], dataset1[3], dataset2[2:11])
n <- dim(dataset)[1]  # the number of elements
table <- as.data.frame(cbind(dataset$day, dataset$daily_tweets)) # formation of the report table
m <- dim(table)[2] # the number of stacks of the initial table
t <- 6# the number of subtables into which to split the original table
k <- round(n/t) # determining the number of rows in each subtable
l <- k + 1# row number that will move to the next subtable
a <- m + 1# the number of the first empty column of the initial table
for (j in 1:(t-1)){
 for (i in 1:k){
  for (q in 0:(m - 1)){    # table compression
   #if (table[l, 1 + q] != is.null)
    table [i, a + q] <- table[l, 1 + q]
   #else
    # stop
  }
  table <- table[-l, ]
 }
 a <- a + m  # determining the number of the next empty column
}
i <- 1# renaming the columns of the created table
while(i <= 2*t){
 names(table)[i] = "Date"
 names(table)[i + 1] = "Amount of tweets"
 i <- i + 2
}# plotting data in the Cartesian coordinate system
plot(1:dim(dataset)[1], dataset$daily_tweets/1000, xlab = "Day of war",
    ylab = "The amount of tweets (in thousands)", main = "Dynamics of discussion about russian-Ukranian war",
    type = "o", pch = 16, lwd = 3, col = "darkorchid4")
polardataset <- cbind(as.numeric(dataset$StopRussia), as.numeric(dataset$StandWithUkraine)) # plotting data in polar coordinate system
for(i in 1:n){ # creating a table for a polar graph regarding the ratio of hashtags StopRussian and StandWithUkraine
 r <- sqrt(polardataset[i, 1]^2 + polardataset[i, 2]^2)  # transition to polar coordinates
 if (polardataset[i, 1] > 0 && polardataset[i, 2] >= 0)
  f <- atan(polardataset[i, 2]/polardataset[i, 1])*180/pi
 if (polardataset[i, 1] > 0 && polardataset[i, 2] < 0)
  f <- atan(polardataset[i, 2]/polardataset[i, 1])*180/pi + 2*pi
 if (polardataset[i, 1] < 0)
  f <- atan(polardataset[i, 2]/polardataset[i, 1])*180/pi + pi
 if (polardataset[i, 1] == 0 && polardataset[i, 2] > 0)
  f <- pi/2
 if (polardataset[i, 1] == 0 && polardataset[i, 2] < 0)
  f <- 3*pi/2
 polardataset[i, 1] <- r
 polardataset[i, 2] <- f
}
library(plotly) # graph construction
plot_ly(  type = "scatterpolar",  r = polardataset[, 1],  theta = polardataset[, 2],  mode = 'markers')
library("psych")# construction of descriptive data statistics
statistics <- describe(dataset[2:12])
hist(as.numeric(dataset$Ukraine[7:37]), breaks = round(1 + log2(n)), # histogram construction of #Ukraine hashtags distribution in March
    xlab = "Grouping intervals", ylab = "Frequency of values",
    col = "seagreen4", main = "Histogram of hastag #Ukraine in March")
k <- round(1 + log2(n)) # construction of cumulate according to histogram data
```

```
h1 <- hist(as.numeric(dataset$UkraineRussiaWar), breaks = k, xlab = "Grouping intervals",
         ylab = "Frequency of values", col = "mediumpurple2",
         main = "Histogram of #UkraineRussiaWar hastags during all war")
plot(h1$mids, cumsum(h1$counts)/n, xlab = "Intervals", ylab = "Frequency (probability)",
    main = "Cumulative plot by histogram", 'l',  lwd = 3)
m <- dim(dataset)[1] - 1# construction of the cumulate by integral interest
h2 <- hist(as.numeric(dataset$UkraineRussiaWar), breaks = m, xlab = "Grouping intervals",
       ylab = "Frequency of values", col = "mediumpurple2",
       main = "Histogram of #UkraineRussiaWar hastags during all war")
plot(h2$mids, cumsum(h2$counts)/n, xlab = "Intervals", ylab = "Frequency (probability)",
    main = "Cumulative plot by integral percentage", 'l', lwd = 3)
```

4. Experiments, results and discussions

The study of a numerical series is mainly based on the analysis of its graphical representation. That is, if we have a graph (a certain line), we can use it to provide conclusions about the state of our initial data. This works well if the basic patterns are not affected by any other, random, external factors. If such an influence is monitored, sometimes in order to "clean" the main trend from "obstacles", smoothing methods are used. Time series for smoothing: number of #NATO hashtags for each day of the war. The moving average method gives an estimate of the average level for a certain period of time (the longer the time interval to which the average belongs, the more the level will be smoothed, but the less accurately the trend of the original series of dynamics will be described).

The built-in function sma() was used in the programming environment.

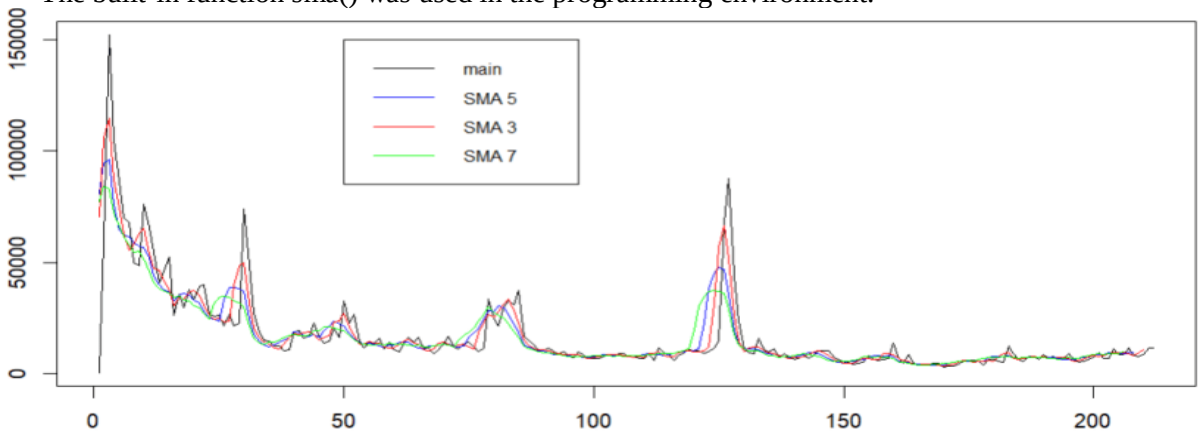

Figure 8: Simple moving average (x – time, y – count of #NATO)

The method of the weighted moving average on each active area of the value of the central level is replaced by the calculated one, which is determined by the formula of the weighted arithmetic average (the weighting factors are determined using the method of least squares).

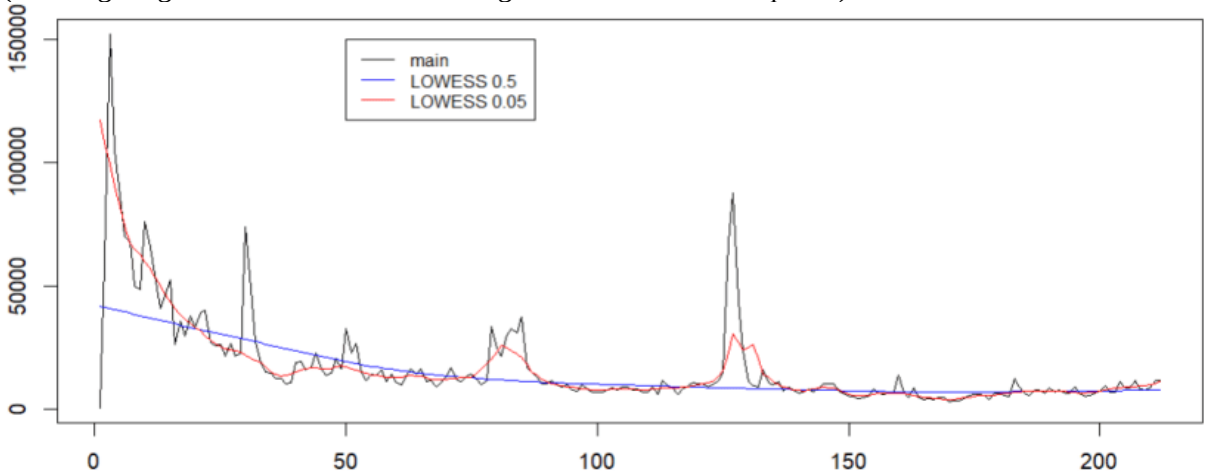

Figure 9: Locally weighted smoothing (x – time, y – count of #NATO)

```
library("zoo")# smoothing with a moving average
plot.ts(dataset$NATO,main = "Simple moving average", ylab = "Count of #NATO")
lines(rollmean(dataset$NATO,5),col = 'blue')
lines(rollmean(dataset$NATO,3),col = 'red')
lines(rollmean(dataset$NATO,7),col = 'green')
legend(50,150000,col = c('black','blue', 'red','green'), legend = c('main', 'SMA 5', 'SMA 3','SMA 7'),lty = 1,cex = 0.8)
plot.ts(dataset$NATO,main = "Locally weighted smoothing", ylab = "Count of #NATO")# lowess smoothing curve
lines(lowess(dataset$NATO,f = 0.5),col = "blue")
lines(lowess(dataset$NATO,f = 0.05),col = "red")
legend(50,150000,col = c('black','blue','red'), legend = c('main', 'LOWESS 0.5', 'LOWESS 0.05'),lty = 1,cex = 0.8)
```

When applying the method of moving averages, the selection of the smoothing interval value should be made on the basis of meaningful considerations and be tied to the period of possibly existing oscillatory processes. If the moving average procedure is used to smooth the time series in the absence of any fluctuations, then most often the value of the smoothing interval is chosen equal to three, five or seven. The larger the averaging interval, the smoother the trend graph looks. Time series for smoothing: number of #UkraineRussiaWar hashtags for each day of the war (in thousands). Linear smoothing for w = 3: $\bar{y}_1 = \frac{5y_1+2y_2-y_3}{6}$; $\bar{y}_i = \frac{y_{i-1}+y_i+y_{i+1}}{3}, \bar{y}_n = \frac{-y_{n-2}+2y_{n-1}+5y_n}{6}, i = 2, 3, \dots, n-1.$

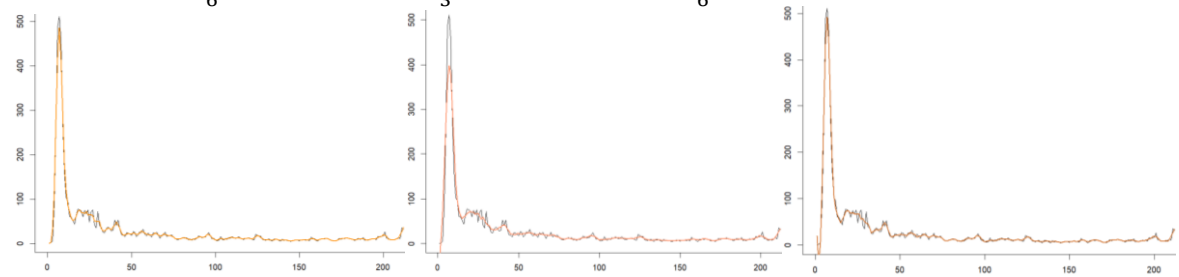

Figure 10: Linear smoothing for w=3, 5, 7 (x – variant number, y – variant value in thousands)

Linear smoothing for w = 5: $\bar{y}_1 = \frac{3y_1+2y_2+y_3-y_5}{5}, \bar{y}_2 = \frac{4y_1+3y_2+2y_3+y_4}{10}, \bar{y}_i = \frac{y_{i-2}+y_{i-1}+y_i+y_{i+1}+y_{i+2}}{5}$, $\bar{y}_{n-1} = \frac{y_{n-3}+2y_{n-2}+3y_{n-1}+4y_n}{10}, \bar{y}_n = \frac{-y_{n-4}+y_{n-2}+2y_{n-1}+3y_n}{5}$, $where\ i = 3, 4, \dots, n-2.$

Nonlinear smoothing for w = 7: $i = 4, 5, \dots, n-3$

$\bar{y}_1 = \frac{39y_1+8y_2-4y_3-4y_4+y_5+4y_6-2y_7}{42}, \bar{y}_2 = \frac{8y_1+19y_2+16y_3+6y_4-4y_5-7y_6+4y_7}{42},$

$\bar{y}_3 = \frac{-4y_1+16y_2+19y_3+12y_4+2y_5-4y_6+y_7}{42}, \bar{y}_i = \frac{-2y_{i-3}+3y_{i-2}+6y_{i-1}+7y_i+6y_{i+1}+3y_{i+2}-2y_{i+3}}{21},$

$\bar{y}_{n-2} = \frac{y_{n-6}-4y_{n-5}+2y_{n-4}+12y_{n-3}+19y_{n-2}+16y_{n-1}-4y_n}{42}, \bar{y}_{n-2} = \frac{4y_{n-6}-7y_{n-5}-4y_{n-4}+6y_{n-3}+16y_{n-2}+19y_{n-1}+8y_n}{42},$

$\bar{y}_n = \frac{-2y_{n-6}+4y_{n-5}+y_{n-4}-4y_{n-3}-4y_{n-2}+8y_{n-1}+39y_n}{42}.$

```
y_n <- dataset$UkraineRussiaWar/1000 # linear smoothing for w = 3
l3_y_n <- vector()
l3_y_n[1] <- (5 * y_n[1] + 2 * y_n[2] - y_n[3])/6
for (i in 2:(n - 1))
  l3_y_n[i] <- (y_n[i - 1] + y_n[i] + y_n[i + 1])/3
l3_y_n[n] <- (- y_n[n - 2] + 2 * y_n[n - 1] + 5 * y_n[n])/6
plot(c(1:n), y_n, xlab = "Variant number", ylab = "Variant value (thousands)", main = "Linear smoothing for w = 3", 'l', lwd = 1)
lines(l3_y_n, col = "darkorange", lwd = 2)
l5_y_n <- vector()# linear smoothing for w = 5
l5_y_n[1] <- (3 * y_n[1] + 2 * y_n[2] + y_n[3] - y_n[5])/5
l5_y_n[2] <- (4 * y_n[1] + 3 * y_n[2] + 2 * y_n[3] + y_n[4])/10
for(i in 3:(n - 2))
  l5_y_n[i] <- (y_n[i - 2] + y_n[i - 1] + y_n[i] + y_n[i + 1] + y_n[i + 2])/5
l5_y_n[n - 1] <- (y_n[n - 3] + 2 * y_n[n - 2] + 3 * y_n[n - 1] + 4 * y_n[n])/10
l5_y_n[n] <- (- y_n[n - 4] + y_n[n - 2] + 2 * y_n[n - 1] + 3 * y_n[n])/5
plot(c(1:n), y_n, xlab = "Variant number", ylab = "Variant value (thousands)", main = "Linear smoothing for w = 5", 'l', lwd = 1)
lines(l5_y_n, col = "coral", lwd = 2)
l7_y_n <- vector()# nonlinear smoothing for w = 7
l7_y_n[1] <- (39*y_n[1] + 8*y_n[2] - 4*y_n[3] - 4*y_n[4] + y_n[5] + 4*y_n[6] - 2*y_n[7])/42
l7_y_n[2] <- (8*y_n[1] + 19*y_n[2] + 16*y_n[3] + 6*y_n[4] - 4*y_n[5] - 7*y_n[6] + 4*y_n[7])/42
l7_y_n[3] <- (-4*y_n[1] + 16*y_n[2] + 19*y_n[3] + 12*y_n[4] + 2*y_n[5] - 4*y_n[6] + y_n[7])/42
for(i in 4:(n - 3))
  l7_y_n[i] <- (-2*y_n[i - 3] + 3*y_n[i - 2] + 6*y_n[i - 1] + 7*y_n[i] + 6*y_n[i + 1] + 3*y_n[i + 2] - 2*y_n[i + 3])/21
l7_y_n[n - 2] <- (y_n[n - 6] - 4*y_n[n - 5] + 2*y_n[n - 4] + 12*y_n[n - 3] + 19*y_n[n - 2] + 16*y_n[n - 1] - 4*y_n[n])/42
```

```
l7_y_n[n - 1] <- (4*y_n[n - 6] - 7*y_n[n - 5] - 4*y_n[n - 4] + 6*y_n[n - 3] + 16*y_n[n - 2] + 19*y_n[n - 1] + 8*y_n[n])/42
l7_y_n[n] <- (-2*y_n[n - 6] + 4*y_n[n - 5] + y_n[n - 4] - 4*y_n[n - 3] - 4*y_n[n - 2] + 8*y_n[n - 1] + 39*y_n[n])/42
l5_y_n[n] <- (- y_n[n - 4] + y_n[n - 2] + 2 * y_n[n - 1] + 3 * y_n[n])/5
plot(c(1:n), y_n, xlab = "Variant number", ylab = "Variant value (thousands)", main = "Nonlinear smoothing for w = 7", 'l', lwd = 1)
lines(l7_y_n, col = "chocolate", lwd = 2)
```

Normalization of time sequences:

- makes it possible to compare the indicators obtained for different objects;
- linear transformation, which consists in the fact that the values of the levels of the time series lead to the interval of values [0,1] according to the formula: $y_i^{\text{H}} = \frac{y_i - y_{min}}{y_{max} - y_{min}}$, where y_i^{H} is normalized value, y_i is level value, y_{min} and y_{max} – the smallest and largest value of the levels of the time series.

```
norm_sequence <- function(x){ # normalization of time sequences
 x_norm <- vector()
 for (i in 1:length(x))
  x_norm[i] <- (x[i] - min(x))/(max(x) - min(x))
 return(x_norm)
}
```

Time series smoothing effectiveness criteria;

- Criterion of turning points: $((I_3 > I_2) \&\& (I_3 > I_4)) \,||\, ((I_3 < I_2) \&\& (I_3 < I_4))$.
- Correlation coefficient: $r_{xy} = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sqrt{\left[n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2\right]\left[n\sum_{i=1}^{n} y_i^2 - \left(\sum_{i=1}^{n} y_i\right)^2\right]}}$.

