Original Article

Methodological Innovations
Volume 9: 1–11

DOI: 10.1177/2059799115622763
mio.sagepub.com
SAGE

Installing computational social science: Facing the challenges of new information and communication technologies in social science

Raphael H. Heiberger[1] and Jan R. Riebling[2]

Abstract

Today's world allows people to connect over larger distances and in shorter intervals than ever before, widely monitored by massive online data sources. Ongoing worldwide computerization has led to completely new opportunities for social scientists to conceive human interactions and relations in unknown precision and quantities. However, the large data sets require techniques that are more likely to be found in computer and natural sciences than in the established fields of social relations. In order to facilitate the participation of social scientists in an emerging interdisciplinary research branch of "computational social science," we propose in this article the usage of the Python programming language. First, we carve out its capacity to handle "Big Data" in suitable formats. Second, we introduce programming libraries to analyze large networks and big text corpora, conduct simulations, and compare their performance to their counterparts in the R environment. Furthermore, we highlight practical tools implemented in Python for operational tasks like preparing presentations. Finally, we discuss how the process of writing code may help to exemplify theoretical concepts and could lead to empirical applications that gain a better understanding of the social processes initiated by the truly global connections of the Internet era.

Keywords

Computational social science, Python, R, data management, research practices, theory development

Introduction

In recent years, the ability of people to connect across the globe has dramatically increased due to technological innovations related to the Internet. Ongoing worldwide computerization provides great opportunities for scientists to enhance our understanding of the complex systems in which humans live today. Increasing availability of massive social media data abets efforts trying to explain collective behavior with methods stemming from the natural and computational sciences (Buldyrev et al., 2010; Goel et al., 2010; Golder and Macy, 2011; Heiberger, 2014, 2015; King, 2011; Lazer et al., 2009; May et al., 2008; Schweitzer et al., 2009).

Rising computer power enables and enhances this development. The capacity to collect and analyze large amounts of data with computational techniques has already transformed physics (quantum computing; Steane, 1998) and biology (computational biology; Kitano, 2002). However, in the most influential journals of the social sciences, the potential of Internet data and computational methods is only starting to take off, although large companies are worth hundreds of billions of dollars mainly because of their understanding to utilize "big (social) data" (e.g. Google, Facebook, Twitter) and state institutions are using similar techniques for the better and worse of their citizens (most prominently the National Security Agency [NSA]). Thus, "computational social science" (CSS) is occurring. The question is whether it happens with or without social scientists.

[1]Institute for Sociology, University of Bremen, Bremen, Germany
[2]Otto-Friedrich-Universität Bamberg, Bamberg, Germany

Corresponding author:
Raphael H. Heiberger, Institute for Sociology, University of Bremen, Mary-Somerville-Str. 9, 28359 Bremen, Germany.
Email: raphael.heiberger@uni-bremen.de

To be sure, there have been recently some prominent publications that, for instance, are using computational text analysis (Bail, 2012), Ebay auction data (Diekmann et al., 2014), the Internet movie database (Lutter, 2015), Facebook friendship networks (Wimmer and Lewis, 2010), or even career histories of video game developers and their creative products (De Vaan et al., 2015). Yet these efforts, valuable as they may be, still seem very few when compared to the overall research endeavor of the Social Sciences. In an age of increasing technological means and challenges, this relative reluctance seems ill-advised.

Given our profession as trained sociologists, we clearly opt for a more active participation of our discipline in this new research paradigm. This article aims to accomplish this in two ways: on one hand, we want to discuss the challenges as well as the potential CSS poses for research efforts in the field of Social Science. Since CSS has sometimes been relegated to either Big Data applications or some very specific methods (mostly Social Network Analysis [SNA]), we want to instead emphasize the broad influence and transformative power CSS will have on the Social Sciences in general. On the practical side, we illustrate the usage of computational methods and techniques in the Python language, which serves as a generalized programming framework to conduct all steps of a "CSS analysis." Being far from the only programming language to cater to the needs of CSS researchers, Python offers some distinct advantages. Most prominent is its focus on the readability and accessibility of code, which is especially important to social scientists, who in general have received little to no training in programming and coding. Python is also easily extendable with a wide variety of libraries for scientific computing. Since most of these extensions are maintained by professional programmers and a very active community, they are exceptionally well documented and help is never far away.

For this purpose, we first discuss the position of social scientists in the young but growing research area of CSS and the challenges that may have led to the relative absence. Next, we illustrate the capabilities of CSS with regard to four major areas of the social scientific research process. The first is the capacity to manage, gain access, and analyze a wide variety of data structures, which is one of the core properties of CSS. Second, we take a look at some innovative methods which are normally associated with CSS, more specifically SNA and quantitative text analysis. These two steps—handling data and applying methods with adequate speed—are strongly related to general problems of CSS (Lazer et al., 2009). From these basic virtues, we elaborate on further opportunities of CSS in general and Python specifically. Thus, third, we highlight the practical tools implemented in Python, for example, techniques to prepare presentations or lectures. Finally, we discuss some ideas about theory building, especially how the process of writing code can help to exemplify theoretical concepts. Altogether, we want to picture the general usefulness of CSS for the "whole" research process in Social Sciences, especially with regard to the challenges and opportunities provided by the Internet as a driving force behind the revolution of data describing and reflecting the social world.

Social scientists and CSS

The claim of CSS was first made prominently by Lazer et al. (2009). They summarized the possibilities (and problems) that arise with the availability of massive social data produced by the usage of the Internet.[1] Most techniques necessary to analyze these kinds of data are adapted from computer science since social scientists are usually not trained to handle "Big Data." However, the principal questions arising from growing global interconnectedness are very much social. For instance, who is related to whom and who has the most influential positions (Lewis et al., 2008)? What impact has the structure of particular societies on these relations (Onnela et al., 2007)? What are general properties of human communications (Karsai et al., 2011)? Thus, "computer-science students had much better methodological chops than their sociology counterparts, but the sociologists had much more interesting questions," as renowned physicist Duncan Watts puts it (cited in Giles, 2012: 450).

