Change point detection in relational event history data based on moving window approach with optimal window length.

Bradley Miles Besselink

Applied Data Science, MSc, Graduate School of Natural Sciences, Utrecht University
July 1, 2022

KEYWORDS

Relational Event Model (REM)
Relational Event History
Moving Window
Change point detection

ABSTRACT

This paper proposes a strategy to study the social dynamics in timestamped data. The analysis is based on the Relational Event History (REH). It introduces Apollo 13 data to illustrate the study objective. The goal of this study is to detect change points in relational event history data based on moving window approach with optimal window length. One of the methods used in the study is the Relation Event Model. The Relational Event model is a method to study social networks over time. This method is later on complemented with the more dynamic method called the moving window approach. The results of the moving window approach, expressed in parameters, are the input variables for the next method called change point detection. Change point detection is introduced to study the change in parameters over the different windows, to analyze the social interaction patterns that can be found in the Apollo 13 data. Data cleaning was performed on the Apollo 13 dataset to make it suitable for the analysis. The analysis is performed in R. Seven effects are proposed to fit the Relational Event Model. Five moving window lengths are considered in order to find the optimal window size. These window sizes are complimented by three different overlapping percentages. Change points are detected using the 'MedoidAI' application. 'MedoidAI' is an R/Shiny app for time-series segmentation and changepoint detection tasks. In the application two different algorithms are considered to detect multiple change points. These algorithms are the Binary Segmentation and the PELT algorithm. Both are considered and compared in their ability to detect change points. The results of the study show an optimal window length at 0.5 hours with 50 percent overlap. This window length showed the lowest BIC score. The optimal window can detect change points that are more specific than the results seen at a larger window size. While the smaller window size showed a results that seems more sensitive to the datapoints over time. This implies that the optimal window could be the optimal size to detect change points in the Apollo 13 dataset.

1. Introduction

In the past years, Relational Event Modeling approaches have been introduced. The first Relational Event Model (REM) is introduced by Butts in 2008. Relational Event Models can be used to study the complexity of interaction patterns that appear between individuals. The model informs about the history of interactions and influences the probability of future social interactions. Relational event models can analyze social interaction based on continuous-time. Continuous-time makes the events dependent on each other within the instance of an event, this creates the relational component. The data used to fit these models is called a Relational Event History (REH) (Butts, 2008). A relational event history consists of information about social interaction between a set of individuals. The dataset

contains information about the actors involved and the time the interaction took place. An instance of relational event data is an interaction between two or more individuals at a certain point in time (Meijerink-Bosman et al., 2022).

The REM model proposes the propensity that a certain event happened at a specific point in time. The chance this event happened, affects the possibility of future events. Each event is an answer to the previous propensity (Butts and Marcum, 2017).

The Apollo 13 dataset consist of timestamped data. The composition of the data makes it possible to analyze using a relational event model (REM). The REM has become a well-established approach to learn the structure of interactions in real-time. The model is used to study how specific drivers (parameters) appear and shape which and when certain interactions occur. Based on the event history the model can use either endogenous or exogenous variables. These variables will be explained in the Method section for the Relational Event Model.

The moving window can be added to REM. The dynamic component of the moving window allows effects to vary over time. Which allows for the detection of changes over time in the parameters. The moving window approach (Mulder and Leenders, 2019) uses a specific length of the data to slide over the whole relational event sequence. Each slice, the relational event model is fitted to the subset of relational events that fall within the window. Theses slides result in an image of the change of social interaction over time, indicated by the predictors. This study applies the moving window approach (Mulder and Leenders, 2019). The implementation of the model saves the estimates of the parameters for each window based on the data. The window sizes can vary. This particular study aim to find the optimal window size.

A change point (Kamalabad and Grzegorczyk, 2018; see also Kamalabad and Grzegorczyk, 2021) is a datapoint that can be detected through a change in their statistical properties in the timespan of the study (Killick and Eckley, 2014). The detection of change points has already found many relevant application areas, such as bioinformatics (Erdman and Emerson, 2008) and climatology (Reeves et all, 2007). Change point detection can be performed for detecting a single change point as well as multiple change points. This will focusses on the detection of multiple change points over the time span of the Apollo 13 voice recordings.

Altogether, this process will lead towards answering the research question: “How can we compute the optimal window length in the moving window approach and use it to detect change points?” A variety of window lengths are considered to find the optimal size. Afterwards change point detection is applied on the optimal window, a smaller sized window and larger window as well. The results of the change point detection at different window lengths are compared to get a better understanding of their performances on the task. In this paper, the first step will be to get a better

understanding of the data that is used for the analysis. Afterwards, the methods that are used are described together with the implementation approach. Then, the results of the analysis will be presented and discussed. This results in a conclusion were the research question will be answered, followed by a discussion on the approach and results of the study.

2. Data

2.1.1 Description of the data

The Apollo 13 dataset is used in this study to support the process of answering the research question. The dataset consists of transcriptions of the voice recordings of the Apollo 13 mission to the moon. The transcriptions come from both the a website and text files from GitHub (Issa Marie Tseng, 2021) that are publicly available. The Apollo 13 dataset consist of time based data over the span of 6 days. There is a change in the interaction dynamics after a problem occurs within the aircraft. This problem occurs on day 3.

2.1.2 Packages

Table 1

Packages used in the study by programming langue.

Package	Use	Programming language
Pandas	Used to read the data files	Python
Requests	Used to make requests to the server for web scraping	Python
Bs4 – BeautifulSoup	Used to gather information from the webpage for web scraping	Python
Re	Used for regex patterns	Python
Functools - reduce	Used to remove duplicates in data cleaning webpages	Python
Csv	Used to store the output files in a csv format	Python
IPython.display - display, HTML	Used to get a better interface to view the results during the data cleaning stage of the webpages	Python
Relevant	Used to fit the Relational Event Model	R
Network	Used to create an edgelist	R
Ggplot2	Used to visualize the data	R
Dplyr	Used to manipulate the data	R
Openxls	Used to open the excel file	R
Viridis	Used to extent the option of colors used in the visualizations	R

Table 1 shows the packages that are used during the data cleaning and data analysis of the study. The data cleaning was performed in Python and the data cleaning was executed in R Studio.

2.1. Story behind the Apollo 13

The story of Apollo 13 will be told get a better understanding of the events that are part of the dataset. On April 11, 1970, the Apollo 13 spacecraft was launched from the Kennedy Space Center. Three astronauts, Jim Lovell, Fred Haise and Jack Swigert were part of the Flight team. The Apollo 13

spacecraft consisted of two crafts, one command model/orbiter (called Odyssey) and a landing module (called Aquarius). They were joined together by a tunnel. After a successful launch the Apollo 13 was on its way to the Fra Mauro highlands of the moon. The goal of the flight was to conduct geological experiments and roam the Imbrium Basin.

During the flight a problem occurred. An explosion ripped one of the oxygen tanks apart. This resulted in crippling the power supply. This is how a routine mission turned into a race for survival. Luckily, the crew of the mission was rescued. This is why the mission is classified as a "successful failure" (Pruitt, 2020).

During the mission there was a lot of interaction between the Flight team, consisting of the astronauts, and the ground control team. The ground control team is mostly referred to as Houston. Nasa provided a full digital audio transcription of the conversations. This data consists of an actor interacting with another actor, the spoken message and a time stamp. This time represents the time that the line is spoken in the recording (starting from zero).

