Master Thesis

im Rahmen des

Universitätslehrganges „Geographical Information Science & Systems“

(UNIGIS MSc) am Interfakultären Fachbereich für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

spatial point pattern analysis of social media feeds during crisis events

vorgelegt von

Dipl.-Ing. Ingo Rickmeyer

U102855, UNIGIS MSc Jahrgang 2012

Zur Erlangung des Grades

„Master of Science (Geographical Information Science & Systems) – MSc(GIS)”

Hannover, den 05.01.2017

Science Pledge

Ich versichere, die vorliegende Arbeit ohne fremde Hilfe und ohne Verwendung anderer als der angeführten Quellen angefertigt zu haben. Alle Stellen, die wörtlich oder inhaltlich den angegebenen Quellen entnommen wurden, sind als solche kenntlich gemacht.

Die vorliegende Arbeit wurde bisher nicht in gleicher oder ähnlicher Form eingereicht.

____

Datum, Unterschrift

Acknowledgments

I would like to thank the entire University of Salzburg UNIGIS team for their support throughout this master's program.

I would like to thank Dr. Bernd Resch from the University of Salzburg who supported me by giving me valuable hints and instructions to the thesis topic.

Furthermore, I would like to thank Yury Kryvasheyeu et al. for the permission to use their data from the Dryad Digital Repository.

Additionally, special thanks to Ulrich Salden for error reading.

Finally, I would like to acknowledge the patience, emotional support and occasional distractions of my family, my wife Anne and my friends.

Thank you.

Ingo Rickmeyer

January 2017

Abstract

The flow of information during crisis events is a critical and an integral part of information management. Contemporary communication platforms like social networks provide an opportunity to study such flow and derive early-warning sensors. In this thesis, an attempt is made to assess the potential of using harvested social media for modelling the distribution of social media feeds during crisis events. Therefore a mechanism based on latent Dirichlet allocation (LDA) and document clustering is proposed to model flood and hurricane semantic information, while spatial point pattern analysis is applied explore spatial patterns and to assess the spatial dependence between incident-topic tweets and crisis events. A global Monte Carlo K-test is indicated that the incident-topic tweets and flickr massages is significantly clustered at different scales up to 2500 m. A covariate from the density of human settlements and transport infrastructure for a better fit of the models is implemented. The fitted model is diagnosed using residual analysis as well and also QQ-Plots with simulated data. To test the occurrence of complete spatial randomness (CSR), a spatial Kolmogorov-Smirnov test in two dimensions is made. A spatial-temporal approach with a visualisation of a 3D-scatterplot shows the relation of the sentiment (“negative”) and emotion (“fear”) marks of the social media stream within the chronology of the observed event.

The results of the thesis support the basic notion that social media feeds as volunteered geographic information can be used as sensors, enhancing the awareness of crisis events and their impact on humans.

Keywords: crisis events, latent Dirichlet allocation (LDA), social media, spatial point pattern analysis, inhomogeneous Poisson process, residual analysis, sentiment analysis

Contents

List of Figures ..... **VII**

List of Tables ..... **XI**

1 Introduction **12**

1.1 Motivation 12

1.2 Related Work 13

1.3 Research Questions and Outline of the Thesis 15

1.3.1 Research Questions 15

1.3.2 Outline of the thesis 15

2 Methods **17**

2.1 Methods of statistical text analysis 17

2.1.1 Introduction 17

2.1.2 Information Retrieval 17

2.1.3 Text Mining 18

2.1.4 Natural Language Processing 21

2.1.5 Sentiment Mining 24

2.1.6 Topic models using Latent Dirichlet Allocation (LDA) 25

2.2 Methods of Spatial Data Analytics 26

2.2.1 Introduction 26

2.2.1.1 Data visualisation 27

2.2.1.2 Exploratory data analysis 27

2.2.1.3 Models of spatial data 27

2.2.2 Techniques of Spatial Data Analysis 27

2.2.2.1 Spatial point patterns 28

2.2.2.2 Visualisation of spatial point pattern 29

2.2.3 Exploratory analysis of spatial point patterns 29

2.2.3.1 Density based analysis 29

2.2.3.2 Distance based analysis 29

2.2.3.3 Dependence between points 31

2.2.4 Model-Driven Spatial Data Analysis 32

2.2.4.1 Modelling spatial point pattern 32

2.2.4.2 Marks and covariates 33

2.2.4.3 Edge effects 33

2.2.4.4 Spatial autocorrelation 34

2.2.4.5 Hot Spots 35

2.2.5 Process schema for the spatial point pattern analysis 36
2.3 Introducing the software R 37
2.3.1 Model - driven spatial point pattern (SPP) analysis with spatstat 37
2.3.1.1 Numeric errors during the calculation 38

3 Results 40
3.1 Statistical text analysis 40
3.1.1 Text Mining using R 40
3.1.1.1 Choosing the scope of the text 40
3.1.1.2 Preparing the corpus 40
3.1.2 Topic Modelling with LDA using R 43
3.1.3 Sentiment Mining using R 48
3.2 German flood in 2013 52
3.2.1 Data preparation 52
3.2.2 Data inspection 53
3.2.3 Data Visualisation 55
3.2.4 Explanatory data analysis 56
3.2.4.1 Kernel smoothing 58
3.2.4.2 Testing for Complete Spatial Randomness (CSR) 60
3.2.4.3 Kolmogorov – Smirnow test of CSR 61
3.2.4.4 G-function and K-function 61
3.2.5 Model analysis of the Twitter feed in Dresden 66
3.2.6 Model analysis of the Twitter feed in Saxony 72
3.2.7 Model analysis of the Flickr feed in Dresden 77
3.3 Hurricane Sandy in 2012 80
3.3.1 Data preparation 80
3.3.2 Data Visualisation 80
3.3.3 Text Analysis 82
3.3.4 Explanatory Analysis 82
3.3.5 Model analysis of the Twitter feed in Miami 85
3.4 Spatio temporal sentiment analysis 86
3.4.1 Spatio temporal analysis of sentiment and emotion marks 87
3.4.2 Spatio temporal analysis of the relation to measurement data 89

4 Discussion 91
4.1 Statistical text analysis 91
4.2 Spatial point pattern analysis 91
4.3 Spatio-temporal analysis 93

5 Conclusions and outlook ... 95

6 References ... 97

7 Appendix ... 104

7.1 Software used ... 104

7.2 R-Scripts repertory ... 104

7.2.1 R-Script Import Twitter ... 104

7.2.2 R-Script Text minig ... 106

7.2.3 R-Script LDA / LDAviz ... 109

7.2.4 R-Script Gauge Dresden ... 110

7.2.5 R-Script Statistical analysis ... 111

7.2.6 R-Script Sentiment analysis ... 119

7.2.7 R-Script 3D-Scatterplot with sentiment ... 120

7.2.8 R-Script Maps ... 121

7.2.9 R-Script import Flickr ... 122

7.2.10 R-Script Flickr analysis ... 123

List of Figures

Figure 2-1: Categorization of IR-models (translated from German entry, original source Dominik Kuropka ("Wikipedia - Information retrieval," 2016) 18

Figure 2-2: The big picture of text mining (NaCTeM, n.d.) 18

Figure 2-3: A Venn diagram of the intersection of text mining and six related fields (shown as ovals) (Miner, 2012).. 20

Figure 2-4: High-Level View of data mining (Wachsmuth, 2015) 22

Figure 2-5: A move tree of the type that results alpha-beta pruning (Samuel, 1969) 22

Figure 2-6: Illustration of supervised learning for (a) classification and (b) regression (Wachsmuth, 2015).. 23

Figure 2-7: Illustration of unsupervised learning for (a) flat clustering and (b) hierarchical clustering (Wachsmuth, 2015) 23

Figure 2-8: Sentiment classification techniques (Medhat et al., 2014) 24

Figure 2-9: The intuitions behind Latent Dirichlet Allocation (Blei, 2012)........................... 25

Figure 2-10: left) Topic Model using plate notation (Mark Steyvers and Tom Griffiths, 2004) p. 5 ; right) Example of a density distribution under LDA for 3 words and 4 topics (Blei et al., 2003) p. 3 26

Figure 2-11: Dot maps with an independent, regular or clustered distribution (Baddeley et al., 2015)... 28

Figure 2-12: left: quadrat count, right: surface from KDE including the original point pattern (O'Sullivan and Unwin, 2010)... 29

Figure 2-13: left: Distances to the nearest neighbour, middle: Determining the K function for a pattern, right: K function for clustered and evenly spaced events (O'Sullivan and Unwin, 2010)... 30

Figure 2-14: Morishita Plot of a point pattern distribution with clumps (Morisita, 1959).... 32

Figure 2-15: an example of the edge effect... 34

Figure 2-16: Varying levels of spatial autocorrelation with Moran's I (Goodchild, 1986)..... 35

Figure 2-17: Schema of the spatial point analysis for social media feeds modified from (Yang et al., 2007) 36

Figure 3-1: dataframe tw_miami, first 10 tweets... 40

Figure 3-2: upper-left: word-frequency, upper-right: number of letters ~ number of words, lower-left: proportion of letters, lower-right: distribution of letter-position 42

Figure 3-3: examples of a word cloud and term frequencies 43

Figure 3-4: LDA-model with 10 topics with 8 terms for each topic 46

Figure 3-5: top words from selected topics from the twitter corpus (Wang et al., 2015) 46

Figure 3-6: topic probabilities by document.. 46

Figure 3-7: Visualisation of LDAviz – right: global view of the model, left: term – topic relation 47

Figure 3-8: total sentiments for the dataset elbe2013_de_hochwasser 49

Figure 3-9: total sentiments for the dataset Miami 49
Figure 3-10: sentiment distribution during the Hurricane Sandy landfall (29.10.2012) 50
Figure 3-11: Emotion „fear“ during the German flood (day of year 151 corresponds to the 01.06.2013) 50
Figure 3-12: Emotion „fear“ during the Hurricane Sandy event (day of year 300 corresponds to the 26.10.2012)... 51
Figure 3-13: Temporal distribution of tweet messages (dataframe elbe2013).................... 53
Figure 3-14: Distribution of longitude and latitude of tweet messages (dataframe elbe2013) 53
Figure 3-15: Temporal distribution of tweet messages (dataframe elbe2013_de)............... 54
Figure 3-16: Temporal distribution of flood related tweet messages (dataframe elbe2013_de_hochwasser) 54
Figure 3-17: The stream flow of the gauge “Pegel Dresden” during the German Flood (http://undine.bafg.de).. 55
Figure 3-18: maps with dataframes elbe2013, elbe2013_de 55
Figure 3-19: maps with dataframes elbe2013_de_hochwasser (flood related content) (left) and the drainage basin of the river Elbe 56
Figure 3-20: spatio – temporal distribution of the flood related tweets in Germany........... 56
Figure 3-21: Heatmap of 2d bin counts (left) and Contours of a 2d density estimate (right) 57
Figure 3-22: the dataset of Dresden and its implementation in spatstat............................ 57
Figure 3-23: spatio – temporal distribution of the flood related tweets in Dresden 58
Figure 3-24: 4 sigmas for the standard variation of the Gaussian smoothing kernel............ 59
Figure 3-25: Quadrat counting and contour density plot 59
Figure 3-26: 3D- contour density plot... 60
Figure 3-27: Quadrat test for myspp_tw_DD 60
Figure 3-28: Results oft he Kolmogorov – Smirnow test of CSR (mySPP_tw_DD)................ 61
Figure 3-29:G-function (dataset myspp_tw_DD).. 62
Figure 3-30: calculated and simulated K-function 63
Figure 3-31: simulation tw_DD_fit: fitted trend and estimated se 63
Figure 3-32: predicted intensity and calculated and simulated K-function for tw_DD.fit..... 64
Figure 3-33: simulated points and predicted intensity (model tw_DD.fitcov2) 64
Figure 3-34: observed K-function and simulated envelopes for (model tw_DD_fitcov2)..... 65
Figure 3-35: residual diagnostics plot Lurking variable plot from model fit......................... 66
Figure 3-36: Q-Q-Plot with Pearson residuals.. 67
Figure 3-37: rastermap of the land use for human settlements and transport infrastructure 68
Figure 3-38: model fitcov0 with fitted trend (Left) and prediction (Right)........................... 69
Figure 3-39: residual diagnostics plot Lurking variable plot from model fitcov0 69

Figure 3-40: fitted, simulated and predicted intensity of model fitcov2 with K_{obs} 70
Figure 3-41: residual diagnostics plot Lurking variable plot from model fitcov2 71
Figure 3-42: residual diagnostics plot Lurking variable plot from model fitcov2 72
Figure 3-43: Heatmap of 2d bin counts (Right) and distribution map (Left) 72
Figure 3-44: ppp – object of the twitter point pattern in Saxony (Left), boundary of Saxony (Right) 72
Figure 3-45: covariate land use 73
Figure 3-46: The estimated G-function and the computed K-function with simulated CSR . 74
Figure 3-47: the residual plot and the lurking variables 75
Figure 3-48: the fitted distribution (Left) and the predicted intensity (Right) 75
Figure 3-49: calculated and simulated K-function (model tw_SN_fitcov2) 76
Figure 3-50: Results oft he Kolmogorov – Smirnow test of CSR (mySPP_tw_DD)................ 76
Figure 3-51: the distribution of the flickr dataset in Germany 77
Figure 3-52: the lat/lon distribution of the flickr dataset of Germany 77
Figure 3-53: the lat/lon distribution of the flickr dataset in Dresden.................................. 78
Figure 3-54: The temporal distribution of the flickr dataset in Dresden 78
Figure 3-55: the predicted intensity and residual diagnostics of the model fitcov3_flickr ... 79
Figure 3-56: calculated and simulated K-function (model fitcov3_flickr) and corresponding QQ-Plot.. 79
Figure 3-57: the temporal distribution of the Miami twitter dataset 81
Figure 3-58: the lat/lon distribution of the Miami twitter dataset 81
Figure 3-59: the word cloud and the computed LDA Topics 82
Figure 3-60: Heatmap of the twitter feed in Miami 82
Figure 3-61: Quadrat test for dataset tw_miami_related_ppp 83
Figure 3-62: Results oft he Kolmogorov – Smirnow test of CSR (mySPP_tw_DD)................ 83
Figure 3-63: the inhomogeneous K-function with corresponding envelope plot 84
Figure 3-64: 3 sigmas for the standard variation of the Gaussian smoothing kernel........... 84
Figure 3-65: the ppp-object and the computed K-function with simulated CSR envelopes . 85
Figure 3-66: the residual analysis 85
Figure 3-67: simulated points and predicted intensity (model miami.fit) (Left) and calculated and simulated K-function (Right) 86
Figure 3-68: QQ-Plot of the model Miami.fit2.. 86
Figure 3-69: scatterplot from Saxony with the sentiment “negative” 87
Figure 3-70: scatterplot from Saxony with the emotion “fear”... 88
Figure 3-71: scatterplot from Miami with the sentiment gathered from Topsy 88
Figure 3-72: scatterplot from Miami with the emotion “fear” 89
Figure 3-73: scatterplot from Miami with the sentiment “negative”................................... 89

Figure 3-74: alarm level of gauge Dresden (Left) alarm level depending on time assigned to the date 90

Figure 3-75: scatterplot of the tweets in Saxony with the covariate “alarm level” 90

List of Tables

Table 2-1: The seven areas of textmining (Miner, 2012) 19

Table 2-2: The three steps of text mining (Ananiadou and McNaught, 2006) 20

Table 2-3: Popular Techniques and Methods in Spatial Data Analysis (Fischer, 2000) 28

1 Introduction

1.1 Motivation

The flow of information during crisis events is a critical and an integral part of information management. Contemporary communication platforms like social networks provide an opportunity to study such flow and derive early-warning sensors.

When the author first read the book “Digital Humanitarians” by Meier (2015) he was overwhelmed with the technological and methodological developments in the face of humanitarian action.

Before that, the author didn’t have in mind that this flurry of new tools involving cell phones and internet-based platforms could be applied in such a useful way with application in data aggregation, analysis, and visualization, exploiting the potential of collective and artificial intelligence involving thousands of people in reporting events, locations of assets, and places of danger to this extent.

Following Howe (2009) business, economics and governance are transforming as traditional state-based institutions are supplemented and indeed eclipsed by non-state networks of a civil society based on **volunteered geographic information (VGI)**. New technologies enable regular citizens to connect, collaborate, and save lives (Goodchild, 2007).

“By motivating individuals to act voluntarily, it is far cheaper than any alternative, and its products are almost invariably available to all” (Goodchild, 2007).

Meier (2015) shows how effectively help could be coordinated during the 2010 Haiti Earthquake. The US response was a large effort for three major agencies working together with the Government of Haiti, the United Nations, and many countries offering help.

This couldn’t have been done without the aid of massive support from knowledge management systems. For the first time, US government agencies employed social media technologies such as wikis and collaborative workspaces as the main knowledge sharing mechanisms (Yates and Paquette, 2011). Yates and Paquette (2011) study the use of these social media technologies, e.g. the effectiveness of the use, and develop further strategies using social media as knowledge management systems, particularly for disaster and emergency management.

Sun et al. (2016) investigated new microwave measurements for sensing surface water bodies under clear-or-cloudy conditions and also a new method of deriving flood maps from these passive microwave observations. During the evaluation of the flood mapping method with corresponding ground observations storm surge flooding technique, they found out that 95% of the corresponding Flickr reports were distributed within the flood area. So, volunteered geographic information also delivers valuable information for remote sensing operations.

In summary, it can be stated that volunteered geographic information that uses social media as a platform has an enormous potential to support a variety of processes in earth and event monitoring especially during crises defence.

1.2 Related Work

Although social media and volunteered geographic information are phenomena of recent years a lot of research has been done, showing the impacts of (geo)social media and Volunteered Geographic Information during crisis events.

Keeping in mind the substantial advantages in the topics mentioned above, there are obviously problems, which are fully discussed by Li and Goodchild (2010). They describe three approaches to quality assurance, termed the crowd-sourcing, social and geographic approaches. They show advantages and limitations of each and also discuss the research that will be needed to operationalize the geographic approach.

In line with the previous topic, Leetaru et al. (2013) measured that on a representative day only 2.02 percent of all tweets included geographic metadata, with 1.8 percent having a place indicator (manually updated by a user), 1.6 percent having the exact location of information (calculated by the mobile device's geolocation features to provide the user's geographic location at the time each tweet is sent).

Spinsanti and Ostermann (2013) are concerned with the issues of the enrichment of social media content with additional geographic context information and the use of spatio-temporal clustering to support scoring and validation and by that reducing the huge volume of social media to credible and relevant content.

Bakillah et al. (2015) indicate that pure text mining the social media feed is not sufficient to detect relevant communities sharing information during crisis events, because of the sheer numbers, the heterogeneity and the noise of the stream. Better results can be obtained when the explicit relations between users are taken into account.

Following Kryvasheyeu et al. (2015a) social networks provide an opportunity to study the information flow and derive early-warning sensors that can optimize emergency preparedness and response. They investigated the 50 million Twitter messages posted before, during, and after Hurricane Sandy and derived early-warning sensors based on topological and behavioural properties of the "friendship paradox". They also show that the gathered geo-location of users within or outside of the event-affected area plays a significant role in determining the scale of such an advantage. Also, the emotional response seems to be universal and independent from the network topology and seems to be an important factor in order to determine patterns of disasters, giving the opportunity to implement a simple "sentiment sensing" technique that can detect and locate disasters (Kryvasheyeu et al., 2015a).

A similar approach was done by Sakaki et al. (2010) who investigated the real-time nature of Twitter, in particular concerning event detection. They analyse the semantic of the tweet messages and classify them into a positive and a negative class. Considering each Twitter user as a sensor, they use location estimation methods (Kalman filtering, particle filtering) to estimate the locations of events and develope an earthquake reporting system based on sensory observations.

De Longueville et al. (2010) discuss an approach to open established trusted sources driven applications of crisis management in the context of Digital Earth. They argue that up-to-date situational awareness data is always needed and can be complemented with information from VGI. Therefore, they develope workflows to create, validate and distribute VGI datasets for various thematic domains. The topics of exploitation in real time and its integration into existing concepts of Digital Earth, such as spatial data infrastructures, still needs to be further addressed. A forest fire scenario is discussed explaining the meaningfulness of Sensor Web Enablement for VGI, where VGI sensing becomes a sense of the Digital Earth's Nervous System (De Longueville et al., 2010).

In this context Resch (2013) defines a concept of "People as Sensors" as a measurement model in addition to measures of hardware sensors where people contribute their subjective awareness and personal observation.

Klonner et al. (2016) identifie similar topics in addition to De Longueville for further research on compiling a systematic literature review inter alia to identify current research and directions for future research in terms of Volunteered Geographic Information (VGI) within natural hazard analysis. They also detect approaches regarding community engagement and data fusion and important research gaps. They agree in the demand of developing methods to establish user integration into various contexts, such as natural hazard analysis.

Fuchs et al. (2013a) postulate that the needs for the usage of Twitter during crisis events – enough tweets with geocoded positive event relation and a corresponding place/time association to the spatio-temporal events – has an intense dependence on user behaviour (especially in Germany with its dominance of personal privacy and data protection themes).

The understanding of spatio-temporal phenomena is investigated by Sagl et al. (2012) putting collective human behaviour in the context to "weather" phenomena researching the dynamics of urban systems.

Westerholt et al. (2016) investigate the specific problem of how outliers are influencing information in the context of spatial analyses of social media data, which appear, when different users contribute heterogeneous information about different phenomena simultaneously from similar locations causing risks of misinterpretation in a spatial analysis.

Steiger et al. (2015) detect a strong positive correlation of semantically and spatiotemporally clustered tweets in comparison to workplace population census data, determining this as an indicator to analyse workplace-based activities. In their research, they also discuss the probable advantages of using the Latent Dirichlet allocation (LDA)-concept for text mining in contrast to keyword based text-mining.

According to their results, gathering data is obviously a problem. Focussing on the importance of the time factor during managing crisis events, it is beyond doubt that the aim of further research can only be a fully automated process of gathering and analysing data in real time. Manually defining key-words for further analysis may be the choice of time but not the last word on the subject. Further research has to be done.

1.3 Research Questions and Outline of the Thesis

1.3.1 Research Questions

Considering the related work as a whole, one might get the impression that a lot of work has been done with a more aerial view of the events. So, one research question will be to investigate a more detailed view of the spatio-temporal process. Also, the investigations in the sentiment analysis often focus on a "positive/negative" - analysis without going into detail.

Therefore, the first research question (**RQ1**) investigates the distribution of the social media feed on a small scale. The point pattern will be modelled as a spatial point pattern process, investigating if a Poisson process is suitable and determining if there is a way back from the map to the process that generated it.

Covariates help to make models more precise. In addition to the first research, a covariate for a possible better fitting for the model out of RQ1 will be searched as the second research question (**RQ2**).

The detection of event-related social media messages is still a challenge, because of the huge number of irrelevant messages in the stream. Most current studies gather social media feeds by identifying keywords without sufficiency of the selected search words. Therefore, a modern approach with a clustering and topic modelling using the latent Dirichlet allocation (LDA) shall be tested for its usability as the third research question (**RQ3**).

Also the investigations in the sentiment analysis often focus on a pure "positive/negative" analysis. The fourth research question (**RQ4**) will investigate a more detailed view, based on emotions, the emotion "fear" during crisis events will be proved as a measurable and reasonable variable in a spatio-temporal context.

The focus of the fifth research question (**RQ5)** is the possible influence of the in-situ measurement data to the spatio temporal point process. A suitable way to integrate in-situ measurement as a covariate for the spatial point pattern analysis will be looked for.

1.3.2 Outline of the thesis

The thesis is divided into five main parts (introduction, methods, results, discussion and conclusions and outlook):

Chapter 1 starts with the introduction and a subsumption of the related research in the thematic area of the thesis.

Chapter 2 introduces the methods and theory that will be used for text mining and statistical text processing, fielded in information retrieval, natural language processing and data mining. It also describes the gathering of data from the social media feed, the analysis of text for relevant information and underlying patterns of information and gives a short focus on the sentimental content of retrieved information.

Chapter 2 also introduces as second the theory and the methods of the spatial point pattern analysis, fielded in data visualisation, exploratory data analysis with a focus of first and second order effects and models of spatial data, in particular homogeneous and heterogeneous Poisson models.

Chapter 3 is the main application part and starts with statistical text analysis, followed by spatial analysis, including an introduction of the selected use-cases (German flood in 2013 and Hurricane Sandy in 2012) and the implementation of their analysis in R.

Chapter 4 discusses the results in relation to the research questions, giving insights and conclusions of the calculated result and discusses them in relation to the related work.

Chapter 5 concludes the thesis with a summary of the obtained insights in a global context and gives some ideas for the future work.

2 Methods

2.1 Methods of statistical text analysis

2.1.1 Introduction

This chapter gives a short introduction to the methods of statistical text processing: information retrieval, natural language processing and text mining. Sometimes it is not so easy to differ between those techniques, because of their close relationship in practical usage.

Any quantitative research project mining the web for information, e.g. the feeds of social media like twitter of flickr, needs statistical analyses and these need structured information (Munzert, 2014). Thus, social media content is a collection of more or less unclassified text, there is a huge demand of automated analysis of the human language – so called natural language processing.

This thesis focusses on analysing text information provided by twitter and flickr. Of these kinds of instant messaging systems, it is typical that the pure text information is short, limited e.g. up to 140 characters by twitter. It contains of mainly semi- unstructured data mined from APIs (e.g. twitter and flickr) and from databases (in this work: the Sandy database Kryvasheyeu et al. (2015b) and the German flood Dataset from the Harvard University.

2.1.2 Information Retrieval

Big data appears to be on everybody's mind in current discourse. Never before was digital information reachable for everyone who has a computer and some knowledge to use it. To google something has become the synonym for searching for – nothing more to say. Nevertheless, this information overload could also be seen as a problem to design the best search requests for mining the proposed data, getting results in in a ranked order, bringing the results in line with further search requests and making it accessible in a meaningful structure.

Information retrieval is dealing with the problems of optimizing the search request considering the pitfalls of synonyms and homographs (e.g. a girl or a hurricane named Sandy), the ranking of the search results and again focussing big data the handling of masses of data within an acceptable time limit.

Information Retrieval deals mainly with adhoc retrieval, filtering documents and browsing (Baeza-Yates and Ribeiro, 2000). In computer science grepping is the method of choice for an adhoc retrieval of information retrieving the segments with occurrence of the search term (Manning et al., 2008). Thinking in the terms of a database leads to the use of indexes as an internal register for the occurrance of the individual mined terms and their derivatives, e.g. inverted indices (Weiss et al., 2005) with all advantages considering the use of computers (e.g. the use of matrixes and lists).

Figure 2-1 shows the categorization of the common Information retrieval models, which are set theoretical models (Standard and Extended Boolean model and the Fuzzy Set), algebraic models (e.g. the vector space model and the Latent semantic indexing (Baeza-Yates and Ribeiro, 2000)) and the probabilistic models (Blei, 2012), e.g. the binary independence model, probabilistic relevance model (Robertson and Jones, 1976) and especially the Probabilistic topic models like the latent Dirichlet allocation (Blei, 2012).

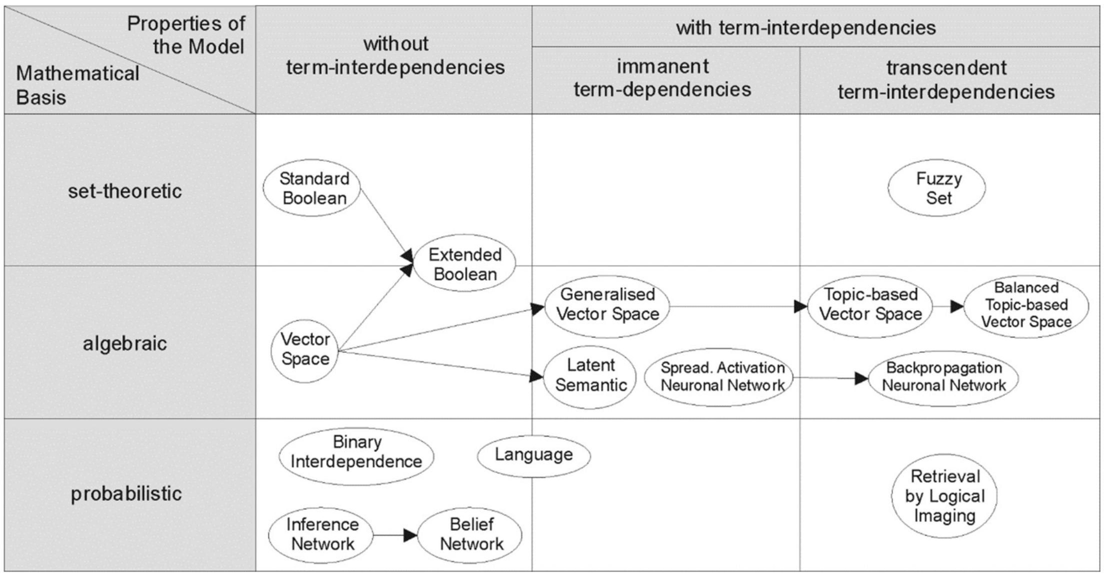


Figure 2-1: Categorization of IR-models (translated from German entry, original source Dominik Kuropka ("Wikipedia - Information retrieval," 2016)

2.1.3 Text Mining

The first step in gathering text based documents from the web is process text mining, shortly explained by Wikipedia as "Text mining (…) refers to the process of deriving high-quality information from text." ("Wikipedia - Text mining," 2016) and this of course by the help of computers automatically extracting information from different written resources ("Marti Hearst: What Is Text Mining?," n.d.). In a nutshell, text mining involves the extraction of high-quality information, the discovery and extraction of knowledge and the revelation of patterns and relationships from unstructured data aiming to further improvements in the relevant processes of scope and also for insights in new business opportunities ("Applying Machine Learning to Text Mining with Amazon S3 and RapidMiner | AWS Big Data Blog," n.d.).

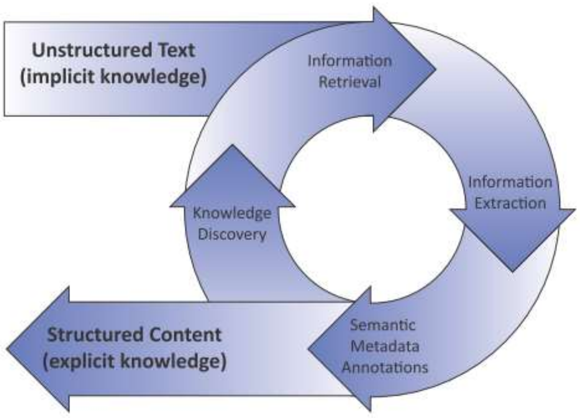


Figure 2-2: The big picture of text mining (NaCTeM, n.d.)

Gary Miner (Miner, 2012) divides the field of text mining into seven different areas:

#	Topic	Chief contents
1	*Search and information retrieval*	Storage and retrieval of text documents and including search engines and keyword search.
2	*Document clustering*	Grouping and categorizing terms, snippets, paragraphs, or documents, using data mining and clustering methods.
3	*Document classification*	Grouping and categorizing snippets, paragraphs, or documents, using data mining classification methods, based on models trained and on labelled examples.
4	*Web mining*	Data and text mining on the Internet, with a specific focus on the scale and interconnectedness of the web.
5	*Information extraction*	Identification and extraction of relevant facts and relationships from unstructured text; the process of making structured data from unstructured and semi structured text.
6	*Natural language processing*	Low-level language processing and understanding tasks (e.g., tagging part of speech); often used synonymously with computational linguistics.
7	*Concept extraction*	Grouping of words and phrases into semantically similar groups.

Table 2-1: The seven areas of text mining (Miner, 2012)

The following diagram (Figure 2-3) shows the interrelating connections between these areas:

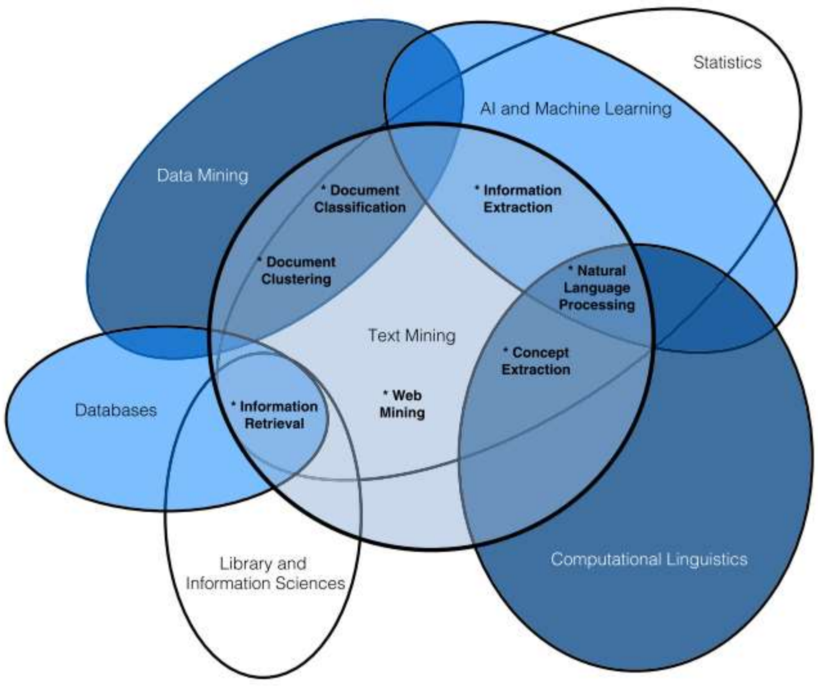


Figure 2-3: A Venn diagram of the intersection of text mining and six related fields (shown as ovals) (Miner, 2012)

Ananiadou and McNaught's propose three steps for structuring the process of text mining (Ananiadou and McNaught, 2006):

#	Topic	Chief contents
1	*Information retrieval*	Gather input texts that are potentially relevant for the given task.
2	*Natural language processing*	Analyses the input texts in order identify and structure relevant information.
3	*Data mining*	Discover patterns in the structured information that has been inferred from the texts.

Table 2-2: The three steps of text mining (Ananiadou and McNaught, 2006)

Information retrieval usually searches for and obtains or - more technical - queries those text-documents from the web or other sources that satisfy the specific information need.

Filtering for keywords (e.g. via APIs like the Twitter API) will result in some text, ranking in relation to the relevance of the search words or just to potentially relevant content (Manning et al., 2008). The problem that goes along with this is that you need to know what you are looking for and this might not be useful for every use case.

2.1.4 Natural Language Processing

Natural language processing outlines algorithms and engineering issues for the understanding and generation of speech and human-readable text (Tsujii, 2011). According to Manning et.al. (Manning and Schütze, 1999), text analysis means that special algorithms give insights in lexical information, syntactical information or the structure of the collected words (Manning and Schütze, 1999). Jurafsky et.al. also analyse the discourse and pragmatic level of a text (Jurafsky and Martin, 2009).

A more technical point of view could describe natural language processing as a kind of annotation of a text or a span of text (Ferrucci, D. and Lally, A., n.d.).

Following Wachsmuth, there are three types of **lexical and syntactical analyses**: The segmentation of a text into single units, the tagging of units, and the parsing of the syntactic structure (Wachsmuth, 2015). Beginning with the smallest unit, like a word or an alphanumerical unit followed by tokenization of text (Manning and Schütze, 1999), sentence splitting and paragraph splitting.

Focussing the terms of tagging the part-of-speech is important, also the categories of tokens (e.g. nouns and verbs) and the more specific part-of-speech-tags which are used in practice (Jurafsky and Martin, 2009). Finally, for identifying different types of phrases – shallow parsing has to be mentioned (Jurafsky and Martin, 2009). Inferring the tree structure of sentences, dependency parsing is important especially for information extraction (Bohnet, 2010).

Information Extracting is the process of analysing unstructured text in relation to real world entities (Jurafsky and Martin, 2009) and also the references between these entities (Sarawagi, 2007). Chiticariu et al. present the important role of information extracting in today's database research (Chiticariu et al., 2010) because of its origin in computational linguistics (Sarawagi 2008).

Text Classification describes the task of assigning each part of a text collection to a specific class in detail (Jurafsky and Martin, 2009).

Data Mining in general discovers potential patterns of new information from mass data already presented in well-structured form. This usually works with the help of machine learning and needs training, based on statistical processes (Figure 2-4). Generalizing these patterns allows analysing new information of current unseen data. Witten et al. postulate that machine learning is the technical basis of data mining (Witten et al., 2011), e.g. Topic Modelling with LDA in the following chapter.

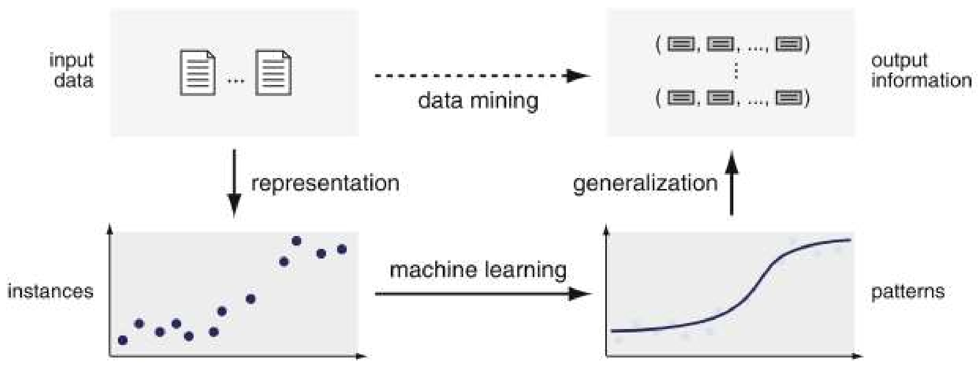


Figure 2-4: High-Level View of data mining (Wachsmuth, 2015)

Machine Learning is a subfield of computer science is closely related to computational statistics and also to prediction-making through the use of computers. The clue is that machine learning "gives computers the ability to learn without being explicitly programmed" (Samuel, 1969).

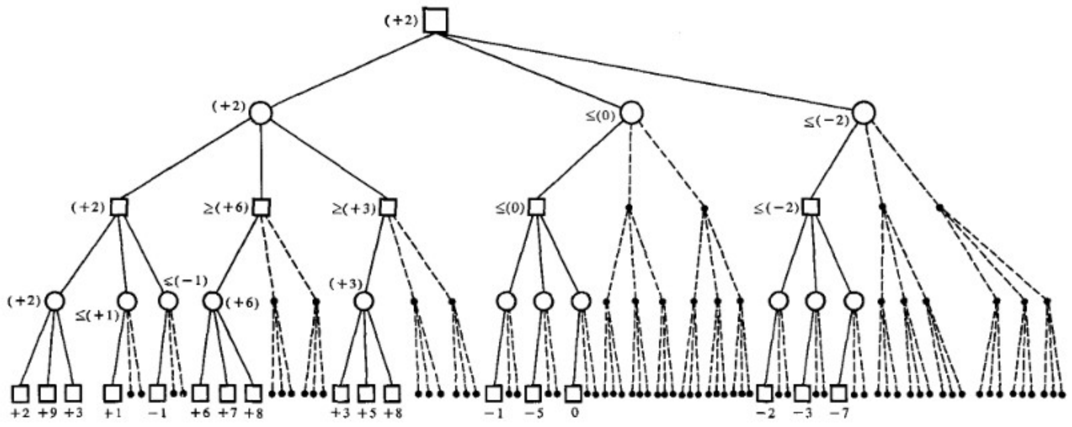


Figure 2-5: A move tree of the type that results alpha-beta pruning (Samuel, 1969)

Decades in advance, the first steps of machine learning were realisations of decision trees in computer games (Figure 2-5). Nowadays keeping in mind the functionality of Google's most important products, it is easy to imagine that machine learning deals with developing of algorithms that can learn from and make predictions on data. The more data is processed the more does measured prediction quality increase (Mitchell, 1997). Machine Learning is also involved in pattern recognition and computational learning theory in artificial intelligence. Thus, modern security gateways including spam filtering, detection of network intruders or malicious insiders couldn't be realized without. Even every day's office life is filled with machine learning, using optical character recognition (OCR) and search engines.

Machine learning could be conflated with data mining in the sense of exploratory data analysis like Clustering, k-means, anomaly detection and Neural Networks and it is known as unsupervised learning.

As far as text mining is concerned, a machine learning algorithm produces a model Y: x → C. Y defines the mapping from represented data x to a target variable C. In text analysis, the target variable may represent classes of texts (e.g. topics or genres) and types of annotations (e.g. part-of-speech tags or entity types) (Wachsmuth, 2015).

Besides the input data, the quality of Y depends on how the data is represented and how the patterns found are generalized.

Supervised Learning uses known training data to fit a model with machine learning algorithms. (Witten et al., 2011). After fitting the model, output information from unknown data can be predicted. The notion of being supervised refers to the fact that the learning process is guided by examples of correct predictions and supervised learning is used for statistical classification and statistical regression shown in Figure 2-6 (Wachsmuth, 2015).

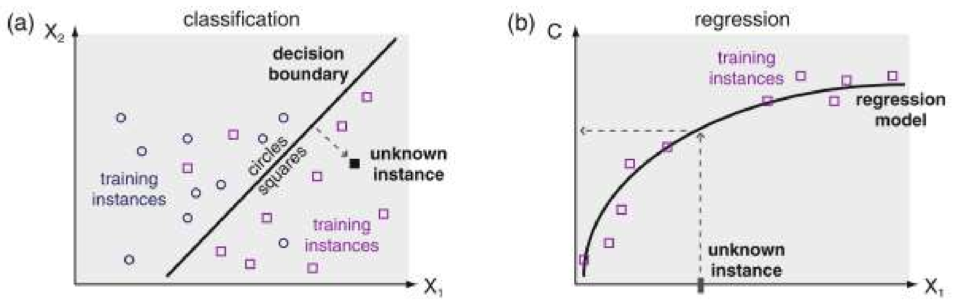


Figure 2-6: Illustration of supervised learning for (a) classification and (b) regression (Wachsmuth, 2015)

Unsupervised Learning obtains its data without output information. Unsupervised learning does not serve in prediction of a target variable, but it gives good returns for identifying underlying rules of the input data, like its organization and association (Hastie et al., 2009).

As mentioned above, clustering is a very common technique in unsupervised learning. Hence, the meaning of a class is usually unknown in clustering, learning patterns based on similarity measurements the resulting model can assign a random number of instances to one of the given clusters. With the focus on text mining, clustering is e.g. used to detect texts with similar properties.

Figure 2-7 shows the two basic types of clustering: flat clustering partitioning without internal associations, and hierarchical clustering with inside instances (Manning et al., 2008).

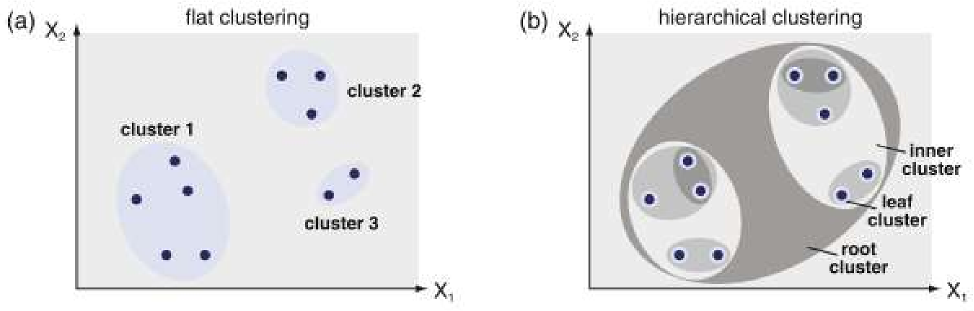


Figure 2-7: Illustration of unsupervised learning for (a) flat clustering and (b) hierarchical clustering (Wachsmuth, 2015)

2.1.5 Sentiment Mining

Sentiment Mining has become one of the most investigated text classification tasks in the last decade. Sentiment analysis is a specialized division of text mining, which uses different classification techniques shown in Figure 2-8. It is an automated process to extract opinion bearing phrases in a piece of text or to classify a piece of text into positive or negative classes (Pang and Lee, 2004). Thereby statistical, linguistic, machine learning and natural language processing tools and techniques are deployed (Paramesha and Ravishankar, 2016).

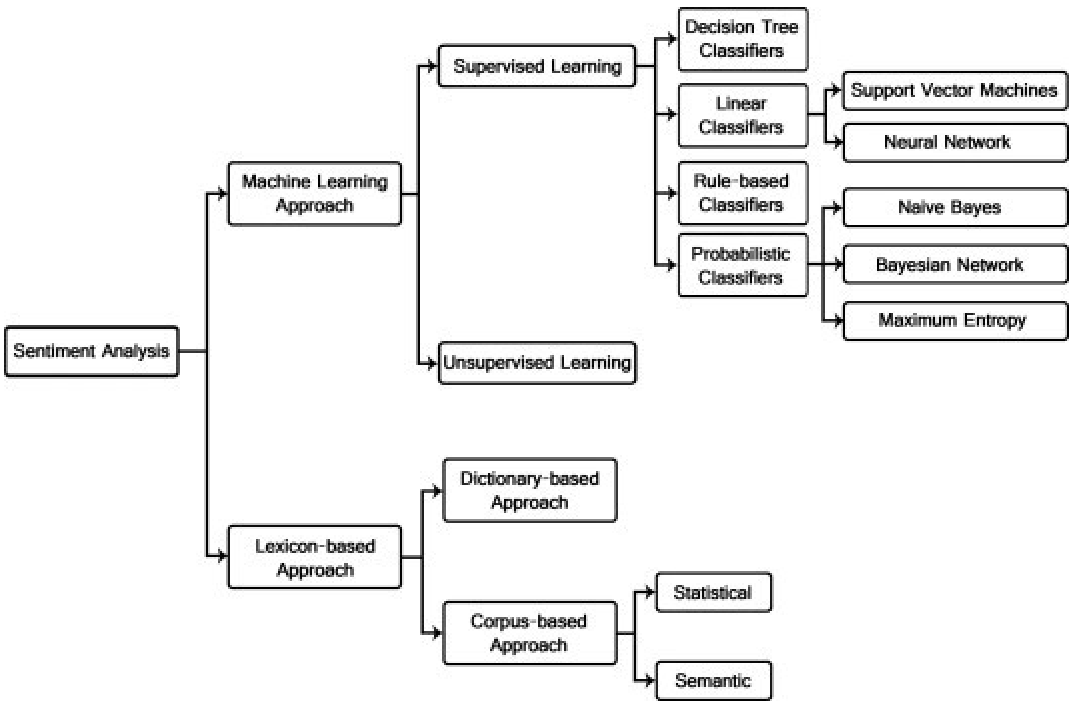


Figure 2-8: Sentiment classification techniques (Medhat et al., 2014)

Facing the fact that capturing social media messages, such as tweets, with their wide diversity of "linguistic code" isn't easy, Bravo-Marquez et al. present a method that combines information from automatically annotated tweets and existing hand-made opinion lexicons to expand an opinion lexicon in a supervised fashion, using machine learning techniques (Bravo-Marquez et al., 2016).

The Stanford CoreNLP system, is an annotation-based NLP processing pipeline system, which uses the NRC Emotion Lexicon with English words and their associations with eight basic emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) and two sentiments (negative and positive) (Mohammad and Turney, 2010).

To conclude, the Sentiment Analysis mentioned above is still a field with enormous demand of research. Keeping in mind ulterior proceedings in automation, it is important to determine techniques automatically considering the context of the text and the user preferences.

Using NLP tools to reinforce the SA process has attracted researchers recently and still needs enhancements (Bravo-Marquez et al., 2016).

2.1.6 Topic models using Latent Dirichlet Allocation (LDA)

In the context of natural language processing the Latent Dirichlet Allocation (LDA) is a generative probabilistic statistical model for collections of discrete data. In the case of twitter data, a collection of tweets could be seen as document consisting of single words and each word could be matched to one of the document's topics as shown in Figure 2-9.

LDA is an example of a topic model and was first presented as a graphical model for topic discovery by Blei et al. (Blei et al., 2003). In practice, we only observe the documents and the other structure remain hidden variables. The goal is to infer the hidden variables with the help of topic modelling.

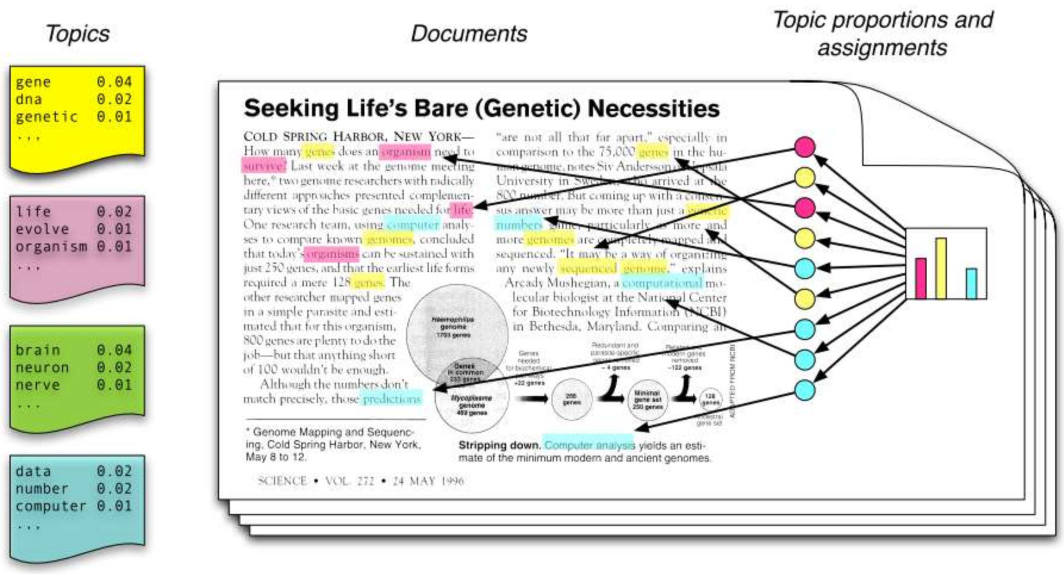
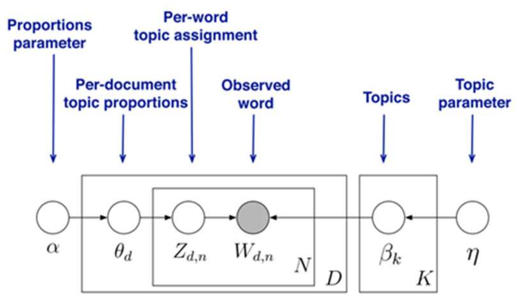


Figure 2-9: The intuitions behind Latent Dirichlet Allocation (Blei, 2012)

LDA has become a topic of major importance in the field of natural language processing (Wang et al., 2007) and can be seen as one of the most popular techniques in text modelling and machine learning. There are many different enhancements of the original model with the focus of e.g. the estimation methods (Mark Steyvers and Tom Griffiths, 2004) and also many extensions to the standard LDA model, e.g. dynamic top models (Blei and Lafferty, 2007) or correlated top models (Blei and Lafferty, 2007).