The diagnosis of advantageous methodical knowledge seems to be true not only for computer scientists as "natural inhabitants" of the new world of digital connections, posts, and tweets but also for physicists who are highly skilled in the analysis of large amounts of data and complex systems. Consequently, there exists a growing movement within physics that tries to take advantage of the increasingly growing amount of online data. In stark contrast to the Social Sciences, there is an atmosphere of departure that enables them to write a "manifesto" (Conte et al., 2012) and to make a strong claim on the analysis of large social (online) data. They propose a computer-based approach that could lead to predictive and explanatory models that utilize the large data sets on every level of behavior (i.e. individuals and their various aggregations). A member of this group also published the first textbook on CSS (Cioffi-Revilla, 2014). He also identifies all Social Sciences as potential application areas and, very wisely, does not narrow CSS to certain methods like SNA nor to certain type or scale of data.

However, it is common sense among these authors that Information and Communication Technologies (ICT) play a key role in CSS since the possibilities provided by connected objects like computers, smartphones, cars, and even houses (as well as many more to come) have produced a worldwide system that rivals our planet's ecology in its sheer complexity. The ICT are transforming (already complex) social systems, most visibly the economy and the financial markets, but also the way we communicate and live in general. These structures can be thought of as "Artificial Social Systems" (Helbing,

2012: 6). They are man-made, yet at the same time give rise to a far greater complexity than intended and understood by their original creators. Consequently, many current societal problems can be traced back to a lack of expertise in dealing with these new social phenomena, including but not limited to economic instabilities, environmental change, and cyber warfare.

To mitigate present and future challenges regarding the ICT, physicists already created the project initiative "FutureICT" (www.futurict.eu/), funded by the European Union (EU) as a "Flagship Pilot Project," which participated in the race for the EU's US$1.3 billion provided "to move the ICT frontier." However, for social scientists, their own absence should be more noteworthy than the overall funding scheme. The project's Principal Investigators (PIs) are physicists, and the list of disciplines involved in the "best of all relevant knowledge" does not even include sociology or political sciences.

This underlines the fact that in the Social Sciences, efforts to break into CSS seem rather modest. To be sure, there have been studies conducted by social scientists that make use of the new possibilities, especially in terms of networks (Lehmann et al., 2015; Lewis et al., 2008; Qin et al., 2015; Watts, 2004). However, compared to the publication and collaboration efforts in the computer and natural sciences, the utilization of computational methods and large social data sets of social scientists is rather expandable.

To be one of the disciplines participating in, by definition, interdisciplinary CSS and not being marginalized from the beginning, social scientists have to rethink their tools and "computerize" their methods and theories. This is reminiscent of the situation in the SNA community around the millennium. SNA tries to explore empirical social interactions with elaborated mathematical modeling (Scott and Carrington, 2011; Wassermann and Faust, 1994)—and is somehow ignored by physicists (Freeman, 2011). For a long time, there existed many approaches in the Social Sciences for the explanation and analysis of cohesive groups with regard to social networks (Freeman and Webster, 1994; Homans, 2003). Physicists widely ignored such efforts until their colleagues Girvan and Newman (2002) developed an algorithm to detect those communities. The missing mutual recognition is also evident in regard to the "Small World" structure (Milgram, 1967; Watts and Strogatz, 1998). This formal approach to describe certain properties of networks resulted in more papers and citations by natural scientists in a few years than Social Science produced in 45 years (Freeman, 2011). Knowing this story, one feels unsurprised that the CSS manifesto mentioned above proposes "traditional tools of social sciences would at most scratch the surface of these issues [of ICT related phenomena], whereas new tools can shed light onto social behaviour from totally different angles" (Conte et al., 2012: 327).

To support the development of new tools that are undoubtedly needed for the interconnected world of the 21st century, we propose a technical instrument that allows the integration of all necessary steps for the analysis: the Python language. As a kind of minimum common sense in the existing literature (Cioffi-Revilla, 2014; Conte et al., 2012; Lazer et al., 2009), CSS approaches focus the utilization of the ever-growing computer power and large data sets stemming from the Internet. We see at least two substantial (and intertwined) barriers: first, to access terabytes of data is not part of standard statistical programs used in Social Sciences. The trace left by the description of real-time interactions of entire populations of humans is far greater than the largest longitudinal social surveys are handling today. Second, existing methods of analysis and theories are based on those surveys (and other rather coarse observations) to conceive social behavior and relations. In network theory, for instance, most theories and methods are built on groups of maybe hundreds or, less often, thousands of people, but are most definitely not adapted to analyze data sets of millions of people, including additional information about their locations, preferences, or other attributes. One first step for social scientists to develop such tools and, hence, to participate in the application of computer-aided methods is a flexible programming language. In our view, it seems preferable to choose a holistic language for this purpose and to use the same language in all steps of analysis, from data mining and transformation to applying methods in practice. In the next sections, we are going to elaborate on each of these steps and show the benefits of using a general programming language.

New types of data

The digital revolution has greatly transformed one of the central domains of Social Science, our data. We concur with Savage and Burrows (2007) that the rise of new types of data can be considered a fundamental "crisis of empirical sociology." This follows from the fact that the careful acquisition and quality of empirical data can be considered one of the cornerstones of the Social Sciences claim to actually be part of the sciences and not just social commentary. Among the many challenges and problems stemming from the rise of new data sources, two of them stand out. First, there is the sheer amount of data generated by digital transactions, which is sometimes labeled as "Big Data." Second, the emergence of new data sources with the promise of solving central questions of the Social Sciences, which are not too big, but are otherwise not suitable for analysis right away. The problem here stems from the fact that much of these data are provided and structured in ways unfamiliar to most social scientists. We call this "Medium Data."