2.1.3 Graphical data exploration

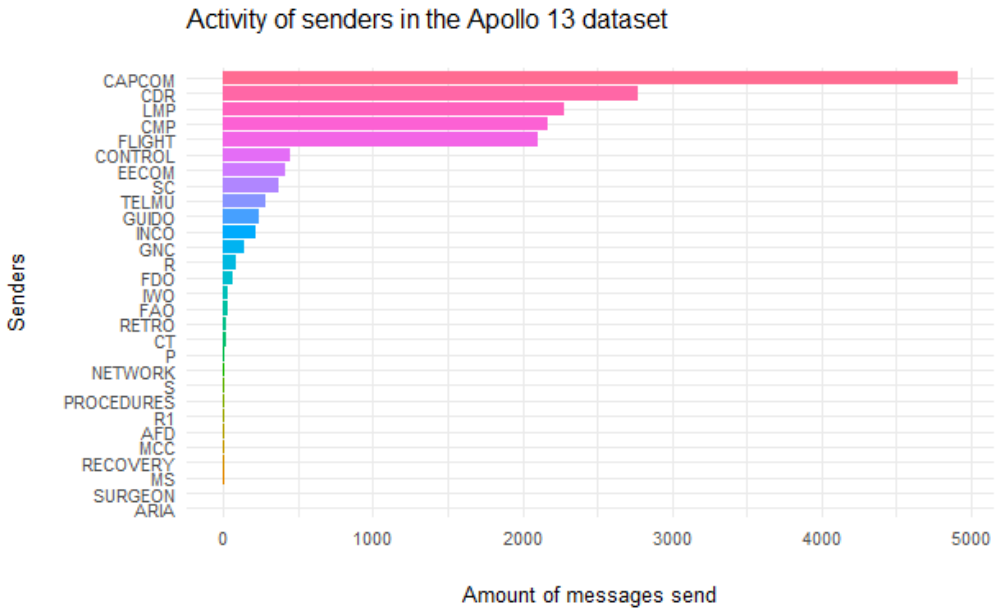

Figure 1: The activities of senders in the dataset, based on the amount of messages that they send.

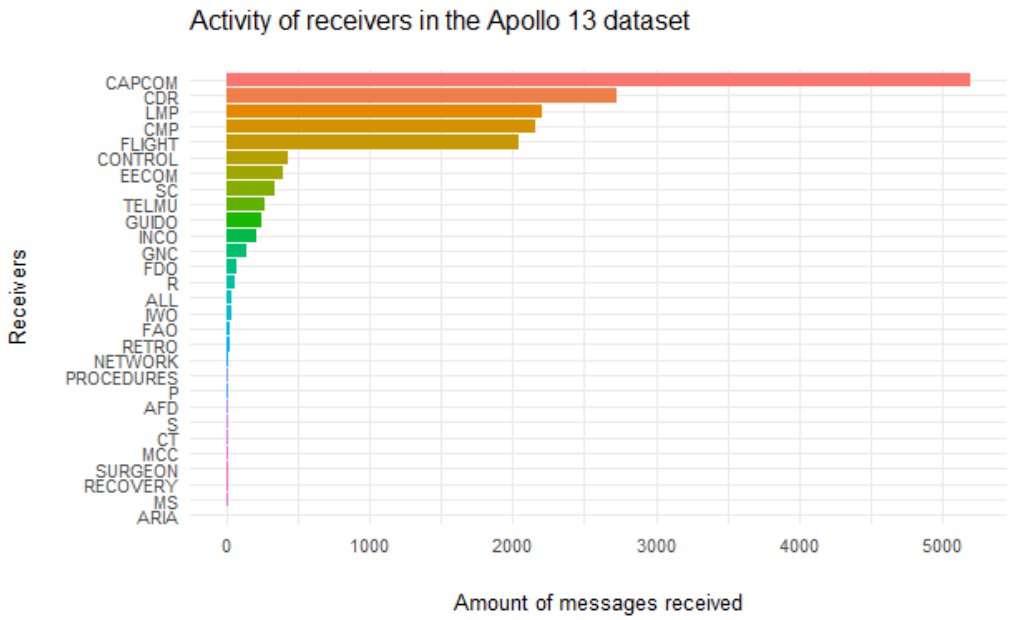

Figure 2: The activities of receivers in the dataset, based on the amount of messages that they have received.

Figures 1 and 2 show the total amount of the messages that are send or received by a certain actor in the dataset. Both graphs are quite similar in their ranking of the different actors in the dataset. The distribution in both graphs show that CAPCOM is the top listed actor at both figure 1 and figure 2. This implies that CAPCOM sends and receives most of the messages. CDR, LMP and CMP are listed after CAPCOM. In this dataset CAPCOM represents Houston, who is part of the ground team. CDR, LMP and CMP are the astronauts that are part of the flight team. Most of the interactions took place between CAPCOM and one of the astronauts. Some interactions took place between the astronauts themselves, as the total sum of all the messages send and received by the astronauts passes the total amount of messages send and received by CAPCOM. There are some actors in the dataset that are not sending or receiving messages. For example the ARIA which represents the Apollo Range Instrumentation Aircraft. The ARIA is an aircraft, that cannot communicate by itself. Another salience is the Surgeon who only receives information.

2.2 Data cleaning

The data cleaning is split up in two parts: cleaning of the text files and cleaning of the web scraped data.

2.2.1 Text files

Part of the GitHub repository called 'Apollo 13 real-time' by Tseng are two transcriptions of the Apollo 13 audio files. These two original files are somewhat unstructured. The audio transcriptions of the flight team and ground team are stored in separated files. The files have the same layout, both

contain information about the event. This information is split up into three parts: the time, sender, and message. This information is split up using a delimiter. The delimiter consist of various amounts of white spaces. What makes the files somewhat unstructured are the events, which are separated by lines. These lines can also occur within a single event.

The data is first opened in Excel to create three columns 'time', 'sender' and 'message'. This way the problem with irregular white spaces is overcome. This data is loaded into Python. A problem appears as there are irregular lines between the various events. These lines occurred since some lines had different sender who spoke at the same timepoint. Other times the irregular line appeared when one sender spoke. This had to do with one message being spread out over multiple lines.

It's not hard to overcome the problem were different sender speak interchangeably. When one row happens to have a missing value in the 'time' column, this indicates that multiple senders are speaking at the same time.

Relational Event data consist of only one event for each timepoint. This raises a problem. To overcome this problem, the rows where multiple senders are talking at the same time are handled manually. For each timepoint that has a missing value, the time that was stated in the previous line is used as a replacement. At this point each sender has an unique time-based value. As one problem is solved, another one presented itself. This time duplicate times appear. These are not allowed to be part of the Relational Event dataset.

Before taking care of this problem, the time structure was converted from an HMS structure to seconds. Working with seconds allows for small time difference to exist in the dataset, which fits the Apollo 13 dataset well. The process is covered by a function that first separates and saves the values for hours, minutes, and seconds. The calculation from hours to seconds is done by multiplying the amount of hours by 3600. The minutes are multiplied by 60. These values, together with the seconds that happened to be in seconds already, are summed for each row in the dataset. These new values that represent the time in seconds replace the original 'time' column.

By now the problem that occurred with the duplicates was handled. Two for loops are introduced to overcome the problem. Simply stated, the first loop over the values of the column 'time'. Each item gets a position in the list. If the current time is already stored in the list, the values receives an additional second, before obtaining their position in the list. In the second for loop, the item in the list replace the original time values. These replacements are executed through the indexes of the items. Some duplicates remain after finishing the procedure. This happens as some items occur more than twice. To solve this problem, both for loops are ran multiple times.

What remains is the problem that some lines have missing values for 'time' and 'sender'. In practice, this refers to the message being spread across several lines. This is also verified in the original

text files (Tseng, 2021). To overcome this problem the message need to be in line with the corresponding time and sender, therefore the 'time' and 'message' are merged.

All these steps were applied separately on both text files. Afterwards the files were merged and verified, this included a duplicate check.