```
turn_points <- function(x, n){ # the search function for the number of turning points
 t <- 0
 for (i in 2:(n - 1)){
  if ((x[i] > x[i - 1]&& x[i] > x[i + 1]) || (x[i] < x[i - 1] && x[i] < x[i + 1]))
   t <- t + 1
 }
 return(t)
}
turn_points_value <- function(x, n){ # turning point search function
 v <- vector()
 for (i in 2:(n - 1)){
  if ((x[i] > x[i - 1]&& x[i] > x[i + 1]) || (x[i] < x[i - 1] && x[i] < x[i + 1]))
   v <- cbind(v, i)
 }
 return(x[v])
}
correlation_coeficient <- function(x, y, n){ # correlation coefficient determination function
 S1 <- 0
 S2 <- 0
 S3 <- 0
 S4 <- 0
 S5 <- 0
 for (i in 1:n){
  S1 <- S1 + x[i]
  S2 <- S2 + y[i]
  S3 <- S3 + x[i]*y[i]
  S4 <- S4 + (x[i])^2
  S5 <- S5 + (y[i])^2
 }
 return((n*S3 - S1*S2)/(sqrt((n*S4 - (S1)^2)*(n*S5 - (S2)^2))))
}
```

Smoothing according to Kendel's formulas is a type of weighted moving average. Made smoothing according to Kendel's formulas for various w (3, 5, 7, 9, 11, 13, 15) for #NATO hashtags. According to fig. 12, we can conclude that the property of linear smoothing (the larger the coefficient, the stronger the graph is smoothed) is also true for smoothing according to Kendel's formulas. Correlation analysis refers to a set of methods that make it possible to detect the presence and degree of relationship between several randomly changing parameters (more details about correlation analysis will be found in clause 4, at this stage we are only interested in such concepts as "correlation table" (table of ratios) and turning points (levels whose values are greater or less than two adjacent ones)).

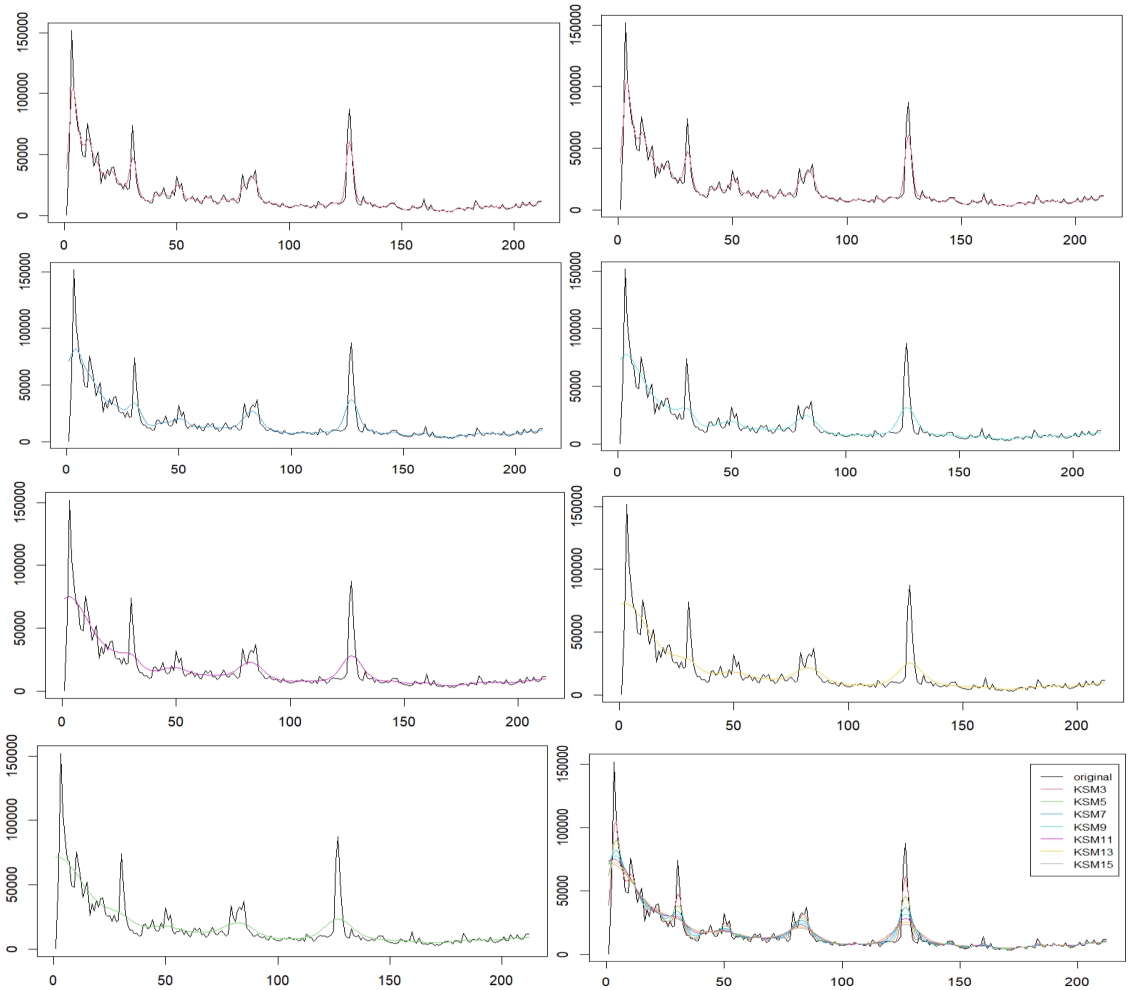

Figure 11: Kendall smoothing for w=3,5, 7, 9, 11, 13. 15 and w = [3;15] (x – time, y – #NATO)

	Y	Yn, w = 3	Yn, w = 5	Yn, w = 7	Yn, w = 9	Yn, w = 11	Yn, w = 13	Yn, w = 15
1	470	38559.105	62198.287	70938.987	73464.266	73636.084	72879.068	71700.904
2	59184	71915.782	76469.861	77059.688	76072.003	74640.840	73078.299	71452.709
3	151971	101788.420	87084.501	81048.556	77498.557	74956.092	72835.200	70897.029
4	105232	104749.978	90457.321	82185.360	77583.998	74534.600	72129.312	70023.097
5	86367	89649.484	86111.591	80358.649	76327.796	73378.230	70958.509	68828.826
6	69978	75055.661	77434.010	76283.332	73923.363	71543.140	69340.825	67321.433
7	67966	64469.676	68905.740	71257.955	70720.424	69135.239	67315.241	65519.509
8	49740	57608.156	63299.104	66514.654	67121.046	66294.576	64940.610	63454.517
9	48362	58199.546	60863.743	62640.591	63450.540	63170.443	62290.597	61166.271
10	75941	63054.169	59808.725	59480.388	59877.872	59898.372	59448.572	58705.311
11	65639	62097.283	57903.447	56506.536	56427.003	56587.849	56502.353	56129.220
12	54535	54685.988	54342.467	53290.948	53056.016	53321.644	53534.109	53498.880
13	40650	48058.981	49922.230	49743.082	49746.396	50165.823	50619.617	50874.827
14	45967	45522.819	45603.117	46075.462	46550.730	47180.861	47824.041	48313.458
15	52386	42332.483	41656.957	42627.914	43581.895	44419.647	45199.463	45864.139
16	26118	36611.638	38235.701	39710.509	40960.477	41926.227	42783.868	43566.921
17	35548	33367.569	35833.622	37519.591	38766.556	39725.582	40600.890	41451.639
18	29852	33401.290	34796.269	36065.632	36997.838	37817.900	38661.247	39537.915
19	37863	34601.428	34813.634	35147.006	35566.298	36178.121	36965.239	37835.339
20	32911	35847.468	35066.474	34411.335	34341.218	34772.533	35506.432	36344.545
21	39224	36673.749	34680.030	33503.066	33206.125	33575.498	34275.433	35058.083
22	39878	34960.279	33174.262	32244.215	32117.061	32574.814	33258.008	33960.637
23	26950	30592.398	30779.280	30731.090	31128.629	31780.118	32437.925	33028.564
24	25730	27004.453	28275.243	29313.233	30376.080	31209.328	31793.567	32230.833
25	26315	25178.411	26478.050	28461.361	30010.681	30865.650	31286.029	31526.817
26	21470	24299.067	26013.134	28572.735	30106.630	30711.321	30860.369	30871.444

Figure 12: A fragment of the generalized correlation table for smoothing intervals according to Kendel's formulas

Plotting pivot points for all smoothing intervals:

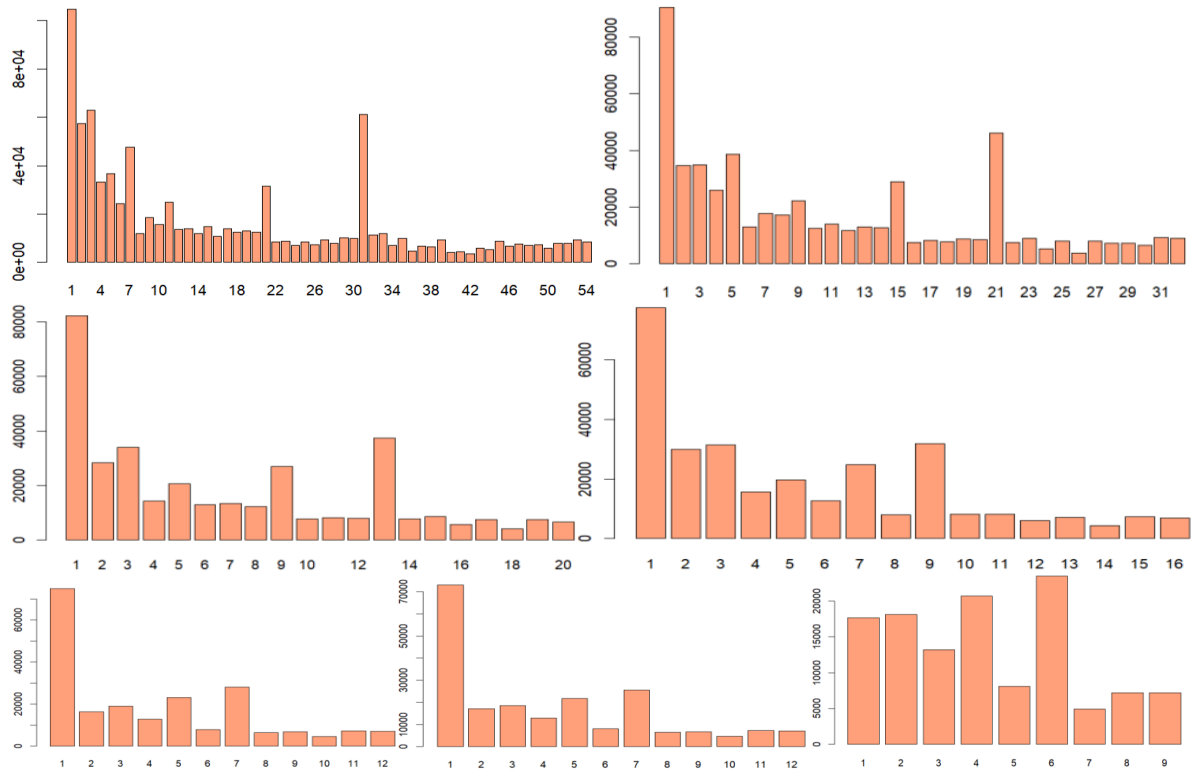

Figure 13: Turned points diagram, w=3, 5, 7, 9, 11, 13 and 15

Determination of efficiency criteria for performed smoothing according to formulas from Kendel:

Table 1

Table of smoothing efficiency criteria according to formulas from Kendel

Smoothing interval w	Number of turning points	Correlation coefficient r_{xy}
3	54	0.9509346
5	32	0.8964826
7	20	0.8611363
9	16	0.8399593
11	12	0.8264445
13	12	0.8166181
15	9	0.8086317

Conclusion: the larger the parameter, the fewer turning points, the smaller the correlation coefficient. Therefore, the more "smoothed" is the resulting graph (for ordinary smoothing according to Kendel's formulas).

```
 Kendall smoothing criteria:
     Smoothing parameter Amount of turned points Correlation coefficient
[1,]                   3                      54                0.9509346
[2,]                   5                      32                0.8964826
[3,]                   7                      20                0.8611363
[4,]                   9                      16                0.8399593
[5,]                  11                      12                0.8264445
[6,]                  13                      12                0.8166181
[7,]                  15                       9                0.8086317
```

Figure 14: Result of software finding the smoothing efficiency criteria according to Kendel's formulas

Re-smoothing is when we apply the data obtained in the previous layer to the input data of the next layer. Re-smoothing according to Kendel's formulas for different w (3, 5, 7, 9, 11, 15):

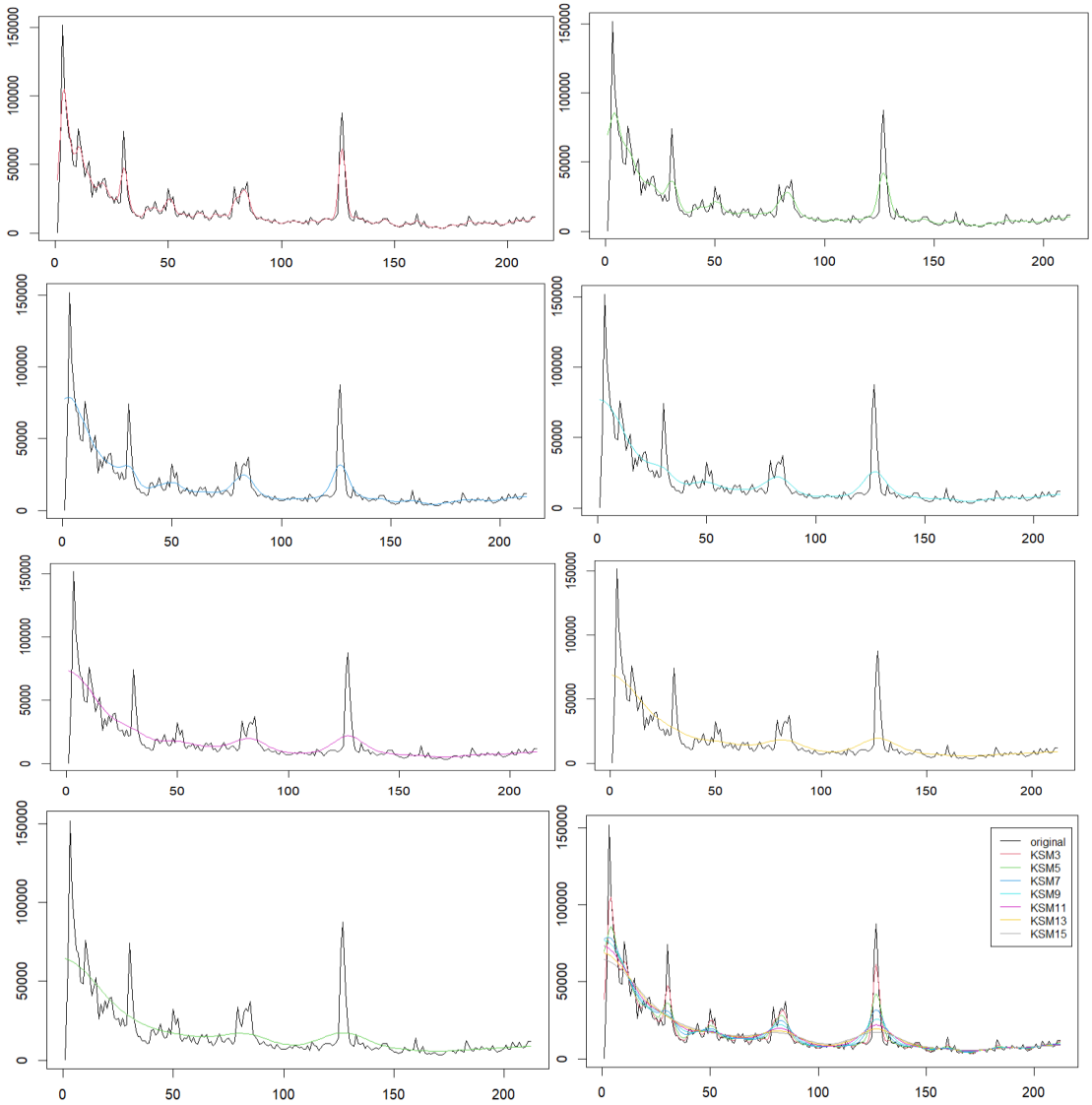

Figure 15: Repeated Kendall smoothing for w=3, 5, 7, 9, 11, 13, 15 and w=[3;15] (x – time, y – #NATO)

Construction of a correlation table for all intervals of re-smoothing according to formulas from Kendel:

	Y	Y'n, w = 3	Y'n, w = 5	Y'n, w = 7	Y'n, w = 9	Y'n, w = 11	Y'n, w = 13	Y'n, w = 15
1	470	38559.105	69770.932	77930.582	76848.760	73189.777	68806.171	64383.923
2	59184	71915.782	77377.077	78518.786	76223.324	72441.589	68131.430	63824.164
3	151971	101788.420	83503.411	78632.521	75355.588	71527.394	67340.824	63184.153
4	105232	104749.978	85852.169	78015.761	74186.893	70420.364	66419.663	62455.185
5	86367	89649.484	83323.035	76480.826	72666.527	69096.802	65354.785	61629.223
6	69978	75055.661	77225.821	74014.347	70764.884	67540.224	64136.191	60699.591
7	67966	64469.676	70362.562	70815.905	68485.322	65745.676	62758.845	59661.753
8	49740	57608.156	64988.983	67216.833	65869.074	63723.146	61224.353	58514.095
9	48362	58199.546	61658.975	63528.160	62989.962	61498.856	59542.118	57258.596
10	75941	63054.169	59459.026	59928.012	59940.977	59113.766	57729.632	55901.254
11	65639	62097.283	57081.634	56452.921	56818.839	56619.506	55811.702	54452.137
12	54535	54685.988	53841.684	53069.773	53712.332	54072.894	53818.673	52925.022

Figure 16: A fragment of the generalized correlation table for re-smoothing intervals according to Kendel's formulas

Construction of turning point diagrams for all re-smoothing intervals according to Kendel's formulas:

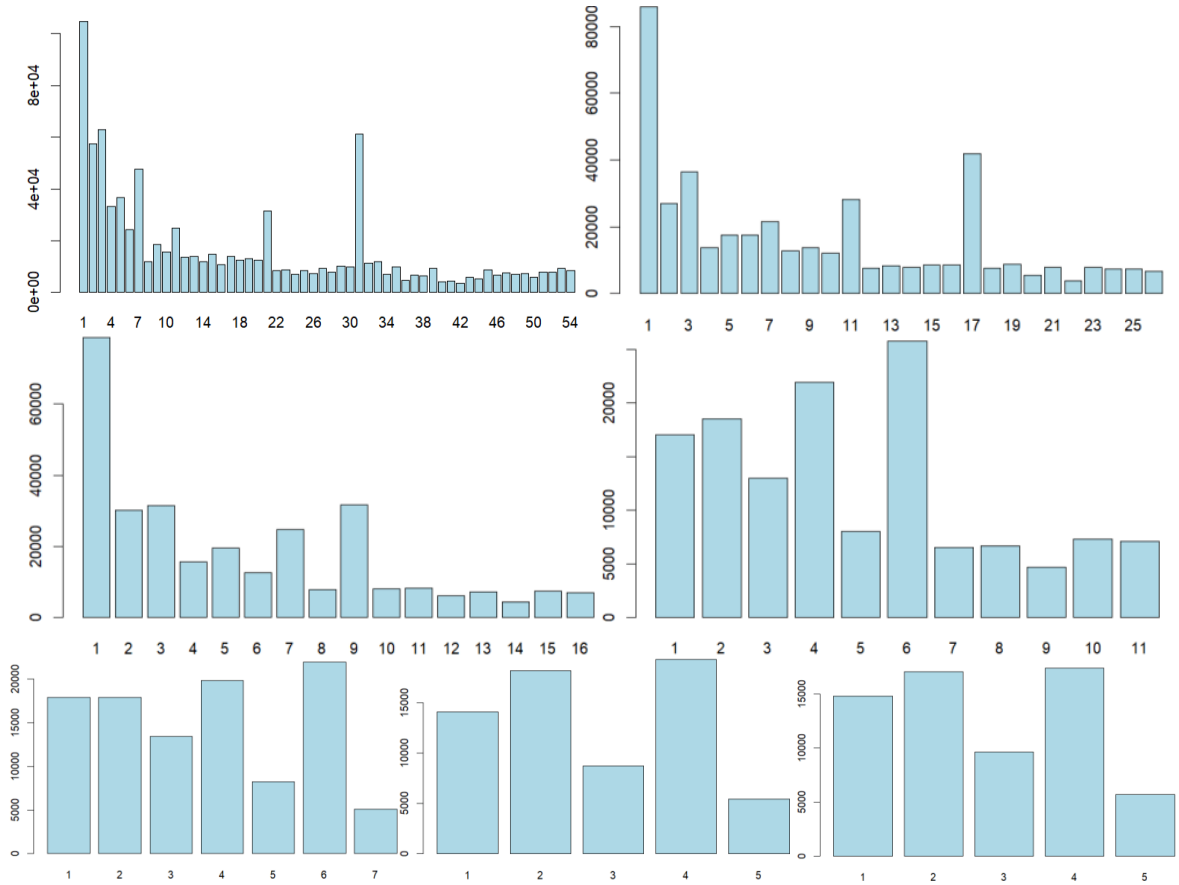

Figure 17: Turned points diagram, w=3, 5, 7, 9, 11, 13, 15

Determination of efficiency criteria for repeated smoothing according to Kendel's formulas:

Table 2

Table of criteria for the effectiveness of repeated smoothing according to Kendel's formulas

Smoothing interval w	Number of turning points	Correlation coefficient r_{xy}
3	54	0.9509346
5	26	0.8758931
7	16	0.8346677
9	11	0.8154749
11	7	0.8021700
13	5	0.7902080
15	5	0.7785690

```
 Repeated Kendall smoothing criteria:
     Smoothing parameter Amount of turned points Correlation coefficient
[1,]                   3                      54                0.9509346
[2,]                   5                      26                0.8758931
[3,]                   7                      16                0.8346677
[4,]                   9                      11                0.8154749
[5,]                  11                       7                0.8021700
[6,]                  13                       5                0.7902080
[7,]                  15                       5                0.7785690
```

Figure 18: The result of software finding criteria for the effectiveness of repeated smoothing according to Kendel's formulas

Compared to the results of "no re-smoothing", with re-smoothing, the number of turning points decreases more rapidly, as does the correlation coefficient. Here already on parameter "11" the number of turning points was 7, when there - 12 (and the smallest value is 9, when here - 5).

