Empirical analysis of complex social and financial networks

PhD thesis

Dániel Kondor

Supervisor: Gábor Vattay, DSc
Consultant: István Csabai, DSc

Department of Physics of Complex Systems
Eötvös Loránd University, Budapest, Hungary

Doctoral School of Physics
Head of School: László Palla, DSc

Doctoral Program for Statistical Physics,
Biological Physics and Physics of Quantum Systems
Head of Program: Jenő Kürti, DSc

Budapest, 2015
Acknowledgements

I would like to thank my supervisor, Gábor Vattay, for over five years of support and guidance over the course of my masters and PhD studies. I would also like to thank István Csabai for many fruitful discussions both about my work and about completely different scientific topics, which provided a lot of inspiration for me in the past few years.

I would also like to thank all the people I had the opportunity to collaborate with over the course of my doctoral studies: Péter Mátray for our collaboration during the first year of my PhD studies and Márton Pósfai for our collaboration on the analysis of the Bitcoin transaction network. László Dobos for his countless advices regarding database systems, Zsófia Kallus and István Gödor at Ericsson for our collaboration during my internship there, and János Szüle, Eszter Bokányi, Tamás Sebők, Norbert Barankai and Tamás Hanyecz for their part in the Twitter data collection and analysis projects.
## Contents

1 Introduction  
1.1 Statistical physics, complex systems and networks  
1.2 The data science paradigm  

2 Large-scale geographic features of language use  
2.1 Related work  
2.2 Data  
2.2.1 Twitter  
2.2.2 Text pre-processing  
2.3 Robust Principal Component Analysis  
2.4 Language use features in the USA  
2.5 Testing scalability of RPCA on New York City  
2.6 Relation to other studies: Public mood assessment  
2.7 Conclusions and future work  

3 Efficient classification of billions of points into complex geographic regions using hierarchical triangular mesh  
3.1 Database systems and data-intensive science  
3.2 A database of billions of tweets  
3.3 Spatial indexing in database systems  
3.4 Indexing Twitter data and geographical regions  
3.5 Spatial joins  
3.6 Performance evaluation  
3.7 Conclusions  

4 Growth, preferential attachment and relation to market effects in Bitcoin  
4.1 The Bitcoin network  
4.1.1 Overview  
4.1.2 Relevant technical details  
4.1.3 Data  
4.1.4 Options for reconstructing the transaction network  
4.2 Evolution of the transaction network  
4.2.1 General characteristics  
4.2.2 Preferential attachment  
4.3 Dynamics of transactions  
4.3.1 Preferential attachment in wealth accumulation  
4.4 Robustness with regard to the network reconstruction method  
4.4.1 Partial temporal aggregation  
4.4.2 Contracted network
4.5 Detecting structural changes ........................................ 72
  4.5.1 Extracting the core network from the Bitcoin transactions ... 72
  4.5.2 The graph time series matrix, and its principal components .... 72
  4.5.3 Main observed features ........................................ 73
4.6 Discussion and future work ...................................... 76

5 Measuring the dimension of partially embedded networks 79
  5.1 Introduction ....................................................... 79
  5.2 Related Work ..................................................... 80
    5.2.1 Random walks and dimensionality of graphs ................ 81
    5.2.2 The role of the link length distribution .................... 82
  5.3 Partially embedded networks .................................. 83
    5.3.1 The connection between $P_0(t)$ and $P(l)$ .................. 85
    5.3.2 Some properties of $d_{e}$ ................................ 85
  5.4 Results ......................................................... 86
    5.4.1 OpenStreetMap ............................................. 87
    5.4.2 Synthetic spatial networks ................................ 88
    5.4.3 Internet infrastructure .................................... 90
    5.4.4 Gowalla geosocial network ................................ 91
  5.5 Randomized versions of the Gowalla network ................... 93
    5.5.1 Twitter user networks .................................... 97
  5.6 Discussion ....................................................... 101

Summary 103
Összefoglalás 104
Publications 105
References 106
Appendix 118
  A.1 Technical details of rank function calculations for the Bitcoin network ... 118
  A.2 Data processing for the Gowalla network ........................ 129
Chapter 1

Introduction

It is a commonly cited statement in complexity sciences nowadays that we are surrounded by digital traces of human activity. Many of the new technologies which became commonly adapted in the past two decades with the start of the information age facilitate the easy and efficient collection and analysis of various data about the core infrastructures of society and behavior of people using them. While there are obvious challenges with respect to the privacy of individuals and related concerns, appropriate use of new, high resolution data presents novel and unique opportunities for scientists as well as for service providers and decision makers. While in social sciences, empirical work was traditionally limited by the high amount of resources needed, recently, the availability of these new data sources resulted in the emergence of a large amount interdisciplinary research focused on the mathematical modeling of various social and economical phenomena, driven by the available data. Based on previous experience gained in statistical physics and complex systems in nature, many physicists have successfully become involved in these research efforts. Now, the main question is that whether we can handle the inherent complexity present in the phenomena being the new focus of attention; in many cases, it is not even certain what the relevant questions can be. Surely, with the advances in data collection and analysis, a first step is assessing the possibilities. Exploiting experience with physical systems, one of the approaches is to evaluate the possibility of applying dimensionality reduction to be able to focus on relevant parameters of a system. This can mean finding a low-dimensional structure in a possibly noisy dataset or possibly separating relevant and irrelevant parts in a system with a complex high dimensional structure.

In this thesis I summarize the results of work on a diverse range of complex systems around us. I present work which utilizes emergent data collection possibilities and analysis methods, aiming to recover the relevant aspects of the underlying systems in a tractable way. In Chapter 1, I review some of the recent developments in complexity sciences, the emerging paradigm of data science, and some of the recent involvement of physicists in various research topics and problems. In Chapter 2, I present the results of a study of applying dimensionality reduction to a large-scale dataset of language use including spatial information. In Chapter 3, I present some of the technical challenges and possible solutions of handling geographically embedded datasets in the magnitude of billions of points. In Chapter 4, I focus on the Bitcoin digital currency system, which presents a unique opportunity to study the details of monetary transactions on the everyday scales. Focusing on the statistics of the growth of the system, I investigate the role of preferential attachment, and the possibility to uncover relevant structural changes in the transaction network and relate them to external measures. In Chapter 5, I deal with the topic of
dimensionality of networks; I give an overview of recent results and methodology and then introduce a new definition of dimensionality applicable to networks with partial spatial embedding. Apart from reviewing the basic theoretical foundations, I present results of extensive simulations on both synthetic and real-world networks.

1.1 Statistical physics, complex systems and networks

In the past decades, physicists successfully participated in a diverse range of interdisciplinary research. Using concepts from statistical physics and applying the approach of mathematical modeling developed on physical systems to a wider range of complex phenomena has lead to important results in various fields; some of these are closer to traditional research foci in physics, while others are more apart. Examples include statistical models of various biological phenomena, including flocking and herding behavior [197, 146, 73, 198, 92], mathematical models for ecology, evolution and the emergence of cooperation [139, 201, 178, 185]; modeling pattern formation in various systems [85, 52, 122] and the application of the concept of fractality in both the geometric and dynamic sense [133, 15, 87]; chaos theory [59] to gain a better understanding of systems with a very high dimensionality [13]; models of the economy and various social phenomena [134, 45, 173, 196, 104, 84]; and the study of networks, which themselves can be used to describe a wide range of complex systems around us [149, 29, 66, 30, 97, 142]. Many of these approaches make use of the concepts of statistical physics, aiming to compute relevant macroscopic parameters of a system made up of many interacting parts. Also, concepts from the theory of critical behavior [75, 96] can be employed in many cases to interpret scaling laws which can be established empirically [69, 60, 66, 33, 129, 16].

When applying the methodology of statistical physics to other systems, there can be important differences presenting new challenges. Indeed, in many cases, an approach which takes the “thermodynamical limit” gives only an approximation or even just an analogy instead of a faithful description. There could be mainly two reasons behind it; first, while complex systems around us are often made up of a very high number of interacting parts, these numbers can still not be large enough to make individual fluctuations irrelevant. While in a system made up of $\sim 10^{23}$ particles, statistical fluctuations of macroscopic properties are taken care of by the central limit theorem and also boundary effects can in most cases be neglected, in a system made up of only a few million components, even standard fluctuations can have significance. Second, in many cases, interactions between the parts of a system can themselves present more complexity than in physical systems which were the standard focus of statistical physics, often resulting in macroscopic correlations requiring special treatment. As a consequence, the term “complex systems” is used when referring to most of the previous examples.

The focus of this thesis is the statistical analysis of human behavior employing concepts from physical sciences; when compared to many of the other complex systems which has been a focus of physicists’ attention, giving a statistical description of human behavior is highly challenging even if only some aspects are considered; theory can result in only limited number of predictive results, limiting the set of relevant questions which science can hope to answer. An ambitious goal would be something like the scientific field of psychohistory speculated by science fiction writer Isaac Asimov, where behavior of a very large group of people is predicted from a statistical behavior of individuals analogously to thermodynamics [17]. However, in reality, usually much less can be expected, due to the facts that behavior and interactions among humans are much more complex than atomic
systems studied by statistical physics. Nevertheless, statistical analysis of human behavior
can be a valuable tool, and empirical models can be of great significance [92, 129, 196, 58,
128, 155, 86, 204, 84, 45]. In many cases, even theoretical work in social systems means
simulations of simplified models of some of the studied phenomena [92, 73, 66, 196, 203,
27, 167, 123].

Many complex systems around us can be represented as networks [26, 151, 29, 97, 66,
30, 55, 29, 149]. Usually, nodes represent basic parts of an interacting system, while links
or edges correspond to possible relevant interactions. In some cases, this seems a natural
way to look at the system; e.g. people use the phrase social network to refer to relations
among individuals in a self-evident way nowadays. Also, in many cases, the notation of
nodes and links has a direct physical meaning e.g. when considering transportation or
communication networks [169, 137, 206]. In other cases, the network representation is
more abstract, e.g. interaction among proteins in a cell can be analyzed from a network
perspective [25, 110]. A common aspect is then that network topology can be a relevant
factor in the dynamics of the complex system; thus, researchers of diverse fields can benefit
from common results in many cases. Also, the network representation can help by giving
results for the constraints of the behavior of a system made up of parts interacting in
possibly complex ways [13, 9, 10, 14, 102, 29]. While networks have been a focus of interest
from mathematicians since decades – often referred to as graphs [130] – the topic gained
more widespread attention in the last two decades, fueled by the availability of network
datasets in several diverse fields of interest. Notable examples include biochemistry, where
possible interactions among a very large number of organic molecules were explored with
new, high-throughput methods [110, 25]; information technology where efficient operation of
networked devices is an important goal [137, 158, 206, 82], sociology where obtaining large-
scale information on social relations among individuals is just becoming a reality [142, 50,
116]; or finance, where the structure of relations among banks and firms can have a strong
impact on the stability of the economy [46, 43, 42, 173, 156]. In many of these cases, one
can hope that the network representation grasps the fundamental structure of the studied
system, giving a valuable tool to researchers for performing analysis.

Apart from conceptual similarities, many networks obtained from diverse sources share
common aspects in their structure. These include low-density or sparseness (only a small
fraction of possibilities is present), small average diameter, heterogeneous degree distrib-
utions, relatively high clustering [124, 151]. On the other hand, there are still significant
differences, e.g. in assortativity [150, 66, 25], fractality [181, 182], spectral properties [72, 82],
community structure [155, 116] or relative abundance of motifs [112, 131, 11]. Assessing the
relevance of the differences can then help in better understanding the network structure
and its relation to dynamics of the studies complex system [35, 166, 171, 179, 7].

1.2 The data science paradigm

The phrase “data-intensive science” was first popularized by Jim Gray and colleagues in
the early 2000s, mainly in relation with new, large-scale datasets which became available
in astronomy [89, 186]. While it can be viewed as the “Fourth Paradigm” [95], methodology
of “data science” follows the traditional scientific method, with high emphasis on the pro-
cess of obtaining and handling data due to the increased importance and resources needed
for this task. More generally, over the course of centuries and millennia, the methodology
of science has evolved into a general framework, which is a continuous collaboration of
theory and experiments. Throughout the history of science, the relative importance of the
two varied based on resources available and the general background of researchers. Also, the effort needed for different aspects of it depends on the studied phenomena. E.g. in some cases, experiments might be relatively easy to perform, and theoretical results hard to get or vice versa. Of course, the focus of theoretical work is in most cases greatly influenced by the experimental possibilities of testing the predictions. Also, every aspect is affected by the availability of technological advances. Theory in many cases is increasingly supplemented by extensive numerical simulations, and experiments or measurements are performed utilizing complex data collection and processing techniques. With the possibility to collect massive amounts of data, these processes gained growing importance, and research involving them is referred to as data-intensive science or data science. The advances in astronomy in the past two decades are a typical example of how advances in measurement techniques and data collection affect the scientific methods. Also, microbiology is currently experiencing something which can be thought of as a “data explosion” thanks to advances in genome sequencing methods, which result in an unprecedented volume of results from various studies where the DNA sequencing of the subjects can now be carried out with rapidly decreasing costs.

Focusing on the statistical analysis of human behavior, the adaptation of digital technologies presents many opportunities in gaining data on previously unprecedented scales. Indeed, empirical work considering humans was previously much more limited than in other fields such as physics, chemistry or biology. There are three main factors in this. First, experiments or measurements are generally much more costly both in terms of resources and work effort required. In many cases, performing just one measurement on a large enough sample is already barely possible, while ideally there could be multiple realizations. Also, in many cases, multiple realizations are not possible practically or fundamentally. A second important concern is the ethical or moral implications of performing certain experiments on humans or even animals. The third concern closely related to the second is possible systematic errors arising from the experimental setup. I.e. it has been found in many studies that subjects’ behavior is affected by their understanding of the scientific goals and merit of the study they participate in [174], or even by the fact that they are participating in a scientific experiment [32]. Circumventing this concerns by performing an experiment on an uninformed set of participants or deceiving subjects is in many cases not approvable, or at least questionable morally or not feasible practically. A famous case which led to many debates in these aspects, were Milgram’s obedience experiments in which participants were asked to administer what they believed were electric shocks to others in a simulated setting [140, 36].

Based on these challenges, experimental and empirical work in social sciences was much more limited than in physical sciences, also limiting the set of studied questions. One of the earliest alleged “scientific” experiments on human subjects is attributed to ancient Egyptians, who supposedly raised a pair of children without any verbal interaction so as to study the language the children invent by themselves [93], something which would be considered highly unethical by modern standards. Throughout history, there have been several examples of people trying to build utopistic societies in communes, which again can be considered early social experiments in some fashion, although in many cases lacking any serious scientific foundation [135]. With the emergence of psychology and sociology as scientific fields by the early 20th century, experiments and measurements on human behavior were started to be performed in a more rigorous fashion, with more clear statement of the studied problem. Of course, obtaining data from large enough sam-

1See e.g. http://www.1000genomes.org/data
amples has always been problematic due to constraints in resources. Famous early examples include the empirical studies which lead to the "two-step flow of communication" hypothesis of information spreading and public opinion formation [105], the original small-world experiments [141, 193] and related research [86]. With the emergence of new large-scale data-collection possibilities, researchers can "repeat" some of these classic experiments on massive scales or revisit the same questions using new approaches. For example, the massive data generated by users of the Twitter microblogging platform was used to study information spreading and opinion dynamics in various contexts [123, 145, 202, 19, 204], while Milgram's famous small-world experiment was repeated by using emails as a method of communication with the number of participants in the range of tens of thousands [64], and via computer simulation of the routing process on geographically embedded social networks [125, 8, 94].

A special case among these new possibilities is passive measurements on human behavior. Throughout this thesis, by passive measurement, I mean any data collection and analysis procedure which does not require active participation from the subjects, and does not affect them significantly, only data which they produce normally while using a product or service is used. This might include social media activity, mobile network usage, GPS traces, financial transactions, etc., which are regularly conducted electronically nowadays. Since these datasets are generated throughout the normal functioning of these systems, concerns about the measurement altering the behavior of the subjects are minimal, while questions of research ethics normally mean implementing proper ways to inform subjects, gain their consent and protect their privacy. These can be thought to be in contrast to traditional methods of measurements, e.g. surveys and experiments usually conducted in controlled artificial environments. Of course, the distinction is not perfect; the design of new products or services in many cases includes considering data collection possibilities, with the business model of service providers' incorporating various uses of data generated by users. Also, there is the question whether people's behavior in some of these online systems is a good proxy for studying their offline behavior; these concerns can be mitigated if the system studied already became a part of the individuals' regular life, like popular online social network providers nowadays.

Clearly, the data-intensive approach can result in big advancements in social sciences. Still, the previously mentioned three main challenges need to be taken into account and their relevance assessed. The most straightforward improvement is clearly in the data collection procedures themselves. Thanks to advances in information technology, sample sizes can be increased by several orders of magnitude. Of course, there are inherent limitations, as large-scale studies can only be carried out if the data processing can be automated; there remain many cases where manual data processing, e.g. conducting in-depth interviews with subjects is necessary. In many cases, the problem of ethics and systematic errors can also be better handled than with traditional methods, but the concern of research ethics is still present, especially when designing large-scale experiments. In the case of passive measurements, the main question is about privacy and consent. While the possibility of aggregating large volumes of data seems frightening in many aspects, the main concern is usually not the usage for scientific purposes. In many studies, privacy issues can be better handled than in survey-based methods, i.e. researchers need not access any data which is not relevant to their study and can work on data aggregated and anonymized during the data collection phase. Also, by collecting data in an automated way, concerns about sampling bias, and reliability of answers provided by participants can be mitigated. Examples include studies on human mobility and activity patterns, where mobile
network, public transport or vehicle GPS sensor operators only need to share aggregated or adequately anonymized data with researchers for them to be able to perform statistical analysis [157, 88, 179, 169, 84, 164, 128, 112, 104]; using movement of money as a proxy for human travel [40, 192, 39]; analyzing behavior of players in a massive multiplayer computer game [177]; analysis of the transaction history of the Bitcoin digital currency system, where this kind of data is public due to the nature of the system [4, 5, 168, 165]; or focusing on public user activity in online social networks (OSNs) [18, 20, 19, 204, 202, 3, 71, 167, 200, 142]. In the case of performing experiments, the possible biases introduced by participant having to be recruited, incentivized and any difference between behavior in a real-world and experimental setup are still present, although the possibility of easily obtaining very large samples of data can still improve the statistical analysis. Large-scale online experiments include repeating Milgram’s small-world experiments with email communication [64], analyzing traders’ behavior in an artificial market [194], the dynamics of musical taste and it’s susceptibility to external influence [170], or exploring the possibility of predicting people’s personality from their online activities [207], where subjects were recruited through Facebook to share their likes with the researchers. A highly controversial recent study was undertaken by the Facebook social network, where the news feed of a set of users was systematically altered to find out how the emotional content of social media activity is influenced by the emotional content of posts seen by the users [113]. While designers of the study stated that their goal was to “improve user experience”, and the terms of service of Facebook includes that the service provider can use any algorithm for selecting the content which displayed to its users, many criticized the study as effectively being an experiment with the emotions of uninformed subjects and as such being unethical. Previous experiments including manipulations of the news feed of users, although not in relation to emotions were carried out by Facebook with less controversy [20].

Taking a look at recent studies, it is clear that study human behavior is a subject which is currently being transformed by the data-intensive science approach as a consequence recent developments in data collection and processing and the wide-spread adaptation of new technologies. It is clear however, that technological developments and changes do not by themselves solve the challenges; nevertheless, they provide exciting new opportunities to aid in modeling and understanding complex phenomena present in society around us. The inherent complexity and the moral and practical limitations of performing experiments on human subjects make research social sciences and related fields a challenging task. Also, the interdisciplinary nature of some of the problems studied and the employed methodology present the challenge of implementing a collaboration among researchers with diverse scientific backgrounds. Many people working as data scientists come with a physics, mathematics or computer science background, which indeed enables them to work with large-scale datasets efficiently and build mathematical or computational models of complex phenomena, but this can also mean the lack of common concepts in sociology and difficulties in communicating results to people with different scientific backgrounds. In this thesis, I present a physicist’s view on some topics of interest in the interdisciplinary fields of complexity sciences and network science which arise when studying human-made complex systems around us. I hope that some of the work presented here will indeed be relevant to a wide audience of researchers from diverse scientific disciplines.
Chapter 2

Large-scale geographic features of language use

In this chapter, I present a study analyzing the content of hundreds of millions of Twitter messages posted in the United States of America with accompanying geographic information. More specifically, I analyze the spatial variation of the content of tweets, and find that relevant geo-social features can be found using unsupervised classification and clustering. I first review some of recent advances in analyzing large corpora of texts and extracting geospatial features, and then present the applied Robust PCA method and relate it to them. I then present the results gained on a set of over 200 million tweets from the USA and compare these with an other relevant text mining method. I show the robustness of the method by zooming in to a higher resolution dataset of about 15 million tweets from New York City.

The core material presented in this chapter appeared in Ref. [3].

2.1 Related work

With the rapidly growing corpus of digitally available textual data, there has been a significant interest in unsupervised language processing techniques. Principal component analysis (PCA) [159] and the closely related latent semantic analysis (LSA) [120] have been widely and successfully applied to various classes of documents to identify relations among them, e.g. cluster the documents according to topics or implement semantic indexing [62, 76, 31]. Nowadays, LSA is employed by many search engines to improve results based on clustering documents to topics of interest. While the basic recipe for LSA is well known, each data source might require special treatment according to its unique characteristic.

A valuable source of digital textual data is the corpus of messages posted by users to online social network (OSN) websites. With recent research focusing on data shared by the users, the possibility to apply text mining methods to this corpus was explored from various different perspectives, including new challenges raised by OSN texts to natural language processing methods [77], exploring whether personality can be determined from user posts [205, 172], news dynamics [123], changes in language use due to excitement [189], or monitoring epidemics [119]. Combining the content of messages posted by users with the network topology was used to study the flow or diffusion of news and memes and influence in opinion dynamics [20, 167, 19, 145, 202]. Focusing on geo-tagged data, various studies analyzed spatial variation in the message texts and its applicability to several different questions, including user localization based on the content of their posts [49, 18],
empirical analysis of the geographic diffusion of novel words, phrases, trends and topics of interest [74, 71], measuring public mood [143], its relation to heart disease rates [70], religion [48] or unemployment [127]. Most of these studies include the modeling of a specific phenomenon with the help of textual or meta-information present in OSN posts. In the following, I focus on the possibility to apply unsupervised learning, i.e. LSA or its properly adapted version to a corpus of geo-tagged tweets so as to identify regional characteristics of language use. Connection to previous studies focusing on sentiment analysis [143, 63] is also established.

The approach presented here is to consider a medium-sized geographical region as one "document" by concatenating the text of all tweets posted there. A term-document matrix is constructed, whose rows correspond to the geographical regions of interest, the columns correspond to the words, and elements indicate how many times each word was used in that region. This method ignores relations between words, possible variation present inside the regions, and the possibility to detect topics which are present globally. This is in accordance with the current goals, which are to identify the regional variations. Detection of topics without a clear regional focus could be pursued using other natural language processing methods [123], and detection of local variations can be implemented by "zooming in" to regions with a high number of tweets to obtain a finer-grained picture of language use.

In many real-world PCA applications, the data matrix (i.e. the term-document here) can be modeled as the weighted sum of two parts, one of which is sparse, while the other is low-rank. The low-rank part can be thought of as some "background" component, while the sparse part can either contain the relevant data or can be composed of nontrivial (non-Gaussian) outliers. Depending on the nature of the problem studied, either one or both of them might be of interest. In these cases, it is desirable to be able to separate these two parts, and to analyze them separately. Also, if elements in the sparse part have large magnitude, the principal components found will be dominated by these outliers and revealing the low-rank structure present in the data can prove difficult. In the case of our corpus of Twitter messages, we can expect the data matrix to be indeed separable into these two parts. We can expect the low-rank component to represent the true variation in language use (i.e. usage frequencies of words will possibly be different in different geographic regions but can be expected to have smooth spatial variations), and the sparse part to contain highly localized topics of interest e.g. landmarks. To deal with these issues, the Principal Component Pursuit technique developed by Candès et.al. is employed, which achieves the separation of the original data matrix into its sparse and low-rank parts effectively [126, 44].

2.2 Data

2.2.1 Twitter

Twitter is an online microblogging platform, where users can post short messages called tweets with a maximum length of 140 characters. Users can access Twitter by signing in on its website, or with the use of one of various client applications on their computers or smartphones. Users can establish social connections by following each other, thus creating a directed link between them. If user A follows user B, then A is notified of B’s tweets. Reciprocal follower relations are sometimes referred to as friends. Normally, tweets are posted as public, which means that anyone (even without a Twitter account) can see them,
e.g. by navigating to the public timeline of the user which can be accessed via a public URL including the user’s Twitter username. Tweets can also be marked as private, when only the user’s followers (or a subset of them) can see it. Users also have the option of attaching geographic coordinates to their tweets, which is referred to as geo-tagging them. This is usually achieved by using the coordinates reported by the poster’s phone or other smart device. Twitter client applications usually have on option where the user can set if they would like this feature to be enabled or not. While most geo-tagged tweets come from GPS-enabled smart devices, users have the option to attach arbitrary coordinates to their tweets, as Twitter enables third-party client programs to be used with their service meaning that they do not have control over the validity of the provided information. This is utilized by e.g. different service providers who have the option of posting the coordinates relevant to the content of their tweet instead of their location with their messages.

Apart from browsing public messages on their website, Twitter and its partners provides several options for anyone to access the content of public messages posted by its users. While the focus is of course commercialization of the content and relevant analytics, a sample of about 0.5% of all messages is available for anyone to download for free using their Streaming API. Users of this service have the option of specifying filters for their download stream. These filters can either contain a keyword or a geographic bounding box for focusing on tweets related to a topic or posted in a specified region of interest. The API also allows specifying the whole world as the “bounding box”, essentially filtering the stream to contain geo-tagged tweets from anywhere around the globe. Rate limits for the filtered stream are the same as for the whole, meaning that if matching tweets are not very numerous, a significant portion can be accessed this way. A very useful practical example is filtering for geo-tagged tweets: it is estimated that a total of about 1.5% of tweets are geo-tagged, meaning that a download stream filtered for any tweet with coordinates is likely to result in approximately one third of all geo-tagged messages. If such a filter is focused on a small or medium region, a nearly complete set of geo-tagged tweets posted in that region can be obtained [144]. While being relatively open by allowing anyone to access a portion of their data, Twitter does impose constraints of sharing data obtained from them and also does not provide a clear description on their sampling method. This essentially makes estimating potential biases in the Twitter sample hard to estimate. Of course, using geo-tagged tweets as a basis of a study already has the drawback of potential bias due to user participation, i.e. any conclusion gained from its analysis will only apply to the part of the population using Twitter and enabling their GPS coordinates to be posted. Having understood this limitation, it is then a relevant scientific question if the Twitter sample can be used to gain knowledge on society or human behavior. With respect to some questions it has already been demonstrated that the free sample of geo-tagged tweets can be a valuable resource [127, 70, 172, 48, 119].

Apart from using the tweets themselves, relations among users can supplement any analysis of the tweets. Effectively, two directed networks can be constructed from Twitter’s data. First, there are the follower relations, which represent general interest among users. These can be obtained with specific queries from the Twitter API. Data collection is in this case heavily limited by rate limits, which can partly be overcome by using multiple developer accounts. This means that obtaining the complete follower graph is not feasible in practice, but subgraphs of relevant size can be obtained. Second, users can explicitly mention other users in their tweets. The widely accepted form for this is putting an ‘@’ sign in front of the mentioned user’s username to indicate that it is a mention. Twitter

[1]https://dev.twitter.com/streaming/overview
registers this, and provides mentions present in the tweet as metadata when downloading the stream. This results in a weighted graph, as the number of consecutive mentions can be associated to each edge.

In this thesis, I use the data collected from Twitter over the course of three years as part of a larger project. The data includes an unfiltered sample stream and a filtered stream where geo-tagged tweets from anywhere around the world were collected. The total number of tweets is over four billion with approximately half of them containing geographic coordinates. The data also includes the follower relations among seven million users who regularly post geo-tagged tweets which was obtained through a concentrated campaign. More information on the data, collection methods and the database used for storage and some of the analysis can be found in Ref. [6] and in Chapter 3. In this chapter, I use a subset of the geo-tagged tweets, while in Chapter 5, the follower and mention graph structure is analyzed among other graph datasets.

2.2.2 Text pre-processing

The USA is subdivided into medium-sized regions using the Hierarchical Triangular Mesh (HTM) scheme [187]. This is a recursive subdivision of the surface of the Earth into spherical triangles called trixels. In our case, the USA is covered by 566 trixels, with a mean area of \( \approx 15,000 \text{ km}^2 \) each (this corresponds to level 6 in the recursive subdivision, with the approximate linear size of trixels being 100 km). This covering, and the distribution of tweets in the trixels is displayed in Fig. 2.1. For each trixel, we concatenate the content of all tweets posted in its area and compile word frequency distributions. From these trixel-level frequencies, we construct a word occurrence matrix \( W \) whose \( W_{ij} \) element is the number of individual occurrences of the \( i \)-th word in the \( j \)-th trixel. To improve the quality of the dataset, we exclude trixels with less than 10000 occurances of at least 1000 individual words and also words with less than 10000 occurances in at least 10 individual trixels. This way, there remain 515 trixels and 14871 words. To account for the highly heterogeneous distribution of the number of tweets, the word occurrence matrix is normalized on the trixel level: \( X_{ij} \equiv W_{ij}/\sum_k W_{kj} \). Due to the structure of our data matrix, and the employed Robust PCA method, we do not subtract averages from the data; of course, this will result in that average word frequencies dominate the first principal component.

Furthermore, we compile a higher resolution data matrix from the same dataset focusing on New York City, with a total of 12.25 million tweets. In this case, we use level 13 HTM trixels; New York City is then covered by 592 trixels with a mean area of 1.25 \text{ km}^2 each.

2.3 Robust Principal Component Analysis

Principal component analysis is carried out by computing the singular value decomposition (SVD) of the data matrix \( X \):

\[
X = U \Sigma V^T
\]  

(2.1)

The diagonal matrix \( \Sigma \geq 0 \) contains the singular values, while the columns of the \( U \) and \( V \) matrices contain the principal components, which are orthonormal vectors, i.e. \( U^T U = V V^T = I \). The decomposition is unique up to a permutation of the columns of \( U \) and \( V \) and the entries of \( \Sigma \) [159, 163]. It is customary to order the principal components such that the corresponding singular values are decreasing in magnitude, i.e. \( \Sigma_{jj} \geq \Sigma_{kk} \) for \( j < k \).
Figure 2.1: Spatial distribution of tweets. Trixels included in the analysis and number of individual tweets in them are displayed. Note that the colorscale is logarithmic.

Figure 2.2: Distribution of tweets in New York. Number of tweets from the corpus in the trixels included for New York City. Note that the colorscale is logarithmic.
This way, the principal components reveal the sources of variation in the data, ordered by their magnitude of contribution.