2.2.2 Web scraped data

The website with Apollo data consists of 28 different pages that represent the communication between the air and ground team of Apollo 13 (Woods et all., 2020). The data consists of transcriptions from the original recordings. A list of URLs is made to scrape the data. This list is looped through. In this loop the data is taken from the URL using the 'beautifulsoup4' package in Python. The loop implies that the data scraping is executed separately for each of the URLs, but the procedure is the same. To store the data, two empty lists are created. One list contains the messages, the other one stores the time and names together. The preferred way to scrape data, is to take the name of the html class the data belongs to. In this case, the text of the messages doesn't belong to any class, therefore Regex patterns are created to scrape the data. The Regex patterns scrape the data based on the html structure. The time value was found to lay between and and should contain 6 digits. The messages are present between and . It's important to note that these patterns scrape data that is beyond the study of this analysis. Therefore data cleaning is introduced.

At first, all the lines that start with a number are kept in the set. These are relevant for the time-based approach of the analysis. The two list that were created earlier are stored in a data frame. In order to proceed this step, the lengths of the lists should be equal. This was not always the case. To overcome this problem the following steps are taken:

- Check the difference in length of the list using the index.
- The difference in length should be zero. Always check for inconsistencies even when the list have a zero length difference after scraping.
- Following the method above, some of the dark blue italic lines that appear on the webpage are part of the set.
- Verify if the findings are correct. This can be done by going five points above and below the point a reduction or insertion of data took place.

Afterwards the list are stored in a data frame. The time and name values were in the same column and are therefore separated. The separation took place by introducing a whitespace. This created a problem with the name values. The names consisted of multiple words that were separated with a whitespace. To overcome this problem the names are replaced by equal names were the whitespaces are replaced by the '_' sign. In two cases the column that contained the time and name, did not split correctly, these values were manually inserted.

Now that the 'time' and 'name' column contain the correct values it's important to have a closer look at the values regarding time. As seen in the process of cleaning the text files, the time structure needed to be converted from an HMS structure to seconds.

The actors in the 'name' column are part of the team from the Apollo 13 mission. Their position in the hierarchy is more interesting and explanatory for the interaction dynamics of the mission. The abbreviations of the different positions are listed in table 2 in the appendix. The information in this table is presented in the GitHub repository by Tseng. Names that do not contain an abbreviation, are described based on the information in the 'Apollo 13 Technical air-to-ground voice transcription' by NASA from 1970.

To analyze interaction dynamics, each sender is paired with a corresponding receiver. These receivers are manually added to the set. By default, the receiver of the previous message, will be the sender of the next message. All lines are verified based on this pattern. When the true interaction pattern differs from the default, the receiver is replaced by the correct actor.

An interesting problem that appeared are the long communication breaks. These breaks are gaps in timeline that did not keep any additional information. The exact locations of these gaps were found searching for 'Long Comm' on the webpages. For each of the messages that came after the break, the index is registered. This message is stored as variable. The difference between this message and the message before the break is calculated. This value is stored in another variable. This variable represents the length of the long communication break. Afterwards all values in the time column are considered separately to see if the values exceed the index. Values that did exceed the index value are corrected by removing the time difference. For the previous five lines before the index value, the difference between that certain value, and the value one line above is calculated. All differences are divided by five to calculate the mean difference. The value in this position is first set to the previous timepoint, as if no event happened. Afterwards the value is added up with the mean difference. This resulted in a corrected value of time. Data points in the timeline that come after this timepoint are affected as well. Therefore these values will be corrected in the same manner. This results in a timeline were the long communication breaks are removed from the set.

2.2.3 Merging the data

The text and web datasets are merged together. This introduced duplicates in the rows and columns. To overcome this problem the duplicates were removed. Duplicates in the 'time' column received an additional second to their original value. Rows that appeared as exact duplicates were removed.

Afterwards there happened to be a unresolved problem in the dataset. Messages that mentioned 'Houston' aimed to be received by CAPCOM. In the data cleaning process the assumption was made that messages containing the word 'Houston' were send by CAPCOM. This assumption

was investigated. The result showed examples of 'Houston' being called by the astronauts in order to contact Houston. Therefore changes are made regarding this phenomenon.

749 duplicates were found after merging the data. Most duplicates contained differences in the writing of names and use of punctuation marks. Some messages were found to contain multiple speakers for one message. By default the name of the first actor was selected. Exceptions were made when this individuals were already present in the 'receiver' column.

3. Method

3.1.1 Relational Event Model (REM)

To form a better understanding of the Relational Event Model, this section contains details on the composition of the model.. The main focus of the REM model is the event rate (λ). The event rate consists of the time (t), the sender (s) and the receiver (r). The risk set (Rt) is a set of events that can potentially occur at time (t). In a more formal understanding, the risk set contains all the possible directed or undirected pairs of senders (s) and receivers (r) (Meijerink-Bosman et al., 2022).

$$\Delta t \sim \text{Exponential}\left(\sum_{\mathcal{R}(t)} \lambda(s,r,t)\right). \tag{1}$$

For each instance of a relational event, the event rate is assumed to remain constant over time. The time between the current event and the next event takes an exponential distribution. The rate parameter is the sum of all the event rates at time point (t). Equation (1) shows the change in event rate in a more formal manner.

High levels of event rates at a certain point in time (t), will result in a decrease of time until the next relation event happens. In the REM model, the event rate is modeled as an outcome variable.

The REM takes predictors into account. Butts (2008) refers to the predictors as statistics. The statistics can consist of exogenous and endogenous predictors of the event rate. Exogenous variables are external to the Relational Event History. Endogenous predictors summarize characteristics of past interactions. This can be the volume of previous interactions for each combination of individuals. The model parameters (βp) that are associated with the predictors, support the inferences that were made about the effects that drive the dynamics of social interactions.

3.1.2 Implementation of the Relational Event Model (REM)

An important part of the implementation of the REM are the statistics. When implemented in the model, the statistics represent the behaviors of interest.

The predictions of the relational events are based on the assumption that actors who have

interacted frequently in the past, are likely to continue interacting in the future. The relational event treats the fraction of previous contacts as a predictor of future contact.

The data preparation took place in R. In this phase the data is transformed into a format that is suitable to be interpreted by the REM. At first the data is transformed into an 'edgelist'. The 'edgelist' is a matrix consisting of the number of events * 3 columns. The three columns are 'time', 'sender' and 'receiver'. This implies that the 'message' column is removed from the dataset. By converting the dataset to a matrix, the sender and receivers are replaced by a number from 1 to n. Where n is the total number of individuals that are part of the dataset. The last row of the 'edgelist' is a null event to ensure exact timing. In practice this row will be ignored when fitting the model.

The data is modeled in R using the package called 'relevent' (Butts, 2021). For this study, the Dyadic Relational Event Model is considered. This model consists of discrete actions, represented by messages, between the senders and receivers, which are both individuals. This description complements the format of the Apollo 13 dataset. Dyadic relational event models in the relevant package can be approached through the rem.dyad() function.

The R documentation of the package shows a variation of arguments and parameters (Butts, 2021). The parameters of the model obtain information present in the 'edgelist', the number of actors in the network and the statistics. The statistics consist of a character vector containing the pre-defined names of the effects. Other parameters in the model are the ordinal. They are represented by a Boolean variable that determines the use of the ordinal or the exact timing likelihood. Covar offers the ability to add associated covariate effects to the model. The hessian arguments specify if the Hessian of the log-likelihood or posterior surface should be calculated. These are used to calculate inferential statistics (Butts and Marcum, 2017).

Model adequacy is examined by the deviance residuals. These are included in the fitted model The deviance residuals are analyzed together with the null deviance. Model selection can be done by comparing BIC scores (Butts and Marcum, 2017). A lower BIC indicates a better fit.

3.2.1 Moving window approach

In addition to the Relational Event Model, the moving window approach (Mulder and Leenders, 2019) will be applied to the Apollo 13 dataset. An important consideration is the specification of the window length. The ideal length of the window depends on the temporal nature of the effects. If the duration of the observation period covers multiple months, but not years, a exceptable window length covers one up to two months.