Big Data has become something of a buzz word in recent years which seems to have finally reached the Social Sciences although it has not been a very warm welcome (Burrows and Savage, 2014; Ruths and Pfeffer, 2014; Uprichard, 2013). Generally, the term refers to data which "exceeds the processing capacity of conventional database systems" (Dumbill, 2012). Data of this magnitude are produced either as

user-generated data in social media and networks (e.g. email, video-streaming, online discussion groups) or as process-generated data (e.g. administrative processes, server logs, machine transactions). To get grasp of what "Big" means in this context, we can take a look at some of the leading companies in this sector. According to Kerzner and Maniyam (2013), these are some of the estimates for active data storage:

- Facebook: 40 petabyte of data storage/100 terrabyte captured a day;
- Yahoo: 60 pb a day;
- Twitter: 8 tb a day.

These figures pale in comparison with the very rough estimates on Google's active storage, which based on the power consumption of Google's data centers is somewhere in the range of 10 exabytes per day.[2]

The problem Big Data poses for the Social Sciences is twofold. On one hand, it is a technical problem; on the other hand, it is mostly a problem on the conceptual level. From a technical perspective, Big Data requires expertise in handling large hierarchical data structures (e.g. HDFS, HBase, S3) and using parallel programming techniques (i.e. MapReduce) to analyze these data. While the statistical models are mostly the same as in standard Social Science methodology, it is the overall framework in which these models are implemented with regard to software and hardware specification that is challenging social scientists. Most of the Big Data applications are oriented around specific programming techniques, like functional programming or similar models, which are easy to grasp for programmers but at the same time are barely known among social scientists.

In our opinion, this limited understanding of the technical aspect sometimes spills over into the Social Sciences perception of Big Data. In general, Big Data is considered to be a threat to the methodology and the empirical claims of Social Science. Instead of incorporating it into the research methodologies, its conceptual and ethical problems are discussed (e.g. Bail, 2014; Uprichard, 2013). Although there are plenty of reasons to be concerned regarding Big Data, without much practical knowledge about its implementation Social Sciences stance on Big Data is in danger of sounding uninformed, whereas other disciplines are already making use of the new possibilities.

Whether we like it or not, Big Data is here to stay. To participate in its ongoing development, social scientists will need to develop the technical expertise to at least understand the basic concepts. Beyond that, there is and will be a growing demand for professionals inside and outside of academia able to actually carry out Big Data analysis. This is especially true with regard to CSS.

Python seems to be a good choice in helping our discipline in getting to grasps with Big Data. There are many packages for interacting with large data sets as well as algorithms designed for parallel computing.[3] In addition to the on-board functionality of Python distributions geared toward scientific computing—most notably the IPython Anaconda distribution—there are also specialized Big Data frameworks (e.g. Hadoop, Sparks) which provide interfaces for high-level languages like R or Python. Besides making the technical entry hurdles much less intimidating, Python provides an easy access to the basics of computer science. It is easily accessed and focuses on readable and easy to maintain code. In short, one of the easiest ways to think like a computer scientist is to "Think Python" (Downey, 2012).

Chances are that the majority of social scientists are never going to work with actual Big Data. The tricky part here is the word "actual." Since Big Data is usually defined as "being too big for conventional means," the problematic nature of data has to be considered with regard to the methods and techniques common in a discipline.[4] Much of the new data structures encountered in the Social Sciences would not be seen as Big Data by computer scientists because they are familiar with conventional means to tackle these kinds of structures. We will refer to such data that are too big and unfamiliar for social scientist, yet not big enough to warrant Big Data analytics as "Medium Data."

So what are those conventional means defining the Social Sciences in terms of data structures? Ask any social scientist to visualize data; chances are they will picture a rectangular table consisting of observations along the rows and variables as columns. Since this is what we are taught in every undergraduate course on methods, this is what data produced by prestigious surveys looks like, and last but not least, this is what most of the computer programs we use expect as input. This tendency to equate data with tables has problematic consequences for research practices and for the ability to access new and unorthodox data structures.

If every row corresponds to one observation and most variables have at least a chance to be present, the data frame is one of the best solutions. Yet, if the sparsity of the data is very high, meaning many cells of the data frame will be left empty, the table structure becomes very cumbersome. This is almost always the case when we look at social network or textual data. For example, the number of nodes occurring in a network is mostly orders of magnitudes greater than the number of edges one node is connected to. Thus, a network consisting of 1000 nodes and 15,000 edges would result in a table of 1000 rows and 1000 columns (1 million cells) while representing the same data as a list of edges would yield a data structure with a magnitude of only 15,000 units.

Network data are also a good example for the general problem of complex, that is, hierarchical data. Consider a network made up of social relationships among a group of people. Besides the edges, there can be many different levels of data attached to this structure. There could be data on the intensity of the relationships or data concerning the persons (e.g. age, sex) as well as meta-data about the entire network (i.e. date of collection, context, etc.). Adding this information to a table would result in many duplicates across lines and if done extensively enough would make the whole

Table 1. Performance of packages in Python and R regarding social network analysis.

Measure	Python *graph-tool*	Python *NetworkX*	R *igraph*
Single-source shortest path	0.0063 s	0.127 s	0.012 s
PageRank	0.555 s	34.26 s	0.781 s
K-core	0.0250 s	0.9586 s	0.0181 s
Minimum spanning tree	0.0296 s	0.413 s	0.0397 s
Betweenness	1977.6 s (~33 min)	53,716.692 s (~14.9 h)	1182.6 s (~19.7 min)

thing indecipherable. The main advantage of using hierarchical data structures (e.g. XML) or relational databases (SQL) over simple tables lies in their ability to manage entire projects within one well-defined data framework. From this data repository, data frames can be easily extracted for further analysis. Instead of focusing on one data structure, CSS emphasizes techniques of converting data between different data structures according to the needs of the overall research goals.