Most of the popular topic models (such as Latent Dirichlet Allocation) have an underlying assumption: bag of words. (Blei, 2012).

a)



b)

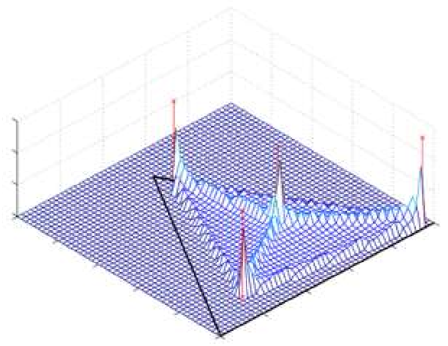


Figure 2-10: left) Topic Model using plate notation (Mark Steyvers and Tom Griffiths, 2004) p. 5 ; right) Example of a density distribution under LDA for 3 words and 4 topics (Blei et al., 2003) p. 3

Figure 2-10 right) shows an example of a density distribution under LDA for 3 words and 4 topics. The vertices of the triangle correspond with the distribution one to one of the words and the middle of the triangle represents the overall distribution over all three words. The four X-marked peaks represent the multinomial distribution for the 4 topics and the 3-dimensional simplex represents the density calculated by the Latent Dirichlet Allocation.

Figure 2-10 left) illustrates the graphical model for Latent Dirichlet Allocation. The nodes where we are looking for – the topic proportions, assignments and topics – are unshaded. The given or observed node – the words of the documents (or the bag of words) - is shaded.

According to the plate notation: the rectangles denote the inside replication of the variables: The N plate denotes the collection words within documents, the D plate denotes the collection of documents within the collection, with the per-word topic assignment is zd, n, the per-document topic proportions θ_d and the per-corpus topic distributions β_k.

To fulfil the posterior expectations, it needs to perform some tasks at hand, e.g., information retrieval, document similarity, exploration, etc..

In this thesis, the “bag of words” is defined by different collections of tweets, gathered during different events at various times (Hurricane Sandy 2012, German flood 2013) and in the sense mentioned above every single tweet represents one single document.

2.2 Methods of Spatial Data Analytics

2.2.1 Introduction

The methods used in spatial data analysis can be categorized in visualizing data, exploratory data analysis and methods for the design of statistical models (Bailey and Gatrell, 1995).

Regarding the analysis of spatial data applied here, a combination of those methods will be used, starting with a visualization of the data, followed by an exploration of potential structures of pattern and finally the development of a model.

2.2.1.1 Data visualisation

Inspecting new data with the use of maps or plots is the first step towards a general picture of this data and a starting point in order to develop an idea of the possible information hidden behind it. This will provide hints to generate hypotheses or a constructional idea for fitting a model of the data.

2.2.1.2 Exploratory data analysis

Developing a credible hypothesis based on specific assumptions concerning on the data is the goal of exploratory data analysis. Intense use of graphical and illustrative presentation of the data with the use of maps and scatter plots will help to understand the data. Especially the focus of the minimums and maximums of the data should not be omitted.

2.2.1.3 Models of spatial data

Once a specific hypothesis about the data is achieved, a formal test of the assumptions is needed. Therefore, a statistical model is used for testing and predicting the hypothesis.

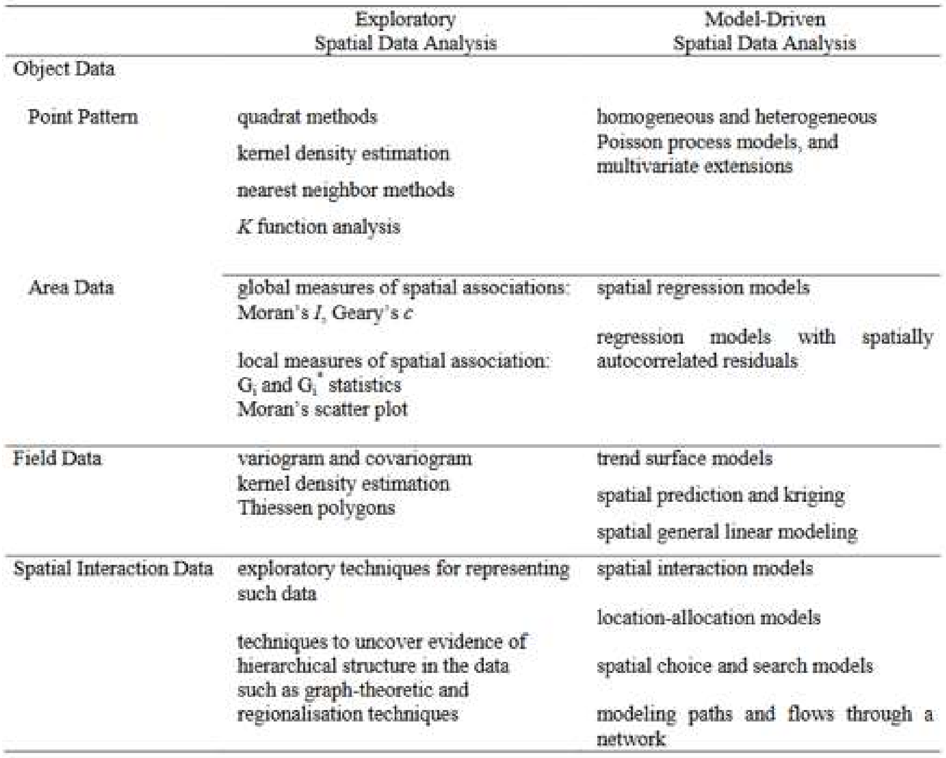
Considering spatial dependence is important in order to get a reasonable representation of the observed effects. The effects can be determined by a large-scale trend or a local effect.

In spatial statistics this is called **first order effects**, which describe the describing overall variation caused by a global variation of a mean value of a parameter, and **second order effects,** caused by spatial dependence describing the tendency of neighbouring values to follow each other depending on their deviation from the mean.

First order effects could be modelled with regression models, second order effects require a consideration of the covariance structure of the inspected data causing theses local effects (O’Sullivan and Unwin, 2010).

2.2.2 Techniques of Spatial Data Analysis

Having in mind the three main categories of in spatial data analysis mentioned above, the methods of analysis can be divided correspondingly: patterns, spatially continuous and area data as displayed in Table 2-3: Popular Techniques and Methods in Spatial Data Analysis (Fischer, 2000).



	Exploratory Spatial Data Analysis	Model-Driven Spatial Data Analysis
Object Data		
Point Pattern	quadrat methods kernel density estimation nearest neighbor methods *K* function analysis	homogeneous and heterogeneous Poisson process models, and multivariate extensions
Area Data	global measures of spatial associations: Moran's *I*, Geary's *c* local measures of spatial association: G_i and G_i^* statistics Moran's scatter plot	spatial regression models regression models with spatially autocorrelated residuals
Field Data	variogram and covariogram kernel density estimation Thiessen polygons	trend surface models spatial prediction and kriging spatial general linear modeling
Spatial Interaction Data	exploratory techniques for representing such data techniques to uncover evidence of hierarchical structure in the data such as graph-theoretic and regionalisation techniques	spatial interaction models location-allocation models spatial choice and search models modeling paths and flows through a network

Table 2-3: Popular Techniques and Methods in Spatial Data Analysis (Fischer, 2000)

2.2.2.1 Spatial point patterns

The data used in this thesis is based on Twitter and Flickr data representing spatial point patterns with properties of coordinates concerning where the tweet has been tweeted or the photo has been shot.

Many other attributes are possibly included, like time of creation or user information, depending on the underlying structure of the data. The basic function of a spatial point pattern analysis is to examine whether it is distributed at random or represents a clustered or independent pattern (Figure 2-11) (O'Sullivan and Unwin, 2010).

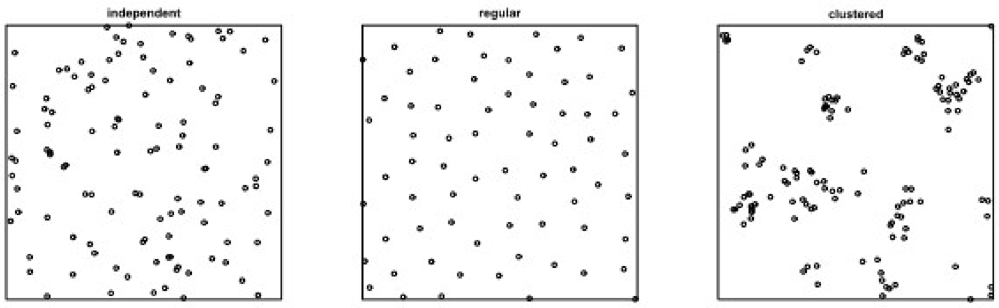


Figure 2-11: Dot maps with an independent, regular or clustered distribution (Baddeley et al., 2015)

2.2.2.2 Visualisation of spatial point pattern

The most frequently used method for displaying spatial point pattern is a dot map. Looking at the examples in Table 2-3, one can imagine that visual inspection of spatial datasets could be difficult.

2.2.3 Exploratory analysis of spatial point patterns

As shown in Table 2-3 the exploratory analysis of point pattern uses methods like quadrat counts, kernel estimation, nearest-neighbour distances and K-function analysis.

2.2.3.1 Density based analysis

First order effects for point patterns can be examined with techniques first mentioned: **quadrat counts** and **kernel estimation** (O'Sullivan and Unwin, 2010):

- Quadrat methods can be compared with counting the points on imaginary graph paper with equal size squares and applying summary statistics to the counts per quadrat divided by area. The result is a simple indication of the variation of the intensity of the underlying process unfortunately often associated with a loss of information caused by the aggregation (Figure 2-12, left).
- Kernel estimation however, uses the original point locations to produce a smooth bivariate histogram of intensity (Figure 2-12, right).

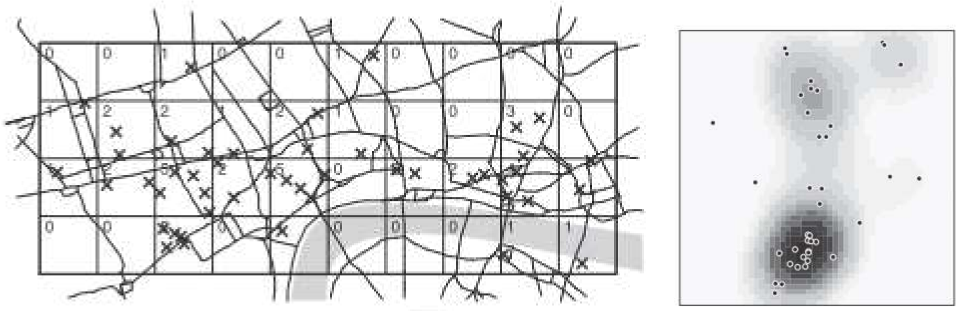


Figure 2-12: left: quadrat count, right: surface from KDE including the original point pattern (O'Sullivan and Unwin, 2010)

2.2.3.2 Distance based analysis

The basic idea of describing the **Second order properties** of point patterns is to investigate the distances between the points, the so-called **nearest-neighbour distances**.

We can distinguish between two approaches: the distance between a randomly selected event and the nearest neighbouring event on the one hand and the distances between randomly selected locations in space and the nearest event on the other hand. Here, the spatial dependence is described by visual analysis of the distribution of the calculated nearest-neighbour distances.

Also, including the greater distances the **K - function** is helpful to describe the kind of distribution of a point pattern. The right picture in Figure 2-13 shows the typical K-function for a clustered and a regular spaced event. Keeping in mind the important role of the K-Function in analysing fitted models it will be explained more detailed in section 0.0.0.0.

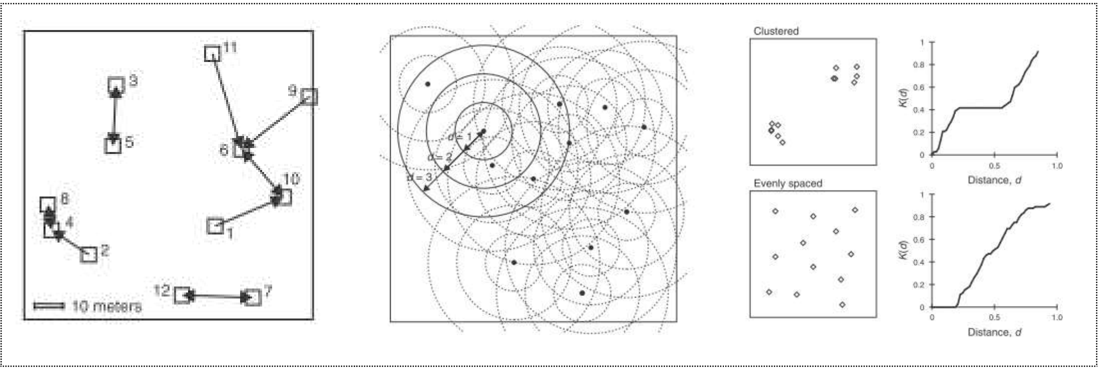


Figure 2-13: left: Distances to the nearest neighbour, middle: Determining the K function for a pattern, right: K function for clustered and evenly spaced events (O'Sullivan and Unwin, 2010)

In practice both approaches cannot be separated.

Ripley's K-function

The K-function is one of the most popular tools for investigating the departure from complete spatial randomness (Ripley, 1977a). The K-function and the additional L-function (Besag, J., 1977) are closely related descriptive statistics for detecting deviations from CSR.

Both describe the interaction or spatial dependence between events varying through space. Under stationary and isotropic hypothesis, the K-function is defined by

$$K(r) = \frac{1}{\lambda}E$$

the number of further events occurring within distance r of an arbitrary event of the process, where r > 0 and E denotes the mathematical expectation.

The quantity λ is the intensity of the point process (events / unit area). The K-function can also be defined by the second order intensity function:

$$K(r) = 2\pi\lambda^{-2}\int_0^r \lambda_2(u)\,u du.$$

To put it simply, the K-function describes the number of events depending on λ, in a specific radius r centred around a random event, best illustrated as a cumulative function.

One advantage of the K-function is that its theoretical values are known for several useful models of spatial point processes, e.g. processes with no spatial dependence the K-function is simply:

$$K(r) = \pi r^2$$

The estimator of the K-function is defined following the second order intensity function:

$$\widehat{K}(r) = \frac{|D|}{n^2}\sum_{i=1}^{n}\sum_{j\neq i}^{n}\frac{I(\|s_i - s_j\| \le r)}{w_{ij}}$$

where |D| is the area of a region D. n is the number of events (e.g. number of tweet or flickr messages), ‖·‖ is the Euclidean spatial distance between the points s_i and s_j, and finally I (·) is the indicator function and w_{ij} is the edge effect.

Ripley's K-function estimator can be compared to the one expected for a CSR process. This comparison provides valuable information on the point process distribution.

If the K-function deviates from CSR, the reason can be that events are interacting or having some effect on each other. Keeping in mind that the intensity of the process does not necessarily have to be constant across the region, also a trend in the pattern could be indicated, this could be a reason for rejecting the CSR hypothesis, or the tendency towards either clustering or regularity.

Thus, the second mentioned L-function L(r) is proportional to r, it can tell the spatial scale on which clustering occurs and the square root transformation stabilizes the fluctuations that could occur in K-function.

$$L(r) = \sqrt{\frac{K(r)}{\pi}}.$$

Under complete spatial randomness L(r) = r.

As mentioned in Section 2.2.4.1, spatial point processes could be inhomogeneous so that homogeneous measures would overestimate the dependences between events. Therefore, some similar second order properties for inhomogeneous K-function and L-function are defined:

$$K_{inhom}(r) = \frac{1}{|D|}E \qquad L_{\text{inhom}}(r) = \sqrt{\frac{K_{\text{inhom}}(r)}{\pi}}$$

The estimator for the inhomogeneous case is:

$$\widehat{K}_{\text{inhom}}(r) = \frac{1}{|D|}\sum_{i=1}^{n}\sum_{j\neq i}^{n}\frac{I(\|s_i - s_j\| \leq r)}{w_{ij}\left(\widehat{\lambda}(s_i)\widehat{\lambda}(s_j)\right)}$$

Under the assumption of inhomogeneity, it is important to recognize that the intensity of events depends on the locations of the events. The process is called inhomogeneous Poisson process.

2.2.3.3 Dependence between points

A feasible method to determine if a point pattern process is independent, regular or clustered is the **Morishita Plot** (Figure 2-14). The X^2- statistics are calculated from the subdivided quadrat counts of the area and are plotted linearly to the quadrat diameter and deliver reasonable plots for the different types of point pattern distributions (Morisita, 1959).

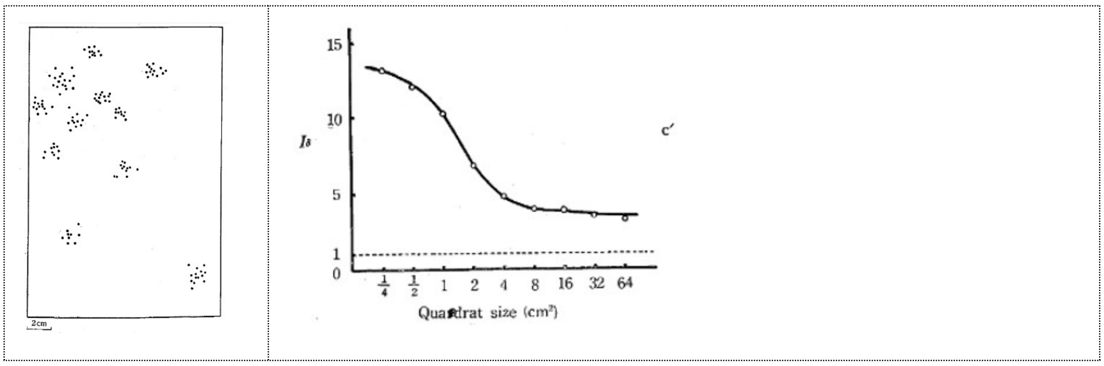


Figure 2-14: Morishita Plot of a point pattern distribution with clumps (Morisita, 1959)

2.2.4 Model-Driven Spatial Data Analysis

In this section point pattern x observed will be treated as a realisation of a random point process X in 2-dimensional space. A point process is a random set of points, the number of points is random and even the locations of the points. Estimating parameters of the distribution of X is the main objective. A complete mathematical definition of spatial point processes is not discussed in this thesis and can be found in (Møller and Waagepetersen, 2004).

The field of Spatial point processes has been investigated by statisticians and researchers for more than 35 years. The basic concepts were introduced and discussed by Ripley (Ripley, 1977a), Møller and Waagepeternsen (Møller and Waagepetersen, 2004), Baddeley (Baddeley et al., 2016, 2016, 2015) and Diggle (Diggle, 2014, 1985; Diggle et al., 2007).

A discussion of residual analysis for spatial point processes and a definition of the residuals of spatial point processes plus proposals for checking goodness of fit for fitted models was done by Baddeley, Turner, Møller and Hazelton (Baddeley et al., 2005a). A voronoi residual analysis of spatial point pattern is discussed as a useful addition to standard or pixel-based residual analyses detecting model misspecification by Bray et al. (Bray et al., 2014).

Most of these concepts have been implemented in the statistical R-package **spatstat** by Adrian Baddely and Rolf Turner (“spatstat - Resources,” n.d.). All calculations in the following chapters are made with R and mostly with the use of the spatstat-package.

2.2.4.1 Modelling spatial point pattern

The explanation of the observed point pattern is the main objective of the spatial point pattern techniques and involves the comparison with the model of **complete spatial randomness (CSR)** (Baddeley et al., 2015; Diggle, 2014; Møller and Waagepetersen, 2004; Ripley, 1977b).

The distribution of a randomly generated process of point pattern can be described by a **homogeneous Poisson process**. This implies that the event has an **equal probability** of occurring at any position in the study area and occurrence is independent from the location of any other event. Also, the first order and second order effects are absent.

Thinking of a natural disaster, e.g. a flood some people might twitter some messages at some points of time. If we mapped this, we would obtain a random number of points at randomly distributed locations at a random time. The process could be treated as a spatio(-temporal) point process.

An important assumption for the probability of the point processes in R^2 is **stationarity**, which means invariance under translation. Determining invariance under rotation, the point process is called **isotropic**.

Analysing if the investigated point process has a, which has a regular, clustered or random distribution (Figure 2-11), is tested against this basic model.

Methods testing the complete spatial randomness are based on quadrat counts or nearest-neighbour distances, like the K-function.

If the observed point pattern couldn't be described with a homogeneous Poisson process model, alternative models could be used, such as the **heterogeneous Poisson process**, the Gibbs process, the Cox process, the Poisson cluster process or Markov point process (Baddeley et al., 2015) (O'Sullivan and Unwin, 2010) (Bailey and Gatrell, 1995).

Getis et al. (Getis and Ord, 2010) explain the use of distance statistic G in order to assess spatial autocorrelation for point patterns.

In Chapter 3 we will examine how spatio – temporal point processes based on the occurrence of social media feeds could be investigated with the help of spatial point pattern processes. Also, we will try to figure out their limitations in the use for this case.

2.2.4.2 Marks and covariates

The main differences between marks and covariates are that marks are associated with the events or respectively the data points and marks are part of the response (the observed point pattern) while covariates are explanatory for the observed area (Baddeley et al., 2015).

Examples of marks are additional variables for points like a point process of earthquake epicentres or hurricane locations, or in the case of twitter data the time a tweet message was sent, which could be alternatively be viewed as a point process in space-time with coordinates (longitude, latitude, time).

Examples of covariates are e.g. information about population density or information from a digital elevation model. Often the covariate pattern is used as a surrogate spatial function Z.

Mathematically a marked point process is defined as a point process of points in space S with marks belonging to a set M defined as a point process in the Cartesian product $S \times M$.

2.2.4.3 Edge effects

Important to mention is the sampling bias determined by the selection of the observation window of the point process. As shown in section 2.2.2.1 a point process X extends throughout 2-D space, but is observed only inside an area W. So biases in the distance measurements are unavoidable because the observations concerning to a window W implies that the observed distance d(u, x) = d(u, X ∩ W) to the nearest data point inside W may be greater than the true distance d(u, X) to the nearest point of the complete point process X as shown in Figure 2-15 (Baddeley et al., 2015).

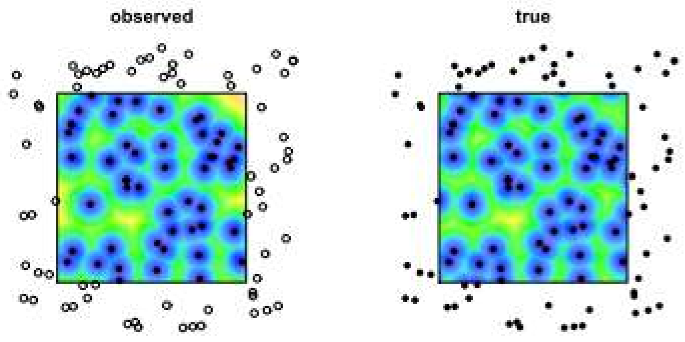


Figure 2-15: an example of the edge effect

2.2.4.4 Spatial autocorrelation

Testing for **spatial autocorrelation** is a common method which helps to understand spatial dependency or in other words the degree to which one object is similar to other nearby objects. E.g. **Moran's I (Index)** is used to measure spatial autocorrelation (Moran, 1950).

Moran's I can be classified as: positive (if similar values cluster together), negative (if dissimilar values cluster together) and no spatial autocorrelation as shown in Figure 2-16.

The Correlation of a variable with itself through space is described by Tobler's first law of geography: *"Everything is related to everything else, but near things are more related than distant things."* (Tobler, 1970).

The importance of spatial auto-correlation rests upon the fact that statistical analyses are based on the assumption that the values of observations in each sample are independent of one another. But if spatial autocorrelation happens in a positive sense (Figure 2-16, d and e), this violates the previous assumption because the samples taken from nearby areas are related to each other and are not independent (Goodchild, 1986) (O'Sullivan and Unwin, 2010).

Common applications of spatial autocorrelation are appearing in the analysis of clusters and dispersion of ecology and disease in the medical disciplines (Munasinghe and Morris, 1996; Wang et al., 2016).

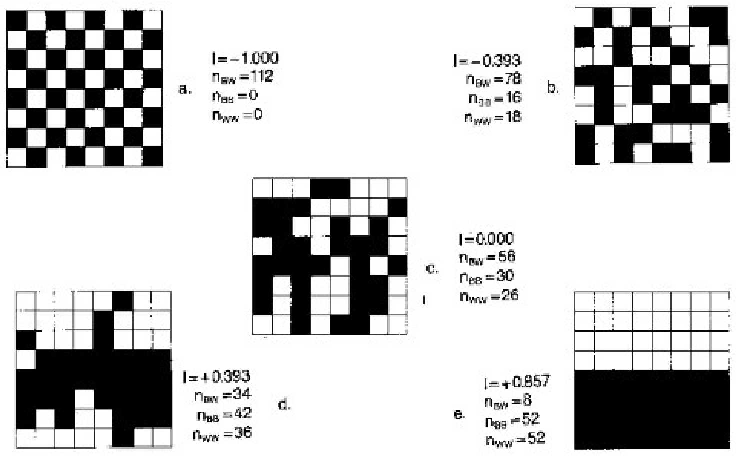


Figure 2-16: Varying levels of spatial autocorrelation with Moran's I (Goodchild, 1986).

Several measures are available e.g.: Join Count Statistic, Moran's I, Geary's C ratio, General (Getis-Ord) G, Anselin's Local Index of Spatial Autocorrelation (LISA).

Another important application area for spatial autocorrelation is the interaction with the ordinary least squares regression (OLS) because the correlation coefficients will be biased and their precision exaggerated. The bias causes correlation coefficients being higher than they really are, e.g. in areas with higher concentrations of events which just have a greater impact on the model estimate.

Moreover considering that events tend to be more concentrated, the precision could be overestimated (leading to a lower standard error) because the events are more likely to be found statistically significant (ESRI, 2016) (Wooldridge, 2009).

2.2.4.5 Hot Spots

The measures e.g. a Moran's I indicate that the spatial patterns of an observed event are clustered, but it gives no hints of the location of the clustering (Getis and Ord, 2010). Therefore, a local statistic of autocorrelation is required. Two important classes of methods for cluster detection are **Getis Ord Gi*** and **Anselin's Local Index of Spatial Autocorrelation (LISA)**. These methods measure the association between a value at a particular place and values for nearby areas (Cromley and McLafferty, 2012).

The Getis Ord Gi* statistic identifies clusters as a region having unusually high counts or rates of events when compared to values in the surrounding areas defined by a spatial weights matrix. The Gi* statistic compares a local sum with a global sum and results positive when high rates of events cluster in a local neighbourhood occur (Getis and Ord, 1992) (Kelejian and Prucha, 1999).

Using the **Local Indicators of Spatial Autocorrelation (LISA),** a statistical correlation is measured between the value of an attribute in subarea and values in nearby subareas (Anselin, 1995).

A positive LISA statistic is an identifier of a spatial concentration of similar values. Thereby, high values representing a hotspot and low values representing cold spots. Negative LISA statistics indicates a spatial pattern where areas with high attribute rates are surrounded with low rates of the specific attribute and vice versa.

The statistical significance of the LISA output could be calculated by using a **Monte Carlo method.** A Monte Carlo significance test involves simulating the distribution of a test statistic such as LISA under a null hypothesis. This null hypothesis is generated via repeated random sampling or random observed value of the test statistic to the simulated values in the reference distribution (Gorr and Olligschlaeger, 2010) (Cromley and McLafferty, 2012).

2.2.5 Process schema for the spatial point pattern analysis

For the analysis of the point pattern in this thesis the chosen process is designed as a process diagram:

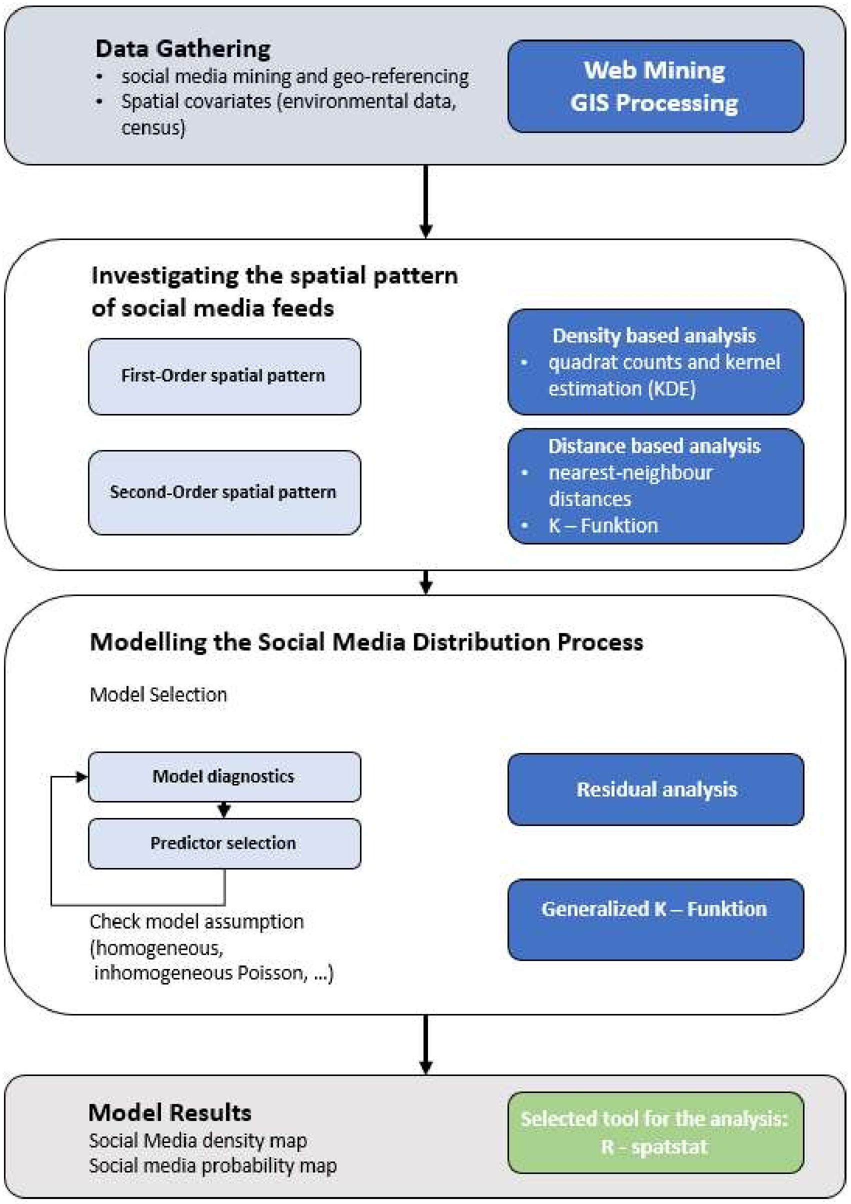

Data Gathering
• social media mining and geo-referencing
• Spatial covariates (environmental data, census)
Web Mining
GIS Processing
Investigating the spatial pattern of social media feeds
First-Order spatial pattern
Second-Order spatial pattern
Density based analysis
• quadrat counts and kernel estimation (KDE)
Distance based analysis
• nearest-neighbour distances
• K – Funktion
Modelling the Social Media Distribution Process
Model Selection
Model diagnostics
Predictor selection
Check model assumption (homogeneous, inhomogeneous Poisson, ...)
Residual analysis
Generalized K – Funktion
Model Results
Social Media density map
Social media probability map
Selected tool for the analysis: R - spatstat

Figure 2-17: Schema of the spatial point analysis for social media feeds modified from (Yang et al., 2007)

2.3 Introducing the software R

The analytical part of this thesis will be performed by using the R programming language. The following description is a direct quote from the R homepage (The R Foundation, 2016):

“R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R can be considered as a different implementation of S. There are some important differences, but much code written for S runs unaltered under R.

R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, …) and graphical techniques, and is highly extensible. The S language is often the vehicle of choice for research in statistical methodology, and R provides an Open Source route to participation in that activity.

One of R’s strengths is the ease with which well-designed publication-quality plots can be produced, including mathematical symbols and formulae where needed. Great care has been taken over the defaults for the minor design choices in graphics, but the user retains full control.”

The version of R in the given context is version 3.2.3 which uses R Studio for editing. Additional extensions for a huge variety of use cases can be downloaded as additional packages from the CARN Project website (CRAN, 2016).

2.3.1 Model - driven spatial point pattern (SPP) analysis with spatstat

Spatstat is an R package made for analysing spatial point pattern data. Exploratory data analysis, model-fitting and simulation its included in the functionality. It is designed to handle realistic datasets, including inhomogeneous point patterns, spatial sampling regions of arbitrary shape, extra covariate data, and ‘marks’ attached to the points of the point pattern(Baddeley et al., 2015). Spatstat also contains techniques to handle inhomogeneous point patterns. They include K_{inhom} (the inhomogeneous pendant of the K-function). The functionality is primarily designed for 2D spatial point patterns. It has a little support for 3D, and very basic support for space-time).

The key feature of spatstat is its ability to fit parametric models of spatial point processes to point pattern data. This includes solutions for calculating spatial trends, interpoint interactions of any order, and dependence on marks am covariates.

Models are fitted by a function ppm which is analogous to Generalized Linear Models. The fitted model objects can be printed, plotted, predicted, and even simulated. Methods for computing residuals and plotting model diagnostics are released step by step. Spatstat has a very active and vital community and is very well supported by its authors: A. Baddeley, E. Rubak and R.Turner (Baddeley et al., 2015).

The most important functions are introduced shortly:

Fitting a model to data

The model-fitting function is called ppm and is strongly analogous glm. It is called in the form

```
fit <- ppm(X, ~trend, interaction, ...)
```

where X is the point pattern dataset, trend is the R-formula describing the spatial trend and interaction is an object of the spatial class “interact” describing the stochastic dependence between the point pattern. In addition, other arguments control residuals and controls of the fitting process).

Interaction Terms can be defined for the Poisson process, the Strauss process with a hard core, the Pairwise soft core interaction, the Pairwise interaction with step function potential, Diggle-Gratton potential, Lennard-Jones potential, Geyer’s saturation process and the Ord’s process with threshold on cell area.

Fitting models to multitype point patterns

The function ppm will also fit models to multitype point patterns. A multitype point pattern is a point pattern in which the points are marked with classifiers as finite number of possible types (e.g. species, colours, on/off states). A marked pattern is in spatstat a multitype point pattern, represented by a ppp- object X whose marks are a factor.

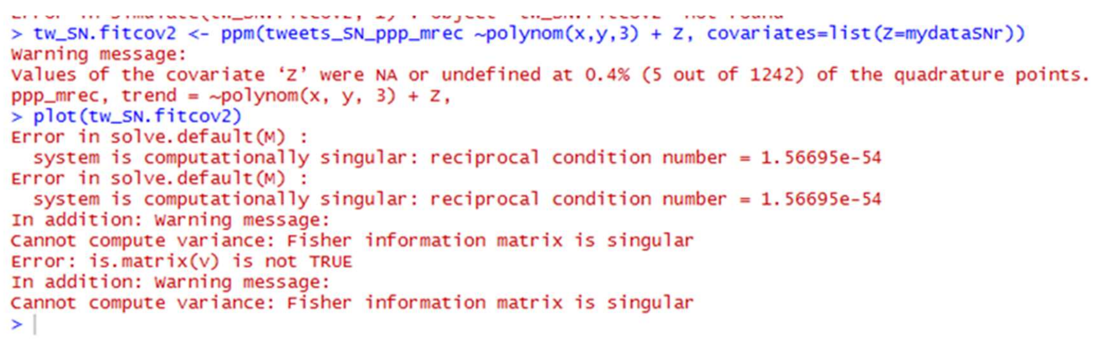
```
ppm(X, ~ marks, Poisson())
#~ marks * polynom(x,2)
#~ marks + marks:polynom(x,2)
```

Models with covariates are point process models in which the point pattern is dependent on spatial covariates. This can be e.g. altitude, population density or a distance to another spatial pattern). Any covariate data may be used under the following conditions: covariate must be a quantity Z(u), the values Z(xi) of Z at each point of the dataset and some other points must be available.

```
ppm(X, ~ log(altitude) + pH, covariates=list(pH=phimage, altitude=image3))
```

2.3.1.1 Numeric errors during the calculation

During the computation, occasionally some error messages arrived, which should be mentioned:



```
[illegible]
> tw_SN.fitcov2 <- ppm(tweets_SN_ppp_mrec ~polynom(x,y,3) + Z, covariates=list(Z=mydataSNr))
Warning message:
Values of the covariate ‘Z’ were NA or undefined at 0.4% (5 out of 1242) of the quadrature points.
ppp_mrec, trend = ~polynom(x, y, 3) + Z,
> plot(tw_SN.fitcov2)
Error in solve.default(M) :
  system is computationally singular: reciprocal condition number = 1.56695e-54
Error in solve.default(M) :
  system is computationally singular: reciprocal condition number = 1.56695e-54
In addition: Warning message:
Cannot compute variance: Fisher information matrix is singular
Error: is.matrix(v) is not TRUE
In addition: Warning message:
Cannot compute variance: Fisher information matrix is singular
> |
```

The problem is known by the author of the R-package and he will fix it by time. The problem is obviously a numerical problem depending on the usage of too high numbers in the coordinates. The recommendation of rescaling the data didn't always help. The whole explanation can be found at: http://stackoverflow.com/questions/39314362/cannot-comprehend-this-error-message-in-spatstat-in-r-while-using-kppm-function.

> *"A matrix is 'singular' if its determinant is zero, so that it cannot be inverted. It is 'computationally singular' if the determinant is very close to zero, so that a computer can't invert the matrix using its standard numerical procedures.*
>
> *The Fisher information matrix is a fundamental property of a fitted model, and it must be inverted if we want to calculate the standard error of a parameter estimate, or confidence intervals, etc.*
>
> *The most likely explanation for your problem is that the coordinates in your dataset are very large numbers (e.g. expressed in metres) so that the fitted model coefficients are correspondingly small numbers, so that the Fisher information matrix has very small entries, so it is computationally singular. Although the model can be fitted, when you print it the software tries to calculate standard errors, and then it falls over."*

3 Results

3.1 Statistical text analysis

3.1.1 Text Mining using R

One focus of this thesis is to analyse text information of social media, e.g. provided by twitter. It is typical of these kinds of messaging systems that the pure text information is short, limited to 140 characters like twitter.

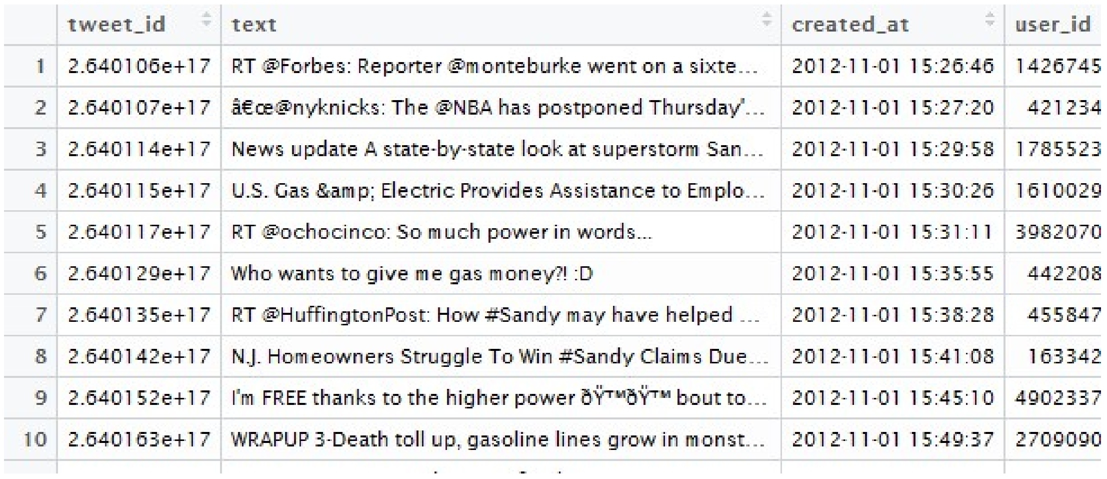
The following processes of Parsing and Filtering, Data Transformation, TM Algorithms, Analysis and Evaluation use the software R with the text mining framework TM published in the *Journal of Statistical Software* (Feinerer et al., 2008) (Feinerer et al., 2015).

Challenging text mining normally processes unstructured and semi structured text into a structured vector-space model. These steps of pre-processing are usually the same for every mining task and they need to be done prior to this. The basic steps are as follows:

3.1.1.1 Choosing the scope of the text

For mining twitter data, the scope of the text can easy be determined. We take the text, which is 140 characters long and translate it into a single vector for each message in a data frame (Figure 3-1).

```
> d <- tw_miami
> head(d$text,10)
```



	tweet_id	text	created_at	user_id
1	2.640106e+17	RT @Forbes: Reporter @monteburke went on a sixte…	2012-11-01 15:26:46	1426745
2	2.640107e+17	â€œ@nyknicks: The @NBA has postponed Thursday'…	2012-11-01 15:27:20	421234
3	2.640114e+17	News update A state-by-state look at superstorm San…	2012-11-01 15:29:58	1785523
4	2.640115e+17	U.S. Gas &amp; Electric Provides Assistance to Emplo…	2012-11-01 15:30:26	1610029
5	2.640117e+17	RT @ochocinco: So much power in words...	2012-11-01 15:31:11	3982070
6	2.640129e+17	Who wants to give me gas money?! :D	2012-11-01 15:35:55	442208
7	2.640135e+17	RT @HuffingtonPost: How #Sandy may have helped …	2012-11-01 15:38:28	455847
8	2.640142e+17	N.J. Homeowners Struggle To Win #Sandy Claims Due…	2012-11-01 15:41:08	163342
9	2.640152e+17	I'm FREE thanks to the higher power ðŸ™ðŸ™ bout to…	2012-11-01 15:45:10	4902337
10	2.640163e+17	WRAPUP 3-Death toll up, gasoline lines grow in monst…	2012-11-01 15:49:37	2709090

Figure 3-1: dataframe tw_miami, first 10 tweets

Choosing the proper scope depends on the goals of the text mining task: for classification or clustering tasks, for sentiment analysis or information retrieval.

3.1.1.2 Preparing the corpus

The main tasks of preparing the corpus is removing noise and clutter to avoid misinterpretations in the text mining algorithms.

```
library(tm)
```

```
docs <- Corpus(VectorSource(d$text))
```

```
toSpace <- content_transformer(function(x, pattern) gsub(pattern, " ", x))
removeNumPunct <- function(x) gsub("[^[:alpha:][:space:]]*", "", x) # remove anything other than Eng-
lish letters or space
docs <- tm_map(docs, content_transformer(removeNumPunct))
docs <- tm_map(docs, toSpace, "/|@|\\|")
docs <- tm_map(docs, content_transformer(tolower))
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, removeWords, stopwords("english"))
docs <- tm_map(docs, removeWords, c("own", "stop", "words"))
docs <- tm_map(docs, stripWhitespace)
toString <- content_transformer(function(x, from, to) gsub(from, to, x))
docs <- tm_map(docs, toString, "specific transform", "ST")
docs <- tm_map(docs, toString, "other specific transform", "OST")
docs <- tm_map(docs, stemDocument)
```

The R TM-package delivers many content transformers for a text corpus. They should be introduced at this point:

- The **lowercase**-transformer converts the entire document to completely lower case, because the uppercase word at the beginning of the sentence should be treated no differently than the same word in lower case.
- The **remove numbers**-transformer removes numbers assuming that numbers are rarely significant for information retrieval
- The **remove punctuation**-transformer removes insignificant information such as punctuation marks.
- The **remove special characters**-transformer removes noise and clutter like emoticons and symbols – shortly every character which is not a word will be removed. Emoticons could be an information for sentiment mining and should in this case be treated separately.
- The **remove own words**-transformer gives the possibility to remove and own dataset of words in addition to the TM-package which could be useful investigating e.g. communities with a special code.
- The **strip whitespace**-transformer removes redundant spaces.
- The **specific transformations**-transformer can be configured for special terms one will need to perform, e.g. to transform abbreviations.
- The **stopwords**-transformer removes commonly used words which are insignificant fill-words and don't determine the topic of a tweet or his sentiment. The TM-package gives the possibility to define own stopword-lists.

- The **sparse terms**-transformer removes terms with a defined occurrence. They will be removed e.g. by a counter of one. Such “sparse” terms can be removed from the document term matrix quite easily using removeSparseTerms().
- The **stemming**-transformer uses an algorithm that removes common word endings for English words, such as “es”, “ed” and “’s” reducing the deviations of words, which have the same assertion.

After cleaning and transforming the corpus, a Document Term Matrix has to be created. A document term matrix is simply a matrix with documents as the rows and terms as the columns and a count of the frequency of words as the cells of the matrix - suitable for input into text mining algorithms. This is done by the DocumentTermMatrix() command of the TM-package.

Storing text as weighted vectors first requires choosing a weighting scheme. The most popular scheme is the TF-IDF weighting approach. TF-IDF stands for term frequency–inverse document frequency. The term frequency for a term is the number of times the term appears in a document.

```
dtm <- DocumentTermMatrix(docs)
```

Some simple analytics using the DocumentTermMatrix() are shown in Figure 3-2.

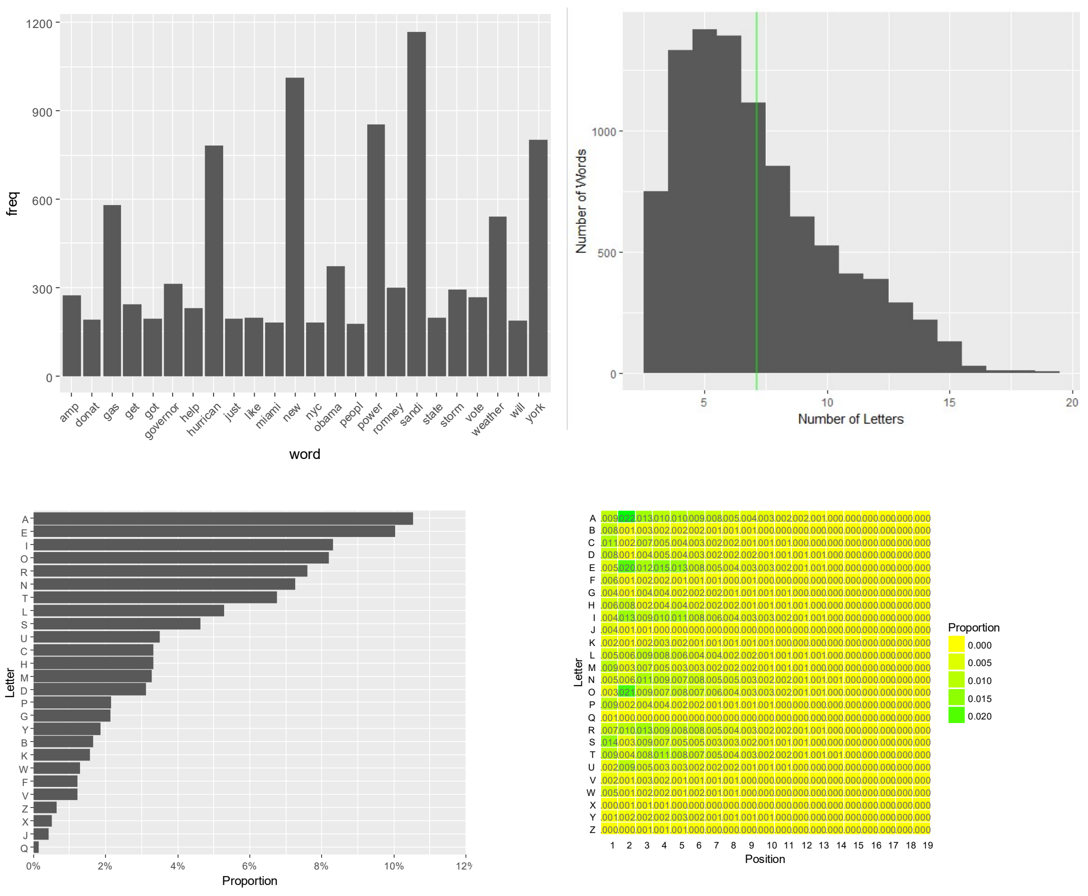


Figure 3-2: upper-left: word-frequency, upper-right: number of letters ~ number of words, lower-left: proportion of letters, lower-right: distribution of letter-position

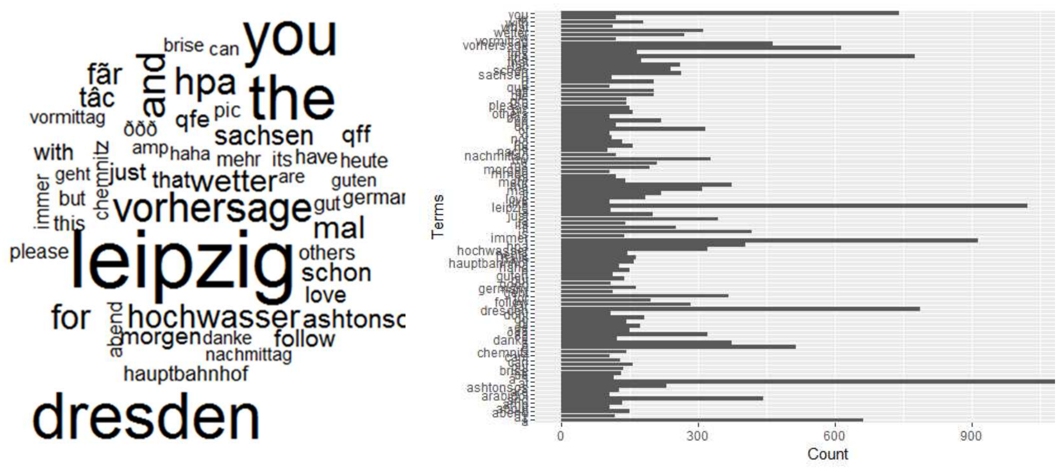


Figure 3-3: examples of a word cloud and term frequencies

Analysing the results of the term count list (the word cloud is just a visualisation of it) (Figure 3-3) is the point where the search words have to be determined. These results give no hints, which terms belong to each other or which topic they figure out. This is far from being satisfactory.

3.1.2 Topic Modelling with LDA using R

As mentioned in the section before searching the web with key words via Google or other search engines is the standard way to search for documents on the internet.

Since you have a large database and know all topics and themes inside, an automatic algorithm is needed. In R we will use the R-packages *topicmodel* (Grün and Hornik, 2016) and *LDA* (Chang, 2015) to do this.

The data

The data one will investigate is stored in a PostgreSQL database. For the study area of Miami, one transfer an extract of the data within a selected bounding box of Miami -81.1258, 25.282, -79.8294, 26.7677 to a R data frame. 262647 tweets were gathered.