A moving window can be implemented by the following ordered steps:

1) Determine a window length

2a) Fit the REM to the subset of the relational events that are part of the first window.

2b) Save the estimates of the parameter

3) Move the window, so that it overlaps in a specific percentage with the previous window, while also introducing new information to the set.

4a) Fit the REM to a new subset of relational events.

4b) Save the parameter estimates.

5) Repeat step 3 and 4 until all the relational events in the sequence are analyzed (Mulder and Leenders, 2019).

Other than Mulder and Leenders, this paper proposes a strategy to detect the optimal window length. The study considers different lengths and compares their BIC values to propose an optimal window length.

3.2.2 Implementation of the moving window approach

To implement the moving window approach, aiming to find the optimal window size, five different window lengths are considered. The amount of considered window lengths is chosen based on a time restriction for the duration of the study.

Deciding on a strategy is the first step to find the optimal window length. Three different windows lengths will be considered at first. The windows are distinguishable in a small, medium and larger sized window. The small window size consist of one hour. The medium window will be 9.5 hours and the larger window will be 20 hours. The maximum value is chosen based on the largest window size that did not introduce any error messages.

The second step in the process is the evaluation based on the BIC score. The model that obtained the lowest BIC score will be compared with a value between itself and the medium value. If the medium size model has the lowest BIC, the new considered window length will be a value between itself and the window corresponding to the second lowest BIC score.

The third step contains the decision on the values of the overlapping percentages. The overlap percentage is the amount of data that a window has in common with the former window. This study considers overlapping percentages of 25 percent, 50 percent and 75 percent. These values that are evenly distributed on a scale of 0 to 100 percent.

Another consideration is the calculation of the relational implementation mechanism of the values in R. Each window variable consist of a data frame with the columns 'begin' and 'end'. This represents the beginning and end of each window. 'begin' and 'end' do take the variables 'from', 'to' and 'by'.

The following steps can be reviewed to calculate the values for the moving window in R:
1) Take the difference between the value in the two columns of the data frame 'begin' and 'end'.
2) Go to the column name 'end' and insert this value at the variable 'from'.
3) Then calculate effect of the overlap on the movement of the window. In case of 25% overlap, the value will be equal to the value of the total amount of time in seconds - the 25% of the total amount of time in seconds.
4) Place this value in the positions: 'from' at the 'begin' section of the variable 'windows' and in both of the 'by' parameters. This is done because the difference between each value in the rows is equal to the overlap value. The first value after zero, will be the equal to the overlap value.
5) The 'to' parameter in the 'end' section will be filled by the difference between the total amount of time present in the dataset minus the overlap value.
6) The 'to' variable in the 'begin' section will the variable in the 'begin' section - the 'from' variable in the 'end' section (as this variable is equal to the window length, which is the difference between the two columns).

3.3.1 Change point detection

A change point appears when the interaction dynamics in the Apollo 13 dataset change. For example, when the amount of messages increase within a specified time frame. Another change point appears when new actors are introduced to the subset and drastically change the message pattern. To get a more formal understanding of the meaning behind a change point, please have a look at the papers presented by Kamalabad and Grzegorczyk in 2018 and 2021. Both papers can be found in the resources section at the end of this study.

The practical approach to change point detection found it's execution in the application 'MedoidAI – Time Series Segmentation & Changepoint Detection'. The developed of the application was done by Panagiotis Papaemmanouil, Lazaros Paschalidis, Efstathios Chatzikyriakidis, Anestis Fachantidis. The application is designed for time-series segmentation and change point detection tasks. The advantage of the application is the ability to detect multiple change points over the whole

range of window at once. 'MedoidAI' is a shiny application which provides an interface to the functions present in the R package called 'changepoint' (Papaemmanouil et all., 2020).

The changepoint package, as described by Killick and Eckley, provides a series of search algorithms for multiple change point detection in addition to a variety of test statistics. The present algorithms are Binary Segmentation, Pruned Exact Linear Time (PELT) and Segment Neighborhood. The packages provides a range of methods as well. The best fitting method can be selected based on the pattern that is seen in the data. This pattern can be a change in mean, a change in variance or a change in both the mean and variance.

A better understanding of the algorithms allows for a more detailed comparison. The first considered algorithm is called the Binary Segmentation algorithm. This algorithm applies a single change point test statistic to the entire data. If a change point is detected, the data will be split into two. The split appears on the location of the change point. The single change point procedure is repeated on the new data sets, this includes both the set before and after the change. If more change points are identified in either of the sets, then the data is split further. This process is repeated until there are no more change points found in the dataset.

$$\sum_{i=1}^{m+1} \left[\mathcal{C}(y_{(\tau_{i-1}+1):\tau_i})\right] + \beta f(m) \tag{2}$$

τ1 = a change point position

m = used for hypotheses testing, if m =0 there is no change point, when m =1 there is a change point

C = the cost function for a segment

βf (m) = the penalty to overcome to problem of overfitting

The segmentation neighborhood algorithm aims to minimize the cost function as seen in equation (2). It uses a dynamic programming technique to obtain the optimal segmentation for m+1 change points. It reuses the information that was calculated for *m* change points. In this method the computational complexity is considerably higher compared to Binary Segmentation algorithm.

The PELT algorithm (Killick et al, 2012) is quite similar to the segment neighborhood algorithm. They both provide an exact segmentation. The PELT algorithm is designed to be more computationally efficient , due to its use of dynamic programming and pruning (Killick and Eckley, 2014).

3.3.2 Implementation of the change point detection

The 'MedoidAI' application offers different options to carry out the change point detection. To upload the dataset an excel format with a maximum of 1MB is required. One column is selected to be

searched through in order to detect multiple change points. To be able to take all the parameter estimations into account, a new column called 'Variance' is introduced. This column consist of the mean variance for all the different outputs of the statistics. This results in an overview of the aggregated performance of the model in detecting multiple change points. The pattern that appears in the data is examined on the right panel of the application. These insights are used to make decisions on the method of measurement. If there happens to be a big difference in height throughout different segments of the data, this can indicate a change point in variance. If there is a difference in value of variance within a segment, without appearing on different height level on scale, this indicates a change point in mean (Paemmanuouil, 2020).

Another consideration is the algorithm that will be used to detect change points. In this study there will be a comparison between the Binary Segmentation and PELT algorithm. The goal of the comparison is to find the best performing model concerning the detection of multiple change points for the Apollo 13 dataset. The Binary Segmentation algorithm there is the opportunity to decide on the maximum amount of change points. In this case the maximum number of change points is set to 25. The value 25 did find change points on low and high peaks for the optimal window length. These can be seen on the right panel of the application. When the value was set to the maximum amount of change points, it showed an error. The distribution of the variance is set to normal. This is important to take into account for the interpretation of the results. The segment length influences the total amount of change points that will be detected. This value is dependent on the window size. The value of 8 is selected as the size of the segments. This results in the amount of change points to analyze. The exact locations of the change points are seen by hovering over the red dots.

Different window sizes will be considered for comparison. Together with the optimal window size of half an hour with 50 percent overlap, the window that is half the size and twice as large as the optimal window are considered. The new considered window sizes are one hour and one quarter of an hour.