Knowing how to manage hierarchical data is even more important if we consider the new data generated by the ICT. Most of the social data available online already exist in a hierarchical standardized form. Webpages, for example, are written in HTML, which is simply a hierarchical ordering of elements. One of the most promising new data sources comes in form of WebAPIs (Web-based Application Programming Interfaces). In this model, a request for specific data points is sent to a server, often mitigated through a wrapper in a specific language, for example, Python. Barring any connection problems, the server responds by sending the requested data to the client. Most of these data come in a hierarchical structured form, mostly XML or JSON. There are many interesting WebAPIs for Social Science research.[5] Most of the big online social networks like Facebook and Twitter provide at least limited access. Data repositories are also starting to implement WebAPIs, for example, the World Bank Indicators, Yahoo! Finance, Google Analytics, which can be directly called from Python's Pandas package (together with some other open data sources).

Innovative methods

The types of data delivered by ICT are mostly interconnected by the design of the Internet as a network of billions of nodes. Network analysis is therefore a direct candidate for understanding the structure of relations between diverse actors around the globe. Furthermore, the Internet produces large amounts of text in blogs, posts, messages, or news. They are an essential part of today's communications, regardless of whether one-to-one (e.g. Skype, Facebook messaging), one-to-many (blogs, comments), many-to-one (Twitter accounts of prominent people), or many-to-many (group conversations, forums). The potential of classic qualitative approaches to understand these texts through "manual" reading is clearly limited because they do not scale very well. A more quantitative understanding as provided by "natural language processing" is needed.

For both methods, convenient solutions are already implemented in Python in terms of the graph-tool (Peixoto, 2015) or NetworkX (Hagberg et al., 2008) as well as the Natural Language Toolkit (NLTK) (Bird et al., 2009). As discussed in the section "New types of data," the volume of Big Data sets makes fast solutions more necessary than ever. To support our arguments, we will test the performance of the Python applications in comparison with the R environment and its respective solutions since R is often seen as standard solution for statistical analysis in Social Sciences (McCullough, 2010).[6]

To demonstrate the performance of Python in regard to networks, we rely on a well-documented test that is conducted by the author of the graph-tool package (Table 1).[7] The network in question is a directed graph, with 39.796 nodes and 301.498 edges. Only calculations with one core are considered (i.e. no multi-threading). As is expected, betweenness centrality as a computational complex measure takes most time (Brandes, 2001). The performance differences between packages are striking, but are not primarily occurring between Python and R, but if the respective packages are assembled in C++. Both graph-tool and igraph use C++ and their performance is rather alike, with slight advances for graph-tool except for the calculation of betweenness centrality. NetworkX, in contrast, is a pure Python implementation and is by far the slowest of the packages. Thus, it is not the language alone that decides the performance, but the package implementation.

With regard to text analysis, until now no comparison between Python and R exists. The most elaborated solution for text analysis in R is included in the tm package (Feinerer et al., 2008). We test the performance by using the "Brown corpus" created in 1961 at Brown University.[8] It was the first electronic corpus in English with more than 1 million words (N = 1,161,192) and is widely used in quantitative linguistics for "Part-of-speech tagging." The comparison has three steps and starts with the same "corpus" consisting only of sequences of tokens (i.e. no sentence or document structure): first, we run a basic cleaning procedure by removing a list of "stop words" (174 words used most often in the English language) and write all words in lower cases to avoid

Table 2. Performance of packages in Python and R regarding text analysis.

Procedure	Python *NLTK*	R *tm*
Cleaning (i.e. lower cases, remove stop words)	2.627 s	799.2216 s (~13 min)
Frequency distribution	0.4869 s	739.971 s (~12 min)
LSA	3.1729 s	5.8021 s

NLTK: Natural Language Toolkit; LSA: Latent Semantic Analysis.

ambiguities. Second, we calculate the word frequency of all words in the "cleaned" corpus. Third, the incorporation of a more elaborated detection of topics within the corpus is conducted using a "Latent Semantic Analysis" (Deerwester et al., 1990), which is implemented in the "genism" package in Python (Rehurek and Sojka, 2010) and the "lsa" package in R (Wild, 2015). For this purpose, all words of each category are treated as one document, and the number of categories (15 in total) is used as the predefined number of topics that should be retrieved.

The results in Table 2 show that Python and the NLTK package largely outperform R. This is true for the simple cleaning procedures as well as for the calculation of the word frequencies, indicating that R has problems with "strings" and, consequently, manipulating and/or counting text elements since it is designed for statistical analysis using vector matrices. The NLTK package, in contrast, provides a very fast and convenient way to analyze large amounts of text. As we have already seen with the analysis of networks, the performance depends very much on the package in use. Striking as the differences between Python and R in terms of simple word transformations may be, the performance of the LSA package is much closer to its Python counterpart. Still, the time R needs for the transformation of the "bag-of-words" in a semantic space is nearly twice of Python.

The merit of this exercise lies not only in comparing different tools and packages but also in showing the differences across the ever-growing number of methods and toolset. Besides contributing heavily to the development of new methods and software solutions, CSS also offers the expertise to decide which tool fits the job. Knowing how to test and compare different implementations of the same method becomes increasingly important. Without some familiarity with algorithms and a basic knowledge of at least one programming language, it is difficult to adequately judge or increase the performance of software. The digital literacy provided by CSS can be seen as one of the greatest merits this emerging field has to offer. Yet, this contribution of CSS can only be properly utilized if it becomes part of the professional training of social scientists. A task for which Python seems to be especially well suited because of the easy access it provides to basic programming techniques, as exemplified by Allen Downey's (2012) aptly titled book *Think Python. How to Think Like a Computer Scientist*.

Doing sociology in the 21st century

It is not only the data sources and methods of Social Sciences that are transformed by ICT. The same can be said about the day-to-day practices, meaning the way we do research, teach, review literature, write articles, and so on. While we do devote conferences, articles, and books to abstract models and research techniques, there is not much discussion about the actual way scientists engage in their daily work life. Some disciplines may have manuals and protocols for interacting with dangerous or valuable equipment, but most professional expertise of scientists can be considered tacit knowledge, which is mostly acquired on the job.