```
cat("\t\tKendall smoothing")# smoothing according to formulas from Kendel
library(irr)
kSmooth1 <- ksmooth(time(dataset$NATO), dataset$NATO, 'normal', bandwidth=3)
kSmooth2 <- ksmooth(time(dataset$NATO), dataset$NATO, 'normal', bandwidth=5)
kSmooth3 <- ksmooth(time(dataset$NATO), dataset$NATO, 'normal', bandwidth=7)
kSmooth4 <- ksmooth(time(dataset$NATO), dataset$NATO, 'normal', bandwidth=9 )
kSmooth5 <- ksmooth(time(dataset$NATO), dataset$NATO, 'normal', bandwidth=11)
kSmooth6 <- ksmooth(time(dataset$NATO), dataset$NATO, 'normal', bandwidth=13)
kSmooth7 <- ksmooth(time(dataset$NATO), dataset$NATO, 'normal', bandwidth=15 )
plot.ts(dataset$NATO, main = 'Kendall Smoothing', ylab = '#NATO') # image of smoothing according to Kendel's formulas on the graph
lines(dataset$NATO, col = '1')
lines(kSmooth1, type = 'l', col = '2')
lines(kSmooth2, type = 'l', col = '3')
lines(kSmooth3, type = 'l', col = '4')
lines(kSmooth4, type = 'l', col = '5')
lines(kSmooth5, type = 'l', col = '6')
lines(kSmooth6, type = 'l', col = '7')
lines(kSmooth7, type = 'l', col = '8')
legend(180,150000, col = c('1','2','3','4','5','6','7','8'),
     legend = c('original', 'KSM3', 'KSM5','KSM7', 'KSM9','KSM11', 'KSM13','KSM15'),lty = 1, cex = 0.8)
kendallSmoothing <- cbind(dataset$NATO, kSmooth1[[2]], kSmooth2[[2]], kSmooth3[[2]], kSmooth4[[2]],
                          kSmooth5[[2]],kSmooth6[[2]], kSmooth7[[2]]) # construction of a generalized correlation table
colnames(kendallSmoothing) <- c("Y", "Yn, w = 3", "Yn, w = 5", "Yn, w = 7", "Yn, w = 9", "Yn, w = 11", "Yn, w = 13", "Yn, w = 15")
View(kendallSmoothing)
k_t <- vector()# the number of turning points when smoothing according to formulas from Kendel
for (i in 1:7){
  k_t[i] <- turn_points(kendallSmoothing[, i + 1], dim(kendallSmoothing)[1])
}# construction of diagrams of turning points for smoothing according to formulas from Kendel
barplot(turn_points_value(kSmooth1[[2]], length(kSmooth1[[2]])), col = "lightsalmon",
        names.arg = c(1:k_t[1]), main = "Turned points diagram, w = 3")
barplot(turn_points_value(kSmooth2[[2]], length(kSmooth2[[2]])), col = "lightsalmon",
        names.arg = c(1:k_t[2]), main = "Turned points diagram, w = 5")
barplot(turn_points_value(kSmooth3[[2]], length(kSmooth3[[2]])), col = "lightsalmon",
        names.arg = c(1:k_t[3]), main = "Turned points diagram, w = 7")
barplot(turn_points_value(kSmooth4[[2]], length(kSmooth4[[2]])), col = "lightsalmon",
        names.arg = c(1:k_t[4]), main = "Turned points diagram, w = 9")
barplot(turn_points_value(kSmooth5[[2]], length(kSmooth5[[2]])), col = "lightsalmon",
        names.arg = c(1:k_t[5]), main = "Turned points diagram, w = 11")
barplot(turn_points_value(kSmooth6[[2]], length(kSmooth6[[2]])), col = "lightsalmon",
        names.arg = c(1:k_t[6]), main = "Turned points diagram, w = 13")
barplot(turn_points_value(kSmooth7[[2]], length(kSmooth7[[2]])), col = "lightsalmon",
        names.arg = c(1:k_t[7]), main = "Turned points diagram, w = 15")
k_r_xy <- vector()# correlation coefficients when smoothing according to formulas from Kendel
for (i in 1:7){
  k_r_xy[i] <- correlation_coeficient(dataset$NATO, kendallSmoothing[, i + 1], dim(kendallSmoothing)[1])
}
k <- c(3, 5, 7, 9, 11, 13, 15) # displaying on the screen the number of turning points and correlation coefficients at different w
kndl_matrix <- cbind(k, k_t, k_r_xy)
colnames(kndl_matrix) <- c("Smoothing parameter", "Amount of turned points", "Correlation coefficient")
cat("\nKendall smoothing criteria:\n")
print(kndl_matrix)
cat("\n\t\tRepeated Kendall smoothing")# re-smoothing according to formulas from Kendel
kSmooth1 <- ksmooth(time(dataset$NATO), dataset$NATO,'normal',bandwidth = 3)
kSmooth2 <- ksmooth(time(kSmooth1[[2]]), kSmooth1[[2]], 'normal', bandwidth = 5)
kSmooth3 <- ksmooth(time(kSmooth2[[2]]), kSmooth2[[2]], 'normal', bandwidth = 7)
kSmooth4 <- ksmooth(time(kSmooth3[[2]]), kSmooth3[[2]], 'normal', bandwidth = 9)
kSmooth5 <- ksmooth(time(kSmooth4[[2]]), kSmooth4[[2]], 'normal', bandwidth = 11)
kSmooth6 <- ksmooth(time(kSmooth5[[2]]), kSmooth5[[2]], 'normal', bandwidth = 13)
kSmooth7 <- ksmooth(time(kSmooth6[[2]]), kSmooth6[[2]], 'normal', bandwidth = 15)
plot.ts(dataset$NATO, main = "Repeated Kendall Smoothing", ylab = '#NATO') # image of re-smoothing using Kendel's formulas on a graph
lines(dataset$NATO,col='1')
lines(kSmooth1,type='l',col='2')
lines(kSmooth2,type='l',col='3')
lines(kSmooth3,type='l',col='4')
lines(kSmooth4,type='l',col='5')
lines(kSmooth5,type='l',col='6')
lines(kSmooth6,type='l',col='7')
lines(kSmooth7,type='l',col='8')
legend(180,150000, col = c('1','2','3','4','5','6','7','8'),
     legend = c('original', 'KSM3', 'KSM5','KSM7', 'KSM9','KSM11', 'KSM13','KSM15'), lty = 1, cex = 0.8)
kendallSmoothing <- cbind(dataset$NATO,kSmooth1[[2]], kSmooth2[[2]], kSmooth3[[2]], # construction of a generalized correlation table
```

```
                kSmooth4[[2]],kSmooth5[[2]], kSmooth6[[2]], kSmooth7[[2]])
colnames(kendallSmoothing) <- c("Y", "Y'n, w = 3", "Y'n, w = 5", "Y'n, w = 7", "Y'n, w = 9", "Y'n, w = 11", "Y'n, w = 13", "Y'n, w = 15")
View(kendallSmoothing)
rep_k_t <- vector()# the number of turning points during re-smoothing according to Kendel's formulas
for (i in 1:7){
  rep_k_t[i] <- turn_points(kendallSmoothing[, i + 1], dim(kendallSmoothing)[1])
} # construction of diagrams of turning points for re-smoothing according to formulas from Kendel
barplot(turn_points_value(kSmooth1[[2]], length(kSmooth1[[2]])), col = "lightblue",
      names.arg = c(1:rep_k_t[1]), main = "Turned points diagram, w = 3")
barplot(turn_points_value(kSmooth2[[2]], length(kSmooth2[[2]])), col = "lightblue",
      names.arg = c(1:rep_k_t[2]), main = "Turned points diagram, w = 5")
barplot(turn_points_value(kSmooth3[[2]], length(kSmooth3[[2]])), col = "lightblue",
      names.arg = c(1:rep_k_t[3]), main = "Turned points diagram, w = 7")
barplot(turn_points_value(kSmooth4[[2]], length(kSmooth4[[2]])), col = "lightblue",
      names.arg = c(1:rep_k_t[4]), main = "Turned points diagram, w = 9")
barplot(turn_points_value(kSmooth5[[2]], length(kSmooth5[[2]])), col = "lightblue",
      names.arg = c(1:rep_k_t[5]), main = "Turned points diagram, w = 11")
barplot(turn_points_value(kSmooth6[[2]], length(kSmooth6[[2]])), col = "lightblue",
      names.arg = c(1:rep_k_t[6]), main = "Turned points diagram, w = 13")
barplot(turn_points_value(kSmooth7[[2]], length(kSmooth7[[2]])), col = "lightblue",
      names.arg = c(1:rep_k_t[7]), main = "Turned points diagram, w = 15")
rep_k_r_xy <- vector()# correlation coefficients with repeated smoothing according to Kendel's formulas
for (i in 1:7){
  rep_k_r_xy[i] <- correlation_coeficient(dataset$NATO, kendallSmoothing[, i + 1], dim(kendallSmoothing)[1])
}
k <- c(3, 5, 7, 9, 11, 13, 15) # displaying on the screen the number of turning points and correlation coefficients at different w
rep_kndl_matrix <- cbind(k, rep_k_t, rep_k_r_xy)
colnames(rep_kndl_matrix) <- c("Smoothing parameter", "Amount of turned points", "Correlation coefficient")
cat("\nRepeated Kendall smoothing criteria:\n")
print(rep_kndl_matrix)
```

Correlation analysis of time series is used when it is necessary to assess the presence and strength of the relationship between certain indicators. In our case, the total number of tweets for each day since the beginning of the war was taken as a series for analysis (we will present the data in the form of a table). The correlation field is a graphical representation of the relationship between the two investigated sequences (in our case, time and quantity). Correlation coefficient: an indicator of the quantitative assessment of the tightness of the connection; in the range from -1 to 1; characterizes a linear relationship, where when one value increases, the other increases (decreases)..

№	Day of war	Amount of tweets
1	Feb-23	22740
2	Feb-24	2352003
3	Feb-25	3997584
4	Feb-26	4027485
5	Feb-27	3762020
6	Feb-28	3167455
7	Mar-1	3050607
8	Mar-2	2628486
9	Mar-3	2892043
10	Mar-4	2396854
11	Mar-5	1934266
12	Mar-6	1982330
13	Mar-7	1838330
14	Mar-8	1466125
15	Mar-9	1453713
16	Mar-10	1067689
17	Mar-11	1107978
18	Mar-12	1085582
19	Mar-13	1138400
20	Mar-14	995092
21	Mar-15	995767

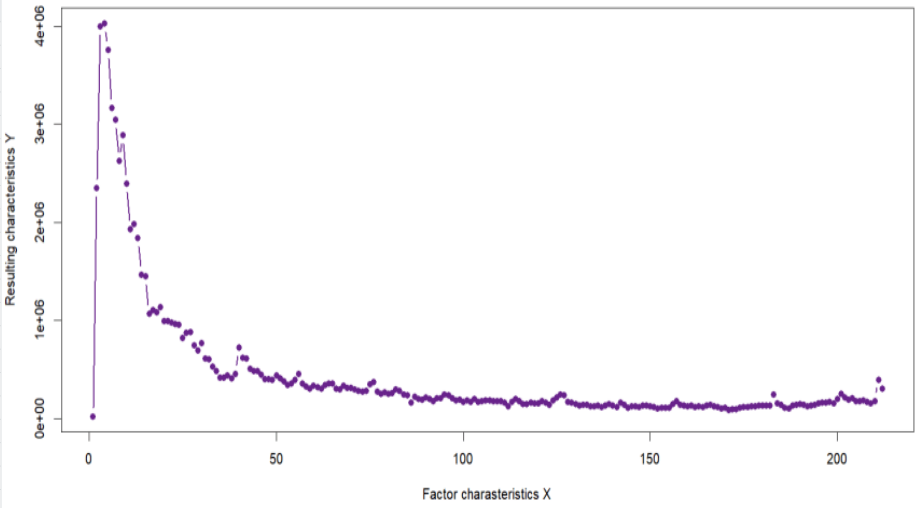

Figure 19: Fragment of a time series for correlation analysis and correlation field

The following formula is used for calculations: $r_{xy} = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sqrt{[n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2][n\sum_{i=1}^{n} y_i^2 - (\sum_{i=1}^{n} y_i)^2]}}$.

```
Correlation analysis
Correlation coefficient: r_xy = -0.5959894
```

Figure 20: Found correlation coefficient

Since the correlation coefficient is negative, we can conclude that the dependence is decreasing, that is, as the value of one value increases, the other decreases. In our case, over time, the number of hashtags decreases (activity and interest in the problem, respectively). Correlation relation:

- used in the study of nonlinear dependencies;
- limits: [0,1], and if is strictly equal to one, then there is an unambiguous functional dependence between the values.

Algorithm for finding a correlation relation:

1. Division of the correlation field by variable X into L grouping intervals of different lengths.
2. Search for "partial" mathematical expectations Y in each of the L selected groups $\overline{m}_{Y_j} = \frac{1}{n_j}\sum_{k=1}^{n_j} y_j, j = \overline{1,L}, k = \overline{1,n_j}, n_j$ is number of sample elements in jth grouping intervals.
3. Mathematical expectation search of partial groupings of reviews using $\overline{m}_Y = \frac{1}{n}\sum_{j=1}^{L} n_j * \overline{m}_{Y_j}$.
4. Calculation of group variance for variable y: $\sigma^2_{\overline{m}_Y} = \frac{1}{n}\sum_{j=1}^{L} n_j * \left(\overline{m}_{Y_j} - \overline{m}_Y\right)^2$.
5. Calculation of variance obtained from ungrouped response $\bar{\sigma}^2_Y = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \overline{m}_Y)^2$.
6. Construction of the correlation relation $\bar{\rho}_{Y*X} = \frac{\bar{\sigma}_{\overline{m}_Y}}{\bar{\sigma}_Y}$.

```
Created intervals:
      The beginning The end Length
 [1,]             1      19     19
 [2,]            20      30     11
 [3,]            31      49     19
 [4,]            50      60     11
 [5,]            61      79     19
 [6,]            80      90     11
 [7,]            91     109     19
 [8,]           110     120     11
 [9,]           121     139     19
[10,]           140     150     11
[11,]           151     169     19
[12,]           170     180     11
[13,]           181     199     19
[14,]           200     210     11
[15,]           211     212      2
        Partial expected values (m_y_j):
Interval 1: m_y_1 = 2177457
Interval 2: m_y_2 = 881761.4
Interval 3: m_y_3 = 497563.2
Interval 4: m_y_4 = 373424.1
Interval 5: m_y_5 = 310320.8
Interval 6: m_y_6 = 233906.5
Interval 7: m_y_7 = 192156.6
Interval 8: m_y_8 = 163432.9
Interval 9: m_y_9 = 160342.7
Interval 10: m_y_10 = 130232.3
Interval 11: m_y_11 = 126400.2
Interval 12: m_y_12 = 113049.7
Interval 13: m_y_13 = 146417.7
Interval 14: m_y_14 = 193193.1
Interval 15: m_y_15 = 351246
Total expected value: m_y = 435301.9
Group dispersion: (σ_m_y)^2 = 331672888069
Total dispersion: (σ_y)^2 = 368919205742
Correlation ratio: ρ_xy = 0.9481769
```

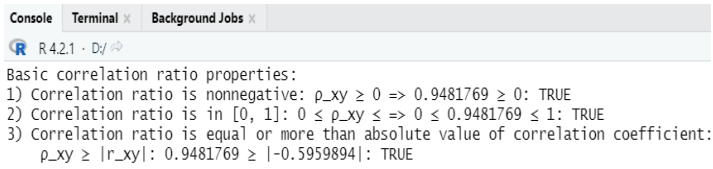

Figure 21: The result of searching for the correlation relation and checking the properties of the correlation relation

Properties of the correlation relation $\bar{\rho}_{X*Y} \geq 0$, $0 \leq \bar{\rho}_{X*Y} \leq 1$, $\rho_{X*Y} \geq |r_{X*Y}|$. The obtained value belongs to the space of permissible values for this indicator. 0.948 is a number close to 1, but not 1. Conclusion: there is a strong relationship between the values, but not an unambiguous dependence.

Correlation matrix:

- a table containing correlation coefficients when analyzing a large number of observations;
- a square table in which the correlation coefficient between the corresponding parameters is located at the intersection of the corresponding row and column.

```
Breaking sequence into 3 equal parts
      part_1 part_2 part_3
 [1,]  22740 300033 119758
 [2,] 2352003 282043 160343
 [3,] 3997584 273280 141980
 [4,] 4027485 279232 110384
 [5,] 3762020 351429 126528
 [6,] 3167455 371957 126866
 [7,] 3050607 272020 120272
 [8,] 2628486 254718 135559
 [9,] 2892043 267200 131626
[10,] 2396854 250351 124005
[11,] 1934266 259501 119036
[12,] 1982330 299676 106318
[13,] 1838330 284562 110939
[14,] 1466125 247941 108822
[15,] 1453713 238620 112776
[16,] 1067689 163192 147314
[17,] 1107978 226020 179954
[18,] 1085582 201747 141515
[19,] 1138400 189271 135196
[20,]  995092 212091 122608
[21,]  995767 199731 131833
[22,]  982095 179317 121159
[23,]  962171 207915 121857
[24,]  960853 207852 120998
[25,]  825088 242831 132187
[26,]  875569 235454 136826
[27,]  884823 206093 125369
[28,]  748688 186994 121519
[29,]  698314 194058 105378
[30,]  770915 166972 110189
[31,]  612233 188766  89380
[32,]  608478 168607  94308
[33,]  529504 200731  93538
[34,]  487217 170903 113157
[35,]  419553 175930 114333
[36,]  421240 183332 115054
[37,]  442421 182514 123191
[38,]  412778 176658 127020
[39,]  454576 176318 129536
[40,]  721685 175696 133841
[41,]  618069 164759 130967
[42,]  613541 128390 131378
[43,]  506522 169412 247779
[44,]  481950 200478 157018
[45,]  483729 178415 137166
[46,]  444231 151051 108005
[47,]  402032 145590 102978
[48,]  401888 166564 136040
[49,]  392054 158248 143758
[50,]  440644 159159 145760
[51,]  409549 179676 140161
[52,]  377294 162005 121972
[53,]  344037 137684 133991
[54,]  359673 182568 139605
[55,]  396723 216178 156857
[56,]  454303 246091 160933
[57,]  355061 240913 163553
[58,]  331034 169089 168072
[59,]  305095 163650 155944
[60,]  334252 145615 199259
[61,]  321408 135203 253597
[62,]  303711 139255 215892
[63,]  343281 141492 196044
[64,]  356886 125128 209552
[65,]  355179 127728 178638
[66,]  305894 133029 179858
[67,]  296992 119984 188446
[68,]  333819 135665 167805
[69,]  314230 145558 157851
[70,]  312784 135234 178182
```

Console | Terminal | Background Jobs

R 4.2.1 · D:/

```
Correlation matrix for these parts:
           part_1     part_2     part_3
part_1  1.0000000  0.7735046 -0.2698853
part_2  0.7735046  1.0000000 -0.3799561
part_3 -0.2698853 -0.3799561  1.0000000
```

Figure 22: The result of dividing the sequence into 3 equal parts and building a correlation matrix

Multiple correlation coefficients $r_{zxy} = \sqrt{\frac{r_{xz}^2 + r_{yz}^2 + 2r_{xy}r_{xz}r_{yz}}{1 - r_{xy}^2}}$.

```
Multiple correlation coefficient:  0.7739401 0.7936755 0.3818403
```

Figure 23: The multiple correlation coefficient was found

Calculation of autocorrelation is an indicator that characterizes the existence of dependence between the previous and next levels of the time sequence and is calculated according to the formula:

$$r(\tau) = \frac{\frac{1}{n-\tau}\sum_{t=1}^{n-\tau}(y_t - \bar{y}) * (y_{t+\tau} - \bar{y})}{\frac{1}{n-1}\sum_{t=1}^{n}(y_t - \bar{y})^2}.$$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Lag	1.0000000	2.0000000	3.0000000	4.0000000	5.0000000	6.0000000	7.0000000	8.0000000	9.0000000	10.0000000	11.0000000	12.0000000	13.000000	14.0000000	15.0000000	16.0000000	17.0000000	18.00000
Autocorrelation coefficient	0.9453164	0.8611884	0.7820488	0.7143672	0.6573266	0.6016455	0.5456268	0.4891999	0.4497458	0.4168458	0.3788123	0.3504834	0.330234	0.3151352	0.3073932	0.2948348	0.2825007	0.26929

Figure 24: A fragment of the autocorrelation table

Graphing the autocorrelation function

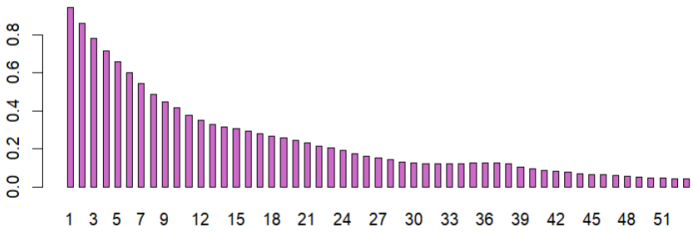

Figure 25: Correlogram of autocorrelation function

Looking at the time series graph, we can conclude that the data has a downward trend. Therefore, it is possible to assume the non-stationarity of the original time series.

To more accurately determine the stationarity of the series, the correlogram of the autocorrelation function is analyzed. In the case of a stationary time series, a rapid decline with increasing t will be depicted already after the first few values. The constructed correlogram demonstrates that the studied series is not stationary, but contains a trend component.

```
cat("\t\tCorrelation analysis\n")# correlational analysis
cor_field <- function(x, y, a){ # construction of the correlation field
 field <- plot(x, y, xlab = "Factor charasteristics X", ylab = "Resulting characteristics Y", main=a, pch=16, lwd=2, type="b", col="darkorchid4")
}
correlation_ratio <- function(x, y, n){ # function for calculating the correlation ratio
 m <- 15 # number of separation intervals
 a <- x[1]
```

```
 L <- matrix(NA, m, 3) # division of the correlation field by variable X into 15 parts of different lengths
 k <- round(n/m)
 for (i in 1:m){  # a random number from a range [10, 15]
  if (a != x[n]){
   if (i %% 2 == 0){
    L[i, 1] <-  a
    if(round(a + 0.7*k) <= x[n])# checking not to go beyond the initial X interval
     L[i, 2] <- round(a + 0.7*k)
    else
     L[i, 2] <- x[n]
    a <- L[i, 2] + 1
   }
   else
   {
    L[i, 1] <-  a
    if(round(a + 1.3*k) <= x[n]) # checking not to go beyond the initial X interval
     L[i, 2] <- round(a + 1.3*k)
    else
     L[i, 2] <- x[n]
    a <- L[i, 2] + 1
   }
  }
 }
 for(i in 1:m){     L[i, 3] <- L[i, 2]-L[i, 1] + 1  }# determination of interval lengths
 colnames(L) <- c("The beginning", "The end", "Length")
 cat("Created intervals:\n")  # displaying the formed intervals on the screen
 print(L)
 m_y_j <- vector()  # finding partial mathematical expectations for each interval
 for (i in 1:m){
  S <- 0
  for(j in L[i, 1]:L[i, 2]){     S <- S + y[j]    }
  m_y_j[i] <- S/L[i, 3]
 }
 cat("\tPartial expected values (m_y_j): \n")  # display of partial mathematical expectations m_y_j
 for(i in 1:m)
  cat("Interval ", i, ": m_y_", i, " = ", m_y_j[i],"\n", sep = "")
 S <- 0  # finding the mathematical expectation of partial groupings
 for (i in 1:m){  S <- S + m_y_j[i]*L[i, 3] }
 m_y <- S/n
 cat("Total expected value: m_y =", m_y)  # displaying the mathematical expectation m_y on the screen
 S <- 0  # calculation of the group variance of the variable y
 for (i in 1:m){ S <- S + L[i, 3]*(m_y_j[i] - m_y)^2 }
 D_m_y <- S/n  # displaying the calculated group variance on the screen
 cat("\nGroup dispersion: (σ_m_y)^2 =", D_m_y)
 S <- 0  # calculation of variance is not by grouped response
 for (i in 1:m){  S <- S + (y[i] - m_y)^2 }
 D_y <- S/n
 cat("\nTotal dispersion: (σ_y)^2 =", D_y)  # displaying the calculated group variance on the screen
 # finding a correlation relation for the dependent variable Y and the independent variable X
 r_xy <- sqrt(D_m_y)/sqrt(D_y)
 return(r_xy)
}
mult_cor_coefficient <- function(r, z, x, y){ # the function of determining the multiple correlation coefficient
 for (i in 1:3){  # z dependent variable, x, y - independent variables
  R_zxy <- sqrt((r[x, z]^2 + r[y, z]^2 - 2 * r[x, z] * r[y, z] * r[x, y])/(1 - r[x, y]^2))
 }
 return(R_zxy)
}
auto_cor_coeficient <- function(y, t, n){ # autocorrelation coefficient determination function
 S1 <- 0
 S2 <- 0
 for (i in 1:(n - t)){   S1 <- S1 + (y[i] - mean(y)) * (y[i + t] - mean(y))  }
 for (i in 1:n){   S2 <- S2 + (y[i] - mean(y))^2  }
 r <- ( (1/(n - t)) * S1 ) / ( (1/(n - 1)) * S2 )
 return(r)
}
table <- cbind(c(1:n), dataset[1], dataset[2]) # table formation
colnames(table) <- c("№", "Day of war", "Amount of tweets")
cor.test(table[, 1], table[, 3]) # checking for correlation
cor_field(table[, 1], table[, 3], "Correlation field")# construction of the correlation field
r <- correlation_coeficient(table[, 1], table[, 3], n) # finding the correlation coefficient
```