In the case of our dataset, principal components can be interpreted as relevant sources of geographical variation of language use. For each of these, the values in the $U$ matrix are weights associated to the words in our corpus, identifying how important the variation in the use of that word is relevant to that principal component. Values in the $V$ matrix can be thought of as spatial distributions which show how usage frequencies of words in that component vary geographically. To analyze the results, we plot these distributions on the map of the USA (see e.g. Fig. 2.3) and try to interpret the spatial variation in terms of geography. We complement this by inspecting the weights associated to the individual words in each component (i.e. the columns of the $U$ matrix).

While the classic PCA method is useful for detecting structure in the data, in some cases, it might not give optimal results. One such case is that when there are outliers in the data; in this case, the main principal components are dominated by them and identifying a possible low-rank structure becomes more challenging. In the case of Twitter messages, localized outliers include spammers, advertisement, job opportunities, weather forecast services and announcements, and of course local tourist attractions. Filtering out these can also prove difficult, as in many cases, it is not straightforward to estimate the relevance of components in the raw data a priori. Also, this would result in leaving out relevant data, and in general would contrast with the unsupervised nature of LSA. A more favorable approach is using a similarly unsupervised method applied to the raw data matrix to find outliers automatically, which can be later inspected and their relevance interpreted. This can be obtained by the Principal Component Pursuit technique [126, 44], which effectively separates the data matrix into two parts:

$$X = X^S + X^{LR},$$

where $X^S$ is a sparse matrix and $X^{LR}$ contains the dense but low-rank part of the data. This is achieved by minimizing the sum

$$\lambda \|X^S\|_1 + \|X^{LR}\|_\sigma,$$

where for a matrix $X$ of dimensions $n_1 \times n_2$ with $n_1 \geq n_2$, $\lambda \equiv 1/\sqrt{n_1}$, and the norms are the $l_1$ and nuclear norms respectively:

$$\|X\|_1 = \sum_{ij} |X_{ij}|$$

$$\|X\|_\sigma = \sum_i \sigma_i(X).$$

Here $\sigma_i(X)$ denotes the $i$-th singular value of $X$. To obtain the results we use the Matlab code developed by Lin et al.\footnote{the code is available at: http://perception.csl.illinois.edu/matrix-rank/sample_code.html.} implementing the inexact augmented Lagrangian method [126].

Having thus separated the data matrix to its low-rank and sparse components, we carry out an SVD for both $X^S$ and $X^{LR}$, and analyze the results simultaneously. We expect the principal components of $X^S$ to contain information about highly localized trends (i.e. features specific to only one or few cells), and the components of $X^{LR}$ to contain the possibly smooth variations in language use. Note that this can mean an indeed smooth variation, but can also mean a variation in the relative abundance of several discrete language-use variants.
we can conclude that some of these can be considered “noise”: meteorological stations, outlier cells in the classic PCA results and in the sparse part, and the related word clouds, vectors, which mimic some of the geographic and social features of the USA. Looking at the found in the sparse component, while the low-rank component contains smoothly varying the rest. Comparing it to the PCA Pursuit results, these are all very similar to the features cells which are isolated centers where the content of the tweets differ significantly from the PCA results of the two methods. The results of the classic PCA are dominated by outliers – words are in abundance.

In Figs. 2.3–2.8, the first six principal components of the term-document matrix compiled for the contiguous USA are shown. Spatial distribution of weights in singular vectors of the original data matrix, and the sparse and low-rank components are shown separately for each component. 50 words with the largest and smallest weights are displayed as word clouds in the case of the sparse and low-rank components. Size of a word corresponds to its relative weight, while color is used in accordance with the maps. As we deliberately chose not to subtract any mean values from the data, the first component does not seem to contain any relevant structure. As the mean values in the original data matrix are the same for each cell, the outlier cells displayed here are the ones where the otherwise rare words are in abundance.

In the case of the higher, relevant components, there is a fundamental difference in the results of the two methods. The results of the classic PCA are dominated by outliers – cells which are isolated centers where the content of the tweets differ significantly from the rest. Comparing it to the PCA Pursuit results, these are all very similar to the features found in the sparse component, while the low-rank component contains smoothly varying vectors, which mimic some of the geographic and social features of the USA. Looking at the outlier cells in the classic PCA results and in the sparse part, and the related word clouds, we can conclude that some of these can be considered “noise”: meteorological stations,

Figure 2.3: First principal component. (a) original data matrix, (b) sparse part, (c) low-rank part, (d) words with negative weight in the sparse part (e) words with positive weight in the sparse part, (f) words with negative weight in the low-rank part, (g) words with positive weight in the low-rank part.

2.4 Language use features in the USA

In Figs. 2.3–2.8, the first six principal components of the term-document matrix compiled for the contiguous USA are shown. Spatial distribution of weights in singular vectors of the original data matrix, and the sparse and low-rank components are shown separately for each component. 50 words with the largest and smallest weights are displayed as word clouds in the case of the sparse and low-rank components. Size of a word corresponds to its relative weight, while color is used in accordance with the maps. As we deliberately chose not to subtract any mean values from the data, the first component does not seem to contain any relevant structure. As the mean values in the original data matrix are the same for each cell, the outlier cells displayed here are the ones where the otherwise rare words are in abundance.

In the case of the higher, relevant components, there is a fundamental difference in the results of the two methods. The results of the classic PCA are dominated by outliers – cells which are isolated centers where the content of the tweets differ significantly from the rest. Comparing it to the PCA Pursuit results, these are all very similar to the features found in the sparse component, while the low-rank component contains smoothly varying vectors, which mimic some of the geographic and social features of the USA. Looking at the outlier cells in the classic PCA results and in the sparse part, and the related word clouds, we can conclude that some of these can be considered “noise”: meteorological stations,

---

3The word cloud generator used is available at http://www.vo.elte.hu/pcapaper/wordcloud; it is based on the Wordle algorithm (http://www.wordle.net/) and a previous open-source implementation of it (https://github.com/whydoidoit/WordCloud).
Figure 2.4: Second principal component. (a) original data matrix, (b) sparse part, (c) low-rank part, (d) words with negative weight in the sparse part (e) words with positive weight in the sparse part, (f) words with negative weight in the low-rank part, (g) words with positive weight in the low-rank part.

Figure 2.5: Third principal component. (a) original data matrix, (b) sparse part, (c) low-rank part, (d) words with negative weight in the sparse part (e) words with positive weight in the sparse part, (f) words with negative weight in the low-rank part, (g) words with positive weight in the low-rank part.
Figure 2.6: Fourth principal component. (a) original data matrix, (b) sparse part, (c) low-rank part, (d) words with negative weight in the sparse part (e) words with positive weight in the sparse part, (f) words with negative weight in the low-rank part, (g) words with positive weight in the low-rank part.

Figure 2.7: Fifth principal component. (a) original data matrix, (b) sparse part, (c) low-rank part, (d) words with negative weight in the sparse part (e) words with positive weight in the sparse part, (f) words with negative weight in the low-rank part, (g) words with positive weight in the low-rank part.
service announcements and tweets of job advertisements. In many cases, outlier cells are those which contain less tweets than the average, thus the same volume of e.g. weather related tweets will represent a higher ratio giving higher than average frequencies to their typical words. An example of this can be seen in the case of the fourth component (see Fig. 2.6(a), (b) and (d)), where the outlier (blue) cells are the Nevada desert where a large fraction of tweets are generated by an automatic reporting service about minor earthquakes and related information. While the total number of earthquake–related tweets (and the occurrences of the ‘#earthquake’ hashtag) are even higher in California, the relative frequencies are much higher in these cells. Looking at the word clouds (Fig. 2.6(d)), one can spot the words ‘poker’ and ‘tournament’; looking at the geography in more detail, it turns out that Las Vegas and Reno, two cities known for their gambling opportunities in neighboring trixels in the south-east and north-west directions respectively. We speculate that these come from people traveling to one of these cities. On the other hand, some of the outliers show real localized attractions: e.g. in the case of the sixth component, the “outlier” cell is the location of the Grand Canyon, where a significant proportion of tweets indeed mention the canyon itself.

Looking at the principal components of the low-rank part obtained by PCA Pursuit, we can try to identify some of the patterns in them. In the case of the second component, there is an apparent spatial structure, with the southern USA being separated from the rest. If we look at the spatial structure present in the results of the classic PCA and the sparse part more closely, we can identify some of the structure present here in them too (Figs. 2.4(a) and (b); note that the color scale is inverted on these), although these are overshadowed by weather–related outlier trixels. This is not the case for the higher components where classic
PCA results and the sparse part are clearly dominated by outliers. Examining the words giving the largest contribution (i.e. the columns of the $U$ matrix, see Fig. 2.4(f) and (g)), we can identify that this separation is mainly due to swear words, slang and abbreviations, some specific to online social networks (e.g. ‘oomf’ can mean ‘one of my followers’). It seems clear that this means that there is a relevant difference in the language people use on online social networks in different parts of the country. Among the terms, somewhat striking is the presence of the presence of the word ‘nigga’, is considered taboo for the general population, while having a slang use by some African-Americans. Looking at a small random sample of tweets containing the term, it seems that it is mainly used as a neutral slang word to refer to people in general. This indicates that this component might be related to language use variations specific to African-Americans. A further notable example here for language use traits is ‘goodmorning’, which seems to be a misspelling quite abundant in some parts of the USA. Looking at a small random sample of tweets, it seems that it is indeed used instead of saying ‘good morning’; deciding whether it’s use is intentional (like some other examples of nonstandard spelling in this component, e.g. ‘kno’ instead of ‘know’, ‘tryna’ instead of ‘try to’ or ‘trying to’) or an unintentional common spelling error would require more detailed knowledge of language and culture in the affected regions.

In the third component, the major cities are separated from the countryside: words like ‘Starbucks’ ‘Grill’, ‘sushi’, ‘mall’, ‘Bar’ or ‘downtown’ are opposed by words like ‘truck’, ‘town’, general neutral words and weather-related words. The latter come from weather reporting stations whose relative density is significantly higher in rural areas. Looking at the map (Fig. 2.5(c)), we can indeed identify areas around important cities as the blue trixels which correspond to the related words. In the case of the fourth component, the spatial structure is still strong, while the language differences are not that apparent. In the case of the fifth component, words related to university studies are opposed with words related to going on a vacation. The former seem to be more abundant in the north-east, while the latter mainly in the south and west, while trixels containing Chicago, New York City and Washington also stand out somewhat in the north-eastern part. In the case of the sixth component, we see two main factors: religion-related words are opposed by excessive use of profanity. We speculate that this can be related, i.e. that people who tweet more related to religion tend to try to avoid profanity or replace it by less strong options, e.g. ‘crap’, ‘heck’, ‘dang’ or ‘freaking’ as apparent in the related word cloud (Fig. 2.8(g)). Looking at the corresponding map (Fig. 2.8(c)), the religious tweets are more abundant in the central USA, which is indeed sometimes referred to as the ‘bible belt’ [195, 208].

### 2.5 Testing scalability of RPCA on New York City

We have seen that we were able to identify the low-rank structure of variations in language use on the scale of the USA. To test if there are such variations present on a much smaller scale, and whether we can similarly detect some of these, we applied the same method to the region containing most of New York City (Fig. 2.2). Results obtained for this corpus are depicted on Figs. 2.9 and 2.10 (principal components of the low-rank part). These components roughly separate the city into its various districts. The main features are the separation of Manhattan and parts of Brooklyn from the rest of the city due to different aspects in language-use differences. In the second component, main features can be linked to tourists and downtown cultural life; this is countered by more general language, notably relatively more swearing in less central districts. The third component is somewhat similar,
Figure 2.9: Results of PCA Pursuit applied to New York City. First three components (from top to bottom), low-rank part.
but in this case, the most important feature is slang use (similarly to the 2nd component in the whole USA) and Spanish words to some extent which are found to be representative of Harlem, the Bronx, and central and east Brooklyn. In the fourth component, words related to culture and social life in Manhattan and Jamaica are countered by sports-related terms in the rest of the city. In the case of the fifth component, Spanish terms are countered by technology-related words with lower Manhattan, parts of Brooklyn and Jamaica showing an increased use of the latter.

We also examine whether the space of principal components obtained for New York City is similar to the components obtained for the whole USA. We project the word occurrence matrix of New York City into the space of principal components obtained for the whole USA: $V^{(proj)} = U^{(USA)} X^{(NY)}$, and then plot the obtained vectors on the map of New York (Fig. 2.11). We get that most components separate the city in a meaningful way, suggesting that some of the differences obtained at large scales are relevant on much smaller scales too. E.g. slang use in the second principal component of the whole USA is somewhat similar to the features of the third principal component of New York City (Figs. 2.11(b) and 2.9(c); note that color scale is opposite according to the sign of relevant singular vectors being opposite). Also, words related to urban lifestyle in the third component (Fig. 2.11(c)) give contributions mostly in Manhattan and nearby neighborhoods. These results are especially interesting given that in when performing the analysis for the whole USA, New York City was represented as a single trixel, giving an average contribution of the features found here.
2.6 Relation to other studies: Public mood assessment

Using written text to analyse the personality or mood of the writer has been studied previously in various contexts. From the perspective of psychology, medium-length texts can be used effectively to detect various personality traits, traditionally by manual evaluation of an expert. On the other hand, when using written text as the medium of personal or professional correspondence, humans readers are generally quite good at inferring the mood or emotions of the writer with relevance to the situation; this is of course much more efficient if the reader is already familiar with the writer. With the adoption of computers, information technology, and an increasing percentage of written communication occurring digitally, the question of utilizing computers and text mining methods to detect personality traits or emotional content in text is relevant from several different aspects. While achieving performance comparable to experts or even regular humans still seems a rather ambitious goal, the ability to easily process large amounts of digital text is something that can be exploited in many ways. While computers might not be able to infer personality in detail, some traits can be predicted with high accuracy in an automated fashion [207, 205, 172]; also, even simple word-count based methods can help experts by presenting statistics of the occurrences of relevant words [190]. In studies where only a less detailed perspective is needed, automated text mining based on established protocols can save significant amount
of researchers’ work of manual reading of large corpora. The ability to evaluate methods on large corpora can be useful in many ways, e.g. using computerized analysis can form a baseline for selecting subjects for manual evaluation in many cases. While some of the digital text produced by individuals is private, information that can be gained from public activities can be valuable in many cases. A relevant example is evaluating the emotional content of tweets related to a person, product or topic, which has already been part of marketing and political campaigns in recent years [21, 101, 81].

In this section, we focus on the recent study by Mitchell et al. [143], where a large corpus of geo-tagged tweets were used to calculate happiness scores of places and regions and relate these to demographic characteristics. Their method was based on word counts with per-word happiness scores a priori established using ratings provided by volunteers recruited through the Amazon Mechanical Turk service in a previous study [63]. The happiness score associated with a set of tweets can then be calculated as the average of scores of words occurring in them. They apply this method to tweets grouped by geographic region, i.e. by state or city. Results are then interpreted as sentiment of the Twitter-using population, with a focus on geographic variation and relation to demographic characteristics. Here, this methodology is applied to our corpus of geo-tagged tweets, and results are compared to the principal components gained with the previously presented methods. Possible interpretations and concerns are discussed then.

Using the previously established term-document matrix, we compute happiness scores with word weights from the Mechanical Turk dataset [63]. As comparison, we also make use of a previous, similar study, where words were annotated with valence scores based on ratings of university students [37]. Following the notation of the authors, we refer to the two datasets as the LabMT and ANEW (from “Affective norms for English words”) dataset respectively. The LabMT dataset contains a total of 10222 words, while the ANEW dataset contains 2475 words. Both datasets contain scores ranging from 1 to 9, with the mean being 5.375 and 5.206 and the standard deviation being 1.085 and 1.692 in the LabMT and ANEW case respectively. Following the methodology of Ref. [143], we omit words with scores between 4 and 6 as neutral or “noise” words. Also in line with Ref. [143], we omit the racial slur ‘nigger’ and its variations from the LabMT dataset, since the perceived valence can vary in great extent depending on the situation where it is used (note: the ANEW dataset did not contain this term or variations). We further limit the set of words used to words with at least 1000 occurrences in our corpus of tweets from the USA. Having carried out these filterings, there remain 3386 words in the LabMT dataset and 1375 words in the ANEW dataset. Among these, 837 words are common. Comparing the scores in this common subset, we find a high degree of similarity, with a correlation coefficient of 0.974 between the happiness and valence scores of the two datasets. Using these datasets, we then compute happiness and valence scores on the trixel level, i.e. on the same spatial aggregation level as the previous Robust PCA calculations. Results are displayed in Figs. 2.12(a) and (b) for the LabMT and ANEW datasets respectively. We see a clear spatial structure which is very similar in both cases in accordance with the high level of correlation of happiness and valence scores of words in the two datasets (the correlation coefficient is 0.875). Of course, words which are not common in the two datasets could result in more differences, but given the high degree of similarity in scores of common words, it is not surprising that using all words in both datasets still gives consistent spatial structure. Looking at the maps in Figs. 2.12(a) and (b), the main features are that the southern and south-eastern parts of the USA (i.e. parts of the states of Louisiana, Mississippi and Alabama) exhibit the lowest happiness or valence scores (apart from a.
Figure 2.12: Computed happiness scores in the trixels covering the USA. (a) happiness scores including words from the LabMT dataset [63]; (b) valence scores computed with words from the ANEW study [37]; (c) relative frequency of occurrences of the most profane words in the corpus from the LabMT set [63]; (d) happiness scores calculated with the LabMT words, but omitting profanity. Note that color scales (i.e. absolute numerical values corresponding to a given color) are different among the different subfigures; as the resulting happiness or valence scores span a very limited range with a few outliers, color scales were adjusted to represent the variation in the central 96% of data with high contrast. While absolute scores differ in the three maps ((a),(b) and (d)), the structure is very similar in all three cases. Also, the relative frequency of profanity shows a clear anti-correlation with happiness and valence scores. Despite this, removing profane words from the corpus does not result in significant alteration in structure, showing that the observed pattern is the result of more subtle language use differences.

Apart from these, the main visible feature is high values of happiness scores over the Rocky Mountains and at most of the coastlines. We speculate that these places being tourist destinations can contribute to these results. Results obtained here are consistent with previous results of Mitchell et al. [143], who carried out a similar analysis on the level of states.

Having thus reproduced the results of [143] on our corpus, we can say that we have found a clear spatial structure in language use, which can be related to the scores or weights of words in the LabMT and ANEW datasets. In relation to this thesis, two main questions arise:

1. Do the obtained happiness scores correspond to relevant measures of well-being, or sentiment of people? Are people for example in Wisconsin or Minnesota really happier...
on average than people in Louisiana or Mississippi? A side question would be whether a measure giving an average score for a given population is descriptive enough, or should we use more detailed measures. Given that we accept an average measure, the previous question then is about the applicability of these methods to real-world sentiment analysis of large groups of people, which would indeed be beneficial for many purposes.

2. How do these results relate to the principal components found with the methods presented earlier in this chapter? Clearly, the results presented in Fig. 2.12 show a relevant spatial variation in language use. We could expect that the same variation is present in the principal components too. We can then ask if there is a simple correspondence to one of the principal components, or a more complex relation is present instead.

Giving a definite answer to the first question is beyond the scope of the current study. This would require proficiency in experimental psychology and sociology, and access to much more diverse sources of data, including the results of more established surveys of life satisfaction measures. Of course, in this case, even classical data collection methods have their drawbacks, especially when considering the ability to compare results obtained from people with different cultural backgrounds. A notable previous study relating more traditional methodology of measuring well-being across the USA found opposite trends as the results presented here [154]; they found e.g. that in Louisiana and Mississippi, people’s well-being is higher than in Wisconsin or Minnesota. This discrepancy was noted by the authors of [143], but no possible explanation for this was attempted. Besides the sampling bias arising from the fact that the public Twitter stream is probably not a representative sample of the population of a region, the other main concern is that emotional content of words can greatly vary based on the context and the cultural background of the people involved in a conversation. This effectively means that even at the level of individual persons, it is debatable whether the differences detected with this methodology can be attributed to differences in well-being or happiness or to differences in style. We tested one possible aspect of it, based on our previous observation that the southeast USA presents an increased use of profanity. Identifying profane words in the LabMT USA dataset, we plotting the relative frequencies of these words on the map in Fig. 2.12(c). It is apparent by visual inspection that the spatial structure is similar to the distribution of happiness scores; the correlation coefficient between the two datasets is $-0.3647$. Determining if frequent swearing is a sign of general dissatisfaction with life of just a stylistic trait can be challenging; nevertheless, we did recalculate the happiness scores with excluding these words from the corpus. Results are displayed in Fig. 2.12(d); although average scores are somewhat higher (average difference is 0.099), the structure is strikingly the same (correlation coefficient is 0.967), meaning that more frequent use of profanity is just one aspect of language use differences which give rise to spatial variation in happiness scores. In addition to possible issues of differences in style between people from different regions, computed happiness or valence scores clearly depend on conversation topics, whose relation to general well-being might be not straightforward. Based on these arguments, I believe that the applicability of this methodology to detect the general mood of people still needs more careful evaluation. After establishing correct baselines and refined methodology, the ability to complement traditional methods of public mood assessment with practices based on the passive collection of large-scale public data available from OSN users still seems quite promising. Also, while comparing the happiness of people living in different places can be quite difficult, this methodology can be readily applied to detecting mood changes locally in time [63].

Apart from possible applicability to measuring public mood, the relation of the hap-
### Table 2.1: Correlation between happiness scores and RPCA results.

Left: correlation among the spatial structure of happiness scores (Fig. 2.12(a)) and principal components of the low-rank part (Figs. 2.3(c)–2.8(c)) and computed significance scores ($t$-scores). Right: correlation among the happiness scores of individual words and associated weights in the principal components. Significance scores were calculated using the assumption of 50 degrees of freedom. Correlation values with $t > 2.3264$ can be considered significant with $p < 0.01$ confidence (99%). Spatial correlation is significant in the case of the first two components, while correlation among happiness scores and word weights in the principal components is not significant for any of the components.

<table>
<thead>
<tr>
<th>component</th>
<th>correlation coeff.</th>
<th>$t$-score</th>
<th>component</th>
<th>correlation coeff.</th>
<th>$t$-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.3733</td>
<td>2.7877</td>
<td>1</td>
<td>-0.0559</td>
<td>-0.3882</td>
</tr>
<tr>
<td>2</td>
<td>-0.6721</td>
<td>-6.289</td>
<td>2</td>
<td>-0.1437</td>
<td>-1.0061</td>
</tr>
<tr>
<td>3</td>
<td>-0.2481</td>
<td>-1.7743</td>
<td>3</td>
<td>-0.1806</td>
<td>-1.2722</td>
</tr>
<tr>
<td>4</td>
<td>-0.1691</td>
<td>-1.1885</td>
<td>4</td>
<td>-0.1016</td>
<td>-0.7078</td>
</tr>
<tr>
<td>5</td>
<td>-0.2132</td>
<td>-1.512</td>
<td>5</td>
<td>-0.1605</td>
<td>-1.1264</td>
</tr>
<tr>
<td>6</td>
<td>0.2016</td>
<td>1.426</td>
<td>6</td>
<td>0.1058</td>
<td>0.7368</td>
</tr>
<tr>
<td>7</td>
<td>0.0405</td>
<td>0.2809</td>
<td>7</td>
<td>-0.0067</td>
<td>-0.0463</td>
</tr>
<tr>
<td>8</td>
<td>0.1507</td>
<td>1.0565</td>
<td>8</td>
<td>0.0212</td>
<td>0.1472</td>
</tr>
</tbody>
</table>

### 2.7 Conclusions and future work

In this chapter, I investigated the feasibility of principal component analysis to identify regional features of user content present in geotagged Twitter messages and related it to a recent relevant study of public mood based on word frequencies distributions. Using the **PCA Pursuit** technique, the low-rank and sparse parts present in the data were effectively...
separated, and some of the main features in both were identified. Examining the spatial distribution of the principal components of the low-rank part, it was found that there are indeed large-scale spatial variations present among Twitter users. I also investigated the scalability of this method and found that some of the principal components found for the whole USA are also relevant if we only consider the much smaller area of New York City.

The methods presented here can be the basis for studying language use on large-scale in the future. The ability to separate the word usage matrix into a low-rank and sparse component opens up many possibilities. The principal components of the sparse part can be used to identify regional or highly localized topics of interest, and also to identify sources of Twitter content generated by bots and services, as these can be easily spotted in otherwise low-density regions. On the other hand, the low-rank part shows relevant regional variations in language use. Exploring possible connections between language use differences revealed by the methods employed here and other demographic or social features will probably yield new insights into the dynamics of society and language. Understanding relevant sources of variation can also result in improvement of supervised models aimed at specific phenomena [143, 70, 127, 48].

On the other hand, the results of PCA might lead to more direct applications. Identifying locations of users based on the content of their messages has been previously evaluated using probabilistic approaches [49]. Now, using the space of PCA components, we might be able to estimate the location of texts (e.g. a small set of tweets from a specific user, or connected group of users) more efficiently. Also, comparing the result obtained here with the results obtained by text mining methods focusing on other (i.e. non geographic) features has the potential to extend those approaches to obtain spatial results too.
Chapter 3

Efficient classification of billions of points into complex geographic regions using hierarchical triangular mesh

Handling large-scale data can present unexpected challenges. In this chapter, I present the implementation of a custom geographic indexing solution which we implemented integrated to our database of tweets (see Section 2.2.1). The main problem to solve was to assign the individual tweets to the geographic or administrative region in which they were posted based on their geographic coordinates. As the regions themselves can be arbitrarily complex polygons described by possibly millions of short line segments making up their boundary, not only the direct computation of this seemed prohibitively expensive in terms of runtime, but also, the indexing solution natively available to our database platform did not offer adequate performance. To amend this, we came up with a solution tailored for this problem at hand, which combines several existing techniques in a simple yet effective manner, providing an efficient solution which is also well integrated to the rest of the database system. Here, I present some overview of challenges and solutions in handling large-scale datasets with relation to current database technologies, also focusing on the topic of geographic indexing, and then describe the main concepts used in our solution.

Material presented in this chapter appeared in Ref. [2].

3.1 Database systems and data-intensive science

Challenges related to handling data have always been a part of scientists’ work. Before computers became widely used, this included tedious manual calculations and results published in written or printed tables which would sometimes span multiple volumes of books. Notable examples include the calculation of the planets’ orbits with increasing accuracy or collections of constants describing materials or even tables containing values of transcendent mathematical functions, with a somewhat extreme example being a book of random numbers [57], referred to as a book designed to contain the minimal amount of information possible while not being blank. Apart from the question of how to work with data efficiently, the related question of how to share data with fellow scientists is also relevant. Traditionally, in many cases, relevant data could be summarized in a table that would fit in the main content or possibly in the supplementary material of a scientific article. Even in cases when this is still viable, sharing not only results, but all the “raw data” (e.g. results of a measurement or experiment) would be desirable, as this would help
in reproducing results and also open up the possibility to analyze data to researchers who could not afford to reproduce the measurements or experiments themselves. In this way, a dataset could have other uses beside the primary use which originally motivated researchers to obtain it.

As computers became widely used by scientists in the second half of the twentieth century, the storage, analysis and possible sharing of relevant data was largely simplified in many cases. Apart from applying the new technology to problems at hand, the availability of computers opened up the ability to use new and different approaches. In line with the trends discussed in Section 1.2, the volume of data on which researchers work has recently increased by several orders of magnitude. While computers in general have kept up with the pace of this increase, approaches to data handling constantly need to be adapted to be able to utilize the available resources efficiently. A possible approach adopted for scientific purposes is the use of relational database management systems (RDBMS) for the storage, sharing, and some of the manipulation of data. In the following, I give a short overview of possibilities and challenges before turning to the more specific topic of handling spatial data with the help of RDBMS products.

Relational database management systems were originally developed by information technology experts to meet demands of various businesses related to reliable data storage. Besides storage, features include analysis tasks based on relations which can be defined in the data stored. Apart from practical usage, theoretical basis and technologies necessary for the efficient functioning of RDBMS systems have been in the focus of research from computer science [54, 47, 79]. The main concepts are then as follows. Data is organized in tables, where a table is defined by a set of columns with each having a fixed data type, which are normally set at the time when the table is created. The motivation for this is that data is represented by a collection of records made up of a fixed number of individual values per record. Each record then corresponds to a row in the table and the set of values is mapped to the columns. E.g. a dataset describing a GPS trace could be stored in a table with three columns: the longitude and latitude as real numbers (stored as double-precision floating point) and the timestamp as either as a date type provided by the database system or as an integer representation. Each point in the trace would be then stored as a row in a table, with the values placed in the appropriate columns. While the definition of a table contains the number and types of columns and is in most cases considered fixed after creation, rows can be freely inserted or deleted during usage. In this way, a database table can be thought to be an equivalent of a datafile containing the records with the format specified by the table definition. Database systems are in many cases programmed via SQL (Structured Query Language), which is a declarative language specifying the task the database engine should carry out. The most basic task is retrieving data from one or more of the tables (called selecting rows), either for the purpose of presenting to the user, some processing program, or for inserting into an other database table. SQL provides the possibility to specify conditions for retrieving data (e.g. retrieving rows which match certain criteria), exploiting relations between data in two or more tables (called joins; the result can be the Descartes product of the set of rows in the tables, or only a subset which satisfies a given condition, e.g. related identifiers in the two tables should match), and performing aggregations, e.g. computing averages, or possibly more complex measures, like correlation coefficients of two datasets. SQL also provides means for creating and deleting tables, importing data from files and managing the database. While there is a standardized version of SQL with the basic functionality, RDBMS vendors typically ship their products vendor-specific extensions for the more complex tasks.
In accordance with industry requirements, RDBMS products also implement efficient methodology for multiple users and tasks to access to same set of data. Data can be separated into databases or schemas with permissions set on the user level (i.e. limiting ability to modify data) and consistency of data can be guaranteed even if multiple users work on the same table at the same time. Modern RDBMS products are also tailored on running on high-performance servers, making use of multiple cores, large amounts of memory, and storage systems in the range of petabytes. In many use-cases, data is also secured by the use of redundant storage systems.

Based on these properties, RDBMS systems can be effectively employed by scientists in use-cases where large amount of data needs to be stored and accessed by multiple users collaborating on the same subject. While commodity personal computers nowadays can effectively store already a few terabytes of data, in cases when the volume of data in a projects approaches this, it can be convenient to store it on a high-performance central server and not replicate it on workstations of every researcher. Delegating storage and analysis to the central server can thus enable researchers to use more lightweight workstations and can provide savings in costs of hardware, maintenance, and effort required to move data around the network. A prime example of exploiting these features offered by RDBMS systems is the Sloan Digital Sky Survey, whose results are stored in a publicly accessible database [188]. In this case, the traditional database system is complemented by a web front-end, allowing anyone to work on the data on the servers of the maintainers¹. On slightly smaller scales, a research group typically shares a high performance server or possibly cluster containing large datasets they work on and enabling running simulations and data processing tasks efficiently.