Since most of our work is nowadays in some way connected or mitigated through digital media, CSS offers a chance to make research more effective. Most of the practical work of scientists involves processing information in one way or the other: sending emails, making presentations, formatting articles, preparing courses, just to name a few. Most of these tasks are necessary byproducts or pre-conditions for doing actual research.

Observing our own work flows and those of our colleagues, we found that there are two main problems that seem to be responsible for a significant loss of time: first, doing a machine's work the way a machine would do it; second, using a patchwork of different, specialized programs most of which are proprietary, thereby making it almost impossible to know their exact inner workings or definitions and resulting in an inflexible workflow.[9]

The overall usefulness of machines lies in the fact that they do exactly what they are told in a repetitive fashion. Humans would fail miserably if they would be given specialized machine tasks, while on the other hand machines would fail at tasks most humans, regardless of their training, would excel at. While we are generalists, being able to switch seamlessly between different tasks, machines are specialists whose essential specialty is repetition. If you have ever worked in the Social Sciences, chances are you were asked to do machine tasks. Copying something from one Excel spreadsheet to another, albeit in a different layout, is still surprisingly (and horrifyingly) common.[10] Most of this work is apparently delegated to student workers, which basically makes sure that future generations of social scientists will have internalized bad practices from the start.

Besides the essentially dehumanizing aspects, there are many practical reasons to avoid giving machine jobs to

humans. Most prominently: humans are bad at these tasks. Our species is prone to making errors when performing repetitive tasks. Because a machine is instructed by humans via code, their errors are essentially our errors as well, but in this case we can find the error by looking at the code. More precisely, human error seems to be mostly erratic, while machine errors are systematic and can therefore be fixed. Tragedies like the "Excel Depression" in economics (Herndon et al., 2014; Reinhart and Rogoff, 2011) should remind us of the separation between human tasks and machine ones. In this case, errors of experienced researchers and the reluctance to use reproducible code led to empirical inadequate political advice in regards to the financial crisis and contributed to the suffering resulting from the implemented austerity policies.

Instead of doing a machine's job, CSS reminds us of the human job at hand. We are the ones constructing and instructing the machines. Only by learning complete programming languages and gaining at least some partial understanding of digital technologies will we be able to delegate machine tasks effectively. Because these are relatively new skills for social scientists, we cannot rely on a pool of well-trained professionals to make this transition for us. Those of us who do engage in CSS do so for the sake of their own research agendas and not to provide tech support to the rest of the discipline. Therefore, an easy accessible programming language like Python seems to be the right tool to bring digital literacy to the Social Sciences. A simple understanding of what a machine can do and how one can access these capabilities to free oneself from the burdens of repetitive tasks in order to have more time for actual research should be enough.

Because Python is rather flexible, fast, and can be easily extended through modules, it is an excellent candidate for an integrated framework for Social Science research (see also the section 'Innovative Methods'). Looking at the standard practices of the social scientists around us, we found that they employ a mixture of different programs. For example, they would manage their data using one program, analyze it in a second one, write it down in a third one, and finally use a fourth one to create the presentation. Often other more specialized programs like citation management software were added to the mix. This is not necessarily detrimental to the speed and effectiveness of the work process because people can become very accustomed to these procedures. The problem is the loss of flexibility and reproducibility because the entire work flow can become broken when some of the steps change. Technical or methodological requirements can change, leading, for example, to data formats no longer being compatible with the programs used for analysis. The chain of programs can break at any point because in such a work flow all points are interconnected but not necessarily maintained in one framework. This problem is enhanced by the use of proprietary software. A closed format has the consequence that the user has to rely on the developers for ensuring compatibility which results in work flows often being guided by the implicit demands of the software instead of the explicit requirements of the subject matter.

An integrated framework can help to lessen some of these problems. However, the flexibility offered by such a framework will inevitably come at the cost of reorientation and makes it necessary to acquire at least some basic technical knowledge. The Jupyter (former IPython) Notebook keeps this price very low, while at the same time offering a huge functionality. Basically, the notebook is a browser-based editor which functions as a client connected to a kernel. Code can be written inside a notebook and sent to the kernel which will then return the results to this notebook. Since the project became language agnostic, many different additional kernels from various programming languages are offered (e.g. R, Julia, C, Perl). Besides its capability to execute code, the Notebook allows for easy text formatting in Markdown and HTML and provides native support for the rendering of mathematical equations through MathJax. The resulting notebook can be converted to many different formats, which given a specific style template can be publication-ready, including figures and tables. Thanks to easy installable plugins like RISE, the notebook can also be rendered as a JSON slideshow by the click of a button. This slideshow is live, meaning one can type in and execute code within the running presentation, which is especially helpful when it comes to teaching and giving workshops. Jupyter Notebooks therefore provide an integrated, open-source framework with all the functionality one could hope for.

Computational theory building

Theory is an instrument to explain what we observe and to put single elements in (abstract) relations. In other words, theories are ideas about the mechanism behind the evidence that a researcher gathers in order to systematize her observations (Harrington, 2005: 3). These ordering frames should not rely on certain schools of thoughts but on general relationships between elements, their assumptions, and restrictions (as done, for instance, in Friedkin (1986)). The best way to do this would be as a formal model because only as such a critical discourse can be ensured. A formal language allows us to focus on the results and their theoretical implications rather than discuss the choice of words or argue semantics.

Although not much thought regarding theory has been published in the current CSS literature, there exist attempts to model relations of actors to analyze social systems through the interaction among autonomous, cognitive individuals (Conte et al., 2001). Theories of social choice point in the same general direction. They are highly intertwined with computational methods and concentrate on social phenomena also known from game theory regarding, for instance, resource allocation and fair division of goods (Chevaleyre et al., 2007). These approaches are mostly used in combination with social simulations, in which theoretically derived

rules are tested in varying virtual contexts and are considered one especially fruitful approach in CSS (Conte et al., 2012).