Pre-processing the data

First, we will clean up the text with the same methods we have used in the section before. In particular, we use English stop words from the SMART information retrieval system(obtained from http://jmlr.csail.mit.edu/papers/volume5/lewis04a/a11-smart-stop-list/english.stop) which is integrated in the TM-package.

```
doc_tweets <- tw_miami02
#load text mining library
library(tm)
#create corpus from vector
docs <- Corpus(VectorSource(doc_tweets))
#start preprocessing
#Transform to lower case
```

```
docs <-tm_map(docs,content_transformer(tolower))
#remove potentially problematic symbols
toSpace <- content_transformer(function(x, pattern) { return (gsub(pattern, " ", x))})
docs <- tm_map(docs, toSpace, "-")
docs <- tm_map(docs, toSpace, "’")
docs <- tm_map(docs, toSpace, "‘")
docs <- tm_map(docs, toSpace, "•")
docs <- tm_map(docs, toSpace, "\"")
docs <- tm_map(docs, toSpace, "\"")

#remove punctuation
docs <- tm_map(docs, removePunctuation)
#Strip digits
docs <- tm_map(docs, removeNumbers)
#remove stopwords
docs <- tm_map(docs, removeWords, stopwords("english"))
#remove whitespace
docs <- tm_map(docs, stripWhitespace)
#Good practice to check every now and then
writeLines(as.character(docs[[30]]))
#Stem document
docs <- tm_map(docs,stemDocument)
docs <- tm_map(docs, removeWords, myStopwords)
#inspect a document as a check
writeLines(as.character(docs[[30]]))


#Create document-term matrix
dtm <- DocumentTermMatrix(docs)
#convert rownames to filenames
#rownames(dtm) <- filenames
#collapse matrix by summing over columns
freq <- colSums(as.matrix(dtm))
```

Topic modelling with LDA

The next step is computing an LDA-model with 10 topics. In this case, we will use the *topic models* package with the Gibbs sampling option (Resnik and Hardisty, 2010).

The LDA function has a large number of parameters which we will set by default, only the parameters that are needed by the Gibbs sampling algorithm were configured.

The idea behind Gibbs sampling is simulating a random walk that describes the characteristics of the desired distribution. The burn-in-period can be understood as an initial stage of the algorithm. The burn-in parameter is set to 4000. After the initial stage the algorithm passes 2000 iterations with the thin parameter defining every 500th iteration for further use in order to prevent correlations between samples. We define the start parameter as five for 5 independent runs, giving 5 random seed integers.

The reason we do this is to avoid correlations between samples. We use 5 different starting points (nstart=5) – that is, five independent runs. Each starting point requires a seed integer (this also ensures reproducibility), so I have provided 5 random integers in my seed list. The parameter best is set to TRUE by default returning the highest posterior probability.

This block of code takes about 8 minutes to run on a XMG notebook using a quad core i7 3,2Ghz processor (and 16 GB RAM).

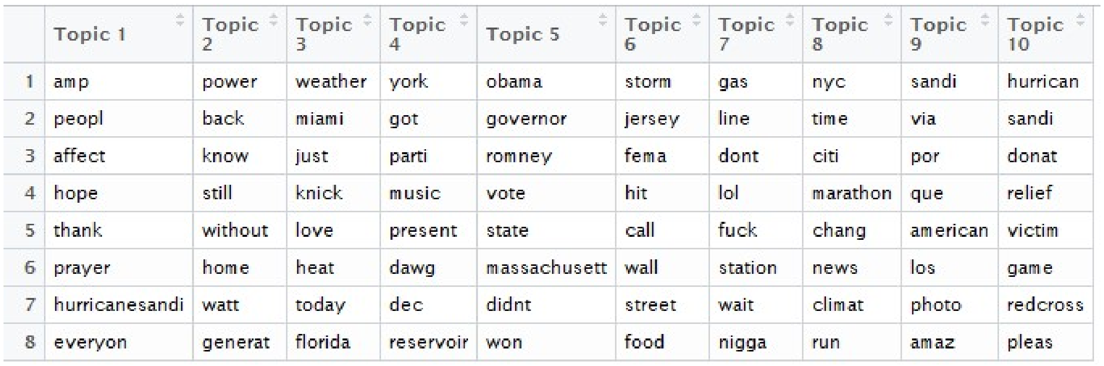
```
library(topicmodels)

# parameters for Gibbs sampling
burnin <- 4000
iter <- 2000
thin <- 500
seed <-list(1969,5,25,102855,2012)
nstart <- 5
best <- TRUE

#Number of topics
k <- 10
#Run LDA with Gibbs
ldaOut <-LDA(dtm, k, method="Gibbs", control=list(nstart=nstart, seed = seed, best = best, burnin =
burnin, iter = iter, thin=thin))
#write out results
#docs to topics
ldaOut.topics <- as.matrix(topics(ldaOut))
#top 8 terms in each topic
ldaOut.terms <- as.matrix(terms(ldaOut,8))
topicProbabilities <- as.data.frame(ldaOut@gamma)
```

The configured parameters are taken from the examples of the documentation of the R-packages used, so one can’t be sure whether these parameters define the best solution for the algorithm. Modifying the settings of the parameters to check the deviation of the results is recommended. Even the parameter k (number of the topics) should be modified. The results must be scored from a practical point of view, even if it looks like a try and error game. Experience will help.

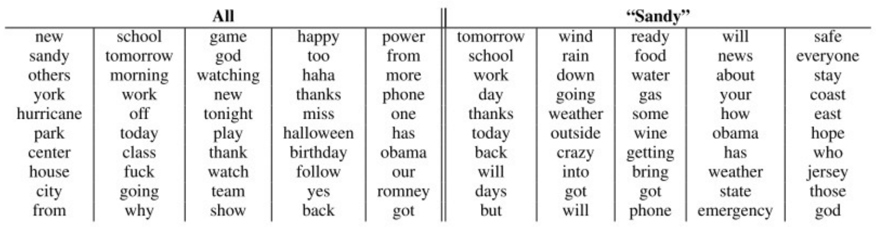
The LDA algorithm returns an object LDAOut (as we have defined) from the Class LDA, containing information about the topic assignments, the top terms in each topic and the calculated probabilities of those terms.



	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6	Topic 7	Topic 8	Topic 9	Topic 10
1	amp	power	weather	york	obama	storm	gas	nyc	sandi	hurrican
2	peopl	back	miami	got	governor	jersey	line	time	via	sandi
3	affect	know	just	parti	romney	fema	dont	citi	por	donat
4	hope	still	knick	music	vote	hit	lol	marathon	que	relief
5	thank	without	love	present	state	call	fuck	chang	american	victim
6	prayer	home	heat	dawg	massachusett	wall	station	news	los	game
7	hurricanesandi	watt	today	dec	didnt	street	wait	climat	photo	redcross
8	everyon	generat	florida	reservoir	won	food	nigga	run	amaz	pleas

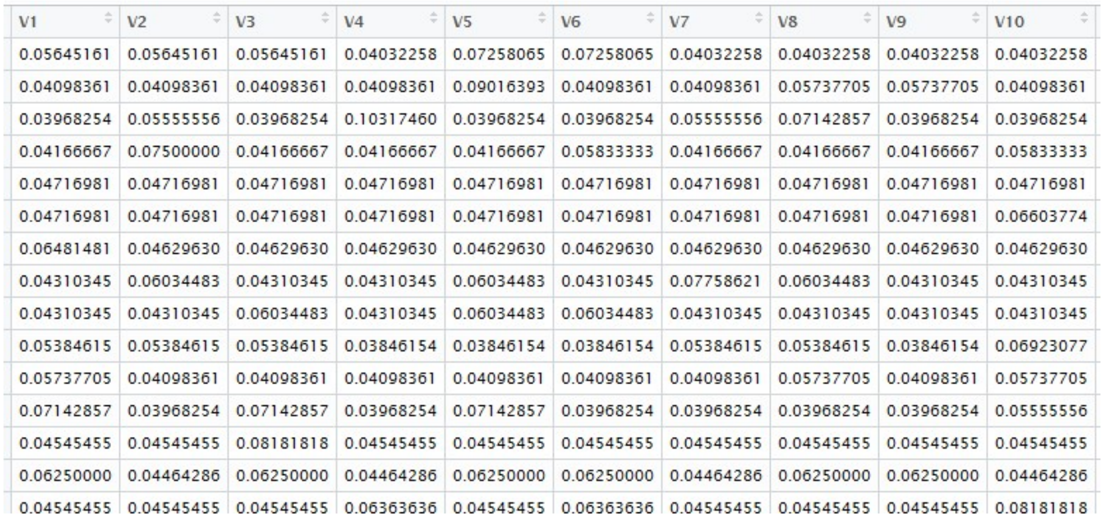
Figure 3-4: LDA-model with 10 topics with 8 terms for each topic

After a first inspection (Figure 3-4), the algorithm has delivered some reasonable results. Topic 2 describes a topic about producing energy, the topic 5 describes themes around election and topic 10 describes something about disaster and victims. Some terms are assigned to multiple topics, but this is implied in the method, examples in the literature Figure 3-5 (Wang et al., 2015) look likewise.



All					"Sandy"				
new	school	game	happy	power	tomorrow	wind	ready	will	safe
sandy	tomorrow	god	too	from	school	rain	food	news	everyone
others	morning	watching	haha	more	work	down	water	about	stay
york	work	new	thanks	phone	day	going	gas	your	coast
hurricane	off	tonight	miss	one	thanks	weather	some	how	east
park	today	play	halloween	has	today	outside	wine	obama	hope
center	class	thank	birthday	obama	back	crazy	getting	has	who
house	fuck	watch	follow	our	will	into	bring	weather	jersey
city	going	team	yes	romney	days	got	got	state	those
from	why	show	back	got	but	will	phone	emergency	god

Figure 3-5: top words from selected topics from the twitter corpus (Wang et al., 2015)



V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
0.05645161	0.05645161	0.05645161	0.04032258	0.07258065	0.07258065	0.04032258	0.04032258	0.04032258	0.04032258
0.04098361	0.04098361	0.04098361	0.04098361	0.09016393	0.04098361	0.04098361	0.05737705	0.05737705	0.04098361
0.03968254	0.05555556	0.03968254	0.10317460	0.03968254	0.03968254	0.05555556	0.07142857	0.03968254	0.03968254
0.04166667	0.07500000	0.04166667	0.04166667	0.04166667	0.05833333	0.04166667	0.04166667	0.04166667	0.05833333
0.04716981	0.04716981	0.04716981	0.04716981	0.04716981	0.04716981	0.04716981	0.04716981	0.04716981	0.04716981
0.04716981	0.04716981	0.04716981	0.04716981	0.04716981	0.04716981	0.04716981	0.04716981	0.04716981	0.06603774
0.06481481	0.04629630	0.04629630	0.04629630	0.04629630	0.04629630	0.04629630	0.04629630	0.04629630	0.04629630
0.04310345	0.06034483	0.04310345	0.04310345	0.06034483	0.04310345	0.07758621	0.06034483	0.04310345	0.04310345
0.04310345	0.04310345	0.06034483	0.04310345	0.06034483	0.06034483	0.04310345	0.04310345	0.04310345	0.04310345
0.05384615	0.05384615	0.05384615	0.03846154	0.03846154	0.03846154	0.05384615	0.05384615	0.03846154	0.06923077
0.05737705	0.04098361	0.04098361	0.04098361	0.04098361	0.04098361	0.04098361	0.05737705	0.04098361	0.05737705
0.07142857	0.03968254	0.07142857	0.03968254	0.07142857	0.03968254	0.03968254	0.03968254	0.03968254	0.05555556
0.04545455	0.04545455	0.08181818	0.04545455	0.04545455	0.04545455	0.04545455	0.04545455	0.04545455	0.04545455
0.06250000	0.04464286	0.06250000	0.04464286	0.06250000	0.06250000	0.04464286	0.06250000	0.06250000	0.04464286
0.04545455	0.04545455	0.04545455	0.06363636	0.04545455	0.06363636	0.04545455	0.04545455	0.04545455	0.08181818

Figure 3-6: topic probabilities by document

Figure 3-6 shows the lists the topic probabilities by document. One can see that the assignments overall look quite poor so only iteration can deliver better values.

Thus, iterating parameters needs time maybe nobody has or the situation demands fast action, a different approach for topic modelling should be introduced as a more playful attempt.

Topic modelling with LDAviz

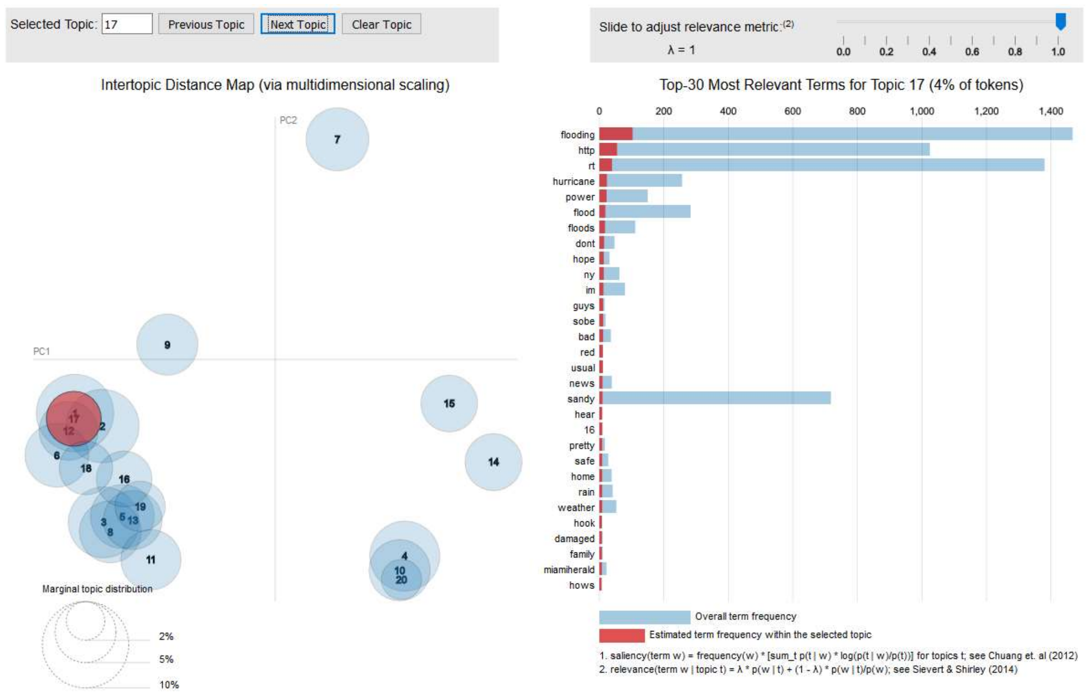
LDAvis is a web based interactive visualisation of the topics estimated by the Latent Dirichlet Allocation. The visualisation allows the user to understand the relevance of each term or topic beneath the estimation. In opposition to the listed topic probabilities, the user can flexibly explore the topic-term relations of the fitted LDA model (Chang et al., 2009).

Discovering the meaning of a topic and the prevalence of each topic and the relation of the topics to each other is a challenge and it is presented much more comfortable if it is visualized. As seen in Figure 3-7 in the left side is a global view of the model presenting the topic as circles on a 2D-plane with calculated distances of relationship in 2 dimensions (Chuang et al., 2012a). The prevalence of each topic correlates with the area of the circle.

On the right side the individual terms are listed in dependence on the topic chosen on the left side helping to decide the meaning of each topic. The bar on the right shows the topic-specific and the global-specific frequency of the topic (Chuang et al., 2012b).

Measuring the relevance of a term as a helpful attribute is following the approach from Bischof et al. (Bischof and Airoldi, 2012) who proposed to rank terms for a given topic regarding the frequency of the term under the topic and the term's exclusivity to the topic.

Concluding, the LDAvis system is a very helpful tool using LDA to obtain a good overview in the jungle of documents. Maybe someday there will be enhancements regarding the correlation between topics and in automated content-related labelling of topics (Figure 3-7).


Selected Topic: 17

Slide to adjust relevance metric:(2) λ = 1

1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Figure 3-7: Visualisation of LDAviz – right: global view of the model, left: term – topic relation

3.1.3 Sentiment Mining using R

For the sentiment analysis an algorithm from the NRC Word-Emotion Association Lexicon is used (Mohammad and Turney, 2010). The underlying idea is that that Mohammad and Turney have built a lexicon containing lots of words with associated scores for eight different emotions and the two sentiments “positive” and “negative”. Each word of the lexicon will be scored for the emotions and then summed up for a total sentiment of a sentence by adding up the individual sentiments for each word in the sentence. The lexicon is provided in over twenty languages.

The R-package providing this functionality is called syuzhet (Jockers, 2016). The package comes with four sentiment dictionaries and provides a method for accessing the robust, but computationally expensive, sentiment extraction tool developed in the NLP group at Stanford.

```
.libPaths("C:/R-packages")
library(syuzhet)

test_tw <- elbe2013_de_hochwasser
mySentiment <- get_nrc_sentiment(as.vector(test_tw$text))
tweets01 <- cbind(test_tw, mySentiment)
sentimentTotals <- data.frame(colSums(tweets01[,c(13:22)]))
names(sentimentTotals) <- "count"
sentimentTotals <- cbind("sentiment" = rownames(sentimentTotals), sentimentTotals)
rownames(sentimentTotals) <- NULL

ggplot(data = sentimentTotals, aes(x = sentiment, y = count)) +
  geom_bar(aes(fill = sentiment), stat = "identity") +
  theme(legend.position = "none") +
  xlab("Sentiment") + ylab("Total Count") + ggtitle("Total sentiment score for all tweets
elbe2013_de_hochwasser")
```

The following Figure 3-8 shows the total scores of the dataset “elbe2013_de_hochwasser”, containing all tweets with a relation to the German flood event. It is obviously, that the emotion “fear” and the sentiment “negative” predominate the other categories.

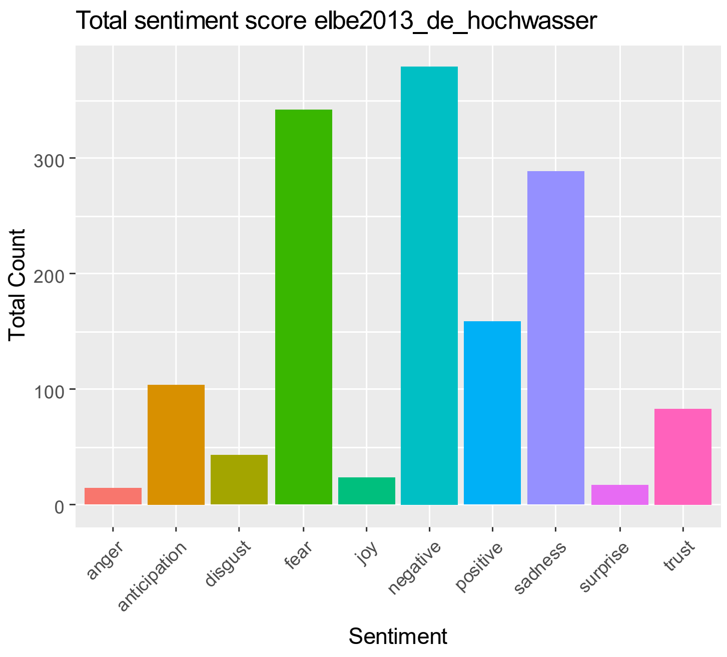


Figure 3-8: total sentiments for the dataset elbe2013_de_hochwasser

Figure 3-9 shows the also a predomination of the sentiment “negative” followed by the emotion “anger” from the data set Hurricane Sandy 2012. Comparing this with the result of the LDA-analysis in section 3.1.2, this can be explained by the topics of the presidential elections in the US included in the dataset.

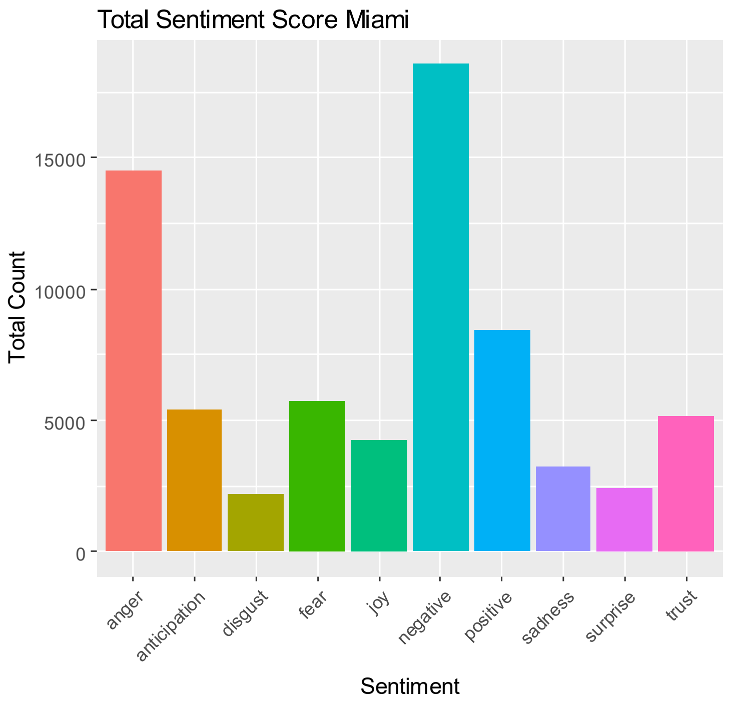


Figure 3-9: total sentiments for the dataset Miami

Investigating the chronological sequence interests in relation to their quantitively relevance to the crisis event. If a significant increase of the sentiment or the emotions would be measurable, this would indicate a dependence between the crisis event and the distribution of the sentimental and emotional content of the tweet messages.

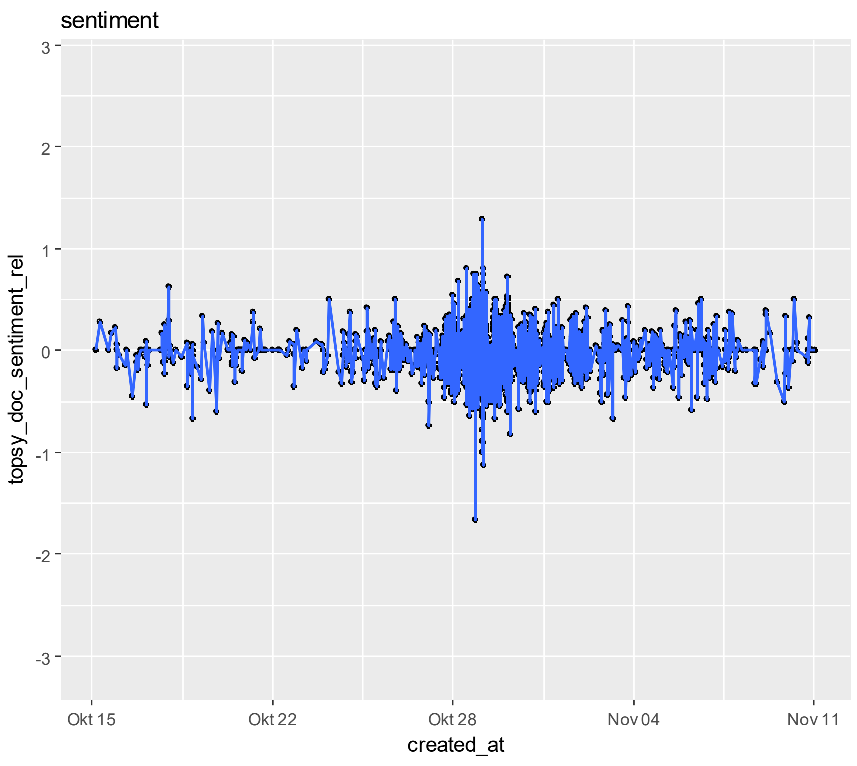


Figure 3-10: sentiment distribution during the Hurricane Sandy landfall (29.10.2012)

The data set from Hurricane Sandy (Kryvasheyeu et al., 2015b) contains the results of a sentiment analysis from the Topsy Labs with a scale from negative -2 to positive +2 (Figure 3-10).

The sentiment / emotion analysis of the syuzhet-method (Jockers, 2016) computes the eight categories in a scale from -1 to +1.

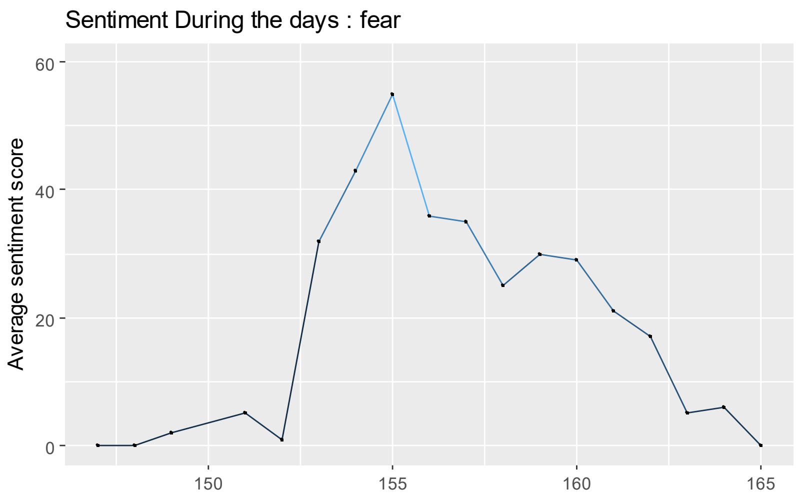


Figure 3-11: Emotion „fear“ during the German flood (day of year 151 corresponds to the 01.06.2013)

Figure 3-11 shows a rise of the emotion fear during the German flood, indicating a significant relation to the event.

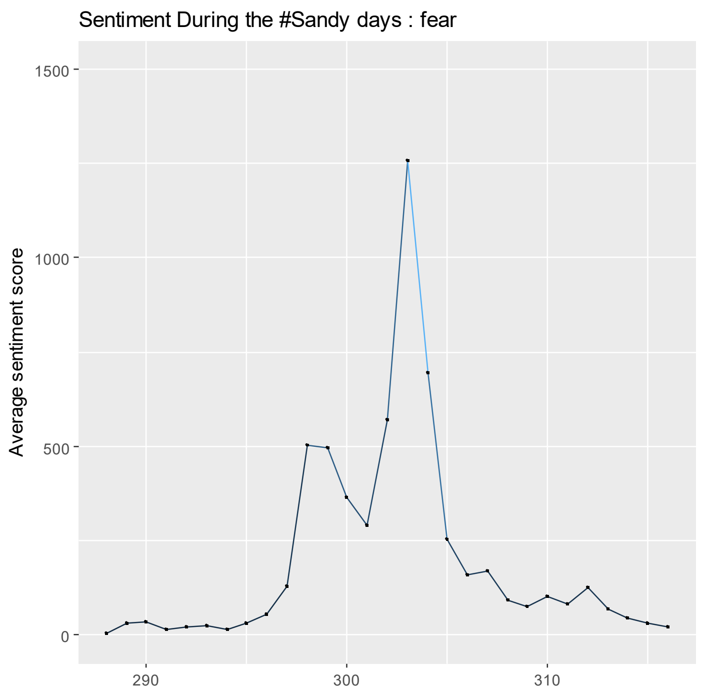


Figure 3-12: Emotion „fear“ during the Hurricane Sandy event (day of year 300 corresponds to the 26.10.2012)

The same increasing of the emotion “fear” could be computed during the hurricane Sandy event. Especially during the landfall at 29.10.2012 the emotion rises extremely strong (Figure 3-12).

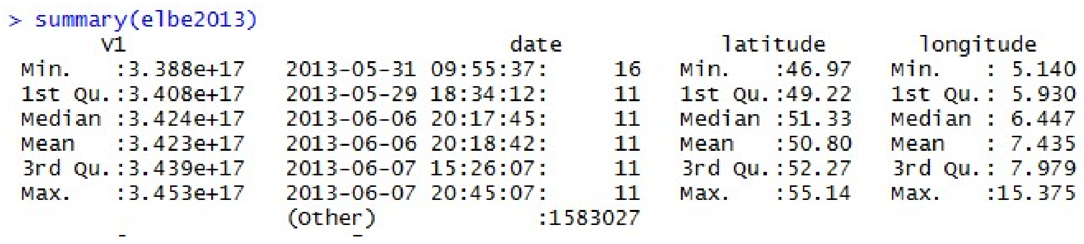
3.2 German flood in 2013

Extreme flooding in Central Europe began after several days of heavy rain in late May and early June 2013. Flooding and damages primarily affected south and east German states, western regions of the Czech Republic, and Austria. The flood crest progressed down the Elbe and Danube drainage basins (Figure 3-19) and tributaries, leading to high water and flooding along their banks.

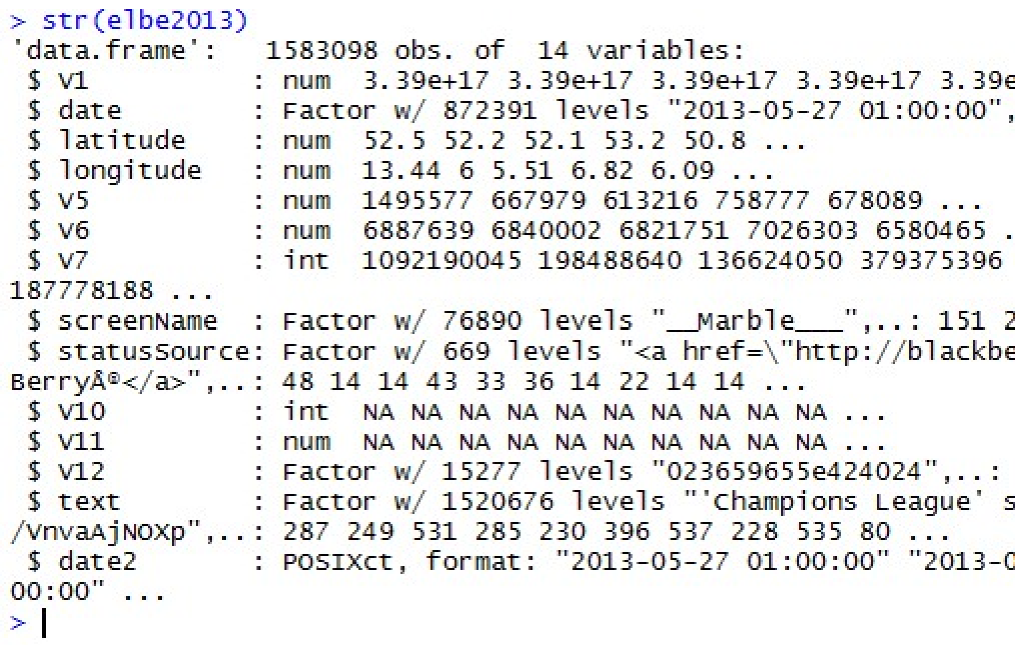
3.2.1 Data preparation

The data of the social media response of the German flood was obtained as csv-files without documentation. The files are named continuously geo_tweets_2013_05_27_00.csv to geo_tweets_2013_ 06_13_23.csv, the lines consist of data, which are separated by commas or other separators from each other. The collection reaches 27.05.2013 until 13.06.2013.

The data was imported with are and stored in a data frame with the following columns described as V1, V2, 0 ... V14). The attributes of the data were changed according to the attribute schema from twitter as best known. The data consists of geo-referenced tweets with lat/lon-coordinates. Getting the min/max of the relevant data is done by the summary-function. The dataset consists of 1583027 tweets massages, with a longitude from 5.140 until 15.375 and a latitude from 46.97 until 55.14.



```
> summary(elbe2013)
       V1                                date               latitude         longitude
 Min.   :3.388e+17   2013-05-31 09:55:37:      16   Min.   :46.97   Min.   : 5.140
 1st Qu.:3.408e+17   2013-05-29 18:34:12:      11   1st Qu.:49.22   1st Qu.: 5.930
 Median :3.424e+17   2013-06-06 20:17:45:      11   Median :51.33   Median : 6.447
 Mean   :3.423e+17   2013-06-06 20:18:42:      11   Mean   :50.80   Mean   : 7.435
 3rd Qu.:3.439e+17   2013-06-07 15:26:07:      11   3rd Qu.:52.27   3rd Qu.: 7.979
 Max.   :3.453e+17   2013-06-07 20:45:07:      11   Max.   :55.14   Max.   :15.375
                     (Other)            :1583027
```



```
> str(elbe2013)
'data.frame':   1583098 obs. of  14 variables:
 $ V1          : num  3.39e+17 3.39e+17 3.39e+17 3.39e+17 3.39e
 $ date        : Factor w/ 872391 levels "2013-05-27 01:00:00",
 $ latitude    : num  52.5 52.2 52.1 53.2 50.8 ...
 $ longitude   : num  13.44 6 5.51 6.82 6.09 ...
 $ V5          : num  1495577 667979 613216 758777 678089 ...
 $ V6          : num  6887639 6840002 6821751 7026303 6580465 .
 $ V7          : int  1092190045 198488640 136624050 379375396
187778188 ...
 $ screenName  : Factor w/ 76890 levels "__Marble___",..: 151 2
 $ statusSource: Factor w/ 669 levels "<a href=\"http://blackbe
BerryÂ®</a>",..: 48 14 14 43 33 36 14 22 14 14 ...
 $ V10         : int  NA NA NA NA NA NA NA NA NA NA ...
 $ V11         : num  NA NA NA NA NA NA NA NA NA NA ...
 $ V12         : Factor w/ 15277 levels "023659655e424024",..:
 $ text        : Factor w/ 1520676 levels "'Champions League' s
/VnvaAjNOXp",..: 287 249 531 285 230 396 537 228 535 80 ...
 $ date2       : POSIXct, format: "2013-05-27 01:00:00" "2013-C
00:00" ...
> |
```

3.2.2 Data inspection

Inspecting the gathered data is always the first step in data mining and it is noticeable, that one day is missing: exactly the data from 30.05.2013 (Figure 3-13).

Plotting the tweets as a time series presents a first impression from the temporal distribution during the observed period. The x-scale is chosen to one hour, so one can see the hourly-daily distribution of the tweets. The maximum peaks appear in the afternoon hours and later.

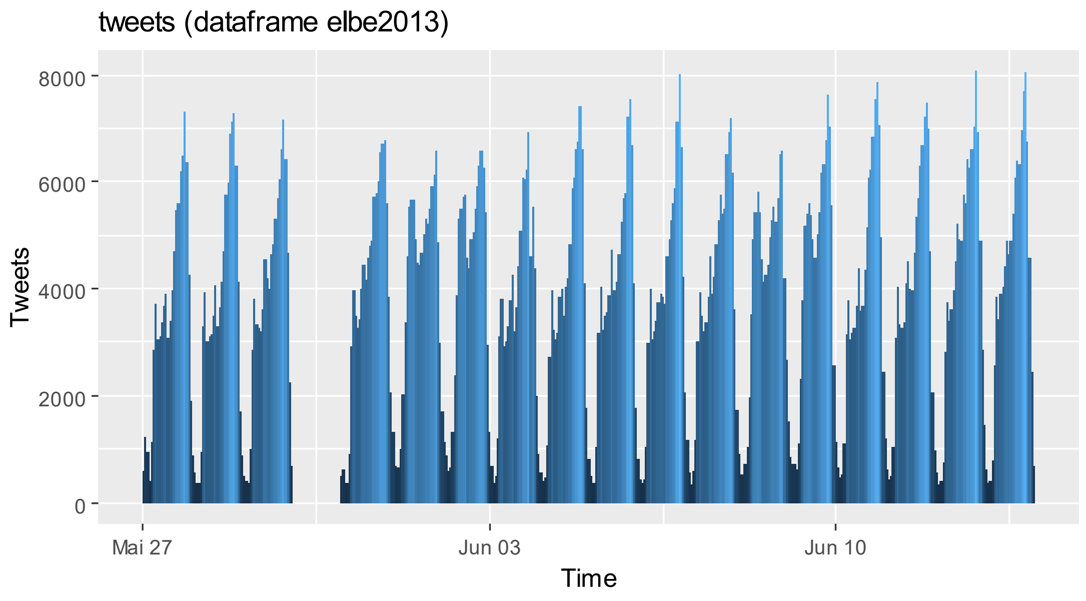


Figure 3-13: Temporal distribution of tweet messages (dataframe elbe2013)

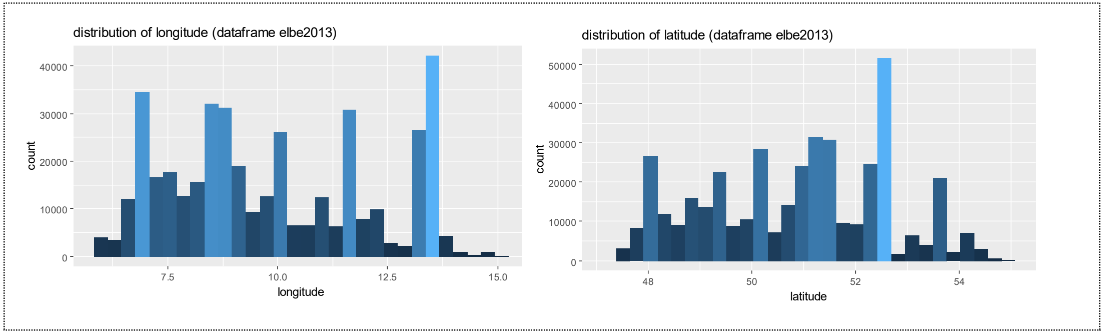


Figure 3-14: Distribution of longitude and latitude of tweet messages (dataframe elbe2013)

The distribution of the coordinates delivers valuable information too. Comparing it with the spatial distribution, a quick overview about the quantitative distribution is given (Figure 3-14).

The next step was the reduction of the dataset within the borders of Germany. 408466 tweets left. The first interpretation is, that the Netherlands using twitter two times as much (Figure 3-15).

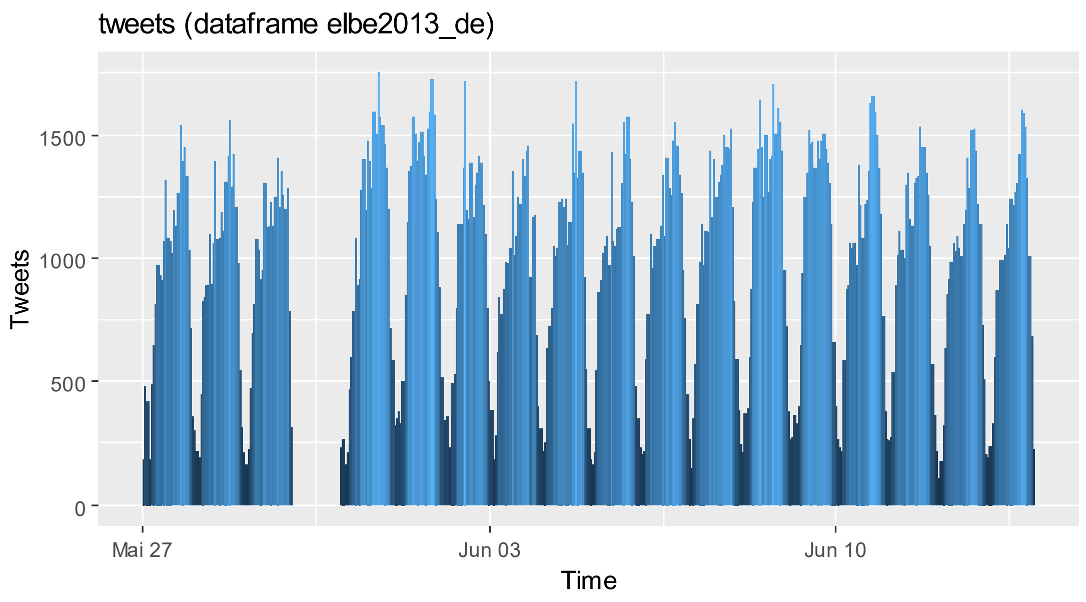


Figure 3-15: Temporal distribution of tweet messages (dataframe elbe2013_de)

Thus, we are primary interested in the flood related tweets, we extract them. The resulting dataset contains at last 1947 flood related tweet massages. In Figure 3-16 can be determined, that the spatial distribution follows the flood as expected and shown in Figure 3-17.

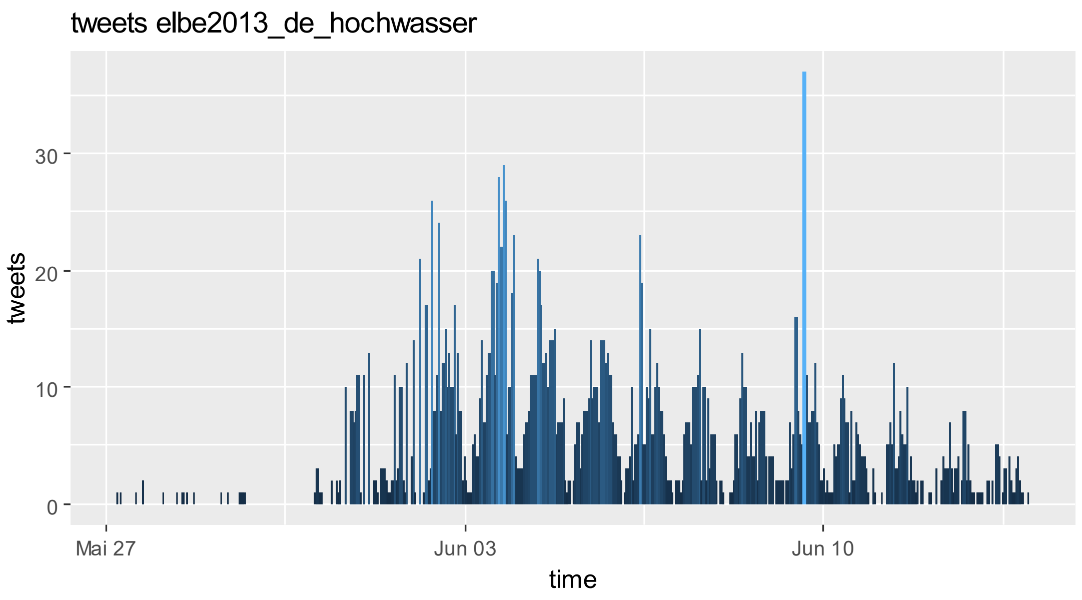


Figure 3-16: Temporal distribution of flood related tweet messages (dataframe elbe2013_de_hochwasser)

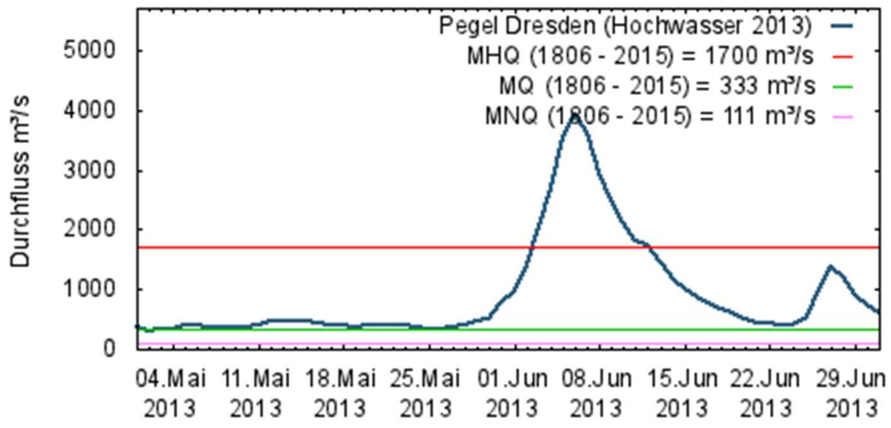


Figure 3-17: The stream flow of the gauge "Pegel Dresden" during the German Flood (http://undine.bafg.de)

3.2.3 Data Visualisation

Mapping the spatial distribution of the tweet massages is the next step (Figure 3-18). It is obviously that flood related stream is rare. The phenome is known in the literature, Fuchs et al. (2013b) gathered 2429 tweet messages spanning the time period from November 25, 2012 to July 25, 2013. Not knowing, with which methods the datasets were gathered exactly, the numbers seem to be plausible.

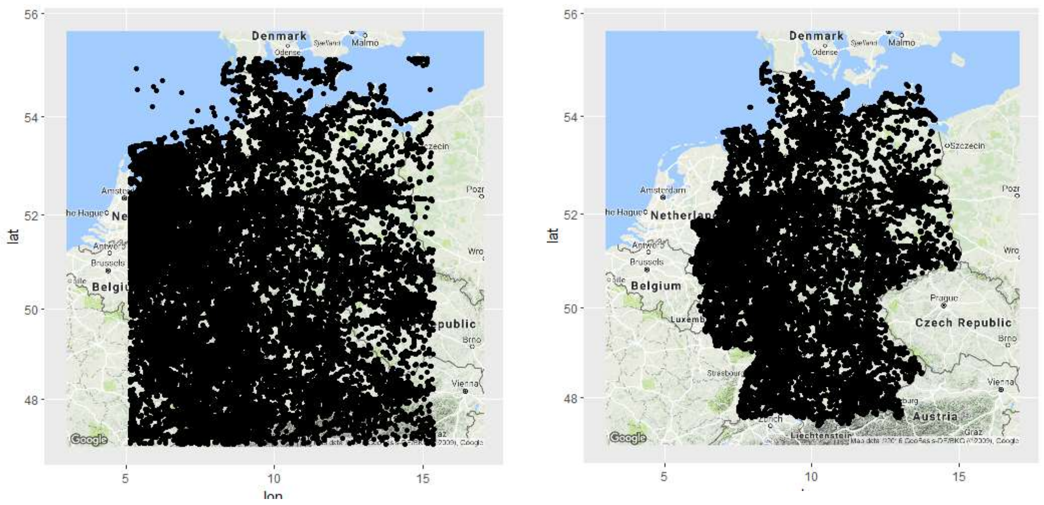
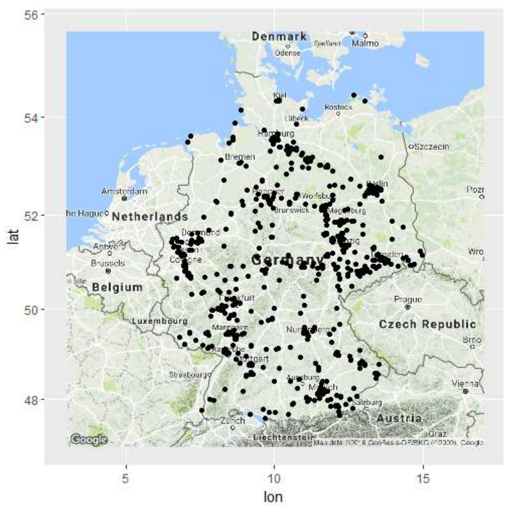


Figure 3-18: maps with dataframes elbe2013, elbe2013_de

Figure 3-19 is showing the drainage basin of the river Elbe, explaining that many tweets were gathered in this area.



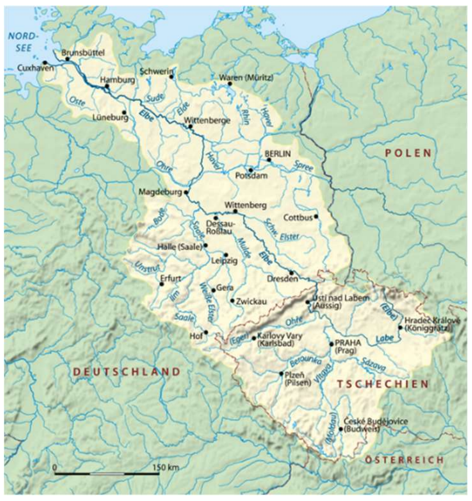


Figure 3-19: maps with dataframes elbe2013_de_hochwasser (flood related content) (left) and the drainage basin of the river Elbe

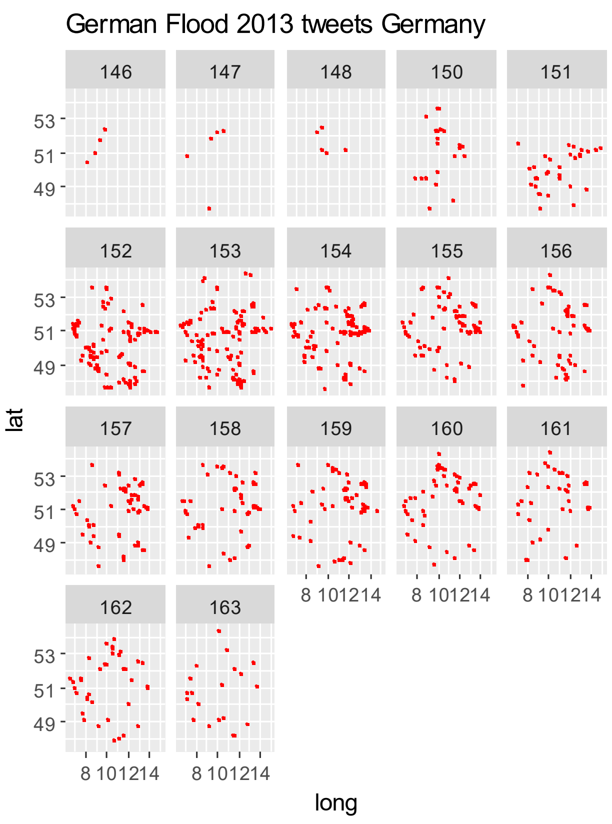


Figure 3-20: spatio – temporal distribution of the flood related tweets in Germany

Additionally, the spatio-temporal distribution is visualized. The time is represented by year days (the year day 153 is 02.06.2013) (Figure 3-20).

3.2.4 Explanatory data analysis

Heatmaps are a common method to find out the points of interest for further investigation. Visually analysing the plots in Figure 3-21 to area are present, situated along the river Elbe.

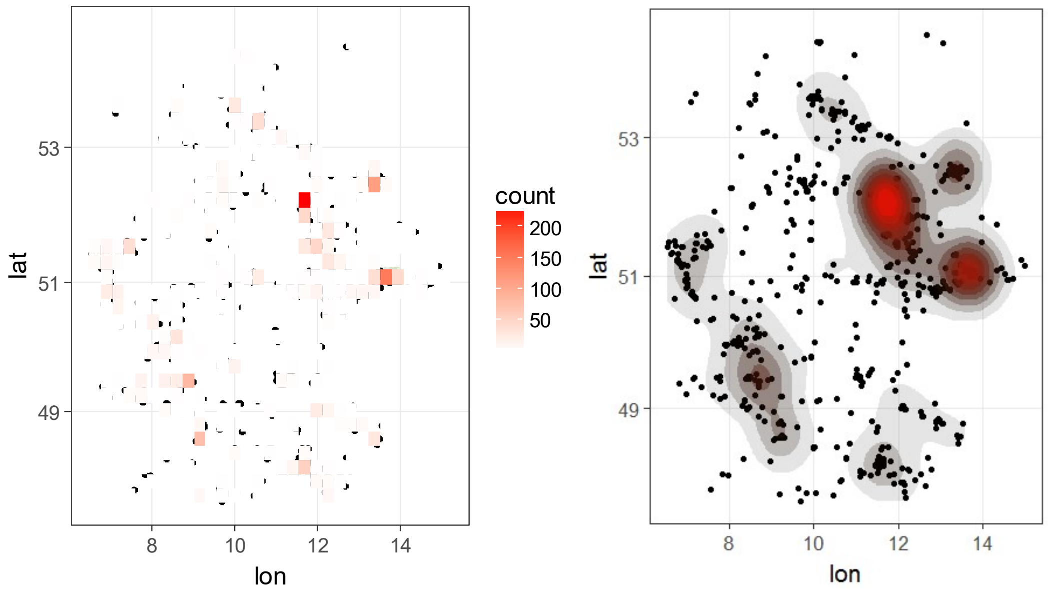


Figure 3-21: Heatmap of 2d bin counts (left) and Contours of a 2d density estimate (right)

The author is choosing the city of Dresden for further analysis, because the city was very affected from the extreme flood and with the spatial point pattern analysis in mind, Dresden is a worthwhile area for spatial modelling (Figure 3-22).