3.2 A database of billions of tweets

Over the course of the past few years, I have participated in a project which focused on publicly available data from the Twitter microblogging platform. Twitter provides a free sample stream of public posts which can be downloaded through regular HTTP requests pointed at the URLs associated with the service². The data is provided as JSON (Javascript Object Notation) objects which are far from ideal for any further processing and analysis. Extracting relevant data from them is in itself a non-trivial task. Based on our previous experience with databases, we decided to use Microsoft SQL Server 2012 as a relational database engine for storing data after processing the JSON files. A RDBMS product was chosen as the ability to optimally execute arbitrary analysis tasks expressed as SQL queries seemed highly valuable. Also, the built-in capability of efficient indexing can greatly help with filtering and aggregation tasks. E.g. all preprocessing for the analysis presented in Chapter 2 was performed within the database, with the data transferred to a Matlab instance running on a different server for carrying out the mathematically intensive SVD steps. In the following, I focus on the challenges relevant to geographic indexing, more specifically to the problem of classifying billions of points into complex geographical regions which was my main contribution to the database system; more details about the database can be found in Ref. [6].

¹available at http://skyserver.sdss.org
²description is available at https://dev.twitter.com/streaming/overview
### 3.3 Spatial indexing in database systems

Spatial indexing of geographic data has long been a focus of interest for many practical reasons. Since the wide-spread adoption of social media and social networks, the size of data to be indexed has grown by multiple orders of magnitude, making even more demand on the efficiency of indexing algorithms and index structures. Certain fields of natural sciences also face similar problems: astronomers, for example, have to perform spatial searches in databases of billions of observations where spatial criteria can be arbitrarily complex spherical regions. Inspired by astronomical problems, Szalay et al. [187, 41] came up with a solution to index spatial data stored in Microsoft SQL Server years before native support of geographic indexing appeared in the product. Their indexing scheme is called Hierarchical Triangular Mesh (HTM) and uses an iteratively refined triangulation of the surface of the sphere to build a quad-tree index. A solution which is somewhat similar in aims and main features is the HealPix indexing scheme, which is available as an open-source library usable from several programming languages and environments.

Following the increased interest in handling spatial data, RDBMS vendors started to include support for it over the course of the last decade. E.g. Microsoft SQL Server includes the ability to store geographic objects since 2008, PostgreSQL supports geographic objects since via the PostGIS library since 2005. In these solutions, a new datatype was introduced, which allows the representation of arbitrary complex shapes made up of points, lines and polygons. This means that such a shape can be stored in one row of a database table, in a column with the appropriate spatial datatype. Manipulation of these objects is then supported through additional functions defined to operate on them. The most basic functionality is to convert to and from standard text and binary representations [i.e. well known text (WKT) and well known binary (WKB), defined by the Open Geospatial Consortium, available at http://www.opengeospatial.org/standards/lfa](http://healpix.sourceforge.net/)

While the describing such functionality can be done easily, the implementation can be quite challenging. Runtime of algorithms performing the manipulation of spatial objects (e.g. computing the union or intersection of two regions) in many cases scale proportionally to the product of the complexities (i.e. number of segments making up the boundary) of the regions involved. In the case when these need to be performed for a large set of complex shapes, total runtimes can be quite significant. In accordance with this, indexing solutions to speed up processing need to be employed.

In the following, we focus on spatial joins, where from two sets of spatial objects, pairs which either intersect or contain each other need to be selected. A naive solution for this would be to perform a test for each possible pair of objects, meaning \( N \times M \) individual tests if the sets contain \( N \) and \( M \) objects respectively. Each of these tests is also a complex operation, depending on the complexity of the objects involved resulting in significant runtimes. Similarly to how database systems can speed up regular join operations with the use of indexes, avoiding the need for the condition to be evaluated \( N \times M \) times, an adequate spatial index could speed up spatial joins with eliminating the need for most of the costly operations.
Microsoft SQL Server supports spatial indexing since version 2008 via a library integrated to the product via the .Net framework runtime (SQL CLR). The library works by projecting the hemisphere onto a quadrilateral pyramid, then projecting the pyramid onto a square, and tessellating the square using four levels of fixed-size rectangular grids to construct a spatial index. The number of grid cells can be set between $4 \times 4$ and $16 \times 16$ providing a maximal index depth of 32 bits. The index structure itself is materialized as a hidden, but otherwise normal database table containing one row for each cell touched by the geography object being indexed. The table uses 11 bytes for the spatial index which is complemented by the primary key of the indexed object. The final index size can be controlled by limiting the number of cells stored for each geography object resulting in less effective pre-filtering of spatial matches while keeping the index size small. As of version 2012, the hard limit on the number of index entries per geography object is 8192, i.e. it is covered by a maximum of 8192 grid cells. This means that in the case of complex objects, higher levels of the index will not be used as that would result in too many cells.

Similarly to the built-in spatial index of SQL Server, HTM indexing is also implemented in .Net. Indexing is achieved by recursively subdividing the surface of the Earth into spherical triangles, called trixels. Each trixel is assigned a 40-bit HTM ID. Hierarchical indexing means that an indexing depth or level can be chosen according to the required precision; converting between different levels is a straightforward process which can be carried out by dividing a trixel into four smaller triangles when increasing depth by one level. When considering HTM IDs, this is achieved by multiplying or dividing the numerical value of the IDs by four at each level. A trixel on a lower level is then represented by a continuous range of IDs on a higher level, which correspond to the higher level trixels contained within. A canonical representation of trixels is level 20; in this case the Earth covered by a total of $8 \times 4^{20} \approx 8.8 \times 10^{12}$ trixels, with the typical linear size of pixels being 10 meters. Lower-level (i.e. bigger) trixels are then represented by the start and end of the range of covering level 20 trixels. As HTM is a custom library, we have full control over how geographical shapes are indexed; we chose to represent points by the ID of the level 20 trixel which contains it, and regions by a list of trixels which cover it on a given level, represented by ranges of corresponding level 20 trixels.

### 3.4 Indexing Twitter data and geographical regions

The Twitter dataset now includes over two billion geo-tagged tweets. In the study presented in this Chapter, an earlier dataset of only one billion tweets were used, and geographic indexing was applied to find tweets which fall within the USA and classify these into the state they belong to. Using the methods presented here, classification of all tweets to countries or states all over the world was carried out later; also, county-level classification for tweets from the USA was done.

For the classification, maps were obtained from gadm.org, an open database of global administrative regions of every country in the world. The maps were loaded into the database as geography objects using the built-in geography type and the native geographical index provided by SQL Server was generated. After carrying out tests, it became obvious that the performance of the native indexing was insufficient for efficiently performing a spatial join involving over one billion coordinates, and geographical regions of the complexity in this scenario (the total size of the maps is about one gigabyte, corresponding to geographic shapes made up of over 25 million individual line segments – the dataset which contains the US states alone is 31 MB large and is made up of almost two million
Figure 3.1: Illustration of the indexing scheme. Note that for the sake of illustration, only level 7 HTM depth and a non-representative sample of tweets was used. Top row: index generation, from left to right: initial covering of a convex hull of California with level 6 trixels using the cover generator present in the HTM Library for astronomical purposes and applicable to convex regions; classification of the level 6 trixels as full (green), partial (red) or distinct (transparent); further iteration with subdividing partial trixels and repeating the classification at a higher level. Middle row: classification of points in full trixels, from left to right: a sample of tweet coordinates overlaid on the level 7 index; identification of points either in full or partial trixels using only the HTM index; points in full trixels are successfully classified with using only the HTM index (i.e. without resorting to functions involving the spatial objects themselves), while points in partial trixels remain. Bottom row: classification of points in partial trixels, from left to right: a close-up of a sample of points, including some in partial trixels; close-up of one partial trixel and some points within; points in partial trixels can be classified by pre-computing the intersection of the original region with the partial trixel and testing if the points fall in this region; the intersection is displayed by red border in the bottom right panel.
line segments). We then decided to use HTM for geographic indexing, by creating a custom index stored in accompanying database tables, and using these for most part of the job of performing the spatial join. The main process is illustrated in Fig. 3.1; the basic idea is that we create a covering of a region at a specified maximum index level, and store the list of trixels accompanied with the information whether they are fully contained in the region or just intersect (we refer to these as full or partial trixels respectively). Coordinates which fall inside full trixels are then classified very efficiently by using only the HTM index; this is an operation which is carried out with a regular database join where the condition is that the HTM ID of the coordinate pair falls in a range corresponding to the full trixel. Here, no reference to the spatial objects is required and can thus it can be carried out very efficiently utilizing regular database indexing. Coordinates which fall into partial trixels however still need to be checked with functions operating on the spatial objects. Since this is expected to be computationally intensive, the number of such coordinates should be minimized by using an appropriate high level index. Of course, increasing the index level used will result in larger index sizes; thus a trade-off between index size and the number of points falling in partial trixels need to be established. We note that since RDBMS systems are generally optimized for storing and performing joins on quite large datasets, having a large spatial index will not generally be an issue, while spatial containment tests could prove challenging.

The HTM library was originally built for astronomical applications, where regions on the sphere are better represented as unions of convex shapes contoured by great or small circles than by vertices of polygons connected by great circles. This union-of-convexes representation is not appropriate for highly detailed complex maps as shapes would need to be decomposed into convexes first, a process that significantly increases the size of the data structures. This means that we had to implement our own solution for computing the HTM tessellation of the polygons in the map. We achieved this by combining the HTM library with the built-in geographic library of SQL Server. Our solution, see Algorithm 1, goes as follows. We construct a coarse tessellation of the region based on the bounding circle, then we intersect each trixel with the region using the built-in functions of SQL Server. If a trixel is completely inside the region it is added to the result set. Similarly, a completely disjoint trixel is discarded. Trixels intersecting with the boundary are refined into four sub-trixels and the algorithm is called recursively. Passing only the intersection of the original map with the trixel to the recursive call reduces the total runtime of the tessellation significantly. The algorithm uses the maximum depth of HTM trixels as a stop condition to limit the resolution of the tessellation but could be easily modified to use an upper limit on the number of trixels instead. Also, instead of trying to keep the index tables small, we store every trixel of the tessellation. Trixels on the deepest level which intersect with the boundary of the geography object are flagged as “partial”. The middle row in Fig. 3.1 illustrates the results of the level 7 HTM covering of California with partial trixels in red. We implemented Algorithm 1 as a loadable module for SQL Server utilizing the .Net runtime with code written in C#. We note that it would be straightforward to port this module to be used with other database systems, other GIS libraries, or even to be run as a standalone program. In the current version, the code is integrated with Microsoft SQL Server, and the index generation can be run as an SQL statement inserting results into a database table. Example usage is given in Query 1. While the input is a spatial object, the output consists only of regular data records, i.e. pairs of HTM IDs making up the ranges corresponding to the trixels covering the input region.
Algorithm 1 The function used for creating the HTM tessellation of a region. The function parameters are the region to be tessellated (region), the list of covering trixels to be refined (trixellist), and the maximum depth (maxlevel). The STContains and STIntersection functions are provided by the SQL Server geography library. Trixels added to the result set are flagged as either full or partial.

```java
function EvalTrixels(region, trixellist, maxlevel)
    retlist ← ∅
    for all t in trixellist do
        if region.STContains(t) then
            ▷ Full trixel
            t.Partial ← false
            retlist.Add(t)
            ▷ Flag as full
        else
            ▷ Partial or disjoint trixel
            region2 ← region.STIntersection(t)
            if region2 ≠ ∅ then
                ▷ Partial trixel
                if t.Level ≥ maxlevel then
                    ▷ Flag as partial
                    t.Partial ← true
                    retlist.Add(t)
                else
                    ▷ Continue recursion
                    tlist2 ← t.Refine(t.Level+1)
                    retlist.AddRange(EvalTrixels(region2, tlist2, maxlevel))
                end if
            end if
        end if
    end for
    return retlist
end function
```
3.5 Spatial joins

Query 1 Simplified schema of our database used for benchmarking HTM. The last statement creates the HTM index using our implementation of Algorithm 1.

CREATE TABLE tweet (  
    ID bigint PRIMARY KEY,  
    HTMID bigint,  
    coord geography );  
CREATE INDEX IX_tweet_htm  
    ON tweet ( HTMID );  
CREATE SPATIAL INDEX IX_tweet_spatial  
    ON tweet ( coord );  
CREATE TABLE region (  
    ID int PRIMARY KEY,  
    geo geography );  
CREATE SPATIAL INDEX IX_region_spatial  
    ON region ( geo );  
CREATE TABLE regionHTM (  
    ID int, --foreign key to region.ID  
    start bigint,  
    end bigint,  
    partial bit );  
CREATE INDEX IX_region  
    ON regionHTM(start,end);  
INSERT INTO regionHTM  
    SELECT ID, lo, hi, partial FROM region  
    CROSS APPLY dbo.HTMIndexCreate(_geom, 12, 0);

In order to explain the internals of HTM indexing, we create the schema of our database with Query 1. The HTM-based pre-filtering of a spatial join between a table containing GPS coordinates and another containing the tessellations of complex regions requires an inner join with a BETWEEN operator in the join constraint. Query 2 is a simplified example of such pre-filtering query. We will refer to these types of queries as range joins. As range joins are highly optimized in the database engine, we expect excellent pre-filtering performance. The LOOP JOIN hint is added to suggest a query plan that consist of a full scan of the regionHTM table, while doing appropriate index seeks on the much larger tweet table. This is optimal as long as the number of index rows is smaller than the number of points of interest (note that in our case, this difference is several orders of magnitude). Using the built-in spatial index of SQL Server, pre-filtering of a spatial join can be done with Query 3. It translates into a rather complex execution plan that uses the spatial index for table tweet only, while calculating the tessellation of the geography objects in table region during query execution, or vice versa. By specifying query hints, one can tell the server which spatial index to use, but it seems impossible to use the spatial indices on both tables at the same time. This behavior has a tremendous impact on the performance of spatial joins when using the built-in indices.

In case of the SQL Server geography index, exact containment testing can be done by simply replacing the function call to Filter with STContains, as in Query 4. When using the HTM index, points in full trixels are already accurately classified with Query 2, only points in partial trixels need further processing to filter out false positive matches. This is done in Query 5 which again relies on the spatial functions of SQL Server. Also note,
that Query 5, by referencing the column `region.geo`, uses the entire region for testing point containment. In case of computing the spatial join of billions of coordinates with a limited number of complex regions, it is well worth to pre-compute the intersections of partial trixels and regions first and use them for containment testing instead of the whole regions. This improved process is displayed in Query 6; all performance metrics presented in the following were measured this way.

### 3.6 Performance evaluation

We measured the index generation time for the 50 continental states (plus Washington D.C.) of the United States using two different depths (8 × 8 and 16 × 16 grids) of the SQL Server geography index and three different depths (level 12, 14 and 16) of HTM. For comparison, the 8 × 8 resolution of the SQL Server index roughly corresponds to a level 12 HTM index and the 16 × 16 grid resolution corresponds to a level 16 HTM index. For illustration, the covering of the coastline near downtown San Francisco is displayed in Fig. 3.2. The benchmarks were run on a 16-core database server with 96 GB memory. As our dataset fits into memory, queries are basically CPU-limited. Index generation times are summarized in Table 3.1. Note, that SQL Server executed the geography index generation on two threads, while HTM ranges were generated on a single thread only. While we have no control over the internals of geography indices, the iterative refining of HTM tessellation could be replaced with a smarter, multi-threaded one. Also, the size of the geography indices is internally limited to 8192 entries per region, while the HTM indices were calculated without pruning, ultimately resulting in much larger index sizes.

**Query 2 Pre-filtering of a spatial join with HTM index.**

```sql
SELECT t.ID, h.ID, h.partial
FROM tweet t
INNER LOOP JOIN regionHTM h
  ON t.htmID BETWEEN h.start AND h.end
```

It is also rather instructive to compare the two indexing schemes by the false positive rates of pre-filtering. The results are listed in Table 3.2. False positives rates of the HTM index are significantly lower in all cases, especially for higher index depths. In the...
Query 3 Pre-filtering of a spatial join with the SQL Server geography index.

```sql
SELECT t.ID, r.ID
FROM tweet t INNER JOIN region r ON r.geo.Filter(t.coord) = 1
```

Query 4 Exact containment testing with the SQL Server geography index.

```sql
SELECT tweet.ID, region.ID
FROM tweet t INNER JOIN region r ON r.geo.STContains(t.coord) = 1
```

Query 5 Classifying points from partial trixels.

```sql
SELECT tweet.ID, regionHTM.ID
FROM tweet t INNER LOOP JOIN regionHTM h ON t.HTMID BETWEEN h.start AND h.end
INNER JOIN region r ON r.ID = h.ID
WHERE h.partial = 0 OR r.geo.STContains(t coord) = 1
```

case of the SQL Server spatial index, increasing the resolution does not help, but the opposite: it just makes things worse, as the number of index rows (and the resolution of the tessellation) is limited to a maximum of 8192 cells, insufficiently small for complex maps. Strictly limiting index size only helps when the number of shapes to be indexed is large and the shapes are relatively small and simple. When spatial indices fit into the memory, or at least can be read quickly from the disk, pre-filtering using range joins is expected to be significantly faster, even for indices with millions of rows, rather than exact containment testing against complex shapes.

To test the performance of point classification, we prepared three samples having approximately 300 thousand, 1 million, 5 million points in each, uniformly sampled from the original database. Some tests were also run using the entire data set of more than one billion tweets. The coordinates covered the entire world but the majority of them were within the continental United States. The geographical distribution of the samples is realistic and follows the population density weighted by the local Twitter usage rate. To evaluate index performance, we computed a spatial join between the samples of GPS coordinates with different cardinality and the 51 regions, first with pre-filtering only, then with exact containment test. Results of pre-filtering are listed in Table 3.3, while exact containment metrics are shown in Table 3.4. All queries were executed using cold buffers, thus include I/O and CPU times.

The spatial join performance of the HTM index turned out to be about a hundred times better than the performance of the built-in geography index. Pre-filtering itself is about a thousand times faster than the built-in index, and usually could be done in a few seconds for the smaller samples. Such short query times are hard to be measured correctly, and values show a significant scatter when the queries are repeated. The two main reasons behind the significantly better performance of HTM are: 1) HTM-based pre-filtering could benefit from the spatial index on both tables, whereas SQL Server’s geography library only used the index for one of the tables and calculated the tessellation for the other table on the fly. 2) The extensive pruning of index entries resulted in a very high rate of false
Query 6 Classifying points using the HTM index. Optimized version: here, even for partial points, containment test is not performed for the complex original region; instead, intersections of level 14 trixels containing points to classify is precomputed with the regions, and only these are used to test for exact containments, making the calculations much faster.

```
-- 1. create a table to hold the result
CREATE TABLE result(tweetID bigint PRIMARY KEY, regionID int);
-- 2. insert points in full cells (using only the index)
INSERT INTO result SELECT tweet.ID, regionHTM.ID
FROM tweet t INNER LOOP JOIN regionHTM h ON
t-HTMID BETWEEN h.start AND h.end AND h.partial = 0;
-- 3. insert points in partial cells into a temporary table with
-- the corresponding candidate region IDs
SELECT tweet.ID AS tweetID, regionHTM.ID AS regionID,
tweet-HTMID, tweet.coord INTO ptstmp
FROM tweet t INNER LOOP JOIN regionHTM h ON
t-HTMID BETWEEN h.start AND h.end AND h.partial = 1;
-- 4. create intersections of level 14 partial cells with the regions and test
-- if the points are contained in these
WITH a AS (SELECT DISTINCT HTMID / 4096, regionID FROM ptstmp),
b AS (SELECT a.*, r.Geo.STIntersection(dbo.GeomFromTrixel(HTMID) AS GeoInt)
FROM a INNER JOIN region r ON a.regionID = r.ID)
INSERT INTO result SELECT tweetID, regionID FROM
ptstmp p INNER JOIN b ON p-HTMID / 4096 = b-HTMID AND
p.regionID = b.regionID AND b.GeoInt.STContains(p.coord) = 1;
```

<table>
<thead>
<tr>
<th>index type</th>
<th>time [s]</th>
<th>index rows</th>
</tr>
</thead>
<tbody>
<tr>
<td>geography 8 × 8</td>
<td>13,352</td>
<td>412,055</td>
</tr>
<tr>
<td>geography 16 × 16</td>
<td>6,215</td>
<td>410,040</td>
</tr>
<tr>
<td>HTM level 12</td>
<td>4,366</td>
<td>267,763</td>
</tr>
<tr>
<td>HTM level 14</td>
<td>5,151</td>
<td>1,331,632</td>
</tr>
<tr>
<td>HTM level 16</td>
<td>9,952</td>
<td>6,354,932</td>
</tr>
</tbody>
</table>

Table 3.1: Index generation time and number of index rows of the regions.

positives in case of SQL Server’s geography index. Because of the pruning, increasing the index resolution could not actually increase the resolution of the tessellation in case of the rather complex maps. By using a significantly larger, but still manageable index table, and by intersecting the trixels of the tessellations with the regions to reduce the complexity of exact containment testing, HTM indexing could reduce the cost of spatial joins to a minimum. Based on these results, it is clear that running the point classification using only the built-in geography index of SQL Server index is not a viable solution for any task similar to ours, namely, when the number of points is in the billions range.

3.7 Conclusions

In this chapter, I presented a study investigating the feasibility of efficient classification of GPS coordinates of Twitter messages by geographic regions using a relational database management system, Microsoft SQL Server 2012. We evaluated the performance of the built-in spatial indexing technology side by side with a customized solution based on
<table>
<thead>
<tr>
<th>index type</th>
<th>CO</th>
<th>IL</th>
<th>MD</th>
<th>WA</th>
</tr>
</thead>
<tbody>
<tr>
<td>geography 8 × 8</td>
<td>&lt;0.01%</td>
<td>0.16%</td>
<td>3.62%</td>
<td>1.11%</td>
</tr>
<tr>
<td>geography 16 × 16</td>
<td>&lt;0.01%</td>
<td>4.66%</td>
<td>22.43%</td>
<td>3.14%</td>
</tr>
<tr>
<td>HTM level 12</td>
<td>0.01%</td>
<td>1.71%</td>
<td>4.82%</td>
<td>1.30%</td>
</tr>
<tr>
<td>HTM level 14</td>
<td>&lt;0.01%</td>
<td>0.18%</td>
<td>1.84%</td>
<td>0.47%</td>
</tr>
<tr>
<td>HTM level 16</td>
<td>&lt;0.01%</td>
<td>0.04%</td>
<td>0.53%</td>
<td>0.23%</td>
</tr>
</tbody>
</table>

**Table 3.2:** False positive rate of spatial join pre-filtering for the U.S. states Colorado, Illinois, Maryland and Washington. Note, that false positive rates depend on the actual distribution of tweets and not only on the geometry of the states.

<table>
<thead>
<tr>
<th>index type</th>
<th>300k [s]</th>
<th>1M [s]</th>
<th>5M [s]</th>
<th>1G [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>geography 8 × 8</td>
<td>223</td>
<td>780</td>
<td>5009</td>
<td>-</td>
</tr>
<tr>
<td>geography 16 × 16</td>
<td>223</td>
<td>883</td>
<td>4053</td>
<td>-</td>
</tr>
<tr>
<td>HTM level 12</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>194</td>
</tr>
<tr>
<td>HTM level 14</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>266</td>
</tr>
<tr>
<td>HTM level 16</td>
<td>7</td>
<td>4</td>
<td>5</td>
<td>232</td>
</tr>
</tbody>
</table>

**Table 3.3:** Pre-filtering time for the spatial join query

<table>
<thead>
<tr>
<th>index type</th>
<th>300k [s]</th>
<th>1M [s]</th>
<th>5M [s]</th>
<th>1G [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>geography 8 × 8</td>
<td>295</td>
<td>915</td>
<td>4276</td>
<td>-</td>
</tr>
<tr>
<td>geography 16 × 16</td>
<td>301</td>
<td>773</td>
<td>4273</td>
<td>-</td>
</tr>
<tr>
<td>HTM level 12</td>
<td>12</td>
<td>24</td>
<td>139</td>
<td>2370</td>
</tr>
<tr>
<td>HTM level 14</td>
<td>7</td>
<td>13</td>
<td>58</td>
<td>1299</td>
</tr>
<tr>
<td>HTM level 16</td>
<td>8</td>
<td>10</td>
<td>42</td>
<td>1032</td>
</tr>
</tbody>
</table>

**Table 3.4:** Total time for the spatial join query
Hierarchical Triangular Mesh (HTM) indexing. The built-in spatial index was found to be inadequate to perform spatial joins between large sets of GPS coordinates (on the scale of billions) and complex geographic regions. We showed that our solution, a heuristic combination of existing techniques for handling spatial data in a relational database environment, can easily be a hundred times faster and makes the computation of the aforementioned spatial join available in reasonable time. We pointed out that the strength of HTM indexing is the great control the database programmer has on both the index structure and query execution (via hints). We also demonstrated that aggressive pruning of spatial indices is not a good idea when indexing of very complex regions is a requirement, as range-join-based pre-filtering is significantly faster than exact containment testing, even in case of millions of index entries. To make exact containment testing even faster, we pre-computed the intersections of the complex geographic regions and partial HTM trixels and use these much smaller shapes to filter out false positives.

Concluding, we note that providing a general purpose indexing solution in a spatially-enabled database can be quite challenging. Consequently, solutions specialized for the problem at hand can provide significant improvements in many cases. To benefit the scientific community, we make the source code of our implementation available at the project's website for possible future re-use\(^5\).

\(^5\)http://www.vo.elte.hu/htmpaper
Complex systems around us can in many cases be represented as networks evolving in time, i.e. temporal networks [97]. Depending on the goals of a study, different actual representations of a temporal network might be used: possible approaches include considering a network to have a continuously evolving structure, or possibly looking at snapshot networks in appropriate time windows. Important questions are then to quantify relevant statistics of the time evolution [124] or identify important changes in network structure and relate these to events in the history of the system [121, 78, 160]. Among a diverse range of complex phenomena where networks, and more specifically, the concept of temporal networks can be applied, economics is an important example of possible fruitful application, gaining additional insight to the more traditional approach [46, 43, 28, 42, 173, 156]. Although a large volume of financial data is available for research, information about the everyday transactions of individuals is usually considered sensitive and is kept private. In this chapter, I focus on Bitcoin, a novel currency system, where the complete list of transactions is publicly available. Using this data, I reconstruct the network of transactions and the time and amount of each payment. I analyze the structure of this transaction network employing several approaches from network science. I believe that this is the first opportunity to investigate the movement of money on everyday scales in such detail.

First, I present an overview of the Bitcoin system from a data science perspective, and discuss how temporal networks can be defined based on the list of transactions. Then, I present results regarding the evolution of network characteristics over time, such as clustering coefficient, degree distribution and degree correlations. Focusing on microscopic statistics of the transactions, I present evidence that linear preferential attachment drives the growth of the transaction network, while wealth accumulation can be described by sub-linear preferential attachment. Finally, I present a possible method for detecting relevant changes in network structure and demonstrate its merit by relating some of the uncovered variation in network structure to the variation of exchange price between bitcoins and conventional currencies.

Material presented in this chapter appeared in Refs. [4] and [5].
4.1 The Bitcoin network

4.1.1 Overview

Bitcoin is an innovative digital currency system, which is based on a peer-to-peer network of users relaying and validating transactions. The main motivation behind it was to create a payment system which is independent of banks and governments. With the financial crisis and the resulting distrust in traditional financial systems by a part of the population, the question if there could exist possible alternatives became popular of course. These questions are in some part economical and relate to the behavior of the very complex system of people interacting in possibly complex ways. On the other hand, part of these questions are mainly technical, i.e. if there could be an implementation of a payment scheme where participants need not trust a central authority with handling their assets, but can validate the proper working of the system and prevent fraud themselves. The Bitcoin system, which was first proposed in 2008 by Satoshi Nakamoto\(^1\) [147] provides an affirmative answer in the technical sense if the majority of its users use it honestly. This essentially means that a possible implementation for a system where any user can check the validity of any transaction was proposed, including the specification of the algorithms ensuring proper functioning and details like the network protocol for communication between the users, which was later implemented as an open-source program\(^2\). Whether it will work as a viable alternative payment system in the economic sense is of course dependent on many factors apart from the technical implementation, e.g. the behavior of and interaction among users and the rest of the economy. Apart from that, we need to note that while the technical implementation secures transactions by state-of-the-art cryptography, it is fundamentally not possible to limit interactions happening outside the system, including stealing or losing the cryptographic keys associated with someone’s holdings, any security issues with users’ computers or people making any agreements among themselves. As a consequence, already many users choose not to install and use a client program themselves, but rather entrust third-parties with handling their transactions, in a way which is very similar to monetary transactions handled by traditional banks\(^3\).

Based on Satoshi Nakamoto’s ideas, the system went online in January 2009 [147, 168, 165, 153]. For over a year, it was only used by a few enthusiasts, and bitcoins did not have any real-world value. A trading website called MtGox was started in 2010, making the exchange of bitcoins and conventional money significantly easier. More people and services joined the system, resulting in a steadily growing exchange rate. Starting from 2011, appearances in the mainstream media drew wider public attention, which led to skyrocketing prices accompanied by large fluctuations (see Fig. 4.3). Since the inception of Bitcoin over 59 million transactions took place (as of February, 2015), and currently the market value of all bitcoins in circulation exceeds 3 billion dollars. As of April 2015, the total computing power of users devoted to validating transactions is over 300 PH/s (300 \(\cdot 10^{15}\) unique hash values per second), which is estimated to correspond to approximately \(4 \cdot 10^{21}\) FLOPS, over 10,000 times the combined computing power of the world’s top 500 supercomputer clusters\(^4\).

\(^1\)Note that the name Satoshi Nakamoto is widely believed to be a pseudonym. See e.g. https://en.bitcoin.it/wiki/Satoshi_Nakamoto

\(^2\)This "official" implementation is available at https://github.com/bitcoin/bitcoin

\(^3\)See e.g. https://bitcoin.org/en/choose-your-wallet for more information; services listed here as “Web wallets” usually have control over the users’ bitcoins

\(^4\)Bitcoin mining computing power estimate from http://bitcoinwatch.com, supercomputer estimate
A very important aspect of the Bitcoin system and perhaps the most radical difference from traditional financial systems is that all transaction data is public to anyone who participates. To provide some privacy to users, no personal information needs to be attached to transactions, participants are only identified by their Bitcoin addresses of which anyone can create an unlimited number. This way, linking these addresses to real-world entities can generally not be carried out, but even with this limitation, the Bitcoin system provides a unique opportunity to study a payment network in detail. Transaction-level data is usually kept confidential in traditional financial systems, and researchers can only use special datasets made available for limited purposes. While results obtained using Bitcoin data might not be directly applicable to more traditional financial systems, I believe that the Bitcoin dataset is still a prime opportunity to evaluate models in a more detailed fashion, including some of the microscopic details. Also, research utilizing Bitcoin can show the possibilities of analyzing detailed transaction data, giving an estimate what kind of results could be gained if this kind of analysis was possible on data from traditional financial systems.