4. Results

4.1 Final results of the Relational Event Model

Table 3

Results of the final Relational Event Model in relation to the effects

Effects	Estimate	Standard Error	Z value	Pr(>|z|)
NTDegSnd	-1.5196E+03	9.5937E-01	-1583.95	<2.2E-16 ***
NTDegRec	-4.5559E+02	7.4418E-01	-612.20	<2.2E-16 ***
FrPSndSnd	-7.3739E+01	6.8757E-02	-1072.45	<2.2E-16 ***
FrRecSnd	-1.6466E+02	2.8603E-01	-575.69	<2.2E-16 ***
OSPSnd	-1.1640E-01	3.5620E-04	-326.78	<2.2E-16 ***
ISPSnd	-5.7451E-01	8.8253E-04	-650.98	<2.2E-16 ***
PSAB-BA	1.1407E+01	5.1874E-01	21.99	<2.2E-16 ***

The results om the Relational Event Model are displayed in table 3. The final model consist of the effects 'NTDegSnd', 'NTDegRec', 'FrPSndSnd', 'FrRecSnd', 'OSPSnd', 'ISPSnd' and 'PSAB-BA'. 'NTDegSnd' and 'NTDegRec' are the normalized total degree of the effects on the future sending and receiving rates. They represent preferential attachment. Preferential attachment allows for predictions on the availability of the actors in the dataset. Actors who have been present in previous communications, are more likely to be present and able to respond in future communications (Butts, 2008). Figure 1 and 2 underline this statement, as the order of the actors in the sender and receiver columns tent to be similar. There is a large difference found in the amount of messages by the top 5 communicator compared to those who are in the lower regions of the list. This makes the statistics interesting to take into account.

'FrPSndSnd' and 'FrPRecSnd' represent persistence from the perspective of both the sender and receiver. Persistence captures the likelihood that past contacts become future contacts (Butts, 2008). This pattern appears in the Apollo 13 dataset. The air team and ground team stay in contact with each other throughout the conversation. Preferential attachment takes the activity history of actors into account. This means that actors who frequently appearances as sender or receiver in the dataset, will also receive a higher ranking position in the prediction of the next sender or receiver. In the Apollo 13 dataset the ground team is mostly represented by CAPCOM, all the astronauts have the ability to talk to CAPCOM. This is in line with the pattern seen in figure 1 and 2. In practice this implies that CAPCOM will receive a higher predictive value.

'OSPSnd' and 'ISPSnd' are both part of the triadic effects. Interaction of actors can motivate them to contact third parties. This concerns the spread of information were one message is received by multiple actors. The interaction between CAPCOM and the astronauts is stored as a one-on-one interaction. In certain cases the information shared between one astronaut and CAPCOM is relevant for other astronauts as well. This leads to the introduction of the outcoming shared partner effect

('OSPSnd'). Astronauts can contact each other as well and discuss information that is relevant to share with CAPCOM. This pattern is represented by the incoming shared partner effect ('ISPSnd').

'PSAB-BA' represents the AB-BA shift. It captures the tendency for B to send to A, given that A has just received from B (Butts and Marcum, 2017). For example, when CAPCOM sends a message towards CDR, CDR will directly respond to CAPCOM. This pattern has been captured in multiple occasions during the data cleaning process.

All the effects in the model are found to be significant. The standard error values are below 0.9, which show an acceptable reliability. The results show a positive value towards the communications pattern that is captured by AB-BA shifts. All other estimates in the model show negative results. The output assumes that past contacts are less likely to interact with previous contacts. The findings suggest that actors who have communicated before are less like to be part of the interactions found at a later time point in the dataset. The preferential attachment assumption towards senders contains a higher standard error, which suggest questionable certainty of the result. The integration of a third party, as described by the shared partner effects, appears to unlikely in this dataset.

Table 4

Results of the final Relational Event Model in relation to the model fit

Residual deviance	Null deviance	AIC	AICC	BIC
16143137 on 14196 degrees of freedom	295652.1 on 14202 degrees of freedom	16134151	16134151	16134204

The value for the null deviance is 295652.1. The null deviance is part of the residual deviance of the model. This value can be found in table 4, together with the results concerning the model fit. The null model takes one parameter for all the data points. This implies the estimation of a single parameter. A small value of the null deviance indicates that the null model explains the data well. This is not seen in the results of table 4. The value of the residual deviance is related to the saturated model. This model estimates the parameters for each individual datapoint. A lower score increases the likelihood of the model explaining the data well. In this case the residual deviance of the model is 16134138. The BIC score of the model is closely related to the AIC, which indicates for a good model fit.

4.2 Results of the moving window approach

Table 5

Results of the moving window approach by window size and overlap percentage.

Window size	Overlap percentage	BIC
One hour	25 percent	-7940.10
	50 percent	-6321.77
	75 percent	-7319.20
9.5 hours	25 percent	-640.37
	50 percent	-532.52
	75 percent	-497.48
20 hours	25 percent	-1278.21
	50 percent	-749.10
	75 percent	-1285.40

Table 5 shows the BIC scores for each combination of hours and overlap percentages. The lowest BIC score for all three different overlap percentages appear at one hour. This outcome allows for consistency in the procedure that follows. This is important in the comparison of the results. Two new window lengths are introduced, these lengths will be considered for all three overlap percentages. The first new window length covers half an hour. This value is of interest to see if a smaller window size will hold a lower BIC score as well. The other new window length lays between one hour and 9.5 hours. This value is calculated by taking the difference between the two values, which resulted in a window length of 4.25 hours.

Table 6

Results of the moving window approach for the new window sizes.

Window size	Overlap percentage	BIC
0.5 hours	25 percent	-12380.10
	50 percent	-13672.61
	75 percent	-13310.81
4.25 hours	25 percent	-283.02
	50 percent	-275.96
	75 percent	- 316.03

Table 6 shows the results for each of the overlap percentages in the 0.5 hour and 4.25 hours. The 0.5 shows a lower BIC score than the earlier considered window lengths. The 4.24 hour window shows higher BIC values in general. Therefore this window size appears less optimal than earlier considered window lengths. The lowest BIC value is seen at the window length of 0.5 hours and 50 percent overlap. This window scores a BIC value of -13672.61. Therefore this model will be considered in the study of change point detection.

4.4 Results of the change point analysis

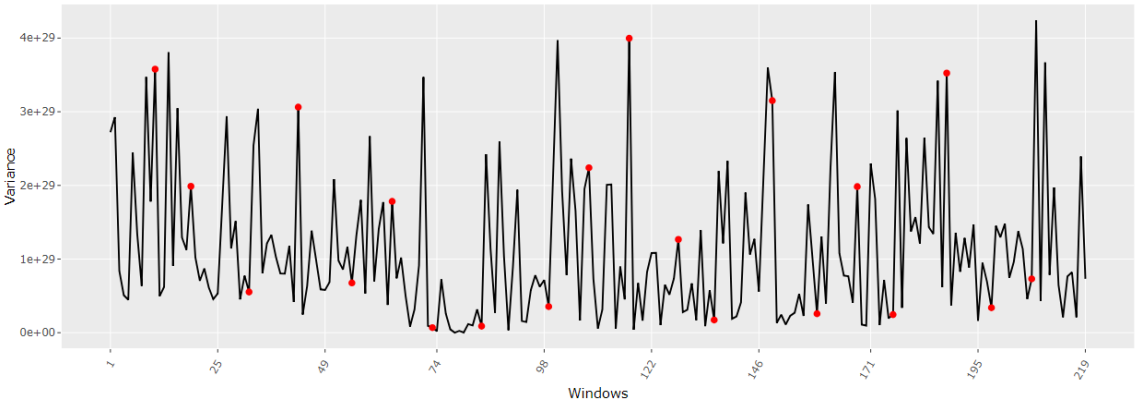

Figure 4: Change point detection result from 'MedoidAI' for the optimal size window using the PELT algorithm