However, agent-based modeling faces problems when it comes to the aggregation of individual actions, which is why these variants of rational choice theory have been heavily criticized in Social Sciences (for a recent overview, see Watts (2014: 320)). The general opinion is that they lack (to a different degree) the second basic element of social theory in addition to individual action: structure. Not speaking for all theories of social action in general, we nevertheless argue that we are describing a common ground of theoretical thinking about social phenomena. For almost all theories, the relation between structure and action is central, regardless of whether it is spheres and individuals in Weber's work (Kalberg, 1980), structures and practices in field theory (Bourdieu, 1977; Fligstein and McAdam, 2012), systems and communications (Bailey, 1994), or, in contemporary rational choice, whether it is the interplay ("bridges") between the frame of actions and the rational agent (Kroneberg and Kalter, 2012). We agree with Watts (2014) that these are all variations on the concept of "rationalizable action" (p. 316) in the sense that rational (and computable) actions are assumed to be limited by some sort of boundary. This has been noted earlier and essentially led to the now famous notion of a "bounded rationality" (Simon, 1982) of actors.

Assuming that this general pattern of interdependence between structure and individual in social theory exists, we want to transfer it to the process of coding. We think that spelling out each theory is a necessary step to write any useful code for empirical studies since any code must be translated in a formal language that the computer understands. In order to do this, the theoretical concepts in mind must be rewritten, that is, they have to be formalized. We exemplify four steps that are essential for theory building, or even better, for theoretical application since most theories should be "testable" with that approach.

(1) The objects (agents) in question have to be defined and, hence, operationalized. In sociology, this can be individuals, organizations, or any other form of (collective) actor. (2) The agents have certain attributes that can be latent (e.g. beliefs, expectations) or manifest (gender, age). The attributes are programmatically inherited by Python's classes. This means that properties of objects are stemming from their affiliation to a certain class, resulting in a pattern of succession for derivations of these objects. (3) The objects have certain relations with each other. This step has to be specified, including whether self-loops (i.e. self-references) are useful. In the social world, these relations translate into interactions between actors, for example, by ego communicating with alter, which may implicate ego as a person (or nation, organization, etc.) to a certain part. (4) The patterns of relations between the attributes of the interacting agents from the relations in (3) resemble Heinz von Foerster's (2007) notion of "order from noise" (p. 13). This means that the cooperation of elements runs along intrinsic ordering properties of a "system," which can be any assemblage of elements. In sociological terms, we then observe structure in a sense that not every relation between every agent has an equal probability, but some relations "are more equal than others." This selection process stems directly from the irritations due to the attributes (and accompanying expectations) of the actors in (2) since ego has to be aware of the expectations of alter to arrange his own expectations. By doing so (and not doing other things), certain patterns will emerge that can be generally interpreted as structures, for example, cultural preferences, shared beliefs, or common personal histories.

The process from (1) to (4) is thought to be recursive, meaning that the "heritage" in terms of classes is memorized, but updated continuously by every loop. In this manner, change is incorporated in the model. This corresponds to the concept of "boundary objects" that "are plastic enough to be adaptable across multiple viewpoints, yet maintain continuity of identity" (Star, 1989: 37). The basic objects have such a—programmatically inherited—contingency that they can be defined as the researcher seems it appropriate but are open enough to change according to interaction and structure. In this sense, a code application of the elements of general social theory avoids to think of interaction as a fixed mode in reality, but as something that is continually renegotiated. Also, it is not the efficiency of agents (and their models) alone but mutual interaction and the iterative derivation of structure and the recursive impact of this structure that condition action.[11] Setting up a model in Python, the general layout of the language enforces a researcher to think along these lines.

In addition to the general structuring of theories by a formal language, Python also provides ways for the easy implementation and construction of simulations. Specialized packages like SimPy[12] and nxsim already exist, which are implemented in an object-oriented fashion. As described under step (1), the object-oriented framework lends itself especially well to agent-based simulations. In this case, we can define the base agents as objects with basic strategies (like cooperation and defection in the prisoner's dilemma) and subsequently use these objects as templates for more elaborate types of agents. This way we can progress from simple games to iterative games to evolutionary and topological simulation (demonstrated in pure Python by Isaac, 2008).

The main advantage of using such an approach to simulation and modeling is twofold. First, it forces the scientist to be explicit in the construction of the model. Every step of the model needs to be created in a formal language, leading to a precise and readable theory. Second, it allows for trial-and-error theory development. To test a new idea, *ad hoc* hypotheses or strategies can be easily incorporated in the simulation, thereby creating a kind of experimental setting to explore new ideas and get a feeling for the far-reaching consequences they might have when introduced to complex systems.

Outlook

The Internet has not only revolutionized our daily lives, global economic trade, or the work of intelligence agencies but also the possibilities for researchers. In this article, we discussed certain obstacles of this development and why social scientists are until now bystanders in conducting "CSS," a branch that studies the data delivered by the online social realm. While physicists and other natural scientists are already exploiting the digital connections and their social consequences, we diagnose that social scientists are generally not well-prepared for this endeavor.

As an initial countermeasure, we proposed the usage of Python as an easy-to-learn, general-purpose programming language to facilitate the entrance to computational methods and complex new data. We argue that Python is a very convenient choice since it provides researcher with ample tools to handle the growing amount of rich data on human interactions and offers the prospect of addressing major, and fundamentally interdisciplinary, scientific and social challenges. We discussed its advantages in terms of appropriate structures for the huge collections of data, the easy and fast implementation of methods, the high practicability for operational tasks, and the usefulness of writing code in order to derive empirical applications from theoretical ideas. Beyond this, it is mostly the superior readability and accessibility that makes Python seem like a good choice to promote digital literacy among Social Scientists and thereby opening up the emerging field of CSS.