4.1.2 Relevant technical details

Bitcoin is a decentralized digital cash system, there is no single overseeing authority [147]. The system operates as an online peer-to-peer network, anyone can join by installing a client application and connecting it to the network. The unit of the currency is one bitcoin (abbreviated as BTC), and the smallest transferable amount is $10^{-8}$ BTC (although this is a technical detail which could be changed in the future). Instead of having a bank account maintained by a central authority, each user has a Bitcoin address, that consists of a pair of public and private keys. Existing bitcoins are associated to the public key of their owner, and outgoing payments have to be signed by the owner using his private key. Each transaction consists of one or more inputs and outputs. In Fig. 4.1 we show a schematic view of a typical Bitcoin transaction. To maintain privacy, a single user may use multiple addresses. Each participating node stores the complete list of previous transactions. Every new payment is announced on the network, and the payment is validated by checking consistency with the entire transaction history which is available to anyone participating in the network. To avoid fraud, it is necessary that the participants agree on a single valid transaction history. This is done by a computationally difficult process, which essentially corresponds to a reverse hash problem solvable only by a brute-force method (i.e. by evaluating a very large number of possible candidates). The result of this process is that transactions are formed into blocks at an approximately constant rate of one block per 10 minutes. Any block can contain a varying number of transactions. These blocks form the block-chain, where each block references the previous one. Changing a previous transaction (e.g. double spending) would require the recomputation of all blocks since then, which becomes practically infeasible after a few blocks. The system is more secure if more resources are devoted to the validation process, so that no single entity can control over 50% of all computing power which would allow them to monopolize on block generation. To provide incentive, new bitcoins are created periodically and distributed among the nodes participating in block generation. Thus, this process is referred to as mining bitcoins as an analogy to the role of gold and other rare metals in the traditional...
To maintain an approximately constant rate of block generation, the difficulty of the generation process can be set in a wide range to match the computing power available in the network. As interest in the Bitcoin system grew, the effort required to generate new blocks, and thus receive the newly available bitcoins, has increased over 40 billion fold; most miners today use specialized hardware, requiring significant investments. Consequently, an average Bitcoin user typically acquires bitcoins by either buying them at an exchange site from someone who already has bitcoins or receiving them as compensation for goods or services. Note that exchange price of bitcoins is completely determined by the market and has indeed had very high fluctuations in the past few years.

Due to the nature of the system, the record of all previous transactions since its beginning are publicly available to anyone participating in the Bitcoin network. From these records, one can recover the sending and receiving addresses, the sum involved and the approximate time of the transaction. Such detailed information is rarely available in financial systems, making the Bitcoin network a valuable source of empirical data involving monetary transactions. Of course, there are shortcomings: only the addresses involved in the transactions are revealed, not the users themselves. While providing complete anonymity is not among the stated goals of the Bitcoin project, identifying addresses belonging to the same user can be difficult, especially on a large scale. Each user can have an unlimited number of Bitcoin addresses, which appear as separate nodes in the transaction records. When constructing the network of users, these addresses would need to be joined to a single entity.

Another issue arises not only for Bitcoin, but for most online social datasets is that it is hard to determine which observed phenomena are specific to the system, and which results are general. We do not know to what extent the group of people using such a system can be considered as a representative sample of the society. In the case of Bitcoin for example, due to the perceived anonymity of the system, it is widely used for commerce of illegal items and substances; these types of transactions are probably overrepresented among Bitcoin transactions. Also, since the employed technological and philosophical ideas, we expect technology enthusiasts and people interested in the idea of an alternative financial system to be overrepresented. Ultimately, the validity of the results presented here could be tested if data becomes available from other sources, and comparison becomes possible.

### 4.1.3 Data

We downloaded the complete list of transactions, and reconstructed the transaction network: each node represents a Bitcoin address, and we draw a directed link between two nodes if there was at least one transaction between the corresponding addresses. In addition to the topology, we also obtained the time and amount of every payment. Therefore, we are able to analyze both the evolution of the network and the dynamical process taking place on it, i.e. the flow and accumulation of bitcoins.

We installed the open-source `bitcoin` client and downloaded the blockchain from the peer-to-peer network on Feb 16th, 2015. We modified the client to extract the list of all transactions in a human-readable format. The data and the source code of the modified client program is available at the project’s website or in a limited form through the Casjobs web database interface.

---

6See e.g. [https://en.bitcoin.it/wiki/Anonymity](https://en.bitcoin.it/wiki/Anonymity).

7[http://www.vo.elte.hu/bitcoin](http://www.vo.elte.hu/bitcoin)

8Available at [http://nm.vo.elte.hu/casjobs](http://nm.vo.elte.hu/casjobs)
The data includes 343,716 blocks, which contain a total of 59,792,412 transactions. This dataset includes 64,605,039 addresses; of these, 4,725,729 addresses were active in the last month (i.e. the month before Feb 16th, 2015, the time the data was downloaded). The Bitcoin network itself does not store balances associated with addresses, these can however be calculated from the sum of received and sent bitcoins for each address; preventing overspending in the network is done by requiring that the input of a transaction corresponds to the output of a previous transaction. Using this method, we found that almost four million addresses had nonzero balance at the time of our analysis.

4.1.4 Options for reconstructing the transaction network

A Bitcoin transaction can have multiple input and output addresses (see Fig. 4.1); when creating a network among addresses, we draw a directed edge pointing from each input address to each output address, thus a transaction with \( n \) inputs and \( m \) outputs will result in \( n \times m \) directed edges. We then also associate a timestamp with these: since transactions in the Bitcoin protocol do not have timestamps themselves, we use the timestamp of the block which contains the transaction. Block timestamps are well-defined, and are included in the block hash used to build the blockchain and are thus ensured to be agreed upon by all participants.

Temporal aggregation

Since the network edges obtained from the Bitcoin transaction data are events with timestamps there are several options for temporal aggregation so as to gain a network representation. These options also depend on the use-case, e.g. whether we would like to look at the state of the network at a given time or at an aggregated network representing the events which happened in a time window.
Possibly the simplest option is to consider at some time $t$ all edges which appeared previously. This way we have a growing network, similarly to e.g. the original model of Barabási and Albert [23] and later generalizations. For edges appearing multiple times, we can associate weights with them according to the number of occurrences. In this case the degree of a node (either in- or outdegree) will be either the number of distinct transaction partners it had or the number of transactions it participated in throughout the whole lifetime of the network. While it is a simple option for reconstructing a network, one might argue that transactions which appeared a long time ago will have a little importance with relation to the state of the network. We will refer to the network gained this way as the total aggregated network.

A possible way to better account for the time dynamics is to consider links to have a finite “lifetime”; this way the network at time $t$ will be the network made up for all edges appearing between $t - T$ and $t$ for some lifetime parameter $T > 0$. A question which need to be addressed here is then choosing an appropriate value for the $T$ parameter. Again, different values of $T$ will result in networks where different aspects are possibly emphasized. Repeating any analysis for multiple values of $T$, we can analyse if the results found are robust with regard to different time scales. In this case, the degrees of nodes will be the number of distinct transaction partners or number of transaction participations in a recent time window of $T$; this way only recent events will be accounted for. A notable difference from the previous method is that in this case, the degree of a node can decrease, while for the total aggregated network, degrees can only increase over time. In the following, we will refer to the results of this method as the partially aggregated networks.

We note that creating total or partially aggregated networks can be applied to result in a very large number of network instances; essentially, we can choose each transaction as the end of the time window, resulting in as many instances as the number of transactions. Of course, if we would like to create partially aggregated networks, having networks from only non-overlapping time windows can be beneficial to avoid overfitting any model.

**Aggregation by Bitcoin users**

As previously noted, anyone can create an unlimited number of Bitcoin addresses. The use of multiple addresses is also encouraged to increase the privacy of users. Some Bitcoin clients create new addresses automatically every time bitcoins need to be moved (e.g. as addresses to return “change” to). When performing an analysis regarding the use of bitcoins, ideally we would want to identify addresses belonging to the same real-world entity (user or company) as a first step. Unfortunately, this cannot be carried out generally, only in special cases, where either additional information provided by the users is present, or the addresses themselves can be linked together based on the transaction records [168, 165]. Nevertheless, there is a simple heuristic procedure which can be applied on large-scale (i.e. for the all the addresses) in an automated fashion which provides a baseline approximation of the real mapping of addresses to users or entities. For this, we make the basic assumption that all input address in a transaction are controlled by the same entity. Since when creating a transaction, it needs to be signed by the private keys of all input addresses, it seems as a valid baseline assumption [78, 184]. We note that two independent entities could create a transaction together by sharing a partially signed transaction by means other than the Bitcoin network; but due to the additional complexity and the limited incentive in doing so, we expect that this option is not used widely. This
way, we map addresses appearing as inputs of the same transaction together as belonging
to the same user or entity. We apply this procedure transitively, e.g. if addresses A and B
appear as inputs in transaction T1, while addresses B and C are the inputs of transaction
T2, we consider all three addresses to belong to the same user. This way, performing
this aggregation corresponds to finding the connected components of the supplementary
undirected graph where nodes are the addresses and a link is drawn between two nodes if
they appear as the inputs of the same transaction. Applying this procedure, we then gain
a contracted transaction network, where each node is actually a collection of the original
Bitcoin addresses; links are then drawn if there is a link in the original network between
any of the addresses of two nodes. Again, we can weight links with the number of times
they appear over the time window in which we aggregate the network. We will think of
nodes in this contracted network as entities; these can be either individual people using
Bitcoin, or some Bitcoin-related company. As the procedure is not complete, still a real-
world user can have multiple such nodes. Still, this is the most widely accepted method
of pre-processing the Bitcoin network [165, 168, 184]; there is also a further possibility
which tries to identify addresses created only for the purpose of receiving “change” [184],
which we did not use in the current analysis. We note that a possible further problem with
any such approach is that many users nowadays do not manage their bitcoins themselves
as that would require running a client program on their computer and downloading and
updating of the list of transactions, but trust this to third-party services in a way somewhat
similar to traditional banks. The bitcoins belonging to these users will then be associated
to addresses managed by the service provider and thus could not possibly be identified
separately anymore. Of course in this case the network topology still represents relations
regarding the control of bitcoins in the technical sense; the fact that someone manages
bitcoins for someone else is actually beyond the system.

We refer to the networks gained with applying this procedure as contracted networks,
and to nodes in it as Bitcoin-using entities. We perform the contraction using all infor-
mation available from the complete list of transactions, meaning that the correspondence
between Bitcoin addresses and entities does not change over time. On the other hand,
when considering the links present in the network, we again have the option to consider
a total aggregated network or partially aggregated networks as described in the previous
section.

In Fig. 4.2 we show the distribution of number of individual addresses of Bitcoin user
entities; this distribution is highly heterogeneous, well approximated by the power-law form
\[ P(n) \sim n^{1.922}. \]

4.2 Evolution of the transaction network

4.2.1 General characteristics

Bitcoin is an evolving network: new nodes are added by creating new Bitcoin addresses,
and links are created if there is a transaction between two previously unconnected ad-
dresses. The number of nodes steadily grows over time with some fluctuations; especially
noticeable is the large peak which coincides with the first boom in the exchange rate in
2011 (Fig. 4.3) [153]. After six years Bitcoin now has had a total of \( N = 64,605,039 \) nodes
and \( L = 410,500,161 \) links; among these, 3,865,446 addresses had nonzero balance on the
16th February 2015, while 4,725,729 addresses took part in at least one transaction in a
one month interval before that date. To study the evolution of the network, we measure the
Figure 4.2: Distribution of the number of individual addresses of entities in the contracted network. Here, the data is logarithmically binned for the sake of better visualization; the black line shows a power-law fit carried out on the original data using the methodology described in Ref. [53], the resulting exponent is 1.922.

Figure 4.3: The growth of the Bitcoin network. Number of addresses with nonzero balance (green), addresses in participating in at least one transaction in one week intervals (red) and the exchange price of bitcoins in US dollars (blue).
change of network characteristics in function of time. We identify two distinct phases of growth: (i) The initial phase lasted until the fall of 2010, in this period the system had low activity and was mostly used as an experiment. The network measures are characterized by large fluctuations. (ii) After the initial phase the Bitcoin started to function as a real currency, bitcoins gained real value. The network measures converged to their typical value by mid-2011 and they did not change significantly afterwards. We call this period the trading phase.

In the following, most results are displayed for the total aggregated network (see the Section 4.1.4 for the discussion on various options for reconstructing the transaction network and the definitions used when referring to them), unless especially noted otherwise; most results are robust with respect to aggregation in time and by possible Bitcoin-using entities. Results obtained when choosing a different aggregation when reconstructing the network are presented as comparison in Section 4.4.

We first measure the degree distribution of the network. We find that both the in- and the outdegree distributions are highly heterogeneous, and they can be modeled with power-laws [53]. Figures 4.4 and 4.5 show the distribution of indegrees and outdegrees at different points of time during the evolution of the Bitcoin network. In the initial phase the number of nodes is low, and thus fitting the data is prone to large error. In the trading phase, the exponents of the distributions do not change significantly, and they are approximated by power-laws $p_{\text{in}}(k_{\text{in}}) \sim k_{\text{in}}^{-2.72}$ and $p_{\text{out}}(k_{\text{out}}) \sim k_{\text{out}}^{-2.27}$. Degree distributions in partially aggregated and contracted networks also show similar behavior with somewhat smaller exponents; for comparison, these are presented in Section 4.4, Figs. 4.13, 4.14 and 4.17.

To further characterize the evolution of the degree distributions we calculate the corresponding Gini coefficients in function of time (Fig. 4.6). The Gini coefficient is mainly used in economics to characterize the inequality present in the distribution of wealth, but it can be used to measure the heterogeneity of any empirical distribution. In general, the
Figure 4.5: Evolution of the outdegree distribution. Left: without weights, i.e. the number of distinct transaction partners up to a given time. Right: weighted network, i.e. the outdegree is the number of times a node has appeared as a transaction input up to a given time. The black lines show a fitted power-laws for the final networks; the exponents are 2.27 and 2.19 for the unweighted and weighted cases respectively. The data is log-binned for ease of visual inspection, the power-law is fitted on the original data [53].

Gini coefficient is defined as

$$G = \frac{2 \sum_{i=1}^{n} ix_i}{n \sum_{i=1}^{n} x_i} - \frac{n + 1}{n}$$

(4.1)

where \(\{x_i\}\) is a sample of size \(n\), and \(x_i\) are monotonically ordered, i.e. \(x_i \leq x_{i+1}\). \(G = 0\) indicates perfect equality, i.e. every node has the same wealth; and \(G = 1\) corresponds to complete inequality, i.e. the complete wealth in the system is owned by a single individual. For example, in the case of pure power-law distribution with \(\alpha \geq 2\) exponent, the Gini coefficient is \(G = 1/(2\alpha - 3)\) [65]. This shows the fact that smaller \(\alpha\) exponents yield more heterogeneous distributions.

In the Bitcoin network we find that in the initial phase the Gini coefficient of the indegree distribution is close to 1 and for the outdegree distribution it is much lower. We speculate that in this phase a few users collected bitcoins, and without the possibility to trade, they stored them on a single address. In the second phase the coefficients quickly converge first to \(G^{\text{in}} \approx 0.629\) and \(G^{\text{out}} \approx 0.521\) and then display a relatively abrupt increase to about 0.697 and 0.643 in the beginning of 2014, indicating that normal trade is characterized by both highly heterogeneous in- and outdegree distributions.

To characterize the degree correlations we measure the Pearson correlation coefficient of the out- and indegrees of connected node pairs:

$$r = \frac{\sum_{e} (j_{\text{out}}^e - \bar{j}^{\text{out}})(k_{\text{in}}^e - \bar{k}^{\text{in}})}{L\sigma_{\text{out}}\sigma_{\text{in}}}.$$  

(4.2)

Here \(j_{\text{out}}^e\) is the outdegree of the node at the beginning of link \(e\), and \(k_{\text{in}}^e\) is the indegree of the node at the end of link \(e\). The summation \(\sum_e\) runs over all links, \(\bar{k}^{\text{in}} = \sum_e k_{\text{in}}^e / L\) and \(\sigma_{\text{in}}^2 = \sum_e (k_{\text{in}}^e - \bar{k}^{\text{in}})^2 / L\). We calculate \(\sigma_{\text{out}}\) and \(\bar{j}^{\text{out}}\) similarly.

We find that the correlation coefficient is negative, except for only a brief period in the initial phase. After mid-2010, the degree correlation coefficient stays between \(-0.01\) and \(-0.05\) for over three years, then slowly approaches zero in 2014, suggesting that the network is disassortative to some extent (Fig. 4.6). However, small values of \(r\) are hard to interpret: it was shown that for large purely scale-free networks \(r\) vanishes as the network...
size increases [138]. Therefore we compute the average nearest neighbor degree function 
\( k_{in}(k_{out}) \) for the final network; 
\( k_{in}(k_{out}) \) measures the average indegree of the neighbors of 
nodes with outdegree \( k_{out} \). We find clear disassortative behavior (Fig. 4.7).

We also measure the average clustering coefficient

\[
C = \frac{1}{N} \sum_{v} \frac{\Delta_v}{d_v(d_v-1)/2}, \tag{4.3}
\]

which measures the density of triangles in the network. Here the sum \( \sum_v \cdot \) runs over all 
nodes, and \( \Delta_v \) is the number of triangles containing node \( v \). To calculate \( \Delta_v \) we ignored 
the directionality of the links; \( d_v \) is the degree of node \( v \) in the undirected network. 
In the initial phase \( C \) is high, fluctuating around 0.15 (see Fig. 4.6), possibly a result 
of transactions taking place between addresses belonging to a few enthusiasts trying 
out the Bitcoin system by moving money between their own addresses. In the trading 
phase, the clustering coefficient stays in the range between 0.05 and 0.15, a value which 
is still significantly higher than the clustering coefficient for random networks with the 
same degree sequence. We speculate that the increase in mid-2012 is the effect of the 
SatoshiDice gambling site which was started around that time, and was alone responsible 
for a significant portion of Bitcoin transactions for some time with usage patterns that 
could increase the relative abundance of triangles in the network\(^9\).

4.2.2 Preferential attachment

To explain the observed broad degree distribution, we turn to the microscopic statistics 
of link formation. Most real complex networks exhibit distributions that can be approxi-
mated by power-laws. Preferential attachment was introduced as a possible mechanism

\(^9\)The site is still available at https://satoshidice.com; note that access was previously reported to 
have been banned from inside the United States of America due to gambling regulations in effect there. The 
site operates a gambling game where anyone wishing to participate only needs to send bitcoins to one of 
their addresses. For every such bet received, they would generate a random number to determine if it won, 
and send back a prize if it did. They advertise that the fairness of this game can be validated by anyone 
thanks to the Bitcoin transaction records being public. 

---

Figure 4.6: Gini coefficients, degree correlation and clustering. Left: Evolution of the 
Gini coefficients of the degree and the balance distributions. We observe the distinct 
initial phase lasting until mid-2011. The trading phase is characterized by approximately 
constant coefficients. Right: Evolution of the clustering coefficient and the out-in degree 
correlation coefficient. After the initial phase, again both measures reach a stationary 
value.
In networks without degree correlations, the degree of connected nodes do not depend on each other, therefore for such networks we expect that $k_{\text{in}}^{\text{nn}}(k_{\text{out}})$ is constant. In the case of the Bitcoin network, we observe a clear disassortative behavior: $k_{\text{in}}^{\text{nn}}(k_{\text{out}})$ is a decreasing function, indicating that nodes with high outdegree tend to connect to nodes with low indegree.

**Figure 4.7:** The average indegree of neighbors in the function of the outdegree $k_{\text{in}}^{\text{nn}}(k_{\text{out}})$. The cumulative distribution function of the $R$ values (see Eq. 4.5) for ten different $\alpha$ exponents. The straight line is an expected perfect fit if we exclude transaction outputs to newly created addresses. The inset shows the Kolmogorov-Smirnoff error for these exponents. This result was obtained for the total aggregated network with considering all transaction outputs as separate events to calculate $R$ values for. Other options yield similar results, see Section 4.4, Figs. 4.16,4.20 and 4.21.
to explain the prevalence of this property [22]. Indeed, direct measurements confirmed that preferential attachment governs the evolution of many real systems, e.g. scientific citation networks [148, 24, 161], collaboration networks [103], social networks [115, 142] or language use [162]. In its original form, preferential attachment describes the process when the probability of forming a new link is proportional to the degree of the target node [23]. In the past decade, several generalizations and modifications of the original model were proposed, aiming to reproduce further structural characteristics of real systems [114, 68, 67, 199]. Here, we investigate the nonlinear preferential attachment model [114], where the probability that a new link connects to node $v$ is

$$
\pi(k_v) = \frac{k_v^\alpha}{\sum_w k_w^\alpha}, \quad (4.4)
$$

where $k_v$ is the indegree of node $v$, and $\alpha > 0$. The probability that the new link connects to \emph{any} node with degree $k$ is $\Pi(k) \sim n_k(t)\pi(k)$, where $n_k(t)$ is the number of nodes with $k$ degree at the time of the link formation. We cannot test directly our assumption, because $\Pi(k)$ changes over time. To proceed we transform $\Pi(k)$ to a uniform distribution by calculating the rank function $R(k,t)$ for each new link given $\pi(k)$ and $n_k(t)$:

$$
R(k,t) = \frac{\sum_{j=0}^{k} n_j(t)j^\alpha}{\sum_{j=0}^{k_{\max}} n_j(t)j^\alpha} = \frac{\sum_{k_v<k} k_v^\alpha}{\sum_v k_v^\alpha}, \quad (4.5)
$$

If Eq. 4.4 holds, $R(k,t)$ is uniformly distributed in the interval $[0,1]$, independently of $t$. Therefore, if we plot the cumulative distribution function, we get a straight line for the correct exponent $\alpha$. To determine the best exponent, we compare the empirical distribution of the $R$ values to the uniform distribution for different exponents by computing the Kolmogorov-Smirnoff distance between the two distributions. We note that in practice, keeping track of the evolution of the degree distribution throughout the history of Bitcoin network and calculating the $R$ values for hundreds of millions of transaction outputs is somewhat challenging in a computational sense. These challenges and an efficient solution used for the calculations presented in this chapter are described in more detail in Appendix A.1.

Evaluating our method for the indegree distribution of the Bitcoin network, we find good correspondence between the empirical data and the presumed conditional probability function; the exponent giving the best fit is $\alpha \approx 1$ (Fig. 4.8; also see Section 4.4 for the discussion of the robustness of this result for different network aggregation options and in time throughout the history of the Bitcoin network). This shows that the overall growth statistics agree well with the preferential attachment process. Of course, preferential attachment itself cannot explain the disassortative degree correlations and the high clustering observed in the network. We argue that preferential attachment is a key factor shaping the degree distribution, however more detailed investigation of the growth process is necessary to explain the higher order correlations.

### 4.3 Dynamics of transactions

In this section, we analyze the detailed dynamics of money flow on the transaction network. The increasing availability of digital traces of human behavior revealed that various human activities, e.g. mobility patterns, phone calls or email communication, are
Figure 4.9: Distribution of time delays and wealth in Bitcoin. Left: distribution of time delay between transactions initiated from a single Bitcoin address. Delays calculated from timestamps with block-level granularity are displayed in red, while delays calculated using the blockchain.info site’s database of transaction timestamps for a subset of the data are displayed in green. We observe a power-law distribution close to the widely observed $P(T) \sim T^{-1}$, the exponential cutoff for large delays corresponds to the finite lifetime of the Bitcoin system. The difference for small delays can be explained by the granularity in the timestamps used: when using block timestamps, the observed delay between two transactions happening close to each other in time will be either zero or the delay between the two successive blocks. Blocks are generated at approximately ten minute (600 s) intervals, but in a stochastic fashion, resulting in that some of the small delays (< 1000 s effectively) will be replaced by zero in the data. Zero delays in the figure are displayed as 1 s so that they remain visible despite of the logarithmic scale of the x-axis, giving the peak at 1 s in the case of the red curve. In the case of more precise but possibly unreliable timestamp from the blockchain.info archives (green curve), we get back approximately a straight line for even these smaller values.

Right: Evolution of the distribution of balances of individual Bitcoin addresses. The black lines are stretched exponential and power-law fits of the last empirical distribution. The tail can be approximated by a power-law with exponent $-1.985$, however, the rest of the fit is unsatisfactory. Therefore, we fit the distribution with a stretched exponential distribution of form $P(b) \sim b^{-\gamma}e^{-(ab)^{1-\gamma}}$. We find a better approximation of the whole distributions; the parameters are $\gamma = 0.852$ and $a = 1641 \text{BTC}^{-1}$. The distinct peak around 50BTC corresponds to the fact that the reward for miners was 50BTC until the end the 28th November 2012 (when it was halved according to the goal of limiting the supply of bitcoins); the peak then represents addresses which hold just one unspent mining reward.
often characterized by heterogeneity [196, 104, 183, 118]. Here we show that the handling of money is not an exception: we find heterogeneity in both balance distribution and temporal patterns. We also investigate the microscopic statistics of transactions.

The state of node $v$ at time $t$ is given by the balance of the corresponding address $b_v(t)$, i.e. the number of bitcoins associated to node $v$. The transactions are directly available, and we can infer the balance of each node based on the transaction list. Note that the overall quantity of bitcoins increases over time: Bitcoin rewards users devoting computational power to sustain the system.

We first investigate the temporal patterns of the system by measuring the distribution of inactivity times $T$. The inactivity time is defined as the time elapsed between two consecutive outgoing transactions from a node. We find a broad distribution that can be approximated by the power-law $P(T) \sim 1/T$ (Fig. 4.9), in agreement with the behavior widely observed in various complex systems [27, 196, 132, 97].

It is well known that the wealth distribution of society is heterogeneous; the often cited --and quantitatively not precise-- 80-20 rule of Pareto states that the top 20% of the population controls 80% of the total wealth. In line with this, we find that the wealth distribution in the Bitcoin system is also highly heterogeneous. The proper Pareto-like statement for the Bitcoin system would be that the 4.1% of the addresses possesses the 95.9% of the total wealth. We measure the distribution of balances at different points of time, and we find a stable distribution in the trading phase. The tail of wealth distribution is generally modeled with a power-law [203, 152, 107], following this practice we find a power-law tail $\sim x^{-1.985}$ for balances $\geq 50$BTC (see Fig. 4.9). However, visual inspection of the fit is not convincing: the scaling regime spans only the last few orders of magnitude, and fails to reproduce the majority of the distribution. Instead we find that the overall behavior is much better approximated by the stretched exponential distribution $P(b) \sim b^{-\gamma}e^{-(ab)^{1-\gamma}}$, where $\gamma = 0.852$ and $a = 1641$BTC$^{-1}$. We note that this fit was carried out on the binned data and as such is more to be thought of as an interesting implication needing further investigation and not as a quantitative result.

To further investigate the evolution of the wealth distribution we measure the Gini coefficient over time. We find that the distribution is characterized by high values throughout the whole lifetime of the network, reaching a stationary value around $G \approx 0.985$ in the trading phase (see Fig. 4.6).

4.3.1 Preferential attachment in wealth accumulation

To understand the origin of this heterogeneity, we turn to the microscopic statistics of acquiring bitcoins. Similarly to the case of degree distributions, the observed heterogeneous wealth distributions are often explained by preferential attachment. Moreover, preferential attachment was proposed significantly earlier in the context of wealth distributions than complex networks [175]. In economics preferential attachment is traditionally called the "rich get richer phenomenon" or the Matthew effect after a biblical verse [191] (note that the context of the original verse is actually more about spiritual values than economics; despite this, the name "Matthew effect" is widely used to refer to the economic phenomenon, too). It states that the growth of the wealth of each individual is proportional to the wealth of that individual. In line with this principle, several statistical models were proposed to account for the heterogeneous wealth distribution [203, 100, 80, 194].

To find evidence supporting this hypothesis, we first investigate the change of balances in fixed time windows. We calculate the difference between the balance of each address at
Figure 4.10: Change of balances in one month windows. Increase (top) and decrease (bottom, vertical axis is inverted) of node balances in one month windows as a function of their balance at the beginning of each month. We show the raw data (red), the average (green), median (blue) and logarithmic average (magenta). The later three are calculated for logarithmically sized bins. We find a clear positive correlation: addresses with high balance typically increase their wealth more than addresses with low balance. The median and the logarithmic average values almost coincide, which suggests multiplicative fluctuations. The median and the logarithmic average increase approximately as power-laws for several orders of magnitude. The black line in the upper panel is a power-law fit for the double logarithmic data; the exponent is 0.752.
the end and at the start of each month. We plot the differences in function of the starting balances (Fig. 4.10). When the balance increases, we observe a positive correlation: the average growth increases in function of the starting balance, and it is approximated by the power-law $\sim b^{0.752}$. This indicates the “rich get richer” phenomenon is indeed present in the system. For decreasing balances, we find that a significant number of addresses lose all their wealth in the time frame of one month. This phenomenon is specific to Bitcoin: due to the privacy concerns of users, it is generally considered a good practice to move unspent bitcoins to a new address when carrying out a transaction.

To better quantify the preferential attachment, we carry out a similar analysis to the previous section. However, there is a technical difference: in the case of the evolution of the transaction network, for each event the degree of a node increases by exactly one. In the case of the wealth distribution there is no such constraint. To overcome this difficulty we consider the increment of a node’s balance by one unit as an event, e.g. if after a transaction $b_v$ increased by $\Delta b_v$, we consider it as $\Delta b_v$ separate and simultaneous events. We only consider events when the balance associated to an address increases, i.e. the address receives a payment. We now calculate the rank function $R(b,t)$ defined in Eq. 4.5, and plot the cumulative distribution function of the $R$ values observed throughout the whole time evolution of the Bitcoin network (Fig. 4.11). We find that the “average” behavior is best approximated by exponents around $\alpha \approx 0.85$, suggesting that $\pi(b_v)$ is a sublinear function. In the context of network evolution, previous theoretical work found that sublinear preferential attachment leads to a stationary stretched exponential distribution [114], in line with our observations.

We have investigated the evolution of both the transaction network and the wealth distribution separately. However, it is clear that the two processes are not independent. To study the connection between the two, we measure the correlation between the indegree and balance associated to the individual nodes. We plot the average balance of addresses

---

10“Most Bitcoin software and websites will help with this by generating a brand new address each time you perform a transaction.” Quote from https://en.bitcoin.it/wiki/Address
We calculate the averages for logarithmically sized bins. We find strong correlation between the balance and the indegree of individual nodes. The main plot shows indegree values up to $k_{\text{in}} \approx 16000$, only 72 nodes (0.00012%) have higher indegree, the averages calculated for such small sample result in high fluctuations (see inset). We also measure both the Pearson and Spearman correlation coefficient: The Pearson correlation coefficient of the full dataset is 0.00212847, while the Spearman rank correlation coefficient is 0.18234. (Note that the Pearson correlation coefficient measures the linear dependence between two variables, while the Spearman coefficient evaluates monotonicity). We test the statistical significance of the correlation by randomizing the dataset 10000 times and calculating the Spearman coefficient for each randomization. We find that the average Spearman coefficient is $4 \cdot 10^{-5}$ with a standard deviation of $10^{-3}$, indicating that the correlation is indeed significant.

as a function of their degrees on Fig. 4.12. For degrees in the range of 1–10000 (over 99.99% of all nodes with nonzero balance), the average balance is a monotonously increasing function of the degree, and it is approximated by the power-law $b \sim k_{\text{in}}^{0.56}$, indicating that the accumulated wealth and the number of distinct transaction partners an individual has are inherently related. A similar scaling law with a slightly higher exponent (0.68) was reported by Tseng et al., who conducted an online experiment where volunteers traded on a virtual market [194].