```
cat("Correlation coefficient: r_xy =", r, "\n")
R_xy <- correlation_ratio(table[, 1], table[, 3], n) # calculation of the correlation ratio
cat("\nCorrelation ratio: ρ_xy =", R_xy)
cat("\n\nBasic correlation ratio properties:") # checking the main properties of the correlation relation
# невід'ємність
cat("\n1) Correlation ratio is nonnegative: ρ_xy ≥ 0 =>", R_xy, "≥ 0:", R_xy >= 0)
cat("\n2) Correlation ratio is in [0, 1]: 0 ≤ ρ_xy ≤ =>", "0 ≤", R_xy, "≤ 1:", R_xy >= 0 || R_xy <= 1) # range of values from 0 to 1 inclusive
cat("\n3) Correlation ratio is equal or more than absolute value of correlation coefficient: # comparison with correlation coefficient
   ρ_xy ≥ |r_xy|: ", R_xy, " ≥ |", r, "|: ", R_xy >= abs(r), sep = "")
q <- 3 # splitting the sequence of tweets into 3 equal part
l <- (n - 2)/q
cor_table <- dataset$daily_tweets[1:210]
part_1 <- vector(length = l)
part_2 <- vector(length = l)
part_3 <- vector(length = l)
for (i in 1:(n - 2)){
 if (i <= l)    part_1[i] <- cor_table[i]
 if (i > l && i <= 2*l)    part_2[i - l] <- cor_table[i]
 else   part_3[i - 2*l] <- cor_table[i]
}
L <- cbind(part_1, part_2, part_3)
cat("\n\nBreaking sequence into 3 equal parts:\n")
print(L)
cor_matrix <- cor(L) # construction of the correlation matrix for parts of the sequence
cat("\nCorrelation matrix for these parts:\n")
print(cor_matrix )
R_part1 <- mult_cor_coefficient(cor_matrix, 1, 2, 3) # search for multiple correlation coefficients
R_part2 <- mult_cor_coefficient(cor_matrix, 2, 1, 3)
R_part3 <- mult_cor_coefficient(cor_matrix, 3, 1, 2)
R_part <- cbind(R_part1, R_part2, R_part3)
cat("\nMultiple correlation coefficient: ", R_part)
auto_cor_value <- matrix(NA, 2, as.integer(n/4)) # autocorrelation of time series
for (i in 1:as.integer(n/4)){
 auto_cor_value[1, i] <- as.integer(i)
 auto_cor_value[2, i] <- auto_cor_coeficient(table[, 3], i, n)
}
rownames(auto_cor_value) <- c("Lag", "Autocorrelation coefficient")
colnames(auto_cor_value) <- c(1:round(n/4))
View(auto_cor_value)
cat("\n\nAutocorrelation coefficient for different lags:\n")
print(t(auto_cor_value))
barplot(auto_cor_value[2, ], space = 1, names.arg = auto_cor_value[1, ],
      col = "orchid3",xlab = "Lag", ylab = "Autocorrelation coefficient",
      main = "Correlogram of autocorrelation function")
```

Construction of a correlation table for all smoothing intervals, including a number of original values:

	Y	Yn, w = 3	Yn, w = 5	Yn, w = 7	Yn, w = 9	Yn, w = 11	Yn, w = 13	Yn, w = 15
1	470	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	59184	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	151971	45591.300	0.000	0.000	0.000	0.000	0.000	0.000
4	105232	36128.730	0.000	0.000	0.000	0.000	0.000	0.000
5	86367	34082.103	17273.400	0.000	0.000	0.000	0.000	0.000
6	69978	28014.483	15722.940	0.000	0.000	0.000	0.000	0.000
7	67966	26599.459	16892.834	33983.000	0.000	0.000	0.000	0.000
8	49740	20383.394	14073.247	28268.300	0.000	0.000	0.000	0.000
9	48362	19206.885	14418.825	30406.130	24181.000	0.000	0.000	0.000
10	75941	26741.328	19668.196	47236.243	40388.600	0.000	0.000	0.000
11	65639	24286.521	18084.806	46808.867	39276.460	31506.720	0.000	0.000
12	54535	21463.285	16106.904	44238.604	37652.106	29327.472	0.000	0.000
13	40650	16769.981	13253.522	40306.549	34474.817	25532.406	19105.500	0.000
14	45967	17613.427	14017.093	43776.248	38767.223	30579.151	23515.040	0.000
15	52386	19154.141	14913.847	47588.146	42824.301	36100.001	28711.524	23573.700

Figure 26: A fragment of the generalized correlation table for smoothing intervals according to the formulas from Pollard

Returning to smoothing, we will modify the graph using Pollard's formulas, and also check their effectiveness (similarly to Kendall's formulas, which were presented above). Series for analysis: number of NATO hashtags. Smoothing according to the formulas from Pollard for different w (3, 5, 7, 9, 11, 13, 15):

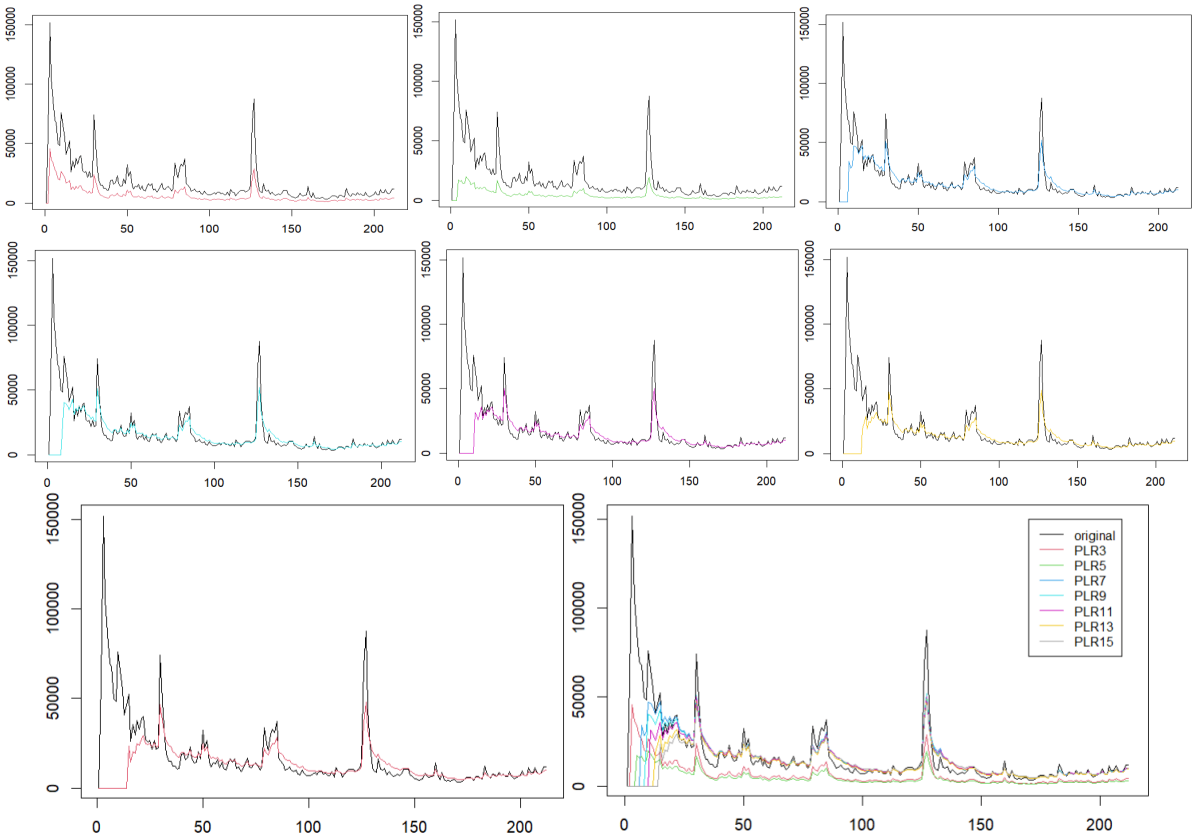

Figure 27: Pollard smoothing for w=3, 5, 7, 9, 11, 13, 15 and w=[3;15] (x – time, y – #NATO)

Plotting pivot points for all smoothing intervals:

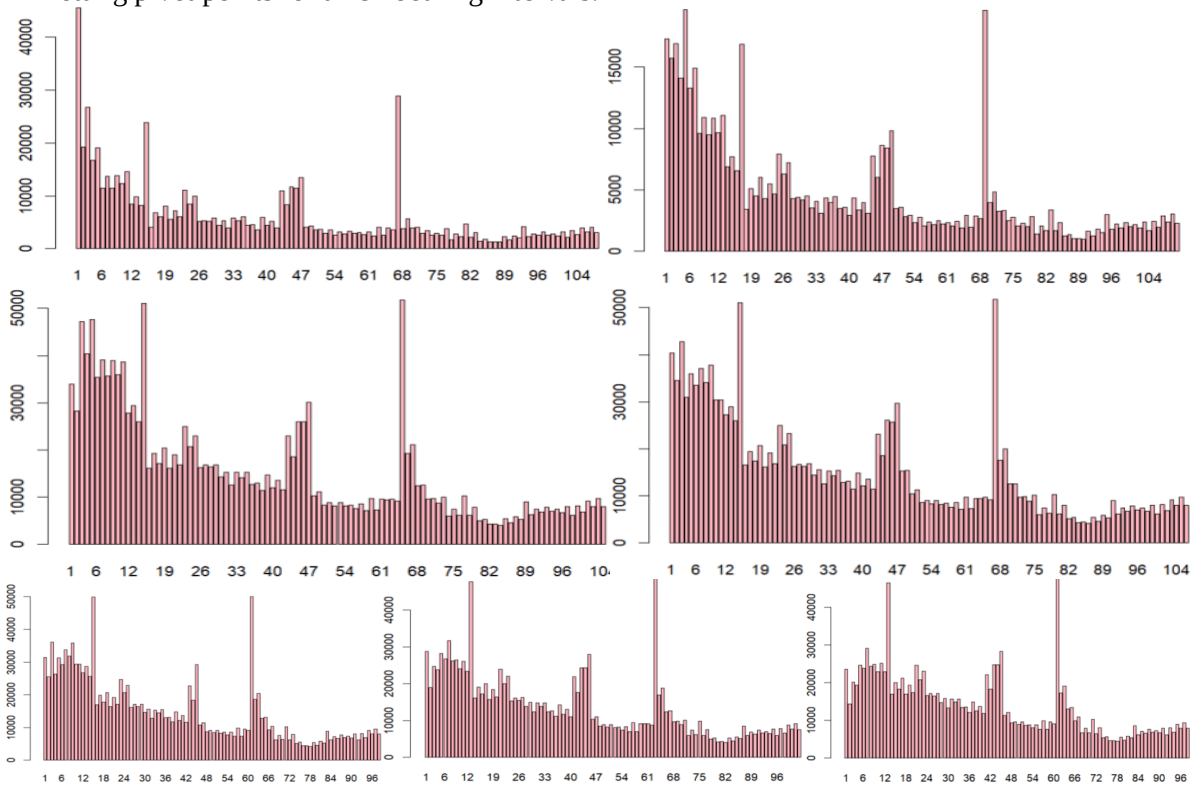

Figure 28: Turned points diagram, w=3, 5, 7, 9, 11, 13, 15

Determination of efficiency criteria for performed smoothing according to formulas from Pollard:

Table 3

Efficiency criteria for performed smoothing according to formulas from Pollard

Smoothing interval w	Number of turning points	Correlation coefficient r_{xy}
3	108	0.9670561
5	110	0.7033669
7	104	0.5275968
9	106	0.4533610
11	98	0.3531801
13	102	0.2738721
15	98	0.2028413

```
                Pollard smoothing
Pollard smoothing criteria:
     Smoothing parameter Amount of turned points Correlation coefficient
[1,]                   3                     108                0.9670561
[2,]                   5                     110                0.7033669
[3,]                   7                     104                0.5275968
[4,]                   9                     106                0.4533610
[5,]                  11                      98                0.3531801
[6,]                  13                     102                0.2738721
[7,]                  15                      98                0.2028413
```

Figure 29: The result of software finding the smoothing efficiency criteria according to the formulas from Pollard

As you can see in the graphs above, Pollard's smoothing also reduces the number of turning points, but unlike the previous smoothing results, it does not decrease as linearly. In contrast, the correlation coefficient when applying these formulas decreases more rapidly, but still linearly. Re-smoothing according to the formulas from Pollard for different w (3, 5, 7, 9, 11, 15):

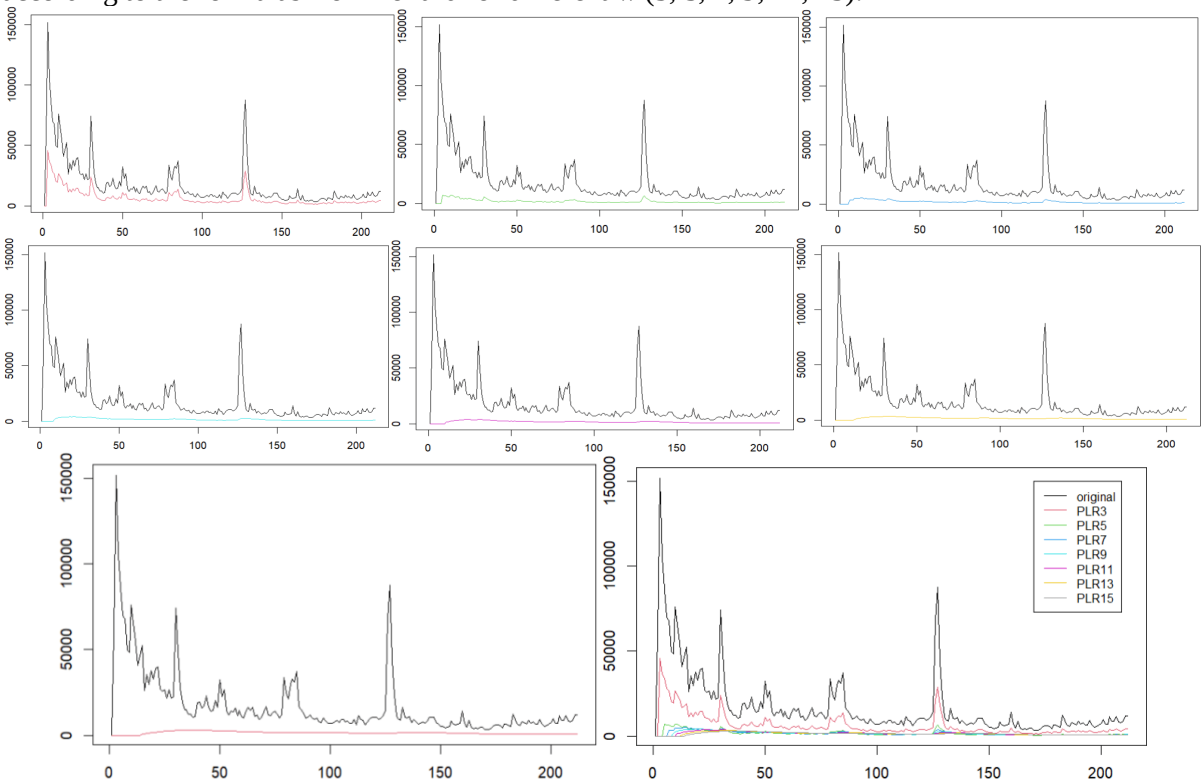

Figure 30: Repeated Pollard smoothing for w=3, 5, 7, 9, 11, 13, 15 and w=[3;15] (x – time, y – #NATO)

Construction of a correlation table for all re-smoothing intervals according to formulas from Pollard:

	Y	Y'n, w = 3	Y'n, w = 5	Y'n, w = 7	Y'n, w = 9	Y'n, w = 11	Y'n, w = 13	Y'n, w = 15
1	470	0.000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2	59184	0.000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
3	151971	45591.300	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
4	105232	36128.730	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
5	86367	34082.103	6816.4206	0.0000	0.0000	0.0000	0.0000	0.0000
6	69978	28014.483	6284.5387	0.0000	0.0000	0.0000	0.0000	0.0000
7	67966	26599.459	6629.9877	3314.9938	0.0000	0.0000	0.0000	0.0000
8	49740	20383.394	5708.9525	3185.9756	0.0000	0.0000	0.0000	0.0000
9	48362	19206.885	5730.3190	3515.2565	1757.6282	0.0000	0.0000	0.0000
10	75941	26741.328	7137.9191	4570.5821	2461.0539	0.0000	0.0000	0.0000
11	65639	24286.521	6761.0751	4839.2183	2841.4774	1363.9091	0.0000	0.0000
12	54535	21463.285	6254.5200	4904.1129	3158.0724	1652.2657	0.0000	0.0000
13	40650	16769.981	5298.9675	4757.4492	3400.5478	1931.1526	907.6417	0.0000
14	45967	17613.427	5372.9838	4928.6898	3694.4007	2262.0127	1153.9102	0.0000
15	52386	19154.141	5548.8030	5121.6405	3975.7371	2594.8428	1417.5625	637.9031

Figure 31: A fragment of the generalized correlation table for re-smoothing intervals according to the formulas from Pollard

Construction of turning point diagrams for all re-smoothing intervals using formulas from Pollard:

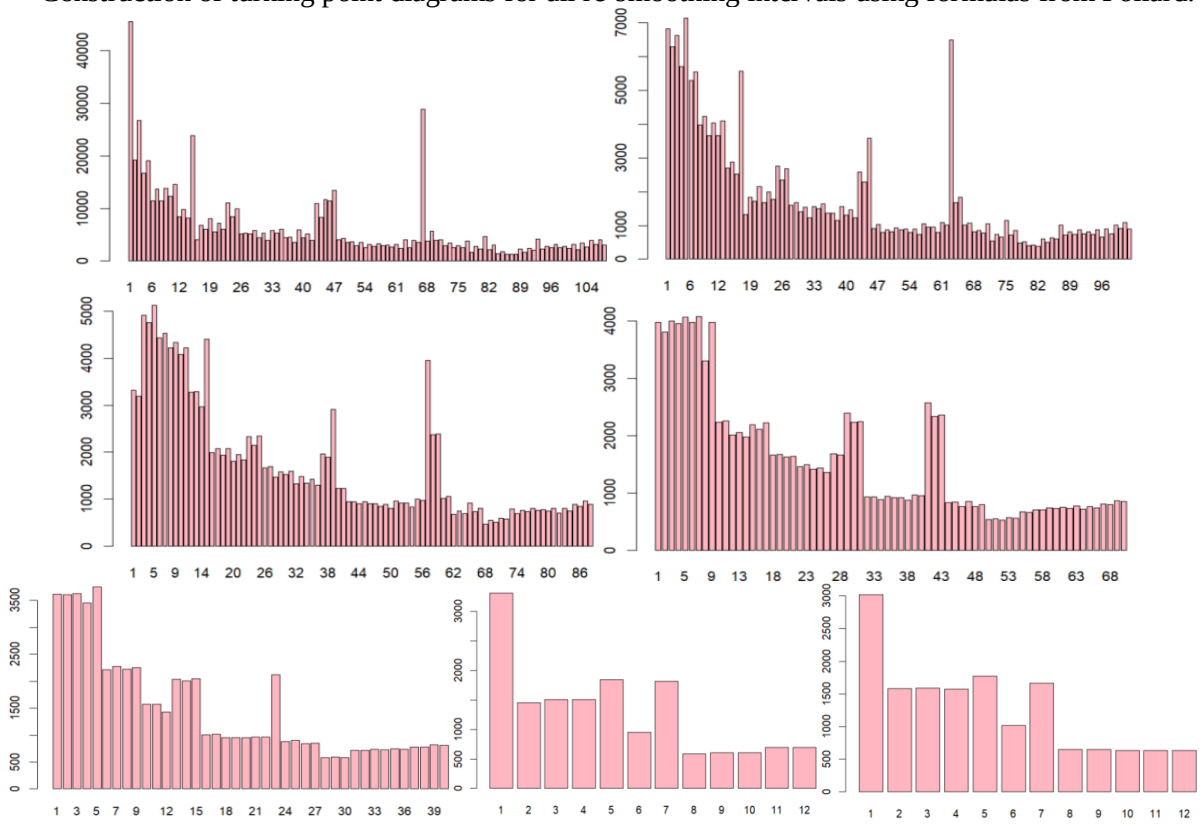

Figure 32: Turned points diagram, w=3, 5, 7, 9, 11, 13 and 15

Determination of efficiency criteria for repeated smoothings according to the formulas from Pollard:

Table 4

The table of criteria for the effectiveness of repeated smoothing according to formulas from Pollard

Smoothing interval w	Number of turning points	Correlation coefficient r_{xy}
3	108	0.9670561
5	102	0.6883252
7	88	0.4623179
9	70	0.2648956
11	40	0.0832726
13	12	-0.0565457
15	12	-0.1771463

```
     Smoothing parameter Amount of turned points Correlation coefficient
[1,]                   3                      108                0.9670561
[2,]                   5                      102                0.6883252
[3,]                   7                       88                0.4623179
[4,]                   9                       70                0.2648956
[5,]                  11                       40                0.0832726
[6,]                  13                       12               -0.0565457
[7,]                  15                       12               -0.1771463
```

Figure 33: The result of software finding the criteria for the effectiveness of repeated smoothing according to the formulas from Pollard

During repeated application of the Pollard method, the number of turning points began to drop rapidly and linearly, and the same applies to the correlation coefficient. The formulas "smoothed out" our graph so much that the relationship between the quantities changed its sign to the opposite. We can conclude that Pollard's repeated method for a given function has too strong an effect, and it is better to use previous methods (ordinary moving average or Kendall's method).