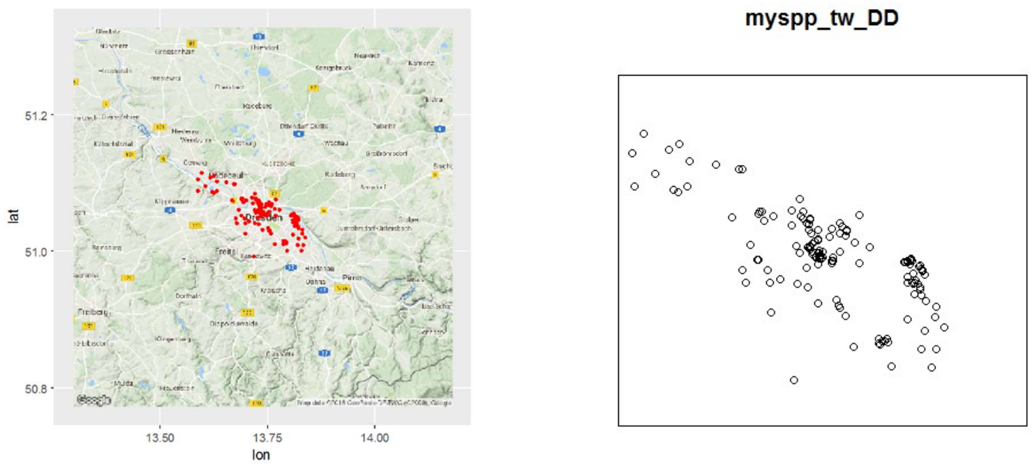


Figure 3-22: the dataset of Dresden and its implementation in spatstat

Due to the problem of measuring distances, the latitude and longitude coordinates has to be changed in a coordinate system in which the x and y coordinates are measured in the same distance units, because the R-package spatstat as the chosen tool for analysing spatial point pattern is using Euclidean distance calculations and these are inappropriate for latitude and longitude coordinates. They would be not more than a reasonable approximation for points scattered over a few dozen kilometres but not for a point pattern scattered over a continent. In this context, surly Dresden is a border case.

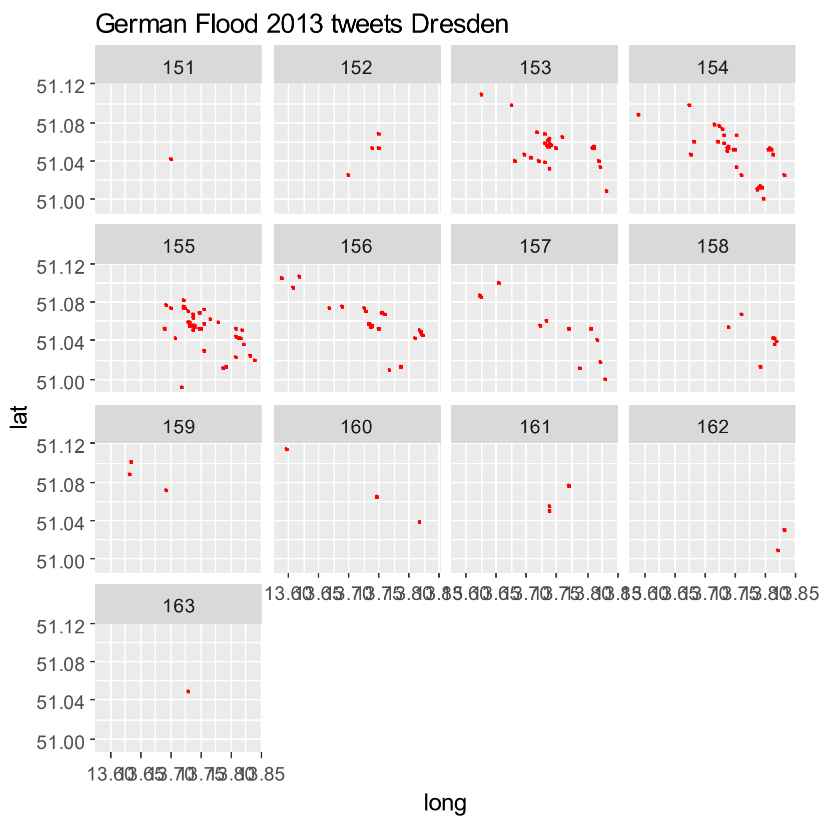


Figure 3-23: spatio – temporal distribution of the flood related tweets in Dresden

A visualisation with a facet-wrap plot gives insight about the spatio- temporal distribution of the data (Figure 3-23).

3.2.4.1 Kernel smoothing

The next step of the explanatory analysis is Kernel smoothing to determine the spatial distribution of the intensity.

Guessing the right sigma value for the standard variation of the Gaussian smoothing kernel is something by trial and error. Generally speaking if h is too small the estimate is too noisy, while if h is too high the estimate may miss crucial elements of the point pattern, so called over smoothing (Scott, 2004).

In spatstat the functions bw.diggle, bw.ppl, and bw.scott are offered to estimate the bandwidth according to difference methods (Bivand et al., 2013; Diggle, 1985; Scott, 2004).

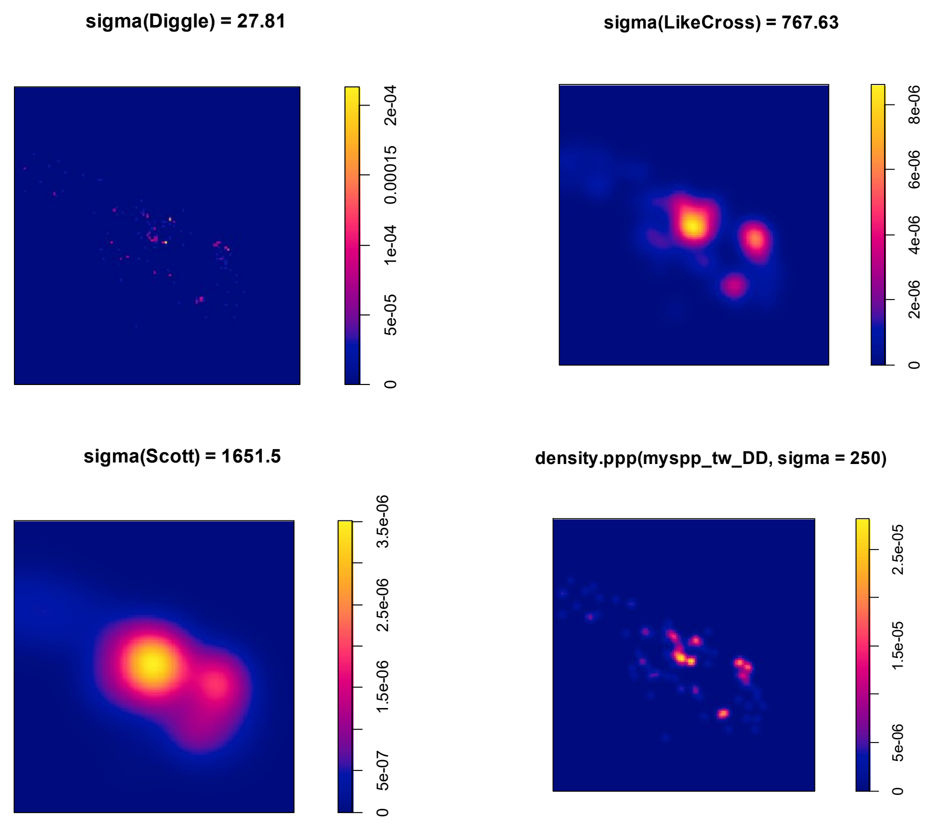


Figure 3-24: 4 sigmas for the standard variation of the Gaussian smoothing kernel

This computation was made with 1 manually chosen value. In the authors view sigma = 250 gives a good impression of the scenario and doesn't smooth too much. The clustering is visibly in the urban space dispersed with different small clusters (Figure 3-24, Figure 3-26).

Quadrat counting is also an expedient method to get a quick overview about the distribution dividing the study region in different rectangles of equal size, counting the number of each rectangle (Figure 3-25).

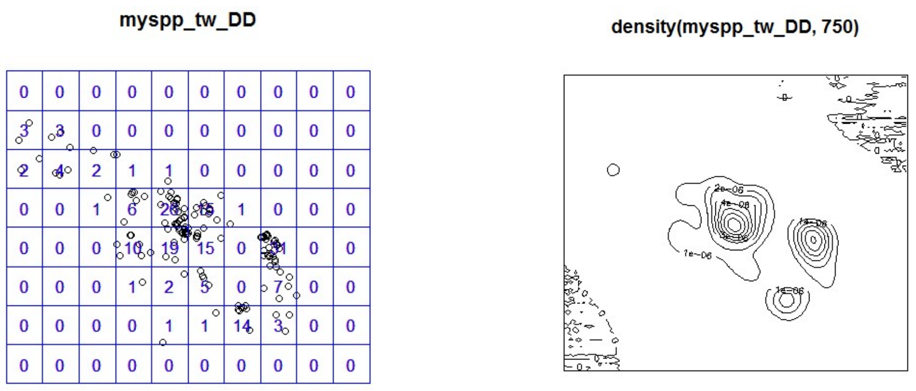


Figure 3-25: Quadrat counting and contour density plot

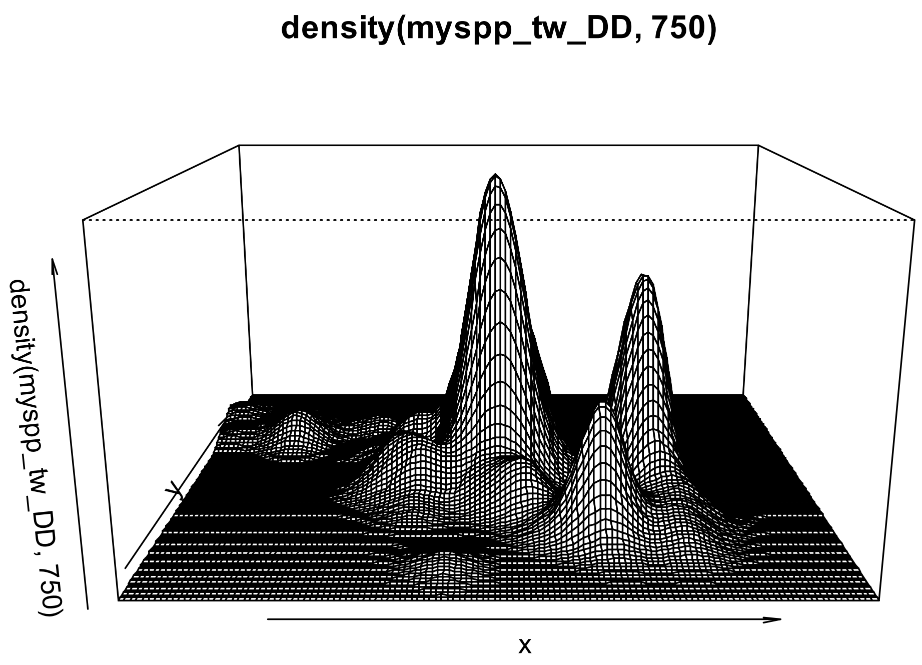


Figure 3-26: 3D- contour density plot

3.2.4.2 Testing for Complete Spatial Randomness (CSR)

In literature, the homogeneous Poisson process (CSR) is taken as the null model for a point pattern. Evidence against the CSR is the main task in analysing a point pattern (Cressie and Read, 1984). The method used is the Pearson χ² goodness-of-fit test. In this case inspecting the p-value is rejecting the null hypothesis. A small p-value suggests that this data set was not generated under CSR (Figure 3-27).

The main disadvantage of the Quadrat counting test for CSR is the lack of information – the alternative hypothesis is just the negation of the null hypothesis. The alternative is that the process is simply not a homogeneous Poisson point process. The "bad" residuals giving a hint in this direction. The low p-value declares it.

```
> quadrat.test(myspp_tw_DD, nx = 5, ny = 4)

            Chi-squared test of CSR using quadrat counts
            Pearson X2 statistic

data:  myspp_tw_DD
X2 = 455.43, df = 19, p-value < 2.2e-16
alternative hypothesis: two.sided
```

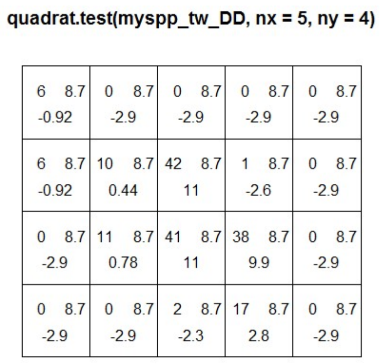


Figure 3-27: Quadrat test for myspp_tw_DD

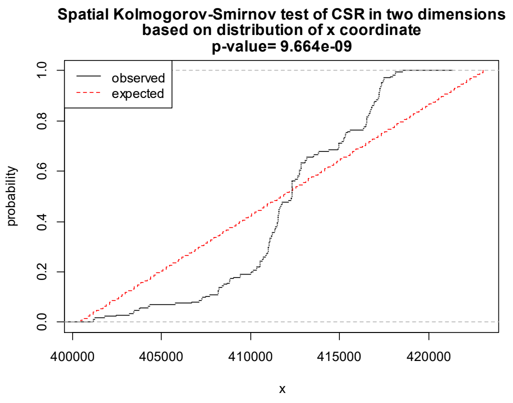
3.2.4.3 Kolmogorov – Smirnow test of CSR

A strong alternative testing of CSR is the Kolmogorov – Smirnow test comparing the observed distributions with distributions of the values of a function T e.g. “x” or “y”. The empirical distribution is compared to the predicted distribution under CSR. Again, a small p-value suggests that the empirical distribution was not generated under CSR (Figure 3-28).

```
> cdf.test(myspp_tw_DD, "x")

        Spatial Kolmogorov-Smirnov test of CSR in two
dimensions

data:  covariate ‘x’ evaluated at points of ‘myspp_tw_DD’
    and transformed to uniform distribution under CSR
D = 0.23457, p-value = 9.663e-09
alternative hypothesis: two-sided
```



```
> cdf.test(myspp_tw_DD, "y")

        Spatial Kolmogorov-Smirnov test of CSR in two
dimensions

data:  covariate ‘y’ evaluated at points of ‘myspp_tw_DD’
    and transformed to uniform distribution under CSR
D = 0.29083, p-value = 3.293e-13
alternative hypothesis: two-sided
```

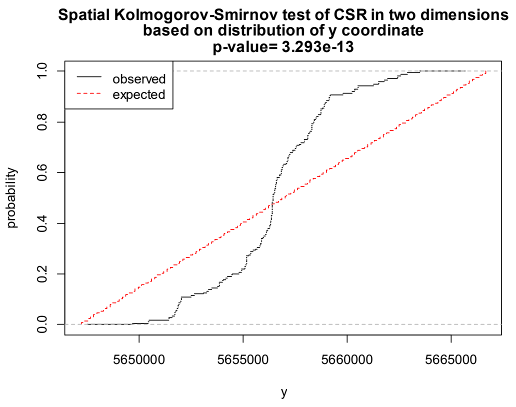


Figure 3-28: Results oft he Kolmogorov – Smirnow test of CSR (mySPP_tw_DD)

3.2.4.4 G-function and K-function

As described in section 2.2.1.3 the K-function belongs to the exploratory analysis of point patterns and is based on summary statistics. The Spatstat package will compute estimates of the summary functions (Ripley, 1977b)

- F (r), the empty space function
- G(r), the nearest neighbour distance distribution
- J(r), the function J= (1−G)/(1−F)
- K(r), the reduced second moment function (”Ripley’s K function”)
- g(r), the pair correlation function g (r) = [d/dr K(r)] / (2πr)

the corresponding spatstat library functions are:

```
Fest estimate of empty space function F
Gest estimate of nearest neighbour distribution functionG
Jest estimate of J -function
```

```
Kest estimate of Ripley's K -function
Allstats estimates of all four functions F, G, J, K
Pcf estimate of pair correlation function g
```

We first calculate the nearest neighbour distance distribution with the G-function.

```
plot(Gest(myspp_tw_DD))
```

The resulting plot (Figure 3-29) tells us, that 65% of the points have a neighbour within 200 m, an 95 % within 800 m. The blue line is the expectation from complete spatial randomness (CSR) the other lines is what we calculated from the data. This plot heavily indicates clustering – a greater proportion of tweet locations have nearest neighbours at each distance expected under CSR.

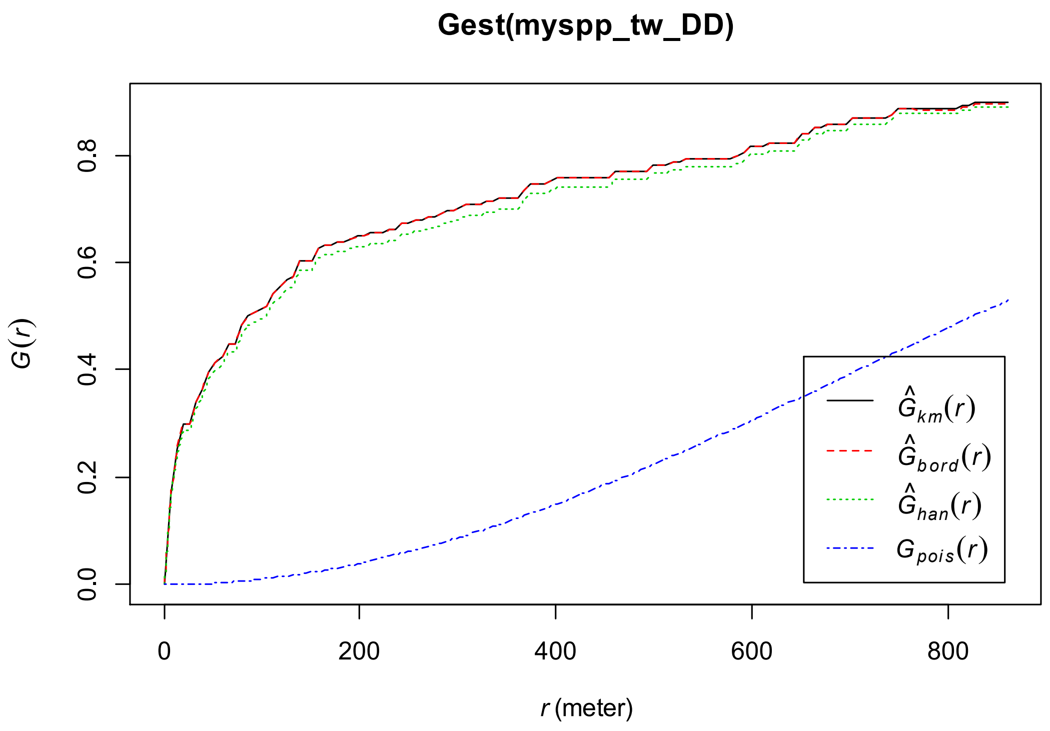


Figure 3-29:G-function (dataset myspp_tw_DD)

The weakness of the G-functions is that it only looks at nearest neighbours.

```
tw_DD.k <- envelope(myspp_tw_DD, Kest, nsim = 99, correction = "border")
plot(tw_DD.k)
```

The function envelope() calculates the K-function for our data and simulates CSR – calculating the K-function for each simulated pattern. The resulting plot shows the range of the values obtained via simulation as a grey envelope (Figure 3-30). Again, the observed K-function (K_{obs}) is much higher than the simulated, which implies clustering – because at all distances there are more point nearer to every point than expected under CSR.

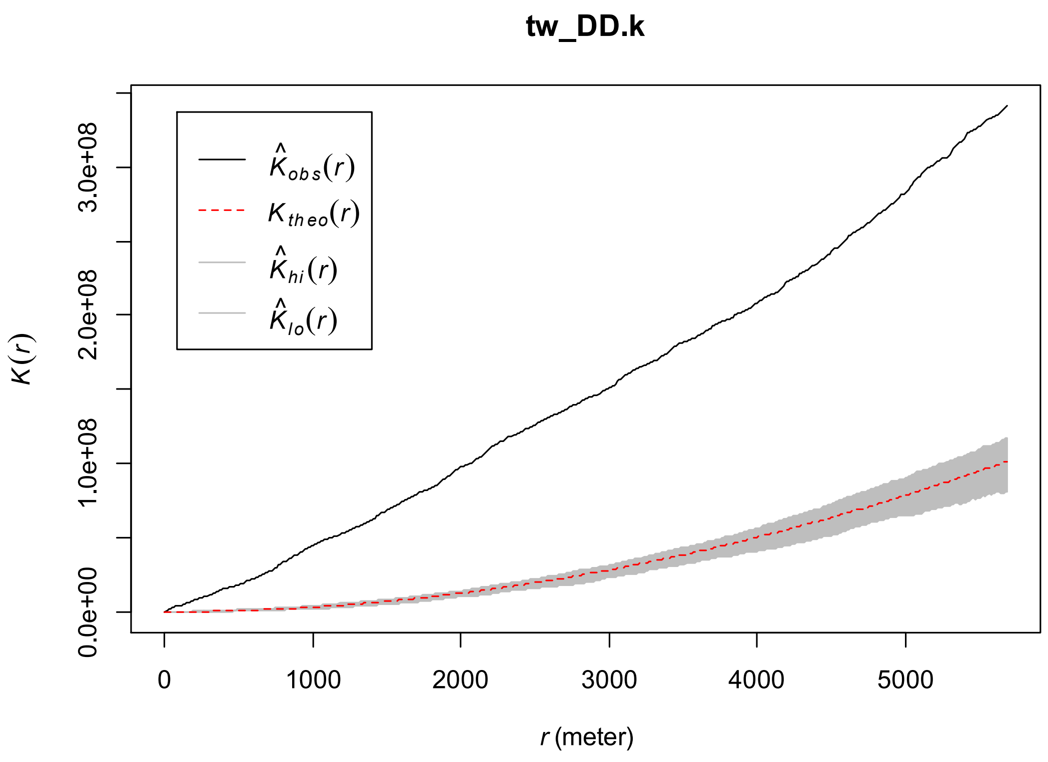


Figure 3-30: calculated and simulated K-function

Would the black line of our observed data would lies beneath the envelope, implying no significant difference from CSR. Generally, a pattern can be clustered and dispersed at different scales, depending on the distribution. All statements representing an homogeneous Poisson process.

Modell 1 with covariate for calculating the K-function

Considering an inhomogeneous Poisson process with a λ being not constant at all locations instead spatially varying. To figure this out, we use raster plot for the density of human settlements and transport infrastructure (Figure 3-31).

```
tw_DD.fit <- ppm(myspp_tw_DD, ~ Cov1, covariates = list(Cov1 = landuse))
plot(tw_DD.fit)
point.sim <- simulate(tw_DD.fit, 1)
plot(point.sim)
```

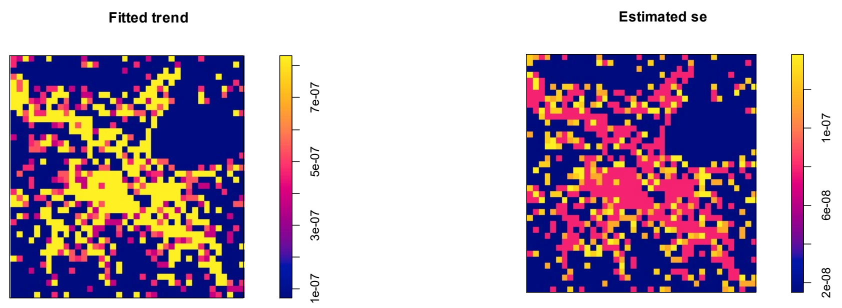


Figure 3-31: simulation tw_DD_fit: fitted trend and estimated se

```
plot(predict.ppm(tw_DD.fit, type = "lambda"), main = "Predicted Intensity (lambda)") #predicted in-
tensity
```

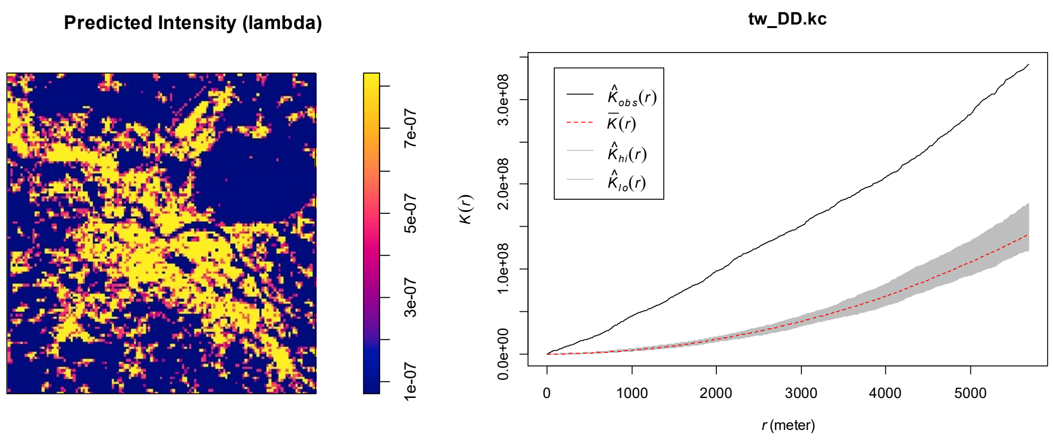


Figure 3-32: predicted intensity and calculated and simulated K-function for tw_DD.fit

```
tw_DD.kc<- envelope(tw_DD.fit, Kest, nsim = 99, correction = "border", verbose = F)
plot(tw_DD.kc)
```

The result shows significant clustering everywhere. This simulation can be viewed as a failed attempt to explain the observed distribution of tweets (Figure 3-32).

Modell 2 with covariate for calculating the K-function

For the next calculation a polynomial trend and the same covariate (Figure 3-33) are chosen.

```
tw_DD.fitcov2 <- ppm(myspp_tw_DD ~polynom(x,y,3) + Z, covariates=list(Z=landuse))
point.sim2 <- simulate(tw_DD.fitcov2, 1)
plot(point.sim2)
plot(predict.ppm(tw_DD.fitcov2, type = "lambda"), main = "Predicted Intensity (lambda)") #predicted intensity
tw_DD.kc2<- envelope(tw_DD.fitcov2, Kest, nsim = 99, correction = "border", verbose = F)
plot(tw_DD.kc2)
diagnose.ppm(tw_DD.fitcov2, type="pearson")
p2 <- predict.ppm(tw_DD.fitcov2)
plot(p2)
```

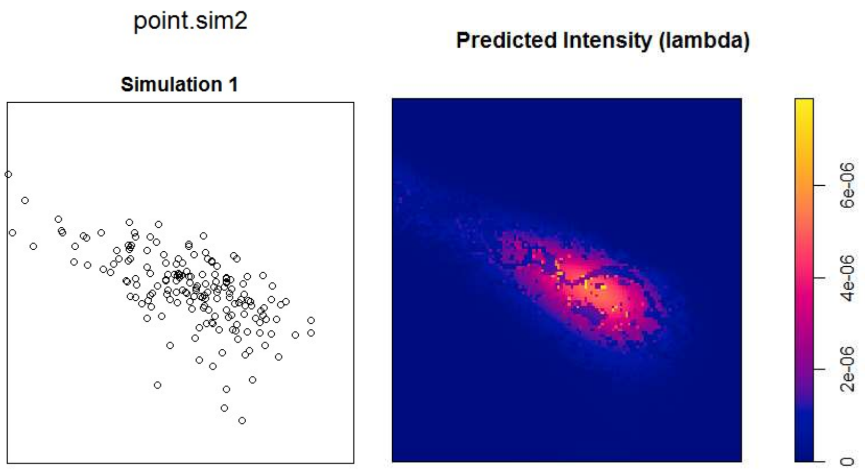


Figure 3-33: simulated points and predicted intensity (model tw_DD.fitcov2)

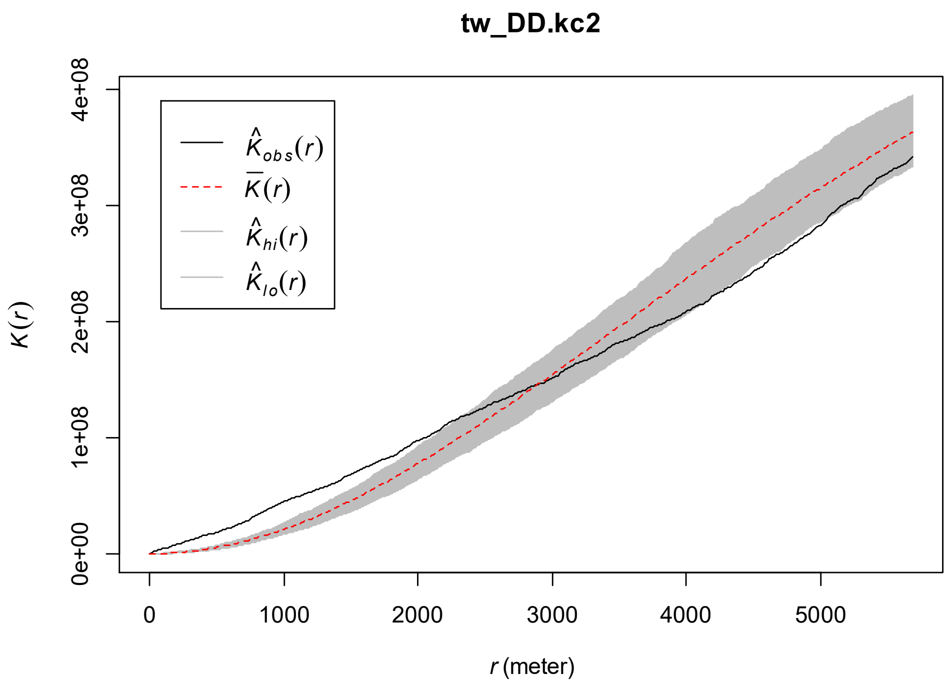


Figure 3-34: observed K-function and simulated envelopes for (model tw_DD_fitcov2)

The resulting lines (Figure 3-34) showing that a pattern can be clustered at some scales and be dispersed at others. From a radius up to 2500 m significant clustering is detected. Then the observed K-function enters the grey envelope and leaves it again between 4100 m and 5000 m showing light dispersion.

3.2.5 Model analysis of the Twitter feed in Dresden

As mentioned above, the spatstat function ppm fits a point process model to an observed point pattern. The model may include spatial trend, interpoint interaction, and dependence on covariates.

```
fit <- ppm(myspp_tw_DD, ~polynom(x,y,3), Poisson())
```

After the modelling process the residual analysis and checking the Q-Q-Plot are following:

Residual analysis

Figure 3-35 shows a standard presentation of the diagnostic plots for spatial trend. The first quadrant is the mark plot. The fourth quadrant is a contour and image plot. He shows the smoothed residual field, rendered always the way that s(u) = 0 is plotted in the same colour for a better interpretation. The lurking variables plots are placed in the second (y coordinate, rotated 90 degrees anticlockwise) and third (x coordinate, aligned with x-axis from mark plot) quadrant. This combination of plot gives the analyst a practicable view for detecting spatial trend when they exist.

The left panel in indicates that the correct model is a tolerably good fit, although it (correctly) suggests the trend is a little underestimated. The lurking variable plots with respect to the x- and y- coordinate variables shows acceptable deviation from the 2б- limits, indicating that the model does account for a variation in intensity with respect to these variables. Maybe the point in max. south could be declared as an outliner.

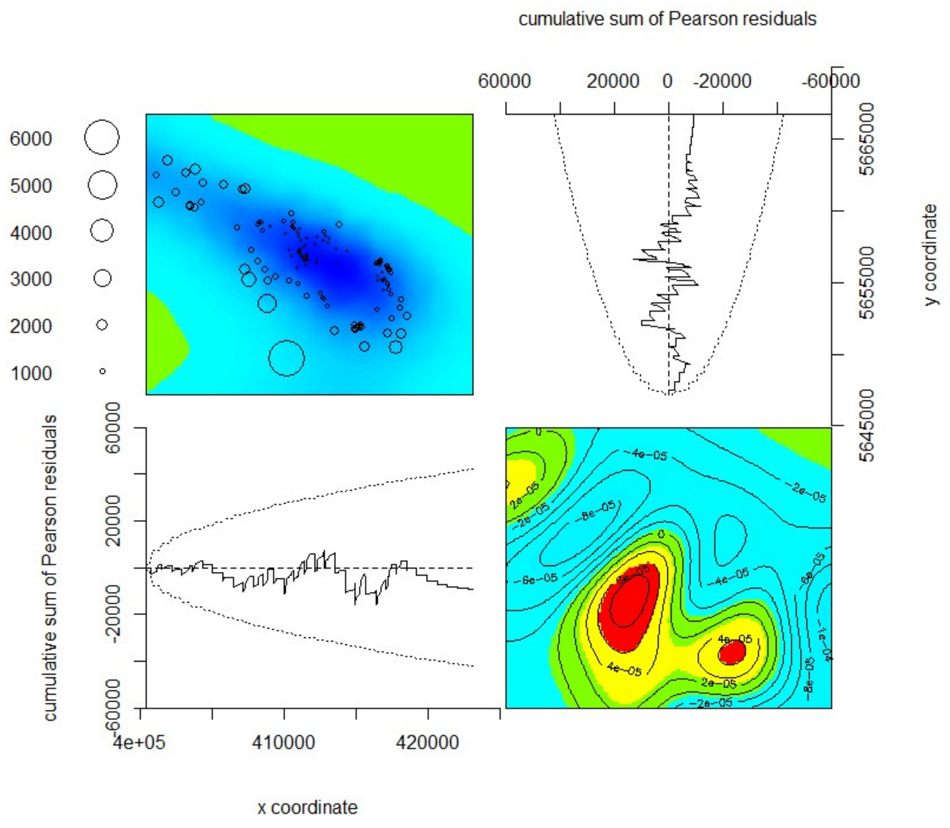


Figure 3-35: residual diagnostics plot Lurking variable plot from model fit

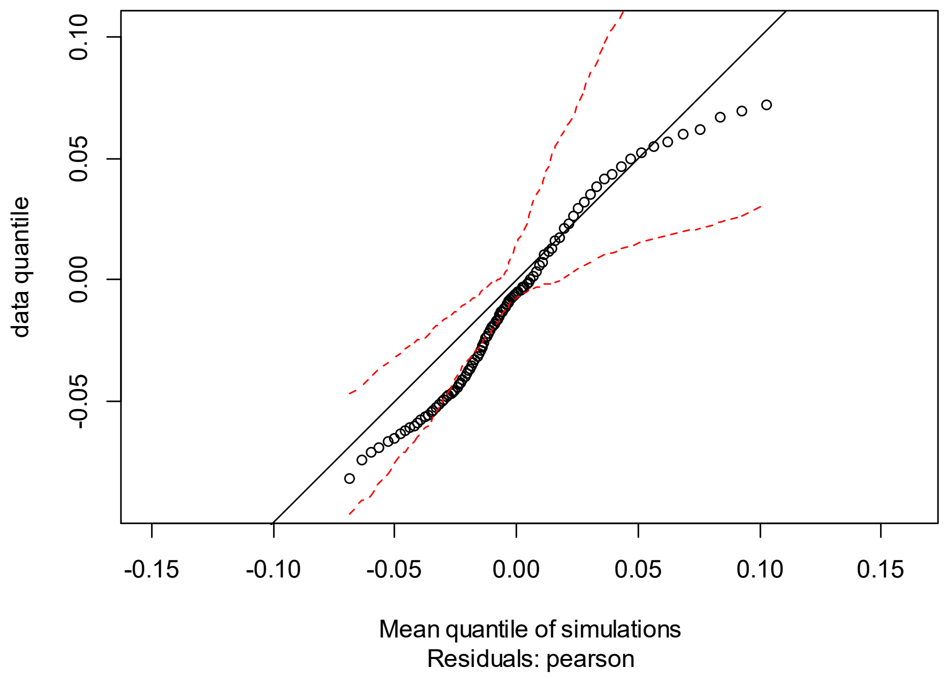


Figure 3-36: Q-Q-Plot with Pearson residuals

The Q-Q-plot could be used to validate the interpoint interaction component of a model. Under the analogy between point processes and generalized linear models, interpoint interaction in a point process has an analogue distribution of residuals like a generalized linear model. An appropriate tool for assessing the distributional assumptions in a generalized linear model is the summary of the empirical distribution of the residuals, called the Q–Q plot.

Therefore, the empirical quantiles of the smoothed residuals (s(u), Figure 3-35) are compared with the corresponding expected empirical quantiles under the fitted model (fit), which are estimated by the Monte Carlo method.

First many simulated realisations from the fitted model have to be calculated, and then for every simulated dataset the same model has to be fit. To each simulated dataset, the same model is fitted with similar calculations, which will be compared (Baddeley et al., 2005b).

Figure 3-36 shows obviously a Q-Q-plot of a model in correct form, suggesting an acceptable agreement between the model and the data. The data seems to have a higher variability than the smoothed residual field for simulations from the fitted Poisson model.

Covariates

An important fact is getting to know, if a covariate is affecting the intensity of the observed points. Spatstat is offering the possibility to define covariates out of from raster data.

It shall be investigated, if people prefer to send tweet messages in through densely populated areas or in the open countryside. Therefore the basis for the covariate is a WMS from ("IÖR-Monitor: Rasterkarten," n.d.) for the density of the human settlements and the transport infrastructure.

Knowing the influence of the areas where tweets massages are generated could be useful for the expectations of information gathering for crisis managers.

Modelling a plausible distribution from tweet massages, which are geolocated after gathering with information from e.g. user's self–reported location with the Google Geocoder or Yahoo! Placemaker API, could be also interesting, because these a posterior georeferenced tweet messages are always noisy.

Overall the covariate intensity can suggest more detailed density models and mainly comes with the second stage of the spatial point pattern analysis. First is always investigating the general structure of the pattern (clustered, regular or random), second is to get insights why it has this structure.

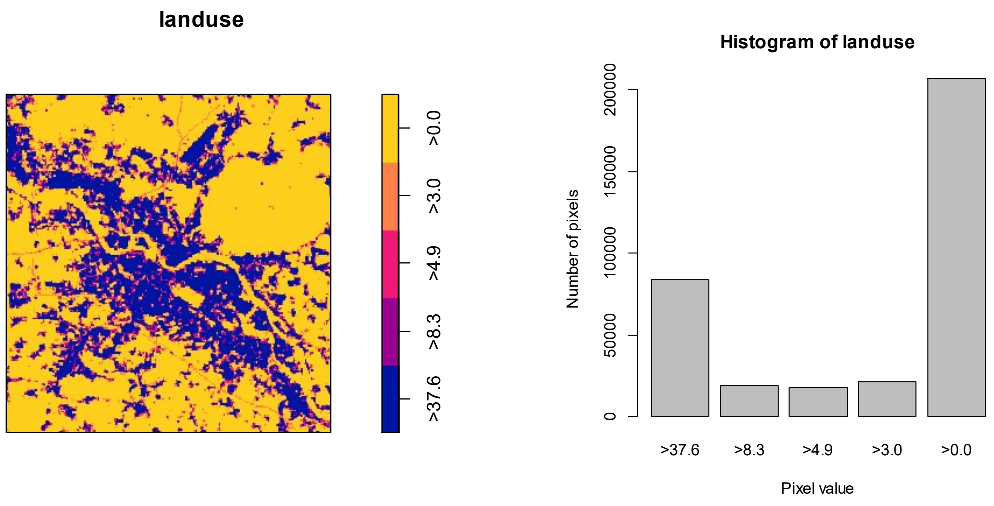


Figure 3-37: rastermap of the land use for human settlements and transport infrastructure

Fitting Model with covariate ~landuse

The next model (Figure 3-38) implies the use of a covariate (Figure 3-37). It will not work properly because of the missing influence of the x and y variable, but will show us a clear presentation of how the influence of the covariate is working.

```
fitcov0 <- ppm(myspp_tw_DD, ~landuse, covariates = list(landuse))
fitcov0
p_fitcov0 <- predict.ppm(fitcov0)
plot(p_fitcov0)
```

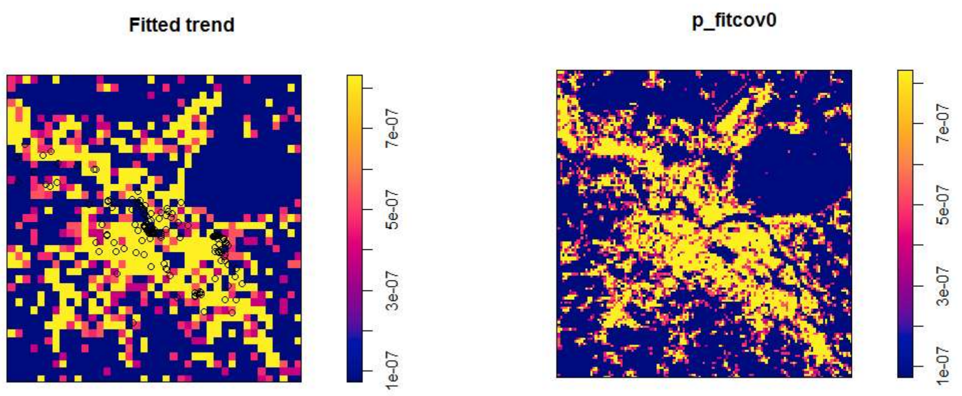


Figure 3-38: model fitcov0 with fitted trend (Left) and prediction (Right)

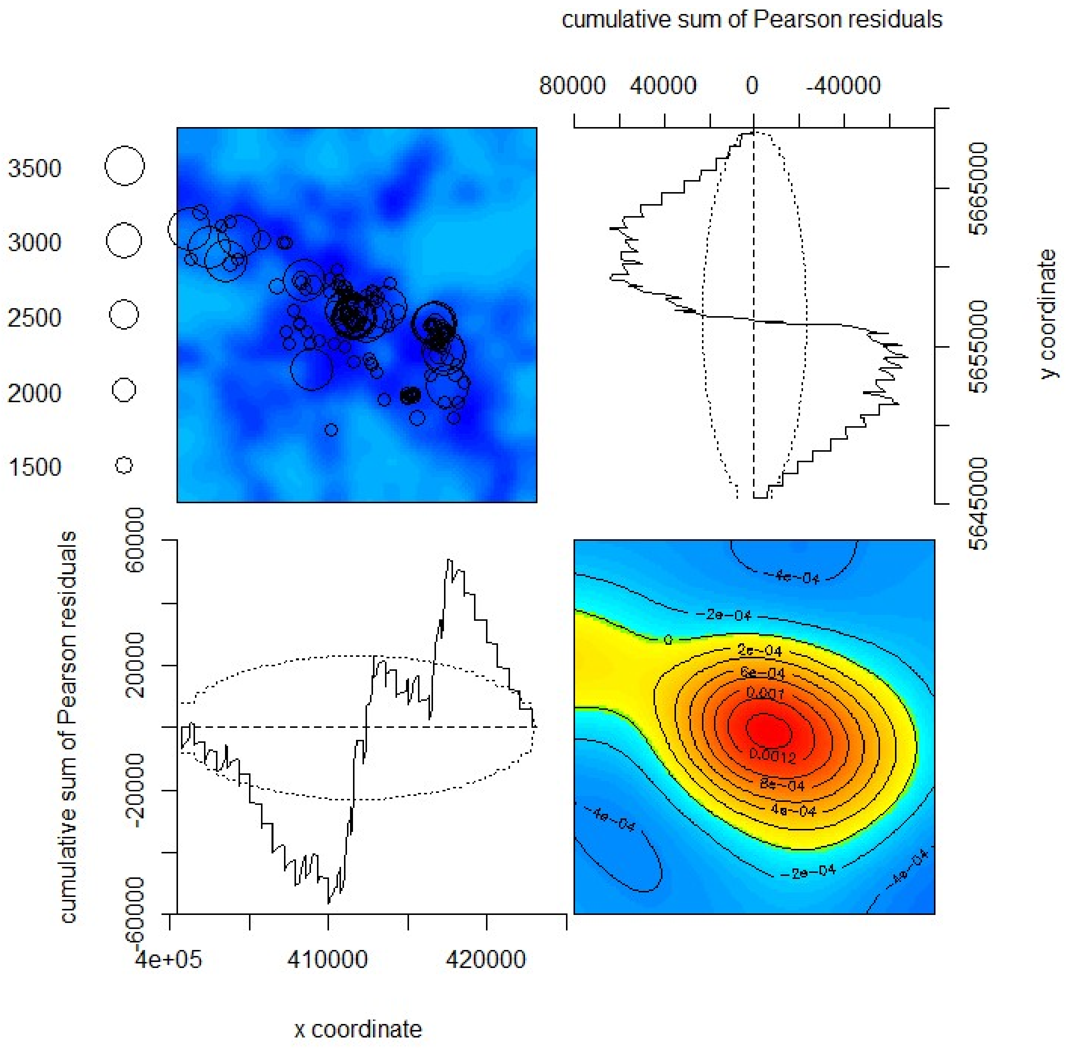


Figure 3-39: residual diagnostics plot Lurking variable plot from model fitcov0

Even the lurking variable plots (Figure 3-39) are not looking good, both lurking variable plot for the x and y coordinate showing a distinctive and persistent dip, which strongly indicates that the model is inappropriate. The mark plots

Fitting model with covariate and ~polynom(x,y,3) trend

The second model (Figure 3-40) (it’s the example from section 3.2.4.4) with covariate and it fits very well. In contrast to the model above it uses a polynomial trend (~polynomal(x,y,3) in combination with the covariate.

```
fitcov2 <- ppm(myspp_tw_DD ~polynom(x,y,3) + Z, covariates=list(Z=landuse))
diagnose.ppm(fitcov2, type="pearson")
p2 <- predict.ppm(fitcov2)
plot(p2)
tw_DD.fitcov2 <- ppm(myspp_tw_DD ~polynom(x,y,3) + Z, covari-ates=list(Z=landuse))
point.sim2 <- simulate(tw_DD.fitcov2, 1)
plot(point.sim2)
plot(predict.ppm(tw_DD.fitcov2, type = "lambda"), main = "Predicted Intensity (lambda)") #predicted intensity
tw_DD.kc2<- envelope(tw_DD.fitcov2, Kest, nsim = 99, correction = "border", verbose = F)
```

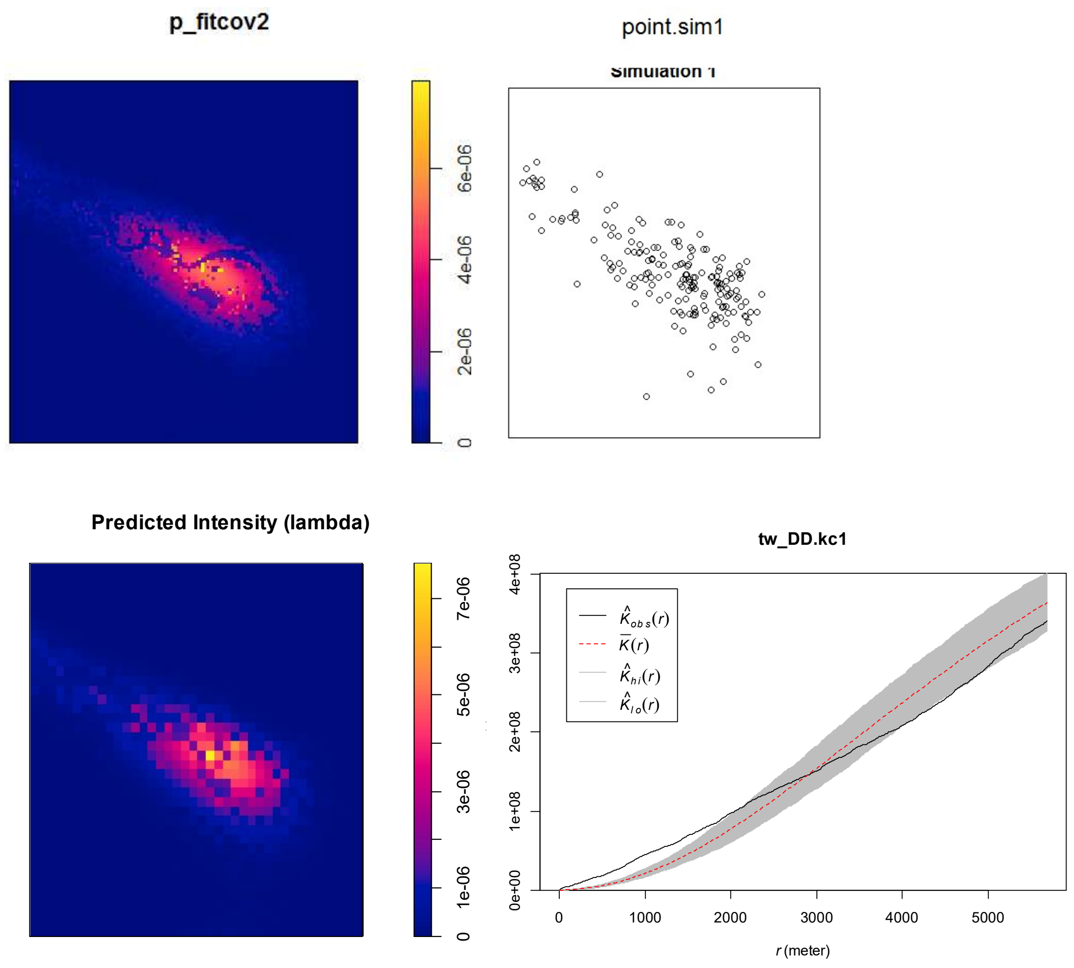


Figure 3-40: fitted, simulated and predicted intensity of model fitcov2 with K_{obs}

The function envelope() calculates the K-function again and simulates CSR – calculating the K-function for each simulated pattern. The resulting plot shows the range of the values obtained via simulation as a grey envelope as introduced above. The simulated point pattern is realisation from the fitted model and looks reasonable to the predicted intensity.

Again, the observed K-function ($K_{obs.}$) shows the relation of the observed distribution to the simulated. The resulting lines showing that the pattern is significant clustered until 2500 m, and then leaves the envelope between 4100 m and 5000m showing light dispersion.

The residual plot (Figure 3-41) shows a trend in the south (red area) which is obviously unaccounted by the model. In these cases the matter of choice is to collect more data with possibly newer explanatory factors or the model assumption has to be revisited.

The Lurking variable plots (Figure 3-41) look good for the x and the y variable, the continuous line of the cumulative sum of the residuals doesn't leave the dot plotted area of the standard derivation.