### 4.4 Robustness with regard to the network reconstruction method

#### 4.4.1 Partial temporal aggregation

So far, all presented results considered totally aggregated networks, where edges have unlimited lifetimes, i.e. we view the network as a growing network, in a way similar to the original preferential attachment model of Barabási and Albert [23] and later extensions [114, 68, 67, 199]. However, in the case of Bitcoin, there could be multiple definitions of edges as it was discussed in Section 4.1.4. One can indeed argue that links are not permanent here,
Figure 4.13: Indegree distribution of partially aggregated networks. Left: without weights, i.e. the number of distinct transaction partners in a given time window. Right: weighted network, i.e. the indegree is the number of times a node has appeared as transaction outputs in the time window. Top row: edges have a life-time of 30 days; bottom row: edges have a life-time of one day. The black lines show a fitted power-laws for the networks on the 16th February, 2015, i.e. the last one displayed here; the exponents are 2.45 and 2.31 for the unweighted and weighted cases of the network with 30 day edge life-times respectively. The same exponents are 2.56 and 2.29 for the networks with one day edge life-times.
Figure 4.14: Outdegree distribution of partially aggregated networks. Left: without weights, i.e. the number of distinct transaction partners in a given time window. Right: weighted network, i.e. the outdegree is the number of times a node has appeared as transaction inputs in the time window. Top row: edges have a life-time of 30 days; bottom row: edges have a life-time of one day. The black lines are fitted power-laws for the last of the considered networks, i.e. the one on 16th February, 2015; the exponents are 2.36 and 2.31 for the unweighted and weighted cases of the network with 30 day edge life-times respectively. The same exponents are 2.35 and 2.41 for the networks with one day edge life-times.
rather represent instantaneous events. In this sense, representing Bitcoin transactions as a network, where a link can represent a transaction which happened a long time ago might not seem appropriate. On the other hand, we argue that for any Bitcoin-using entity, it is a relevant property that what transaction partners it had previously, which is exactly the information which is represented in the evolving network view of Bitcoin transactions.

Nevertheless, the time scale we consider here could be relevant in many ways: transactions which happened a very long time ago might be considered less important in some sense than more recent ones. One of the possibilities to account for this is using partial temporal aggregation, i.e. considering a network reconstructed from only those transactions which happened in a limited time window before the time of analysis as discussed in Section 4.1.4. In this section, we consider one day and 30 day long time windows and show that the results presented previously for the network measures and preferential attachment remain valid on these scales too.

In Figs. 4.13 and 4.14 we display the in- and outdegree distributions of such partially aggregated networks at some points in time. We find that these are quite similar to the degree distributions of the total aggregated network, still highly heterogeneous; we note that the exponents of the fitted power-law distributions are slightly smaller, but are still over 2.

In Fig. 4.15, results for the Gini-coefficients of the degree distributions, degree correlations and clustering for partially aggregated networks with 30 day time windows are displayed. We find that these show remarkably similar behavior to the ones obtained for the total aggregated network, with significantly more “noise”, especially at the beginning where these partially aggregated networks are very small. There are also some notable peaks, e.g. in the spring of 2012 for the degree correlation and clustering coefficients, which we again speculate to be the effect of the SatoshiDice gambling site which was launched in the spring of 2012 and became extremely popular in only a few months.

In Fig. 4.16 we display results for the evaluation of preferential attachment for various cases. Apart from considering temporal aggregation of one day, 30 days and the total lifetime of the system, we also test if considering multiple occurrences of the same edge as multiple events (i.e. weighted edges in a sense) results in a difference. Our results show similar behavior in all cases, although the quality of the fit worsens when we take shorter aggregation intervals, resulting in a somewhat curved rank function statistics. This
Figure 4.16: Rank function for different temporal aggregation options for the original Bitcoin network. Left column: all transaction outputs are considered; right column: only the first occurrence of each edge is considered. Top row: total aggregated network. Middle row: partially aggregated network, edges have a lifetime of 30 days. Bottom row: partially aggregated network, edges have a lifetime of 1 day. Notice that the main features are the same while there are some differences. The insets show the Kolmogorov-Smirnoff errors for the exponents.
suggests that long-term activity of a node might be a better measure of its attractiveness as a possible transaction partner; also, while a simple form for the preferential attachment kernel as in Eq. 4.4 can explain the “average” behavior, a more elaborate model is needed to account for the details of the transaction dynamics.

4.4.2 Contracted network

A main concern about the Bitcoin dataset is its applicability to financial systems. This concern is actually made up of two parts. The first question is whether the behavior of Bitcoin–using entities represents the behavior of participants in real-world financial systems, e.g. the everyday monetary transactions of individuals. The second question is a more technical one and is about whether the statistics obtained from and the analysis performed on Bitcoin addresses is relevant to the behavior of Bitcoin–using entities, each of whom can have an unlimited number of individual addresses. Study of the first question is beyond the scope of the current work; we note that we expect that Bitcoin users will be a quite uneven sample of the general population, but this does not necessarily mean...
that the properties studied here would be completely different for a different population. However, this could only be tested if similar datasets from other sources became available for researchers. While giving a definite answer to the second question is not possible either as we do not know which Bitcoin addresses belong to the same entity, we can make a baseline test based on the heuristic contraction method presented in Section 4.1.4. In this section, we thus present results obtained using the contracted network gained by applying that procedure to original the Bitcoin transaction graph.

In Fig. 4.17 we show in- and outdegree distributions for the total aggregated network case. Again, we see that these can be well approximated by power-law functions $p_{in}(k_{in}) \sim k_{in}^{-2.28}$ and $p_{out}(k_{out}) \sim k_{out}^{-2.07}$. We obtained similar results also for weighted networks and for partially aggregated networks, with slightly smaller exponents, around 2. Time evolution of the related Gini-coefficients is shown in Fig. 4.18; the main features resemble that of the original network, with separate initial and trading phases, while the values of the coefficients are slightly smaller. Examining degree correlations and clustering (Fig. 4.19), again we gain a similar picture with behavior mostly similar to the original network, suggesting that the local structure of the two does not differ significantly.

Looking at the microscopic statistics of network evolution, we again find that preferential attachment is present in link formation statistics, providing an overall explanation for the broad degree distributions. Statistics for the $R$ function values (see Eq. 4.5) are displayed in Fig. 4.20, while Kolmogorov–Smirnov errors for different exponents are shown in Fig. 4.21. Evaluating the empirical distributions is less straightforward in this case; while in that case, the only macroscopic difference from an ideal distribution was that a high number of edges connected the nodes with zero indegree, here the beginning of the distributions shows a much larger “gap” in terms of $R$ values. We speculate that this is due to the fact that the distributions are defined on discrete values, resulting in jumps between $k = 0$, $k = 1$ and possibly $k = 2$. To account for this, we not only compare the empirical distributions to ideal uniform distributions between $R = 0$ and $R = 1$, but instead we compute a best linear fit for the data, which can than be interpreted as a uniform distribution in a limited range of $R$ values. This way, we get much better agreement, with significant differences between the empirical data and the fitted uniform distribution only at the beginning where we expect the effect of having discrete values and linking to nodes with zero indegree to be significant. Looking at the results (Fig. 4.20 and 4.21 for the error values), we now see a trend with respect to the time window of aggregation: for smaller time windows, we have larger exponents, suggesting that recent involvement of a
Figure 4.20: Rank function for different temporal aggregation options for the contracted Bitcoin network. Left column: all transaction outputs are considered; right column: only the first occurrence of each edge is considered. Top row: total aggregated network. Middle row: partially aggregated network, edges have a life-time of 30 days. Bottom row: partially aggregated network, edges have a life-time of 1 day. The black lines show a uniform distribution for $R > 0$ (the one which crosses the $x$-axis at the same value as the empirical data) and a best fit to the data, as the uniform distribution differs significantly from the observed behavior for at the beginning. Errors are displayed separately in Fig. 4.21.
Figure 4.21: Kolmogorov-Smirnoff errors for the empirical rank function in the case of the contracted Bitcoin network. Top row: all transaction outputs are considered; bottom row: only the first occurrence of each edge is considered. Left column: total aggregated network. Middle column: partially aggregated network, edges have a life-time of 30 days. Right column: partially aggregated network, edges have a life-time of 1 day. The red squares show the difference from a uniform distribution. The green squares show the difference from a “best fit” for the particular case, which would correspond to a uniform distribution in a more constrained range of $R$ values. See the text for more discussion.

node contributes more to its attractiveness than its complete history.

Looking at the dynamics of transactions and wealth accumulation, the results still resemble those of the original network. In Fig. 4.22 we display the distribution of delays between successive activity of the same node, which again shows a distribution resembling $P(T) \sim 1/T$. Also in Fig. 4.22, the distribution of balances of nodes shows is very similar to the distribution of balances of individual addresses (Fig. 4.9). Repeating the same analysis for the growth of balances as in the case of the original transaction network, we again get very similar results (see Figs. 4.24 and 4.23); furthermore, a connection between degree and wealth can be established here too (Fig. 4.25; the exponent here is 0.449, slightly smaller than previously).
Figure 4.22: Distribution of delays and balances in the contracted network. These are very similar to the distributions arising if we consider each address separately. The exponent of the fitted power-law for the distribution of (large) balances is $1.99$, while the parameters for the stretched exponential fit $P(b) \sim b^{-\gamma} e^{-ab^{1-\gamma}}$ are $\gamma = 0.857$ and $a = 1641 \text{ BTC}^{-1}$.

Figure 4.23: Rank function for the growth of balances. The cumulative distribution function of the $R$ values (see Eq. 4.5) for exponents 0, 0.2, 0.4, 0.5, 0.6, 0.7, 0.85 and 1. The inset shows the maximum Kolmogorov-Smirnov error for these exponents. Here, the results are not as obvious as in the case of link creation (Fig. 4.8; a simple power-law form like in Eq. 4.4 is not sufficient to accurately model the statistics of money flow. On the other hand, the “average” behavior shows a correlation between the balance and the increase of the balance: the uncorrelated assumption ($\alpha = 0$) clearly gives a much worse approximate than the exponents that presume preferential attachment ($\alpha > 0$).
Figure 4.24: Change of balances in one month windows. Increase (top) and decrease (bottom, vertical axis is inverted) of node balances in one month windows as a function of their balance at the beginning of each month. We show the raw data (red), the average (green), median (blue) and logarithmic average (magenta). The later three are calculated for logarithmically sized bins. We find a clear positive correlation: addresses with high balance typically increase their wealth more than addresses with low balance. The median and the logarithmic average values almost coincide, which suggests multiplicative fluctuations. The median and the logarithmic average increase approximately as power-laws for several orders of magnitude. The black line is a power-law fit for the double logarithmic data; the exponent is 0.747.
Figure 4.25: Average node balances as a function of the indegrees for the contracted network. Similarly to the case of individual addresses, we find a strong correlation. The power-law fit for the average balances in the $k_{\text{in}} \lesssim 16000$ gives an exponent of 0.449. The Pearson correlation coefficient of the full dataset is 0.044, while the Spearman rank correlation coefficient is 0.162. The average Spearman correlation coefficient for 10000 randomized realizations of the same dataset is $6 \cdot 10^{-5}$ while the standard deviation is $10^{-5}$, indicating that the correlation is indeed significant.
4.5 Detecting structural changes

After analysing Bitcoin as an evolving network in the previous sections, here I present how Principal Component Analysis can be applied to a network of monetary transactions to identify relevant changes in network structure over time and to uncover the relation of network structure and macroeconomic indicators of the system [160, 121, 173, 78]. Using the contracted network (see Section 4.1.4), and limiting the analysis only to the most active users, we construct daily snapshots of the transaction network. We then represent these series of networks by a graph time series matrix, where each row corresponds to a daily snapshot. We carry out PCA on this dataset identifying key features of network evolution, and we link some of these variations to the exchange rate of bitcoins.

4.5.1 Extracting the core network from the Bitcoin transactions

To study structural changes in the network and their connection with outside events, we extract the subgraph of the most active Bitcoin users. We start from the contracted network (see Section 4.1.4), and limit our analysis to transactions which happened in 2012 or 2013 (note: as the analysis presented here was performed in 2014, the contracted network used here corresponds to the one obtained with taking transactions only up to the end of 2013 into account). We thus have a graph where each node $v$ represents a Bitcoin-using entity, and each link $(u \rightarrow v)$ represents that there was at least one transaction between entities $u$ and $v$ during the observed two-year period.

We identify the active core of the transaction graph using two different approaches: (i) We include entities who appear in at least 100 individual transactions and were active for at least 600 days, i.e. at least 600 days passed between their first and last appearance in the dataset. We call these long-term users, and we refer to the extracted network as the LT core. (ii) We simply include the 2,000 most active entities. All users are considered, hence the resulting network is referred to as AU core. In both cases, we take the largest connected component of the graph induced by the selected nodes. Furthermore, we exclude nodes which are associated with the SatoshiDice gambling site; in 2012, this site and its users produced over half of all Bitcoin activity, which is not related to the normal functioning of the system.

In the case of long-term users, the LT core consists of $n_{LT} = 1,639$ nodes and $l_{LT} = 4,333$ edges; these users control a total of 1,894,906 Bitcoin addresses which participated in 4,837,957 individual transactions during the examined two-year period. In the case of most active users, the AU core has $n_{AU} = 1,288$ nodes and $l_{AU} = 7,255$ edges; the users in this subgraph have a total of 3,326,526 individual Bitcoin addresses and participated in a total of 12,900,964 transactions during the two years. The total number of Bitcoin transactions in this period is 27,930,580, meaning that the two subgraphs include 17.3% and 46.2% of all transactions respectively.

4.5.2 The graph time series matrix, and its principal components

To extract important changes in the graph structure, we compare successive snapshots of the active core of the transaction network using PCA. The goal is to obtain a set of base networks, and represent each day’s snapshot as a linear combination of these base networks.

---

11http://satoshidice.com; their addresses used for the service start with ‘1Dice’
We calculate the daily snapshots of the active core, each snapshot is a weighted network, and the weight of link \((u \rightarrow v)\) is equal to the number of transactions that occurred that day between \(u\) and \(v\). A snapshot for day \(t\) can be represented by an \(n \times n\) weighted adjacency matrix \(W_t\), where \(n \equiv n_{LT/AU}\) is the number of nodes in the aggregate network. Since there are \(l \equiv l_{LT/AU}\) links overall, each \(W_t\) has at maximum \(l\) nonzero elements. We rearrange \(W_t\) into an \(l \times l\) long vector \(w_t\). Note that we include all possible links, even if for that specific day, some are missing. For \(T\) snapshots, we now construct the \(T \times l\) graph time series matrix \(X\) such that the \(t\)th row of \(X\) equals \(w_t\) [121]. This way, we can consider \(X\) as a data matrix with \(T\) samples and \(l\) features, i.e. we consider each day as a sample, and the activities of possible edges as features.

To account for the for the high variation of the daily number of transactions, we normalize \(X\) such that the sum of each row equals 1. After that, as usual in PCA, we subtract the column averages from each column. As a result, both the row and column sums in the matrix will be zero. We compute the singular value decomposition (SVD) of the matrix \(X\):

\[
X = U \Sigma V^T, \tag{4.6}
\]

where \(\Sigma\) is a \(T \times l\) diagonal matrix containing the singular values (which are all positive), and the columns of the \(T \times T\) matrix \(U\) and the \(l \times l\) matrix \(V\) contain the singular vectors. Since in our case \(T < l\), there will be only \(T\) nonzero singular values and only \(T\) relevant singular vectors; as usual, we keep only the relevant parts of the matrices, this way \(\Sigma\) will be only \(T \times T\) and \(V\) will be only \(l \times l\) [163, 159]. The singular vectors are orthonormal, i.e. \(U^T U = V V^T = I\), where \(I\) is the \(T \times T\) identity matrix. It is customary to order the singular values and vectors such that the singular values are in decreasing order, so that the successive terms in the sum implied by the matrix multiplication in (4.6) give decreasing contribution to the result, thus giving a successive approximation of the original matrix. Note that these matrices can also be computed as the eigenvectors of the covariance matrices: \(XX^T\) and \(X^TX\) for \(U\) and \(V\) respectively, and as such, and the columns of \(U\) and \(V\) span the row and column space of \(X\). In accordance with this, we can interpret the singular vectors based on the interpretation of \(X\). The columns of \(V\) can be considered as ‘base networks’, the matrix element \(v_{ji}\) provides the weight of edge \(j\) in base network \(i\). According to (4.6), edge weights in the daily snapshots can be calculated as a linear combination of edge weights in the base networks. The singular values give the overall importance of a base network in approximating the whole time series, while the columns of \(U\) account for the temporal variation: the matrix element \(u_{ti} \equiv u_i(t)\) provides the contribution of base network \(i\) at time \(t\), e.g. the (normalized) weight of edge \(j\) on day \(t\) is given by:

\[
x_{tj} = \sum_{i=1}^{T} \Sigma_{ji} u_j(t) v_{ji}. \tag{4.7}
\]

### 4.5.3 Main observed features

Examining the singular values, we find that for both type of core extraction they decay only relatively slowly, i.e. a large number of components are required to obtain a good approximation of the original matrix \(X\) (see Fig. 4.26). This indicates that the system possibly contains non-Gaussian noise and high dimensional structure. Also, the distribution of edge weights \(p_i(v)\) in the base networks is fat-tailed; for the first six base networks we find very similar distributions, all well approximated with \(p_i(v) \sim v^{-1.8}\) for the LT core and \(p_i(v) \sim v^{-1.9}\) for the AU core.
Figure 4.26: Singular values of $X$. The inset shows the relative contribution up to the given index. The distribution of singular values is fat-tailed, a high number of components is required to explain the variations in the data. The best power-law fit is $\Sigma_{ii} \sim i^{-1.37}$.

Figure 4.27: Time-varying contributions of the first 6 base networks. The $u_i(t)$ values for the LT core (left) and AU core (right).

Examining the edges with large weights for the LT core, we find that most of these are repeated within the first few base networks. For example, if we consider the top-20 ranking edges (by the absolute value of weights) in the first 10 base networks, we find only 46 distinct edges instead of the 200 maximally possible. Among these, 44 induce a weakly connected graph, including a total of 29 users; considering all edges among these users, 20 of them form a strongly connected component by themselves and all are weakly connected. These 29 users have a total of 1,349,815 separate Bitcoin addresses, forming a highly active subset in our core network.

We show the time-varying contribution $u_i(t)$ of the first six base networks on Fig. 4.27. In most cases, $u_i(t)$ features a few abrupt changes, partitioning the history of Bitcoin into separate time periods. This is especially true for the AU core, where highly active but short lived users can significantly contribute. Identifying the individual nodes and edges responsible for network activity in a given period would require more information about Bitcoin addresses, which is difficult to obtain on a large scale.

The most striking feature uncovered by our analysis is a clear correspondence between the first singular vector of the graph of long-term users and the market price of bitcoins as shown on Fig. 4.28. Apart from visual similarity, the two datasets have a significantly high correlation coefficients (see Table 4.1).
Figure 4.28: The time-varying contribution of the first base network $u_1(t)$ for the LT core and the time series of the logarithm of exchange price. To illustrate how this corresponds to changes in the network, snapshots of the subnetworks induced by top 100 edges with the largest absolute weight in the base network are also shown for a few dates. Visual inspection shows a good correspondence of the two; also, the two datasets have high correlation coefficients, see Table 4.1.

Motivated by this result, we tested whether the price of bitcoins can be estimated with the $u_i(t)$ coefficients. To proceed, we subtract the average value from the price time series, and estimate this as a linear combination of singular vectors:

$$\tilde{B}(t) = \langle B(t) \rangle + \sum_{i=1}^{N} c_i u_i(t).$$

Here the coefficients $c_i$ can be computed as the dot product of the price time series and $u_i(t)$, and therefore $c_i$ is proportional to the Pearson correlation coefficients shown in Table 4.1. We are interested how large $N$ is needed to model the price time series with acceptable accuracy. We show the residual standard error as a function of $N$ on Fig. 4.29. It is apparent that there are sudden decreases in the error after the inclusion of certain base networks. We note that the base networks are ranked by their contribution to the original network time series matrix, i.e. the singular values of $X$. On the other hand, we can rank these components by their similarity to the price time series by using correlation coefficients. This results in a more rapid decrease of the error, and the large jumps are all at the beginning. In Table 4.1, we show the first few components with the highest correlation coefficients. Two approximations of the time series for the LT core is shown on Fig. 4.30; one with the first 4 base networks, the other with the 4 base networks whose time-varying contribution $u_i(t)$ has the highest correlation to the price. We find that in both cases most features of the time series are well approximated, but the fitted time series
Table 4.1: Correlation coefficients between the singular vectors of the network time series matrix and the Bitcoin exchange price. Left: first 5 singular vectors. Right: the 5 singular vectors with the highest correlation. Here, $\rho_p$ is the Pearson correlation coefficient and $\rho_s$ is the Spearman rank correlation coefficient.

<table>
<thead>
<tr>
<th>Component</th>
<th>long-term nodes</th>
<th>most active nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho_p$</td>
<td>$\rho_s$</td>
<td>$\rho_p$</td>
</tr>
<tr>
<td>1</td>
<td>0.8528</td>
<td>0.8654</td>
</tr>
<tr>
<td>2</td>
<td>-0.0335</td>
<td>0.4367</td>
</tr>
<tr>
<td>3</td>
<td>-0.3782</td>
<td>0.4407</td>
</tr>
<tr>
<td>4</td>
<td>-0.074</td>
<td>0.5913</td>
</tr>
<tr>
<td>5</td>
<td>-0.0148</td>
<td>-0.2991</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>long-term nodes</th>
<th>most active nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho_p$</td>
<td>$\rho_s$</td>
<td>$\rho_p$</td>
</tr>
<tr>
<td>10</td>
<td>-0.5395</td>
<td>0.0075</td>
</tr>
<tr>
<td>7</td>
<td>0.3928</td>
<td>0.3633</td>
</tr>
<tr>
<td>1</td>
<td>0.3689</td>
<td>0.9231</td>
</tr>
<tr>
<td>11</td>
<td>0.3276</td>
<td>-0.0578</td>
</tr>
<tr>
<td>8</td>
<td>-0.2632</td>
<td>-0.3631</td>
</tr>
</tbody>
</table>

Figure 4.29: Residual standard error of approximating the exchange rate of bitcoins with a linear combination of the time–varying contribution of base networks. is still apparently noisy. In the first case, the shape of the peak at the end of 2013 is missed, while in the second case it is approximated much better. A closer examination of the correlation values (Table 4.1) and the singular vectors reveals that the 21st component is responsible for this change, which contains high resemblance to the final section of the time series.

4.6 Discussion and future work

In this chapter, I presented analysis focusing on Bitcoin, a novel digital currency system. A key difference from traditional currencies handled by banks is the open nature of the Bitcoin: each transaction is publicly announced, providing unprecedented opportunity to study monetary transactions of individuals. We have downloaded and compiled the complete list of transactions, and we have extracted the time and amount of each payment. I presented studies exploiting some of the possibilities of this new kind of data; in these,
the structure and evolution of the transaction network was studied, and the dynamics taking place on the network, i.e. the flow of bitcoins was investigated.

Measuring basic network characteristics in function of time, we have identified two distinct phases in the lifetime of the system: (i) When the system was new, no businesses accepted bitcoins as a form of payment, therefore Bitcoin was more of an experiment than a real currency. This initial phase is characterized by large fluctuations in network characteristics, heterogeneous indegree- and homogeneous outdegree distribution. (ii) Later Bitcoin received wider public attention, the increasing number of users attracted services, and the system started to function as a real currency. This trading phase is characterized by stable network measures, disassortative degree correlations and power-law in- and outdegree distributions. We have measured the microscopic link formation statistics, finding that linear preferential attachment drives the growth of the network.

To study the accumulation of bitcoins we have measured the wealth distribution at different points in time. We have found that this distribution is highly heterogeneous throughout the lifetime of the system, and it converges to a stable stretched exponential distribution in the trading phase. We have found that sublinear preferential attachment drives the accumulation of wealth. Investigating the correlation between the wealth distribution and network topology, we have identified a scaling relation between the degree and wealth associated to individual nodes, implying that the ability to attract new connections and to gain wealth is fundamentally related.

Apart from focusing internal dynamics of the Bitcoin network to external measures, we also investigated whether connection between the network structure and macroscopic properties (i.e. the exchange price) can be established in the Bitcoin network. For our analysis, we reconstructed daily network snapshots of the networks of the most active users in a two-year period. We organized these snapshots into the graph time series matrix of the system. We analyzed this matrix using PCA which allowed us to identify changes in the network structure. A striking feature we found was that the time-varying contribution of some of the base networks show a clear correspondence with the market price of bitcoins. The correlation with the first base network was found to be exceptionally significant. Using the linear combination of only 4 vectors, we were able to reproduce most of the features of the long-term time evolution of the market price.

Based on our results, it is apparent that the analysis of the structure of the underlying network of a financial system can provide important new insights complementing analysis.
of the external features. Research could focus on qualitative and quantitative tests of existing econophysics models, which was traditionally only possible on real-world datasets in a limited fashion and for data from experiments with artificial markets. In the case of Bitcoin, the main question is of course if this dataset can be taken to be representative for economical behavior of individuals. Unfortunately, this could only be determined if datasets of similar nature would become available from traditional financial systems, i.e. data concerning bank transactions. Nevertheless, analysis and conclusions drawn from the Bitcoin network could provide valuable input to modeling efforts on more traditional financial systems too; demonstrating the usability of models and analysis methodology could benefit researchers with access to more traditional datasets. Work on Bitcoin can also outline what kind of data is needed for fruitful analysis of transaction networks, making possible to evaluate any potential privacy concerns with respect to expected benefits. Drawing on the results presented in this chapter, further research could focus on establishing causal relationship between the observed features; in the case of the observed preferential attachment, a main challenge would be to e.g. establish an agent-based model which reproduces the observed statistics without explicitly relying on knowledge of degree distributions, which we expect to not be known or considered directly by individuals when making decisions about transaction partners. With relation to market effects, establishing whether price fluctuations affect network structure or vice versa and evaluating the possibility of better predicting price changes based on structural changes in the network is an interesting outstanding research question. Also, collecting publicly available information about Bitcoin addresses identified as members of the highly active core of the network could result in a better understanding of the role of the associated users in the Bitcoin ecosystem, and help explain the correlations observed here.

Concluding, I believe that the data presented in this chapter has great potential to be used for evaluating and refining econophysics models, as not only the bulk properties, but also the microscopic statistics can be readily tested. To this end, we make all the data used here available online to the scientific community in easily accessible formats\textsuperscript{12}.

\textsuperscript{12}See http://www.vo.elte.hu/bitcoin
Chapter 5

Measuring the dimension of partially embedded networks

The dimensionality of a physical system is an essential parameter reflecting its spatial scaling properties. As an analogy to the fractal dimension of point sets, several definitions and their interpretation has been proposed as a measure of the fractal dimension of a complex network. A specific focus is spatially embedded networks, i.e. a network where the nodes have coordinates associated to them which are interpreted as their position in a metric space. In this case the relation of the network fractal dimension, the dimension of the embedding space and the fractal dimension of the set of nodes can pose an interesting question. Also, a related problem is finding a suitable embedding of an abstract network in a possibly low-dimension metric space. In this chapter, I review the fractal dimension concept and its applicability to networks, and then present a new concept which is a generalization of the spectral dimension measure for partially embedded networks, a class of networks where the embedding information is constrained to travel times or delays on the edges but coordinates for the nodes are not provided. I then present the application of these dimensionality concepts to various real-world networks, including the Twitter follower and mention networks introduced previously.

Material presented in this chapter appeared in Ref. [12].

5.1 Introduction

The dimensionality of a physical system is an essential parameter reflecting its spatial scaling properties. The dimension influences the behavior near a critical point, affecting the scaling of various static and dynamic physical quantities [75, 96]. Magnetic systems can be considered as a classic example, but in the last decades the concepts and methodology of critical behavior have been successfully applied to macroscopic and real-world inspired systems too [69, 60, 16, 33, 197, 15, 45]. Recently, the connection between criticality and dimensionality has also been extensively studied in the context of complex networks [66].

For general abstract networks however, it is not straightforward to obtain a proper definition of dimensionality. During the last several years, there appeared a number of methods that were successful in identifying scaling laws in complex networks, giving suitable generalizations of existing concepts of dimensionality [12, 99, 34, 181, 180, 106]. Recently, these methods were also adapted to include global spatial information present in spatially embedded networks [61].

In this chapter the feasibility of a dimensionality measurement for complex networks
that relies merely on *local* information is investigated. To do so, we give a generalization of the spectral dimension [12] to record what a random walker “perceives” while traversing the network. While a random walk is not a realistic model for every possible process taking place on a network, it is suitable to gain some information about the structure of the network. Both the random walk process and the concept of spectral dimension have been successfully applied to networks previously [34, 176]. Now, we refine the concept of spectral dimension to include the information present in distances or delays associated with the links, gaining a more complete measure of network dimensionality. Our new approach is readily applicable to spatially embedded networks, and additionally it allows the treatment of partially embedded networks, where the embedding information is only present as link-wise properties (e.g. distances or delays). These networks can be considered a special kind of weighted networks, where the link weights are related to the time needed to traverse the link. Note that we use the weights as generalized distances, which is fundamentally different from the treatment of networks with arbitrary weighted links [99]. The motivation for considering such networks is twofold.

First, for many real-world networks the spatial embedding is only partially feasible in practice. For instance, let us suppose a traversal or transport process taking place on a complex network lacking spatial information. As a result, we can generally obtain some sort of local, link-wise information (e.g. “delivery times” along the links), but without gaining any global knowledge on the physical layout of the network. In such situations, we can assign the measured link-wise property to each link to obtain a *partial embedding* of the network. A straightforward illustration is the Internet, where delays are relatively easy to measure, but reliably determining the geographic position of the nodes is not feasible on a large scale [117]. For similar partially embedded networks, a random walk processes can effectively utilize local knowledge to characterize the large-scale structure of the system.

Another motivation is for “fully” embedded networks. Even if there exists a natural (2 or 3D) embedding space for the network in question, it may still be relevant to study the network’s scaling behavior via local, link-wise properties. As an example, imagine a typical road network, where different types of roads have different speed limits. In such a network, travel times are not in a simple relationship with the metric distance of the embedding space (the length of road segments between two intersections). It is meaningful to investigate scaling in the light of the “overlay” property (travel time), instead of the metric distance.

### 5.2 Related Work

In the last several years, a number of methods have appeared in the literature that were successful in identifying scale-invariant properties in small-world complex networks. Probably, the earliest such concept is the *spectral dimension* [12] which originates from random walks on the network (see Section 5.2.1), and was applied to both theoretical models of networks [99] and empirical datasets [34]. Additionally, the application of the box-counting dimension [59] to networks was also proposed [180], and further generalized to reveal fractal properties of complex networks [181, 106]. The scaling exponents arising from these methods are usually interpreted as a special type of network dimension.

Beyond these methods, which handle networks as abstract graphs, there have been an increasing interest in including the spatial properties of networks, too (see e.g. [30, 206, 118, 125, 50, 137, 83]). Particularly, in the context of dimension measurements, the presence of spatial information enables the application of well-known approaches [87, 59]
to determine the fractal dimension of the point set of network nodes [206]. A shortcoming of these approaches may be, that while they take into account the geometric layout of the network, they entirely neglect its connectivity information. Recently, in Ref. [61] Daqing et al. have proposed more suitable methods to overcome this limitation. The authors combine both metric and topological knowledge to yield more comprehensive measures of dimensionality.