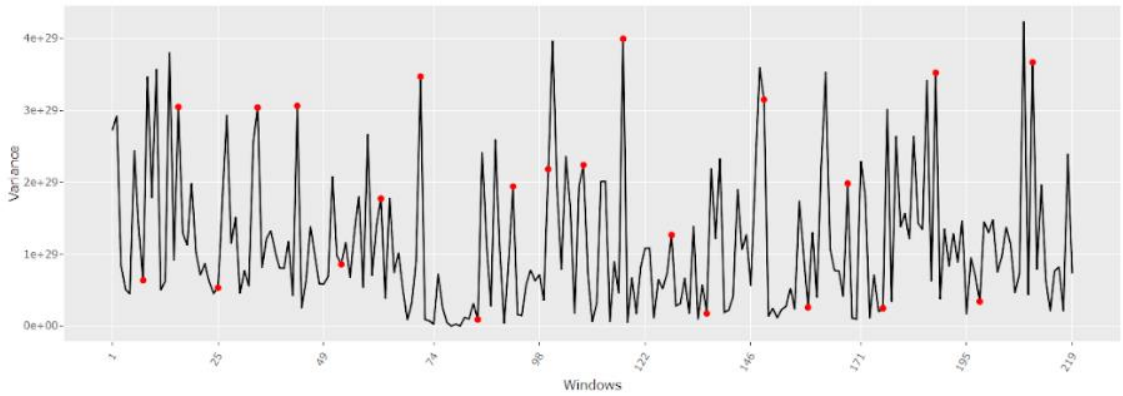

Figure 5: Change point detection result from 'MedoidAI' for the optimal size window using the PELT algorithm

Figure 4 and 5 show the results of the multiple change point detection. The x-axis represents the different windows. The y-axis shows the mean variance of the outcomes from the parameters of the REM model. The optimal window size resulted in 219 windows that we're taking into consideration.

The Binary Segmentation algorithm detected 20 change points. The settings allowed for the algorithm to detect a maximum of 25 change points. Most change points found by the algorithm are assigned to 'dips' in the data. Ten of the change points are detected at lower points in the overall variance. Change points appear were previous and following statistics differ from a certain point in time. Spikes are more likely to show a low value before and after the peak point. This proposes a form of reasoning to clarify why the peaks are less likely to be considered change points.

In figure 5 the PELT algorithm detected a total of 22 changepoints. No maximum value was specified in advance. The results shows an increase in the amount of change points in the peaks of the data. Both graphs show relevant data points. By hovering over the red dots, the graphs have the following change points in common: 43, 84, 108, 117, 128, 136, 149, 159, 168, 176, 188, 198.

In window 43 an event happens, CDR request CAPCOM for an earlier waste water dump and

fuel cell purge. This event requires changes in the schedule which can lead to changes in the communication pattern. This indicates that window 43 can be a relevant change point.

Window 136 captures a total of 420 interactions, while the previous window covers 118 interactions. In window 136 they talk about a checklists. The people that are part of the set are working overtime. This might explain the increase of messages. At the same time there are a few unclear problems, which is a topic that is captured simultaneously.

The point where CDR tells CAPCOM that there appears to be a problem is situated in window 35,36 and 37. Neither of the algorithms detected this situation as a change point. The closest change point is detected by the PELT algorithm, they noticed window 34 contains a change point.

To allow for more thoroughly comparison between the methods used to detect change points, the values for the detected changepoint locations were collected and standardized on a scale from 0 to 100. This has been done for the optimal window, as well as a larger and smaller window.

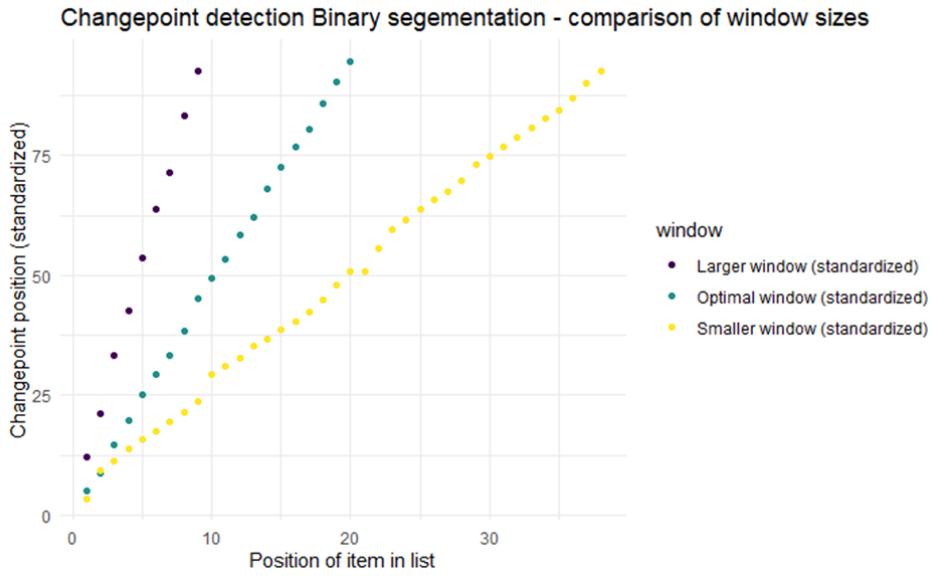

Figure 6: Comparison of standardized changepoint location for the Binary Segmentation algorithm

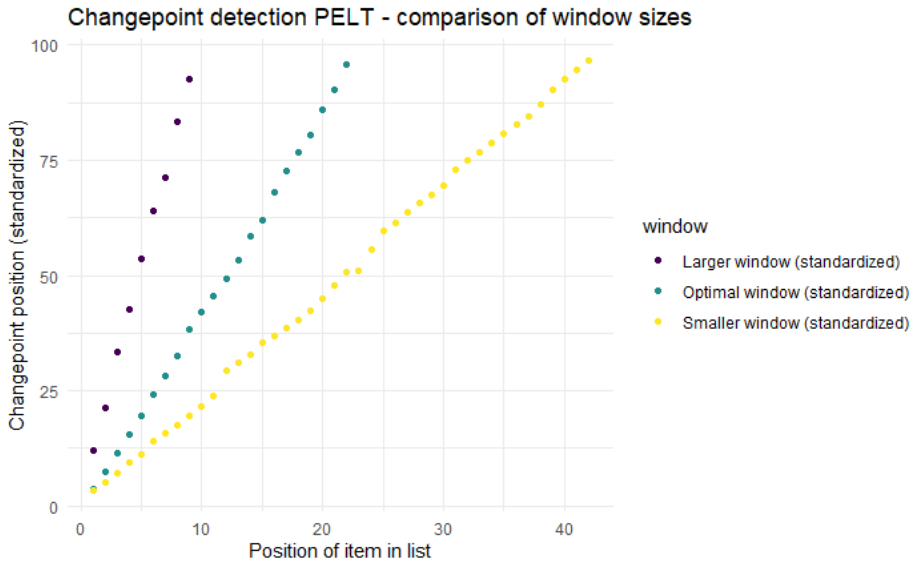

Figure 7: Comparison of standardized changepoint location for the PELT algorithm

Figures 6 and 7 show similar patterns in the positions of the change points over time. This means that when the results are standardized at this specific window length, there is only a small difference in the output values. The visualizations beneath are introduced to get a better understanding of the differences between the performance of the Binary segmentation and PELT algorithms.