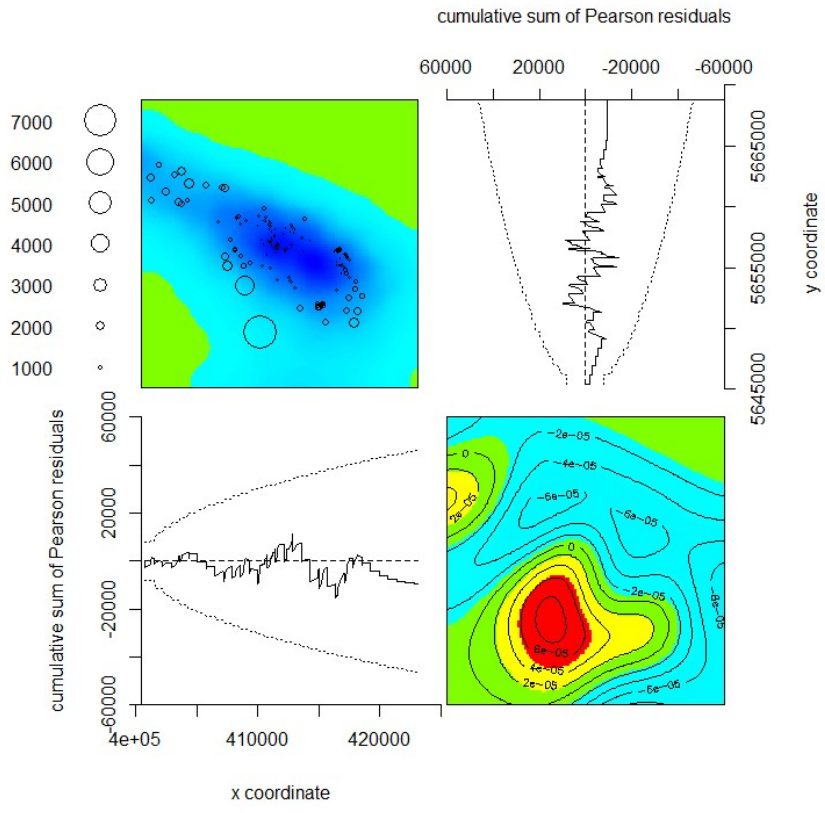


Figure 3-41: residual diagnostics plot Lurking variable plot from model fitcov2

The Q-Q-plot (Figure 3-42) shows an acceptable relation between the distributions of the quantiles from booth statistical fitted and simulated data.

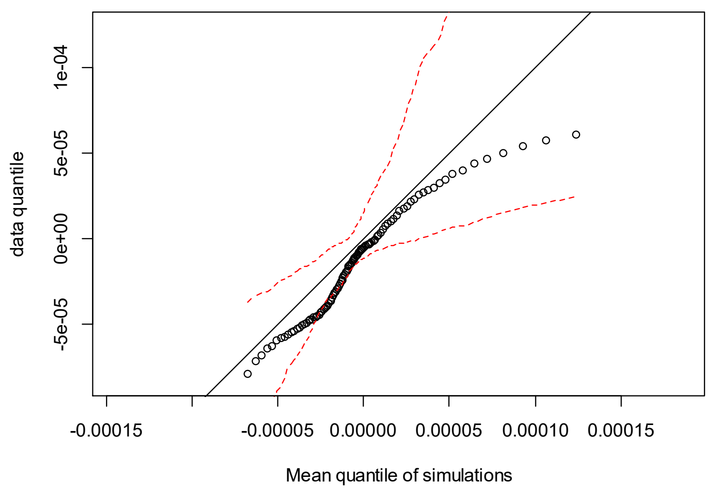


Figure 3-42: residual diagnostics plot Lurking variable plot from model fitcov2

3.2.6 Model analysis of the Twitter feed in Saxony

The distribution of the twitter messages in Saxony is shown in Figure 3-43. The Heatmap indicates a Hotspot in Dresden and some clustering in Leipzig.

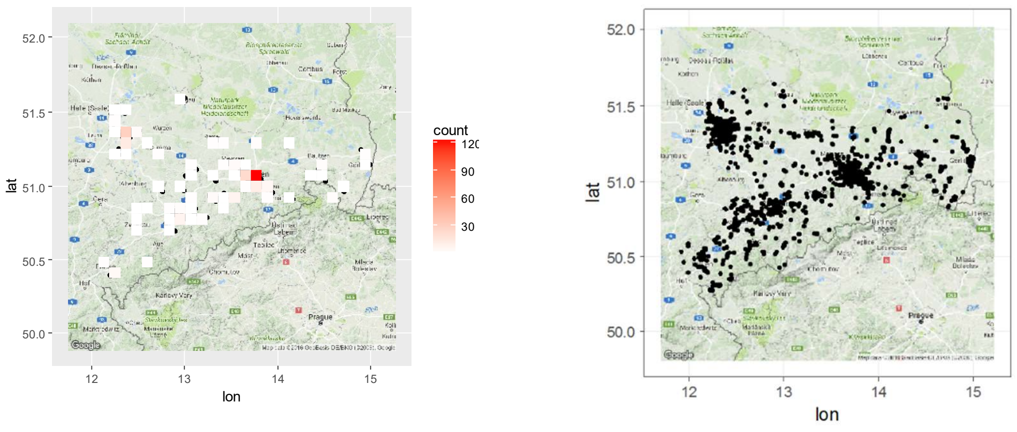


Figure 3-43: Heatmap of 2d bin counts (Right) and distribution map (Left)

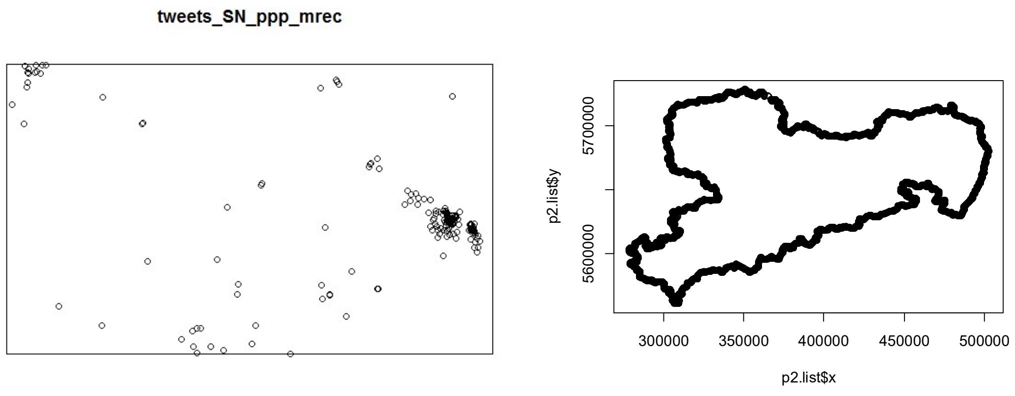


Figure 3-44: ppp – object of the twitter point pattern in Saxony (Left), boundary of Saxony (Right)

The ppp-object of the spatstat point pattern is shown in Figure 3 44 and is describing a rectangle from Leipzig in the upper left and Dresden in the middle right. The ppp-object concludes out of 214 points with unequal coordinates. The first idea to choose the boundary of Saxony as the model – windows was discard because of numerical problems with the integration of the raster picture from the covariate. Maybe it had also caused problems with the edge effect handling.

```
> tweets_SN_ppp_mrec
Planar point pattern: 214 points
window: rectangle = [309907, 421429] x [5628824, 5690340] meter
> plot(tweets_SN_ppp_mrec)
> tweets_SN_ppp_mrec
Planar point pattern: 214 points
window: rectangle = [309907, 421429] x [5628824, 5690340] meter
Warning message:
Semi-Transparenz ist für dieses Gerät nicht unterstützt; nur eine Meldung pro Seite
> mydataSNr
factor-valued pixel image
factor levels:
[1] ">37.6" ">8.3"  ">4.9"  ">3.0"  ">0.0"  "NA"
232 x 421 pixel array (ny, nx)
enclosing rectangle: [309960, 421350] x [5628800, 5690200] units
```

The raster map from the covariate (Figure 3-45) indicating the areas of land use for human settlements and the transport infrastructure like town and roads implicating a strong influence of the covariate in the areas with a high density of them. This should be visible also in the simulated point pattern (Figure 3-45).

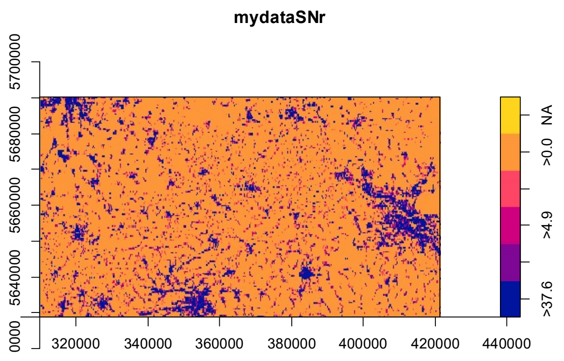


Figure 3-45: covariate land use

As shown in the figure the simulated point pattern has a high dispersion in the area of the cities. This is a hint that the chosen model failed to attempt the observed distribution of the ppp-object (Figure 3-46).

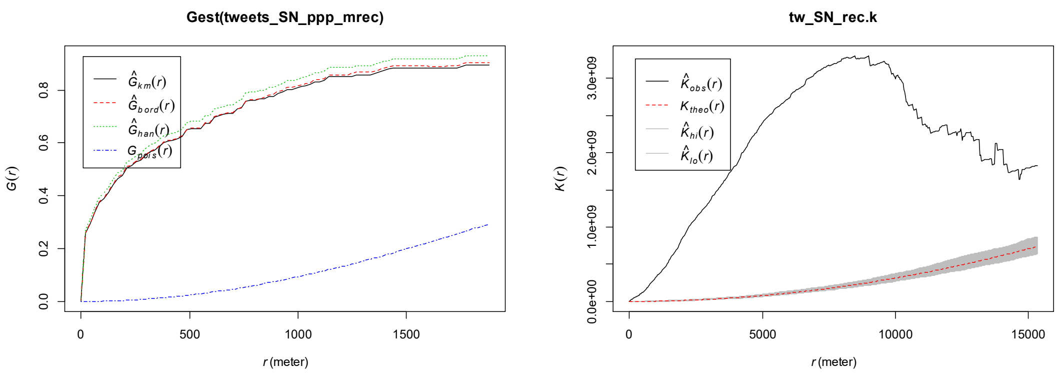


Figure 3-46: The estimated G-function and the computed K-function with simulated CSR

The plot of the computed nearest neighbours of the G-function indicates that there is a clustering and the distribution is far away from CSR.

The calculated K-function of the observed data is again higher as the simulated K-functions under complete spatial randomness plotted in the grey envelope suggesting clustering at all distances (Figure 3-46).

```
tw_SN.fitcov2 <- ppm(tweets_SN_ppp_mrec ~polynom(x,y,3) + Z, covariates=list(Z=my-
dataSNr))
plot(tw_SN.fitcov2)
diagnose.ppm(tw_SN.fitcov2, type="pearson")

point.SN.sim2 <- simulate(tw_SN.fitcov2, 1)
plot(point.SN.sim2)

plot(predict.ppm(tw_SN.fitcov2, type = "lambda"), main = "Predicted Intensity
(lambda)") #predicted intensity
tw_SN.kc1<- envelope(tw_SN.fitcov2, Kest, nsim = 99, correction = "border", verbose =
F)
plot(tw_SN.kc1)
```

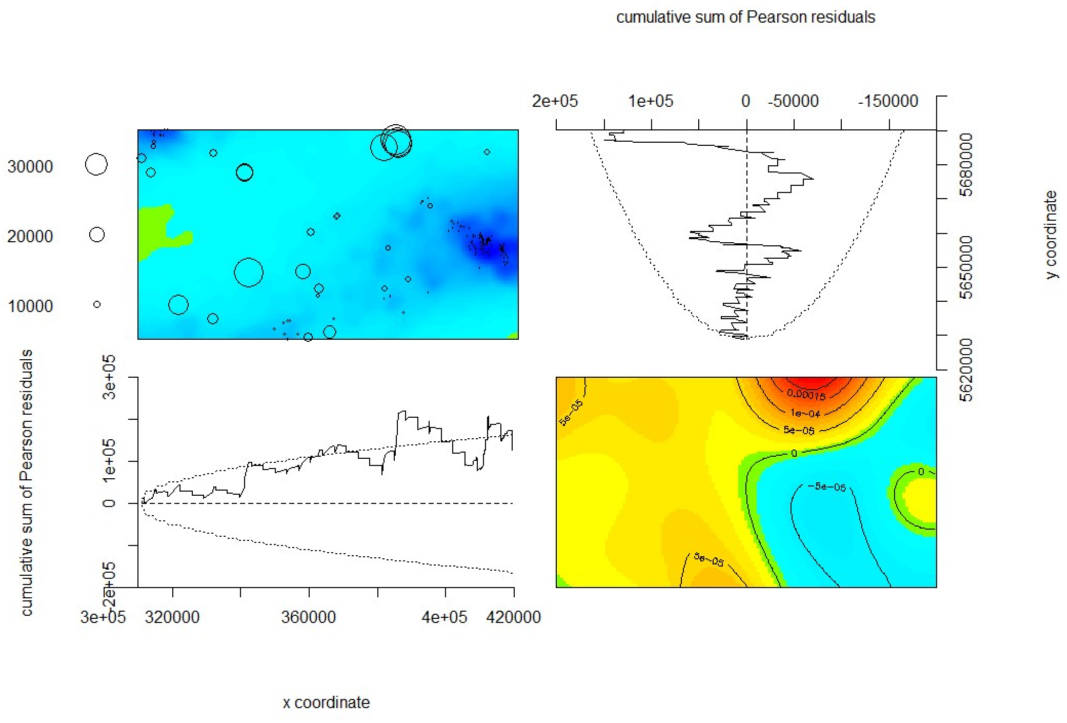


Figure 3-47: the residual plot and the lurking variables

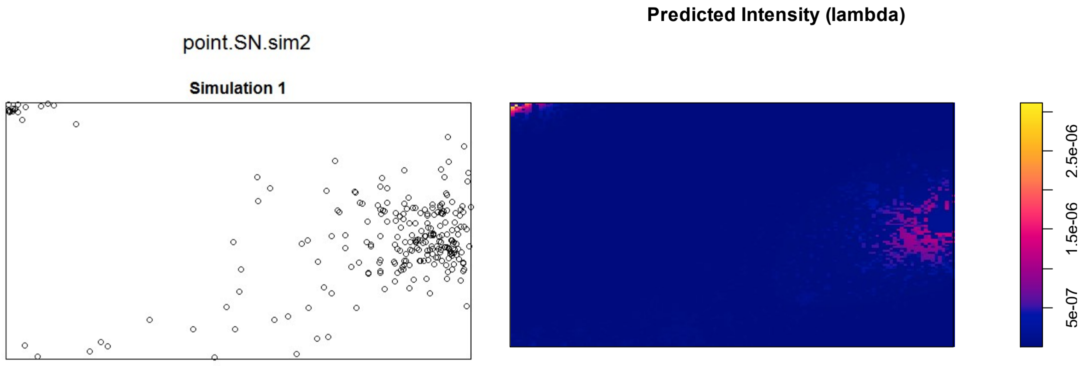


Figure 3-48: the fitted distribution (Left) and the predicted intensity (Right)

The residual plot (Figure 3-47) shows a trend in the north (red area) which is caused by a few points at the northern edge of the model. The Lurking variable plots shows this deviation perfectly for the right end of the y-axis. The observed K-function is still clustered up to a radius of 11.000 m and the crosses the envelope ending in light dispersion at 14.000 m (Figure 3-49). The simulated point pattern (Figure 3-48) look reasonable.

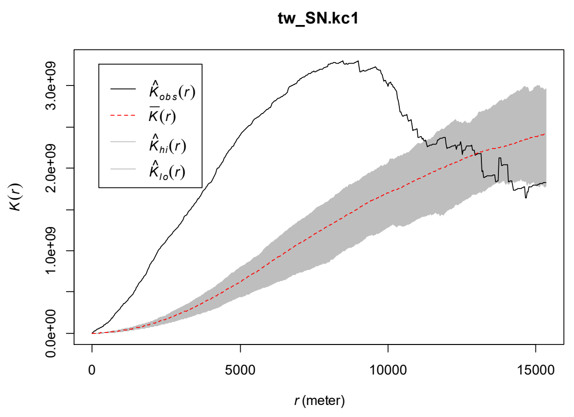
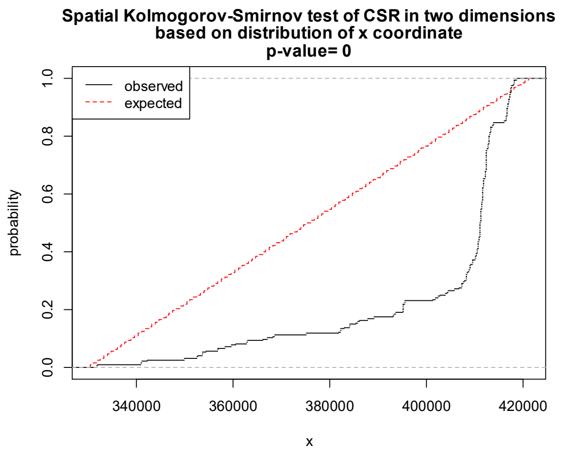


Figure 3-49: calculated and simulated K-function (model tw_SN_fitcov2)

```
> cdf.test(tweets_SN_ppp_mrec, "x")

        Spatial Kolmogorov-Smirnov test of CSR
in two dimensions

data:  covariate 'x' evaluated at points of
'tweets_SN_ppp_mrec'
   and transformed to uniform distribution under
CSR
D = 0.56827, p-value < 2.2e-16
alternative hypothesis: two-sided
```



```
> cdf.test(tweets_SN_ppp_mrec, "y")

        Spatial Kolmogorov-Smirnov test of CSR
in two dimensions

data:  covariate 'y' evaluated at points of
'tweets_SN_ppp_mrec'
   and transformed to uniform distribution under
CSR
D = 0.3547, p-value < 2.2e-16
alternative hypothesis: two-sided
```

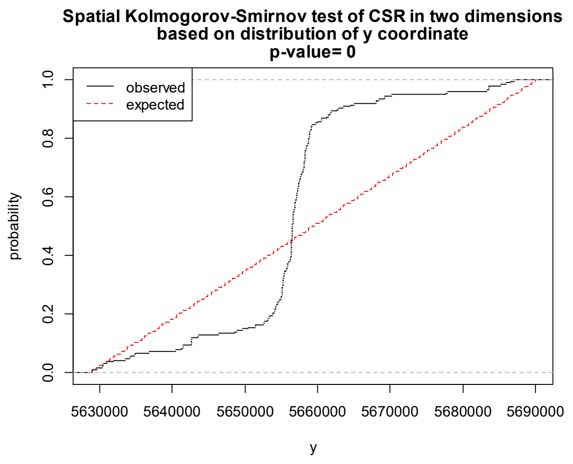


Figure 3-50: Results oft he Kolmogorov – Smirnow test of CSR (mySPP_tw_DD)

The Kolmogorov – Smirnow test with a small p-value suggests that the empirical distribution was not generated under CSR (Figure 3-50).

3.2.7 Model analysis of the Flickr feed in Dresden

Flickr is an image hosting and video hosting website. It is a popular website for users who want to share and embed their personal photographs. It is also a source for photo researchers and it is used by bloggers to host images that they embed in their blogs and other social media.

The dataset was collected with R using the Flickr – API between the 27.05.2013 and the 15.06.2013 like corresponding twitter dataset (Figure 3-51).

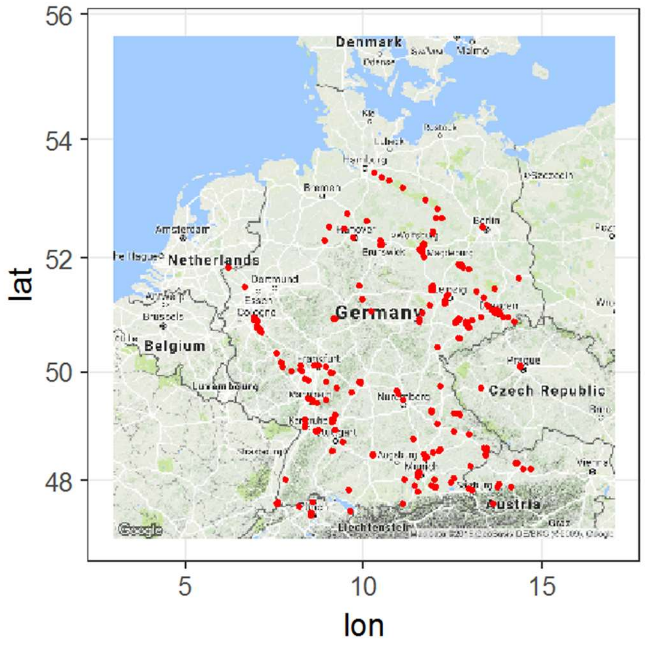


Figure 3-51: the distribution of the flickr dataset in Germany

Figure Figure 3-52 shows a cumulation of the points neat the river Elbe, but also near the river Rhine.

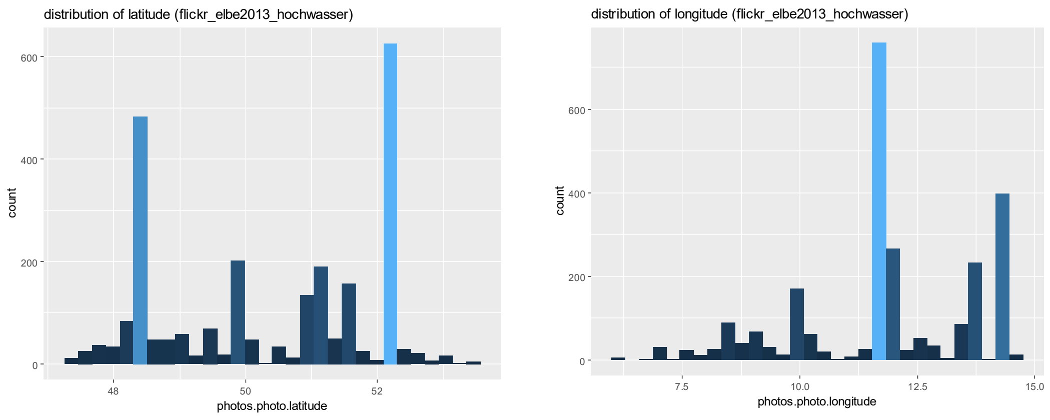


Figure 3-52: the lat/lon distribution of the flickr dataset of Germany

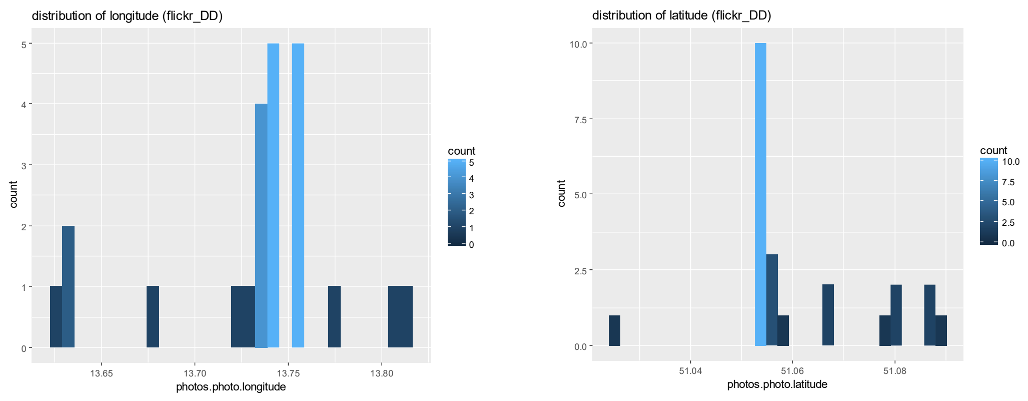


Figure 3-53: the lat/lon distribution of the flickr dataset in Dresden

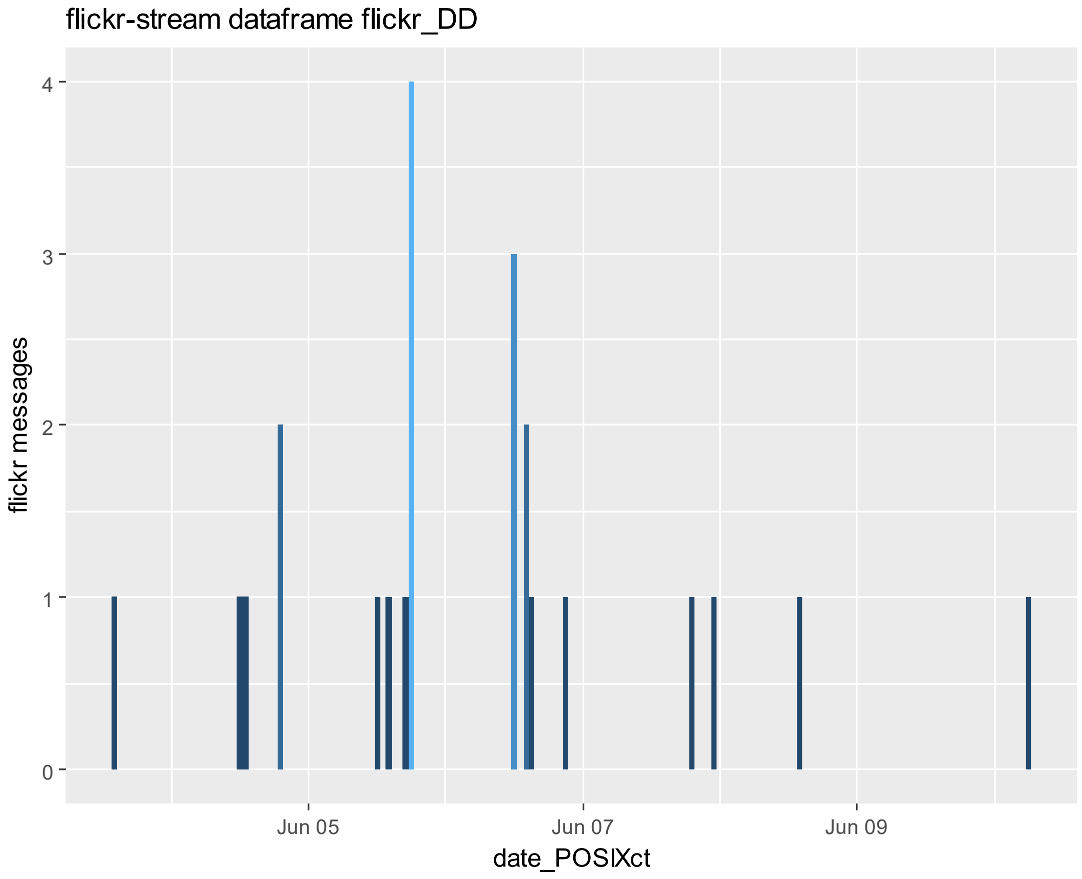


Figure 3-54: The temporal distribution of the flickr dataset in Dresden

Due to the similarity of the distribution of the Dresden twitter dataset, the same model as for that dataset was chosen (Figure 3-53, Figure 3-54).

```
fitcov3_flickr <- ppm(myspp_flickr_DD ~polynom(x,y,3) + Z, covariates=list(Z=landuse))  ## passt , dann hinweis auf numerische instabilität
diagnose.ppm(fitcov3_flickr, type="pearson")
fitcov3_flickr
```

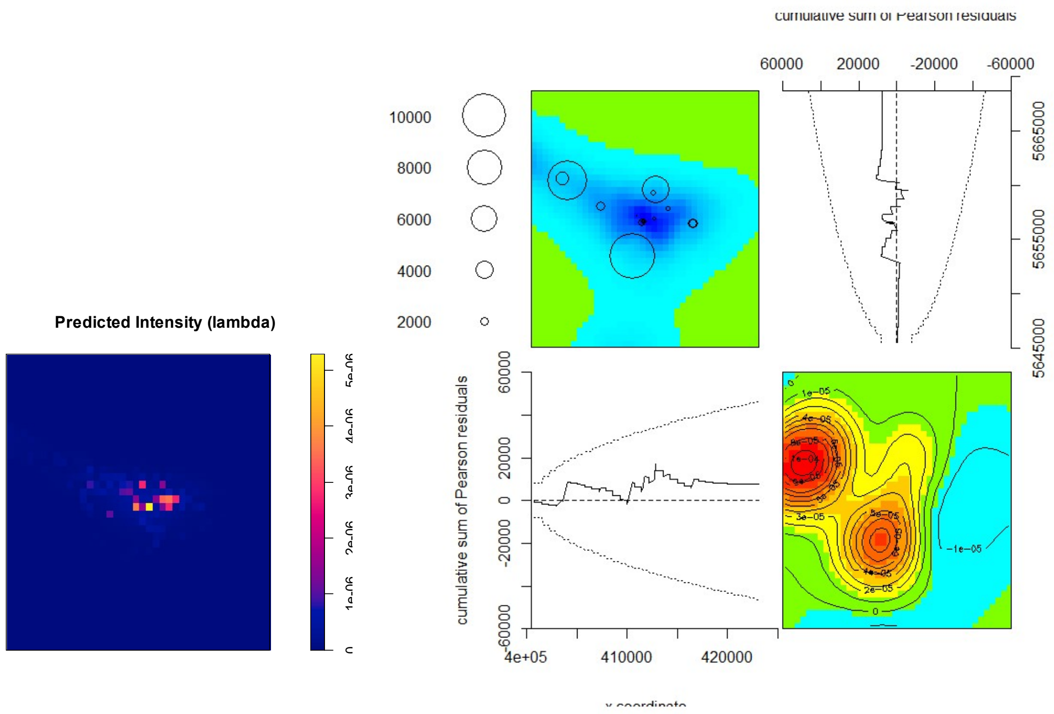


Figure 3-55: the predicted intensity and residual diagnostics of the model fitcov3_flickr

The resulting predicted intensity (Figure 3-55) seems similar to the point pattern and the residual plot shows a problem with the trend at the left corner. This is an indication that the model doesn't fit well in this area. Also, the Lurking variable plots show this deviation at the right end of the y-axis, probably because of the low number of points.

The observed K-function is still clustered up to a radius of 150 m and then the line crosses the envelope, following the line of the envelope perfectly (Figure 3-56). The QQ-plot shows that the data has higher variability than the smoothed residual field from the simulated Poisson model.

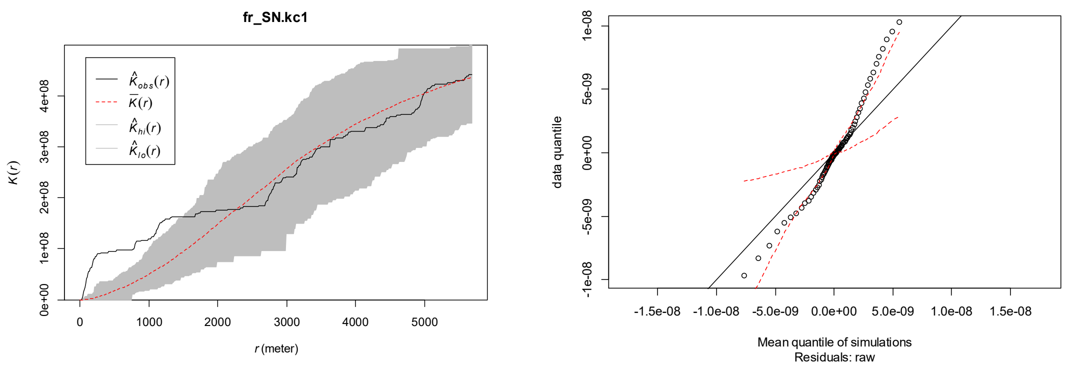


Figure 3-56: calculated and simulated K-function (model fitcov3_flickr) and corresponding QQ-Plot

3.3 Hurricane Sandy in 2012

The disaster event of Hurricane Sandy was the largest hurricane of the 2012 season. Sandy formed on the 22.10.2012 and had its landfall on the 30.10.2012 near Brigantine in New Jersey. At the time of landfall, the wind reached 70 knots and the storm surge was as high as 3.85 meters, with prevalent levels 0.8 and 2.8 meters along the coast of New Jersey and New York. The storm surge was responsible for up to 650,000 destroyed or damaged buildings. Over 8.5 million people were affected by power losses that lasted for weeks in some areas. Sandy caused 147 direct casualties along its path and brought damage in excess of $50 billion for the United States. Using the data DRYAD datasets for the following investigations was kindly allowed from the researchers Yury Kryvasheyeu et al. (Kryvasheyeu et al., 2015b).

3.3.1 Data preparation

The DRYAD datasets are perfectly documented and have a detailed installing instruction.

The dataset chosen for analysis mainly consists of messages with messages contains one or more instances of specific keywords, deemed to be relevant to the event and its consequences (“sandy”, “hurricane”, “storm”, “superstorm”, “flooding”, “blackout”, “gas”, “power”, “weather”, “climate”, etc.) posted between October 15 and November 12.

The data includes the text of messages and a range of additional information from the twitter data specification. An addition are the results from a Topsy sentiment analysis for wach tweet, resulting in sentiment levels.

Since only a minor fraction of the messages (about 1.2% until 1.5%) are geo-tagged by Twitter, the raw data was filtered to include only those messages that contain location information. The missing geocoordinates were gathered via the Google Maps API interpretation from self–reported location strings. Due to this, the precision of geocoding varies between the exact latitude and longitude of a user, as recorded by Twitter, and the coordinates of the centre of an administrative unit returned by the Google API.

The data was imported into a PostgreSQL database with PostGIS extension following the documentation. For further examination, the data was extracted for the region of Miami as an representative city before the landfall.

3.3.2 Data Visualisation

Mapping the spatial distribution of the tweet massages is the next step. The above-mentioned problem is visualized in Figure 3-58, the coordinates of the maximum amount of data is concentred in one location as the result of the Google – API translation. This data cannot be used because it would cause numerical problems like division by zero (“A matrix is 'singular' if its determinant is zero, so that it cannot be inverted). Also, the number of points for calculation is limited due to memory allocation.

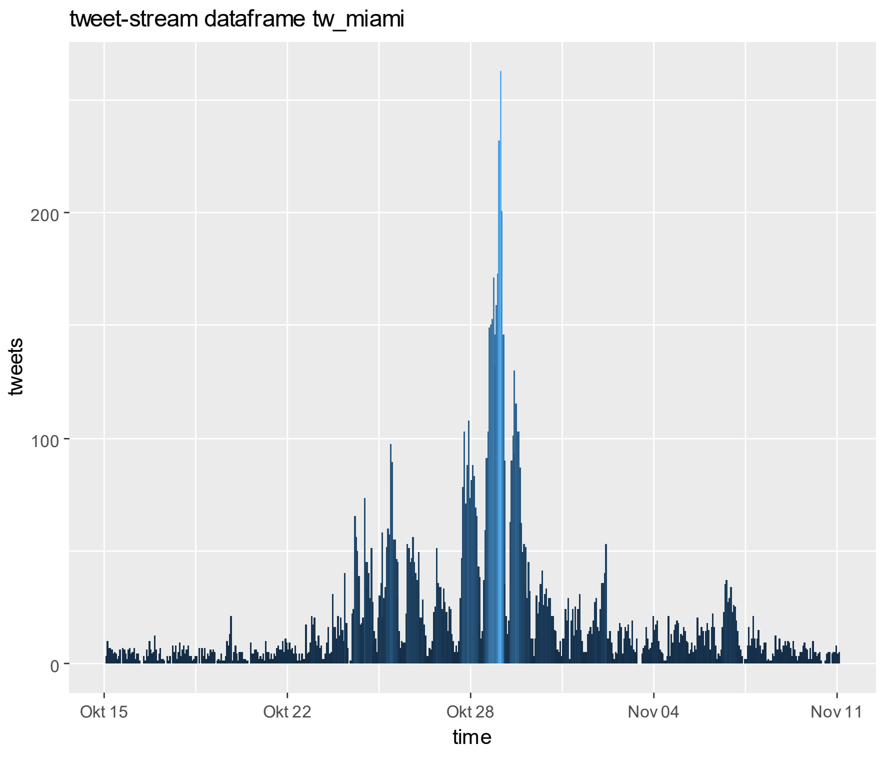


Figure 3-57: the temporal distribution of the Miami twitter dataset

Figure 3-57 shows the temporal distribution of the twitter feed in Miami. On 25.10.2012 the first small maximum occurred during the passing of Miami of hurricane Sandy and during the landfall of Hurricane Sandy on 30.10.2012 in Jersey the maximum with 250 related tweets per hour could be measured.

The longitude and latitude distribution as shown in Figure 3-58 is presenting the posterior correction of the coordinates. About 98% of the tweets are automatically geolocated by analysing location terms and the calculating the coordinates of the centre of an administrative unit – that is for 1000 tweets the number of 980 and Figure 3-58 show the result.

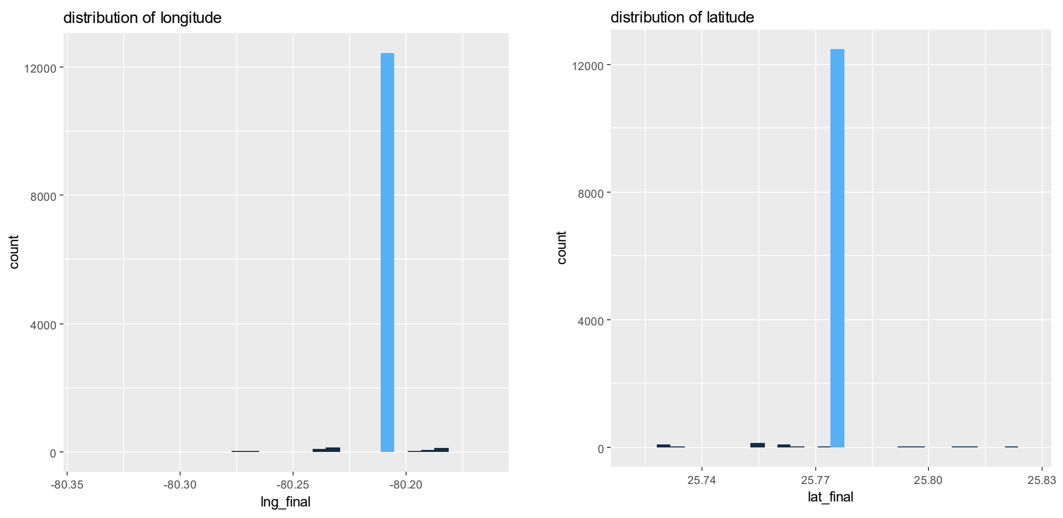
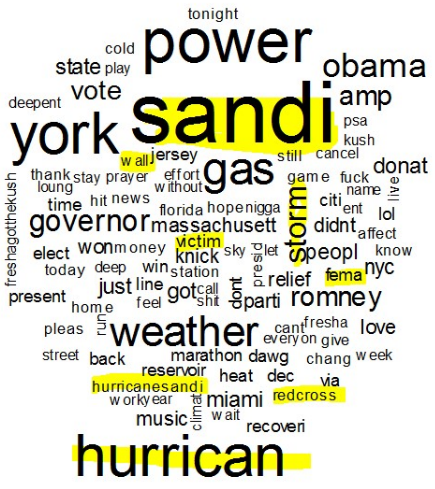
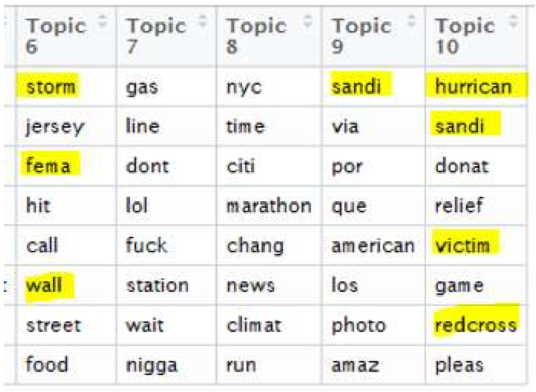


Figure 3-58: the lat/lon distribution of the Miami twitter dataset

3.3.3 Text Analysis

The result of the text analysis in both ways with R – textmining and the R-LDA and LDAviz packages offer expectable results. The terms “storm”, “sandi”, “hurricane”, “redcross”, “fema” and “wall” are detected from both methods. The LDAviz method was also able to group the terms to topics with topic modelling using LDA as expected (Figure 3-59).





Topic 6	Topic 7	Topic 8	Topic 9	Topic 10
storm	gas	nyc	sandi	hurrican
jersey	line	time	via	sandi
fema	dont	citi	por	donat
hit	lol	marathon	que	relief
call	fuck	chang	american	victim
wall	station	news	los	game
street	wait	climat	photo	redcross
food	nigga	run	amaz	pleas

Figure 3-59: the word cloud and the computed LDA Topics

3.3.4 Explanatory Analysis

The Heatmaps shows the city of Miami and the big number of georeferenced tweets in the centre of Miami as mentioned above (Figure 3-60).

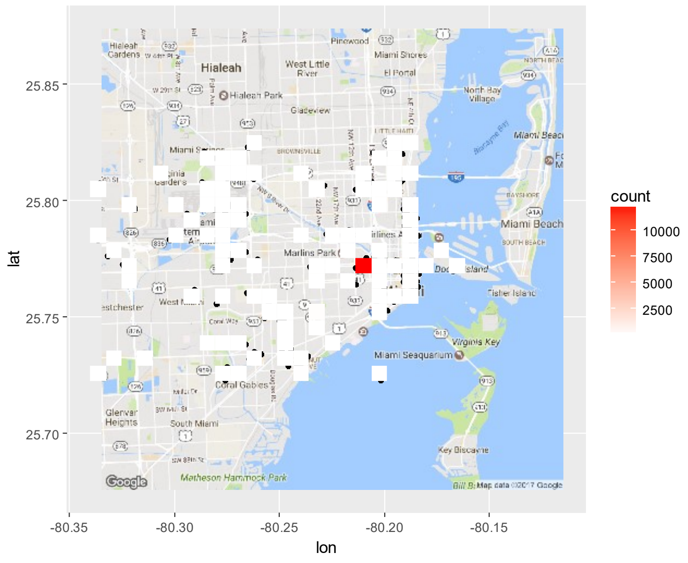


Figure 3-60: Heatmap of the twitter feed in Miami

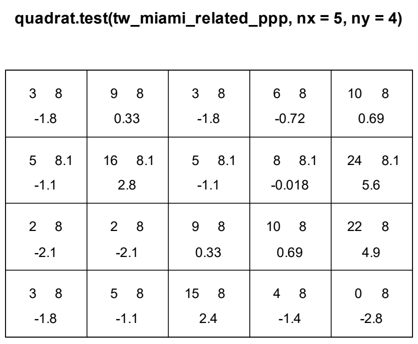
The Quadrat counting test for CSR shows that the point pattern are not a homogeneous Poisson point process. The low p-value declares it (Figure 3-61).

```
> > plot(quadrat.test(tw_miami_related_ppp, nx = 5, ny =
4))
> quadrat.test(tw_miami_related_ppp, nx = 5, ny = 4)

            Chi-squared test of CSR using quadrat counts
            Pearson X2 statistic

data:  tw_miami_related_ppp
X2 = 103.47, df = 19, p-value = 2.506e-13
alternative hypothesis: two.sided

Quadrats: 5 by 4 grid of tiles
```



3 8 -1.8	9 8 0.33	3 8 -1.8	6 8 -0.72	10 8 0.69
5 8.1 -1.1	16 8.1 2.8	5 8.1 -1.1	8 8.1 -0.018	24 8.1 5.6
2 8 -2.1	2 8 -2.1	9 8 0.33	10 8 0.69	22 8 4.9
3 8 -1.8	5 8 -1.1	15 8 2.4	4 8 -1.4	0 8 -2.8

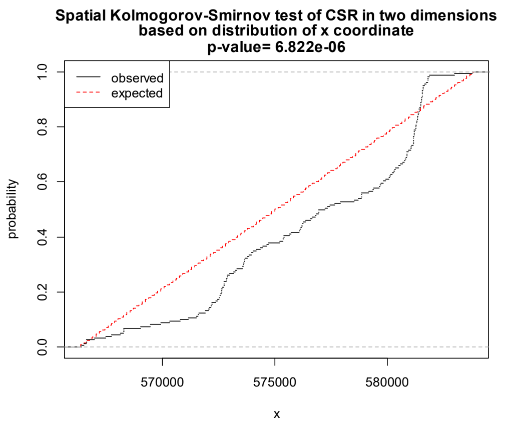
Figure 3-61: Quadrat test for dataset tw_miami_related_ppp

Testing of CSR with the Kolmogorov – Smirnow suggests with a small p-value that the empirical distribution was not generated under CSR (Figure 3-62).

```
> cdf.test(tw_miami_related_ppp, "x")

            Spatial Kolmogorov-Smirnov test of CSR in two
dimensions

data:  covariate ‘x’ evaluated at points of ‘tw_miami_re-
lated_ppp’
    and transformed to uniform distribution under CSR
D = 0.19772, p-value = 6.822e-06
alternative hypothesis: two-sided
```



```
> > cdf.test(tw_miami_related_ppp, "y")

            Spatial Kolmogorov-Smirnov test of CSR in two
dimensions

data:  covariate ‘y’ evaluated at points of ‘tw_miami_re-
lated_ppp’
    and transformed to uniform distribution under CSR
D = 0.12976, p-value = 0.008836
alternative hypothesis: two-sided
```

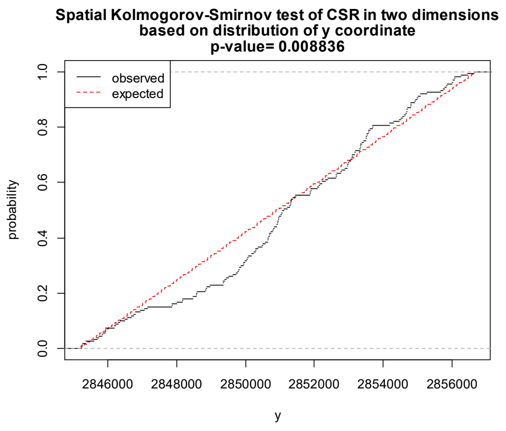


Figure 3-62: Results oft he Kolmogorov – Smirnow test of CSR (mySPP_tw_DD)

The plot of the inhomogeneous K-function is indicating an inhomogeneous Poisson distribution. The envelope plot shows a light dispersion from a radius = 2250 m (Figure 3-63).

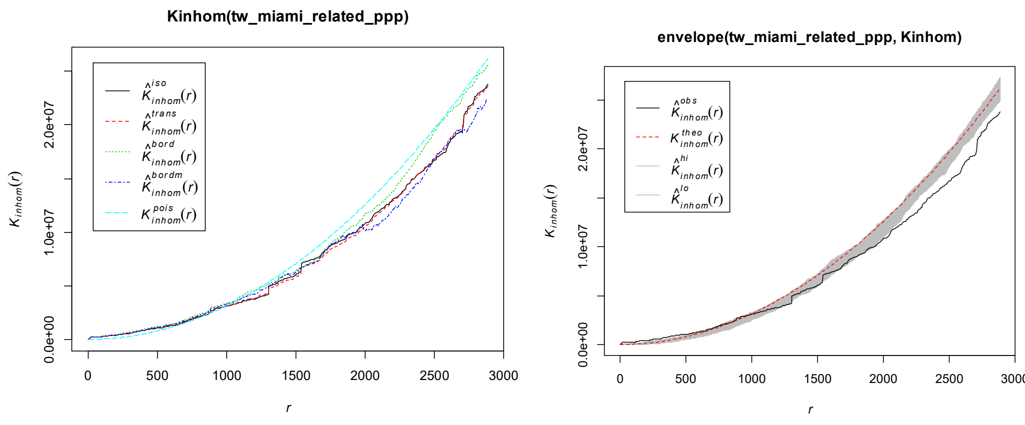


Figure 3-63: the inhomogeneous K-function with corresponding envelope plot

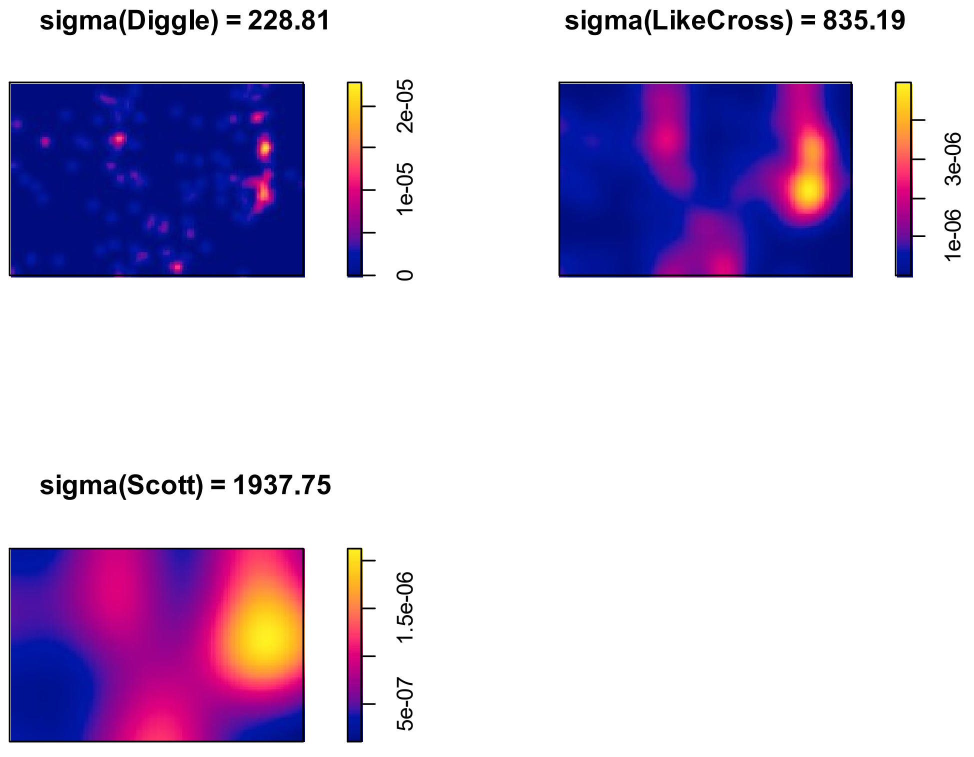


Figure 3-64: 3 sigmas for the standard variation of the Gaussian smoothing kernel

This computation indicates clustering the urban space dispersed with different small clusters along the costal line of Miami (Figure 3-64).

3.3.5 Model analysis of the Twitter feed in Miami

For the dataset concerning Miami, a polynomial model was chosen due to the good results from the previous models.

```
miami.fit2 <- ppm(tw_miami_related_ppp ~polynom(x,y,3
```

After the modelling process, residual analysis and checking the Q-Q-Plot follow:

Residual analysis

Figure 3-66 shows a diagnostic plot for a model with the correct form of the spatial trend a log quadratic spatial trend. The lurking variable plots with respect to the x- and y- coordinate variables show acceptable deviation from the 2б- limits, indicating that the model does account for a variation in intensity with respect to these variables. The K-function in Figure 3-65 shows that there is clustering over the entire range of the radius.