### 5.2.1 Random walks and dimensionality of graphs

A principal method to define the dimensionality of an abstract graph $G$ with $n$ nodes is performed by examining the properties of a random walk process. We consider undirected graphs, which we represent by the adjacency matrix $A$, where $A_{ij} = A_{ji} = 1$ if there is an edge between nodes $i$ and $j$, and otherwise 0. Let $D$ denote the degree matrix, the diagonal matrix with entries $D_{ii} = \sum_j A_{ij}$.

The spectral dimension can be estimated from the scaling behavior of the probability that a random walker returns to the origin of the walk [12, 91]. The random walk is defined as a Markov process. Let the $n$-dimensional vector $X(t)$ denote the probability distribution of a random walker at discrete time $t$, meaning that at time $t$ the walker is at node $i$ with probability $x_i(t)$ (here $\sum x_i = 1$). Then the same probability distribution after one step is

$$X(t+1) = T X(t), \quad (5.1)$$

where $T \equiv A D^{-1}$ is the transition matrix of the process; defining $T$ in this way means that in each step, the random walker follows each of the possible edges with equal probability. Let $s = (s_0, s_1, \ldots, s_m)$ denote a specific walk of length $|s| = m$ from node $s_0$ to node $s_m$. For each walk $s$ on the graph we can determine its probability:

$$P(s) = \prod_{i=0}^{m-1} T_{s_{i+1} s_i}. \quad (5.2)$$

This is indeed a probability distribution, since for a given starting node $v_0$ and length $m$ it satisfies $\sum_{|s|=m, s_0=v_0} P(s) = 1$.

If we wish to neglect the details of the walk and only focus on its two endpoints, we can define the transition probability $p_{j\leftarrow i}(t)$ as the probability of a walker starting at site $v_i$ to be at site $v_j$ at time $t$. It follows from Eq (5.1) that

$$p_{j\leftarrow i}(t) = \sum_{|s|=t} \sum_{s_0=i, s_t=j} P(s) = (T^t)_{ji}. \quad (5.3)$$

We will be interested in the $P_0(t) = \langle p_{i\leftarrow i}(t) \rangle_i$ return probability of the walk, that is the average probability that the random walk returns to the origin after $t$ steps (the average goes over all possible $i$ starting nodes). The return probability can be expressed with the transition matrix in the following way:

$$P_0(t) = \langle p_{i\leftarrow i}(t) \rangle_i = \sum_{|s|=t} \sum_{s_0=i, s_t=i} P(s) = \frac{1}{n} \text{Tr} T^t \quad (5.4)$$

If the return probability exhibits a

$$P_0 \sim t^{-\alpha}$$

81
scaling for a sufficiently large range, we define the spectral dimension of \( G \) as \( d_s = 2\alpha \). If \( G \) is a \( d \)-dimensional regular lattice, then \( d_s = d \) [12, 91].

Ideally, when evaluating Eq. 5.4, we would look at the \( t \to \infty \) case. In a finite network however, \( P_0 \) will approach its stationary value exponentially for large \( t \), as expected on the properties of the power iteration of a matrix [130]. In this case, we then need to look at smaller \( t \) values, before the exponential convergence to the stationary value dominates, and examine if there is a scaling there for a sufficiently large range. On the other hand, analytical results for some models in the infinite network limit can be obtained by direct calculations using Eq. 5.3. If we take a continuous limit in both time and space, then \( P_0 \) arises as the Laplace-transform of the eigenvalue density of the transition operator of the process [99]; if this is a power-law function \( \rho(\varepsilon) \sim \varepsilon^{d/2-1} \) then \( P_0(t) \sim t^{-d/2} \).

A possible generalization of \( d_s \) to spatially embedded networks is given in [61]. A graph \( G \) is said to be embedded into a metric space \((X,\rho)\) if each node of it corresponds to a point in \( X \), and \( \rho \) is a metric on \( X \), i.e. for each pair of nodes \( i \) and \( j \) we have a distance \( \rho(i,j) \). In a general setting \( \rho \) is the two or three dimensional Euclidean distance, but for many large-scale real world networks it is the spherical distance that plays the role of \( \rho \).

In case of an embedded graph, the random walk process can be interpreted as a diffusion in the embedding space \( X \). Consequently, we can measure the exponent of the diffusion on the embedded graph via the scaling relation

\[
 r \sim t^{\beta},
\]

where \( r(t) \) is the root mean square (r.m.s.) displacement of the random walker at time \( t \):

\[
 r^2(t) = \frac{1}{n} \sum_{ij} \rho^2(i,j)p_{ji}(t).
\]

The diffusion exponent is \( \beta = 0.5 \) for regular lattices in any dimension, while for real world systems it often exhibits anomalous behavior with \( \beta \neq 0.5 \) [91].

The spectral dimension concept employed by Daqing et al. [61] can be extracted from the scaling relation

\[
 P_0(t) \sim r^{-\gamma},
\]

where \( r = r(t) \). Here, the exponent gives an alternative measure for the dimension of the network: \( d_\rho = \gamma \). In the case where the three scaling laws (Eqs. 5.4, 5.5 and 5.7) are all valid in the same range, the three exponents are related:

\[
 \gamma = \alpha/\beta .
\]

For regular \( d \) dimensional lattices this relationship is satisfied as \( d_\rho = d_s = d \) and \( \beta = 1/2 \). Nevertheless, for more complex networks, the scaling regimes may not coincide, or some of the scaling relationships might not hold at all.

### 5.2.2 The role of the link length distribution

A remarkable result of Ref. [61] is that the authors demonstrate, that the \( P(\rho) \) distribution of link lengths has a central role in controlling the dimensionality of a spatial network. Inspired by that result, we also pay special attention to the distribution of link lengths of the networks considered. We emphasize, that the distribution of link lengths can be defined in two different ways. Throughout this chapter, by \( P(\rho) \) we refer to the observed
distribution, i.e. the probability that a link in the network has length \( \rho \). Instead of \( P(\rho) \), many authors also use the \( P_c(\rho) \) conditional version: the probability that two nodes are connected given that they are separated by distance \( \rho \). The two definitions are not independent: 
\[
P_c(\rho) = \frac{P(\rho)}{C(\rho)},
\]
where \( C(\rho) \) is the distribution of distances among any two points regardless of whether there is a link between them or not. For a set of uniformly distributed points in \( d \) dimensions, we have \( C(\rho) \sim \rho^{d-1} \) and thus \( P_c(\rho) \sim \rho^{1-d}P(\rho) \). On the other hand, the nodes in a real-world spatially embedded network are in many cases inhomogeneously distributed, which complicates the relation between \( P(\rho) \) and \( P_c(\rho) \). To this end, we only use the observed distribution \( P(\rho) \) as it can be estimated directly from the data.

Based on empirical observations, \( P(\rho) \) is often assumed to follow a power law decay: 
\[
P(\rho) \sim \rho^{-a} \quad [118, 125, 137, 83].
\]
The special case of \( a = 1 \) has been of interest from both theoretical and empirical perspective. In his seminal papers [109, 108], Jon Kleinberg showed that for an important family of small-world networks the \( a \) exponent controls the navigability of the network: at \( a = 1 \) efficient routing is achievable based merely on local information, while if \( a \neq 1 \), this is not feasible. (Note, that Kleinberg used the \( P_c(\rho) \) conditional version of the link length distribution and thus stated his results for \( P_c(\rho) \sim \rho^{-2} \) in two dimensions and nodes distributed over a grid, giving \( C(\rho) \sim \rho \).) Kleinberg’s results are in agreement with the empirical finding that in many real-world networks, \( a \approx 1 \). Early evidence for the Internet was found by Yook et al. [206], which has also been supplemented since then by more precise measurements [137]. The same phenomenon was also observed in online social networks [125, 8, 50], mobile communication networks [118], and e-mail networks [83]. The implications for navigability are in agreement with the famous original results of Travers and Milgram [141, 193] and later repetitions [64], and also with results gained via computer simulation on large-scale geographically embedded social network datasets [125, 8, 94]. Recently, in Ref. [98], Hu et al. proposed a statistical model which reproduces this peculiar scaling phenomenon. The authors found that the \( P(\rho) \sim 1/\rho \) behavior arises naturally from an entropy-maximization constraint. The real-world networks used in our analysis can also be well characterized by \( a \approx 1 \) (see Fig. 5.5).

5.3 Partially embedded networks

In the following, we propose a new method to measure the dimensionality of spatially embedded networks, which can also be generalized to the special type of networks which we refer to as partially embedded.

This class of networks lies in-between abstract networks (with topological information only) and spatial networks (with both topological and embedding information). These networks emerge either when the \( \rho(i,j) \) distances are not available generally, but only for connected node pairs (i.e. when \( (i,j) \in E \)), or when we wish to study the network as per a link-wise overlay property possibly different from \( \rho \) (e.g. link-wise delays). This relaxation yields networks that are spatially constrained to some extent, but the embedding information is incomplete, as the nodes do not have coordinates. Due to their transient character, we refer to these networks as partially embedded networks\(^1\). To distinguish spatially embedded networks from partially embedded networks, we refer to the link length function in a partially embedded network as \( \tau(i,j) \), which needs to be defined only for the

\(^1\)Note that the term “partially embedded” is also used in the mathematics literature for graphs that have a subgraph embedded in the plane.
Figure 5.1: Different methods to measure scaling via the random walk process. (a) shows a specific walk \( s = (a, b, c, d) \) on an abstract network in \( t = 3 \). Here, the \( d_s \) spectral dimension can be calculated via the Monte Carlo simulation of Eq. 5.4. (b) If the network is embedded into a metric space, we can obtain \( \rho(a,d) \) (and generally \( \rho(\cdot,\cdot) \) for any node pair), hence it is feasible to determine \( d_\rho \) via Eq. 5.7. (c) In case the network is only partially embedded, \( \rho(a,d) \) is not available generally, and thus it is not possible to obtain \( d_\rho \). However, if a \( \tau \) length (or delay) function is present for the links, we are able to cumulate the \( \tau \) values along the walk to obtain \( d_\tau \) via Eq. 5.10. In this particular case \( l(s) = \tau(a,b) + \tau(b,c) + \tau(c,d) \).

edges of the network. As both \( \rho \) and \( \tau \) can denote the length of a link, we use \( \tau \) to emphasize when the distance needs to be defined only for the edges of the network. Also, for a spatially embedded network we can define \( \tau \) as an arbitrary function of \( \rho \).

To estimate the dimensionality of such networks, we consider random walks similarly to Section 5.2.1, and exploit the inherent information in the \( \tau \) measure. Let \( s = (s_0, s_1, \ldots, s_m) \) denote a specific walk of length \( |s| = m \) from node \( s_0 \) to node \( s_m \). We can measure the cumulated length of a walk \( s \) consisting of \( m \) steps as

\[
l(s) = \sum_{i=0}^{m-1} \tau(s_i, s_{i+1}).
\]

(5.9)

Considering only walks which return to the origin, we denote the distribution of the lengths of such walks by \( P(l) \). We are then interested in the scaling relation:

\[
P(l) \sim l^{-\delta},
\]

(5.10)

where the proposed dimension measure can be estimated as \( d_\tau = 2\delta \) (if \( P(l) \) exhibits scaling in a suitably large range). Note, that this definition resembles that of the spectral dimension. Indeed, Eq 5.10 can be interpreted as a generalization of Eq 5.4. Assuming that the random walker travels with unit velocity, it is natural to interpret the \( \tau(u,v) \) distance as the “elapsed time” as perceived by the random walker while traversing the link between
Thus, by $d_\tau$ we achieve a dimension concept that grasps the network’s large-scale spatial structure through local characteristics observed by a random walker. Trivially, for a regular $d$-dimensional lattice $l \sim t$, which gives $d_\tau = d_s = d$. In Fig. 5.1 we illustrate the different approaches applied to measure a network’s dimension: $d_s$ for abstract networks, $d_\rho$ for spatially embedded networks, and $d_\tau$ for partially embedded networks.

5.3.1 The connection between $P_0(t)$ and $P(l)$

We note, that the length distribution of returning walks ($P(l)$ in Eq. 5.10), and the arising dimension measure $d_\tau$, can be considered the generalization of the $P_0(t)$ return probability in Eq. 5.4 and the $d_s$ dimension.

To show that, we now give a possible alternative definition of Eq. 5.4 for finite networks, which can be generalized to give the scaling relation Eq. 5.10 in a natural way. Let us define some upper limit $T$ and consider walks $s$ which have at maximum $T$ steps ($|s| \leq T$). Of course, for a finite network we will have a finite number of such walks. Now, among these walks, we shall consider those, that return to their origin:

$$S_0^{(T)} = \{ s : m \equiv |s| \leq T \text{ and } s_0 = s_m \}.$$  

(5.11)

Among these, we can define the discrete probability distribution $p_0^{(T)}(t)$ as the relative abundance of walks of length $t$ in $S_0^{(T)}$. For a finite network and a fixed $T$, the $p_0^{(T)}(t)$ distribution is simply the rescaled version of $P_0(t)$ in Eq. 5.4: $p_0^{(T)}(t) = C(T)P_0(t)$, where $C(T)$ is an appropriate normalizing constant so that $\sum_{t=1}^{T} p_0^{(T)}(t) = 1$. Hence, we obtain a scaling relation with the very same exponent as in Eq. 5.4:

$$p_0^{(T)}(t) \sim t^{-\alpha},$$  

(5.12)

This relation can be easily generalized to give an estimate of the scaling relation Eq. 5.10. In this case, instead of limiting the number of steps taken, we will introduce a limit $L$ for the distances along the walk: we will consider walks $s$ which have $l(s) \leq L$. Among these, we will again define the set of those walks which return to the origin, and consider their probability distribution denoted by $p_0^{(L)}(l)$. As the choice of $L$ does not alter the obtained scaling exponents, we choose an $L$ value appropriate for the system considered, and omit the superscript and write $P(l) \equiv p_0^{(L)}(l)$.

Note that now $l$ is a continuous variable, meaning that for a finite network, we have

$$P(l) \sim \sum_{\text{walks } s \text{ where } s_0 = s_m} \delta(l - l(s)),$$  

(5.13)

where $\delta(x)$ is the Dirac delta function. Choosing an appropriate binning size, $P(l)$ and the scaling exponents can be well estimated numerically.

5.3.2 Some properties of $d_\tau$

In accordance Section 5.3.1, for some finite $L$ value we define $P(l)$ as the distribution of lengths $l(s)$ of returning walks $s$ with $l(s) < L$. This distribution can be calculated by summing the individual distributions which arise from walks with a given number of steps:

$$P(l) = \sum_{i=1}^{\infty} p_i(l)$$  

(5.14)
where \( p_i(l) \) is the term corresponding to walks with \(|s| = i\) number of steps and maximum cumulated length \( L \). To give an example, the first four terms in the sum are:

\[
\begin{align*}
  p_1(l) &= 0 \\
  p_2(l) &= \frac{1}{k} P(2\tau = l) \\
  p_3(l) &= \frac{C(3)}{k^2} P(\tau_1 + \tau_2 + \tau_3 = l) \\
  p_4(l) &= \frac{1}{k^3} P(4\tau = l) + \frac{2}{k^3} P(2(\tau_1 + \tau_2) = l) + \frac{C(4)}{k^3} P(\tau_1 + \tau_2 + \tau_3 + \tau_4 = l)
\end{align*}
\]

Here, \( k \) denotes the average node degree, \( C(i) \) is the probability of finding a circle of \( i \) nodes (e.g. \( C(3) \) is the clustering coefficient) and \( P(\tau) \) denotes the distribution of link lengths (see also Section 5.2.2). The higher order terms (e.g. \( P(2(\tau_1 + \tau_2) = l) \) in Eq. 5.18) correspond to the distribution of the sum of the lengths of two or more links, which can be calculated by the convolution of \( P(\tau) \). This is only true in an uncorrelated network. In a real-world setting, the distribution of link lengths along circles of a specific length may deviate from the overall link length distribution. For instance, in case of \( C(3) \), geographically concentrated triangles are expected to be much more prevalent than ones spanning large distances [171]. In this case, the terms in Eq. 5.14 can only be determined by the numerical analysis of the specific network at hand. Still, we can make some general statements about the relevance of the different terms.

In a network where all link lengths are positive, we expect that the consecutive terms in Eq. 5.14 will be centered around increasing \( l_i \) values (with \( l_i \sim i l_1 \)). Also, there is certainly a minimum and a maximum length that can be spanned in \( i \) steps: \( p_i(l) \equiv 0 \) if \( l < i\tau_{\min} \) or \( l > i\tau_{\max} \), where \( \tau_{\min} \) and \( \tau_{\max} \) denote the minimum and maximum link length in the network, respectively. This means that for a finite \( L \), we will only have a finite number of nonzero terms in Eq. 5.14. More interestingly, for each \( \lambda < L \) length, there will be an \( i_0 \) value, that the terms after \( i_0 \) only give decreasing contribution to \( p_0(\lambda) \). This implies that for some \( i_1 \geq i_0 \), we can neglect the terms with \( i > i_1 \). Thus, for a small \( \lambda \) value, we have to take into account only a few terms in Eq. 5.14, and the behavior of \( p_0(l) \) in the \( l < \lambda \) case can be well approximated by examining the behavior of these few terms. In the case of the real-world networks considered in Sections 5.4.3 and 5.4.4, the behavior of \( P(l) \) for small \( l \) values can be readily explained by the \( P(\tau) \) link length distribution alone.

### 5.4 Results

To study the scaling relations described in Sects. 5.2.1 and 5.3, we considered simulations of random walks on synthetic and real-world networks. In accordance with Section 5.2.2 we specifically focused on the distribution of link lengths in the networks. All networks considered here have full embedding information: the distance \( \rho(i, j) \) can be calculated for
any (i,j) pair of nodes, and also \( \rho \) qualifies as a metric. In accordance with that, when calculating \( d_\tau \) we simply set \( \tau(i,j) = \rho(i,j) \) for node pairs (i,j) for which there is a link. This allows us to easily compare the two dimension measures \( d_\rho \) and \( d_\tau \).

### 5.4.1 OpenStreetMap

We used the road network of several cities and countries as a test case for the dimension measures presented previously. We obtained the data from the OpenStreetMap (OSM) database, which is a large collaborative project to create a free editable map of the world\(^2\). Due to the self-organizing nature of the project, OSM data are freely available in various flavors. We used several maps from which we extracted the underlying road network \(^3\). We regard the OSM networks as benchmarks of the dimensionality measurements, as locally they can be regarded as two dimensional lattices. Hence, we expect these networks to have estimated dimensions close to 2.

Fig. 5.2a shows the road network of New York City’s central urban area, obtained from the OpenStreetMap database. Note that the calculations were done on the entire New York area map, including suburbs and outskirts, which were omitted from this illustration. In Fig. 5.2b we depict the obtained scaling for \( d_\tau \) for the road network of the New York City area, an example where many parts of the network are close to a regular lattice. Table 5.1 shows the obtained dimensions for the several road networks considered. Not surprisingly, we have found that all three methods result in dimensions close to 2.

To inspect the sensitivity of the dimension measures, we introduced a modified version of the road maps, where we removed all nodes with degree 2 (i.e. if i is a node with \( d_i = 2 \) and with links (i,j) and (i,k) then we remove i and add the link (j,k) instead, recursively).

---

\(^2\)http://www.openstreetmap.org

\(^3\)As raw OSM networks contain rich auxiliary information (e.g. the outline of buildings) and also introduce network nodes to describe the curvature of road segments, we applied a thorough data cleaning procedure to eliminate these artifacts and obtain the clean road network.
Typically, these 2-degree nodes are included for the proper visualization of a road (e.g. a curved segment is actually stored as a sequence of chords), but do not carry information about the topological structure of the actual network. Our processing procedure can be interpreted as a coarse-graining step that reflects the road topology more precisely, while fading out the fine-resolution spatial details of the network. To see how this change in resolution affects the properties of the random walk processes, we also give the resulting dimension values in Table 5.1. For the majority of the maps there is no significant change in the estimated dimensions. The cleaning procedure only slightly modifies the exponents; in several cases, the quality of fits is better however. It can also be observed that \( d_\rho \) tends to produce higher values than \( d_s \) and \( d_\tau \), and that it slightly overestimates the embedding dimension of the road networks.

### Table 5.1: The estimated dimensions (best fit) for several road networks.

We depict the results for four metropolitan areas and three larger scale maps. Calculated statistical error of the exponents is approximately 0.05.

<table>
<thead>
<tr>
<th>City / Region</th>
<th>( d_s )</th>
<th>( d_\rho )</th>
<th>( d_\tau )</th>
<th>( d_s )</th>
<th>( d_\rho )</th>
<th>( d_\tau )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boston</td>
<td>2.0</td>
<td>2.2</td>
<td>1.9</td>
<td>1.9</td>
<td>2.1</td>
<td>1.9</td>
</tr>
<tr>
<td>Budapest</td>
<td>2.0</td>
<td>2.0</td>
<td>1.7</td>
<td>1.9</td>
<td>1.8</td>
<td>1.8</td>
</tr>
<tr>
<td>New York City</td>
<td>2.1</td>
<td>2.3</td>
<td>2.0</td>
<td>2.0</td>
<td>2.2</td>
<td>1.9</td>
</tr>
<tr>
<td>Rome</td>
<td>2.0</td>
<td>2.2</td>
<td>2.0</td>
<td>2.1</td>
<td>2.3</td>
<td>2.0</td>
</tr>
<tr>
<td>Connecticut</td>
<td>2.2</td>
<td>2.4</td>
<td>2.0</td>
<td>2.1</td>
<td>2.3</td>
<td>2.0</td>
</tr>
<tr>
<td>Estonia</td>
<td>2.0</td>
<td>2.2</td>
<td>1.8</td>
<td>2.0</td>
<td>2.3</td>
<td>1.8</td>
</tr>
<tr>
<td>New Mexico</td>
<td>1.7</td>
<td>2.0</td>
<td>1.6</td>
<td>1.9</td>
<td>2.2</td>
<td>1.9</td>
</tr>
</tbody>
</table>

Typically spatially embedded random networks with a given link length distribution, following the method of [111]. First, we construct a \( d = 2 \) dimensional regular lattice, and then for each node \( u \) we draw a random degree \( d_u \) from a Poisson distribution with mean \( k \) (note, that this process leads to a network with an average degree \( \approx 2k \)). Next, for each \( d_u \) links of \( u \) we generate a random length \( \rho_1 \) from \( P(\rho) \), and link to a random node \( v \) such that \( \rho(u,v) \approx \rho_1 \). We scale the lattice such that all coordinates fall between 0 and 1. Considering a lattice with linear size \( m \), we have \( N = m^d \) nodes. Consequently, the minimum distance between any two nodes is \( \rho_{\text{min}} = 1/(m-1) \), while the maximum distance is \( \rho_{\text{max}} = \sqrt{d} \). In the networks generated, we used power-law distributions \( P(\rho) = C\rho^{-a} \), where \( C \) is a normalizing constant such that \( \int_{\rho_{\text{min}}}^{\rho_{\text{max}}} P(\rho) d\rho = 1 \). Note that since \( P(\rho) \) has a finite support, we can use any exponent \( a \) (the distribution can be normalized even for \( a \leq 1 \)). For our simulations we used \( a = 1, 2, 3 \) and 4, and several different values of \( m \) and \( k \); the results shown here are for typical networks with \( m = 500 \) and \( k = 2 \), i.e. with \( N = 250,000 \) nodes and approximately \( L \approx 500,000 \) symmetric links. For each \( a \) value, eight realizations were generated to test for any variations due to the stochastic network generation process. Results displayed were typical among these.

In Fig. 5.3a we plot the return probability as a function of the steps taken for a specific realization of the spatially embedded random networks. Note that in this case we only used the connectivity information – the spatial embedding and \( P(\rho) \) only affect the results through the random generation of the networks. For \( a \geq 2 \) we get increasing ranges of...
Figure 5.3: Two different dimension measurement methods for synthetic networks. (a) The $d_s$ spectral dimension (Eq. 5.4) gives scaling in a substantially large range, with monotonically decreasing exponents. The resulting dimensions are $d_s = 4.75 \pm 0.02, 2.61 \pm 0.01$ and $2.163 \pm 0.003$ for the parameters $a = 2, 3$ and 4 respectively (the lines are fitted in the ranges $t \in [12,81], t \in [7,148]$ and $t \in [7,403]$). In the $a = 1$ case we cannot identify scaling, the return probability decreases faster than a power-law, which can be interpreted as an infinite dimension. (b) The $d_\rho$ dimension (Eq. 5.7). Here we have increasing exponents again as $a$ decreases, while in the $a = 1$ case there is apparently no scaling regime, which again indicates infinite dimension. The estimated dimensions are $d_\rho = 4.61 \pm 0.01, 2.174 \pm 0.001$ and $1.967 \pm 0.001$ for $a = 2, 3$ and 4. The lines were fitted for the ranges $r > 0.183, r > 0.03$ and $r > 0.015$ for $a = 2, 3$ and 4.

Figure 5.4: The $d_r$ dimension measure for synthetic networks with different link length distributions. (a) For $a = 2, 3$ and 4 the estimated dimensions are $3.71 \pm 0.01, 2.029 \pm 0.004$ and $1.918 \pm 0.003$ respectively (the lines were fitted in the ranges $l \in [0.1,3], [0.4,20]$ and $[0.4,20]$ for $a = 2, 3$ and 4). (b) For $a = 1$ there emerge two distinct scaling regimes. We have $d_1 = 1.63 \pm 0.01$ for the $l \lesssim \rho_{\text{max}}$ ($l \in [0.01,1.35]$) range and $d_2 = 7.75 \pm 0.02$ for the $l \gtrsim \rho_{\text{max}}$ ($l \in [3.15,8.2]$) range.
scaling regimes, with decreasing estimates of $d_s$. This is in accordance with the fact that with increasing values of $a$, there will be fewer long-range links and the network will be dominated by short-range links, which augment the underlying two dimensional structure. In the $a = 1$ case, we have no scaling – the return probability can be well approximated with the stretched exponential function $P_0(t) \sim \exp(\varepsilon t^\delta)$, which is also found in uncorrelated random networks (Erdős – Rényi networks), and in glassy systems [38]. We can interpret this result as the network having an infinite spectral dimension.

A similar phenomenon can be observed for the $d_\rho$ dimension in Fig. 5.3b. Networks with larger values of the $a$ exponent give good scaling relations and decreasing dimension estimates, while for $a = 1$ the broad link length distribution yields no scaling, which can again be regarded as infinite dimension.

We display the results for the new dimension measure $d_\tau$ in Fig. 5.4. In the $a \geq 2$ cases (Fig. 5.4a) we get apparent scaling behavior in all cases, with decreasing dimensions. However, in the $a = 1$ case, we get an anomalous behavior. As seen on Fig. 5.4b a separation of length scales emerges at approximately the size of the system ($\rho_{\text{max}} = \sqrt{2}$ in the case of our synthetic random graphs). We can identify two scaling exponents: $\delta_1$ for $l \lesssim \rho_{\text{max}}$ and $\delta_2$ for $l \gtrsim \rho_{\text{max}}$. Walks with $l(s) < \rho_{\text{max}}$ behave in a clearly different way, resulting in $\delta_1 < \delta_2$. This means that on the different length scales, different terms dominate in the summation of Eq. 5.14 (see Section 5.3.2). A plausible argument is that as the $a$ exponent decreases in $P(\rho)$, the long-range links gain prevalence and the finite size of the system affects its properties more. For instance, the mean value of link lengths is determined by $\rho_{\text{max}}$ in the $a \leq 2$ case (the mean becomes infinite if $a \leq 2$ and $\rho_{\text{max}} \to \infty$).

Based on this fact, one may argue that the $\delta_2$ exponent characterizes the system, and that the $\delta_1$ exponent is not relevant. Still, the $l < \rho_{\text{max}}$ range can provide valuable information about the system, especially if we can identify which terms give relevant contribution to the behavior of $p_0(l)$ in Eq. 5.14.

### 5.4.3 Internet infrastructure

We also study a sample of the Internet router network, which was obtained from a global topology discovery campaign presented in Ref. [137]. The locations of the nodes were measured by the Spotter method [117] which gives a median error of $\approx 30 \text{ km}$ providing sufficient city level precision. To minimize the effect of mislocalizations a thorough consistency test was applied on the collected data. The data set contains 13,120 addresses which are considered well positioned. Between these, there are 44,116 links for which both endpoints have reliable location information.

For our Internet data set, the distribution of link lengths can be approximated as $P(\rho) \sim 1/\rho$ over four orders of magnitude (see Fig. 5.5 and also [137]).

The results of the random walk processes are shown in Fig. 5.6 for the three dimension measures. The spectral dimension shows scaling, although only for a rather narrow range. The resulting dimension is $d_s = 3.06$, which is higher than the embedding dimension.

The $d_\rho$ dimension is not applicable, as instead of the power–law relationship we get an exponential decay, as it is shown on the lin–log plot of Fig. 5.6b. Again, we can interpret this result as an infinite $d_\rho$ dimension.

Considering the $d_\tau$ dimension, similarly to the $a = 1$ case in the synthetic networks, we have two scaling regimes, which are again separated by the approximate size of the system (now the longest link can be at most $\approx 20,000 \text{ km}$ long). We have $\delta_1 \approx 1$, implying $d_\tau \approx 2$. While this would be an appealing result as $d_\tau$ would reproduce the embedding
Figure 5.5: The distribution of link lengths for the real-world datasets considered in this chapter. Left: CCDF of link lengths for the Internet infrastructure and the Gowalla geosocial network. Right: the same for the two kinds of Twitter user networks (follower and mention). In all cases the CCDF shows a linear decay over 4 orders of magnitude on the log-lin plot, indicating that \( P(\rho) \sim 1/\rho \). A more detailed analysis for the Internet dataset was already presented in [137], while for the Twitter follower network in [8]. The Pearson correlation coefficients between the CDF values and the logarithms of distances are \( C_{\text{Gowalla}} = 0.994 \), \( C_{\text{Internet}} = 0.991 \), \( C_{\text{follower}} = 0.996 \) and \( C_{\text{mention}} = 0.981 \) indicating that linear dependence is indeed significant.

dimension, we note that this can also be readily explained by arguments based on the link length distribution \( P(\tau) \). Considering Eq. 5.14 (see Section 5.3.2), if we only take the first term \( (p_2) \) into account, we get \( P(l) \sim P(\tau = l/2) \sim 1/l \) in this case. This means that the behavior observed here can be explained by the argument that all the other terms give only insignificant contribution on length scales \( 100 \text{ km} \lesssim l \lesssim 20,000 \text{ km} \), and thus the scaling is dominated by the term proportional to the distribution of link lengths. For the larger length scales, we get \( \delta_2 = 1.44 \), giving \( d_r = 2.87 \), which is larger than the embedding dimension, but slightly slower than the \( d_s \) spectral dimension measure.

5.4.4 Gowalla geosocial network

Gowalla was a geosocial network where users also provided spatial information via “check-ins” (the service was shut down in 2012 after an agreement with Facebook\(^4\)). We used the dataset shared by Cho et al. described in Ref. [50], which is freely available at the Stanford Large Network Dataset Collection website\(^5\). This dataset includes 196,591 users (nodes) and 950,327 (undirected) links (friendships), with a total of 6,442,890 check-ins, where a check-in is the position of a specific user at a specific time. The dataset includes check-ins collected in the interval between Feb. 2009 and Oct. 2010.