```
cat("\t\tPollard smoothing")# smoothing according to formulas from Pollard
pol_m <- function(data, w, a){
 if(w == length(a)){
  for(i in 1:length(data)){
   d <- 0
   if(i < w){
    data[i] <- 0
   }else{
    for (k in 0:(w-1)){
     data[i] <- (a[k+1] * data[i-k]) + d
     d <- data[i]
    }
   }
  }
  return(data)
 }
}
a3 <- c(0.3,0.1,0.1)
a5 <- c(0.2,0.1,0.1,0.05,0.05)
a7 <- c(0.5,0.1,0.1,0.1,0.1,0.05,0.05)
a9 <- c(0.5,0.1,0.1,0.1,0.1,0.025,0.025,0.025,0.025)
a11 <- c(0.48,0.1,0.098,0.098,0.08,0.075,0.025,0.02,0.02,0.002,0.002)
a13 <- c(0.47,0.1,0.091,0.09,0.075,0.06,0.025,0.02,0.02,0.01,0.008,0.007,0.002)
a15 <- c(0.45,0.11,0.11,0.091,0.08,0.05,0.02,0.02,0.019,0.01,0.01,0.01,0.008,0.007,0.005)
k1 <- pol_m(dataset$NATO, 3, a3)
k2 <- pol_m(dataset$NATO, 5, a5)
k3 <- pol_m(dataset$NATO, 7, a7)
k4 <- pol_m(dataset$NATO, 9, a9)
k5 <- pol_m(dataset$NATO, 11, a11)
k6 <- pol_m(dataset$NATO, 13, a13)
k7 <- pol_m(dataset$NATO, 15, a15)
plot.ts(dataset$NATO,main = 'Pollard Smoothing', ylab = '#NATO') # image of smoothing according to the formulas from Pollard on the graph
lines(k1, col='2')
lines(k2, col='3')
lines(k3, col='4')
lines(k4, col='5')
lines(k5, col='6')
lines(k6, col='7')
lines(k7, col='8')
legend(170, 150000,col = c('1','2','3','4','5', '6', '7', '8'),
    legend = c('original', 'PLR3', 'PLR5','PLR7', 'PLR9', 'PLR11', 'PLR13', 'PLR15'),lty = 1,cex = 0.8)
pollardSmoothing <- cbind(dataset$NATO, k1, k2, k3, k4, k5, k6, k7) # construction of a generalized correlation table
colnames(pollardSmoothing) <- c("Y", "Yn, w = 3", "Yn, w = 5", "Yn, w = 7", "Yn, w = 9", "Yn, w = 11", "Yn, w = 13", "Yn, w = 15")
View(pollardSmoothing)
p_t <- vector()# the number of turning points when smoothing according to formulas from Pollard
for (i in 1:7){  p_t[i] <- turn_points(pollardSmoothing[, i + 1], dim(pollardSmoothing)[1])
} # construction of diagrams of turning points for smoothing according to formulas from Pollard
barplot(turn_points_value(k1, length(k1)), col = "lightpink",  names.arg = c(1:p_t[1]), main = "Turned points diagram, w = 3")
barplot(turn_points_value(k2, length(k2)), col = "lightpink",  names.arg = c(1:p_t[2]), main = "Turned points diagram, w = 5")
barplot(turn_points_value(k3, length(k3)), col = "lightpink",  names.arg = c(1:p_t[3]), main = "Turned points diagram, w = 7")
barplot(turn_points_value(k4, length(k4)), col = "lightpink",  names.arg = c(1:p_t[4]), main = "Turned points diagram, w = 9")
barplot(turn_points_value(k5, length(k5)), col = "lightpink",  names.arg = c(1:p_t[5]), main = "Turned points diagram, w = 11")
barplot(turn_points_value(k6, length(k6)), col = "lightpink",  names.arg = c(1:p_t[6]), main = "Turned points diagram, w = 13")
```

```
barplot(turn_points_value(k7, length(k7)), col = "lightpink",  names.arg = c(1:p_t[7]), main = "Turned points diagram, w = 15")
p_r_xy <- vector()# correlation coefficients when smoothing according to formulas from Pollard
for (i in 1:7){  p_r_xy[i] <- correlation_coeficient(dataset$NATO, pollardSmoothing[, i + 1], dim(pollardSmoothing)[1])}
k <- c(3, 5, 7, 9, 11, 13, 15) # displaying on the screen the number of turning points and correlation coefficients at different w
plrd_matrix <- cbind(k, p_t, p_r_xy)
colnames(plrd_matrix) <- c("Smoothing parameter", "Amount of turned points", "Correlation coefficient")
cat("\nPollard smoothing criteria:\n")
print(plrd_matrix)
cat("\n\t\tRepeated Pollard smoothing")# re-smoothing according to formulas from Pollard
k1 <- pol_m(dataset$NATO, 3, a3)
k2 <- pol_m(k1, 5, a5)
k3 <- pol_m(k2, 7, a7)
k4 <- pol_m(k3, 9, a9)
k5 <- pol_m(k4, 11, a11)
k6 <- pol_m(k5, 13, a13)
k7 <- pol_m(k6, 15, a15)
plot.ts(dataset$NATO, main = "Repeated Pollard Smoothing", ylab = '#NATO') # image of re-smoothing using formulas from Pollard on graph
lines(k1, col='2')
lines(k2, col='3')
lines(k3, col='4')
lines(k4, col='5')
lines(k5, col='6')
lines(k6, col='7')
lines(k7, col='8')
legend(170, 150000,col = c('1', '2','3','4','5','6', '7', '8'),
     legend = c('original', 'PLR3', 'PLR5','PLR7', 'PLR9', 'PLR11', 'PLR13', 'PLR15'),lty = 1,cex = 0.8)
pollardSmoothing <- cbind(dataset$NATO, k1, k2, k3, k4, k5, k6, k7) # construction of a generalized correlation table
colnames(pollardSmoothing) <- c("Y", "Y'n, w = 3", "Y'n, w = 5", "Y'n, w = 7","Y'n, w = 9", "Y'n, w = 11", "Y'n, w = 13", "Y'n, w = 15")
View(pollardSmoothing)
rep_p_t <- vector()# the number of turning points during re-smoothing according to formulas from Pollard
for (i in 1:7){
  rep_p_t[i] <- turn_points(pollardSmoothing[, i + 1], dim(pollardSmoothing)[1])
}# construction of diagrams of turning points for re-smoothing according to formulas from Pollard
barplot(turn_points_value(k1, length(k1)), col = "lightpink", names.arg = c(1:rep_p_t[1]), main = "Turned points diagram, w = 3")
barplot(turn_points_value(k2, length(k2)), col = "lightpink", names.arg = c(1:rep_p_t[2]), main = "Turned points diagram, w = 5")
barplot(turn_points_value(k3, length(k3)), col = "lightpink", names.arg = c(1:rep_p_t[3]), main = "Turned points diagram, w = 7")
barplot(turn_points_value(k4, length(k4)), col = "lightpink", names.arg = c(1:rep_p_t[4]), main = "Turned points diagram, w = 9")
barplot(turn_points_value(k5, length(k5)), col = "lightpink", names.arg = c(1:rep_p_t[5]), main = "Turned points diagram, w = 11")
barplot(turn_points_value(k6, length(k6)), col = "lightpink", names.arg = c(1:rep_p_t[6]), main = "Turned points diagram, w = 13")
barplot(turn_points_value(k7, length(k7)), col = "lightpink", names.arg = c(1:rep_p_t[7]), main = "Turned points diagram, w = 15")
rep_p_r_xy <- vector()# correlation coefficients with repeated smoothing according to formulas from Pollard
for (i in 1:7){  rep_p_r_xy[i] <- correlation_coeficient(dataset$NATO, pollardSmoothing[, i + 1], dim(pollardSmoothing)[1])}
k <- c(3, 5, 7, 9, 11, 13, 15) # displaying on the screen the number of turning points and correlation coefficients at different w
rep_plrd_matrix <- cbind(k, rep_p_t, rep_p_r_xy)
colnames(rep_plrd_matrix) <- c("Smoothing parameter", "Amount of turned points", "Correlation coefficient")
cat("\nRepeated Pollard smoothing criteria:\n")
print(rep_plrd_matrix)
```

Exponential smoothing - the smoothed value is determined by only two values - the current and the last smoothed level and the ratio of their weights. Series for analysis: the share of #Ukraine hashtags among the total number of tweets. Carrying out exponential smoothing for different weights - α (0.1, 0.15, 0.2, 0.25, 0.3):

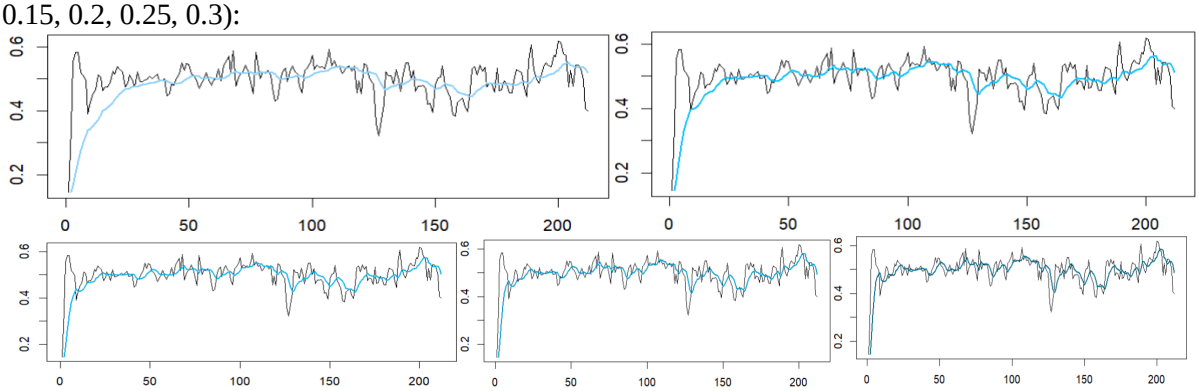

Figure 34: Exponential smoothing (α = 0.1, 0.15, 0.2, 0.25, 0.3) for #Ukraine hashtag share in all tweets

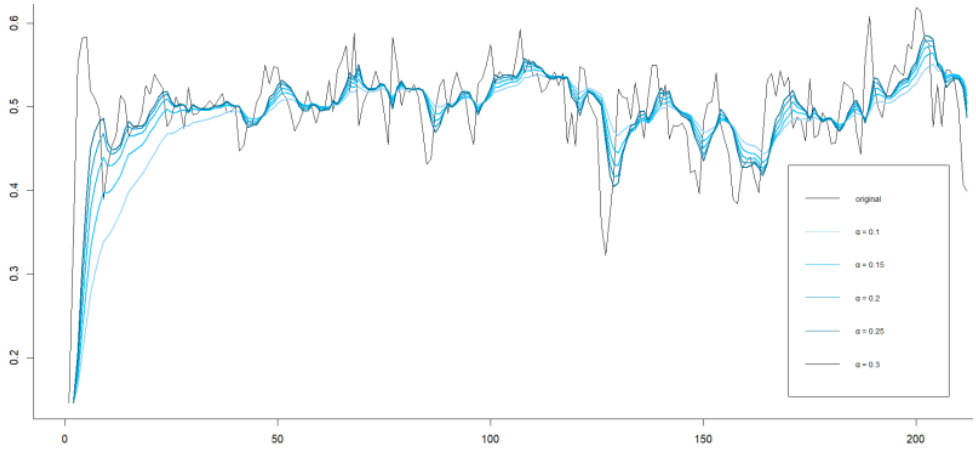

Figure 35: Exponential smoothing (α = [0.1;0.3]) for #Ukraine hashtag share in all tweets

Therefore, the smaller the α indicator is, the more the levels in the analyzed series are smoothed. We will conduct studies similar to those conducted for the Kendall and Pollard methods. Construction of a correlation table for all smoothing intervals, including a number of original values:

	Y	Yn, α = 0.1	Yn, α = 0.15	Yn, α = 0.2	Yn, α = 0.25	Yn, α = 0.3
1	0.1451187	0.1451187	0.1451187	0.1451187	0.1451187	0.1451187
2	0.3453962	0.1651465	0.1751604	0.1851742	0.1951881	0.2052020
3	0.5520259	0.2038344	0.2316902	0.2585446	0.2843976	0.3092492
4	0.5824409	0.2416951	0.2843028	0.3233238	0.3589084	0.3912067
5	0.5836128	0.2758868	0.3291993	0.3753816	0.4150845	0.4489285
6	0.5195092	0.3002491	0.3577458	0.4042071	0.4411907	0.4701027
7	0.5101522	0.3212394	0.3806068	0.4253962	0.4584311	0.4821176
8	0.4961883	0.3387343	0.3979440	0.4395546	0.4678704	0.4863388
9	0.3895440	0.3438153	0.3966840	0.4295525	0.4482888	0.4573004
10	0.4271236	0.3521461	0.4012499	0.4290667	0.4429975	0.4482473
11	0.4531440	0.3622459	0.4090340	0.4338822	0.4455341	0.4497163
12	0.4661933	0.3726406	0.4176079	0.4403444	0.4506989	0.4546594
13	0.5134263	0.3867192	0.4319807	0.4549608	0.4663808	0.4722895
14	0.5060414	0.3986514	0.4430898	0.4651769	0.4762959	0.4824151
15	0.4645339	0.4052397	0.4463064	0.4650483	0.4733554	0.4770507

Figure 36: A fragment of the generalized correlation table for all intervals of exponential smoothing

Plotting pivot points for all smoothing intervals:

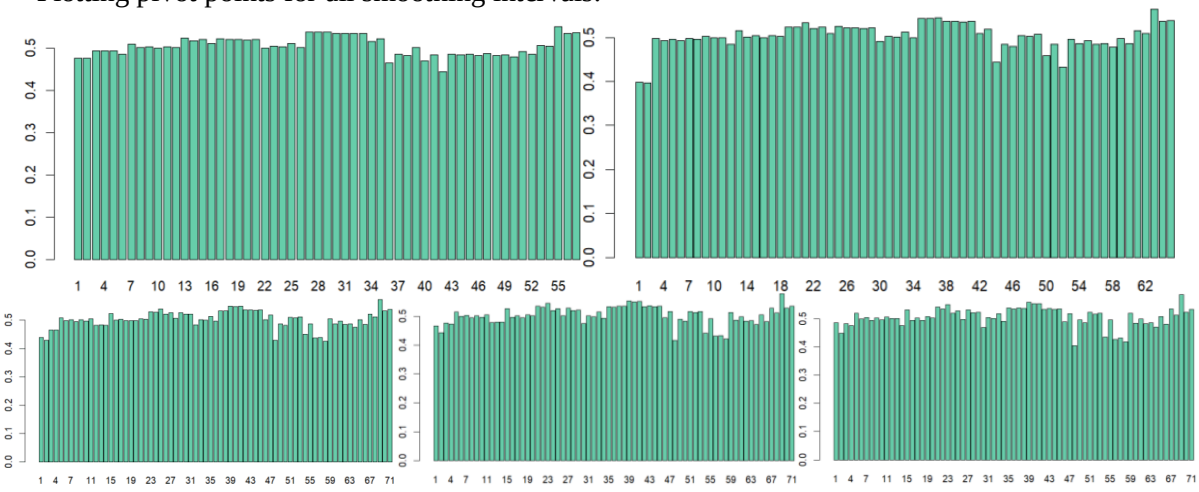

Figure 37: Turned points diagram, α = 0.1, 0.15, 0.2, 0.25, 0.3

Determination of performance criteria for exponential smoothing:

Table 5

Table of efficiency criteria of exponential smoothing

Parameter α 0,1 ≤ α ≤ 0,3	Number of turning points	Correlation coefficient r_{xy}
0.1	57	0.4009399
0.15	65	0.5053544
0.2	71	0.5937667
0.25	71	0.6687464
0.3	71	0.7317945

```
     Smoothing parameter Amount of turned points Correlation coefficient
[1,]                0.10                      57                0.4009399
[2,]                0.15                      65                0.5053544
[3,]                0.20                      71                0.5937667
[4,]                0.25                      71                0.6687464
[5,]                0.30                      71                0.7317945
```

Figure 38: The result of software finding the efficiency criteria of exponential smoothing

This table differs from the ones we observed before, because now it is the opposite: the smaller the parameter, the more the function is smoothed (the smaller the number of turning points and the correlation coefficient).