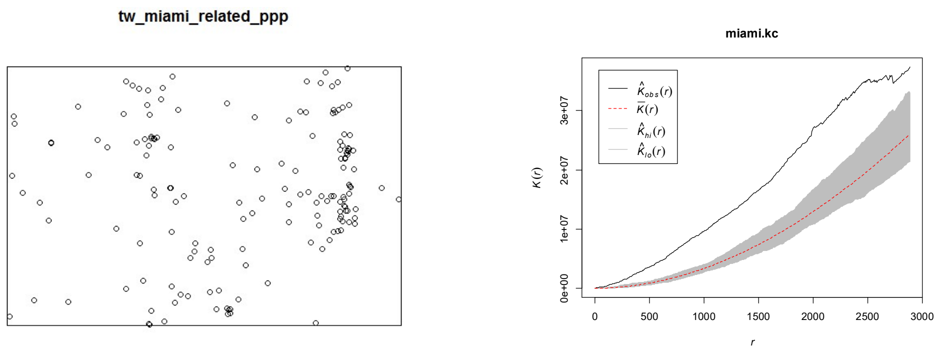


Figure 3-65: the ppp-object and the computed K-function with simulated CSR envelopes

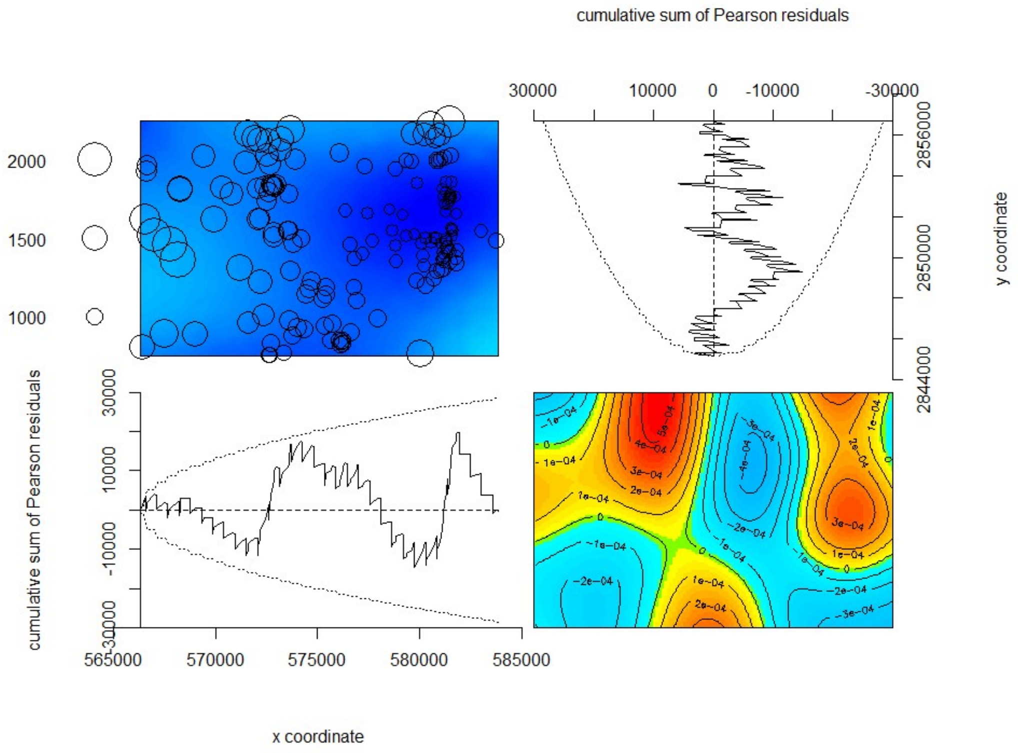


Figure 3-66: the residual analysis

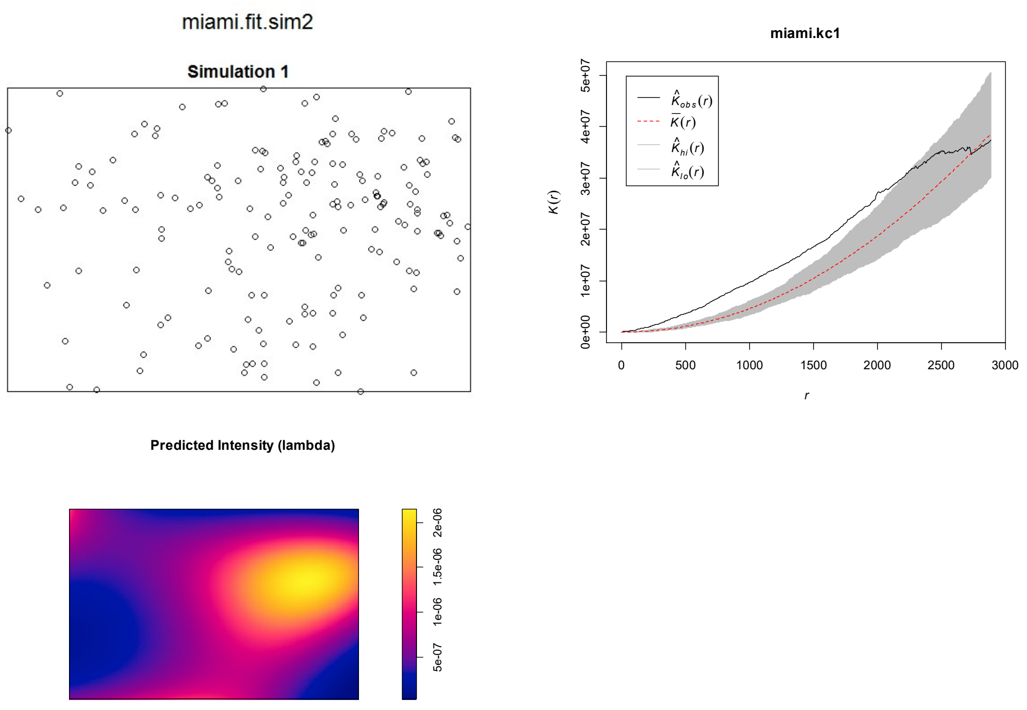


Figure 3-67: simulated points and predicted intensity (model miami.fit) (Left) and calculated and simulated K-function (Right)

The simulated K-function in Figure 3-65 shows that there is clustering over the entire range. Again, the K-function observed in Figure 3-67 shows the relation of the observed to the simulated distribution. The resulting lines showing that the pattern is significantly clustered until 2500 m. Figure 3-68 shows a Q-Q-plot with an acceptable fit to the interaction.

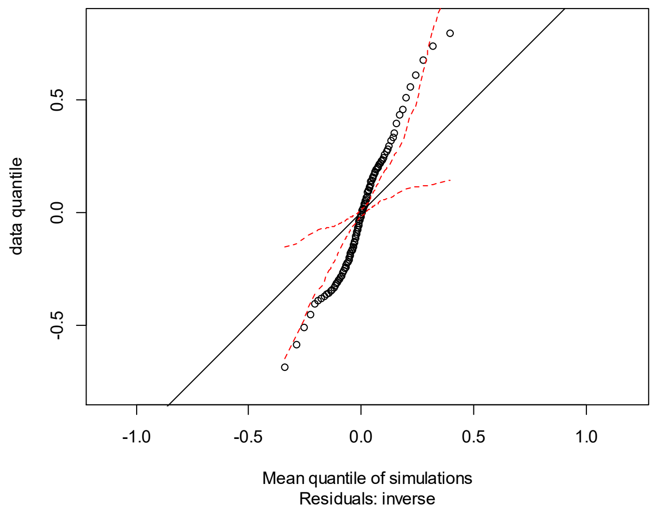


Figure 3-68: QQ-Plot of the model Miami.fit2

3.4 Spatio temporal sentiment analysis

For the spatio temporal analysis a 3D scatter plot will be used. A 3D scatter plot allows the visualization of multivariate data by taking using multiple scalar variables for different axes in phase space. The different variables are combined to form coordinates in the phase space and they are displayed using glyphs and colour so that the form of the association between the variates can be seen.

The advantage of this method is the direct exploration and discovery in the context of spatiotemporal environmental data (Friendly and Denis, 2005).

3.4.1 Spatio temporal analysis of sentiment and emotion marks

The visualisation of the following data is an approach to determine the relation of the sentiment marks of the social media stream with the chronology of the observed event.

Therefore we complement the data from the tweets from both dataset from the German flood in Dresden and the Hurricane Sandy in Miami with the covariate from the sentiment analysis and display the modified data with the covariate time. For an easier handling of the data, the time is represented by year days.

The Figure 3-69 up to Figure 3-73 show a significant dependence of the covariate emotion, respectively sentiment for each event. A second observation is that the biggest emotional response can be observed in the cities. Especially in Saxony there are few tweets but this may be due to the circumstance that in Germany there are generally only a few tweets observable because people don't use twitter very much.

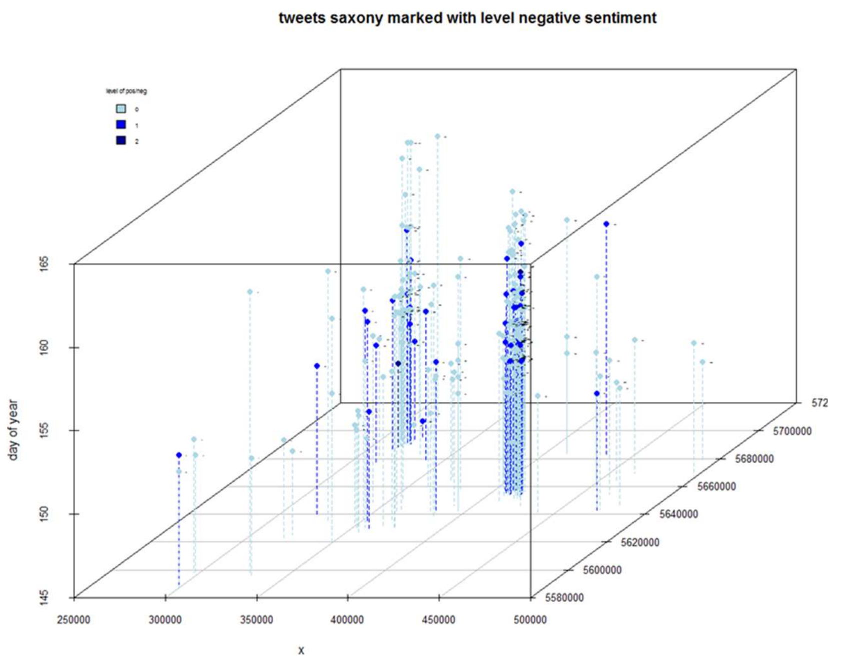


Figure 3-69: scatterplot from Saxony with the sentiment "negative"

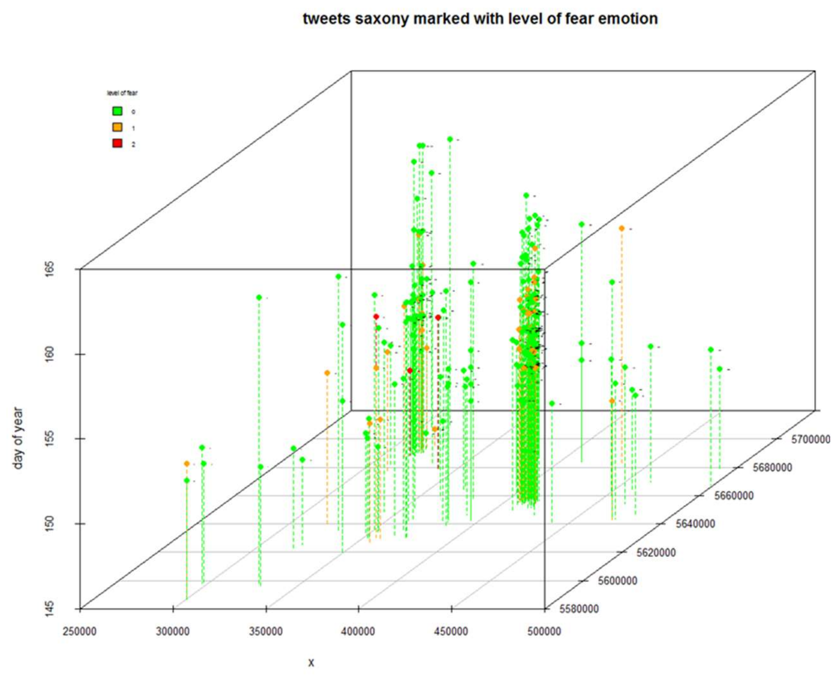


Figure 3-70: scatterplot from Saxony with the emotion "fear"

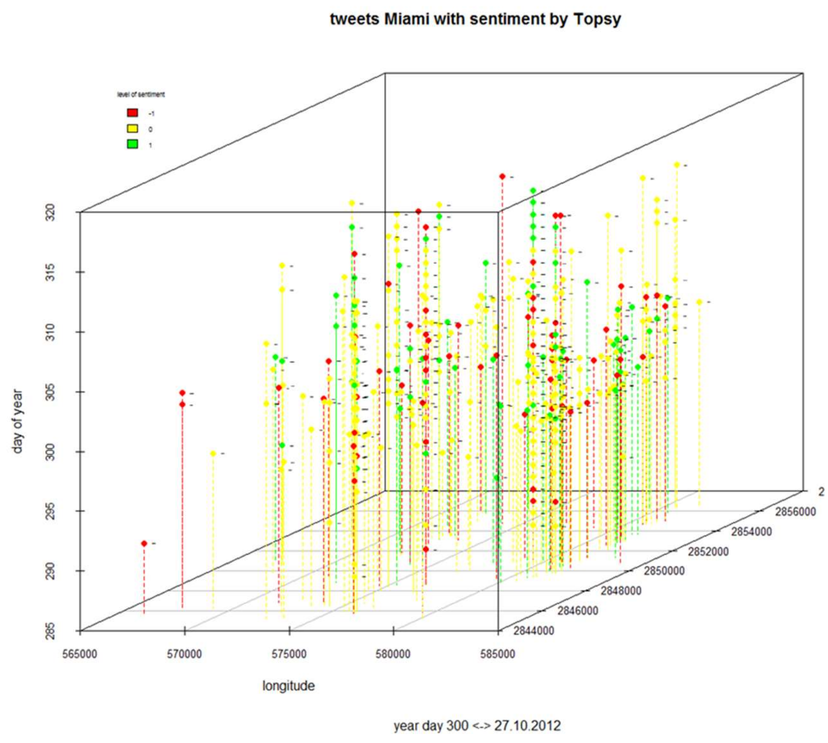


Figure 3-71: scatterplot from Miami with the sentiment gathered from Topsy

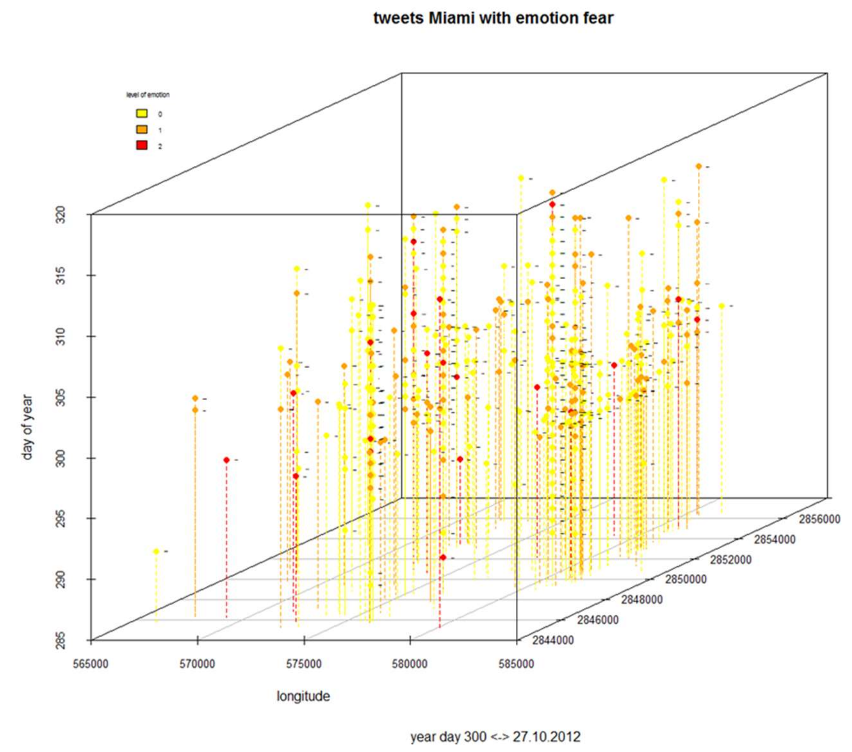


Figure 3-72: scatterplot from Miami with the emotion "fear"

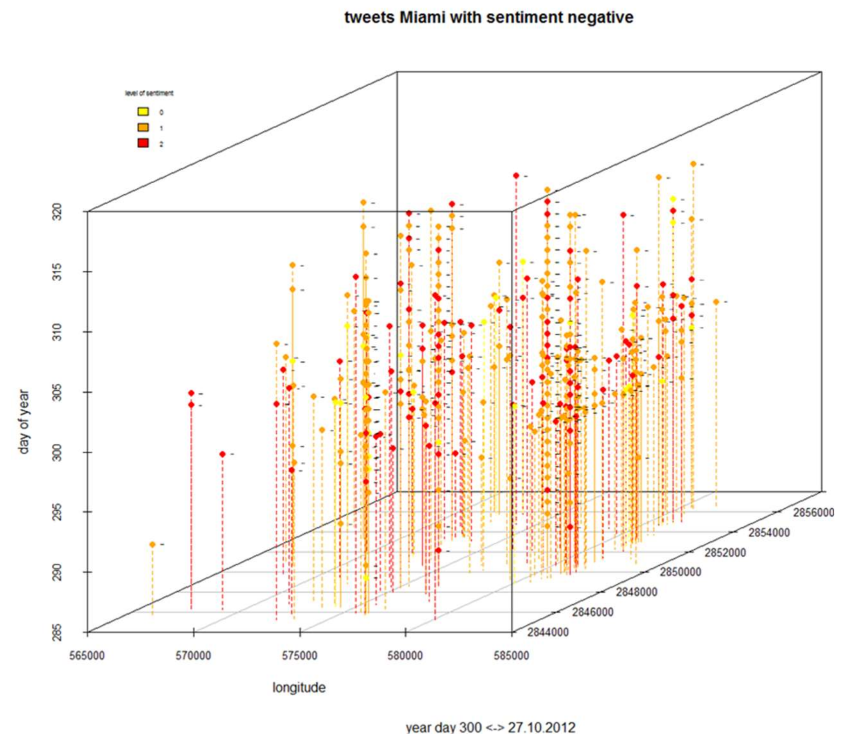
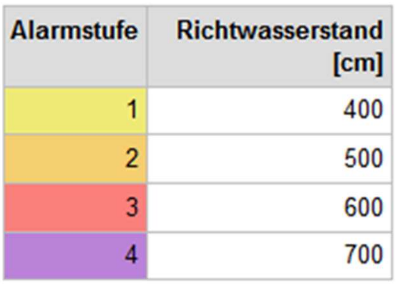


Figure 3-73: scatterplot from Miami with the sentiment "negative"

3.4.2 Spatio temporal analysis of the relation to measurement data

The following approach will show that in-situ measurement data can be used as a suitable covariate for point pattern analysis, giving answers to the question if such measures correlate with the social media stream.

Therefore, we convert the "real" measurement data in a more meaningful categorical variable. The alarm levels depend on the water levels of the river Elbe. Briefly explained, the alarm level 1 means that the water level is rising up to level 3, when interventions against the flood are staring up to, followed by level 4 with immediate danger to public or animal health. The gauge in Dresden at km 55,6 is assigned with the alarm levels shown in Figure 3-74.



Alarmstufe	Richtwasserstand [cm]
1	400
2	500
3	600
4	700

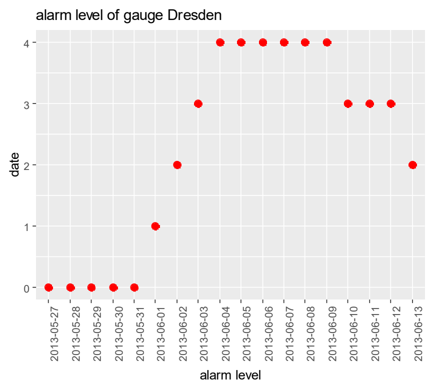


Figure 3-74: alarm level of gauge Dresden (Left) alarm level depending on time assigned to the date

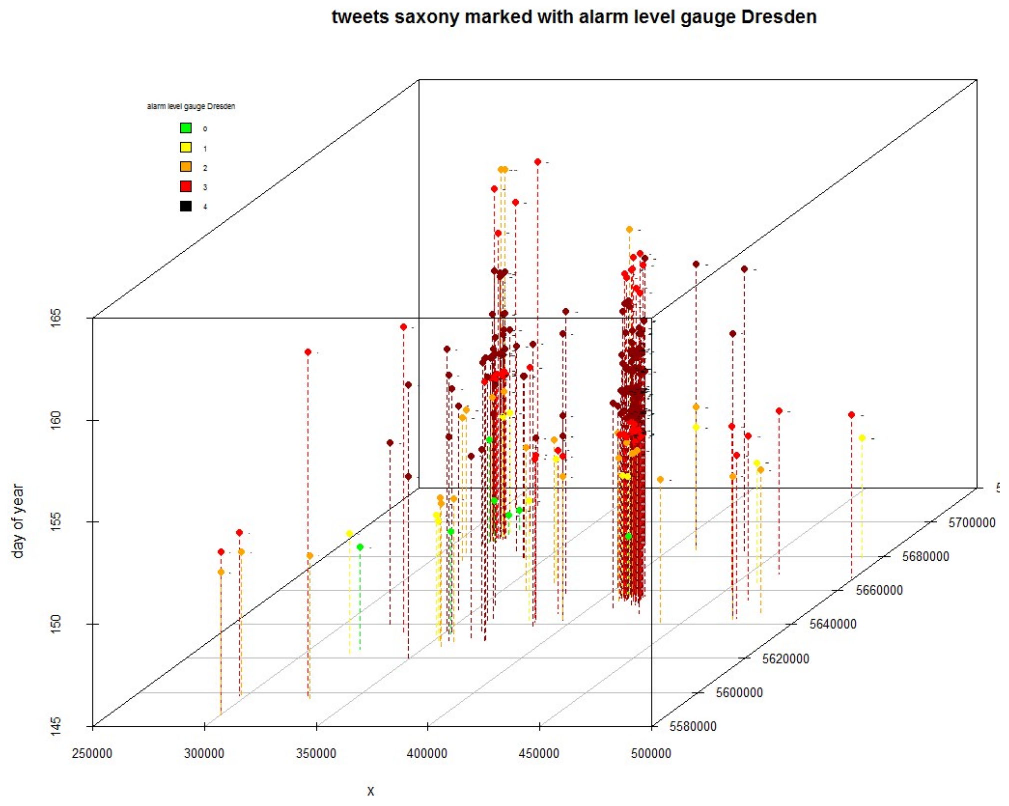


Figure 3-75: scatterplot of the tweets in Saxony with the covariate "alarm level"

The alarm levels were obtained from the administration and are displayed in Figure 3-74. Therefore, the dataset from Saxony was complemented with the covariate alarm level. For an easier handling of the data, the time is represented by year days (the year day 153 is 06.01.2013). The scatterplot shows that the most tweets were tweeted during the highest alarm level (Figure 3-75).

4 Discussion

4.1 Statistical text analysis

As described in the Methods chapter the software R is fully adequate to perform the statistical text analysis, spatial point pattern analysis and spatio-temporal analysis. The packages mainly used were TM, LDA, LDAviz, sp, spatstat and ggplot2. In very few instances numerical instabilities occurred, which may be caused by too large numbers due to UTM coordinates. They are documented in the Methods chapter. It is used by a huge community of researchers dealing with similar issues like this thesis, which give a very positive feedback (Wang et al., 2016).

Gathering data was at times quite a challenge, especially for a case study in Germany. The use of social media in Germany is poor. The work of Fuchs (2013a) (which investigates the twitter stream during the centennial flood in Germany, too and retrieved with round about 2500 flood related tweets the same number of tweets during the observed period. But identical results also have an affirming component.

During the preliminary phase of this thesis and during the literature review, the identified works relating to social media analysis including methods of spatial point pattern analysis analysing Poisson distributions and methods dealing with a sophisticated approach in statistically text analysis were poorly detected. Zhang et al. (2015) did an investigation using a hybrid mechanism based on latent Dirichlet allocation and document clustering to extract incident-level semantic information and they used spatial point pattern analysis to explore the spatial patterns and to assess the spatial dependence between incident-topic tweets and traffic incidents. With the LDA, they were able to drop the clutter and get the results they searched for.

The results of using LDA for this thesis were also positive. Especially the analysis of the tweets from Hurricane Sandy delivered identical results up to ten terms between the search word based approach and the LDA-based approach. That may be caused by the pre-filtered dataset of the Dryad Digital Repository (Kryvasheyeu et al., 2015b), but as mentioned before, similar finding temper the methods. The smart thing about the LDA is that it suggests trending topics and the R package LDAviz is a very helpful approach for researchers looking straight forward to finish the mining process. They get an intelligent tool to use, which visualizes the complexity of the LDA on the desktop. Moreover having the topics served is big advantage even during a crisis event when time is scarce. These methods have its charm because all approaches use text data for their analysis.

Therefore, the modern approach in clustering and topic modelling using the latent Dirichlet allocation (LDA) is reasonable and the third research question is closed successfully.

4.2 Spatial point pattern analysis

The spatial point pattern analysis with its methods in first-order and second-order analysis are well known and approved methods. Similar hotspots as in Albuquerque (2014) and in

Fuchs et al. (2013b) were discovered, heatmaps are still a common method to find out the points of interest for further investigation.

The Kernel Density Estimations always give a good impression of the scenario in the small areas of the case studies. In the author's opinion, it is reasonable to investigate the scenarios also on small scales for a better understanding of the distributions. Other researches mostly show a view related to the national borders (Albuquerque, 2014) (Fuchs et al., 2013b).

A strong alternative testing of complete spatial randomness (CSR) is the Kolmogorov – Smirnow test. Small p-value suggests that the empirical distribution was not generated under CSR for the entire area observed. Clustering is often discovered in the social media stream of both of twitter and flickr, impressive mapped for events around the US by Li et al. (Li et al., 2013)

The G-function in Figure 3-29 proves itself an appropriate method to indicate clustering or dispersing. Analysing the social media data sets results always clustering. The weakness of the G-functions is that it only looks at nearest neighbours and the resulting plot from the twitter dataset Dresden indicates, that 65% of the points have a neighbour within 200 m and 95 % within 800 m, always far away from complete spatial randomness (CSR).

The best results in modelling the datasets are indicated with interaction terms defined as an inhomogeneous Poisson process and a covariate from the density of the human settlements and transport infrastructure. The simulated points and predicted intensity e.g. from the fitted model (model tw_DD.fitcov2) of the tweets related to Dresden shown in Figure 3-33 and observed K-function with the calculated envelope for the same fitted model in Figure 3-34 show a model with a good fit.

According to Jian Yang et al. (2007) the residual analysis for the spatial point process used in the R-package spatstat from Baddeley et al. (2005b) is considered a milestone in the development of point pattern statistics that provides an excellent technique for determining the needs for model refinement or for diagnosing the quality of the model-fitting for spatial point processes in a statistically rigorous way.

For all models the residual analysis was done and shows acceptable fits of the models, especially under the use of the covariate. Apart from the model for Saxony, all of the Lurking variable plots look good for the x and the y variable, and don't leave the dot plotted area of the standard derivation. This may be caused by the dispersed events that act like outlines, but are still there.

To summarize, the investigations of the distribution of the social media feed on a small scale were done and the point pattern could be modelled reasonable as a spatial point pattern process with a polynomial Poisson process with a covariate for a possible better fitting of the model, the first and the second research question are answered successfully.

Following Yang et al. (2007), the model selection could be realized in a more sophisticated way by implementing a model selection process. They used the Akaike information criterion as a measure for selecting the best among competing models for a fixed data set. The question is, what is this good for. Remembering Figure 3-58, events with unique coordinates are a problem. Maybe for some use case it could be reasonable to calculate a model for the local distribution based on adequate data and the simulate of the coordinates for the "false" georeferenced events with that model. That could reduce misinterpretations of the first and second order analytics in some case.

4.3 Spatio-temporal analysis

For this approach the data is displayed in a 3D scatter plot with the coordinates used for the different axes in phase space. The different variables are combined to form coordinates in the phase space and they are displayed using glyphs and colour so that the form of the association between the variates can be seen.

Covariate Sentiment and emotion

Compared with other works, e.g. Shalunts et a. (2014), who did a sentiment analysis of German social media data for natural disaster with a classification in positive, negative mixed and neutral, or Buscaldi and Hernandez-Farias (2015) who investigated the sentiment from microblogs during the 2014 Genoa flooding, this work additionally considers not only the sentiment but also the emotions.

Mandel et al. (2012), who analysed the online sentiment during Hurricane Irene, did an appropriate approach, also identifying the gender of a sender, detected that female messages are generally more concerned. In future work, both approaches could be combined.

The scatterplots are created with the sentiment "negative" and the emotion "fear" under the assumption that during a crisis event relevant information is closely connected with emotions. In the author's opinion, fear is the most relevant variable.

Lu et al. (2015) investigate the Visualizing of Social Media Sentiment in Disaster Scenarios with the use of maps with kernel density estimation on positive and negative Tweets and a summarizing time line. In the author's opinion, scatterplots are a useful method to combine the view of time and geolocation in one figure.

Figures 3-69 up to figure 3-73 show an observable dependence of the covariate emotion respectively sentiment for each event. A second observation is that the biggest emotional response can be observed in the cities. Especially in Saxony there are few tweets but this may be due to the fact that in Germany there are in general only a few tweets to be observed because people don't use twitter very much.

In relation to Tobler's first law (1970) "everything is related to everything else, but near things are more related than distant things.", it can be postulated that a person "nearer to the action" acts with more emotion or has a higher emotional response to a crisis event than someone who is further away.

Hence, this observation concludes that sentiment and emotion are measurable variables during a crisis event and must be taken into consideration in the process of analysing social media therefore and the fourth research question can be closed successfully.

Covariate alarm level

Analysing the scatterplot in Figure 3-75 delivers two findings: First, the maximum of the tweet stream is during the phase of alarm level 4. Second, the locations of the tweets are mainly in the urban areas like Dresden and Leipzig. There is only little distribution over rural Saxony.

Hence, this observation concludes that comparing the social media feed with a categorical covariate as a translation of numeric measurement data is an appropriate way to visualize the correlation between those two factors.

Like the variables sentiment and emotion from the section above, the measurable variables can be a meaningful covariate during a crisis event and have to be taken into consideration in the process of analysing social media. The fifths and last research question can also be closed successfully.

Related work to this topic cannot be found. But for future work, this method could be useful to determine a covariate for modelling spatial point pattern processes.

5 Conclusions and outlook

Overall, it can be concluded that the methods of the spatial point pattern analysis are a reasonable, even if there is a limited amount of accurately georeferenced tweets and flickr messages during the German flood and during Hurricane Sandy. A spatial point pattern analysis and a spatiotemporal visual analysis of the data makes it still possible to detect significant events with reasonable accuracy. In all selected areas, it was possible to find an inhomogeneous Poisson model which could fit the point pattern process in a satisfactory way by the use of a covariate for the density of human settlements and transport infrastructure.

Also, a process model for the spatial point pattern analysis was evaluated, including the process step data gathering, first-order (KDE) and second order (K-function) spatial pattern analysis, the modelling of the spatial point process and the analysis of the model (residual analysis) as a guideline for further investigations.

Considering the related work as a whole, one might get the impression that a lot of work has been done with a more aerial view of the events. This work may be a reasonable addition with a smaller scale of view, focussing on single locations like towns or other reasonable cuttings from the observed areas with a focus of the distribution of the local social media feed.

For further research, the following questions can be suggested:

- The development of a suitable model selection process for choosing a reasonable model to be probably used in an automatic online process.
- The research of a feasible method to handle the geo-referencing problem with e.g. the Yahoo API. Is there a hidden pattern which would allow us to determine a more realistic geo-reference process than just calculating the administrative middle of an area? Or – with the view on using spatstat, which doesn't allow unique coordinates: Is the information loss of the excluded points relevant to the model?
- The development of a software which combines the modelling process with a spatio-temporal component bringing the possibility to simulate these processes. Such a system can be used to simulate the social media response of a fictitious disaster event and train other systems or disaster response organisations under realistic conditions. Different covariates must be defined, describing the variety of possible crisis events. They could also be used to calibrate realistic scenarios.

Gathering data and extracting information is a very important process. Therefore, the classic method of search, word based text mining, was added to the unsupervised and machine learning based method of the Latent Dirichlet Allocation (LDA). Even researchers with less skills in extracting text data can act quickly to extract relevant topics out of their social media documents.

The investigation of the spatial distribution of sentiment especially of the emotion "fear" with a stronger relationship to crisis events as the sentiment "negative" will be a challenge to the process of statistical text mining. Machine Learning will be a helpful, but the human language is complex and the Babylonian linguistic confusion still exists. Analysing social media from so many languages and cultures is a respectable mission.

For further research, the following questions can be suggested:

- How can the emotional factor be integrated into the analysis of social media feeds in the context of crisis events?
- How could a possible early warning sensor focused on an observation of only emotional content be realized, even in multi crisis events?

Translating real in-situ measurement data into reasonable categories is a quite similar challenge. Nobody knows what a tweet like “Huh, it’s wet!” means in relation to 4.70 m. But in relation to an alert level of 4 meaning immediate danger to public or animal health it will perhaps be interpreted differently. An abstract from a single value with a strong relation of its meaning for an environmental hazard defines a good covariate.

The visualisation with 3D-scatterplott is helpful, it is a descriptive way to visualize multidimensional data. And visualizing data and their interpretation through graphic representation seems to be a research field on its own. While writing the section “Related work” it was noticeable that similar statements were differently visualized and perceived different first impressions. So, for further research the following question can be suggested:

- Would it be an advantage, if there were standards for visualisation in specific areas of analysing social media?

6 References

Albuquerque, J.P. de, 2014. Does the spatiotemporal distribution of tweets match the spatiotemporal distribution of flood phenomena ? A study about the River Elbe Flood in June 2013. Proc. 11th Int. ISCRAM Conf.

Ananiadou, S., McNaught, J. (Eds.), 2006. Text mining for biology and biomedicine. Artech House, Boston.

Anselin, L., 1995. Local indicators of spatial association - LISA. Geogr. Anal. 27, 93.

Applying Machine Learning to Text Mining with Amazon S3 and RapidMiner | AWS Big Data Blog [WWW Document], n.d. URL https://aws.amazon.com/de/blogs/big-data/applying-machine-learning-to-text-mining-with-amazon-s3-and-rapidminer/ (accessed 12.4.16).

Baddeley, A., Rubak, E., Turner, R., 2015. Spatial point patterns: methodology and applications with R, Champan & Hall/CRC Interdisciplinary Statistics Series. CRC Press, Taylor & Francis Group, Boca Raton ; London ; New York.

Baddeley, A., Turner, R., Moller, J., Hazelton, M., 2005a. Residual analysis for spatial point processes (with discussion). J. R. Stat. Soc. Ser. B Stat. Methodol. 67, 617–666. doi:10.1111/j.1467-9868.2005.00519.x

Baddeley, A., Turner, R., Moller, J., Hazelton, M., 2005b. Residual analysis for spatial point processes. J. R. Stat. Soc. Ser. B-Stat. Methodol. 67, 617–651.

Baddeley, A., Turner, R., Rubak, E., 2016. Adjusted composite likelihood ratio test for spatial Gibbs point processes. J. Stat. Comput. Simul. 86, 922–941. doi:10.1080/00949655.2015.1044530

Baeza-Yates, R., Ribeiro, B. de A.N., 2000. Modern information retrieval, 3. [pr.]. ed. ACM Press ua, New York, NY [u.a.].

Bailey, T.C., Gatrell, A.C., 1995. Interactive spatial data analysis. Longman Scientific & Technical ; J. Wiley, Harlow Essex, England : New York, NY.

Bakillah, M., Li, R.-Y., Liang, S.H.L., 2015. Geo-located community detection in Twitter with enhanced fast-greedy optimization of modularity: the case study of typhoon Haiyan. Int. J. Geogr. Inf. Sci. 29, 258–279. doi:10.1080/13658816.2014.964247

Besag, J., 1977. Discussion of "Modelin g spatial patterns" by B. D. Ripley. J. R. Stat. Soc. B39, 193–195.

Bischof, J., Airoldi, E., 2012. Summarizing topical content with word frequency and exclusivity, in: Langford, J., Pineau, J. (Eds.), Proceedings of the 29th International Conference on Machine Learning (ICML-12), ICML '12. Omnipress, New York, NY, USA, pp. 201–208.

Blei, D.M., 2012. Probabilistic topic models. Commun. ACM 55, 77. doi:10.1145/2133806.2133826

Blei, D.M., Lafferty, J.D., 2007. A correlated topic model of Science. Ann. Appl. Stat. 1, 17–35. doi:10.1214/07-AOAS114

Blei, D.M., Ng, A.Y., Jordan, M.I., 2003. Latent Dirichlet Allocation. J Mach Learn Res 3, 993–1022.

Bohnet, B., 2010. Very High Accuracy and Fast Dependency Parsing is Not a Contradiction, in: Proceedings of the 23rd International Conference on Computational Linguistics, COLING ’10. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 89–97.

Bravo-Marquez, F., Frank, E., Pfahringer, B., 2016. Building a Twitter opinion lexicon from automatically-annotated tweets. Knowl.-Based Syst., New Avenues in Knowledge Bases for Natural Language Processing 108, 65–78. doi:10.1016/j.knosys.2016.05.018

Bray, A., Wong, K., Barr, C.D., Schoenberg, F.P., 2014. Voronoi residual analysis of spatial point process models with applications to California earthquake forecasts. Ann. Appl. Stat. 8, 2247–2267. doi:10.1214/14-AOAS767

Buscaldi, D., Hernandez-Farias, I., 2015. Sentiment Analysis on Microblogs for Natural Disasters Management: a Study on the 2014 Genoa Floodings. ACM Press, pp. 1185–1188. doi:10.1145/2740908.2741727

Chang, J., 2015. lda: Collapsed Gibbs Sampling Methods for Topic Models.

Chang, J., Boyd-Graber, J., Wang, C., Gerrish, S., Blei, D.M., 2009. Reading Tea Leaves: How Humans Interpret Topic Models, in: Neural Information Processing Systems. Vancouver, BC.

Chiticariu, L., Krishnamurthy, R., Li, Y., Raghavan, S., Reiss, F.R., Vaithyanathan, S., 2010. SystemT: An Algebraic Approach to Declarative Information Extraction, in: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL ’10. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 128–137.

Chuang, J., Manning, C.D., Heer, J., 2012a. Termite: visualization techniques for assessing textual topic models. ACM Press, p. 74. doi:10.1145/2254556.2254572

Chuang, J., Ramage, D., Manning, C., Heer, J., 2012b. Interpretation and trust: designing model-driven visualizations for text analysis. ACM Press, p. 443. doi:10.1145/2207676.2207738

CRAN, 2016. The Comprehensive R Archive Network [WWW Document]. Httpcranr-Proj. URL https://cran.r-project.org/ (accessed 12.20.16).

Cressie, N., Read, T., 1984. Multinomial goodness-of-fit tests. J. R. Stat. Soc. Ser. B Methodol. 46, 440.

Cromley, E.K., McLafferty, S., 2012. GIS and public health, 2nd ed. ed. The Guilford Press, New York.

Cui, C., Wang, J., Wu, Z., Ni, J., Qian, T., 2016. The Socio-Spatial Distribution of Leisure Venues: A Case Study of Karaoke Bars in Nanjing, China. ISPRS Int. J. Geo-Inf. 5, 150. doi:10.3390/ijgi5090150

De Longueville, B., Annoni, A., Schade, S., Ostlaender, N., Whitmore, C., 2010. Digital Earth’s Nervous System for crisis events: real-time Sensor Web Enablement of Volunteered Geographic Information. Int. J. Digit. Earth 3, 242–259. doi:10.1080/17538947.2010.484869

Diggle, P., 2014. Statistical analysis of spatial and spatio-temporal point patterns, Third edition. ed, Monographs on statistics and applied probability. CRC Press, Taylor & Francis Group, Boca Raton.

Diggle, P., 1985. KERNEL METHOD FOR SMOOTHING POINT PROCESS DATA. J. R. Stat. Soc. Ser. C Appl. Stat. 34, 138–147.

Diggle, P.J., Gómez-Rubio, V., Brown, P.E., Chetwynd, A.G., Gooding, S., 2007. Second-Order Analysis of Inhomogeneous Spatial Point Processes Using Case-Control Data. Biometrics 63, 550–557. doi:10.1111/j.1541-0420.2006.00683.x

ESRI, 2016. Regression analysis basics [WWW Document]. URL http://resources.esri.com/help/9.3/arcgisengine/java/gp_toolref/Spatial_Statistics_toolbox/regression_analysis_basics.htm (accessed 12.18.16).

Feinerer, I., Hornik, K., Meyer, D., 2008. Text Mining Infrastructure in R. J. Stat. Softw. 25. doi:10.18637/jss.v025.i05

Feinerer, I., Hornik, K., Software, A., Ghostscript), I. (pdf_info ps taken from G., 2015. tm: Text Mining Package.

Ferrucci, D., Lally, A., n.d. UIMA: An Architectural Approach to Unstructured Information Processing in the Corporat e Research Environment [WWW Document]. URL https://www.fbi.h-da.de/fileadmin/personal/b.harriehausen/JIM-Project_UIMA/JNLE2004_UIMA_Paper_markedToAppear_20040122.pdf (accessed 12.4.16).

Fischer, M.M., 2000. Recent Advances in Spatial Data Analysis [WWW Document]. URL http://epub.wu.ac.at/4243/ (accessed 12.16.16).

Friendly, M., Denis, D., 2005. The early origins and development of the scatterplot. J. Hist. Behav. Sci. 41, 103–130. doi:10.1002/jhbs.20078

Fuchs, G., Andrienko, N., Andrienko, G., Bothe, S., Stange, H., 2013a. Tracing the German centennial flood in the stream of tweets: first lessons learned. ACM Press, pp. 31–38. doi:10.1145/2534732.2534741

Fuchs, G., Andrienko, N., Andrienko, G., Bothe, S., Stange, H., 2013b. Tracing the German centennial flood in the stream of tweets: first lessons learned. Presented at the Proceedings of the Second ACM SIGSPATIAL International Workshop on Crowdsourced and Volunteered Geographic Information, ACM, pp. 31–38. doi:10.1145/2534732.2534741

Getis, A., Ord, J.K., 2010. The Analysis of Spatial Association by Use of Distance Statistics. Geogr. Anal. 24, 189–206. doi:10.1111/j.1538-4632.1992.tb00261.x

Getis, A., Ord, J.K., 1992. The Analysis of Spatial Association by Use of Distance Statistics. Geogr. Anal. 24, 189–206. doi:10.1111/j.1538-4632.1992.tb00261.x

Goodchild, M.F., 2007. Citizens as sensors: the world of volunteered geography. GeoJournal 69, 211–221. doi:10.1007/s10708-007-9111-y

Goodchild, M.F., 1986. Spatial autocorrelation, CATMOG. Geo Books, Norwich.

Gorr, W.L., Olligschlaeger, A.M., 2010. Weighted Spatial Adaptive Filtering: Monte Carlo Studies and Application to Illicit Drug Market Modeling. Geogr. Anal. 26, 67–87. doi:10.1111/j.1538-4632.1994.tb00311.x

Grün, B., Hornik, K., 2016. topicmodels: Topic Models.

Hastie, T., Tibshirani, R., Friedman, J.H., 2009. The elements of statistical learning: data mining, inference, and prediction, 2nd ed. ed, Springer series in statistics. Springer, New York, NY.

Howe, J., 2009. Crowdsourcing: why the power of the crowd is driving the future of business, 1st paperback ed. ed. Three Rivers Press, New York.

IÖR-Monitor: Rasterkarten [WWW Document], n.d. URL http://maps.ioer.de/detailviewer/raster/ (accessed 12.25.16).

Jockers, M., 2016. syuzhet: Extracts Sentiment and Sentiment-Derived Plot Arcs from Text.

Jurafsky, D., Martin, J.H., 2009. Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition, 2nd ed. ed, Prentice Hall series in artificial intelligence. Pearson Prentice Hall, Upper Saddle River, N.J.

Kelejian, H.H., Prucha, I.R., 1999. A generalized moments estimator for the autoregressive parameter in a spatial model. Int. Econ. Rev. 40, 509.

Klonner, C., Marx, S., Usón, T., Albuquerque, J., Höfle, B., 2016. Volunteered Geographic Information in Natural Hazard Analysis: A Systematic Literature Review of Current Approaches with a Focus on Preparedness and Mitigation. ISPRS Int. J. Geo-Inf. 5, 103. doi:10.3390/ijgi5070103

Kryvasheyeu, Y., Chen, H., Moro, E., Van Hentenryck, P., Cebrian, M., 2015a. Performance of Social Network Sensors during Hurricane Sandy. PLOS ONE 10, e0117288. doi:10.1371/journal.pone.0117288

Kryvasheyeu, Y., Chen, H., Moro, E., Van Hentenryck, P., Cebrian, M., 2015b. Data from: Performance of social network sensors during Hurricane Sandy.

Leetaru, K., Wang, S., Padmanabhan, A., Shook, E., 2013. Mapping the global Twitter heartbeat: The geography of Twitter. First Monday 18. doi:10.5210/fm.v18i5.4366

Li, L., Goodchild, M.F., 2010. The Role of Social Networks in Emergency Management: A Research Agenda. Int. J. Inf. Syst. Crisis Response Manag. 2, 48–58. doi:10.4018/jiscrm.2010100104

Li, L., Goodchild, M.F., Xu, B., 2013. Spatial, temporal, and socioeconomic patterns in the use of Twitter and Flickr. Cartogr. Geogr. Inf. Sci. 40, 61–77. doi:10.1080/15230406.2013.777139

Lu, Y., Hu, X., Wang, F., Kumar, S., Liu, H., Maciejewski, R., 2015. Visualizing Social Media Sentiment in Disaster Scenarios. ACM Press, pp. 1211–1215. doi:10.1145/2740908.2741720

Mandel, B., Culotta, A., Boulahanis, J., Stark, D., Lewis, B., Rodrigue, J., 2012. A Demographic Analysis of Online Sentiment During Hurricane Irene, in: Proceedings of the Second Workshop on Language in Social Media, LSM ’12. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 27–36.

Manning, C.D., Raghavan, P., Schütze, H., 2008. Introduction to information retrieval. Cambridge University Press, New York.

Manning, C.D., Schütze, H., 1999. Foundations of statistical natural language processing. MIT Press, Cambridge, Mass.

Mark Steyvers, Tom Griffiths, 2004. Probabilistic approaches to semantic representation.

Marti Hearst: What Is Text Mining? [WWW Document], n.d. URL http://people.ischool.berkeley.edu/~hearst/text-mining.html (accessed 12.4.16).

Medhat, W., Hassan, A., Korashy, H., 2014. Sentiment analysis algorithms and applications: A survey. Ain Shams Eng. J. 5, 1093–1113. doi:10.1016/j.asej.2014.04.011

Meier, P., 2015. Digital humanitarians: how big data is changing the face of humanitarian response. CRC Press, Taylor & Francis Group, Boca Raton, FL.

Miner, G. (Ed.), 2012. Practical text mining and statistical analysis for non-structured text data applications, 1st ed. ed. Academic Press, Waltham, MA.

Mitchell, T.M., 1997. Machine learning. McGraw-Hill, New York, NY [u.a.].

Mohammad, S.M., Turney, P.D., 2010. Emotions evoked by common words and phrases: Using Mechanical Turk to create an emotion lexicon, in: Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text. Association for Computational Linguistics, pp. 26–34.

Møller, J., Waagepetersen, R.P., 2004. Statistical inference and simulation for spatial point processes. Chapman & Hall/CRC, Boca Raton.

Moran, P.A.P., 1950. Notes on Continuous Stochastic Phenomena. Biometrika 37, 17. doi:10.2307/2332142

Morisita, M., 1959. Measuring of the dispersion of individuals and analysis of the distributional patterns. Mem Fac Sci Kyushu Univ Ser E 2, 5–235.

Munasinghe, R.L., Morris, R.D., 1996. LOCALIZATION OF DISEASE CLUSTERS USING REGIONAL MEASURES OF SPATIAL AUTOCORRELATION. Stat. Med. 15, 893–905. doi:10.1002/(SICI)1097-0258(19960415)15:7/9<893::AID-SIM258>3.0.CO;2-M

Munzert, S., 2014. Automated data collection with R: a practical guide to Web scraping and text mining. John Wiley & Sons Inc, HobokenChichester, West Sussex, United Kingdom.

NaCTeM, n.d. The National Centre for Text Mining — University of Manchester [WWW Document]. URL http://www.nactem.ac.uk/brochure/NaCTeM_Brochure.pdf (accessed 12.4.16).

O'Sullivan, D., Unwin, D., 2010. Geographic information analysis, 2nd ed. ed. John Wiley & Sons, Hoboken, N.J.

Pang, B., Lee, L., 2004. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. Association for Computational Linguistics, p. 271–es. doi:10.3115/1218955.1218990

Paramesha, K., Ravishankar, K.C., 2016. A Perspective on Sentiment Analysis. ArXiv160706221 Cs.

Resch, B., 2013. People as Sensors and Collective Sensing-Contextual Observations Complementing Geo-Sensor Network Measurements, in: Krisp, J.M. (Ed.), Progress in Location-Based Services. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 391–406.

Resnik, P., Hardisty, E., 2010. Gibbs sampling for the uninitiated. DTIC Document.

Ripley, B.D., 1977a. Modelling Spatial Patterns. J. R. Stat. Soc. Ser. B Methodol. 39, 172–212.

Ripley, B.D., 1977b. Modelling Spatial Patterns. J. R. Stat. Soc. Ser. B Methodol. 39, 172–212.

Robertson, S.E., Jones, K.S., 1976. Relevance weighting of search terms. J. Am. Soc. Inf. Sci. 27, 129–146. doi:10.1002/asi.4630270302

Sagl, G., Blaschke, T., Beinat, E., Resch, B., 2012. Ubiquitous Geo-Sensing for Context-Aware Analysis: Exploring Relationships between Environmental and Human Dynamics. Sensors 12, 9800–9822. doi:10.3390/s120709800

Sakaki, T., Okazaki, M., and Matsuo, Y., 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. Proc. 19th Int. Conf. World Wide Web 851 – 860.

Samuel, A.L., 1969. Some studies in machine learning using the game of checkers. II-Recent progress. Annu. Rev. Autom. Program. 6, 1–36.

Sarawagi, S., 2007. Information Extraction. Found. Trends® Databases 1, 261–377. doi:10.1561/1900000003

Scott, D.W., 2004. Multivariate Density Estimation and Visualization (Papers / Humboldt-Universität Berlin, Center for Applied Statistics and Economics (CASE) No. 2004,16).

Shalunts, G., Backfried, G., Prinz, K., 2014. Sentiment analysis of German social media data for natural disasters. Presented at the 11th International conference on information systems for crisis response and management, ISCRAM.

Shelton, Taylor, Ate Poorthuis, Mark Graham, and Matthew Zook, n.d. 1992. “Hexagon Mosaic Maps for Display of Univariate and Bivariate Geographical Data,” Cartography and Geographic Information Systems 19 (4): 228–236.

spatstat - Resources [WWW Document], n.d. URL https://spatstat.github.io/resources.html (accessed 11.17.16).