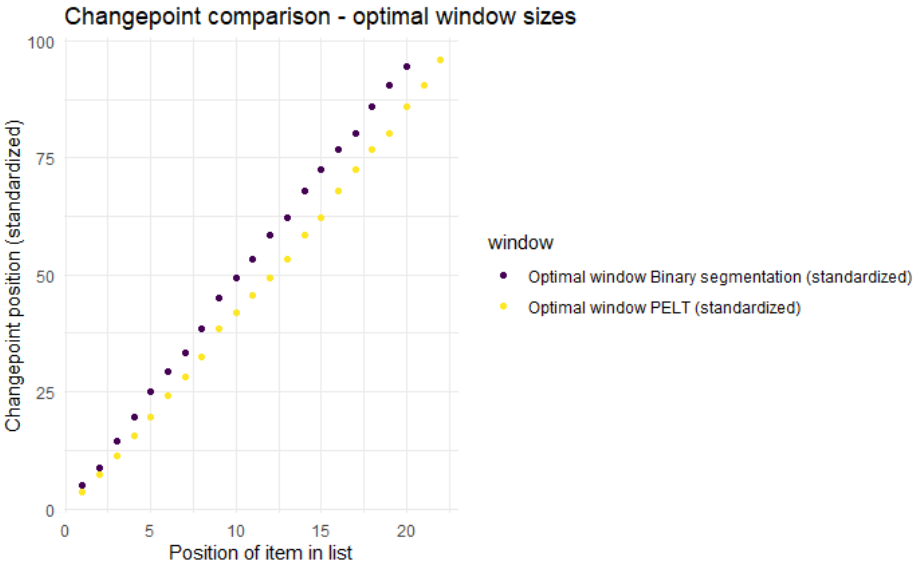

Figure 8: Comparison of standardized changepoint location for the optimal window size

In figure 8 the change point locations for the optimal window are shown together. The line that represents the Binary Segmentation algorithm increases faster in the change points positions over time. The Binary segmentation algorithm has detected less changepoints then the PELT algorithm. At the start of the lines the detected change points closer together. Over time the distance between the points increases gently. The distances between the changepoints shows a linear pattern over time without any gaps.

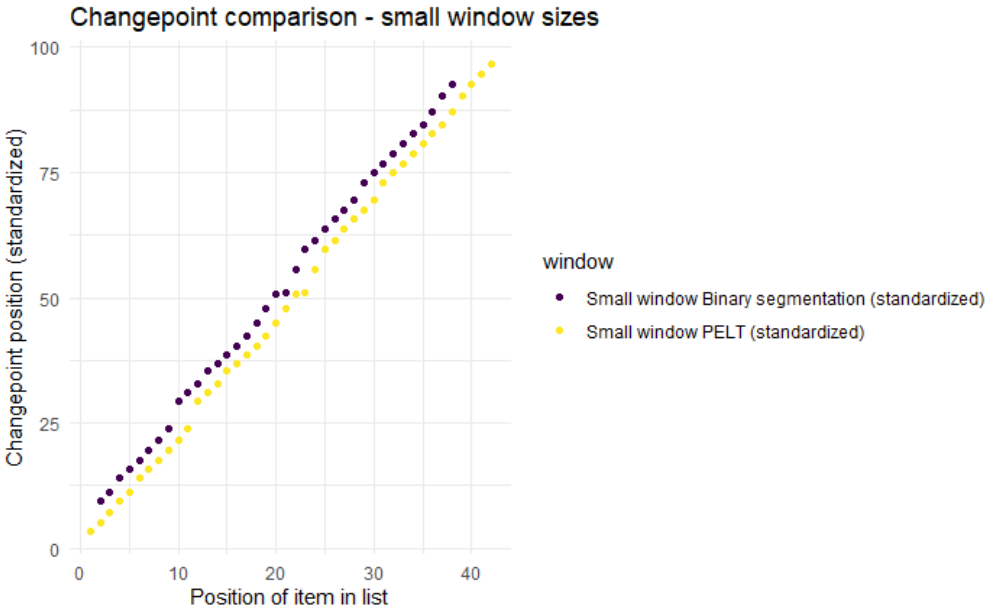

Figure 9: Comparison of standardized changepoint location for the optimal window size

In figure 9 the first change point that is found by the Binary Segmentation algorithm has a higher value then the first changepoint detected by the PELT algorithm. The PELT algorithm detected more change points in the smaller window then the Binary segmentation method. The lines in the graph show less of a straight line and includes gaps. The position of the gaps are similar for both algorithms. The gaps appeared at position 25/100 and 52/100.

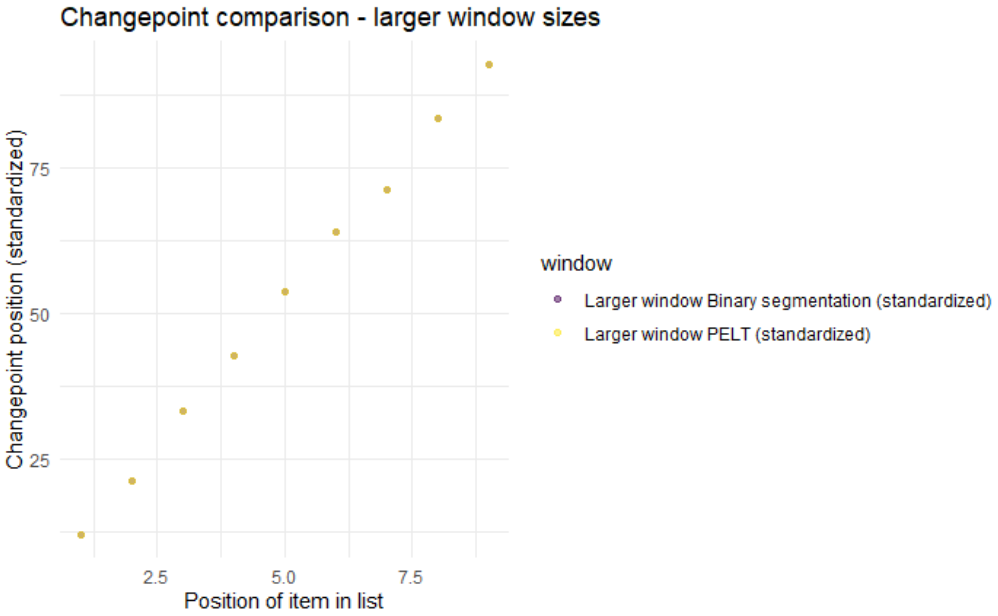

Figure 10: Comparison of standardized changepoint location for the larger window size

The lines in figure 10 appear as one line for both algorithms. The shade of the color lays somewhere between yellow and purple. The windows show an equal amount of points in the exact same locations. The pattern of the change point is linear, were the distances between the points over time appear even.

5. Conclusion

In this paper the REM and moving window approach are introduced. Together with the change point detection they appear as the methods to complement the strategy of answering the research question stated in the introduction of this paper: "How can we compute the optimal window length in the moving window approach and use it to detect change points?". We can compute the optimal window length in the moving window approach by comparing the results of different window lengths and overlapping percentages based on the BIC score. The output of the parameters, used to fit the optimal moving window, are used as an input value to detect change points. The context of the detected change points can be studied by analyzing the specific interactions that appeared in the different windows.

A brief overview of the study starts with the analysis of the data. CAPCOM and the astronauts are the most dominant actors in the Apollo 13 dataset. This has been shown in figure 1 and 2 for both the senders and receivers. The Relational Event Model is applied to the dataset to learn the social network dynamics. Seven statistics were part of the modelling process. Most of the statistics turned out to appear negative. This implies that the interaction patterns in the dataset are unlikely to apply to the Apollo 13 dataset. This suggestion stands for 'NTDegSnd', 'NTDegRec', 'FrPSndSnd', 'FrRecSnd', 'OSPSnd', 'ISPSnd'. The 'PSAB-BA' effect proved to be a positive effect. This statistic captures the tendency for B to send to A, given that A has just received from B. This could mean for example that when CAPCOM messages an CDR, CDR will directly respond to CAPCOM. This pattern is seen a lot during the data cleaning process.

In the application of the moving window approach, five different windows lengths were considered. For each of the lengths, three different overlapping percentages were taken into account. Together, this resulted in an optimal window size of 0.5 hour in length with an overlap of 50 percent. The comparison took place based on the BIC score.