\(^4\)Original website: http://www.gowalla.com; as of August 2012, the gowalla.com domain is no longer available, see for example (the version cached by Google): http://webcache.googleusercontent.com/search?q=cache:blog.gowalla.com/post/13782997303/gowalla-going-to-facebook; on the other hand, as of May 2015, while the gowalla.com domain in itself is still unavailable, the blog.gowalla.com URL is available and is served by a DNS redirect through domains.tumblr.com giving what seems to be an archived version of the blog of the Gowalla developers

\(^5\)Stanford Large Network Dataset Collection, http://snap.stanford.edu/data
Figure 5.6: Comparing the different dimension measures for the Internet and the Gowalla geosocial network. (a) For $d_s$ the network exhibits a scaling behavior only in a limited regime with estimated spectral dimension $d_s^{(I)} = 3.06$ and $d_s^{(G)} = 4$. The fitting ranges are $t \in [29,330]$ and $t \in [15,148]$ for the Internet and Gowalla datasets respectively.(b) The $d_p$ dimension measure for the Internet and the Gowalla geosocial network. In this case we cannot identify a scaling behavior in any of the networks; the beginning of the curves seems to exhibit an exponential decay (note that the $x$ axis is linear and the $y$ is logarithmic). (c) The $d_\tau$ dimension measure for the Internet infrastructure network. Similarly to the synthetic case, we have two distinct scaling regimes, and the scaling behavior changes at approximately the size of the system. The dimensions are $d_{\tau,1}^{(I)} = 2.1 \pm 0.04$ for $l \lesssim \tau_{\text{max}}$ and $d_{\tau,2}^{(I)} = 2.87 \pm 0.02$ for $l \gtrsim \tau_{\text{max}}$. The fitting ranges are $l \in [148 \text{ km}, 8103 \text{ km}]$ for $d_{\tau,1}^{(I)}$ and $l \in [36315 \text{ km}, 268337 \text{ km}]$ for $d_{\tau,2}^{(I)}$. (d) The $d_\tau$ dimension measure for the Gowalla geosocial network. Again, we have two scaling regimes.In the first regime we get $d_{\tau,1}^{(G)} = 2.11 \pm 0.06$ for $l \lesssim \tau_{\text{max}}$, and we have $d_{\tau,2}^{(G)} = 3.99 \pm 0.02$ for $l \gtrsim \tau_{\text{max}}$. The fitting ranges are $l \in [544 \text{ km}, 8103 \text{ km}]$ for $d_{\tau,1}^{(G)}$ and $l \in [22026 \text{ km}, 268337 \text{ km}]$ for $d_{\tau,2}^{(G)}$. 

92
As the location of these check-ins varies, we used a preprocessing technique to assign a unique position to the users (see Appendix A.2). In our analysis, we only included users where these “typical” positions could be determined with sufficient confidence. This resulted a network which consists of 94,798 nodes with reliable position information and 289,961 links between them. Examining the distribution of link lengths, we get a behavior which is very similar to that observed for the Internet. Plotting the CCDF, it can be well approximated with $\ln(\rho)$, indicating that the PDF scales as $P(\rho) \sim 1/\rho$ (see Fig. 5.5).

The evaluation of the previous dimension concepts is illustrated in Fig. 5.6 for the three dimension measures. The qualitative behavior is quite similar to that obtained for the Internet, while there are quantitative differences. The spectral dimension is $d_s = 4$, although the scaling regime is again rather limited. The $d_\rho$ dimension again does not produce scaling, but exhibits the exponential decay also seen previously.

Considering the $d_\tau$ dimension, we can again identify the two scaling regimes, with the boundary at $\approx 20,000 \text{ km}$, the approximate size of the system. Similarly to the Internet, we have $\delta_1 \approx 1$, giving $d_\tau \approx 2$ for the $l \leq 20,000 \text{ km}$ range. For the longer walks, we get $\delta_2 = 1.997$ giving $d_\tau = 3.99$, which is again larger than the embedding dimension, and is approximately the same as $d_s$.

Based on our results, we argue that some, but not all of the qualitative aspects of these two real-world networks of different origin can be explained with the effect of the very broad link length distribution $P(\rho) \sim 1/\rho$. Similarly to the random networks, $d_\tau$ shows the same separation of length scales, and again we find that $d_\rho$ is not applicable here. However, for these networks, $d_\rho$ exhibits a clear exponential decay which was not the case for the synthetic network. Also, $d_\tau$ behaves in a quantitatively different way: the arising $\delta_2$ exponents differ significantly from the synthetic case, and also between the two real-world networks. Regarding the $d_s$ spectral dimension, we find significantly better scaling compared to the synthetic case. These observations mean that the link length distribution is not sufficient in itself to entirely control the dimensionality of complex networks, but further structural properties are also expected to have a traceable effect. In case of $d_\tau$, these structural characteristics can be approached via the varying significance of the different terms in Eq. 5.14. A rather interesting property is that for the $l \leq 20,000 \text{ km}$ range both the Internet infrastructure and the Gowalla network gives $\delta_1 \approx 1$, implying $d_\tau \approx 2$, matching the embedding dimension of the system. In both networks, the link length distribution can be well approximated by $P(\tau) \sim 1/\tau$. Using this, the scaling $P(l) \sim 1/l$ can be explained by the arguments presented in Section 5.4.3, based on Eq. 5.14. Still, the synthetic random networks produce different exponents, meaning that this peculiar property arises only in these two real-world networks of different origin.

### 5.5 Randomized versions of the Gowalla network

In this section, I present results for random networks generated in a way to better match the properties of real-world networks presented in the previous sections. To achieve this, we use the locations of nodes in the Gowalla network as input (instead of a regular grid as in Section 5.4.2) and generate a random network with a specified number of links and link length distribution according to Algorithm 2. It essentially works by taking a set of points on the surface of the Earth, and a desired link length distribution $P(l)$, generating link lengths independently for $L$ links, and for each link, trying to find points which are separated by the distance according to the generated link length up to a given relative error. This way, there are two main differences from the random networks in
Algorithm 2 Algorithm to generate a random network among a given set of points on the surface of the Earth. \( V \) is a set points to be used as nodes in the network, \( L \) is the number of links to generate, \( P(l) \) is the desired link length distribution, and \( d(x,y) \) is the distance between points \( x \) and \( y \). For each edge, we generate a desired length \( l_1 \) from the given link length distribution, and then try to find two points which are \( l_1 \) distance apart. The \( \varepsilon \) parameter gives a maximum relative deviation from \( l_1 \) to accept for the distance. While evaluating all possible point pairs (\( O(|V|^2) \) possible combinations) would be prohibitively resource-intensive (either in terms of computing power or memory or possibly both) even for networks with a few tens of thousands of nodes, we apply a heuristic search: we choose a random \( x \in V \) point and check if there are any points whose distance falls in the range of \([l_1(1-\varepsilon); l_1(1+\varepsilon)]\); this means finding points which fall between two circles of sizes \( l_1(1-\varepsilon) \) and \( l_1(1+\varepsilon) \) respectively, which can be done efficiently with the help of the HTM indexing scheme (see Section 3.3 and Refs. [187, 41]) on the surface of the Earth. As this search might give no result for a specific pair of \( l_1 \) distance and \( x \in V \) point, for each distance, we allow at maximum \( n \) attempts with different random points. In practice, the parameters \( \varepsilon = 0.01 \) and \( n = 10 \) worked well when generating random networks with power-law link length distributions and using the coordinates of the Gowalla users as the \( V \) set of points.

\[
\text{for all } L \text{ links to be generated do}
\]
\[
\text{\hspace{1em}} \text{repeat}
\]
\[
\text{\hspace{2em}} l_1 \leftarrow \text{random length from the specified } P(l) \text{ distribution}
\]
\[
\text{\hspace{2em}} \text{repeat}
\]
\[
\text{\hspace{3em}} x \leftarrow \text{random node from } V
\]
\[
\text{\hspace{3em}} S(x, l_1) \leftarrow \{ y \in V : y \neq x, |d(x,y) - l_1| < \varepsilon l_1 \}
\]
\[
\text{\hspace{3em}} \text{if } S(x, l_1) \neq \emptyset \text{ then}
\]
\[
\text{\hspace{4em}} \text{choose random node } y \text{ from } S(x, l_1)
\]
\[
\text{\hspace{4em}} \text{add } x - y \text{ edge}
\]
\[
\text{\hspace{3em}} \text{end if}
\]
\[
\text{\hspace{2em}} \text{until an edge was chosen or at most } n \text{ attempts were made for } l_1
\]
\[
\text{\hspace{1em}} \text{until an edge was chosen}
\]
\[
\text{end for}
\]
Section 5.4.2: (1) instead of a two dimensional square, points are distributed on the surface of a sphere; (2) instead of a regular grid, points are distributed according to the positions of users in the Gowalla network. While the process described in Algorithm 2 is conceptually simple, due to the uneven distribution of points on the surface of the Earth, it can become computationally challenging even for moderately-sized networks: in the case of the Gowalla network, runtimes were the range of $30 - 60$ minutes on a typical desktop computer; running it for larger networks (e.g. the Twitter user networks, see Sections 2.2.1 and 5.5.1) seemed unfeasible without investing significant effort in further improving the applied spatial indexing implementation.

In accordance with the observed link length distribution in the various real-world networks studied in this chapter, and the previous study of random networks, we generated networks with power-law link length distributions $P(l) \sim l^{-a}$ for $a = 1$ and 2. Using $a = 3$ was also attempted, but it resulted in networks consisting of many small disconnected components; as any modification of Algorithm 2 to keep the network connected could easily introduce various artifacts in the resulting network structure, we did not pursue any such option, instead we focus here on exponents $a = 1$ and 2. To be able to use $P(l)$ as a probability distribution, we have to specify the range of allowed $l$ values and calculate its normalization appropriately. Since the maximum possible distance is naturally limited by being on the surface of a sphere, we only need to specify a minimum distance $l_{\text{min}}$; from a theoretical perspective, this ensures that $P(l)$ can be normalized, while from an empirical perspective, this corresponds to the fact that even friends don’t live arbitrarily close to each other and to the computational requirement that distances generated from $P(l)$ should correspond to distances between the points in the dataset. As our intention was to match the empirical link length distribution of the Gowalla network, we chose $l_{\text{min}} = 637 \text{ m (0.0001 rad)}$. Typical resulting link length distributions are then shown in Fig. 5.7.

We display the dimension measures for two typical realizations (for $a = 1$ and $a = 2$ exponents) in Fig. 5.8. Qualitatively, the results are similar to the previous random networks (see Figs. 5.3 and 5.4 in Section 5.4.2), while the distinction between the $a = 1$ and $a = 2$ cases is much more pronounced. We speculate that this is a consequence of the non-uniform distribution of points which can influence the structure of the resulting networks in a non-trivial way. In the case of the spectral dimension ($d_s$, see Fig. 5.8a),

![Figure 5.7: Link length distribution in random networks generated using the Gowalla users' locations. Left: $P(l) \sim 1/l$ case, with comparison to the original network; right: $P(l) \sim 1/l^2$ case. Link length distribution in the random networks is much more “regular” as expected.](image)
Figure 5.8: Dimension measures for random networks among the Gowalla users. (a) The spectral dimension shows that the two networks with different link length distributions \((a = 1)\) and \((a = 2)\) behave in a prominently different way. The \(a = 1\) case shows an exceptionally fast relaxation; the power-law fit in range gives \(d_s = 10.34\), while considering the rapid relaxation and the very limited range of the fit, one could argue that the spectral dimension concept is not applicable here, or the dimension should be considered infinite instead. In the \(a = 2\) case, we have \(d_s = 1.69\), remarkably smaller than in any previous case, corresponding to a very slow relaxation, which might be the consequence of the network consisting of many loosely connected parts. (b) The \(d_\rho\) dimension measure; in the \(a = 1\) case (similarly to previous random and most real-world networks with \(a = 1\)), we have no scaling, which can be interpreted as \(d_\rho \to \infty\). For the \(a = 2\) case, the somewhat limited scaling range gives \(d_\rho = 2.67\), a significantly larger value than \(d_s\). (c) Illustration of the diffusion process as observed in the evolution of the rms distance from the origin. The fitted power law is \(r \sim t^\beta\), with \(\beta = 0.316\), indication sub-diffusion. (d) Estimating the \(d_\tau\) measure. For the \(a = 1\) case, we again have two scaling regimes with \(d_{\tau,1} = 1.855\) and \(d_{\tau,2} = 6.28\). In the \(a = 2\) case, we get \(d_\tau = 1.46\) for the whole range, a value even smaller than \(d_s\).
in the $a=1$ case we have a very fast relaxation to the stationary solution of Eq. 5.1. A power-law fit for this very limited range gives $d_s = 10.34$, while we can argue that interpreting this result as an infinite dimension is more appropriate in this case. On the other hand, for $a=2$, the relaxation is exceptionally slow giving a dimension measure of $d_s = 1.69$. This is significantly smaller than spectral dimension measures obtained for previous random networks (Fig. 5.3a), indicating that the distribution of points can have a significant effect on the resulting network structure even in the case of the simple random network models studied here. The slow relaxation indicates that the network is made up of loosely connected components; this can be a consequence that using the $a=2$ exponent to generate the network, dense regions of points form well-connected clusters as short links are preferred, while establishing links between these clusters (i.e. over a range of more sparsely inhabited areas) occurs rarely. We note that in this case, the resulting networks still span the whole Earth, while networks generated with similar parameters and $a=3$ exponent were already disconnected.

Looking at $d_\rho$, we can draw similar conclusions (Fig. 5.8b). Here, $d_\rho$ is again not applicable for the $a=1$ case; this is a consequence of the broad link length distribution, as after only one step, rms displacement from the origin is already in the range of several thousand kilometers. In the $a=2$ case, we have scaling in a limited range giving $d_\rho = 2.67$ a value significantly larger than before, indicating that the diffusion process differs when looked at in the abstract space of the network topology and the embedding space. Furthermore, this difference indicates an anomalous diffusion process in the embedding space, since the two dimension measures are connected through the $\beta$ diffusion exponent according to Eq. 5.8. Looking at the diffusion process, we get $\beta = 0.316$, indicating that the random walk process on the network corresponds to sub-diffusion in the embedding space (Fig. 5.8c).

Looking at $d_\tau$, we see a behavior similar to the previous cases. For the $a=2$ case, we have scaling in a large range, giving a dimension estimate of $d_\tau = 1.46$, even smaller than $d_s$. For the $a=1$ case, we again have two distinct scaling regimes, which are again separated approximately by the maximum link length, i.e. $\approx 20,000$ km. The dimension measures are $d_{\tau,1} = 1.855$ and $d_{\tau,2} = 6.28$.

### 5.5.1 Twitter user networks

We use the user networks extracted from the Twitter dataset (see Section 2.2.1 and also Ref. [6]), namely the mention and follower graphs constructed among geo-located users. A main difference from the Gowalla geosocial network presented in the previous section is the size of these networks, which is much larger. Also, while comprising of much more nodes and links, these networks form only a sample of the complete picture of relations among Twitter users. This is due to the data collection limits imposed by Twitter but also a consequence of the fact that we only include users who post GPS coordinates with their tweets so that we can have a geographical embedding. After more than three years of data collection, we identified over 26 million individual users who posted at least one tweet with attached GPS coordinates. This is just a fraction of almost 143 million individual users in the whole dataset. We further limit the set of users considered to those whose “typical” position can be estimated with adequate confidence. As each user can have possibly hundreds or even thousands of coordinates, we apply a clustering procedure as described in Ref. [6] to the coordinates of each user; we prune the obtained clusters until all coordinates fall inside a $3\sigma$ range and then take the average coordinates in the
largest cluster as a location estimate. In the present analysis, we then limit the users considered to those who have at least 15 coordinate pairs in the largest cluster, i.e. at least 15 coordinate pairs were used as the basis of their location estimate. This results in a total of 6.87 million users with typical coordinates which we consider to have adequate accuracy for further analysis.

Having a set of geo-located users, we then extract the user networks from the Twitter dataset. We construct the mention network based on the content of the tweets in the downloaded sample, i.e. a link in the network corresponds to a user mentioning another user (with preceding their user name with an @ character) in their tweets. On the other hand, we obtained the follower network among the most active set of geo-located users with a separate download campaign [6, 8]. In this case, the size of the graph is further constrained by limits on download rates in the Twitter API. In the following, we only consider mutual links in both networks so as to filter out links due to spammers, Twitter-specific services, celebrities and news sources, etc. We speculate that a mutual link in either the mention or follower network more accurately accounts for real interaction taking place among users. We then consider the largest connected component in these networks. This way, the mention network contains 4,209,543 users and 10,658,331 symmetric edges, while the follower network contains 2,849,375 users and 20,848,273 edges.

Looking at basic properties of these networks, we find that both have a broad degree distribution, but there is still a distinction between the two (see Fig. 5.9). Clearly, based just on the number of edges, the follower network is much denser; this can be seen in the degree distribution as there are much more high degree nodes in this network than in the mention network. Apart from difference in average or typical degree values, the two distributions show more fundamental differences: the degree distribution of the follower network can be modeled with a power-law $P(d) \sim d^{-2.98}$ for a quite large range, while in the case of the mention graph, a power-law fit only approximates the tail of the distribution, and the exponent is much higher, around 4.75, indicating a much less broad distribution in general. This observation is plausible given that establishing a mutual follower relationship is a less direct interaction than mentioning each other mutually; based on this, we could expect not only smaller average degrees but generally a less broad degree distribution for the mention network. Note that there could also be a sampling bias in this case; since

![Figure 5.9: Degree distribution of the Twitter user networks.](image)

The straight lines are fitted power-law functions with exponents 2.98 and 4.75 in the case of the follower and mention graph respectively. The fits were carried out with the algorithm described in Ref. [53].
Figure 5.10: Comparing the different dimension measures for the Twitter user networks.
(a) For $d_s$ both networks exhibit a scaling behavior only in a limited regime with estimated spectral dimension $d_s^{(M)} = 3.27$ and $d_s^{(F)} = 4.55$ for the mention and follower networks respectively. (b) The $d_\rho$ dimension measure, fitting a power-law function for the double logarithmic data. In both cases, the ranges of the fit is very limited; this is mainly caused by the rapid diffusion process. After just one step, the rms displacement from the origin is already in the range of a few thousand kilometers, while the rms displacement after relaxation is not more than 10,000 km. This is of course due to the heterogeneous distribution of link lengths. The fit is more convincing for the mention graph, spanning almost the entire distance range. The fitted exponents are $d_\rho^{(M)} = 7.56$ and $d_\rho^{(F)} = 9.35$ for the mention and follower graphs respectively. (c) Return probability as a function of the rms displacement from the origin, exponential fit (semi-logarithmic plot). Especially in the case of the follower network, an exponential form for $P_0(r)$ seems similarly plausible as a power-law. (d) The $d_\tau$ dimension measure for the Twitter user networks. The dimension measures are $d_\tau^{(M)} = 2.35$ and $d_\tau^{(F)} = 3.65$ for the mention network and $d_\tau^{(M)} = 2.36$ and $d_\tau^{(F)} = 4.88$ for the follower network.
the mention network was constructed from the text of the sample stream of Twitter activity, most mentions were actually missed. On the other hand, this sampling bias actually corresponds to just a filtering in a somewhat random fashion, i.e. using only links which are strong enough to appear in the sample stream at least once. This way, we still expect that results gained from the sample stream to be relevant, while ideally, repeating the tests on a more complete data source and analyzing the effect of filtering the network by link weights could clarify the effects of sampling. Looking at the link length distribution, we find that it can again be well approximated by the $P(\rho) \sim 1/\rho$ form for both networks (see Fig. 5.5 and also Ref. [8] in the case of the follower network, where it was analyzed in the context of performing distributed search), similarly to the other two real-world network datasets studied in this chapter.

Results for the dimension measures are shown in Fig. 5.10. The standard spectral dimension $d_s$ (Fig. 5.10a) shows relatively good scaling, especially in the case of the follower network. The resulting dimension measures are $d_s^{(M)} = 3.27$ and $d_s^{(F)} = 4.55$ for the mention and follower networks respectively, again significantly higher than the embedding dimension. In the case of the $d_\rho$ measure, the picture is somewhat more complicated as shown in Fig. 5.10b. In the case of the mention network, we have a relatively good fit, with $d_\rho^{(M)} = 7.56$ in almost the entire range of $r$ values, although this range is still rather short as rms distance from the origin of a random walk is almost $2000 \text{ km}$ already after the first step, while it does not exceed $10000 \text{ km}$ even after relaxation. This is of course the consequence of the broad link length distribution and the fact that distance between any two points on the surface of the Earth is bounded by approximately $20000 \text{ km}$. In the case of the follower network, the fit here is much less convincing, yielding an even higher exponent of $d_\rho^{(F)} = 9.35$. Looking at the same plots on a semi-logarithmic scale (Fig. 5.10c), we see that in the case of the follower network, an exponential function approximates the data equally well, while in the case of the mention network the exponential fit seems much less plausible. Comparing this with previous results on the Internet infrastructure, the Gowalla, and random networks with $a = 1$ link length distribution exponent, we can conclude that the mention network seems to be the exceptional case here, where a finite, although quite high $d_\rho$ value is plausible, while in all other cases, the exponential or even faster decay of $P_0(r)$ suggested $d_\rho \to \infty$.

Looking at $d_\tau$, we see qualitatively similar results to the previous real-world networks (see Fig. 5.6) and to the synthetic network with similar link length distribution (Fig. 5.4b). For both networks, we have two scaling regimes, which are again separated by approximately the maximum possible link length, i.e. $20,000 \text{ km}$. In this case however, the dimension measure for $l \lesssim 20,000 \text{ km}$ is apparently larger than 2: it was found to be around $d_{\tau,1} \approx 2.35$ for both networks. While in the case of the Internet infrastructure and Gowalla networks previously, a $d_{\tau,1} \approx 2$ result was found to match the embedding dimension, which was then speculated to be the consequence of the link length distribution, here, it seems that higher order terms in Eq. 5.14 are also significant, even for small $l$ values. Assessing the significance of individual terms via a more detailed statistical analysis of the walks returning to the origin is beyond the scope of the current work however. Looking at scaling for $l \gtrsim 20,000 \text{ km}$, we find that the two networks again show quite different behavior. The dimension estimates are $d_{\tau,2}^{(M)} = 3.65$ for the mention network and $d_{\tau,2}^{(F)} = 4.88$ for the follower network; these are qualitatively similar to the results for $d_s$, with only slightly larger values in both cases.
5.6 Discussion

In this chapter I presented work regarding the applicability of the concept of fractal dimension to complex networks via the analysis of random walk processes on the network. The main contribution is the introduction of a generalization of the spectral dimension of networks applicable to networks with full or partial spatial embedding. Contrary to recently proposed methods to measure the dimension of spatially embedded networks [61], this definition does not utilize global information on the embedding (i.e. the distance between any two nodes in the network), but exploits only local, link-wise information arising naturally from a random walk process. Consequently, the new dimension concept grasps the large-scale structure as seen “from the eyes of the random walker”. An appealing property of this method is that it can be readily applied to partially embedded networks, where distances or delays are only available as a link-wise property and not generally. In the case of a spatially embedded network, these delays can be a simple function of the distances between the nodes, but we can utilize arbitrary delays possibly arising from physical processes on the network. A consequence of this relaxation is that this link-wise property does not need to qualify as a metric in a mathematical sense. Indeed, in many real-world settings delays can violate the triangle inequality, as taking a “detour” may result in less accumulated delay.

To compare the new dimension concept (\(d_\tau\)) with the classical spectral dimension (\(d_s\)) and the method of Daqing et al. [61] (\(d_\rho\)), we considered spatially embedded random networks, and different real-world networks: road networks, Internet routers, the Gowalla geosocial network and two kind of networks defined among Twitter users. We used the road networks as benchmarks, since they substantially resemble a two dimensional lattice. As expected, all three methods provide dimensions \(\approx 2\). We also studied the effect of coarse graining the road networks and found that in most cases the methods yield similar results. Our random networks were generated to have a scale-free distribution of link lengths: \(P(\rho) \sim \rho^{-a}\). We found that for exponents \(a \geq 2\) all methods exhibit scaling, and in accordance with [61], the value of \(a\) directly affects the measured dimension of the network.

Contrary to previous works that we are aware of, here we have also considered the theoretically and empirically designated case of \(a = 1\). In the case of the random networks considered here, for this very broad distribution of link lengths, \(d_s\) and \(d_\rho\) decay faster than a power law, and thus can be considered infinite. Interestingly, \(d_\tau\) remains finite, but there appears a separation of length scales with two different exponents. In accordance with several empirical studies [206, 118, 125, 137, 83, 50, 8, 94], the real-world network datasets studied here can also be well characterized by \(a \approx 1\). An important empirical result is then that in these cases too, \(d_\tau\) remains finite, giving a possible indicator of network structure which can be directly applied here.

For the Internet, these results seem to contradict the previous result of Ref. [61], where \(d_{\rho}^{(I)}\) was found to be 4.5. A possible explanation for this mismatch lies in the nature of the datasets applied. For instance, there is a significant difference in the distribution of link lengths: in Ref. [61], the authors find \(a \approx 1.6\) \(^6\). We believe that due to the applied geolocalization technique, the Internet infrastructure dataset studied here is more accurate in terms of the geographic position of the nodes [117, 137].

Comparing the results for the real-world and synthetic networks in the \(a = 1\) case, we

\(^6\)The \(\delta\) exponent used in Ref. [61] corresponds to the conditional probability, which they estimate based on the assumption that the network nodes are uniformly distributed.
find a qualitatively similar behavior. While $d_\rho$ is not applicable for most networks (i.e. with the exception of the Twitter mention network), for $d_\tau$ there emerge two scaling regimes with different exponents, separated by the system size. On the other hand, the behavior of $d_s$ is apparently different: in all real-world networks there is a scaling regime, which is absent in the synthetic settings. Furthermore, even the behavior of $d_s$ is quantitatively different: the arising exponents differ both between the synthetic and real world case, and between the two real-world networks. This result implies that an uncorrelated random network model is not able to reproduce the structure of a real network with a similar $P(\rho)$. Thus, in spite of its distinguished significance, the link length distribution is not sufficient in itself to characterize the dimensionality of a complex network.
In this thesis, I presented research focused on several complex networks which provide novel opportunities for the statistical analysis of human behavior. In the first part, I focused on the Twitter social network, where a large sample of public messages is freely available for research. Focusing on messages which have the poster’s geographic coordinates attached to them, in Chapter 2, I presented a method for identifying relevant regional variation of the content. The method is a variation of latent semantic analysis, employing the Robust Principal Component Analysis technique to separate non-Gaussian outliers from low-rank structure. An appealing property of this methodology is that it performs this separation and identifies relevant sources of variation in the data without needing any a priory assumption about the possible structure, i.e. in an unsupervised manner. Using this method, I was able to show that there are indeed present highly localized outliers and smooth variations in language use. The former can be characterized both as localized topics of interests and Twitter-specific service providers and advertisements. In the low-rank structure, the main features found were related to slang use, urban lifestyle, tourism and religion.

In Chapter 3, I presented some of the technical challenges of handling geographic data in the range of billions of points. More specifically, I focused on the problem of assigning each of these points into administrative regions represented by arbitrary complex polygons, and presented an efficient solution which is also well integrated to the database system which is used to store the data.

In Chapter 4, I focused on the Bitcoin digital currency system which presents a unique opportunity to study the movement of money on everyday scales as every transaction which took place since its inception is publicly available to anyone. I studied the structure of the transaction network by looking at the time evolution of network characteristics. I presented evidence that linear preferential attachment describes the statistics of link formation in the network, while sublinear preferential attachment is present in the evolution of the wealth distribution. I linked the two processes by showing that there is a positive correlation between the network degree and the wealth of nodes. Focusing on the temporal evolution of the network structure, I showed that it can be linked to variation of the exchange price of bitcoins a measure entirely determined by the market.

In Chapter 5, I presented a study dealing with the topic of fractal dimensionality in the context of complex networks. Extending the standard concept of spectral dimension, I gave a possible new definition of fractal dimension applicable to networks with full or partial spatial embedding. Using this new definition, I performed measurements on several real-world and synthetic networks, with a distinct focus of the effect of the distribution of link lengths on the network dimensionality.

Concluding, I hope that the work and results here grasp some of the possibilities which lie in the rapidly increasing amount of data produced by everyday activity of people and will provide valuable and inspirational for researchers from diverse scientific backgrounds.
Összefoglalás

Doktori dolgozatom fókuszában néhány komplex hálózat állt, amik izgalmas új lehetődégeket jelentenek az emberi tevékenység statisztikai módszerekkel történő vizsgálatára. Az első részben a Twitter közösségi hálózattal foglalkoztam, ahonnan a felhasználók által nyilvánosan megosztott üzenetekből egy jelentős méretű minta szabadon elérhető tudományos kutatás céljából is. A földrajzi koordinátákkal is ellátott üzenetek koncentrált fel az üzenetek tartalmában levő regionális variabilitás kimutatása. Az itt bemutatott módszer egy előnyös tulajdonsága az, hogy nem igényel előzetes feltételezéseket az adatok struktúrájáról, így jól használható az adatokban rejlő változó komponensek is. Az előbbiek lokalizált témákhoz, Twitter-spezifikus szolgáltatásokhoz és reklámokhoz, míg az alacsony rangú struktúrában a fő variabilitás szleng elemek használatahoz, a nagyvárosi életformához, turizmushoz és vallássághoz köthető.

A 3. fejezetben egy technikai kihívást mutattam be, ami több milliárd pont kezelése során előfordul. A probléma, amivel foglalkoztam, a pontok tetszőleges komplex határvonalla rendelkező adminisztratív régiókba sorolása, ennek kezelésére egy hatékony megoldást fejlesztettem ki, ami jól illeszkedik a kutatás alatt használt adatbázisrendszerhez is.


A 5. fejezetben a fraktál dimenzió fogalmának hálózatokra való alkalmazhatóságát kapcsolatos kutatásainak bemutatásátra koncentrálhatsz. Az adatok alapján nem csak a hálózatok struktúrának időbeli változását, hanem az adatok alapján is megfigyelhető és vizsgálható, hogy a hálózatok struktúrájának változásait megfigyeli a rendszer.

Zárásképpen remélem, hogy az itt bemutatott munkával és eredményekkel sikerült megragadnom néhányat a mindennapi tevékenységek által létrehozott, gyorsuló ütemen növekvő adatmennyiségekben rejlő lehetőségek közül, és ez tudományterületek minél szélesebb körének művelői számára hasznos és inspiráló lesz a jövőben.
Publications

Publications supporting the thesis


Other relevant publications


References


[93] Herodotus. *An Account of Egypt (History, II:2)*.


tails in e-mail communication. PNAS, 105(47):18153–18158, 2008.

[133] B Mandelbrot. How long is the coast of britain? Statistical self-similarity and fractional

[134] Rosario N. Mantegna and H. Eugene Stanley. An Introduction to Econophysics: Correlations


Building a prototype for network measurement virtual observatory. Proceedings of the 3rd

[137] Péter Mátray, Péter Hága, Sándor Laki, Gábor Vattay, and István Csabai. On the spatial


[139] Géza Meszéna, Mats Gyllenberg, Frans J. Jacobs, and Johan a J Metz. Link between popu-

1963.