Spinsanti, L., Ostermann, F., 2013. Automated geographic context analysis for volunteered information. Appl. Geogr. 43, 36–44. doi:10.1016/j.apgeog.2013.05.005

Steiger, E., Westerholt, R., Resch, B., Zipf, A., 2015. Twitter as an indicator for whereabouts of people? Correlating Twitter with UK census data. Comput. Environ. Urban Syst. 54, 255–265. doi:10.1016/j.compenvurbsys.2015.09.007

Sun, D., Li, S., Zheng, W., Croitoru, A., Stefanidis, A., Goldberg, M., 2016. Mapping floods due to Hurricane Sandy using NPP VIIRS and ATMS data and geotagged Flickr imagery. Int. J. Digit. Earth 9, 427–441. doi:10.1080/17538947.2015.1040474

The R Foundation, 2016. R: What is R? [WWW Document]. URL https://www.r-project.org/about.html (accessed 12.20.16).

Tobler, W.R., 1970. A Computer Movie Simulating Urban Growth in the Detroit Region. Econ. Geogr. 46, 234. doi:10.2307/143141

Tsujii, J. ’ichi, 2011. Computational linguistics and natural language processing. Presented at the Proceedings of the 12th international conference on Computational linguistics and intelligent text processing - Volume Part I, Springer-Verlag, pp. 52–67.

Wachsmuth, H., 2015. Text Analysis Pipelines: Towards Ad-hoc Large-Scale Text Mining, 1st ed. 2015. ed. Cham, Springer International Publishing, Imprint: Springer.

Wang, C., Li, X., Zhang, Y., Xu, Q., Huang, F., Cao, K., Tao, L., Guo, J., Gao, Q., Wang, W., Fang, L., Guo, X., 2016. Spatiotemporal Cluster Patterns of Hand, Foot, and Mouth Disease at the County Level in Mainland China, 2008-2012. PLOS ONE 11, e0147532. doi:10.1371/journal.pone.0147532

Wang, H., Hovy, E., Dredze, M., 2015. The Hurricane Sandy Twitter Corpus, in: AAAI Workshop on the World Wide Web and Public Health Intelligence.

Wang, X., McCallum, A., Wei, X., 2007. Topical N-Grams: Phrase and Topic Discovery, with an Application to Information Retrieval. IEEE, pp. 697–702. doi:10.1109/ICDM.2007.86

Weiss, S.M., Indurkhya, N., Zhang, T., Damerau, F.J., 2005. Text Mining. Springer New York, New York, NY.

Westerholt, R., Steiger, E., Resch, B., Zipf, A., 2016. Abundant Topological Outliers in Social Media Data and Their Effect on Spatial Analysis. PLOS ONE 11, e0162360. doi:10.1371/journal.pone.0162360

Wikipedia - Information retrieval, 2016. . Wikipedia.

Wikipedia - Text mining, 2016. . Wikipedia.

Witten, I.H., Frank, E., Hall, M.A., 2011. Data mining: practical machine learning tools and techniques, 3rd ed. ed, Morgan Kaufmann series in data management systems. Morgan Kaufmann, Burlington, MA.

Wooldridge, J.M., 2009. Introductory econometrics: a modern approach, 4th ed. ed. South Western, Cengage Learning, Mason, OH.

Yang, J., He, H., Shifley, S., Gustafson, E., 2007. Spatial Patterns of Modern Period Human-Caused Fire Occurrence in the Missouri Ozark Highlands. For. Sci. 53, 1–15.

Yates, D., Paquette, S., 2011. Emergency knowledge management and social media technologies: A case study of the 2010 Haitian earthquake. Int. J. Inf. Manag. 31, 6–13. doi:10.1016/j.ijinfomgt.2010.10.001

Zhang, S., Tang, J., Wang, H., Wang, Y., 2015. Enhancing Traffic Incident Detection by Using Spatial Point Pattern Analysis on Social Media. Transp. Res. Rec. J. Transp. Res. Board 2528, 69–77. doi:10.3141/2528-08

7 Appendix

7.1 Software used

The following software was used for this thesis:

Operating system: Windows 10

Data analysis: R 3.3.1, RStudio 0.99.902

R packages:

Database: PostgreSQL 9.5 with PostGIS

GIS: ArcMap 10.4, QGIS 2.14

Figures: R, Microsoft Powerpoint

Bibliography management: Zotero 4.0.29.10

R-Script

7.2 R-Scripts repertory

7.2.1 R-Script Import Twitter

```
#####################################################################
### MT-Elbe2013-01-Data /// DataMining
### ELBE 2013 Hochwasser Harvard
### BB Germany: 5.23, 47.11, 15.18, 55.02
### Lat Long (51.011811, 13.738403) Dresden
### Test einlesen der Daten

.libPaths("C:/R-packages")

#####################################################################
# Phase 01 : Import der Daten
#####################################################################
library(maps)       # Provides functions that let us plot the maps
library(mapdata)    # Contains the hi-resolution points that mark out the countries.
library(ggplot2)
getwd()
setwd("E:/Elbe2013/")
#"C:/R-packages"

#scan(file ="geo_tweets_2013_06_13_23.csv", what = "character", quiet = FALSE)
# test
#data <- read.csv(file="geo_tweets_2013_05_27_00.csv", header = FALSE, sep=",")
# Einlesen aller csv
#
filenames <- list.files(path=getwd())
numfiles <- length(filenames)
# zeile mit Dateinamen anf?gen?
for (i in filenames){
  data <-  read.csv(file=i, header = FALSE, sep=",")
  data2 <- rbind(data2, data)
}

elbe2013 <- data2

# Archiv
# save(elbe2013, file = "tw_elbe2013.rda")
# write.csv(data2, file = "elbe2013.csv")

# Prüfe auf unique
#duplicated(data2)
data2 <- unique(elbe2013)

#Daten bereinigen und
setwd("C:/#Daten_R/")
load(file = "tw_elbe2013.rda",envir = parent.frame())
# header einf?gen
names(elbe2013_saschsen) <- c("V1", "date", "latitude", "longitude", "V5", "V6", "V7", "screenName", "statusSource", "V10", "V11", "V12",
"text")

elbe2013 <- elbe2013[ -c(1, 5:7,10:12) ]
save(elbe2013, file = "tw_elbe2013ok.rda")


################################################################################################### Grenzen
```

```
library(GISTools)
library(sp)
library(maptools)
library(maps)
library(ggplot2)
library(ggmap)

setwd("C:/#Daten_R/")
setwd("C://#rstudio/Elbe2013/")
setwd("C://#rstudio/Elbe2013/DEU_adm_shp/")

nc0 <- readShapePoly("DEU_adm0.shp",proj4string=CRS("+proj=longlat +datum=NAD27")) #Deutschland
nc1 <- readShapePoly("DEU_adm1.shp",proj4string=CRS("+proj=longlat +datum=NAD27")) #Länder
nc2 <- readShapePoly("DEU_adm2.shp",proj4string=CRS("+proj=longlat +datum=NAD27")) #Landkreise
nc3 <- readShapePoly("DEU_adm3.shp",proj4string=CRS("+proj=longlat +datum=NAD27")) #PLZ

#Tweets aus Deutschland mit QGIS
elbe2013ok <- readShapePoints("tw_elbe2013ok.shp",proj4string=CRS("+proj=longlat"))
elbe2013ok <- data.frame(elbe2013ok)


elbe2013ok_elbanrei <- readShapePoints("tw_elbe2013ok_elbeanreihner.shp",proj4string=CRS("+proj=longlat"))
#elbe2013ok_elbanrei_owin <- readShapePoly("tw_elbe2013ok_elbeanreihner.shp",proj4string=CRS("+proj=longlat"))
elbe2013_de_elban <- data.frame(elbe2013ok_ealbanrei)
#poly.counts(pts, polys)


elbe2013_sachsen <- readShapePoints("tw_elbe2013_sachsen.shp",proj4string=CRS("+proj=longlat"))
elbe2013_sachsen <- data.frame(elbe2013_sachsen)


# col_no comes from the calculations above
par(mar=c(0,0,0,0))
plot(nc3, col=NA, border=grey(.9), lwd=.5)
plot(nc2, col=NA, border=grey(.9), lwd=.5)
# Land
plot(nc0, col=NA, border=grey(.2), lwd=1)
# Länder
plot(nc1, col=NA, border=grey(.5), lwd=1, add=TRUE)

plot(nc0, col=NA, border=grey(.2), lwd=1, add=TRUE)

plot(nc0, col=NA, border=grey(.2), lwd=1)
plot(elbe2013_de_elban, col=NA, border=grey(.5), lwd=1, add=TRUE)

plot(nc2, col=NA, border=grey(.5), lwd=1)

index <- nc1$NAME_1 =="Sachsen"
sachsen <- nc1[index,]
plot(test)



summary(elbe2013ok)

#proj4string(elbe2013) = CRS("+proj=longlat +datum=NAD27 +ellps=clrk66 +nadgrids=@conus,@alaska,@ntv2_0.gsb,@ntv1_can.dat")
#spTransform
#proj4string(nc0) = CRS("+proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0")
# nur Tweets in Deutschland
plot(nc0, col=NA, border=grey(.2), lwd=1, axes=TRUE)
plot(elbe2013ok, add = T, pch = 1, col = "#FB6A4A4C", cex = 0.4)
plot(nc0, col=NA, border=grey(.2), lwd=1, add=TRUE)
plot(nc0, col=NA, border=grey(.2), lwd=1)
plot(elbe2013_de_regen, col=NA, border=grey(.5), lwd=1, add=TRUE)

ggsave("plot_elbe_2013_DEU.png", width = 5, height = 5)

heatmap(elbe2013ok)


######################### filter
library(sqldf)
# sqldf('SELECT *
#       FROM df
#       WHERE v1 < 0.5 OR v2 = "g"')


elbe2013_de_hochwasser <- sqldf("SELECT * FROM elbe2013_de WHERE lower(text) LIKE '%hochwasser%'", user="postgres", password = "admin",
host = "localhost", port=5432, dbname="test_sqldf")
elbe2013_de_regen <- sqldf("SELECT * FROM elbe2013_de WHERE lower(text) LIKE '%regen%'", user="postgres", password = "admin", host =
"localhost", port=5432, dbname="test_sqldf")
elbe2013_de_elban_hw <- sqldf("SELECT * FROM elbe2013_de_elban WHERE lower(text) LIKE '%hochwasser%'", user="postgres", password = "admin",
host = "localhost", port=5432, dbname="test_sqldf")
#map(data3)
# stop 29.10.


###################################################################### StatusSource screenname

elbe2013_de_hochwasser$statusSource = substr(elbe2013_de_hochwasser$statusSource, regexpr('>', elbe2013_de_hochwasser$statusSource) + 1,
regexpr('</a>', elbe2013_de_hochwasser$statusSource) - 1)
tmp1 = sort(table(elbe2013_de_hochwasser$statusSource), decreasing = T)
dotchart((tmp1[5:1])):mtext('Number of tweets posted by statussource')

tmp = sort(table(elbe2013_de_hochwasser$screenName), decreasing = T)
dotchart((tmp[10:1])):mtext('Number of tweets posted by screenname')
tmp [50:1]

## screenname
```

```
hist(aggregate(elbe2013_de_hochwasser$screenName))
#
tmp = sort(table(elbe2013_de_hochwasser$screenName), decreasing = T)
dotchart((tmp[50:1])):mtext('Number of tweets posted by screenname')
tmp [50:1]
hist(aggregate(elbe2013_de_hochwasser$screenName))



################################################################### Bounding Box erstellen

library(sp)
e <- as(raster::extent(78.46801, 78.83157, 19.53407, 19.74557), "SpatialPolygons")
proj4string(e) <- "+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"
plot(e)
```

7.2.2 R-Script Text minig

```
#####################################################################
### MT-Elbe2013-01-TM1 /// Text Mining
### ELBE 2013 Hochwasser Harvard
### BB Germany: 5.23, 47.11, 15.18, 55.02
### Lat Long (51.011811, 13.738403) Dresden ca.geographische Breite,geographische Länge
### Test einlesen der Daten
.libPaths("C:/R-packages")
# Library
lapply(c('twitteR', 'dplyr', 'ggplot2', 'lubridate', 'network', 'sna', 'qdap', 'tm'), library, character.only = TRUE)
library(tm)
library(wordcloud)
library(ggplot2)
library(stringr)
##############################################################################
# Get the data
setwd("C:/#rstudio/Elbe2013/")
#load("",envir = parent.frame(), verbose = FALSE)

# theme_set(new = theme_bw())
# source('../../R/twitterAuth.R')
# set.seed(95616)

d <- elbe2013_sachsen
dat <- head(elbe2013_de, 25000)

##############################################################################
# What platforms apple/android are people using?
 # head(dnow,4)
par(mar = c(3, 3, 3, 2))
d$statusSource = substr(d$statusSource, regexpr('>', d$statusSource) + 1, regexpr('</a>', d$statusSource) - 1)

tmp = sort(table(d$statusSource), decreasing = T)
dotchart(tmp[5:1])
tmp [5:1]

dotchart(sort(table(d$statusSource)))
mtext('Number of tweets posted by platform')
barplot(tmp[1:5], main="Counts")


###############################################################################
# kill bloede Sonderzeichen
rm(m)
#non ascii weg
d$text <- gsub("[^\x20-\x7E]", "", d$text)
typeof(d$text)
d$text <- as.String(d$text)
d$text <- gsub(" ?(f|ht)(tp)(s?)(://)(.*)[.|/](.*)", "", d$text)
d$text <- gsub("<U+FFFD>", "", d$text)
#d$text <- gsub("(f|h")tp(s?)://(.*)[.][a-z]+", "", d$text)
d$text <- gsub("<.*>", "", d$text)
d$text <- gsub("<.*>", "", d$text)

d$text <- str_replace(d$text, "<.*>", "")
d$text <- str_replace(d$text, "<U+FFFD>", "")


corp <- Corpus(VectorSource(d$text))

#####################################################################################
library(stringi)
d$text <- gsub(" ?(f|ht)(tp)(s?)(://)(.*)[.|/](.*)", "", d$text)
d$text <- iconv(d$text, from = "UTF-8", to = "ASCII", sub = "")

dim(d)
# build a corpus, and specify the source to be character vectors
corp <- Corpus(VectorSource(d$text))
# convert to lower case # myCorpus <- tm_map(myCorpus, tolower)
# tm v0.6
myCorpus <- tm_map(corp, content_transformer(tolower))
# remove punctuation
myCorpus <- tm_map(myCorpus, removePunctuation)
# remove numbers
myCorpus <- tm_map(myCorpus, removeNumbers)
# remove URLs
removeURL <- function(x) gsub("http[[:alnum:]]*", "", x)
myCorpus <- tm_map(myCorpus, content_transformer(removeURL))  #??
myStopwords <- c(stopwords("german"))  #, "rt", "via")
```

```
# remove 'r' and 'big' from stopwords
#myStopwords <- setdiff(myStopwords, c("r", "big"))
# remove stopwords from corpus
myCorpus <- tm_map(myCorpus, removeWords, myStopwords)
#
#※# keep a copy of corpus to use later as a dictionary for stem
# completion
myCorpusCopy <- myCorpus
myCorpus <- myCorpusCopy
# stem words
myCorpus <- tm_map(myCorpus, stemDocument)

# inspect the first 5 documents (tweets) inspect(myCorpus[1:5])
# The code below is used for to make text fit for paper width
for (i in 1:5) {
  cat(paste("[[", i, "]] ", sep = ""))
  #writeLines(myCorpus[[i]])
  writeLines(as.character(myCorpus[[i]]))
}

tdm <- TermDocumentMatrix(myCorpus, control = list(wordLengths = c(1, Inf)))
tdm


idx <- which(dimnames(tdm)$Terms == "hochwasser")
inspect(tdm[idx + (0:5), 10:110])
#inspect frequent words
(freq.terms <- findFreqTerms(tdm, lowfreq=10))
library(ggplot2)

term.freq <- rowSums(as.matrix(tdm))
term.freq <- subset(term.freq, term.freq >=100)
df <- data.frame(term = names(term.freq), freq = term.freq)

df <- arrange(df, freq)
df <- df[order(df$freq, decreasing = TRUE),]
df

df <- df[1:10,] # index
head(df, 10)
library(dplyr)
ggplot(df, aes(x=term, y=freq))  + geom_bar(stat = "identity") + xlab("Terms") + ylab("Count") + coord_flip()
ggplot(df, aes(x=term, y=freq)) + geom_bar(stat = "identity") + xlab("Terms") + ylab("Count") + coord_flip()


######################################################################
# Phase 0  : Wordanalyse
######################################################################
##### s?ubern
library(stringi)
d <- head(elbe2013_de, 5000)
names(d)[5] <- "statusSource"

### statusSource
elbe2013_de$statusSource = substr(elbe2013_de$statusSource, regexpr('>', elbe2013_de$statusSource) + 1, regexpr('</a>', elbe2013_de$sta-
tusSource) - 1)
d$statusSource = substr(d$statusSource, regexpr('>', d$statusSource) + 1, regexpr('</a>', d$statusSource) - 1)
tmp = sort(table(d$statusSource), decreasing = T)
dotchart((tmp[15:1])):mtext('Number of tweets posted by platform')
tmp [5:1]

### text
d$text <- gsub(" ?(f|ht)(tp)(s?)(://)(.*)[.|/](.*)", "", d$text)
d$text <- iconv(d$text, from = "UTF-8", to = "ASCII", sub = "")

### Corpus bilden
lapply(c('twitteR', 'dplyr', 'ggplot2', 'lubridate', 'network', 'sna', 'qdap', 'tm'), library, character.only = TRUE)
library(tm)
library(wordcloud)

# build a corpus, and specify the source to be character vectors
myCorpus_elbe <- Corpus(VectorSource(d$text))
# convert to lower case # myCorpus <- tm_map(myCorpus, tolower)
# tm v0.6
myCorpus_elbe <- tm_map(myCorpus_elbe, content_transformer(tolower))
# remove punctuation
myCorpus_elbe <- tm_map(myCorpus_elbe, removePunctuation)
# remove numbers
myCorpus_elbe <- tm_map(myCorpus_elbe, removeNumbers)

# remove URLs
removeURL <- function(x) gsub("http[[:alnum:]]*", "", x)
### myCorpus <- tm_map(myCorpus, removeURL, lazy=TRUE)
myCorpus_elbe <- tm_map(myCorpus_elbe, content_transformer(removeURL))  #??
# add two extra stop words: 'available' and 'via'
myStopwords <- c(stopwords("english"))
myStopwords <- c(stopwords("german"))

# remove stopwords from corpus
myCorpus_elbe <- tm_map(myCorpus_elbe, removeWords, myStopwords)
myCorpusCopy_elbe <- myCorpus_elbe
# stem words
myCorpus_elbe <- tm_map(myCorpus_elbe, stemDocument)

# inspect the first 5 documents (tweets) inspect(myCorpus[1:5])
# The code below is used for to make text fit for paper width
for (i in 1:5) {
  cat(paste("[[", i, "]] ", sep = ""))
  #writeLines(myCorpus[[i]])
  writeLines(as.character(myCorpus_elbe[[i]]))
}

tdm_elbe <- TermDocumentMatrix(myCorpus_elbe, control = list(wordLengths = c(1, Inf)))
tdm_elbe

idx <- which(dimnames(tdm_elbe)$Terms == "rain")
inspect(tdm_elbe[idx + (0:5), 101:110])
#inspect frequent words
(freq.terms <- findFreqTerms(tdm_elbe, lowfreq=5))
library(ggplot2)

term.freq <- rowSums(as.matrix(tdm_elbe))
term.freq <- subset(term.freq, term.freq >=100)
df <- data.frame(term = names(term.freq), freq = term.freq)

df <- arrange(df, freq)
df <- df[order(df$freq, decreasing = TRUE),]
df

#df <- df[1:10,] # index
#head(df, 10)
#library(dplyr)
ggplot(df, aes(x=term, y=freq))  + geom_bar(stat = "identity") + xlab("Terms") + ylab("Count") + coord_flip()



#####################################################################
# Phase 0  : wordcloud
#####################################################################
#### wordcloud

library(tm)
library(SnowballC)
library(wordcloud)

wordcloud(myCorpus_elbe, max.words = 20, random.order = TRUE)


#############################################################################################################################
#####################################################################
# Phase 0  : Wordanalyse
#####################################################################
##### s?ubern
library(stringi)
d <- head(elbe2013_de, 5000)
names(d)[5] <- "statusSource"

### statusSource
elbe2013_de$statusSource = substr(elbe2013_de$statusSource, regexpr('>', elbe2013_de$statusSource) + 1, regexpr('</a>', elbe2013_de$sta-
tusSource) - 1)
d$statusSource = substr(d$statusSource, regexpr('>', d$statusSource) + 1, regexpr('</a>', d$statusSource) - 1)
tmp = sort(table(d$statusSource), decreasing = T)
dotchart((tmp[15:1])):mtext('Number of tweets posted by platform')
tmp [5:1]

### text
d$text <- gsub(" ?(f|ht)(tp)(s?)(://)(.*)[.|/](.*)", "", d$text)
d$text <- iconv(d$text, from = "UTF-8", to = "ASCII", sub = "")

### Corpus bilden
lapply(c('twitteR', 'dplyr', 'ggplot2', 'lubridate', 'network', 'sna', 'qdap', 'tm'), library, character.only = TRUE)
library(tm)
library(wordcloud)

# build a corpus, and specify the source to be character vectors
myCorpus_elbe <- Corpus(VectorSource(d$text))
# convert to lower case # myCorpus <- tm_map(myCorpus, tolower)
# tm v0.6
myCorpus_elbe <- tm_map(myCorpus_elbe, content_transformer(tolower))
# remove punctuation
myCorpus_elbe <- tm_map(myCorpus_elbe, removePunctuation)
# remove numbers
myCorpus_elbe <- tm_map(myCorpus_elbe, removeNumbers)
# remove URLs
removeURL <- function(x) gsub("http[[:alnum:]]*", "", x)
myCorpus_elbe <- tm_map(myCorpus_elbe, content_transformer(removeURL))  #??
# add two extra stop words:
myStopwords <- c(stopwords("english"))
myStopwords <- c(stopwords("german"))

# remove stopwords from corpus
myCorpus_elbe <- tm_map(myCorpus_elbe, removeWords, myStopwords)
#
#???# keep a copy of corpus to use later as a dictionary for stem
# completion
myCorpusCopy_elbe <- myCorpus_elbe
# stem words
myCorpus_elbe <- tm_map(myCorpus_elbe, stemDocument)

# inspect the first 5 documents (tweets) inspect(myCorpus[1:5])
# The code below is used for to make text fit for paper width
for (i in 1:5) {
  cat(paste("[[", i, "]] ", sep = ""))
  #writeLines(myCorpus[[i]])
  writeLines(as.character(myCorpus_elbe[[i]]))
}

tdm_elbe <- TermDocumentMatrix(myCorpus_elbe, control = list(wordLengths = c(1, Inf)))
```

```
tdm_elbe

idx <- which(dimnames(tdm_elbe)$Terms == "rain")
inspect(tdm_elbe[idx + (0:5), 101:110])
#inspect frequent words
(freq.terms <- findFreqTerms(tdm_elbe, lowfreq=5))
library(ggplot2)

# objects()
# ls()
# help("memory.size")
# memory.size()
# memory.limit()

term.freq <- rowSums(as.matrix(tdm_elbe))
term.freq <- subset(term.freq, term.freq >=100)
df <- data.frame(term = names(term.freq), freq = term.freq)

df <- arrange(df, freq)
df <- df[order(df$freq, decreasing = TRUE),]
df

#df <- df[1:10,] # index
#head(df, 10)
#library(dplyr)
ggplot(df, aes(x=term, y=freq))  + geom_bar(stat = "identity") + xlab("Terms") + ylab("Count") + coord_flip()



######################################################################
# Phase 0  : wordcloud
######################################################################
#### wordcloud

library(tm)
library(SnowballC)
library(wordcloud)

wordcloud(myCorpus_elbe, max.words = 20, random.order = TRUE)
```

7.2.3 R-Script LDA / LDAviz

```
### --------------------------------------------------------------------------------------- LDA with LDAviz

.libPaths("C:/R-packages")
.libPaths()
library(tm)
library(topicmodels)

doc_tweets <- tw_miami_flood$text
doc_tweets <- tw_miami02
doc_tweets <- head(tw_miami$text, 5000)

#create corpus from vector
docs <- Corpus(VectorSource(doc_tweets))
#inspect a particular document in corpus
writeLines(as.character(docs[[30]]))

#start preprocessing
#Transform to lower case
docs <-tm_map(docs,content_transformer(tolower))

toSpace <- content_transformer(function(x, pattern) { return (gsub(pattern, " ", x))})
docs <- tm_map(docs, toSpace, "-")
docs <- tm_map(docs, toSpace, "’")
docs <- tm_map(docs, toSpace, "‘")
docs <- tm_map(docs, toSpace, "•")
docs <- tm_map(docs, toSpace, "\"")
docs <- tm_map(docs, toSpace, "\"")

docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, removeWords, stopwords("english"))
docs <- tm_map(docs, stripWhitespace)
docs <- tm_map(docs,stemDocument)

myStopwords <- c("can", "say","one","way","use")
docs <- tm_map(docs, removeWords, myStopwords)

dtm <- DocumentTermMatrix(docs)
freq <- colSums(as.matrix(dtm))
length(freq)
ord <- order(freq,decreasing=TRUE)
freq[ord]
write.csv(freq[ord],"word_freq.csv")

#parameters Gibbs sampling
burnin <- 4000
iter <- 2000
thin <- 500
seed <-list(2003,5,63,100001,765)
nstart <- 5
best <- TRUE

#Number of topics
```

```
k <- 20
ldaOut <-LDA(dtm, k, method="Gibbs", control=list(nstart=nstart, seed = seed, best = best, burnin = burnin, iter = iter, thin=thin))

ldaOut.topics <- as.matrix(topics(ldaOut))
#top 6 terms in each topic
ldaOut.terms <- as.matrix(terms(ldaOut,8))
topicProbabilities <- as.data.frame(ldaOut@gamma)
ldaOut
ldaOut.terms
ldaOut.topics
```

7.2.4 R-Script Gauge Dresden

```
#update.packages()
.libPaths("C:/R-packages")
.libPaths()
library(ggplot2)
### ---------------------------------------------------------------------------------------------------- gauge data "Pgel Dresden"
pegel_dresden <- readClipboard(sep='\t')
pegel_dresden <- read.table(file = "clipboard", sep = "\t", header=TRUE)
pd <- pegel_dresden

#rm(pd)
#pd$date <- as.Date(pd$date)
#pd$date <- format(pd$date, format="%Y-%m-%d")
#test_tw$date2 = strptime(test_tw$date2,format='%d-%b-%Y')
test_tw$date2 <- as.Date(test_tw$date2)

ggplot(pegel_dresden, aes(date, alarm_level)) +
  geom_point() +
  xlab("date") + ylab("alarm level") + ggtitle("alarm level of gauge Dresden") +
  geom_point(data = pegel_dresden, aes(y = alarm_level), colour = 'red', size = 3) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

# ggplot(test_tw, aes(date, al)) +
#   geom_point() +
#   xlab("alarm level") + ylab("date") + ggtitle("alarm level  ") +
#   geom_point(data = pegel_dresden, aes(y = alarm_level), colour = 'red', size = 3) +
#   theme(axis.text.x = element_text(angle = 90, hjust = 1))

### ------------ prepare the test tweets
test_tw <- elbe2013_sachsen
test_tw

#elbe2013_dresden_hochwasser
tw_DD_m <- elbe2013_dresden_hochwasser

### ---- fix the coordinates to UTM

lat = elbe2013_dresden_hochwasser$latitude
lon = elbe2013_dresden_hochwasser$longitude
xy = data.frame(lon, lat)
coordinates(xy) <- c("lon", "lat")
proj4string(xy) <- CRS("+proj=longlat +datum=WGS84")
NE <- spTransform(xy, CRS("+proj=utm +zone=33U ellps=WGS84"))
NE <- as.data.frame(NE)
elbe2013_dresden_hochwasser$UTM_x  <- NE$lon
elbe2013_dresden_hochwasser$UTM_y  <- NE$lat

### ---- fix the format of date
library(lubridate)

test_tw$date2 <- test_tw$date
test_tw$date2 <- as.Date(test_tw$date2)
class(test_tw$date2)

tw_DD_m$date2 <- tw_DD_m$date
tw_DD_m$date2 <- as.Date(tw_DD_m$date2)
class(tw_DD_m$date2)
summary(tw_DD_m)
#test_tw$date2 = strptime(test_tw$date2,format='%d-%b-%Y')
#test_tw$date2 <- format(test_tw$date2, format="%d-%M-%Y")
#test_tw$date2 <- format(test_tw$date2, format="%d.%m.%Y")
summary(test_tw)
##pegel_dresden$date <- as.Date(pegel_dresden$date,"%Y-%m-%d" )

test_tw$al <- NA


### ---------------------------------------------------------------tweet df alarm leven over date - funzt

days <- seq(from=as.Date('2013-05-27'), to=as.Date("2013-06-13"),by='days')
for ( i in seq_along(days) )
{
  print(i)
  as.character(days[i])
  test_tw$al[which(test_tw$date2 == as.character(days[i]))] <- pd$alarm_level[which(pd$date == as.character(days[i]))]
  #print(test_tw$date2 == i)
 #test_tw[test_tw$date==i,]$al <- pd[i]$alarm_level
  print(days[i])
}

### ----------------------------------------------------------------------------------
### ---------------------------------------------------------------tweet df alarm leven over date - funzt
tw_DD_m$al <- NA

days <- seq(from=as.Date('2013-05-27'), to=as.Date("2013-06-13"),by='days')
for ( i in seq_along(days) )
{
  print(i)
  as.character(days[i])
  tw_DD_m$al[which(tw_DD_m$date2 == as.character(days[i]))] <- pd$alarm_level[which(pd$date == as.character(days[i]))]
  #print(test_tw$date2 == i)
  #test_tw[test_tw$date==i,]$al <- pd[i]$alarm_level
  print(days[i])
}

tw_DD_m$al <- as.factor(tw_DD_m$al)

### ---------------------------------------------------------------------------------
#elbe2013_SN_hwr_backup <- elbe2013_SN_hwr

elbe2013_SN_hwr$date2 <- elbe2013_SN_hwr$date
elbe2013_SN_hwr$date2 <- as.Date(elbe2013_SN_hwr$date2)
elbe2013_SN_hwr$al <- NA

days <- seq(from=as.Date('2013-05-27'), to=as.Date("2013-06-13"),by='days')
for ( i in seq_along(days) )
{
  print(i)
  as.character(days[i])
  elbe2013_SN_hwr$al[which(elbe2013_SN_hwr$date2 == as.character(days[i]))] <- pd$alarm_level[which(pd$date == as.character(days[i]))]
  #print(test_tw$date2 == i)
  #test_tw[test_tw$date==i,]$al <- pd[i]$alarm_level
  print(days[i])
}

elbe2013_SN_hwr$al <- as.factor(elbe2013_SN_hwr$al)
summary(elbe2013_SN_hwr)
#test_tw$al[which(test_tw$date2 == '2013-06-01')] <- 1

### ------------------------ alarm_level füllen covariate

#http://stackoverflow.com/questions/21712384/updating-column-in-one-dataframe-with-value-from-another-dataframe-based-on-matc
### ---------------------------------------------------------------------------------
#df[df=="" | df==12] <- NA
#df$JoiningDate <- as.Date(df$JoiningDate , "%m/%d/%y")
#x.sub <- subset(x.df, y > 2)


### --- Idee aus schleige day als covariante
ggplot(test_tw, aes(date, al)) +
  geom_point() +
  geom_point(data = test_tw, aes(y = al), colour = 'red', size = 3)

hist(test_tw$al)
```

7.2.5 R-Script Statistical analysis

```
#####################################################################
### ELBE 2013 Hochwasser Harvard
### BB Germany: 5.23, 47.11, 15.18, 55.02
### Lat Long (51.011811, 13.738403) Dresden ca.geographische Breite,geographische Länge
### Test einlesen der Daten
#LAT- Latitude = -90° - 90°
#LONG- Longitude = -180° - 180°

#http://tinyheero.github.io/2015/09/15/semi-transparency-r.html
#install.packages("Cairo")
# setHook(packageEvent("grDevices", "onLoad"),
#         function(...) grDevices::X11.options(type='cairo'))
# options(device='x11')

.libPaths("C:/R-packages")
library(spatstat)
library(mapproj)
library(maptools)
library(ggmap)
library(sp)
library(splines)
library(raster)
library(sqldf)

### data preparation

### elbe2013 contains the whole gathered data from Harvard
### elbe2013_de contains the extrat in germany
### UTM-coordinates (WGS84) of Dresden are: Zone 33U E: 411491.73 N: 5656189.12
rm(tw_DD1)
as.data.frame(tw_DD)
#tw_DD <- elbe2013_dresden[,c(1-5)]
tw_DD <- elbe2013_dresden_hochwasser
tw_DD <- tw_DD[, -(7:10)]
edit(bbox_DD)

### ----- transform from latlon to UTM Zone 33 for Dresden

# check bbox from Dresden
# 13.581333,50.967454,13.900967,51.146671
```

```
#bbox_DD <- data.frame(x=c(13.581333, 13.900967), y=c(50.967454, 51.146671))
bbox_DD$UTM_x[1] <- 400404.8
bbox_DD$UTM_x[2] <- 423145.3
bbox_DD$UTM_y[1] <- 5645095
bbox_DD$UTM_y[2] <- 5668739

#(400404.8, 5645095, 423145.3, 5668739)
fix(bbox_DD)

## 11.8542,50.1646,15.0623,51.6947
bbox_DD
#          x        y    UTM_x   UTM_y
# 1 13.58133 50.96745 400385.2 5647164
# 2 13.90097 51.14667 423125.7 5666710

lat = bbox_DD$y
lon = bbox_DD$x
xy = data.frame(lon, lat)
coordinates(xy) <- c("lon", "lat")
proj4string(xy) <- CRS("+proj=longlat +datum=WGS84")
NE <- spTransform(xy, CRS("+proj=utm +zone=33U ellps=WGS84"))
NE <- as.data.frame(NE)
bbox_DD$UTM_x  <- NE$lon
bbox_DD$UTM_y  <- NE$lat


lat = tw_DD$latitude
lon = tw_DD$longitude
xy = data.frame(lon, lat)
coordinates(xy) <- c("lon", "lat")
proj4string(xy) <- CRS("+proj=longlat +datum=WGS84")
NE <- spTransform(xy, CRS("+proj=utm +zone=33U ellps=WGS84"))
NE <- as.data.frame(NE)
tw_DD$UTM_x  <- NE$lon
tw_DD$UTM_y  <- NE$lat

### ------------------------------------------------------------------------------------------ Sptatial Analysis elbde_de_hochwasser
### --- Inspecting data
PP_tw_DD <- as.vector(tw_DD[, c(7,8)])
names(PP_tw_DD) <- c("x","y")
PP_tw_DD <- unique(PP_tw_DD)
summary(PP_tw_DD)

w_tw_DD <- owin(c(bbox_DD$UTM_x[1],bbox_DD$UTM_x[2]), c(bbox_DD$UTM_y[1],bbox_DD$UTM_y[2]), unitname=c("meter","meter"))
myspp_tw_DD <- ppp(PP_tw_DD$x, PP_tw_DD$y, window = w_tw_DD)
unique(myspp_tw_DD)
unitname(myspp_tw_DD)
plot(myspp_tw_DD, axes = TRUE )

##### ----
Q <- quadratcount(myspp_tw_DD, nx = 10, ny = 8)
plot(myspp_tw_DD )
plot(Q, add = TRUE, cex =1, pch ="+", col = "blue")
plot(density(myspp_tw_DD, 750))
contour(density(myspp_tw_DD, 750))

# perspective
persp(density(myspp_tw_DD, 500))
summary(myspp_tw_DD)
contour(distmap(myspp_tw_DD))

### -------------------------------- intensity

par(mfrow=c(2,1))
plot(density.ppp(myspp_tw_DD, sigma = bw.diggle(myspp_tw_DD),edge=T),main=paste("sigma(Diggle) =",round(bw.diggle(myspp_tw_DD),2)))
plot(density.ppp(myspp_tw_DD, sigma = bw.ppl(myspp_tw_DD),edge=T),main=paste("sigma(LikeCross) =",round(bw.ppl(myspp_tw_DD),2)))
#plot(density.ppp(myspp_tw_DD, sigma = bw.scott(myspp_tw_DD)[2],edge=T),main=paste("sigma(Scott) =",round(bw.scott(myspp_tw_DD)[2],2)))
plot(density.ppp(myspp_tw_DD, sigma = bw.scott(myspp_tw_DD)[1],edge=T),main=paste("sigma(Scott) =",round(bw.scott(myspp_tw_DD)[1],2)))
par(mfrow=c(1,1))

par(mfrow=c(2,2))
plot(density.ppp(myspp_tw_DD, sigma = 50))
plot(density.ppp(myspp_tw_DD, sigma = 250))
plot(density.ppp(myspp_tw_DD, sigma = 750))
plot(density.ppp(myspp_tw_DD, sigma = 1000))
par(mfrow=c(1,1))


plot(density.ppp(myspp_tw_DD))
plot(Kinhom(myspp_tw_DD))
plot(envelope(myspp_tw_DD, Kinhom))


### -------------------------------- Complete Spatial Randomness CHI quad

quadrat.test(myspp_tw_DD, nx = 5, ny = 4)
plot(quadrat.test(myspp_tw_DD, nx = 5, ny = 4))

### -------------------------------- Complete Spatial Randomness Kolmogorov-Smirnov

cdf.test(myspp_tw_DD, "x")
plot(cdf.test(myspp_tw_DD, "x"))

cdf.test(myspp_tw_DD, "y")
plot(cdf.test(myspp_tw_DD, "y"))

### ------------------------------------------------------------------------------  section K-funtion

Gest(myspp_tw_DD)
```

```
plot(Gest(myspp_tw_DD))

tw_DD.k <- envelope(myspp_tw_DD, Kest, nsim = 99, correction = "border")
plot(tw_DD.k)
tw_DD.k

tw_DD.fit <- ppm(myspp_tw_DD, ~ Cov1, covariates = list(Cov1 = landuse))
plot(tw_DD.fit)
point.sim <- simulate(tw_DD.fit, 1)
plot(point.sim)
plot(predict.ppm(tw_DD.fit, type = "lambda"), main = "Predicted Intensity (lambda)") #predicted intensity
tw_DD.kc<- envelope(tw_DD.fit, Kest, nsim = 99, correction = "border", verbose = F)
plot(tw_DD.kc)

#~ marks * polynom(x,2)
#~ marks + marks:polynom(x,2)
tw_DD.fitcov2 <- ppm(myspp_tw_DD ~polynom(x,y,3) + Z, covariates=list(Z=landuse))

### ---------------------------------------------------------------------------------------------- good fitted, good K-function
tw_DD.fitcov2 <- ppm(myspp_tw_DD ~polynom(x,y,3) + Z, covariates=list(Z=landuse))
plot(tw_DD.fitcov2)
point.sim2 <- simulate(tw_DD.fitcov2, 1)
plot(point.sim2)
plot(predict.ppm(tw_DD.fitcov2, type = "lambda"), main = "Predicted Intensity (lambda)") #predicted intensity
tw_DD.kc2<- envelope(tw_DD.fitcov2, Kest, nsim = 99, correction = "border", verbose = F)
plot(tw_DD.kc2)
diagnose.ppm(tw_DD.fitcov2, type="pearson")
p2 <- predict.ppm(tw_DD.fitcov2)
plot(p2)
plot(tw_DD.fitcov2, cif =FALSE, how = "persp")

qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "pearson") # okok
qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "inverse") # okokok

qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "eem") #
qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "raw") #


### --- check it with simulation
point.sim1 <- simulate(tw_DD.fitcov2, 1)
plot(point.sim1)
plot(predict.ppm(tw_DD.fitcov2, type = "lambda"), main = "Predicted Intensity (lambda)") #predicted intensity
tw_DD.kc1<- envelope(tw_DD.fitcov2, Kest, nsim = 99, correction = "border", verbose = F)
plot(tw_DD.kc1)

### ---
plot(Kest(myspp_tw_DD))
plot(Kinhom(myspp_tw_DD))
Kinhom(myspp_tw_DD)

plot(envelope(myspp_tw_DD, Kest))
plot(Kest(myspp_tw_DD))

myspp_tw_DD.K <- envelope(myspp_tw_DD, Kest, nism = 99, correction ="border")
plot(myspp_tw_DD.K)




### -------------------------------- model inhomogeneous Poisson
### -------------------------------- checking a fitted model 111
fit <- ppm(myspp_tw_DD, ~x)
plot(fit, cif =FALSE, how = "persp")
test <- quadrat.test(fit, nx = 5, ny = 4)
test
# X2 = 457.66, df = 18, p-value < 2.2e-16 rejected!!!

# get more information, why the modell is bad

plot(myspp_tw_DD, pch = ".", size = 2)
plot(test, add = TRUE, cex = 1.0, col = "red")

kstest(myspp_tw_DD, "y")

### ---------------------------------------------------------------------------------------------------------- fit xy / Residual Measure

fit <- ppm(myspp_tw_DD, ~x+y)
plot(fit, how = "persp")
plot(predict(fit))
plot(myspp_tw_DD, add =  TRUE, pch="+")

fitx <- ppm(myspp_tw_DD, ~x)
diagnose.ppm(fitx, which = "smooth")
diagnose.ppm(fitx)

#covariate
lurking(fitx, x, type = "raw")


### - search better model ~ polynom(x,2) # ~ bs(x,df=3)
fits <- ppm(myspp_tw_DD, ~ polynom(x,y,3))
fits
plot(fits, cif =FALSE, how = "persp")

library(splines)
fits <- ppm(myspp_tw_DD, ~ bs(x, y,df=3)) # no plot(predict(fits)) spatstat crashes!!!   #B-Spline Basis for Polynomial Splines

fits <- ppm(myspp_tw_DD, ~ x, Strauss(13), correction="periodic")
```

```
plot(predict(fits))
plot(myspp_tw_DD, add =  TRUE, pch="+")
diagnose.ppm(fits, which = "smooth")
diagnose.ppm(fits)

#fits
#coef(fits)
#coef(summary(fits))

# M. Morisita (1959) Measuring of the dispersion of individuals and analysis of the distributional patterns.
miplot(myspp_tw_DD)
fryplot(myspp_tw_DD, width = 5000,  axes = TRUE)

### ------------------------------------------------------------------------------------------------------------ model ok 2512
### ------------------------------------------------------------------------------------------------------------


myspp_tw_DD_re <- rescale(myspp_tw_DD, 1000)

rm(fit)
fit <- ppm(myspp_tw_DD_re, ~polynom(x,y,2), Poisson(),correction = "border" )
plot(fit, cif =FALSE, how = "persp")

fit
plot(fit)

Kcross()

fitA1 <- ppm(myspp_tw_DD_re, ~polynom(x,y,3), Softcore(0.5), correction = "border")


# k_fit <- kppm( myspp_tw_DD, ~polynom(x,y,2), "Thomas")
# k_rr <- residuals(k_fit)
# k_rr
# plot(k_rr)

#ppm( myspp_tw_DD, ~polynom(x,y,3), Poisson())
### change from m to km

par(mfcol=c(2,2))
plot(myspp_tw_DD, main="myspp_tw_DD")
plot(fit)
par(mfcol=c(1,1))
# plot some diagnostics on the fitted model: Pearson residuals (see references)
diagnose.ppm(fit, type="pearson")

qqplot.ppm(fit, nsim = 39)
qqplot.ppm(fit, nsim = 39, type= "pearson") # okok
qqplot.ppm(fit, nsim = 39, type= "inverse") # okokok

qqplot.ppm(fit, nsim = 39, type= "eem") #
qqplot.ppm(fit, nsim = 39, type= "raw") #



#https://codedump.io/share/0mxJ1w6x5DOo/1/cannot-comprehend-this-error-message-in-spatstat-in-r-while-using-kppm-function
myspp_tw_DD <- rescale(myspp_tw_DD, 1000)

kill_me_fit <- kppm(myspp_tw_DD, ~1)
kill_me_fit <- kppm(myspp_tw_DD, ~x+y)
kill_me_fit <- kppm(myspp_tw_DD, ~polynom(x, y, 2))

correction = "border"
kill_me_fit
plot(kill_me_fit)


diagnose.ppm(kill_me_fit, type="pearson")


# Fit model and predict:
spatial.m.1 <- ppm(myspp_tw_DD ~ anthro)
p <- predict.ppm(spatial.m.1)




### ------------------------------------------------------------------------------ distance methods

# pairwie distance
plot(pairdist(myspp_tw_DD))

# nearest neighbour
plot(nndist(myspp_tw_DD))




### ------------------------------------------------------------------------------ empty space distances
plot(distmap(myspp_tw_DD))

# Stienen diagramm
plot(myspp_tw_DD %mark% (nndist(myspp_tw_DD)/2), markscale = 1, main = "Stienen diagram")

# --- Edge Effects F
Fest(myspp_tw_DD, correction = "km")
Fest(myspp_tw_DD, correction = c("km","cs"))
```

```
plot(Fest(myspp_tw_DD))

plot(Fest(myspp_tw_DD), hazard ~ r, main="Hazard rate of F")

# -------------------------------------------------------------------------------- nnd G-Function
Gest(myspp_tw_DD)
plot(Gest(myspp_tw_DD))

plot(G, cbind(km, rs, theo) ~ r)

# -------------------------------------------------------------------------------- pairwise distances K-Function

L <- Lest(myspp_tw_DD)
plot(L, main ="L-function")

plot(pcf(myspp_tw_DD))

plot(allstats(myspp_tw_DD))


# -------------------------------------------------------------------------------- Monte Carlo Tests

E <- envelope(myspp_tw_DD, Kest, nsim = 39, rank =1)
E
plot(E, main = "pointwise envelopes")

E <- envelope(myspp_tw_DD, Kest, nsim = 19, rank =1, global = TRUE)
plot(E, main = "global envelopes")

E <- envelope(myspp_tw_DD, Lest, nsim = 19, rank =1, global = TRUE)
plot(E, main = "global envelopes of L(r)")

### - enveloppes for any fitted model 136

#fit <- ppm(myspp_tw_DD, )


### --- model fitting using summary statistics

# --- fitting a cluster process

fit <- kppm(myspp_tw_DD, ~1, "Thomas")
fit
plot(fit)


# --- fitting Matern

fitM <- kppm(myspp_tw_DD, ~1, "MatClust")
plot(simulate(fit, nsim = 4))

seed("100")

plot(simulate(fit, Lest, nsim = 39))

fitp <- kppm(myspp_tw_DD, ~1, "Thomas", statistic = "pcf")
fitp
plot(fitp)

### -------------------------------- exploring local features // don't work

plot(myspp_tw_DD, pch=".", main="")

# LISA nn feature and noise
Z <- nnclean(myspp_tw_DD, k = 20)
plot(Z, chars =c(".","+"), main = "nearest neigbour cleaning")

### -------------------------------- adjusting for inhomogeneity

fit <- ppm(myspp_tw_DD)
Ki <- Kinhom(myspp_tw_DD)
plot(Ki, main="inhomogeneous K-function")
Ki

#Ki2 <-

Linhom(myspp_tw_DD)
plot(Linhom(myspp_tw_DD))

Ginhom(myspp_tw_DD)
plot(Ginhom(myspp_tw_DD))

pcfinhom(myspp_tw_DD)
plot(pcfinhom(myspp_tw_DD))

# ---------------------------------------------------------------------------------------------------- Raster laden

par(mfrow=c(1,1))
e <- extent(400404.8, 5645095, 423145.3, 5668739)
# bbox_DD$UTM_x[1] <- 400404.8
# bbox_DD$UTM_x[2] <- 423145.3
# bbox_DD$UTM_y[1] <- 5645095
# bbox_DD$UTM_y[2] <- 5668739

# --- Then read in the tiff and crop it
landuse <- as.factor(raster("DDpopdentbbox.tif"))
landuse
landuse <- setMinMax(landuse)
landuse <- crop(landuse, e)
```

```
# --- plot
#w_tw_DD <- owin(c(bbox_DD$UTM_x[1],bbox_DD$UTM_x[2]), c(bbox_DD$UTM_y[1],bbox_DD$UTM_y[2]), unitname=c("meter","meter"))
#myspp_tw_DD <- ppp(PP_tw_DD$x, PP_tw_DD$y, window = w_tw_DD)

plot(landuse, axes = TRUE, legend=TRUE)
points( PP_tw_DD$x, PP_tw_DD$y, col ="white")
plot(w_tw_DD, add = TRUE)

# Now let's convert that raster to an im object for use in spatstat
# (and make it into a factor):
#anthro <- asImRaster(anthro)
landuse <- as.im.RasterLayer(landuse)
landuse <- eval.im(as.factor(landuse))

hist(landuse)
plot(landuse)
#anthrodf <- as.data.frame(anthro)
#rat <- levels(anthro)[[1]]

#levels(landuse) <- c("37.6 bis 100.0", "> 8.3 bis 37.6", "> 4.9 bis 8.3", "> 3.0 bis 4.92", "0.0 bis 3.0")
levels(landuse) <- c(">37.6", ">8.3", ">4.9", ">3.0", ">0.0")
#levels(landuse) <- c("0", "1", "2", "3", "4")

### -------------------------------------------------------------------------------------------------------- model mit covariate
# Fit model and predict:

### --- final
fitcov0 <- ppm(myspp_tw_DD, ~landuse, covariates = list(landuse))
fitcov0
plot(fitcov0, cif =FALSE, how = "persp")
plot(fitcov0)
diagnose.ppm(fitcov0, type="pearson")
p_fitcov0 <- predict.ppm(fitcov0)
plot(p_fitcov0)


fitcov <- ppm(myspp_tw_DD, ~polynom(x,y,3), Poisson())
coef(summary(fitcov))
diagnose.ppm(fitcov, type="pearson")

qqplot.ppm(fitcov, nsim = 39)
qqplot.ppm(fitcov, nsim = 39, type= "pearson") # okok
qqplot.ppm(fitcov, nsim = 39, type= "inverse") # okokok


qqplot.ppm(fitcov, nsim = 39, type= "eem") #
qqplot.ppm(fitcov, nsim = 39, type= "raw") #

p <- predict.ppm(fitcov)
plot(p)

### -------------------------------------------------------------------------------------------------------- Cov1

fit <- ppm(myspp_tw_DD, ~ Cov1, covariates = list(Cov1 = landuse))
plot(fitcov0, cif =FALSE, how = "persp")
k1 <- kppm(myspp_tw_DD ~ x, "MatClust")
plot.kppm(k1)

k2 <- kppm(myspp_tw_DD ~ x, "MatClust", statistic="pcf", statargs=list(stoyan=0.2))
plot.kppm(k2)
k3 <- kppm(myspp_tw_DD ~ x, cluster="Cauchy", statistic="K")
plot.kppm(k3)
k4 <- kppm(myspp_tw_DD, cluster="VarGamma", nu = 0.5, statistic="pcf")
plot.kppm(k4)
## -------------------------------------------------------------------------------------------------------- Cov1

### --- final fitcov1
fitcov1 <- ppm(myspp_tw_DD ~ y + Z, covariates=list(Z=landuse))
coef(summary(fitcov1))
diagnose.ppm(fitcov1, type="pearson")
p_fitcov1 <- predict.ppm(fitcov2)
plot(p_fitcov1)
## -------------------------------------------------------------------------------------------------------- Cov1

### --- final fitcov2
fitcov2 <- ppm(myspp_tw_DD ~polynom(x,y,3) + Z, covariates=list(Z=landuse))
plot(fitcov2, cif =FALSE, how = "persp")
diagnose.ppm(fitcov2, type="pearson")
fitcov2
plot(fitcov2)
diagnose.ppm(fitcov2, type="pearson")
p_fitcov2 <- predict.ppm(fitcov2)
plot(p_fitcov2)
qqplot.ppm(fitcov2, nsim = 39, type= "pearson") # okok
qqplot.ppm(fitcov2, nsim = 39, type= "inverse") # okokok
qqplot.ppm(fitcov2, nsim = 39, type= "eem") #
qqplot.ppm(fitcov2, nsim = 39, type= "raw") #

## -------------------------------------------------------------------------------------------------------- Cov1

### ---- test 29.12.
fitcov3 <- ppm(myspp_tw_DD ~ x+y + landuse)

plot(fitcov3, cif =FALSE, how = "persp")
diagnose.ppm(fitcov3, type="pearson")
p3 <- predict.ppm(fitcov3)
plot(p3)
```

```
# ### ----------------------------- kppm
#
# kfitcov2 <- kppm(myspp_tw_DD ~polynom(x,y,3) + Z, covariates=list(Z=landuse))
# diagnose.kppm(kfitcov2, type="pearson")
# kp2 <- predict.kppm(kfitcov2)
# plot(kp2)
## ---------------------------------------------------------------------------------------------------------- Cov1



# ----- ANOVA Fit
fit0 <- ppm(myspp_tw_DD, ~1, Poisson())
plot(fit0)
fit1 <- ppm(myspp_tw_DD, ~x, Poisson())
plot(fit1)



anova(fit0,fit1)

#
# sims <- simulate(fitcov2, nsim=2*195)
# SIMS <- list()
# for(i in 1:nrow(sims)) SIMS[[i]] <- as.solist(sims[i,,drop=TRUE])
# Hplus <- cbind(H, hyperframe(Sims=SIMS))



#killme <- na.omit(landuse)
##############################################################################################

## good works to
# ---
#mydata.factor2 <- as.factor(raster("DDpopdent2.tif"))

mydata2 <- raster("DDpopdent2.tif")
mydata2 <- crop(mydata2, e)
#mydatasp2 <- rasterToPoints(mydata2)
#mydatadf2 <- as.data.frame(mydatasp2) #ok


mydata2 <- as.im.RasterLayer(mydata2)
mydata2 <- eval.im(as.factor(mydata2))

#levels(mydata2) <- c("37.6 bis 100.0", "> 8.3 bis 37.6", "> 4.9 bis 8.3", "> 3.0 bis 4.92", "0.0 bis 3.0", "NA")
plot(mydata2)

###############################################################################################
### -------------------------------- marked point pattern proces with sentiment, time, warnstufe

#tw_DD_m
# myspp_tw_DD_m <- myspp_tw_DD
# marks(myspp_tw_DD_m) <- tw_DD_m$al
# summary(tw_DD_m) # 187 points
# str(tw_DD_m)

PP_tw_DD_m <- as.vector(tw_DD_m[, c(13, 14, 16)])

names(PP_tw_DD_m) <- c("x","y","AL")
PP_tw_DD_m <- unique(PP_tw_DD_m)
summary(PP_tw_DD_m)
str(PP_tw_DD_m) # 194 points

#w_tw_DD <- owin(c(bbox_DD$UTM_x[1],bbox_DD$UTM_x[2]), c(bbox_DD$UTM_y[1],bbox_DD$UTM_y[2]), unitname=c("meter","meter"))
myspp_tw_DD_m <- ppp(PP_tw_DD_m$x, PP_tw_DD_m$y, marks = (PP_tw_DD_m$AL), window = w_tw_DD)
unique(myspp_tw_DD_m) #178 points
unitname(myspp_tw_DD_m)
plot(myspp_tw_DD_m, axes = TRUE )

myspp_tw_DD_m$marks

fit_m <- ppm(myspp_tw_DD_m, ~x)

plot(alltypes(myspp_tw_DD_m, "G"))
M_m <- marktable(myspp_tw_DD_m, AL = 3)



tw_DD.fitcov2 <- ppm(myspp_tw_DD ~polynom(x,y,3) + Z, covariates=list(Z=landuse))
plot(tw_DD.fitcov2)
point.sim2 <- simulate(tw_DD.fitcov2, 1)
plot(point.sim2)
plot(predict.ppm(tw_DD.fitcov2, type = "lambda"), main = "Predicted Intensity (lambda)") #predicted intensity
tw_DD.kc2<- envelope(tw_DD.fitcov2, Kest, nsim = 99, correction = "border", verbose = F)
plot(tw_DD.kc2)
diagnose.ppm(tw_DD.fitcov2, type="pearson")
p2 <- predict.ppm(tw_DD.fitcov2)
plot(p2)
plot(tw_DD.fitcov2, cif =FALSE, how = "persp")

qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "pearson") # okok
qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "inverse") # okokok

qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "eem") #
qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "raw") #

### -------------------------------- Map DD mit Tweets elbe2013_dresden
hdf <- get_map("Dresden, sachsen")
ggmap(hdf, extent = "normal") + geom_point(aes(x = longitude, y = latitude),colour = "red", size = 1 , data = tw_DD)
```

```
bg_ger <- get_map(location = "germany", zoom = 2, source = "osm")
ggmap(bg_ger, extent = "normal")
ggmap(bg_ger, extent = "normal") + geom_point(aes(x = longitude, y = latitude),colour = "blue", size = .1 , data = elbe2013_de)

#### -------------------------------------------------------------------------------- Model Sachsen
# alarmstufe dazu ok
# sentiment dazu ok
# covariate ok

#landsat.stack2.cropped <- crop(landsat.stack2, landsat.stack.cropped.pol)
mydataSN <- raster("SNlanduseok.tif")
mydataSN <- crop(mydataSN, p2.list)
mydataSN <- as.im.RasterLayer(mydataSN)
mydataSN <- eval.im(as.factor(mydataSN))
plot(mydataSN)
plot(mydataSN, axes = TRUE, legend=TRUE)
points( PP_tw_DD$x, PP_tw_DD$y, col ="white")
plot(mydataSN, add = TRUE)
hist(mydataSN)
levels(mydataSN) <- c(">37.6", ">8.3", ">4.9", ">3.0", ">0.0", "NA")
levels(mydataSN) <- c("0", "1", "2", "3", "4", "5")
#### -------------------------------------------------------------------------------- coordinates
lat = elbe2013_SN_hwr$latitude
lon = elbe2013_SN_hwr$longitude
xy = data.frame(lon, lat)
coordinates(xy) <- c("lon", "lat")
proj4string(xy) <- CRS("+proj=longlat +datum=WGS84")
NE <- spTransform(xy, CRS("+proj=utm +zone=33U ellps=WGS84"))
NE <- as.data.frame(NE)
elbe2013_SN_hwr$UTM_x  <- NE$lon
elbe2013_SN_hwr$UTM_y  <- NE$lat

cond1 <- elbe2013_SN_hwr$screenName == "lzmontour"
cond1 <- elbe2013_SN_hwr$screenName == "Sakurion"
cond1 <- elbe2013_SN_hwr$screenName == "cPradi"
cond1 <- elbe2013_SN_hwr$screenName == "thwbeckingen"
#cond2 <- df$sub == 3 & df$day == 4
elbe2013_SN_hwr <- elbe2013_SN_hwr[!cond1,]
summary(elbe2013_SN_hwr)

#### -------------------------------------------------------------------------------- build ppp

#tw_Sachsen <- elbe2013_sachsen2
W <- owin(poly=p2.list)
elbe2013_SN_hwr <- unique(elbe2013_SN_hwr)
summary(elbe2013_SN_hwr)

tweets_SN_ppp_m <- ppp(elbe2013_SN_hwr$UTM_x, elbe2013_SN_hwr$UTM_y, window=W, marks =elbe2013_SN_hwr$al )
tweets_SN_ppp_m <- ppp(elbe2013_SN_hwr$UTM_x, elbe2013_SN_hwr$UTM_y, window=W)
tweets_SN_ppp_m <- unique.ppp(tweets_SN_ppp_m)
tweets_SN_ppp_m
#elbe2013_SN_hwr <- sqldf("SELECT * FROM elbe2013_sachsen WHERE lower(text) LIKE '%wasser%' OR 'regen' OR 'sandsack'", user="postgres",
password = "admin", host = "localhost", port=5432, dbname="test_sqldf")
#elbe2013_SN_hwr <- sqldf("SELECT * FROM elbe2013_sachsen WHERE lower(text) LIKE '%wasser%' OR '%feuerwehr%' OR 'sandsack'", user="post-
gres", password = "admin", host = "localhost", port=5432, dbname="test_sqldf")
#flut, schwemmung, wasser, feuerwehr, sandsack, hilfe, katastrophe,
rescale(tweets_SN_ppp_m, 1000)

#### -------------------------------------------------------------------------------- build model 1
tw_SN_hwr.fitcov1 <- ppm(tweets_SN_ppp_m ~ x+y + mydataSN)
plot(tw_SN_hwr.fitcov1)
diagnose.ppm(tw_SN_hwr.fitcov1, type="pearson")


#### -------------------------------------------------------------------------------- build model 2 thwbeckingen
tw_SN_hwr.fitcov2 <- ppm(tweets_SN_ppp_m ~polynom(x,y,3) + Z, covariates=list(Z=mydataSN))
plot(tw_SN_hwr.fitcov2)
diagnose.ppm(tw_SN_hwr.fitcov2, type="pearson")

#### -------------------------------------------------------------------------------- build model 3
tw_SN_hwr.fit1 <- ppm(tweets_SN_ppp_m ~polynom(x,y,3))
plot(tw_SN_hwr.fit1)
diagnose.ppm(tw_SN_hwr.fit1, type="pearson")

#### -------------------------------------------------------------------------------- build model 4

weirdfunction <- function(x,y){ 10 * x^2 + 5 * sin(10 * y) }
# (a) covariate values as function
tw_SN_hwr.fitcov3 <- ppm(tweets_SN_ppp_m ~ y + weirdfunction)


#### -------------------------------------------------------------------------------- build model 5
# fit <- ppm(myspp_tw_DD, ~x)

#### -------------------------------------------------------------------------------- build model 6
# fit <- ppm(myspp_tw_DD, ~x+y)
# fits <- ppm(myspp_tw_DD, ~ polynom(x,y,3))

# plot(fits, cif =FALSE, how = "persp")

library(splines)

tw_SN_hwr.fit2 <- ppm(tweets_SN_ppp_m, ~ bs(x, y,df=3)) # no plot(predict(fits))!!!   #B-Spline Basis for Polynomial Splines

# fits <- ppm(myspp_tw_DD, ~ x, Strauss(13), correction="periodic")

plot(tw_SN_hwr.fit1)
tw_SN_hwr.fit3 <- ppm(tweets_SN_ppp_m, ~ bs(x, y,df=3))
#glm.control(maxit = 1000)
```

```
plot(elbe2013_SN_hwr$UTM_x,elbe2013_SN_hwr$UTM_y )

diagnose.ppm(tw_SN_hwr.fit1, type="pearson")

plot(tw_SN_hwr.fit1)
point_SN.sim2 <- simulate(tw_SN_hwr.fit1, 1)
plot(point_SN.sim2)


plot(predict.ppm(tw_DD.fitcov2, type = "lambda"), main = "Predicted Intensity (lambda)") #predicted intensity
tw_DD.kc2<- envelope(tw_DD.fitcov2, Kest, nsim = 99, correction = "border", verbose = F)
plot(tw_DD.kc2)

p2 <- predict.ppm(tw_DD.fitcov2)
plot(p2)
plot(tw_DD.fitcov2, cif =FALSE, how = "persp")

qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "pearson") # okok
qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "inverse") # okokok

qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "eem") #
qqplot.ppm(tw_DD.fitcov2, nsim = 39, type= "raw") #


### ----------------- check lon lat   ¯\_(ツ)_/¯  cPradi        Sakurion       lzmontour
hist(elbe2013_SN_hwr$UTM_y)
hist(elbe2013_SN_hwr$UTM_x)

table(elbe2013_SN_hwr$screenName)
tmp = sort(table(elbe2013_SN_hwr$screenName), decreasing = T)
dotchart((tmp[10:1])):mtext('Number of tweets posted by screenname')
tmp [50:1]
tmp = sort(table(elbe2013_SN_hwr$UTM_y), decreasing = T)
dotchart((tmp[10:1])):mtext('Number of tweets posted by screenname')
tmp [50:1]
tmp = sort(table(elbe2013_SN_hwr$UTM_x), decreasing = T)
dotchart((tmp[10:1])):mtext('Number of tweets posted by screenname')
tmp [50:1]

tmp = sort(table(testdf$screenName), decreasing = T)
dotchart((tmp[10:1])):mtext('Number of tweets posted by screenname')
tmp = sort(table(testdf$UTM_x), decreasing = T)
dotchart((tmp[10:1])):mtext('Number of tweets posted by screenname')

summary(testdf)
hist(testdf$UTM_y)
hist(testdf$UTM_x)

### --- elbe2013_de_hochwasser Kapitel 4
myLocation <- c(12, 50, 15, 52)
hdf1 <- get_map(myLocation)
ggmap(hdf1, extent = "normal") + geom_point(aes(x = longitude, y = latitude), data = testdf) + guides(colour=FALSE) +
  stat_density2d(aes(x = longitude, y = latitude, fill = ..level..,  alpha = ..level..), bins = 10, data = testdf, geom = "polygon") +
  scale_fill_gradient(low = "black", high = "red")

ggmap(hdf1, extent = "normal") + geom_point(aes(x = longitude, y = latitude), data = testdf)  +
  stat_bin2d(aes(x = longitude, y = latitude), data = testdf) +
  scale_fill_gradient(low = "white", high = "red")

drops <- c(2, 5, 22, 56)
data <- data[-drops,]
# fit <- ppm(myspp_tw_DD, ~x)
# fit <- ppm(myspp_tw_DD, ~x+y)
# fits <- ppm(myspp_tw_DD, ~ polynom(x,y,3))
# plot(fits, cif =FALSE, how = "persp")
# library(splines)
# fits <- ppm(myspp_tw_DD, ~ bs(x, y,df=3)) # no plot(predict(fits))!!!   #B-Spline Basis for Polynomial Splines
# fits <- ppm(myspp_tw_DD, ~ x, Strauss(13), correction="periodic")
```