In the change point detection process the R shiny app 'MedoidAI' was used. Three different lengths of windows were considered. A smaller window of 0.25 hours is compared with the optimal window and a larger window of 1 hour. The model with the optimal window length turns out to be is able detect change points that are not too general. These more general point can be found by fitting a larger window. Neither, is the optimal window too specific or overfitting as seen at a smaller window

length. This implies that this window could be the optimal size to detect change points in the Apollo 13 dataset. The relevance of the change points seems justified. Neither of the algorithms detected the lines 'Okay Houston, I believe we had a problem here.' to be a change point. The PELT algorithm detected the window before this event appeared. The change point detection is dependent on the neighboring windows. Therefore we can state that the PELT algorithm considered this event to be related to a change point.

6. Discussion

The REM model showed significant results for all the considered parameters. However the negative values in the estimates turned out to be questionable. The opportunity to have seen the data many times during the cleaning process, lead to questioning the correctness of the results. Part of the dataset are the conversations between two astronauts, while one is communicating with CAPCOM as well in the meantime. This illustrates the incoming shared partner effect. The model output as seen in table 4 assumes that past contacts are less likely to interact with previous contacts. This cannot be correct. As seen in figure 1 and 2, there are four parties that are most likely to be in contact. This should therefore result in a positive value for the persistence statistics. Another consideration is the result of the null deviance. A small value of the null deviance would imply that the null model explains the data pretty well. This was not the case. The accuracy can be improved by a method like parameter tuning.

Another point for improvement is a test for confounding between the parameters. This test is relevant for the parameters 'NTDegSnd' and 'NTDegRec' that are dependent on the amount of previous activities for the senders and receivers. In figure 1 and 2 show that the ranking of the senders and receiver is quite similar. This might lead to a correlation between the stated statistics related to preferential attachment. Therefore a correlation test would be interesting. This is not yet performed due to time limitations of the study.

The estimation of the optimal window length is based on five different window length. The true optimal window might differ from the estimation. In order to find the true optimal window, all combinations of windows and overlapping percentages should be considered. Although this solution is the computational power intensive. Which is ethically questionable due to it's environmental impact.

A problem in the study appeared for the calculation of the medium sized windows. These window lengths were supposed to be the exactly in the middle, between the smallest and largest window. The calculation took the following approach: (20 – 1)/2 & (9.5 – 1)/2. The outcomes should have had an addition of one in order to be correct in the timeframe that starts from zero. A correct calculation would have been ((20-1)/2)+1 .

Another consideration is the validity of comparing BIC scores for different window lengths. At least a certain percentage of the total amount of actors should be present in the subset of the window to allow for a valid comparison. The subset should include a minimum amount of messages that are part of the dataset. A future study can include the specific understanding of the validity test for the BIC scores. An interesting paper to consider for this study would be 'The Power, Accuracy, and Precision of the Relational Event Model' by Schecter and Quintane.

The study aims to bring an overview of the interaction dynamics over the whole dataset. This was successful by selecting a multiple change point detection method. However, the use of the 'MedoidAI' application came with limitations. The parameters cannot be analyzed on an individual level. The specification of the statistics that were responsible for the peaks and dips that are found in the data remain unknown. A future study could focus on the greed of the timepoint, by slicing it into five windows. This allows for a more detailed understanding of the exact timepoint in which the change occurred. Another consideration is the assumption that the data is normally distributed. This assumption was made in the analysis of the change point detection using the 'MedoidAI' application.

7. References

Butts, C. T. (2008). 'A Relational Event Framework for Social Action'. Sociological Methodology, 38(1), 155–200. Retrieved from https://doi.org/10.1111/j.14679531.2008.00203.x

Butts, C.T. and Marcum, C.S. (2017). 'A Relational Event Approach to Modeling Behavioral Dynamics'. In Andrew Pilny and Marshall Scott Poole, eds., Group Processes: Data-Driven Computational Approaches, pages 51-92. Springer International Publishing: Cham, Switzerland.

Buttc, C.T. (2021). 'Package 'relevent''. Retrieved from https://cran.r-project.org/web/packages/relevent/relevent.pdf

Huang, K., Qiao, M., Liu, X., Liu, S., Dai, M. (2019). 'Computer Vision and Metrics Learning for Hypothesis Testing: An Application of Q-Q Plot for Normality Test'. Retrieved from https://arxiv.org/pdf/1901.07851.pdf

Kamalabad, M. S., & Grzegorczyk, M. (2018). 'Improving nonhomogeneous dynamic Bayesian networks with sequentially coupled parameters'. Statistica Neerlandica, 72(3), 281-305. https://doi.org/10.1111/stan.12136

Kamalabad, M.S., Grzegorczyk, M. ‘A new Bayesian piecewise linear regression model For dynamic network reconstruction. BMC Bioinformatics 22, 196 (2021). https://doi.org/10.1186/s12859-021-03998-9

Killick, R., Eckley, I.A. (2014) ‘changepoint: An R Package for Changepoint Analysis’. Journal of Statistical Software. June 2014, Volume 58, Issue 3. Retrieved from https://eprints.lancs.ac.uk/id/eprint/51975/1/v58i03.pdf

Meijerink-Bosman, M., Back, M., Geukes, K., Leenders, R. and Mulder, J. (2022). ‘Discovering trends of social interaction behavior over time: An introduction to relational event modeling’. Behavior Research Methods. Retrieved from https://doi.org/10.3758/s13428-022-01821-8

Mulder, J. and Leenders, R. T. (2019). ‘Modeling the evolution of interaction behavior in social networks: A dynamic relational event approach for real-time analysis’. Chaos, Solitons & Fractals, 119, 73–85. Retrieved from https://doi.org/10.1016/j.chaos.2018.11.027

National Aeronautics and Space Administration. (1970). ‘Apollo 13 Technical air-to-ground voice transcription’. Retrieved from https://www.hq.nasa.gov/alsj/a13/AS13_TEC.PDF

Papaemmanouil, P. Paschalidis, L., Chatzikyriakidis, E. Fachantidis, A. (2020). ‘Medioid AI Time Series Segmentation & Changepoint Detection’. GitHub repository. Retrieved from https://github.com/medoidai/ts-changepoint-detection-app

Pruitt. S. (2020, April 16).’How Apollo 13 Became NASA's 'Successful Failure’. Retrieved from https://www.history.com/news/apollo-13-mission-lessons

Schecter, A., Quintane, E. (2020). ‘The Power, Accuracy, and Precision of the Relational Event Model’. Organization Research Methods 2021, Vol. 24(4) 802-829. Retrieved from https://journals.sagepub.com/doi/10.1177/1094428120963830

Tseng, I.M. (2021). ‘Apollo 13 real-time’. GitHub repository. Retrieved from https://github.com/issa-tseng/apollo13rt

Woods, D., Kemppanen, J., Turhanov, A., Waugh L.J. (2020, April 6). ‘The Apollo 13 Flight Journal’. Retrieved from https://history.nasa.gov/afj/ap13fj/index.html

8. Appendix

Table 2

Actors that are present in the dataset, their position in the hierarchy and abbreviation which they go by in the dataset.

Name	Position in the hierarchy	Abbreviation
Liebergot	The Electrical, Environmental and	EECOM
Kranz	Consumables Manager	FLIGHT
William Fenner	The Flight Director	GUIDO
Alan Glines	Guidance Officer	INCO
Elliot-High Eagle	The Integrated Communications Officer	RETRO
Weitz	The Retro Fire Officer	CONTROL (& LCC)
Mattingly	Launch control center	-
Slayton	Should have been one of the original crew members	-
Champlain Jerauld	Assigned the crew members	-
IWO Jima	Commander ships chaplain for IWO Jima	IWO
-	Ship	ARIA
-	Apollo Range Instrumentation Aircraft	TELMU
-	The Telemetry, Electrical, and EVA Mobility Unit Officer	GNC
-	The Guidance, Navigation, and Controls Systems Engineer	FDO
-	The Flight Dynamics Officer	FAO
-	Commtech	CT
-	Photographic helicopter	P
-	Recovery service	R
-	Relay bus	Relay