```
cat("\t\tExponential smoothing")# exponential smoothing
exp_1 <- HoltWinters(dataset$Ukraine/dataset$daily_tweets, alpha = 0.1, beta = FALSE, gamma = FALSE) # exponential smoothing at a = 0.1
t1 <- turn_points(exp_1$fitted[, 1], n - 1) # number of turning points at a = 0.1
r1_xy <- correlation_coeficient(dataset$Ukraine[1:(n-1)]/dataset$daily_tweets[1:(n-1)], exp_1$fitted[, 1], n - 1) # coefficient at a = 0.1
exp_2 <- HoltWinters(dataset$Ukraine/dataset$daily_tweets,  alpha = 0.15, beta = FALSE, gamma = FALSE) # smoothing at a = 0.15
t2 <- turn_points(exp_2$fitted[, 1], n - 1) # the number of turning points at a = 0.15
r2_xy <- correlation_coeficient(dataset$Ukraine[1:(n-1)]/dataset$daily_tweets[1:(n-1)], exp_2$fitted[, 1], n - 1) # coefficient at a = 0.15
exp_3 <- HoltWinters(dataset$Ukraine/dataset$daily_tweets, alpha = 0.2, beta = FALSE, gamma = FALSE) # exponential smoothing at a = 0.2
t3 <- turn_points(exp_3$fitted[, 1], n - 1) # the number of turning points at a = 0.2
r3_xy <- correlation_coeficient(dataset$Ukraine[1:(n-1)]/dataset$daily_tweets[1:(n-1)], exp_3$fitted[, 1], n - 1) # coefficient at a = 0.2
exp_4 <- HoltWinters(dataset$Ukraine/dataset$daily_tweets, alpha = 0.25, beta = FALSE, gamma = FALSE) # smoothing at a = 0.25
t4 <- turn_points(exp_4$fitted[, 1], n - 1) # the number of turning points at a = 0.25
r4_xy <- correlation_coeficient(dataset$Ukraine[1:(n-1)]/dataset$daily_tweets[1:(n-1)], exp_4$fitted[, 1], n - 1) # coefficient at a = 0.25
exp_5 <- HoltWinters(dataset$Ukraine/dataset$daily_tweets, alpha = 0.3, beta = FALSE, gamma = FALSE) # exponential smoothing at a = 0.3
t5 <- turn_points(exp_5$fitted[, 1], n - 1) # the number of turning points at a = 0.3
r5_xy <- correlation_coeficient(dataset$Ukraine[1:(n-1)]/dataset$daily_tweets[1:(n-1)], exp_5$fitted[, 1], n - 1) # coefficient at a = 0.3
plot.ts(dataset$Ukraine/dataset$daily_tweets, xlab = "Day of war", # an image of exponential smoothing on a graph
     ylab = "The share of #Ukraine hashtag in all tweets", main = "Exponential smoothing")
lines(exp_1$fitted[, 1], col = "lightskyblue", lwd = 2)
lines(exp_2$fitted[, 1], col = "deepskyblue1", lwd = 2)
lines(exp_3$fitted[, 1], col = "deepskyblue2", lwd = 2)
lines(exp_4$fitted[, 1], col = "deepskyblue3", lwd = 2)
lines(exp_5$fitted[, 1], col = "deepskyblue4", lwd = 2)
legend(170,0.43,col=c('black', 'lightskyblue','deepskyblue1','deepskyblue3','deepskyblue4'),
    legend=c('original', 'α = 0.1', 'α = 0.15','α = 0.2', 'α = 0.25','α = 0.3'),lty=1,cex=0.7)
exp_cor_table <- cbind(dataset$Ukraine[1:(n-1)]/dataset$daily_tweets[1:(n-1)],  exp_1$fitted[, 1], exp_2$fitted[, 1], exp_3$fitted[, 1],
          exp_4$fitted[, 1], exp_5$fitted[, 1]) # construction of a generalized correlation table
colnames(exp_cor_table) <- c("Y", "Yn, α = 0.1", "Yn, α = 0.15", "Yn, α = 0.2", "Yn, α = 0.25", "Yn, α = 0.3")
View(exp_cor_table) # plotting pivot points for exponential smoothing
barplot(turn_points_value(exp_cor_table[, 2], n - 1), col = "aquamarine3", names.arg = c(1:t1), main = "Turned points diagram, α = 0.1")
barplot(turn_points_value(exp_cor_table[, 3], n - 1), col = "aquamarine3", names.arg = c(1:t2), main = "Turned points diagram, α = 0.15")
barplot(turn_points_value(exp_cor_table[, 4], n - 1), col = "aquamarine3", names.arg = c(1:t3), main = "Turned points diagram, α = 0.2")
barplot(turn_points_value(exp_cor_table[, 5], n - 1), col = "aquamarine3", names.arg = c(1:t4), main = "Turned points diagram, α = 0.25")
barplot(turn_points_value(exp_cor_table[, 6], n - 1), col = "aquamarine3", names.arg = c(1:t5), main = "Turned points diagram, α = 0.3")
# виведення на екран кількості поворотних точок та коефіцієнтів кореляції при різних a
a <- c(0.1, 0.15, 0.2, 0.25, 0.3)
exp_r_xy <- c(r1_xy, r2_xy, r3_xy, r4_xy, r5_xy)
exp_t <- c(t1, t2, t3, t4, t5)
exp_matrix <- cbind(a, exp_t, exp_r_xy)
colnames(exp_matrix) <- c("Smoothing parameter", "Amount of turned points", "Correlation coefficient")
cat("\n Exponential smoothing criteria:\n")
print(exp_matrix)
```

Median smoothing is a type of smoothing based on the distributed average, which instead of the arithmetic mean (depending on the indicator) takes the value corresponding to the median of the interval (depends on the indicator). Performing median smoothing for different w (3, 5, 7, 9, 11, 13, 15), data for analysis: number of hashtags #NATO:

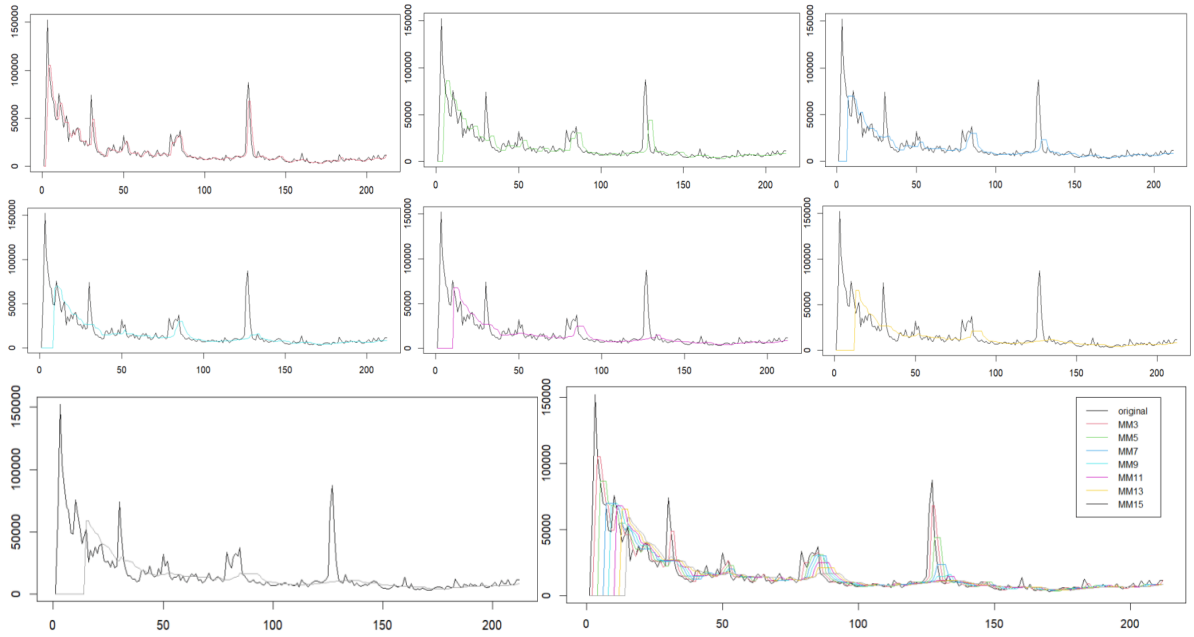

Figure 39: Moving median for w = 3, 5, 7, 9, 11, 13, 15 and w =[3;15] (x – time, y – #NATO)

Construction of a correlation table for all smoothing intervals, including a number of original values:

	Y	Yn, w = 3	Yn, w = 5	Yn, w = 7	Yn, w = 9	Yn, w = 11	Yn, w = 13	Yn, w = 15
1	470	0	0	0	0	0	0	0
2	59184	0	0	0	0	0	0	0
3	151971	59184	0	0	0	0	0	0
4	105232	105232	0	0	0	0	0	0
5	86367	105232	86367	0	0	0	0	0
6	69978	86367	86367	0	0	0	0	0
7	67966	69978	86367	69978	0	0	0	0
8	49740	67966	69978	69978	0	0	0	0
9	48362	49740	67966	69978	67966	0	0	0
10	75941	49740	67966	69978	69978	0	0	0
11	65639	65639	65639	67966	69978	67966	0	0
12	54535	65639	54535	65639	67966	67966	0	0
13	40650	54535	54535	54535	65639	67966	65639	0
14	45967	45967	54535	49740	54535	65639	65639	0
15	52386	45967	52386	52386	52386	54535	65639	59184

Figure 40: A fragment of the generalized correlation table for all median smoothing intervals

Plotting pivot points for all smoothing intervals:

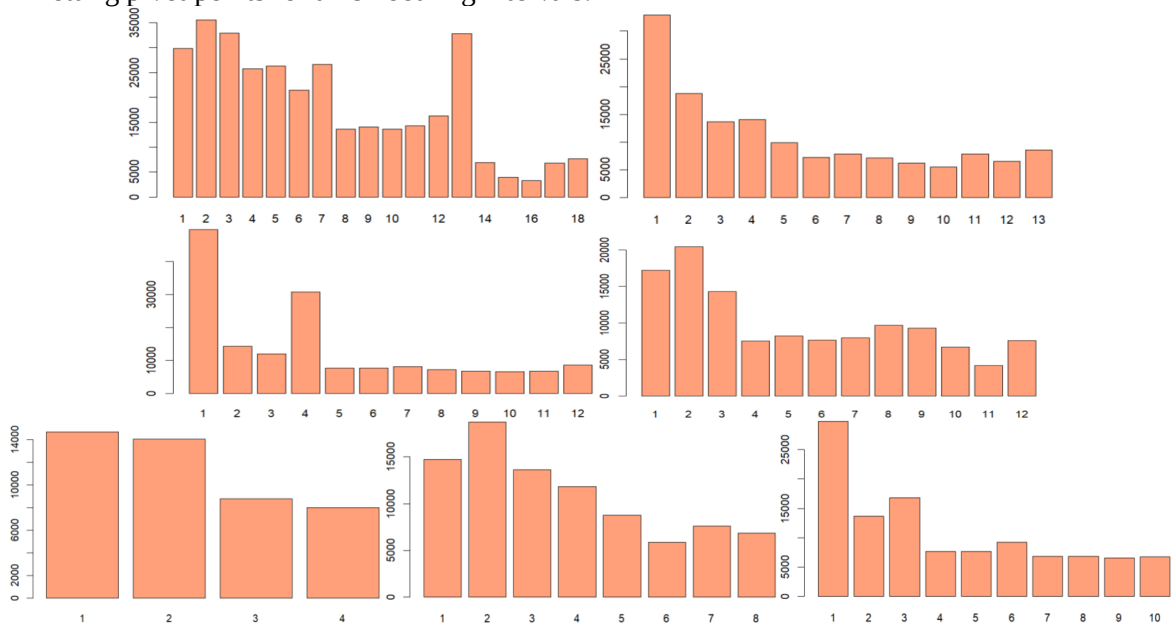

Figure 41: Turned points diagram, w=3, 5, 7, 9, 11, 13, 15

Determination of efficiency criteria for performed median smoothing:

Table 6
Table of criteria for efficiency of median smoothing

Smoothing interval w	Number of turning points	Correlation coefficient r_{xy}
3	18	0.8384473
5	13	0.5863781
7	12	0.4542598
9	12	0.3784868
11	4	0.2784487
13	8	0.1790616
15	10	0.1166237

```
     Smoothing parameter Amount of turned points Correlation coefficient
[1,]                   3                      18                0.8384473
[2,]                   5                      13                0.5863781
[3,]                   7                      12                0.4542598
[4,]                   9                      12                0.3784868
[5,]                  11                       4                0.2784487
[6,]                  13                       8                0.1790616
[7,]                  15                      10                0.1166237
```

Figure 42: The result of software finding criteria for the efficiency of median smoothing

As a conclusion: the larger the parameter, the more the function is smoothed. Given the given data, we can observe a slight deviation from the rule at parameter "15". We will consider this as an error of this method given the given input data.

Performing repeated median smoothing for different w (3, 5, 7, 9, 11, 15):

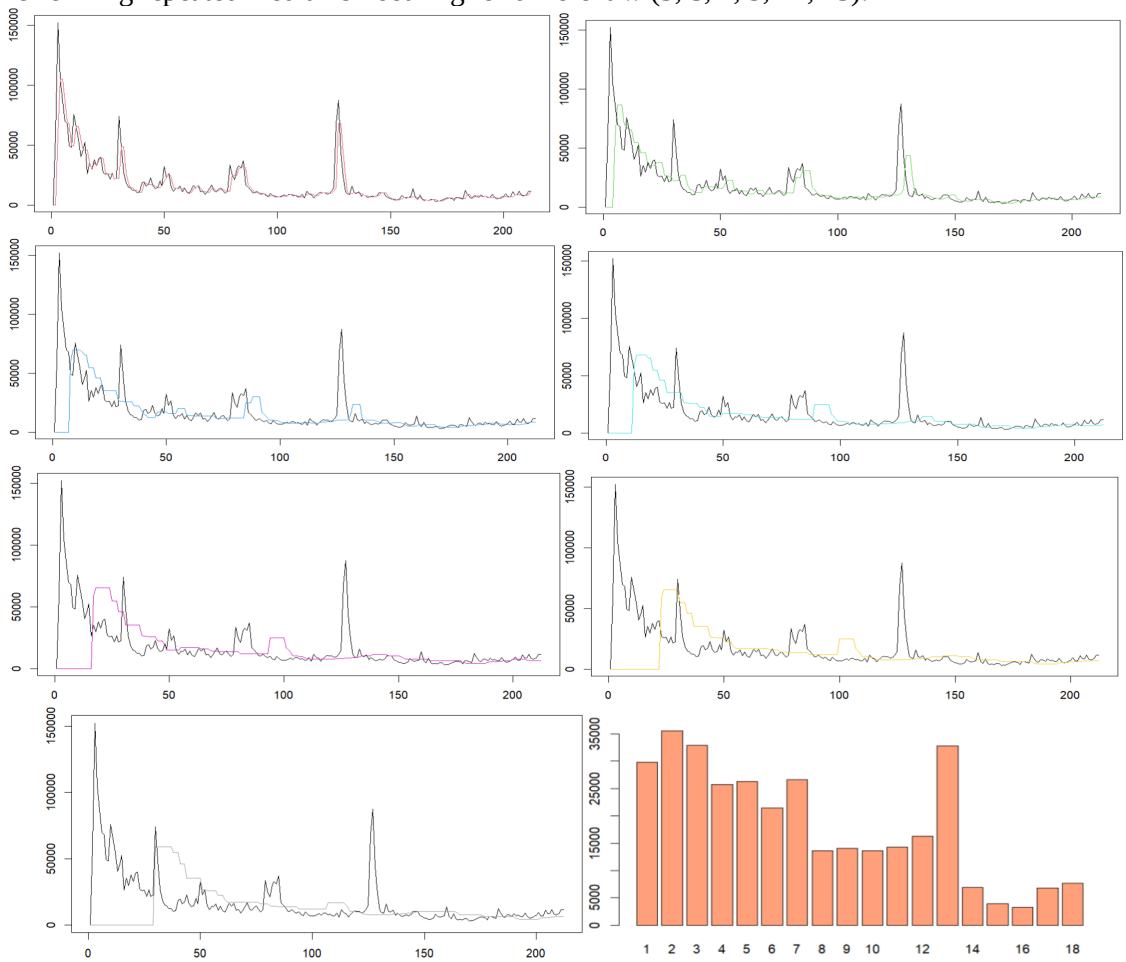

Figure 43: Repeated moving median for w=3, 5, 7, 9, 11, 13, 15 and Turned points diagram, w=3

Construction of a correlation table for all intervals of repeated median smoothing:

	Y	Y'n, w = 3	Y'n, w = 5	Y'n, w = 7	Y'n, w = 9	Y'n, w = 11	Y'n, w = 13	Y'n, w = 15
1	470	0	0	0	0	0	0	0
2	59184	0	0	0	0	0	0	0
3	151971	59184	0	0	0	0	0	0
4	105232	105232	0	0	0	0	0	0
5	86367	105232	59184	0	0	0	0	0
6	69978	86367	86367	0	0	0	0	0
7	67966	69978	86367	0	0	0	0	0
8	49740	67966	86367	59184	0	0	0	0
9	48362	49740	69978	69978	0	0	0	0
10	75941	49740	67966	69978	0	0	0	0
11	65639	65639	65639	69978	0	0	0	0
12	54535	65639	65639	69978	59184	0	0	0
13	40650	54535	54535	67966	67966	0	0	0
14	45967	45967	54535	65639	67966	0	0	0
15	52386	45967	54535	65639	67966	0	0	0

Figure 44: A fragment of the generalized correlation table for all repeated median smoothing intervals

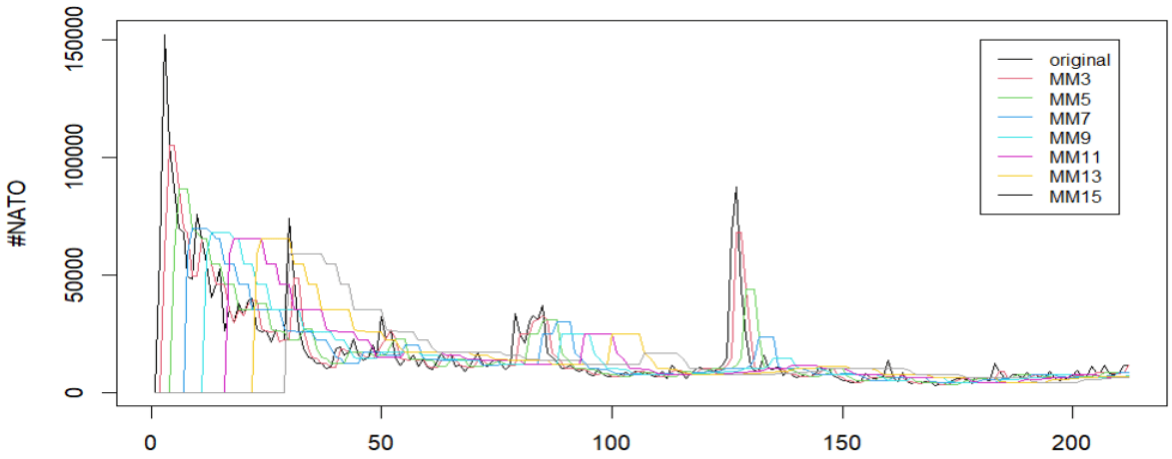

Figure 45: Repeated moving median for w =[3;15]

Determination of efficiency criteria for repeated median smoothing:

Table 7

Table of criteria for the efficiency of repeated median smoothing

Smoothing interval w	Number of turning points	Correlation coefficient r_{xy}
3	18	0.83844726
5	0	0.53877313
7	0	0.37528145
9	0	0.21193231
11	0	0.08120878
13	0	- 0.01236933
15	0	- 0.08042950

```
     Smoothing parameter Amount of turned points Correlation coefficient
[1,]                   3                      18                0.83844726
[2,]                   5                       0                0.53877313
[3,]                   7                       0                0.37528145
[4,]                   9                       0                0.21193231
[5,]                  11                       0                0.08120878
[6,]                  13                       0               -0.01236933
[7,]                  15                       0               -0.08042950
```

Figure 46: The result of the software finding criteria for the effectiveness of repeated median smoothing

The repeated median smoothing method, as we can see, like the Pollard method, was not very effective against our data, as the number of turning points variously fell to zero and the correlation coefficient became negative.

```
library("zoo")# median smoothing
cat("\t\tMedian smoothing")
my_movingMedian1 <- rollmedian(dataset$NATO, k = 3, fill= 0, align = "right")
my_movingMedian2 <- rollmedian(dataset$NATO, k = 5, fill = 0, align = "right")
my_movingMedian3 <- rollmedian(dataset$NATO, k = 7, fill = 0, align = "right")
my_movingMedian4 <- rollmedian(dataset$NATO, k = 9, fill = 0, align = "right")
my_movingMedian5 <- rollmedian(dataset$NATO, k = 11, fill = 0, align = "right")
my_movingMedian6 <- rollmedian(dataset$NATO, k = 13, fill = 0, align = "right")
my_movingMedian7 <- rollmedian(dataset$NATO, k = 15, fill = 0, align = "right")
plot.ts(dataset$NATO, main = "Moving median", ylab = "#NATO")# an image of median smoothing on a graph
lines(my_movingMedian1, col = "2")
lines(my_movingMedian2, col = "3")
lines(my_movingMedian3, col = "4")
lines(my_movingMedian4, col = "5")
lines(my_movingMedian5, col = "6")
lines(my_movingMedian6, col = "7")
lines(my_movingMedian7, col = "8")
legend(180,150000, col = c('1','2','3','4','5','6','7'),
     legend = c('original', 'MM3', 'MM5','MM7', 'MM9','MM11', 'MM13','MM15'),lty = 1, cex = 0.8)
movingMedian<- cbind(dataset$NATO, my_movingMedian1, my_movingMedian2, my_movingMedian3, my_movingMedian4,
                my_movingMedian5, my_movingMedian6, my_movingMedian7) # generalized correlation table from sliding medians
colnames(movingMedian) <- c("Y", "Yn, w = 3", "Yn, w = 5", "Yn, w = 7", "Yn, w = 9", "Yn, w = 11", "Yn, w = 13", "Yn, w = 15")
View(movingMedian)
m_t <- vector()# the number of turning points with median smoothing
for (i in 1:7){
  m_t[i] <- turn_points(movingMedian[, i + 1], dim(movingMedian)[1])
}# plotting pivot points for median smoothing
barplot(turn_points_value(my_movingMedian1, length(my_movingMedian1)), col = "lightsalmon",
      names.arg = c(1:m_t[1]), main = "Turned points diagram, w = 3")
barplot(turn_points_value(my_movingMedian2, length(my_movingMedian2)), col = "lightsalmon",
      names.arg = c(1:m_t[2]), main = "Turned points diagram, w = 5")
barplot(turn_points_value(my_movingMedian3, length(my_movingMedian3)), col = "lightsalmon",
      names.arg = c(1:m_t[3]), main = "Turned points diagram, w = 7")
barplot(turn_points_value(my_movingMedian4, length(my_movingMedian4)), col = "lightsalmon",
      names.arg = c(1:m_t[4]), main = "Turned points diagram, w = 9")
barplot(turn_points_value(my_movingMedian5, length(my_movingMedian5)), col = "lightsalmon",
      names.arg = c(1:m_t[5]), main = "Turned points diagram, w = 11")
barplot(turn_points_value(my_movingMedian6, length(my_movingMedian6)), col = "lightsalmon",
      names.arg = c(1:m_t[6]), main = "Turned points diagram, w = 13")
barplot(turn_points_value(my_movingMedian7, length(my_movingMedian7)), col = "lightsalmon",
      names.arg = c(1:m_t[7]), main = "Turned points diagram, w = 15")
m_r_xy <- vector()# correlation coefficients with median smoothing
for (i in 1:7){
  m_r_xy[i] <- correlation_coeficient(dataset$NATO, movingMedian[, i + 1], dim(movingMedian)[1])
}
k <- c(3, 5, 7, 9, 11, 13, 15) # displaying on the screen the number of turning points and correlation coefficients at different w
mdn_matrix <- cbind(k, m_t, m_r_xy)
colnames(mdn_matrix) <- c("Smoothing parameter", "Amount of turned points", "Correlation coefficient")
cat("\nMedian smoothing criteria:\n")
print(mdn_matrix)
cat("\n\t\tRepeated median smoothing")# repeated median smoothing
my_movingMedian1 <- rollmedian(dataset$NATO, k = 3, fill = 0,align = "right")
my_movingMedian2 <- rollmedian(my_movingMedian1, k = 5, fill = 0,align = "right")
my_movingMedian3 <- rollmedian(my_movingMedian2, k = 7, fill = 0,align = "right")
my_movingMedian4 <- rollmedian(my_movingMedian3, k = 9, fill = 0,align = "right")
my_movingMedian5 <- rollmedian(my_movingMedian4, k = 11, fill = 0,align = "right")
my_movingMedian6 <- rollmedian(my_movingMedian5, k = 13, fill = 0,align = "right")
my_movingMedian7 <- rollmedian(my_movingMedian6, k = 15, fill = 0,align = "right")
plot.ts(dataset$NATO,main="Repeated moving median",ylab="#NATO")# an image of repeated median smoothing on a graph
lines(my_movingMedian1, col = "2")
lines(my_movingMedian2, col = "3")
lines(my_movingMedian3, col = "4")
lines(my_movingMedian4, col = "5")
lines(my_movingMedian5, col = "6")
lines(my_movingMedian6, col = "7")
lines(my_movingMedian7, col = "8")
```