7.2.6 R-Script Sentiment analysis

```
#update.packages()
.libPaths("C:/R-packages")
.libPaths()
library(syuzhet)
library(lubridate)
library(ggplot2)
library(scales)
library(reshape2)
library(dplyr )

test_tw <- sandy_miami_df
mySentiment <- get_nrc_sentiment(as.vector(tw_miami_related$text))
tw_miami_related <- cbind(tw_miami_related, mySentiment)
col(tweets01)

#ggplot(elbe2013_sachsen, aes(latitude, longitude)) + geom_bin2d(bins=300, binwidth = .25) + coord_map("ortho") #+ coord_fixed(ratio =
.5)# xlim(4, 10) + ylim(4, 10)
ggplot(tweets01, aes(lat_final, lng_final)) + geom_point() + geom_bin2d()#geom_density_2d(bandwidth=0.4)

sentimentTotals <- data.frame(colSums(tweets01[,c(15:24)]))
names(sentimentTotals) <- "count"

sentimentTotals <- cbind("sentiment" = rownames(sentimentTotals), sentimentTotals)
rownames(sentimentTotals) <- NULL

ggplot(data = sentimentTotals, aes(x = sentiment, y = count)) +
  geom_bar(aes(fill = sentiment), stat = "identity") +
  theme(legend.position = "none") +
  xlab("Sentiment") + ylab("Total Count") + ggtitle("Total Sentiment Score Miami") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
### ----------------------------------------------------------

names(posnegtime) <- c("timestamp", "sentiment", "meanvalue")
posnegtime$sentiment = factor(posnegtime$sentiment,levels(posnegtime$sentiment)[c(2,1)])

# pos neg
ggplot(data = posnegtime, aes(x = as.Date(timestamp), y = meanvalue, group = sentiment)) +
  geom_line(size = 2.5, alpha = 0.7, aes(color = sentiment)) +
  geom_point(size = 0.5) +
  ylim(0, NA) +
  scale_colour_manual(values = c("springgreen4", "firebrick3")) +
  theme(legend.title=element_blank(), axis.title.x = element_blank()) +
  scale_x_date(breaks = date_breaks("1 days"),
               labels = date_format("%d")) +
  ylab("Average sentiment score") +
  ggtitle("Sentiment Over Time") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# ------------------------------------------------------------------------------------ fear and negative
feartime <- tweets01 %>%
  group_by(timestamp = cut(timestamp, breaks="1 days")) %>%
  summarise(fear = mean(fear),
            neagative = mean(negative)) %>% melt

feartime1 <- tweets01 %>%
  group_by(timestamp = cut(timestamp, breaks="1 days")) %>%
  summarise(fear = mean(fear)) %>% melt

names(feartime) <- c("timestamp", "sentiment", "meanvalue")
feartime$sentiment = factor(feartime$sentiment,levels(feartime$sentiment)[c(2,1)])

# --- ggplot
ggplot(data = feartime, aes(x = as.Date(timestamp), y = meanvalue, group = sentiment)) +
  geom_line(size = 2.5, alpha = 0.7, aes(color = sentiment)) +
  geom_point(size = 0.5) +
  ylim(0, 2) +
  scale_colour_manual(values = c("red", "firebrick3", "black")) +
  #scale_colour_manual(values = c("red", "blue", "green")) +
  #theme(legend.title=element_blank(), axis.title.x = element_blank()) +
  scale_x_date(breaks = date_breaks("1 days"), labels = date_format("%d")) +
  ylab("Average sentiment score") + # xlab("day")
  ggtitle("Sentiment Over Time Miami") +
  #theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  theme(legend.title=element_blank(), axis.title.x = element_blank()) +
  guides(colour = guide_legend(override.aes = list(size=4))) +
  theme(legend.key=element_rect(fill='pink'))
  #scale_colour_manual(name='', values=c('fear'='grey', 'negative'='red','Point values'='black'), guide='legend') +
  #scale_colour_manual(name='', values=c('Important line'='grey', 'Point values'='red'))
  #guides(colour = guide_legend(override.aes = list(linetype=c(1,0))))
  #guides(colour = guide_colorbar(), size = guide_legend(), shape = guide_legend())

### -------------------------------------------------------------------------------------------

tweets01$weekday <- wday(tweets01$timestamp, label = TRUE)
tweets01$day <- yday(tweets01$timestamp)

weeklysentiment <- tweets01 %>% group_by(weekday) %>%
  summarise(anger = mean(anger),
            anticipation = mean(anticipation),
            disgust = mean(disgust),
            fear = mean(fear),
            joy = mean(joy),
            sadness = mean(sadness),
            surprise = mean(surprise),
            trust = mean(trust)) %>% melt
names(weeklysentiment) <- c("weekday", "sentiment", "meanvalue")

##tweets01$timestamp <- with_tz(ymd_hms(tweets01$date), "Europe/Berlin")

daysentiment <- tweets01 %>% group_by(day) %>%
  summarise(anger = mean(anger),
            anticipation = mean(anticipation),
            disgust = mean(disgust),
            fear = mean(fear),
            joy = mean(joy),
            sadness = mean(sadness),
            surprise = mean(surprise),
            trust = mean(trust)) %>% melt
names(daysentiment) <- c("day", "sentiment", "meanvalue")


ggplot(data = weeklysentiment, aes(x = weekday, y = meanvalue, group = sentiment)) +
  geom_line(size = 2.5, alpha = 0.7, aes(color = sentiment)) +
  geom_point(size = 0.5) +
  ylim(0, 0.6) +
  theme(legend.title=element_blank(), axis.title.x = element_blank()) +
  ylab("Average sentiment score") +
  ggtitle("Sentiment During the Week")


ggplot(data = daysentiment, aes(x = variable, y = fear) + group = variable) +
  geom_line(size = 2.5, alpha = 0.7, aes(color = fear)) +
  geom_point(size = 0.5) +
  ylim(0, 0.6) +
  theme(legend.title=element_blank(), axis.title.x = element_blank()) +
  ylab("Average sentiment score") +
  ggtitle("Sentiment During the days : fear")

### ------------------------------------------------------------

#ggplot sentiment fear 2912
ggplot(data = dayfear, aes(x = day, y = fear)) +
  geom_line(aes(color = fear)) +
  geom_point(size = 0.5) +
  ylim(0, 1500) +
  theme(legend.title=element_blank(), axis.title.x = element_blank()) +
  ylab("Average sentiment score") +
  ggtitle("sentiment fear dataframe Miami")
```

7.2.7 R-Script 3D-Scatterplot with sentiment

```
### ----------------------------------------------------------------------------------

library(syuzhet)
library(lubridate)
library(ggplot2)
library(scales)
library(reshape2)
library(dplyr )

mySentiment <- get_nrc_sentiment(as.vector(test_tw$text))
tweets <- cbind(test_tw, mySentiment)
col(tweets)

sentimentTotals <- data.frame(colSums(tweets[,c(31:38)]))
names(sentimentTotals) <- "count"
sentimentTotals <- cbind("sentiment" = rownames(sentimentTotals), sentimentTotals)
rownames(sentimentTotals) <- NULL
ggplot(data = sentimentTotals, aes(x = sentiment, y = count)) +
  geom_bar(aes(fill = sentiment), stat = "identity") +
  theme(legend.position = "none") +
  xlab("Sentiment") + ylab("Total Count") + ggtitle("Total Sentiment Score for All Tweets")

### --------------------------------------------------------

posnegtime <- tweets %>%
  group_by(timestamp = cut(timestamp, breaks="1 days")) %>%
  summarise(negative = mean(negative),
            positive = mean(positive)) %>% melt

names(posnegtime) <- c("timestamp", "sentiment", "meanvalue")
posnegtime$sentiment = factor(posnegtime$sentiment,levels(posnegtime$sentiment)[c(2,1)])

ggplot(data = posnegtime, aes(x = as.Date(timestamp), y = meanvalue, group = sentiment)) +
  geom_line(size = 2.5, alpha = 0.7, aes(color = sentiment)) +
  geom_point(size = 0.5) +
  ylim(0, NA) +
  scale_colour_manual(values = c("springgreen4", "firebrick3")) +
  theme(legend.title=element_blank(), axis.title.x = element_blank()) +
  scale_x_date(breaks = date_breaks("1 days"),
               labels = date_format("%d")) +
  ylab("Average sentiment score") +
  ggtitle("Sentiment Over Time")

warnings()

### -------------------------------------------------------------------------------------------

tweets$weekday <- wday(tweets$timestamp, label = TRUE)
tweets$weekday <- day(tweets$timestamp) #, label = TRUE)

weeklysentiment <- tweets %>% group_by(weekday) %>%
  summarise(anger = mean(anger),
            anticipation = mean(anticipation),
            disgust = mean(disgust),
            fear = mean(fear),
            joy = mean(joy),
            sadness = mean(sadness),
            surprise = mean(surprise),
            trust = mean(trust)) %>% melt
names(weeklysentiment) <- c("weekday", "sentiment", "meanvalue")

ggplot(data = weeklysentiment, aes(x = weekday, y = meanvalue, group = sentiment)) +
  geom_line(size = 2.5, alpha = 0.7, aes(color = sentiment)) +
  geom_point(size = 0.5) +
  ylim(0, 0.6) +
  theme(legend.title=element_blank(), axis.title.x = element_blank()) +
  ylab("Average sentiment score") +
  ggtitle("Sentiment During the Week")
```

7.2.8 R-Script Maps

```
#####################################################################
# Phase 0  : Karten
#####################################################################
library(mapproj)

par(mfrow=c(2,2))
### Dataframe elbe2013
albers_proj<-map("world", regions=c("Germany"), proj="albers", param=c(39, 45), col="#999999", fill=FALSE, bg=NA, lwd=0.2, add=FALSE,
resolution=1)
points(mapproject(elbe2013$longitude, elbe2013$latitude), col=NA, bg="#00000030", pch=21, cex=0.25) # cex=1.0)
mtext("tweets Dataframe elbe2013", side = 3, line = -2.0, outer = T, cex=1.5, font=3)
# speichern
cowplot::plot_grid(timeDist, timeDistDayOf)

### Dataframe elbe2013_de
albers_proj<-map("world", regions=c("Germany"), proj="albers", param=c(39, 45), col="#999999", fill=FALSE, bg=NA, lwd=0.2, add=FALSE,
resolution=1)
points(mapproject(elbe2013_de$longitude, elbe2013_de$latitude), col=NA, bg="#00000030", pch=21, cex=0.25) # cex=1.0)
mtext("tweets dataframe elbe2013_de", side = 3, line = -2.0, outer = T, cex=1.5, font=3)

### Dataframe elbe2013_sachsen
albers_proj<-map("world", regions=c("Germany"), proj="albers", param=c(39, 45), col="#999999", fill=FALSE, bg=NA, lwd=0.2, add=FALSE,
resolution=1)
points(mapproject(elbe2013_sachsen$longitude, elbe2013_sachsen$latitude), col=NA, bg="#00000030", pch=21, cex=0.25) # cex=1.0)
mtext("tweets dataframe elbe2013_sachsen", side = 3, line = -2.0, outer = T, cex=1.5, font=3)

### Dataframe elbe2013_de_hochwasser
```

```
albers_proj<-map("world",  regions=c("Germany"),  proj="albers",  param=c(39,  45),  col="#999999",  fill=FALSE,  bg=NA,  lwd=0.2,  add=FALSE,
resolution=1)
points(mapproject(elbe2013_de_hochwasser$longitude, elbe2013_de_hochwasser$latitude), col=NA, bg="#00000030", pch=21, cex=0.25) # cex=1.0)
mtext("tweets dataframe elbe2013_de_hochwasser", side = 3, line = -2.0, outer = T, cex=1.5, font=3)

### Dataframe elbe2013_de_regen
albers_proj<-map("world",  regions=c("Germany"),  proj="albers",  param=c(39,  45),  col="#999999",  fill=FALSE,  bg=NA,  lwd=0.2,  add=FALSE,
resolution=1)
points(mapproject(elbe2013_de_regen$longitude, elbe2013_de_regen$latitude), col=NA, bg="#00000030", pch=21, cex=0.25) # cex=1.0)
mtext("tweets dataframe elbe2013_de_regen", side = 3, line = -2.0, outer = T, cex=1.5, font=3)

# Dresden: latitude: 51.05, longitude: 13.7333
### Dataframe elbe2013_de_elban
albers_proj<-map("world",  regions=c("Germany"),  proj="albers",  param=c(39,  45),  col="#999999",  fill=FALSE,  bg=NA,  lwd=0.2,  add=FALSE,
resolution=1)
points(mapproject(elbe2013_de_elban$longitude, elbe2013_de_elban$latitude), col=NA, bg="#00000030", pch=21, cex=0.25) # cex=1.0)
mtext("tweets dataframe elbe2013_de_elban", side = 3, line = -2.0, outer = T, cex=1.5, font=3)

### Dataframe elbe2013_de_elban_hw
albers_proj<-map("world",  regions=c("Germany"),  proj="albers",  param=c(39,  45),  col="#999999",  fill=FALSE,  bg=NA,  lwd=0.2,  add=FALSE,
resolution=1)
points(mapproject(elbe2013_de_elban_hw$longitude, elbe2013_de_elban_hw$latitude), col=NA, bg="#00000030", pch=21, cex=0.25) # cex=1.0)
mtext("tweets dataframe elbe2013_de_elban_hw", side = 3, line = -2.0, outer = T, cex=1.5, font=3)

### Dataframe elbe2013_dresden
albers_proj<-map("world",  regions=c("Germany"),  proj="albers",  param=c(39,  45),  col="#999999",  fill=FALSE,  bg=NA,  lwd=0.2,  add=FALSE,
resolution=1)
points(mapproject(elbe2013_dresden$longitude, elbe2013_dresden$latitude), col=NA, bg="#00000030", pch=21, cex=0.25) # cex=1.0)
mtext("tweets dataframe elbe2013_de_elban_hw", side = 3, line = -2.0, outer = T, cex=1.5, font=3)


################################################################### coole Maps mit Google
library(sp)
library(RColorBrewer)
library(ggmap)

myLocation <- c(2,45,18,58)
hdf <- get_map(myLocation)
ggmap(hdf, extent = "normal") + geom_point(aes(x = longitude, y = latitude), data = elbe2013)

myLocation <- c(2,45,18,58)
hdf <- get_map(myLocation)
ggmap(hdf, extent = "normal") + geom_point(aes(x = longitude, y = latitude), data = elbe2013_de_hochwasser)

myLocation <- c( 11.89,50.16,15.03,51.69)
hdf <- get_map(myLocation)
ggmap(hdf, extent = "normal") + geom_point(aes(x = longitude, y = latitude), data = elbe2013_sachsen)

#Map DD mit Tweets elbe2013_dresden
hdf <- get_map("Dresden, sachsen")
ggmap(hdf, extent = "normal") + geom_point(aes(x = longitude, y = latitude, size = 1), data = elbe2013_dresden)

#Map DD mit Tweets elbe2013_dresden_hochwasser
hdf <- get_map("Dresden, sachsen")
gg <-
ggmap(hdf, extent = "normal",legend = "topleft" ) + geom_point(aes(x = longitude, y = latitude), colour = "red", size = .1 ,  data =
elbe2013_dresden_hochwasser) # + geom_point(colour = "red", size = 4)
gg <- gg + rectan
gg

  #geom_segment(aes(y = 50.967454, x = 13.581333, yend = 51.146671, xend = 13.900967), size = 1)


sz  <- 1
xx1 <- as.numeric(13.581333)
xx2 <- as.numeric(13.900967)
yy1 <- as.numeric(50.967454)
yy2 <- as.numeric(51.146671)

#rectan <- (geom_segment(aes(x=xx1, y=yy1, xend = xx1, yend = yy2), size = sz))
           # + geom_segment(aes(xx1, yy2, xx2, yy2), size = sz) +
          #geom_segment(aes(xx2, yy2, xx2, yy2), size = sz) + geom_segment(aes(xx2, yy2, xx1, yy1), size = sz) )
```

7.2.9 R-Script import Flickr

```
############################################################################
#### Import für Hurricane Daten in FlickR
####
.libPaths("C:/R-packages")
.libPaths()
#library(rjson)
#library(Rflickr)
require(ggplot2)
require(lubridate)
library(plyr)
require(stringr)
library(jsonlite)


YOUR_API_KEY_HERE = ''xxxxxxxxxxxxxx''
api_secret = 'xxxxxxxxxxxxxx'
# Get the data
setwd("C:/#rstudio/Elbe2013/")
bbox <- "5,47.06,15.5,56"
pages <- 10
maxdate <- "2013-05-27"
```

```
mindate <- "2013-06-15"
searchstring <- ""
options(OutDec= ".")

for (i in 1:pages) {
  api <- paste("https://api.flickr.com/services/rest/?method=flickr.pho-
tos.search&api_key=",YOUR_API_KEY_HERE,"&text=",searchstring,"&min_taken_date=",min-
date,"&max_taken_date=",maxdate,"&bbox=",bbox,"&has_geo=1&extras=date_upload%2C+date_taken%2C+geo%2C+descrit-
ption%2C+tags&per_page=500&page=",i,"&format=json&nojsoncallback=1", sep="")

  raw_data_elbe2013 <- readLines(api, warn="F", encoding = "UTF-8" )

  rd_elbe2013 <- fromJSON(raw_data_elbe2013)

  ## die datei
  if(i==1) {
    #writeLines()
    write.table(rd_elbe2013, "raw_data_elbe2013a.csv", sep = ";", na = "NA", dec = ".",col.names = T,row.names = FALSE, append = F)# ,
fileEncoding= "UTF-8")
   } else {
    write.table(rd_elbe2013, "raw_data_elbe2013a.csv", sep = ";", na = "NA", dec = ".", col.names = F,row.names = FALSE, append = T,
fileEncoding= "UTF-8")
   }
}


#DF$DTCHGUS <- as.numeric(gsub(",",".",DF$DTCHGUS))

flickr_elbe2013_hochwasser_null<- read.delim("raw_data_elbe2013a.csv", sep = ";" ,header = TRUE) #, quote = "") # quote
flickr_elbe2013_hochwasser_null <- as.data.frame(flickr_elbe2013_hochwasser_null)
summary(flickr_elbe2013_hochwasser)

library(ggmap)
#flickr_elbe2013_hochwasser$photos.photo.longitude

hdf <- get_map("Dresden, sachsen")
ggmap(hdf, extent = "normal") + geom_point(aes(x = photos.photo.longitude, y = photos.photo.latitude),colour = "red", size = 1 , data =
flickr_elbe2013_hochwasser)

# ggmap(hdf, extent = "normal") +
#   geom_point(aes(x = photos.photo.longitude, y = photos.photo.latitude),colour = "red", size = 1 , data = flickr_elbe2013_hochwasser)
+
#   facet_wrap(~ photos.photo.datetaken)


hdf <- get_map("Germany")
hdf <- get_map(location = c(2,44,18,56))
ggmap(hdf, extent = "normal") + geom_point(aes(x = photos.photo.longitude, y = photos.photo.latitude),colour = "red", size = 1 , data =
flickr_elbe2013_hochwasser)

hdf <- get_map("Germany")
hdf <- get_map(location = c(2,44,18,56))
ggmap(hdf, extent = "normal") + geom_point(aes(x = photos.photo.longitude, y = photos.photo.latitude),colour = "red", size = 1 , data =
flickr_elbe2013_hochwasser_null)


### ok

### ---------------------------------------------------------------------------------------------------------
```

7.2.10 R-Script Flickr analysis

```
### MT-Flickr

.libPaths("C:/R-packages")
library(spatstat)
library(mapproj)
library(maptools)
library(ggmap)
library(sp)
library(splines)
library(raster)
library(sqldf)

# gather via sqldef
lat = flickr_elbe2013_hochwasser$photos.photo.latitude
lon = flickr_elbe2013_hochwasser$photos.photo.longitude
xy = data.frame(lon, lat)
coordinates(xy) <- c("lon", "lat")
proj4string(xy) <- CRS("+proj=longlat +datum=WGS84")
NE <- spTransform(xy, CRS("+proj=utm +zone=33U ellps=WGS84"))
NE <- as.data.frame(NE)
flickr_elbe2013_hochwasser$UTM_x  <- NE$lon
flickr_elbe2013_hochwasser$UTM_y  <- NE$lat

par(mfrow=c(1,1))
e <- extent(400404.8, 5645095, 423145.3, 5668739)
#WHERE (UTM_y >= 5645095 AND UTM_y <= 5668739) AND (UTM_x >= 400404.8 AND UTM_x <= 423145.3)
flickr_DD <- sqldf("SELECT * FROM flickr_elbe2013_hochwasser WHERE (UTM_y >= 5645095 AND UTM_y <= 5668739) AND (UTM_x >= 400404.8 AND
UTM_x <= 423145.3)", user="postgres", password = "admin", host = "localhost", port=5432, dbname="test_sqldf")

### ------------------------------------------------------------------------------------------- Sptatial Analysis elbde_de_hochwasser
### --- Inspecting data  flickr_DD_red
flickr_DD <- flickr_DD_backup
#flickr_DD_backup <- flickr_DD
flickr_DD <- flickr_DD_red
flickr_DD <- unique(flickr_DD)
flickr_DD

summary(flickr_DD)
str(flickr_DD)
PP_flickr_DD <- as.vector(flickr_DD[, c(30,31)])
names(PP_flickr_DD) <- c("x","y")

PP_flickr_DD <- unique(PP_flickr_DD)
summary(PP_flickr_DD)

w_flickr_DD <- owin(c(bbox_DD$UTM_x[1],bbox_DD$UTM_x[2]), c(bbox_DD$UTM_y[1],bbox_DD$UTM_y[2]), unitname=c("meter","meter"))

myspp_flickr_DD <- ppp(PP_flickr_DD$x, PP_flickr_DD$y, window = w_flickr_DD)

unique(myspp_flickr_DD)
unitname(myspp_flickr_DD)
plot(myspp_flickr_DD, axes = TRUE )

Q_flickr <- quadratcount(myspp_flickr_DD, nx = 10, ny = 8)

plot(myspp_flickr_DD )
plot(Q_flickr, add = TRUE, cex =1, pch ="+", col = "blue")

plot(density(myspp_flickr_DD, 750))
contour(density(myspp_flickr_DD, 750))

# perspective
persp(density(myspp_flickr_DD, 500))


### -------------------------------- intensity

#par(mfrow=c(2,1))
plot(density.ppp(myspp_flickr_DD, sigma = bw.diggle(myspp_flickr_DD),edge=T),main=paste("sigma(Diggle) =",round(bw.dig-
gle(myspp_flickr_DD),2)))
plot(density.ppp(myspp_flickr_DD, sigma = bw.ppl(myspp_flickr_DD),edge=T),main=paste("sigma(LikeCross)
=",round(bw.ppl(myspp_flickr_DD),2)))
#plot(density.ppp(myspp_flickr_DD, sigma = bw.scott(myspp_flickr_DD)[2],edge=T),main=paste("sigma(Scott)
=",round(bw.scott(myspp_flickr_DD)[2],2)))
plot(density.ppp(myspp_flickr_DD, sigma = bw.scott(myspp_flickr_DD)[1],edge=T),main=paste("sigma(Scott)
=",round(bw.scott(myspp_flickr_DD)[1],2)))
#par(mfrow=c(1,1))

### -------------------------------- Complete Spatial Randomness CHI quad

quadrat.test(myspp_flickr_DD, nx = 5, ny = 4)
plot(quadrat.test(myspp_flickr_DD, nx = 5, ny = 4))



### -------------------------------- Complete Spatial Randomness Kolmogorov-Smirnov

cdf.test(myspp_flickr_DD, "x")
plot(cdf.test(myspp_flickr_DD, "x"))

cdf.test(myspp_flickr_DD, "y")
plot(cdf.test(myspp_flickr_DD, "y"))

### -------------------------------- K-funtion

plot(Kest(myspp_flickr_DD))
plot(Kinhom(myspp_flickr_DD))
Kinhom(myspp_flickr_DD)
plot(envelope(myspp_flickr_DD, Kest))

albers_proj<-map("world", regions=c("Germany"), proj="albers", param=c(39, 45), col="#999999", fill=FALSE, bg=NA, lwd=0.2, add=FALSE,
resolution=1)
points(mapproject(flickr_DD$photos.photo.longitude, flickr_DD$photos.photo.latitude), col=NA, bg="#00000030", pch=21, cex=0.25, main
="Dresden") # cex=1.0)

plot(Kest(myspp_flickr_DD))

## -------------------------------- checking a fitted model 1
fit_flickr <- ppm(myspp_flickr_DD, ~x)
test_flickr <- quadrat.test(fit_flickr, nx = 5, ny = 4)
test_flickr
# X2 = 977.61, df = 18, p-value < 2.2e-16 rejected!!!

# get more information, why the modell is bad

plot(myspp_flickr_DD, pch = ".", size = 2)
plot(test_flickr, add = TRUE, cex = 1.0, col = "red")

kstest(myspp_tw_DD, "y")

### - Residual Measure

fit01_flickr <- ppm(myspp_flickr_DD, ~x+y)
plot(predict(fit01_flickr))
plot(myspp_flickr_DD, add =  TRUE, pch="+")

fit02_flickr <- ppm(myspp_flickr_DD, ~x)
diagnose.ppm(fit02_flickr, which = "smooth")
diagnose.ppm(fit02_flickr)

#covariate
l+urking(fit02_flickr, x, type = "raw")
```

```
### ------------------------------------------------------------------------------- ok ok
### - search better model ~ polynom(x,2) # ~ bs(x,df=3)
fit03_flickr <- ppm(myspp_flickr_DD, ~ polynom(x,y,2))
fit03_flickr
diagnose.ppm(fit03_flickr)
#library(splines)
#fit04_flickr <- ppm(myspp_flickr_DD, ~ bs(x, y,df=3)) # no plot(predict(fits))!!!   #B-Spline Basis for Polynomial Splines

fit04_flickr <- ppm(myspp_flickr_DD, ~ x, Strauss(13), correction="periodic")

plot(predict(fit04_flickr))
plot(myspp_flickr_DD, add =  TRUE, pch="+")
diagnose.ppm(fit04_flickr, which = "smooth")
diagnose.ppm(fit04_flickr)

#fits
#coef(fits)
#coef(summary(fits))

# M. Morisita (1959) Measuring of the dispersion of individuals and analysis of the distributional patterns.
miplot(myspp_flickr_DD)
fryplot(myspp_flickr_DD, width = 5000,  axes = TRUE)

### -------------------------------- distance methods

# pairwie distance
plot(pairdist(myspp_flickr_DD))

# nearest neighbour
plot(nndist(myspp_flickr_DD))

### ---------------------------------------------------------------------------- empty space distances
plot(distmap(myspp_flickr_DD))

# Stienen diagramm
plot(myspp_flickr_DD %mark% (nndist(myspp_flickr_DD)/2), markscale = 1, main = "Stienen diagram")

# --- Edge Effects F
Fest(myspp_flickr_DD, correction = "km")
Fest(myspp_flickr_DD, correction = c("km","cs"))

plot(Fest(myspp_flickr_DD))

plot(Fest(myspp_flickr_DD), hazard ~ r, main="Hazard rate of F")

# ------------------------------------------------------------------------------ G-Function
Gest(myspp_flickr_DD)
plot(Gest(myspp_flickr_DD))

plot(G, cbind(km, rs, theo) ~ r)

# ------------------------------------------------------------------------------ pairwise distances K-Function

L <- Lest(myspp_flickr_DD)
plot(L, main ="L-function")
plot(pcf(myspp_flickr_DD))
plot(allstats(myspp_flickr_DD))

# ------------------------------------------------------------------------------ Monte Carlo Tests

E_flickr <- envelope(myspp_flickr_DD, Kest, nsim = 39, rank =1)
E_flickr
plot(E_flickr, main = "pointwise envelopes")

E_flickr <- envelope(myspp_flickr_DD, Kest, nsim = 19, rank =1, global = TRUE)
plot(E_flickr, main = "global envelopes")

E <- envelope(myspp_flickr_DD, Lest, nsim = 19, rank =1, global = TRUE)
plot(E_flickr, main = "global envelopes of L(r)")

### - enveloppes for any fitted model 136
#fit <- ppm(myspp_tw_DD, )

### --- model fitting using summary statisticsVarGamma

# --- fitting a cluster process

fit_flickr <- kppm(myspp_flickr_DD, ~1, "Thomas")
fit_flickr
plot(fit_flickr)

fit_flickr <- kppm(myspp_flickr_DD, ~ polynom(x,y,2), "VarGamma")
fit_flickr
plot(fit_flickr)


# --- fitting Matern

fitM_flickr <- kppm(myspp_flickr_DD, ~1, "MatClust")
plot(simulate(fit_flickr, nsim = 4))

seed("100")

plot(simulate(fit, Lest, nsim = 39))

fitp <- kppm(myspp_flickr_DD, ~1, "Thomas", statistic = "pcf")
fitp
plot(fitp)
```

```
### -------------------------------- exploring local features // don't work

plot(myspp_flickr_DD, pch=".", main="")

# LISA nn feature and noise
Z <- nnclean(myspp_flickr_DD, k = 20)
plot(Z, chars =c(".","+"), main = "nearest neigbour cleaning")

### -------------------------------- adjusting for inhomogeneity

fit <- ppm(myspp_flickr_DD)
Ki <- Kinhom(myspp_flickr_DD)
plot(Ki, main="inhomogeneous K-function")
Ki

Linhom(myspp_flickr_DD)
plot(Linhom(myspp_flickr_DD))

Ginhom(myspp_flickr_DD)
plot(Ginhom(myspp_flickr_DD))

pcfinhom(myspp_flickr_DD)
plot(pcfinhom(myspp_flickr_DD))

### ----------------------------------------------------------------------------------------------------- flickr

### --- final fitcov2
#myspp_flickr_DD_backup <- myspp_flickr_DD
myspp_flickr_DD <- myspp_flickr_DD_backup
myspp_flickr_DD <- rescale(myspp_flickr_DD)

fitcov2_flickr <- ppm(myspp_flickr_DD ~polynom(x,y,2) + Z, covariates=list(Z=landuse))  ## passt , dann hinweis auf numerische instabi-
lität
diagnose.ppm(fitcov2_flickr, type="pearson")
fitcov2_flickr
plot(fitcov2_flickr)
diagnose.ppm(fitcov2_flickr, type="pearson")
p_fitcov2_flickr <- predict.ppm(fitcov2_flickr)
plot(p_fitcov2_flickr)
qqplot.ppm(fitcov2_flickr, nsim = 39, type= "pearson") # okok
qqplot.ppm(fitcov2_flickr, nsim = 39, type= "inverse") # okokok
qqplot.ppm(fitcov2_flickr, nsim = 39, type= "eem") #
qqplot.ppm(fitcov2_flickr, nsim = 39, type= "raw") #

### ---- test 29.12.
fitcov3_flickr <- ppm(myspp_flickr_DD ~ x+y + landuse)
diagnose.ppm(fitcov3_flickr, type="pearson")
p3_flickr <- predict.ppm(fitcov3_flickr)
plot(p3_flickr)

### ----------------------------------------------------------------------------------------------------- flickr 31.12.

fitcov3_flickr <- ppm(myspp_flickr_DD ~polynom(x,y,3) + Z, covariates=list(Z=landuse))  ## ok, but Fisher Matrix problem
diagnose.ppm(fitcov3_flickr, type="pearson")
fitcov3_flickr
plot(fitcov3_flickr)
diagnose.ppm(fitcov3_flickr, type="pearson")
p_fitcov3_flickr <- predict.ppm(fitcov3_flickr)
plot(p_fitcov3_flickr)
qqplot.ppm(fitcov3_flickr, nsim = 39)
qqplot.ppm(fitcov3_flickr, nsim = 39, type= "pearson") # okok
qqplot.ppm(fitcov3_flickr, nsim = 39, type= "inverse") # okokok
qqplot.ppm(fitcov3_flickr, nsim = 39, type= "eem") #
qqplot.ppm(fitcov3_flickr, nsim = 39, type= "raw") #

point.DD_fr.sim2 <- simulate(fitcov3_flickr, 1)
plot(point.DD_fr.sim2)
plot(predict.ppm(fitcov3_flickr, type = "lambda"), main = "Predicted Intensity (lambda)") #predicted intensity
fr_SN.kc1<- envelope(fitcov3_flickr, Kest, nsim = 99, correction = "border", verbose = F)
plot(fr_SN.kc1)
### ------------------------------------------------------------------------------------------------------------

#######################################################################
# lat = flickr_elbe2013_hochwasser$photos.photo.latitude
# lon = flickr_elbe2013_hochwasser$photos.photo.longitude
library(lubridate)

is.POSIXct(flickr_elbe2013_hochwasser$photos.photo.datetaken)
flickr_elbe2013_hochwasser$date_POSIXct <- as.POSIXct(flickr_elbe2013_hochwasser$photos.photo.datetaken, format = "%Y-%m-%d %H:%M")
# pro stunde 3600
ggplot(data=flickr_elbe2013_hochwasser, aes(x=date_POSIXct)) +
  geom_histogram(aes(fill=..count..), binwidth=60) + #3600
  guides(fill = "none") +
  scale_y_continuous("flickr messages") + ggtitle("flickr-stream dataframe flickr_elbe2013_hochwasser") # + ylim(0, 80000)

ggplot(data=flickr_elbe2013_hochwasser, aes(x=photos.photo.longitude)) + geom_histogram(aes(fill=..count..)) + ggtitle("distribution of
longitude (flickr_elbe2013_hochwasser)")
ggplot(data=flickr_elbe2013_hochwasser, aes(x=photos.photo.latitude))  + geom_histogram(aes(fill=..count..)) + ggtitle("distribution of
latitude (flickr_elbe2013_hochwasser)")

ggplot(data=flickr_DD, aes(x=date_POSIXct)) +
  geom_histogram(aes(fill=..count..), binwidth=3600) +
  guides(fill = "none") +
  scale_y_continuous("flickr messages") + ggtitle("flickr-stream dataframe flickr_DD") # + ylim(0, 80000)

ggplot(data=flickr_DD, aes(x=photos.photo.longitude)) + geom_histogram(aes(fill=..count..)) + ggtitle("distribution of longitude
(flickr_DD)")
ggplot(data=flickr_DD, aes(x=photos.photo.latitude))  + geom_histogram(aes(fill=..count..)) + ggtitle("distribution of latitude
(flickr_DD)")
```

```
### --- flickr_red
ggplot(data=flickr_DD_red, aes(x=photos.photo.longitude)) + geom_histogram(aes(fill=..count..)) + ggtitle("distribution of longitude
(flickr_DD_red)")
ggplot(data=flickr_DD_red, aes(x=photos.photo.latitude))  + geom_histogram(aes(fill=..count..)) + ggtitle("distribution of latitude
(flickr_DD_red)")



#tw_miami_uid4 <- tw_miami_storm[tw_miami_storm$user_id %in% names(which(table(tw_miami_storm$user_id) < 4)), ]

flickr_DD_red <- flickr_DD[flickr_DD$photos.photo.owner%in% names(which(table(flickr_DD$photos.photo.owner) < 4)), ]

###################################################################### time series
library(lubridate)

is.POSIXct(flickr_DD$photos.photo.datetaken)
flickr_DD$date_POSIXct <- as.POSIXct(flickr_DD$photos.photo.datetaken, format = "%Y-%m-%d %H:%M")
# pro stunde 3600
ggplot(data=flickr_DD, aes(x=date_POSIXct)) +
  geom_histogram(aes(fill=..count..), binwidth=3600) +
  guides(fill = "none") +
  scale_y_continuous("flickr messages") + ggtitle("flickr-stream dataframe flickr_DD") # + ylim(0, 80000)


flickr_elbe2013_hochwasser$photos.photo.owner
flickr_DD$photos.photo.owner

hist(flickr_elbe2013_hochwasser$photos.photo.owner)
hist(table(flickr_elbe2013_hochwasser$photos.photo.owner), freq=TRUE, xlab = levels(flickr_elbe2013_hochwasser$photos.photo.owner), ylab
= "Frequencies")

table(flickr_elbe2013_hochwasser$photos.photo.owner)


#flickr_DD.sub <- subset(flickr_DD, flickr_DD.sub$photos.photo.owner != "100230183@N06")

hist(table(flickr_elbe2013_hochwasser$photos.photo.owner), freq=TRUE, xlab = levels(flickr_elbe2013_hochwasser$photos.photo.longitude),
ylab = "Frequencies")

hist(flickr_DD_red$photos.photo.latitude)
hist(flickr_DD_red$photos.photo.longitude)
```