[142] Alan Mislove. Online Social Networks: Measurement, Analysis, and Applications to Dis-

[143] Lewis Mitchell, Morgan R Frank, Kameron Decker Harris, Peter Sheridan Dodds, and
Christopher M Danforth. The geography of happiness: Connecting Twitter sentiment and

[144] Fred Morstatter, Jürgen Pfeffer, Huan Liu, and Kathleen M Carley. Is the Sample Good
Enough? Comparing Data from Twitter’s Streaming API with Twitter’s Firehose. In ICWSM

In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery
and data mining, 2012.

[146] Máté Nagy, Zsuzsa Ákos, Dóra Bíró, and Tamás Vicsek. Hierarchical group dynamics in


114


[191] The Bible (King James translation), Matthew 13:12. “For whosoever hath, to him shall be given, and he shall have more abundance: but whosoever hath not, from him shall be taken away even that he hath.”


Appendix

A.1 Technical details of rank function calculations for the Bitcoin network

In this section I present some of the technical challenges and solutions regarding calculating the microscopic statistics of preferential attachment, i.e. the rank function introduced in Eq. 4.5. The main difficulty in obtaining these statistics is that \( R(k,t) \) needs to be calculated for each transaction output using the degree distribution at the time of the transaction as parameter in the calculation. In the case of the original transaction network, where we consider each Bitcoin address as a separate entity, there are over 400 million total transaction outputs, which is already a large number. Using a naive implementation (e.g. a direct calculation of the sum in Eq. 4.5), obtaining each \( R \) value would involve a summation over possibly millions of nodes active at any time in the network. As over 400 million \( R \) values need to be calculated, the total number of arithmetic operations needed would be in the range of \( 10^{12} \text{–} 10^{13} \); while not necessarily impossible on today's hardware, performing this calculation on average desktop computers or servers would require a significant runtime. Also, obtaining the set of nodes, where \( k_v < k \) holds for a given \( k \) value is not trivial; clearly, the naive implementation of checking all nodes (i.e. over 60 million) would also greatly contribute to runtime, while in a more complicated implementation, updating the degree distribution can be problematic. While a possibility would be to implement these calculations in a massively parallel way to be run on GPUs, this solution also seemed to be too complex to be worth pursuing. Instead, when carrying out the calculations presented in the previous chapters, we employed a bit more specialized solution where calculating each \( R \) value can be achieved in \( O(\log_2(N)) \) time in a network with \( N \) nodes with nonzero degrees or balances. In this section, I present this approach, where the degree sequence or the balance distribution of the network and partial sums from Eq. 4.5 are stored in a suitably adapted red-black tree; using it, not only calculating an \( R \) value but updating the degree sequence after the addition or deletion of a link can be achieved in \( O(\log_2(N)) \) time. This way, calculations for large number of transaction outputs with large number of possible nodes can be carried out efficiently, enabling us to calculate the rank function statistics for many possible exponents. In the following, I give an overview of the standard concepts of binary search trees, and their extension to red-black trees and order statistic trees. I then present how these concepts can be used to store the degree sequence and partial sums of the rank function in a way which allows calculating the sum in Eq. 4.5 efficiently.

Binary search trees  In computer science, a binary tree or binary search tree is a widely used concept of a data structure which enables one to store data elements where each element has a key value which allows ordering by it. In the simplest case, the key is
Figure A.11: Examples of binary search trees. (a) a balanced tree of nine elements; (b) an unbalanced tree with the same elements.

A numeric value, and possibly arbitrary data can be associated with keys. In the case of the degree sequence of the Bitcoin transaction network, we will use the node degrees or balances as keys and will not need to store any further data. The main purpose of a binary tree is allowing the application to find data by its key in an efficient manner, i.e. in $O(\log_2 N)$ time for $N$ elements in an ideal case. The main advantage over using binary search on a sorted array is that not only finding an element, but also inserting new elements and deleting existing elements can also be achieved with similar performance. The main disadvantage is that significantly more memory is required for storage. Here, I give an informal overview of the workings of a binary tree; a more formal introduction can be found in e.g. [56] (Chapters 12, 13 and 14).

A binary tree is defined then as follows: each data element is referred to as a node and such nodes are connected with parent-child relationships. Each node can have one parent and maximum two children, and the tree has exactly one root node, which can be reached by following parent relationships recursively staring from any node. Nodes which are reached by following only parent relationships are referred to as ancestors of a node, while nodes which can be reached by following only child relationships are the descendants of a node; all nodes are thus the descendants of the root node, which is in turn an ancestors of all nodes. The level of a node can then be defined as the number of steps it is separated from the root node. There can be at most $2^l$ nodes on the $l$th level of the tree. To enable efficient search, insertion and deletion, the following rule regarding the keys of nodes needs to be employed. For each node, its children are distinguished as left and right. Then, the key values of the left child and all of its descendants are required to be less than or equal to the parent’s key value. Similarly, the key values of the right child and all of its descendants are required to be greater than or equal to the parent’s key. Note that equality might be allowed only for one of children or possibly none in the case when the tree is to store only elements with unique keys; this decision is up to the tree implementation. In our case, multiple nodes can have the same key, as multiple nodes in the Bitcoin network can have the same degree or balance.

Using the previous rules, it is easy to see that a total of $N$ elements can be stored in a tree with $L \equiv \log_2(N + 1)$ levels (including the root node). An example of a such tree is given in Fig. A.11(a). $L$ is then referred to as the height of the tree. It can be established that finding an element in a binary tree of height $L$ requires at most $L$ iterations of basic comparisons and key lookups (i.e. by binary decisions starting from the
root node). Thus, if \( L \sim \log_2 N \), searches are efficient even for large \( N \) values. The same also applies for insertion of new elements and deletion of elements which are essentially a search followed by updating relations between nodes. While \( N \) elements can be stored in a tree with \( \log_2(N + 1) \) levels, this is just an ideal case. In practice, if the tree is created by inserting elements in an arbitrary order, there is no guarantee for the height of the resulting tree; in the worst case, a tree with \( N \) elements can have \( N \) levels, making operations on the tree quite ineffective. Trees with an ideal layout, i.e. having a height of \( L = \log_2(N + 1) \) are called balanced, while trees with \( L > \log_2(N + 1) \) are unbalanced. An example of a somewhat unbalanced binary tree is given in Fig A.11(b). Ideally, a binary tree implementation would ensure that the tree stays balanced so as to obtain ideal performance on the tree. E.g. in the simplest case, where all elements to be inserted in the tree are known in advance, building an ideal tree can be achieved easily; of course, in this case, a sorted array could also be employed which is much easier to implement, and requires less memory. In cases where the elements are not known in advance, keeping the tree balanced is not achievable in general in an effective way; on the other hand, several options exist which keep the tree close to being balanced, meaning that \( L \sim \log_2 N \) and all tree operations also still require only \( O(\log_2 N) \) complexity. These are referred to as self-balancing trees. One such possibility is a red-black tree, which is discussed next.

Red-black trees A red-black tree is a binary search tree where each node has a further property: its color, which can be either red or black. Note that this is essentially one bit of extra information per node; defining it as color, and using red and black as the two possibilities is of course arbitrary and its meaning is not understood any more by the current author than the title of the famous 19th century novel by Stendhal. Nevertheless, adding this extra property to the nodes, and adding some extra rules regarding the relations between nodes based on color will result in a tree whose height is then bounded by \( 2\log_2(N + 1) \) [90]. While insertion and deletion of nodes will become more complex, including further operations to re-balance the tree, their complexity also stays \( \sim \log_2 N \), resulting in a quite effective option for storing data elements with arbitrary distribution of keys and arbitrary sequence of insertions and deletions. Also, these bounds strictly hold even in worst-case scenarios (i.e. it can be proved that no sequence of insertions and
deletions can be given which will result in worse performance), red-black trees are a good candidate for general-purpose applications; also, this property makes red-black trees a good candidate for critical applications where having poor worst-case performance might be a security risk, which an attacker could exploit by providing a specifically crafted input. While this was not the case of our implementation for calculating rank functions for the Bitcoin network, red-black trees seemed a good option for storing the degree sequence as adequate performance could be achieved in a relatively simple way without having to worry about special cases.

In order to define the additional rules which will ensure that the tree will stay close to being balanced, it is convenient to alter the definition of a binary tree by adding empty leaf nodes at places where a node does not have children anymore. In Fig. A.12 an example for a red-black tree is given, with leaf nodes represented by rectangles (as opposed to "real" nodes, represented by ellipses). In the following figures, leaf nodes are omitted for simplicity. Using this updated definition, the conditions which a red-black tree needs to satisfy are as follows:

1. the root node and leaf nodes are always black
2. both children of a red node are black
3. every path from a node to any of its descendent leaves contains the same number of black nodes

From condition (2), it follows that in any path, red nodes cannot be present in succession, they must be separated by black nodes. Consequently, if there are \( k \) black nodes in a path, then there can be at most \( k \) red nodes in it. Condition (3) then means that for any node, the minimum and maximum length of paths to its descendant leaves can be separated by a factor of two at maximum. Applying this condition to the root node then yields that the maximum height of the tree with \( N \) elements cannot be larger than \( 2 \log_2 (N + 1) \). I.e. if there was a longer path, with length \( l > 2 \log_2 (N + 1) \), then all paths would need to be at least \( l/2 \) long according to condition (3); but given that there are only \( N \) elements, if the height of the tree was \( l \), there would not be enough elements left too "fill up" all levels lower than \( l/2 \), resulting in paths shorter than \( l/2 \) giving a contradiction.

Note that the previous three conditions do not actually require any red nodes to be present in the tree. Indeed, in the special case when \( N = 2^k - 1 \) (for any positive integer \( k \)), the three conditions can be satisfied with the very special tree where all elements are black and the tree is perfectly balanced. On the other hand, for any general \( N \), having some red nodes is required as the downmost level will not be completely filled so as to satisfy condition (3). Also, for a given sequence of elements, there will be generally many possible tree configurations satisfying the conditions. The above conditions alone do not yet provide how to construct the tree in an efficient way. Nevertheless, such options exist which make possible to not only have a limit on the height of the tree, but also carry out tree operations in logarithmic time.

The main addition which enables the tree to stay balanced, is an operation called rotation. Basically, when a rotation is performed on the tree, a node \( x \) changes position with its parent node, \( y \). If \( x \) was the left child of \( y \), then \( y \) becomes the right child of \( x \) (and vice versa), thus maintaining the basic rules for the ordering of keys for a binary search tree. The children of \( x \) and \( y \) are also reassigned to maintain these rules: if \( x \) was the left child of \( y \), then \( x \) keeps its left child, \( y \) keeps its right child, while the right child of \( x \) becomes the left child of \( y \) (or the other way around, if \( x \) was a right child). The net effect
Figure A.13: **Illustration of rotations in a binary tree.** Possible colorings are omitted for simplicity. The ordering of elements is \( b < x < c < y < a \); this is honored in both configurations. Generally, nodes \( a, b \) and \( c \) can have further descendants, while the top node (\( y \) on the left side, \( x \) on the right side) can have ancestors.

Figure A.14: **Illustration of re-coloring of a red-black tree.** The simplest case in Algorithm 3: as both the parent and the uncle of \( x \) is red, both nodes are colored black and the grandparent (\( b \)) is colored red. Iteration then needs to continue at node \( b \). Note: this is only an extract, the tree could be larger with nodes \( x, y, c \) and \( d \) having further descendants; also, node \( a \) should have further ancestors, as the root node cannot be red.

of the operation then will be that the subtree belonging to the left child of \( x \) ascends one level, while the subtree belonging to the right child of \( y \) descends one level. If the height of the subtree of \( x \)'s left child is larger than that of \( y \)'s right child, this operation will result in the total height of the whole tree being decreased, thus helping the tree get closer to being balanced. The main means of keeping a binary search tree close to being balanced are then performing rotations in an adequate way; coloring the nodes and defining the above conditions are just a means of selecting which rotations should be performed in an efficient way; according to Algorithm 3, inserting a new element in the tree will require at maximum two rotations. Rotation operations are illustrated in Fig. A.13.

Using rotations, then the steps needed to be taken after inserting a new node are those according to Algorithm 3. At first, the new node is inserted at the downmost level, similarly to insertion in a simple binary search tree, and is always colored red. Looking at the conditions for a red-black tree, after this, condition (2) can be violated, while conditions (1) and (3) will remain satisfied if the tree was originally a red-black tree (note: inserting a node into an empty tree will violate condition (1), but this can be trivially taken care of with just coloring that node black again; we can thus assume in the following that the tree was non-empty). Since the inserted node is red, the number of black nodes on any path does not change, thus condition (3) will not be altered, while if the parent is also
red, condition (2) will be violated. If this is the case, then some further steps need to be taken to restore these properties. It can then be achieved iteratively. Let $x$ denote the newly inserted node, and assume that the parent of $x$ is red. Thus there are two cases to consider. The simpler one is when the uncle if $x$ (the sibling of its parent) exists and is also red. In this case, we change the color of the parent and the uncle to black, and the grandparent to red, and continue iteratively with considering $x$’s grandparent instead of $x$. Note that with these color changes, condition (3) will stay satisfied, while condition (2) can get violated at $x$’s grandparent, requiring the iteration to be continued from there. In the special case when $x$’s grandparent is the root node, then condition (1) will get violated, but it can be again solved trivially by coloring the root black again (note that if $x$’s grandparent is not the root of the tree, then leaving it black can result in condition (3) being violated, thus we must color it red and continue the iteration). This step is illustrated in Fig. A.14. The more complicated case is when $x$’s parent is red and it’s uncle is black; now, simply changing colors would result in condition (3) getting violated. Instead, we can now proceed with performing rotations. Here, there are two sub-cases to consider: if $x$’s parent is a same-sided child as $x$ (i.e. both $x$ and its parent are left children, or both are right children), then flipping the colors of $x$’s parent and grandparent (i.e. setting the parent to black and the grandparent to red) followed by a rotation applied at $x$’s parent will result in a red-black tree satisfied all three conditions. This case is illustrated in Fig. A.15(a). Notice that since $x$ retains its parent which was changed to black, their relation will still satisfy condition (2). The parent takes the place of a black node, so condition (2) will stay satisfied there, too. The uncle stays the child of the grandparent; the grandparent is changed to red, but the uncle stays black, again keeping condition (2) satisfied. If $x$ originally had a sibling, then it must have been black (since $x$’s parent was originally red); this node now becomes the child of the grandparent too, again a relation which satisfies condition (2). Also, counting the number of black nodes in each path that leads through the nodes affected by this operation, we see that this number does not change; this means that if condition (3) was satisfied originally, then it will be satisfied also after performing the rotation. An even more complicated case is when $x$ is on the different side as its parent, e.g. $x$ is a right child, while its parent is a left child. In this case, a rotation first needs to be applied at $x$, thus creating a tree where the previous case can be applied as if we were looking at $x$’s original parent which is now it’s child. This case is illustrated in Fig. A.15(b); note that this operation again does not change if condition (3) is satisfied.

Order statistic trees A useful extension of a binary search tree is an order statistic tree, where the rank of each node (with respect to ordering by the keys) can also be calculated in an efficient way, in $O(log_2 N)$ time. This is achieved by adding an extra property to each node which stores the size of the subtree belonging to that node. This can be referred to as augmenting the tree; in general, any tree can be augmented with possibly many different kinds of extra information; of course, augmentation will be useful if it makes calculating some measure of interest possible efficiently, i.e. in $O(log_2 N)$ time, while managing this extra information upon tree operations does not worsen the complexity of tree operations, i.e. it can also be done in $O(log_2 N)$ time. This is indeed the case for an order statistic tree, where the extra information is the size of subtrees; now let $S_i$ denote the size of the subtree belonging to node $i$ (including node $i$). Note that based on the definition, it is clear that $S_i = S_j + S_k + 1$, if $j$ and $k$ are the children of node $i$ in the tree. In this case, the rank of any node can be calculated in maximum $L$ steps for a tree with height $L$ with
Figure A.15: Illustration of rotations in a red-black tree. Top: the second case in Algorithm 3: as the uncle of $x$ is black, re-coloring of the parent and grandparent ($y$ and $a$ respectively) is followed by a rotation. Note that the number of black nodes visited in any path from the ancestors of $a$ to any possible leaves does not change in the operation. The starting condition (i.e. the tree on the left) can occur e.g. if $x$ was originally black, $d$ and $e$ were red and a red node was inserted as the child of $d$ or $e$. Because of size constraints, illustration of this possibility was omitted. Bottom: The most complicated case in Algorithm 3 including two rotations: the uncle of $x$ is black, and $x$ is not on the same side as its parent (i.e. $x$ is a right child, while $y$ is left). In this case, a rotation is first applied to $x$, which results in the tree on the middle. After that, evaluation continues at $y$ (the original parent of $x$), where the top case can be applied; this then results in the final tree on the right, where condition (2) is again satisfied. Note that this operation again does not affect condition (3), the number of black nodes in any path remains the same.
Algorithm 3 Adjusting a red-black tree after inserting an element. In the beginning, $x$ should be the newly inserted node, and it should be colored red. The function $\text{Parent}(x)$ represents the parent of node $x$, while $\text{Uncle}(x)$ represents the sibling of the parent. The function $\text{Side}(x)$ yields $\text{left}$ if $x$ is the left child of its parent and $\text{right}$ if it is the right child.

\begin{algorithm}
\begin{algorithmic}
\While{$\text{Parent}(x)$ is red}
\If{$\text{Uncle}(x)$ exists and is red}
  \State color $\text{Parent}(x)$ and $\text{Uncle}(x)$ black
  \State color $\text{Parent}(\text{Parent}(x))$ red
  \State $x$ ← $\text{Parent}(\text{Parent}(x))$
\Else
  \If{$\text{Side}(x) = \text{Side}(\text{Parent}(x))$}
    \State color $\text{Parent}(x)$ black
    \State color $\text{Parent}(\text{Parent}(x))$ red
    \State Rotate($\text{Parent}(x)$)
    \State break
  \Else
    \State $y$ ← $\text{Parent}(x)$
    \State Rotate($x$)
    \State $x$ ← $y$
  \EndIf
\EndIf
\EndWhile
\State color root black
\end{algorithmic}
\end{algorithm}

Algorithm 4. The calculation makes use of the fact that the key of $i$ is always greater than or equal to the key of its left sibling and its descendants if such nodes exist. Consequently, the size of the subtree starting from a left sibling node can just be added to the sum which is being evaluated without needing to descend into that subtree. Then, left siblings of ancestors of $i$ also need to be considered, as those and their descendants also will have key values less than $i$. In a tree with height $L$, this calculation will thus have at most $L$ steps (if $i$ is at the downmost level). In a balanced tree with $N$ nodes, this calculation then takes at most $\log_2(N+1)$ steps. If the tree is not balanced, but $L \sim \log_2 N$ as in the case of a red-black tree, the rank can still be calculated in $O(\log_2 N)$ steps. An example of an order statistic tree and an illustration of computing the rank of a node is given in Fig. A.16. More formal introduction to the topic can be found in e.g. [56].

Note that in the case when the tree can contain multiple elements with the same key, the rank calculated here will possibly include nodes with the same key. To calculate the number of nodes which have strictly smaller keys, Algorithm 4 needs to be started from the leftmost node with the given key, which can be defined as follows: for a given key value $k$, a node $i$ is a leftmost node if (1) the key of node $i$ is $k$; (2) if $i$ is the right child of its parent, then the key of the parent is strictly smaller than $k$; (3) if $i$ has a left child, then the key of the left child is strictly smaller than $k$. Essentially, such a node can be found with standard binary search with the additional condition that if a node with the desired key is found, then its left child also needs to be checked and the search there recursively continued until the left child has strictly smaller key value than the required key or does not exist.

To be able to use an order statistic tree, the $S_i$ values need to be updated when
Figure A.16: Illustration of calculating the rank of a node in an order statistic tree. Partial sums (i.e. $S$ values associated with the nodes) are displayed in parentheses. In the example given here, the purple node's rank need to be calculated; the process starts there with setting $R = 1$, then proceeds following the colored arrows. For green nodes, $R$ needs to be increased by one; for blue nodes, their $S$ values need to be added; for red nodes, nothing needs to be done. Calculation terminates after the root node and its left child has been both visited. The result is then $R = 12$, whose correctness can easily be checked for the example tree given here. Notice that the complexity of the calculation only depends on the number of steps needed to reach the root from the starting node; adding further descendants to the tree would not increase the steps needed for the calculation.
Algorithm 4 Basic algorithm to calculate the rank of node $i$ in an order statistic tree. Here, $\text{Left}(x)$ and $\text{Right}(x)$ represent the left and right children of node $x$ respectively, while $\text{Parent}(x)$ represents the parent of $x$. The rank of node $i$ is calculated in $R$.

Algorithm 4: Basic algorithm to calculate the rank of node $i$ in an order statistic tree.

```
x ← i
R ← 1
if $\text{Left}(x)$ exists then
  y ← $\text{Left}(x)$
  R ← R + $S_y$
end if
while $x \neq \text{root}$ do
  z ← $\text{Parent}(x)$
  if $x = \text{Right}(z)$ then
    if $\text{Left}(z)$ exists then
      y ← $\text{Left}(z)$
      R ← R + $S_y$
    end if
    R ← R + 1
  end if
  x ← z
end while
```

inserting and deleting nodes. This is also an operation which can be carried out in $L$ steps, i.e. in $O(\log_2 N)$ time for a red-black tree. In the case of a change, updating the sum associated with a node is a trivial step based on the values of its children if they exist. Having done that, this change then needs to be propagated upwards, until it reaches the root node. Since there is no need to descend into subtrees of sibling nodes, this update propagation also takes at maximum $L$ steps.

Order statistic trees with arbitrary rank functions  In order to calculate the values of the $R(k,t)$ rank function introduced in Eq. 4.5, the previous order statistic tree definition can be adapted. Instead of ranks, an order statistic tree can be generalized to calculate the sum of any function of the keys easily. Let $f(k)$ denote this function; for the calculations with the preferential attachment model, we will use $f(k) \equiv k^\alpha$ in accordance with Eq. 4.4. We then set the $S_i$ measure associated with a node as following: $S_i \equiv f(k_i)$ if $i$ has no children; $S_i \equiv f(k_i) + S_j$ if $j$ is the only child of $i$, and $S_i \equiv f(k_i) + S_j + S_k$ if $j$ and $k$ are the two children of $i$. Note that this is a natural generalization of storing subtree sizes as the $S_i$ values in the case of an ordinary order statistic tree; indeed, if we set $f(k) \equiv 1$, we get back the order statistic tree definition in the previous section. Now the rank function of node $i$ can be calculated in a similar way as the rank in an order statistic tree as outlined in Algorithm 5, which is essentially the same as in the previous case; the main difference is that $f(k_i)$ is added to the sum instead of just 1 (and of course the $S_i$ values have different meaning here). Note that $R$ is given 0 as an initial value (instead of 1 in the case of an order statistic tree, which would correspond to $f(k_i)$ here). We do this so as to be compatible with the definition of $R(k,t)$ in Eq. 4.5. The $R_i$ value calculated

---

We note, that generally, $f$ need not be a function of the keys used in the tree; generally, each node in the tree can have additional data associated with it, and $f$ can take this data as parameters; nevertheless, the methods of calculation presented here will result the partial sum of $f$ values corresponding to nodes with key values smaller than the given $k$. 

127
here then corresponds to the numerator in Eq. 4.5, while the denominator is simply the $S_r$ value of the root node. Updating the $S_i$ values after insertion or deletion is done in the same manner as for an order statistic tree.

Algorithm 5 Algorithm to calculate a rank function for node $i$ (i.e. the sum of function $f(k)$ for nodes with smaller key values than node $i$) in a modified order statistic tree. Here, the definition is slightly altered so that the result will not include $f(k_i)$ to be compatible with the rank function defined in Eq. 4.5.

\[
\begin{align*}
x &\leftarrow i \\
R &\leftarrow 0 \\
&\text{if Left}(x) \text{ exists then} \\
&\quad y &\leftarrow \text{Left}(x) \\
&\quad R &\leftarrow R + S_y \\
&\text{end if} \\
&\text{while } x \neq \text{root do} \\
&\quad z &\leftarrow \text{Parent}(x) \\
&\quad &\text{if } x = \text{Right}(z) \text{ then} \\
&\quad &\quad &\text{if Left}(z) \text{ exists then} \\
&\quad &\quad &\quad y &\leftarrow \text{Left}(z) \\
&\quad &\quad &\quad R &\leftarrow R + S_y \\
&\quad &\quad &\text{end if} \\
&\quad &\quad R &\leftarrow R + f(k_z) \\
&\quad &\text{end if} \\
&\quad x &\leftarrow z \\
&\text{end while}
\end{align*}
\]

Again, in the case where multiple nodes can have the same key value, we can limit the algorithm to only include nodes with key values strictly less than a given key by starting the calculation from the leftmost node with the key in question. When defining Eq. 4.5, we chose to carry out the summation for nodes with strictly smaller degrees than the node in question.

Using this adapted order statistic tree, our solution to calculate rank functions can then be summarized in the following steps:

1. store the degree sequence or balance distribution (i.e. the values of degrees or balances of all nodes) in a red-black tree augmented with the $S_i$ partial sums

2. when the degree or balance of any node changes (either due to addition or deletion of a link in the case of degrees or due to sending or receiving bitcoins in the case of balances) find any node in the tree with a key corresponding to it, delete it and add a new node with a key corresponding to the updated value; also update the $S_i$ values while performing tree operations so that the relations defining them stay satisfied

3. each time $R(k,t)$ needs to be calculated, calculate the nominator by Algorithm 5 started from the leftmost tree node with key $k$, and take $S_r$ as the denominator

To implement this procedure, we used an existing open-source red-black tree implemen-
tation\textsuperscript{8}, and modified it to also keep track of partial sums, and to calculate rank functions according to Algorithm 5. This modified red-black tree implementation is available at http://www.vo.elte.hu/bitcoin.

A.2 Data processing for the Gowalla network

The original Gowalla data set consists of a social network (where nodes represent users and links symbolize friendships between them) and check-in data with approximately 6.5 million records of the form: (userID, timestamp, lat, lon). As most of the users performed check-ins from various locations, we need to designate a single base-location for each user, to enable the spatial embedding of the social network. We have chosen to reject the spatial embedding of users who does not seem to have a characteristic position over the check-ins.

To perform the embedding, we “pixelate” check-in locations of a given user with a hierarchical spherical indexing technique, the Hierarchical Triangular Mesh (HTM) \cite{HTM}. At a given resolution level HTM divides the sphere into triangles, referred to as trixels. We use base indices at depth 20, which corresponds to trixels of area \( \approx 100 \text{ m}^2 \) or \( \approx 1000 \text{ ft}^2 \). Next, for each user \( u \) we iteratively coarse-grain the trixel-resolution (by decreasing the index depth) until there emerges a single trixel containing at least 50\% of \( u \)'s check-ins. We calculate the \( \mu_u \) mean position and \( \sigma_u \) standard deviation within this cell, and accept \( \mu_u \) as the position estimate for \( u \) if \( \sigma_u < 10 \) km, otherwise we reject the spatial embedding of \( u \). After processing the network with this procedure there remains 94,798 nodes and 289,961 links, for which both endpoints have a well-defined location estimates.

---

\textsuperscript{8}available under a BSD-style license at http://web.mit.edu/~emin/www.old/source_code/red_black_tree/index.html
ADATLAP
a doktori értekezés nyilvánosságra hozatalához

I. A doktori értekezés adatai
A szerző neve: Kondor Dániel
MTMT-azonosító: 10044936
A doktori értekezés címe és alcímé: Empirical analysis of complex social and financial networks
DOI-azonosító:10.15476/ELTE.2015.083
A doktori iskola neve: Fizika Doktori Iskola
A doktori iskolán belüli doktori program neve: Statisztikus fizika, biológiai fizika és kvantumrendszerek fizikája program
A témavezető neve és tudományos fokozata: Vattay Gábor, DSc
A témavezető munkahelye: ELTE Komplex Rendszerek Fizikája Tanszék

II. Nyilatkozatok
A doktori értekezés szerzőjeként

a) hozzájárulok, hogy a doktori fokozat megszerzését követően a doktori értekezésem és a tézisek nyilvánosságra kerüljenek az ELTE Digitális Intézményi Tudástárán.
Felhatalmazom a Természettudományi Kar Tudományszervezési és Egyetemközi Kapcsolatok Osztályának ügyintézőjét.............................................., hogy az értekezést és a téziseket feltöltse az ELTE Digitális Intézményi Tudástárba, és ennek során kitöltse a feltöltséhez szükséges nyilatkozatokat.
b) kérem, hogy a mellékelt kérelemben részletezett szabadalmi, illetőleg oltalmi bejelentés közzétételéig a doktori értekezést ne bocsássák nyilvánosságra az Egyetemi Könyvtárban és az ELTE Digitális Intézményi Tudástárban; 41

c) kérem, hogy a nemzetbiztonsági okból minősített adatait tartalmazó doktori értekezést a minősítés (datum) ig tartó időtartama alatt ne bocsássák nyilvánosságra az Egyetemi Könyvtárban és az ELTE Digitális Intézményi Tudástárban; 42

d) kérem, hogy a mű kiadására vonatkozó mellékelt kiadó szerződésre tekintettel a doktori értekezést a könyv megjelenésénél ne bocsássák nyilvánosságra az Egyetemi Könyvtárban, és az ELTE Digitális Intézményi Tudástárban csak a könyv bibliográfiai adatait tegyék közzé. Ha a könyv a fokozatserzést követően egy évig nem jelenik meg, hozzájárulok, hogy a doktori értekezésen és a tézisek nyilvánosságra kerüljenek az Egyetemi Könyvtárban és az ELTE Digitális Intézményi Tudástárban. 43

2. A doktori értekezés szerzőjeként kijelentem, hogy

a) az ELTE Digitális Intézményi Tudástárba feltöltendő doktori értekezés és atézisek sajátérendeti, önálló szellemi munkámés legjobb tudomásom szerint nem sértem vele senki szerzői jogait;
b) a doktori értekezés és a tézisek nyomtatott változatai és az elektronikus adathordozón benyújtott tartalmak (szöveg és ábrák) mindenben megegyeznek.

3. A doktori értekezés szerzőjeként hozzájárulok a doktori értekezés és a tézisek szövegének plágiumkereső adatbázisba helyezéséhez és plágiumellenőrző vizsgálatok lefuttatásához.

Kelt: Budapest, 2015.05.18.

38 Beiktatta az Egyetemi Doktori Szabályzat módosításáról szóló CXXXIX/2014. (VI. 30.) Szen. sz. határozat.
39 A kari hivatal ügyintézője tölti ki.
40 A megfelelő szöveg aláhúzandó.
41 A doktori értekezés benyújtásával egyidejűleg be kell adni a tudományági doktori tanácnokhoz a szabadalmi, illetőleg oltalmi bejelentést tanúsító okiratot és a nyilvánosságra hozatal elhalasztása iránti kérelmet.
42 A doktori értekezés benyújtásával egyidejűleg be kell nyújtani a minősített adatravonatot közokiratot.
43 A doktori értekezés benyújtásával egyidejűleg be kell nyújtani a mű kiadásáról szóló kiadói szerződést.