```
legend(180,150000,col = c('1','2','3','4','5','6','7'),legend =
       c('original', 'MM3', 'MM5','MM7', 'MM9','MM11', 'MM13','MM15'),lty = 1,cex = 0.8)
movingMedian<- cbind(dataset$NATO,my_movingMedian1,my_movingMedian2, my_movingMedian3,my_movingMedian4
                     ,my_movingMedian5,my_movingMedian6,my_movingMedian7) # generalized correlation table from sliding repeated medians
colnames(movingMedian) <- c("Y", "Y'n, w = 3", "Y'n, w = 5", "Y'n, w = 7", "Y'n, w = 9", "Y'n, w = 11", "Y'n, w = 13", "Y'n, w = 15")
View(movingMedian)
rep_m_t <- vector()# the number of turning points during repeated median smoothing
for (i in 1:7){
 rep_m_t[i] <- turn_points(movingMedian[, i + 1], dim(movingMedian)[1])
}
barplot(turn_points_value(my_movingMedian1, length(my_movingMedian1)), col = "lightsalmon",
      names.arg = c(1:rep_m_t[1]), main = "Turned points diagram, w = 3")# plotting pivot points for repeated median smoothing
rep_m_r_xy <- vector()# коефіцієнти кореляції при повторному медіанному згладжуванні
for (i in 1:7){
 rep_m_r_xy[i] <- correlation_coeficient(dataset$NATO, movingMedian[, i + 1], dim(movingMedian)[1])
}
k <- c(3, 5, 7, 9, 11, 13, 15) # displaying on the screen the number of turning points and correlation coefficients at different w
rep_mdn_matrix <- cbind(k, rep_m_t, rep_m_r_xy)
colnames(rep_mdn_matrix) <- c("Smoothing parameter", "Amount of turned points", "Correlation coefficient")
cat("\nRepeated median smoothing criteria:\n")
print(rep_mdn_matrix)
```

Hierarchical agglomerative cluster analysis of multivariate data:

- solves the problem of group homogeneity of data, ensures the selection of compact, distant groups of objects, that is, looks for a "natural" division of the population into areas of accumulation of objects;
- allows dividing objects not by one parameter, but by a whole set of features;
- allows you to view fairly significant volumes of data, sharply shorten and compress them, make them compact and clear.

For the method is necessary:

- normalize the data (so that the variance is equal to 1);
- make a matrix of closeness (relationship of the form "indicator-indicator");
- choose a strategy of unification.

Let's build the "operator-individual indicators" table. Dimensions: $n \times m$, where $n = \overline{1,9}$ is the number of hashtags, and$m = \overline{1,11}$ is the number of descriptive statistics indicators used. For cluster analysis: the set G , which includes m objects, each of which is characterized by n features.

	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
Ukraine	218002.11	343803.61	102657.0	135736.818	63797.019	3300	2345772	2342472	4.002991	18.06601	23612.529
Russia	76491.29	95887.61	41951.0	54255.594	24768.316	3058	659808	656750	3.374894	13.28812	6585.588
StandWithUkraine	49080.83	87354.85	19680.0	29995.200	14023.172	440	660822	660382	4.692934	26.13242	5999.556
Putin	36974.48	59639.86	16620.0	22337.424	10100.212	2224	406355	404131	3.716964	14.76966	4096.082
UkraineRussiaWar	28976.79	63444.17	11736.5	16160.788	6799.945	0	510488	510488	5.848403	37.27886	4357.364
NATO	17204.54	19334.69	10394.5	12834.865	5707.269	470	151971	151501	3.254004	13.87084	1327.912
StopRussia	15967.20	31107.25	5011.5	8530.429	4420.372	2	236950	236948	4.324306	22.49984	2136.455
Russian	13736.26	17464.54	7887.0	10019.671	5415.938	318	154006	153688	4.269705	25.08915	1199.470
StopPutin	12960.83	37107.50	1909.0	4033.300	1255.762	15	304824	304809	4.994643	28.21328	2548.554

Figure 47: Table "operator - indicator" - values of indicators of objects according to descriptive statistics

Normalization of the object-property table:

	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
Ukraine	1.000000000	1.000000000	1.00000000	1.00000000	1.00000000	1.0000000000	1.0000000000	1.0000000000	0.28869399	0.19915569	1.000000000
Russia	0.309842282	0.240311603	0.39744710	0.38132842	0.37595268	0.9266666667	0.2314872680	0.2306050605	0.04659653	0.00000000	0.240311603
StandWithUkraine	0.176159674	0.214164699	0.17639060	0.19712382	0.20414380	0.1333333333	0.2319494795	0.2322627730	0.55462931	0.53538595	0.214164699
Putin	0.117116178	0.129237706	0.14601779	0.13897976	0.14141785	0.6739393939	0.1159558228	0.1153050406	0.17844593	0.06175441	0.129237706
UkraineRussiaWar	0.078110894	0.140895265	0.09754536	0.09208173	0.08864841	0.0000000000	0.1634227535	0.1638483576	1.00000000	1.00000000	0.140895265
NATO	0.020696868	0.005730686	0.08422500	0.06682862	0.07117712	0.1424242424	0.0000000000	0.0000000000	0.00000000	0.02428920	0.005730686
StopRussia	0.014662280	0.041805305	0.03079466	0.03414586	0.05060035	0.0006060606	0.0387359656	0.0389996034	0.41254330	0.38396984	0.041805305
Russian	0.003781843	0.000000000	0.05933617	0.04545338	0.06651890	0.0963636364	0.0009276138	0.0009981876	0.39149746	0.49189953	0.000000000
StopPutin	0.000000000	0.060191862	0.00000000	0.00000000	0.00000000	0.0045454545	0.0696749614	0.0699726286	0.67092165	0.62212187	0.060191862

Figure 48: Normalized table "operator - indicator"

Choosing a metric for building a proximity matrix at Euclidean metric:

$$D_E(\vec{x}_1, \vec{x}_2) = \sqrt{\sum_{i=1}^{N}(x_{1j} - x_{2j})^2}.$$

	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
Ukraine	1.000000000	1.000000000	1.00000000	1.00000000	1.00000000	1.0000000000	1.0000000000	1.0000000000	0.28869399	0.19915569	1.000000000
Russia	0.309842282	0.240311603	0.39744710	0.38132842	0.37595268	0.9266666667	0.2314872680	0.2306050605	0.04659653	0.00000000	0.240311603
StandWithUkraine	0.176159674	0.214164699	0.17639060	0.19712382	0.20414380	0.1333333333	0.2319494795	0.2322627730	0.55462931	0.53538595	0.214164699
Putin	0.117116178	0.129237706	0.14601779	0.13897976	0.14141785	0.6739393939	0.1159558228	0.1153050406	0.17844593	0.06175441	0.129237706
UkraineRussiaWar	0.078110894	0.140895265	0.09754536	0.09208173	0.08864841	0.0000000000	0.1634227535	0.1638483576	1.00000000	1.00000000	0.140895265
NATO	0.020696868	0.005730686	0.08422500	0.06682862	0.07117712	0.1424242424	0.0000000000	0.0000000000	0.00000000	0.02428920	0.005730686
StopRussia	0.014662280	0.041805305	0.03079466	0.03414586	0.05060035	0.0006060606	0.0387359656	0.0389996034	0.41254330	0.38396984	0.041805305
Russian	0.003781843	0.000000000	0.05933617	0.04545338	0.06651890	0.0963636364	0.0009276138	0.0009981876	0.39149746	0.49189953	0.000000000
StopPutin	0.000000000	0.060191862	0.00000000	0.00000000	0.00000000	0.0045454545	0.0696749614	0.0699726286	0.67092165	0.62212187	0.060191862

Figure 49: The "original table" was built

	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
Ukraine	1.000000000	1.000000000	1.00000000	1.00000000	1.00000000	1.0000000000	1.0000000000	1.0000000000	0.28869399	0.19915569	1.000000000
Russia	0.309842282	0.240311603	0.39744710	0.38132842	0.37595268	0.9266666667	0.2314872680	0.2306050605	0.04659653	0.00000000	0.240311603
StandWithUkraine	0.176159674	0.214164699	0.17639060	0.19712382	0.20414380	0.1333333333	0.2319494795	0.2322627730	0.55462931	0.53538595	0.214164699
Putin	0.117116178	0.129237706	0.14601779	0.13897976	0.14141785	0.6739393939	0.1159558228	0.1153050406	0.17844593	0.06175441	0.129237706
UkraineRussiaWar	0.078110894	0.140895265	0.09754536	0.09208173	0.08864841	0.0000000000	0.1634227535	0.1638483576	1.00000000	1.00000000	0.140895265
NATO	0.020696868	0.005730686	0.08422500	0.06682862	0.07117712	0.1424242424	0.0000000000	0.0000000000	0.00000000	0.02428920	0.005730686
StopRussia	0.014662280	0.041805305	0.03079466	0.03414586	0.05060035	0.0006060606	0.0387359656	0.0389996034	0.41254330	0.38396984	0.041805305
Russian	0.003781843	0.000000000	0.05933617	0.04545338	0.06651890	0.0963636364	0.0009276138	0.0009981876	0.39149746	0.49189953	0.000000000
StopPutin	0.000000000	0.060191862	0.00000000	0.00000000	0.00000000	0.0045454545	0.0696749614	0.0699726286	0.67092165	0.62212187	0.060191862

Figure 50: Built "copy table"

Formation of the proximity table according to the defined metric:

	Ukraine	Russia	StandWithUkraine	Putin	UkraineRussiaWar	NATO	StopRussia	Russian	StopPutin
Ukraine	0.000000	2.0129736	2.4462875	2.4910462	2.8881866	2.8909688	2.9114586	2.9272550	2.9686466
Russia	2.012974	0.0000000	1.1426592	0.5920176	1.7624964	1.0991498	1.3189270	1.2993285	1.5115275
StandWithUkraine	2.446287	1.1426592	0.0000000	0.8434325	0.7016520	0.9115220	0.5405982	0.5601943	0.5295263
Putin	2.491046	0.5920176	0.8434325	0.0000000	1.4223876	0.6294907	0.8258859	0.8114353	1.0453304
UkraineRussiaWar	2.888187	1.7624964	0.7016520	1.4223876	0.0000000	1.4376388	0.8880631	0.8602348	0.5600772
NATO	2.890969	1.0991498	0.9115220	0.6294907	1.4376388	0.0000000	0.5741790	0.6127830	0.9269777
StopRussia	2.911459	1.3189270	0.5405982	0.8258859	0.8880631	0.5741790	0.0000000	0.1701307	0.3619027
Russian	2.927255	1.2993285	0.5601943	0.8114353	0.8602348	0.6127830	0.1701307	0.0000000	0.3608628
StopPutin	2.968647	1.5115275	0.5295263	1.0453304	0.5600772	0.9269777	0.3619027	0.3608628	0.0000000

Figure 51: Proximity table built according to the Euclidean metric

General view of any strategy:

$$d_{hk} = \alpha_i d_{hi} + \alpha_j d_{hj} + \beta d_{ij} + \gamma \left| d_{hi} - d_{hj} \right|.$$

For our data, we chose the nearest neighbor strategy: distance between groups - the distance between the two most distant elements of the groups. For her, the parameters acquire the following values: $\alpha_i = \alpha_j = 0.5, \beta = 0, \gamma = 0.5$. Features of the strategy: monotonous, greatly stretches the space.

Carrying out cluster analysis. Finding the smallest value in the proximity matrix and combining the objects it corresponds to into one group.

Table 8

Finding the smallest element in the proximity matrix

№	1	2	3	4	5	6	7	8	9
	Ukraine	russia	StandWithUkraine	putin	UkraineRussiaWar	NATO	StopRussia	russian	StopPutin
Ukraine	0.000000	2.0129736	2.4462875	2.4910462	2.8881866	2.8909688	2.9114586	2.9272550	2.9686466
russia	2.012974	0.0000000	1.1426592	0.5920176	1.7624964	1.0991498	1.3189270	1.2993285	1.5115275
StandWithUkraine	2.446287	1.1426592	0.0000000	0.8434325	0.7016520	0.9115220	0.5405982	0.5601943	0.5295263
putin	2.491046	0.5920176	0.8434325	0.0000000	1.4223876	0.6294907	0.8258859	0.8114353	1.0453304
UkraineRussiaWar	2.888187	1.7624964	0.7016520	1.4223876	0.0000000	1.4376388	0.8880631	0.8602348	0.5600772
NATO	2.890969	1.0991498	0.9115220	0.6294907	1.4376388	0.0000000	0.5741790	0.6127830	0.9269777
StopRussia	2.911459	1.3189270	0.5405982	0.8258859	0.8880631	0.5741790	0.0000000	0.1701307	0.3619027
russian	2.927255	1.2993285	0.5601943	0.8114353	0.8602348	0.6127830	0.1701307	0.0000000	0.3608628
StopPutin	2.968647	1.5115275	0.5295263	1.0453304	0.5600772	0.9269777	0.3619027	0.3608628	0.0000000

Extracting columns belonging to these objects. Eliminate empty space by shifting all columns to the left and rows up.

Table 9

Elimination of an empty space in the proximity matrix

№	1	2	3	4	5	6		9	№
	Ukraine	russia	StandWithUkraine	putin	UkraineRussiaWar	NATO	StopRussia & russian	StopPutin	
Ukraine	0.000000	2.0129736	2.4462875	2.4910462	2.8881866	2.8909688		2.9686466	Ukraine
russia	2.012974	0.0000000	1.1426592	0.5920176	1.7624964	1.0991498		1.5115275	Russia
StandWithUkraine	2.446287	1.1426592	0.0000000	0.8434325	0.7016520	0.9115220		0.5295263	StandWithUkraine
putin	2.491046	0.5920176	0.8434325	0.0000000	1.4223876	0.6294907		1.0453304	Putin
UkraineRussiaWar	2.888187	1.7624964	0.7016520	1.4223876	0.0000000	1.4376388		0.5600772	UkraineRussiaWar
NATO	2.890969	1.0991498	0.9115220	0.6294907	1.4376388	0.0000000		0.9269777	NATO
StopRussia & russian									StopRussia & Russian
StopPutin	2.968647	1.5115275	0.5295263	1.0453304	0.5600772	0.9269777		0.0000000	StopPutin

Enumeration of the value of the extracted columns according to the selected strategy:

$$d_{hk} = \alpha_i d_{hi} + \alpha_j d_{hj} + \beta d_{ij} + \gamma |d_{hi} - d_{hj}|.$$

The nearest neighbor strategy: $\alpha_i = \alpha_j = 0.5, \beta = 0, \gamma = 0.5$.

Table 10

Columns whose elements are to be enumerated

StopRussia	russian
2.9114586	2.9272550
1.3189270	1.2993285
0.5405982	0.5601943
0.8258859	0.8114353
0.8880631	0.8602348
0.5741790	0.6127830
0.0000000	0.1701307
0.1701307	0.0000000
0.3619027	0.3608628

$$d_{hk} = 0.5 * d_{hi} + 0.5 * d_{hj} + 0 * d_{ij} + 0.5 * |d_{hi} - d_{hj}|.$$
$$d_{h1} = 0.5 * 2.9114586 + 0.5 * 2.9272550 + 0.5 * |2.9114586 - 2.9272550| = 2.927255.$$
$$d_{h2} = 0.5 * 1.3189270 + 0.5 * 1.2993285 + 0.5 * |1.3189270 - 1.2993285| = 1.318927.$$
$$d_{h3} = 0.5 * 0.5405982 + 0.5 * 0.5601943 + 0.5 * |0.5405982 - 0.5601943| = 0.5601943.$$
$$d_{h4} = 0.5 * 0.8258859 + 0.5 * 0.8114353 + 0.5 * |0.8258859 - 0.8114353| = 0.8258859.$$
$$d_{h5} = 0.5 * 0.8880631 + 0.5 * 0.8602348 + 0.5 * |0.8880631 - 0.8602348| = 0.8880631.$$
$$d_{h6} = 0.5 * 0.5741790 + 0.5 * 0.6127830 + 0.5 * |0.5741790 - 0.6127830| = 0.612783.$$
$$d_{h7} = 0.5 * 0.0000000 + 0.5 * 0.1701307 + 0.5 * |0.0000000 - 0.1701307| = 0.1701307.$$
$$d_{h8} = 0.5 * 0.1701307 + 0.5 * 0.1701307 + 0.5 * |0.1701307 - 0.0000000| = 0.1701307.$$
$$d_{h8} = 0.5 * 0.3619027 + 0.5 * 0.3608628 + 0.5 * |0.3619027 - 0.3608628| = 0.3619027.$$

Search for the two smallest values in the listed column. Replacement of the upper minimum with zero, elimination of the lower by shifting up all the lower cells in these columns.

Table 11

The results of applying the selected strategy to the removed columns

StopRussia & russian
2.9272550
1.3189270
0.5601943
0.8258859
0.8880631
0.6127830
0.1701307
0.1701307
0.3619027

Table 12

Forming a null element in the listed column

StopRussia & russian
2.9272550
1.3189270
0.5601943
0.8258859
0.8880631
0.6127830
0.0000000
0.3619027

Insertion of the listed column in the place (empty) of the first removed column. Checking whether its zero lies on the main diagonal.

Table 13

Proximity matrix after inserting an enumerated column in place of an empty column

№	1	2	3	4	5	6		9
	Ukraine	russia	StandWithUkraine	putin	UkraineRussiaWar	NATO	StopRussia & russian	StopPutin
Ukraine	0.000000	2.0129736	2.4462875	2.4910462	2.8881866	2.8909688	2.9272550	2.9686466
russia	2.012974	0.0000000	1.1426592	0.5920176	1.7624964	1.0991498	1.3189270	1.5115275
StandWithUkraine	2.446287	1.1426592	0.0000000	0.8434325	0.7016520	0.9115220	0.5601943	0.5295263
putin	2.491046	0.5920176	0.8434325	0.0000000	1.4223876	0.6294907	0.8258859	1.0453304
UkraineRussiaWar	2.888187	1.7624964	0.7016520	1.4223876	0.0000000	1.4376388	0.8880631	0.5600772
NATO	2.890969	1.0991498	0.9115220	0.6294907	1.4376388	0.0000000	0.6127830	0.9269777
StopRussia & russian							0.0000000	
StopPutin	2.968647	1.5115275	0.5295263	1.0453304	0.5600772	0.9269777	0.3619027	0.0000000

Copying the values of this column, transposing them into a ribbon and replacing it with the ribbon of the first removed column.

Table 14

Proximity matrix after empty row filling

№	1	2	3	4	5	6		9
	Ukraine	russia	StandWithUkraine	putin	UkraineRussiaWar	NATO	StopRussia & russian	StopPutin
Ukraine	0.000000	2.0129736	2.4462875	2.4910462	2.8881866	2.8909688	2.9272550	2.9686466
russia	2.012974	0.0000000	1.1426592	0.5920176	1.7624964	1.0991498	1.3189270	1.5115275
StandWithUkraine	2.446287	1.1426592	0.0000000	0.8434325	0.7016520	0.9115220	0.5601943	0.5295263
putin	2.491046	0.5920176	0.8434325	0.0000000	1.4223876	0.6294907	0.8258859	1.0453304
UkraineRussiaWar	2.888187	1.7624964	0.7016520	1.4223876	0.0000000	1.4376388	0.8880631	0.5600772
NATO	2.890969	1.0991498	0.9115220	0.6294907	1.4376388	0.0000000	0.6127830	0.9269777
StopRussia & russian	2.9272550	1.3189270	0.5601943	0.8258859	0.8880631	0.6127830	0.0000000	0.3619027
StopPutin	2.968647	1.5115275	0.5295263	1.0453304	0.5600772	0.9269777	0.3619027	0.0000000

Assignment to the new object formed as a result of merging the removed objects, next in order of number.

Table 15

Assigning the next sequence number to the new proximity matrix object

№	1	2	3	4	5	6	10	9
	Ukraine	russia	StandWithUkraine	putin	UkraineRussiaWar	NATO	StopRussia & russian	StopPutin
Ukraine	0.000000	2.0129736	2.4462875	2.4910462	2.8881866	2.8909688	2.9272550	2.9686466
russia	2.012974	0.0000000	1.1426592	0.5920176	1.7624964	1.0991498	1.3189270	1.5115275
StandWithUkraine	2.446287	1.1426592	0.0000000	0.8434325	0.7016520	0.9115220	0.5601943	0.5295263
putin	2.491046	0.5920176	0.8434325	0.0000000	1.4223876	0.6294907	0.8258859	1.0453304
UkraineRussiaWar	2.888187	1.7624964	0.7016520	1.4223876	0.0000000	1.4376388	0.8880631	0.5600772
NATO	2.890969	1.0991498	0.9115220	0.6294907	1.4376388	0.0000000	0.6127830	0.9269777
StopRussia & russian	2.9272550	1.3189270	0.5601943	0.8258859	0.8880631	0.6127830	0.0000000	0.3619027
StopPutin	2.968647	1.5115275	0.5295263	1.0453304	0.5600772	0.9269777	0.3619027	0.0000000

Repeating the procedure until the matrix is reduced to size 2×2.

Table 16

K Step #2: proximity matrix by size 8×8

№	1	2	3	4	5	6	10	9
	Ukraine	russia	StandWithUkraine	putin	UkraineRussiaWar	NATO	StopRussia & russian	StopPutin
Ukraine	0.000000	2.0129736	2.4462875	2.4910462	2.8881866	2.8909688	2.9272550	2.9686466
russia	2.012974	0.0000000	1.1426592	0.5920176	1.7624964	1.0991498	1.3189270	1.5115275
StandWithUkraine	2.446287	1.1426592	0.0000000	0.8434325	0.7016520	0.9115220	0.5601943	0.5295263
putin	2.491046	0.5920176	0.8434325	0.0000000	1.4223876	0.6294907	0.8258859	1.0453304
UkraineRussiaWar	2.888187	1.7624964	0.7016520	1.4223876	0.0000000	1.4376388	0.8880631	0.5600772
NATO	2.890969	1.0991498	0.9115220	0.6294907	1.4376388	0.0000000	0.6127830	0.9269777
StopRussia & russian	2.9272550	1.3189270	0.5601943	0.8258859	0.8880631	0.6127830	0.0000000	0.3619027
StopPutin	2.968647	1.5115275	0.5295263	1.0453304	0.5600772	0.9269777	0.3619027	0.0000000

Table 17

Step #3: proximity matrix by size 7×7

№	1	2	3	4	5	6	11
	Ukraine	russia	StandWithUkraine	putin	UkraineRussiaWar	NATO	StopRussia & russian & StopPutin
Ukraine	0.000000	2.0129736	2.4462875	2.4910462	2.8881866	2.8909688	2.9686466
russia	2.012974	0.0000000	1.1426592	0.5920176	1.7624964	1.0991498	1.5115275
StandWithUkraine	2.446287	1.1426592	0.0000000	0.8434325	0.7016520	0.9115220	0.5601943
putin	2.491046	0.5920176	0.8434325	0.0000000	1.4223876	0.6294907	1.0453304
UkraineRussiaWar	2.888187	1.7624964	0.7016520	1.4223876	0.0000000	1.4376388	0.8880631
NATO	2.890969	1.0991498	0.9115220	0.6294907	1.4376388	0.0000000	0.9269777
StopRussia & russian & StopPutin	2.9686466	1.5115275	0.5601943	1.0453304	0.8880631	0.9269777	0.0000000

Table 18

Step #4: proximity matrix by size 6×6

№	1	2	12	4	5	6
	Ukraine	russia	StandWithUkraine & StopRussia & russian & StopPutin	putin	UkraineRussiaWar	NATO
Ukraine	0.000000	2.0129736	2.9686466	2.4910462	2.8881866	2.8909688
russia	2.012974	0.0000000	1.5115275	0.5920176	1.7624964	1.0991498
StandWithUkraine & StopRussia & russian & StopPutin	2.968647	1.5115275	0.0000000	1.0453304	0.8880631	0.9269777
putin	2.491046	0.5920176	1.0453304	0.0000000	1.4223876	0.6294907
UkraineRussiaWar	2.888187	1.7624964	0.8880631	1.4223876	0.0000000	1.4376388
NATO	2.890969	1.0991498	0.9269777	0.6294907	1.4376388	0.0000000

Table 19

Step #5: proximity matrix by size 5×5

№	1	13	12	5	6
	Ukraine	russia & putin	StandWithUkraine & StopRussia & russian & StopPutin	UkraineRussiaWar	NATO
Ukraine	0.000000	2.491046	2.9686466	2.8881866	2.8909688
russia & putin	2.491046	0.000000	1.511528	1.762496	1.099150
StandWithUkraine & StopRussia & russian & StopPutin	2.968647	1.511528	0.0000000	0.8880631	0.9269777
UkraineRussiaWar	2.888187	1.762496	0.8880631	0.0000000	1.4376388
NATO	2.890969	1.099150	0.9269777	1.4376388	0.0000000

Table 20

Step #6: proximity matrix by size 4×4

№	1	13	14	6
	Ukraine	russia & putin	StandWithUkraine & StopRussia & russian & StopPutin & UkraineRussiaWar	NATO
Ukraine	0.000000	2.491046	2.968647	2.8909688
russia & putin	2.491046	0.000000	1.762496	1.099150
StandWithUkraine & StopRussia & russian & StopPutin & UkraineRussiaWar	2.968647	1.762496	0.000000	1.437639
NATO	2.890969	1.099150	1.437639	0.0000000

Table 21

Step #7: proximity matrix by size 3 × 3

№	1	15	14
	Ukraine	russia & putin & NATO	StandWithUkraine & StopRussia & russian & StopPutin & UkraineRussiaWar
Ukraine	0.000000	2.890969	2.968647
russia & putin & NATO	2.890969	0.000000	1.762496
StandWithUkraine & StopRussia & russian & StopPutin & UkraineRussiaWar	2.968647	1.762496	0.000000

Table 22

Step #8: proximity matrix by size 2 × 2

№	1	16
	Ukraine	russia & putin & NATO & StandWithUkraine & StopRussia & russian & StopPutin & UkraineRussiaWar
Ukraine	0.000000	2.968647
russia & putin & NATO & StandWithUkraine & StopRussia & russian & StopPutin & UkraineRussiaWar	2.968647	0.000000

Table 23

Results of cluster analysis - table "union - node - metric"

Step	Association	Node	Metrics
1	7 + 8	d10	0.1701307
2	10 + 9	d11	0.3619027
3	2 + 4	d12	0.5601943
4	3 + 11	d13	0.5920176
5	13 + 5	d14	0.8880631
6	12 + 6	d15	1.099150
7	15 + d14	d16	1.762496
8	1 + 16	d17	2.968647

Step	Unification	Node	Metric
1	StopRussia & Russian	d10	0.170130730957401
2	StopRussia & Russian & StopPutin	d11	0.361902734013184
3	StandWithUkraine & StopRussia & Russian & StopPu...	d12	0.560194309353352
4	Russia & Putin	d13	0.59201763526809
5	StandWithUkraine & StopRussia & Russian & StopPu...	d14	0.88806314332077
6	Russia & Putin & NATO	d15	1.09914979181058
7	Russia & Putin & NATO & StandWithUkraine & StopR...	d16	1.76249641442001
8	Ukraine&Russia & Putin & NATO & StandWithUkrain...	d17	2.96864655832415

Figure 52: The result of the software construction of the "union - node - metric" table

The dimensionality of the matrix 2 × 2 => STOP. Construction of a dendrogram:

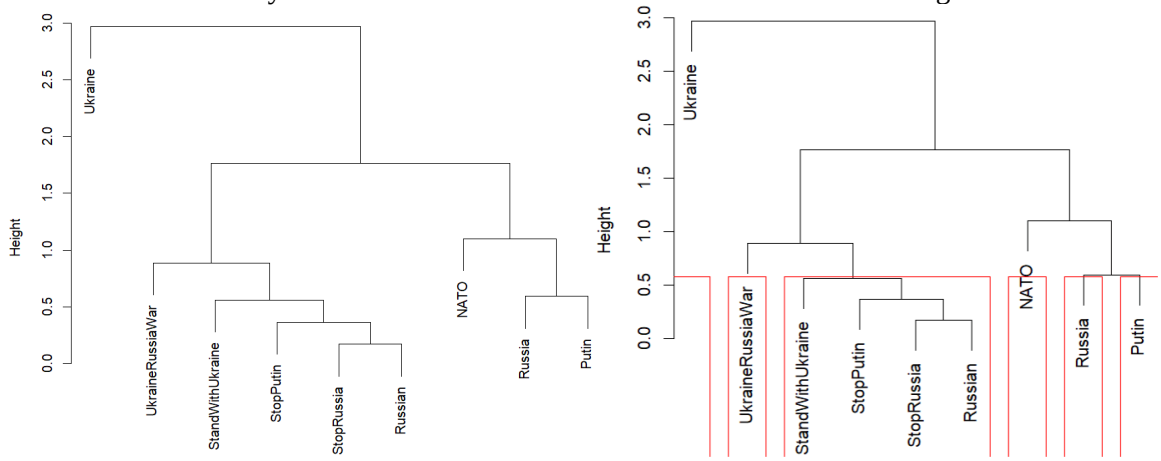

Figure 53: Cluster dendrogram for formed clusters and clusters at level 0.6

Interpretation of the result of cluster analysis at level 0.6, 6 clusters are formed: 1 cluster – object Ukraine; 2nd cluster – object Russia; cluster 3 – objects StandWithUkraine, StopPutin, StopRussia, russian; cluster 4 – putin facility; 5th cluster – object UkraineRussiaWar; cluster 6 is a NATO facility.

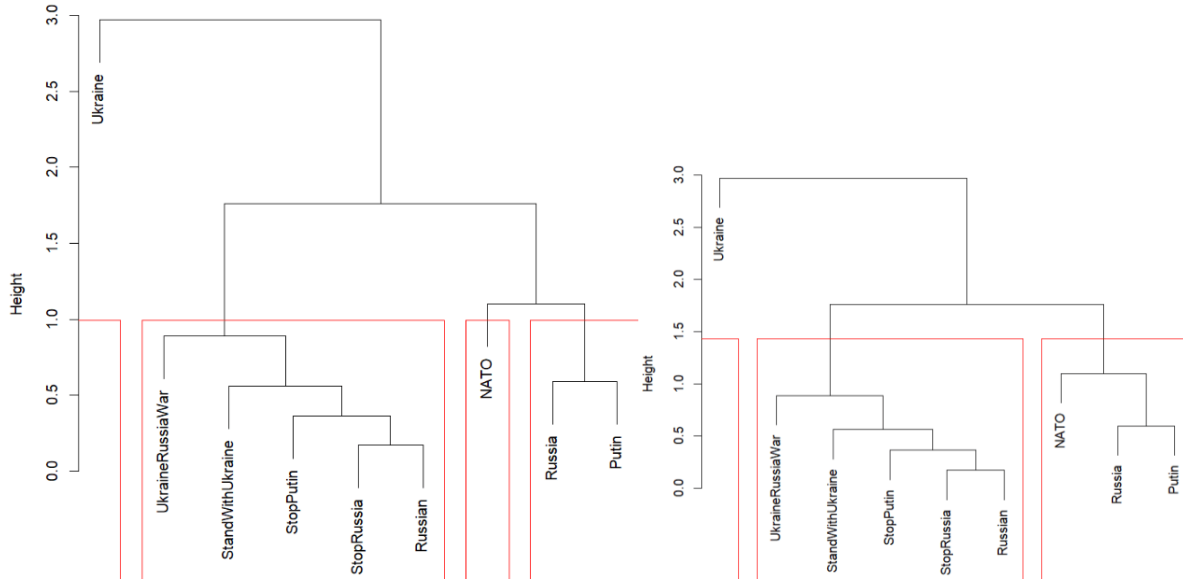

Figure 54: Cluster dendrogram for clusters at level 1.0 and 1.4

At level 1.0, 4 clusters are formed: 1 cluster – object Ukraine; 2nd cluster – objects russia, putin; cluster 3 – objects UkraineRussiaWar, StandWithUkraine, StopPutin, StopRussia, russian; cluster 4 is a NATO facility. At level 1.4, 3 clusters are formed: 1 cluster – object Ukraine; cluster 2 – NATO, russia, putin facilities; 3rd cluster - objects of UkraineRussiaWar, StandWithUkraine, StopPutin, StopRussia, russian.

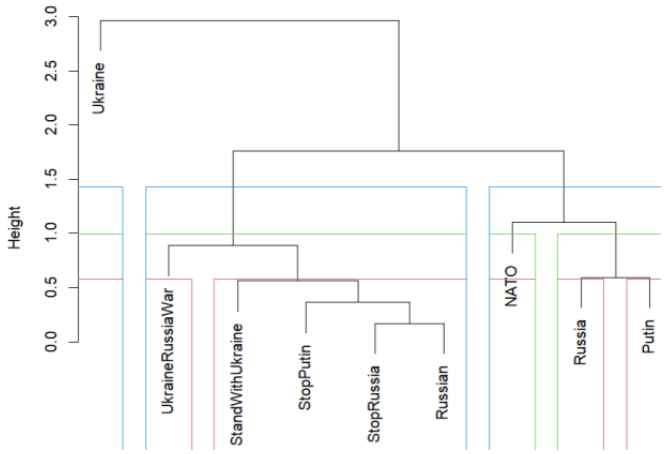

Figure 55: Cluster dendrogram for clusters at three levels simultaneously (0.6, 1.0, 1.4)

	cluster division on level '0.6'	cluster division on level '1'	cluster division on level '1.4'
Ukraine	1	1	1
Russia	2	2	2
StandWithUkraine	3	3	3
Putin	4	2	2
UkraineRussiaWar	5	3	3
NATO	6	4	2
StopRussia	3	3	3
Russian	3	3	3
StopPutin	3	3	3

Figure 56: The result of the software implementation of the interpretation of the cluster analysis result

As a general conclusion of the results of the cluster analysis, the following can be defined: the largest share of all hashtags was occupied by #Ukraine, followed by #NATO and #UkraineRussiaWar, respectively. Detailed information about which indicators apply to which clusters at which level is provided above.