A growing utilization of Python in Social Sciences may also lead to an enhanced applicability of social scientists' research results. Consultation of practitioners could improve from (rather costly) samples, their distributions, and correlations to very accurate, near real-time analysis of empirical social behavior that we can observe now "live and in action," that is, how people relate to each other in various social contexts and not "only" agglomerations about such social connections or limited examples with specific properties. In this sense, the adaption of computational techniques could bring many new opportunities to social scientists in order to share their expertise and to derive everyday-life applications out of our efforts—which would also increase the probability of getting heard in public discussions and by decision makers. For instance, it would be very practical in many circumstances to develop more efficient ways to connect people and/or ideas over large geographical distances (Heiberger and Riebling, in press). Theoretically and methodically, this would mean to close structural holes, a procedure that is already well established in SNA (Burt, 1995). The challenge at hand would be to adapt these solutions developed for rather small groups on a much larger scale, that is, in fact, the whole globe. What is already done in some privileged corporations and consultancies may be worthwhile to be explored by social scientists. In this way, social scientists could contribute to a better understanding of the complex social system that is now connected by a giant global infrastructure with more transparent processes than any before.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

Notes

1. We will not further elaborate in this article on general problems of "computational social science." This is done in other articles in greater detail (Lazer et al., 2009, 2014; Ruths and Pfeffer, 2014).
2. Calculated by Randal Munroe in his "What If" Blog. This figure does not contain data in "cold" storage.
3. An overview over the essential packages needed to take a look into Big Data analytics using Python and/or R can be found here.
4. There is a second, much more subtle reason to be skeptical about the urgency to utilize Big Data methods in the Social Sciences—the general timeframe in which a task should be executed. For example, no social network could afford to make its customers wait a week to connect with their friends. In the context of research and the construction of abstract models, there is no need to rely on almost instant results.
5. We provided mostly hyperlinks to python wrappers for the general Application Programming Interfaces (APIs) since we focus on Python in this article. It should be mentioned, however, that wrappers in many different languages exist.
6. The tests were run on an Intel Core i5-4300M CPU with 8 GB RAM.
7. https://graph-tool.skewed.de/performance
8. http://clu.uni.no/icame/brown/bcm.html
9. Of course, there are many more inefficiencies and nonsensical time-consuming activities forced upon scientists which are essentially beyond our own control. For example, journals not providing formatting templates or administrative red-tape doled out generously by universities.
10. To give short anecdotal evidence of the "tantalusian" consequences: at the university of one of the authors, a research assistant was specifically hired to transfer data from a website into an Excel sheet and to update that Excel sheet whenever the data on the site changed. The real tragedy stems from the fact that the website essentially serves XML data. Years ago, we had been asked to do a similar task; using an ordinary phone we called the site's administrator and simply asked him politely to send us a copy of the data in question. This helps to show that the problem lies not only in the absence of technical expertise. As long as we are willing to do a machine's job, someone else gets to do the human job.
11. These issues of mutual influence within and between complex social entities are, for instance, also discussed in Niklas Luhmann's (1990) comprehensive work about social systems.

His ideas, in turn, are strongly based on findings about mechanisms of self-organization and perpetuation in biological systems (Maturana and Varela, 1973).

12. A module for process-oriented discrete simulations.

References

Bail CA (2012) The Fringe Effect Civil Society Organizations and the evolution of media discourse about Islam since the September 11th attacks. *American Sociological Review* 77: 855–879.

Bail CA (2014) The cultural environment: Measuring culture with big data. *Theory and Society* 43: 465–482.

Bailey KD (1994) *Sociology and the New Systems Theory: Toward a Theoretical Synthesis*. Albany, NY: SUNY Press.

Bird S, Klein E and Loper E (2009) *Natural Language Processing with Python—Analyzing Text with the Natural Language Toolkit*. Sebastopol, CA: O'Reilly Media.

Bourdieu P (1977) *Outline of a Theory of Practice*. Cambridge, UK: Cambridge University Press.

Brandes U (2001) A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology* 25: 163–177.

Buldyrev SV, Parshani R, Paul G, et al. (2010) Catastrophic cascade of failures in interdependent networks. *Nature* 464: 1025–1028.

Burrows R and Savage M (2014) After the crisis? Big Data and the methodological challenges of empirical sociology. *Big Data & Society* 1: 1–6.

Burt RS (1995) *Structural Holes: The Social Structure of Competition* (reprint edition). Cambridge, MA: Harvard University Press.

Chevaleyre Y, Endriss U, Lang J, et al. (2007) A short introduction to computational social choice. In: van Leeuwen J, Italiano GF, van der Hoek W, et al. (eds) *SOFSEM 2007: Theory and Practice of Computer Science*. Berlin, Heidelberg: Springer, pp. 51–69.

Cioffi-Revilla C (2014) *Introduction to Computational Social Science* (Texts in Computer Science). London: Springer.

Conte R, Edmonds B, Moss S, et al. (2001) Sociology and social theory in agent based social simulation: A symposium. *Computational & Mathematical Organization Theory* 7: 183–205.

Conte R, Gilbert N, Bonelli G, et al. (2012) Manifesto of computational social science. *European Physical Journal: Special Topics* 214: 325–346.

De Vaan M, Stark D and Vedres B (2015) Game changer: The topology of creativity. *American Journal of Sociology* 120: 1144–1194.

Deerwester S, Dumais ST, Furnas GW, et al. (1990) Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41: 391–407.

Diekmann A, Jann B, Przepiorka W, et al. (2014) Reputation formation and the evolution of cooperation in anonymous online markets. *American Sociological Review* 79: 65–85.

Downey A (2012) *Think Python: How to Think Like a Computer Scientist*. Sebastopol, CA: O'Reilly Media.

Dumbill E (2012) *What Is Big Data?* O'Reilly Media. Available at: https://beta.oreilly.com/ideas/what-is-big-data (accessed 3 August 2015).

Feinerer I, Hornik K and Meyer D (2008) Text mining infrastructure in R. *Journal of Statistical Software* 25: 1–54.

Fligstein N and McAdam D (2012) *A Theory of Fields*. New York: Oxford University Press.