```
cat("\tCluster analysis\n")# cluster analysis of multivariate data
operator_property <- describe(dataset[3:11])[3:13] # formation of the "object-property" table
norm_operator_property <- matrix(NA, dim(operator_property)[1], dim(operator_property)[2]) # object-property table normalization
View(operator_property)
for (j in 1:dim(operator_property)[2])
  norm_operator_property[, j] <- norm_sequence(operator_property[, j])
colnames(norm_operator_property) <- colnames(operator_property)
row.names(norm_operator_property) <- row.names(operator_property)
View(norm_operator_property)
norm_op_pr_1 <- norm_operator_property # creation of "original table" and "copy table"
norm_op_pr_2 <- norm_operator_property
View(norm_op_pr_1)
View(norm_op_pr_2)
proximity_matrix <- matrix(NA, dim(operator_property)[1], dim(operator_property)[1]) # building the proximity matrix
colnames(proximity_matrix) <- row.names(operator_property)
rownames(proximity_matrix) <- row.names(operator_property)
for (i in 1:dim(operator_property)[1]){ # cycle through the rows of the original table
 # цикл по рядках таблиці-копії
 for(j in 1:dim(operator_property)[1]){
   S  <- 0
   for(k in 1:dim(operator_property)[2]){    # cycle through stacks of tables
     S <- S + (norm_op_pr_1[i, k] - norm_op_pr_2[j, k])^2      # definition of the Euclidean metric
   }
   proximity_matrix[j, i] <- sqrt(S)
 }
}
View(proximity_matrix) # conducting cluster analysis
t <- vector()# the position of the smallest element
l <- vector()# table of future clusters
cluster_matrix <- proximity_matrix
while(dim(cluster_matrix)[1] != 2){ # combining elements into clusters until the dimension of the matrix is 2 x 2
 for(i in 1:dim(cluster_matrix)[1]){
   for(j in 1:dim(cluster_matrix)[1])
     if(cluster_matrix[i, j] == min(as.dist(cluster_matrix))){ # finding the smallest value in the proximity matrix
       t <- c(i, j)
       break
     }
 }
 a <- t[1]  # ensuring the shift of columns (rows) to the left (up)
 t[1] <- min(t)
 t[2] <- a   # formation of a new column name of the proximity matrix (based on the names of the combined elements)
 colnames(cluster_matrix)[t[1]] <- paste(colnames(cluster_matrix)[t[1]], colnames(cluster_matrix)[t[2]], sep=" & ")
 row.names(cluster_matrix)[t[1]] <- colnames(cluster_matrix)[t[1]]  # forming a new row name of the proximity matrix
 l <- rbind(l, c(colnames(cluster_matrix)[t[1]], cluster_matrix[t[1], t[2]]))  # adding generated nodes and corresponding metrics to the table
 # enumeration of the values in the removed columns by the nearest neighbor strategy
 cluster_matrix[, t[1]] <- 0.5*cluster_matrix[, t[1]] + 0.5*cluster_matrix[, t[2]] +
                     0.5*abs(cluster_matrix[, t[1]] - cluster_matrix[, t[2]])
 cluster_matrix[t[1], ] <- t(cluster_matrix[, t[1]])  # filling an empty row (transposing a calculated column)
 cluster_matrix[t[1], t[1]] <- 0  # replacing the minimum value with 0
 cluster_matrix <- cluster_matrix[, -t[2]]  # reducing the dimensionality of the matrix (removing empty rows and columns)
 cluster_matrix <- cluster_matrix[-t[2], ]
}
View(cluster_matrix)
# creation of the "union - node - metric" table, adding the last combination of elements to the cluster table
l <- rbind(l, c(paste(colnames(cluster_matrix)[1], colnames(cluster_matrix)[2], sep = "&"), cluster_matrix[1, 2]))
n <- vector()# creating a column of nodes
for (i in 1:dim(l)[1])
  n[i] <- paste("d", dim(proximity_matrix)[1] + i, sep = "")
l <- cbind(c(1:dim(l)[1]), l[, 1], n, l[, 2])
colnames(l) <- c("Step", "Unification", "Node", "Metric")
cluster_groups <- as.matrix(l)
View(cluster_groups)
cat("Created clusters:\n")
print(as.character(cluster_groups[, 2]))
library(cluster) # construction of a dendrogram
clust <- hclust(as.dist(proximity_matrix), method = "complete")
plot(clust, xlab = "Hashtags")
interpretation <- vector() # interpretation of cluster analysis
cl <- c(6, 4, 3)
cl_level <- c(0.6, 1.0, 1.4)
```

```
for(i in 1:3){
 rect.hclust(clust, cl[i], border = i + 1)
 cluster <- cutree(clust, cl[i])
 interpretation <- cbind(interpretation, colnames(as.dist(proximity_matrix)), cluster)
 colnames(interpretation)[i] <- paste("cluster division on level ", """, cl_level[i], """, sep = "")
}
cat("\nInterpretation of cluster analysis:\n")
print(interpretation)
```

5. Conclusions

Summing up, it can be stated that the hypothesis put forward at the beginning is confirmed. Having analyzed the appearance of graphs of functions, correlations between time-quantity values, we can confirm the opinion that at the beginning of the war (the first week), information about the situation in Ukraine became a kind of explosion in the media space of the world. The number of comments, posts, "tweets" on this topic reached a record high of 4,000,000. After a week, this trend changed dramatically - the number of discussions decreased significantly. This is due in particular to the fact that the crisis stage has passed (there has been a lot of information, it is no longer so "interesting" for the world). Although it is also worth saying that this decrease continued to a specific mark. For 50 days from the start of the invasion, the number of publications decreased from 4,000 thousand to 500 thousand. At the level of 500 thousand, the number is maintained until now (with insignificant deviations during certain military events). We can assume that under constant circumstances, without significant changes, this trend will take on a downward trend. At the same time, if the circumstances change, it is impossible to make objective assumptions. Another interesting conclusion can be that during the cluster analysis, it was found that the world focuses more on Ukraine, as a victim of the war, than on russia, as the aggressor. Also, a large share of foreigners supports the initiative to help Ukraine (in particular, the issue of joining NATO). On the contrary, it should not be forgotten that the assumptions "support" - "do not support" are not objective, since people tagging the NATO hashtag can also write negative posts. A fair statement would be the following: foreigners are concerned about the situation in Ukraine and its request for protection from NATO. In addition, the last conclusion is that, although interest in Ukraine decreases over time, it still does not fall lower than before the war. It can be said that the war drew attention to our country, foreigners heard about us and will not forget our name or location. The conducted analysis demonstrated the dynamics of changes in the activity of discussing the war and confirmed the previously put forward assumptions.

6. References

[1] O. Snopok, We watch, read, listen: how the media consumption of Ukrainians changed in the conditions of a full-scale war. URL: https://www.pravda.com.ua/columns/2022/06/22/7353987/

[2] Media consumption of Ukrainians in conditions of full-scale war. URL: https://www.oporaua.org/report/polit_ad/24068-mediaspozhivannia-ukrayintsiv-v-umovakh-povnomasshtabnoyi-viini-opituvannia-opori

[3] M. Ulyanovska, The Russian information war in Ukraine is failing - Nina Yankovich. Interview. URL: https://ukrainian.voanews.com/a/jankowicz-russian-disinformation-failing/6961072.html

[4] E. Musk: https://twitter.com/elonmusk/status/1526834598935949312?cxt=HHwWgMDTxc3-s7AqAAAA

[5] S. McSweeney. Our current approach to the war in Ukraine. URL: https://help.twitter.com/uk/safety-and-security/our-ongoing-approach-to-the-war-in-ukraine

[6] MediaSapiens, Twitter will warn about disinformation about the war in Ukraine. URL: https://ms.detector.media/sotsmerezhi/post/29536/2022-05-20-twitter-poperedzhatyme-pro-dezinformatsiyu-shchodo-viyny-v-ukraini/

[7] MediaSapiens, Musk announced how many bots he had on Twitter. URL: https://ms.detector.media/it-kompanii/post/29532/2022-05-20-mask-povidomyv-skilky-botiv-narakhuvav-u-twitter/

[8] https://twitter.com/elonmusk/status/1526465624326782976?ref_src=twsrc%5Etfw%7Ctwcamp%5Etweetembed%7Ctwterm%5E1526465624326782976%7Ctwgr%5E%7Ctwcon%5Es1_&ref_url=https%3A%2F%2Fwww.bloomberg.com%2Fnews%2Farticles%2F2022-05-17%2Felon-musk-says-twitter-must-prove-bot-claims-for-deal-to-proceed
[9] MediaSapiens, Musk said he was accused of violating an NDA when he bought Twitter. URL: https://ms.detector.media/trendi/post/29509/2022-05-16-mask-povidomyv-shcho-yogo-zvynuvatyly-v-porushenni-nda-pid-chas-kupivli-twitter/
[10] Y. Rorh. Introducing our crisis misinformation policy. URL: https://blog.twitter.com/en_us/topics/company/2022/introducing-our-crisis-misinformation-policy
[11] Y. Rorh. URL: https://twitter.com/yoyoel/status/1527320114072195072
[12] MediaSapiens, Twitter has blocked the ability to register accounts in Russia. URL: https://ms.detector.media/sotsmerezhi/post/29049/2022-02-26-twitter-zablokuvav-mozhlyvist-reiestruvaty-akaunty-v-rosii-fedorov/
[13] MediaSapiens, Twitter started flagging the accounts of state media in Belarus. URL: https://ms.detector.media/sotsmerezhi/post/29162/2022-03-11-twitter-pochav-markuvaty-akaunty-derzhavnykh-zmi-bilorusi/
[14] MediaSapiens, Twitter deleted over 50,000 false tweets about the war in Ukraine. URL: https://ms.detector.media/sotsmerezhi/post/29198/2022-03-17-twitter-vydalyv-ponad-50-tysyach-brekhlyvykh-tvitiv-pro-viynu-v-ukraini/
[15] V. Vysotska, S. Mazepa, L. Chyrun, O. Brodyak, I. Shakleina, V. Schuchmann, NLP Tool for Extracting Relevant Information from Criminal Reports or Fakes/Propaganda Content, in IEEE 17th International Conference on Computer Sciences and Information Technologies (CSIT), 2022, pp. 93-98, doi: 10.1109/CSIT56902.2022.10000563.
[16] O. Darwish, et al, Identifying Fake News in the Russian-Ukrainian Conflict Using Machine Learning, in Advanced Information Networking and Applications: Proceedings of the 37th International Conference on Advanced Information Networking and Applications (AINA-2023), 2023, Volume 3 (pp. 546-557). Cham: Springer International Publishing.
[17] W. Tao, Y. Peng, Differentiation and unity: A Cross-platform Comparison Analysis of Online Posts' Semantics of the Russian–Ukrainian War Based on Weibo and Twitter, Communication and the Public 2023, 20570473231165563.
[18] M. Mazza, G. Cola, M. Tesconi, Ready-to-(ab) use: From fake account trafficking to coordinated inauthentic behavior on Twitter. Online Social Networks and Media, 31 (2022) 100224.
[19] D. Weinberg, J. Dawson, A. Edwards, How the Russian Influence Operation on Twitter Weaponized Military Narratives, in International Conference on Cyber Warfare and Security 18(1) (2023) 431-439.
[20] S. Velichety, U. Shrivastava, Quantifying the impacts of online fake news on the equity value of social media platforms–Evidence from Twitter, International Journal of Information Management 64 (2022) 102474.
[21] B., Smart, J., Watt, S., Benedetti, L., Mitchell, M. Roughan, # IStandWithPutin versus# IStandWithUkraine: The interaction of bots and humans in discussion of the Russia/Ukraine war, in Social Informatics: 13th International Conference, SocInfo 2022, Glasgow, UK, October 19–21, 2022, Proceedings (pp. 34-53). Cham: Springer International Publishing.