Freeman LC (2011) The development of social network analysis—With an emphasis on recent events. In: Scott J and Carrington PJ (eds) *The SAGE Handbook of Social Network Analysis*. London: SAGE, pp. 26–54.

Freeman LC and Webster CM (1994) Interpersonal proximity in social and cognitive space. *Social Cognition* 12: 223–247.

Friedkin NE (1986) A formal theory of social power. *Journal of Mathematical Sociology* 12: 103–126.

Giles J (2012) Computational social science: Making the links. *Nature News* 488: 448.

Girvan M and Newman ME (2002) Community structure in social and biological networks. *Proceedings of the National Academy of Sciences* 99: 7821–7826.

Goel S, Hofman JM, Lahaie S, et al. (2010) Predicting consumer behavior with Web search. *PNAS* 107: 17486–17490.

Golder SA and Macy MW (2011) Diurnal and seasonal mood vary with work, sleep, and daylength across diverse cultures. *Science* 333: 1878–1881.

Hagberg AA, Schult DA and Swart PJ (2008) Exploring network structure, dynamics, and function using NetworkX. In: Varoquaux G, Vaught T and Millman J (eds) *Proceedings of the 7th Python in Science Conference (SciPy2008): Presented at the SciPy2008*. Pasadena, CA, pp. 11–15. Available at: http://conference.scipy.org/proceedings/scipy2008/paper_2/

Harrington A (2005) *Modern Social Theory*. Oxford: Oxford University Press.

Heiberger RH (2014) Stock network stability in times of crisis. *Physica A: Statistical Mechanics and its Applications* 393: 376–381.

Heiberger RH (2015) Collective attention and stock prices: Evidence from Google Trends Data on Standard and Poor's 100. *PLoS ONE* 10: e0135311.

Heiberger RH and Riebling J (2015) U.S. and whom? Structures and communities of international economic research. *Journal of Social Structure* 16: 1–12.

Helbing D (2012) Introduction: The FuturICT knowledge accelerator towards a more resilient and sustainable future. *European Physical Journal: Special Topics* 214: 5–9.

Herndon T, Ash M and Pollin R (2014) Does high public debt consistently stifle economic growth? A critique of Reinhart and Rogoff. *Cambridge Journal of Economics* 38: 257–279.

Homans GC (2003) *The Human Group*. New Brunswick, NJ: Transaction Publishers.

Isaac AG (2008) Simulating evolutionary games: A Python-based introduction. *Journal of Artificial Societies and Social Simulation* 11: 8.

Kalberg S (1980) Max Weber's types of rationality: Cornerstones for the analysis of rationalization processes in history. *American Journal of Sociology* 85: 1145–1179.

Karsai M, Kivelä M, Pan RK, et al. (2011) Small but slow world: How network topology and burstiness slow down spreading. *Physical Review E* 83: 025102.

Kerzner M and Maniyam S (2013) *Hadoop Illuminated*. Hadoop Illuminated LLC. Available at: http://hadoopilluminated.com/hadoop_illuminated/hadoop-illuminated.pdf

King G (2011) Ensuring the data-rich future of the social sciences. *Science* 331: 719–721.

Kitano H (2002) Computational systems biology. *Nature* 420: 206–210.

Kroneberg C and Kalter F (2012) Rational choice theory and empirical research: Methodological and theoretical contributions in Europe. *Annual Review of Sociology* 38: 73–92.

Lazer D, Kennedy R, King G, et al. (2014) The parable of Google Flu: Traps in Big Data analysis. *Science* 343: 1203–1205.

Lazer D, Pentland A, Adamic L, et al. (2009) Computational Social Science. *Science* 323: 721–723.

Lehmann TC, Rolfsen JA and Clark TD (2015) Predicting the trajectory of the evolving international cyber regime: Simulating the growth of a social network. *Social Networks* 41: 72–84.

Lewis K, Kaufman J, Gonzalez M, et al. (2008) Tastes, ties, and time: A new social network dataset using Facebook.com. *Social Networks* 30: 330–342.

Luhmann N (1990) *Essays on Self-Reference*. New York: Columbia University Press.

Lutter M (2015) Do women suffer from network closure? The moderating effect of social capital on gender inequality in a project-based labor market, 1929 to 2010. *American Sociological Review* 80: 329–358.

McCullough BD (2010) Econometric computing with R. In: Vinod HD (ed.) *Advances in Social Science Research Using R*. New York: Springer, pp. 1–21.

Maturana HR and Varela F (eds) (1973) *Autopoiesis and Cognition: The Realization of the Living*. Dordrecht: D. Reidel.

May RM, Levin SA and Sugihara G (2008) Complex systems: Ecology for bankers. *Nature* 451: 893–895.

Milgram S (1967) The small world problem. *Psychology Today* 2: 60–67.

Onnela J-P, Saramäki J, Hyvönen J, et al. (2007) Structure and tie strengths in mobile communication networks. *PNAS* 104: 7332–7336.

Peixoto TP (2015) graph-tool: Efficient network analysis with python. Available at: https://graph-tool.skewed.de/ (accessed 29 July 2015).

Qin X, Cunningham P and Salter-Townshend M (2015) The influence of network structures of Wikipedia discussion pages on the efficiency of WikiProjects. *Social Networks* 43: 1–15.

Rehurek R and Sojka P (2010) Software framework for topic modelling with large corpora. In: *Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks*, 22 May, pp. 45–50. Malta: ELRAValetta.

Reinhart CM and Rogoff KS (2011) The forgotten history of domestic debt. *The Economic Journal* 121: 319–350.

Ruths D and Pfeffer J (2014) Social media for large studies of behavior. *Science* 346: 1063–1064.

Savage M and Burrows R (2007) The coming crisis of empirical sociology. *Sociology* 41: 885–899.

Schweitzer F, Fagiolo G, Sornette D, et al. (2009) Economic networks: The new challenges. *Science* 325: 422.

Scott J and Carrington PJ (2011) *The SAGE handbook of social network analysis*. Los Angeles, CA: SAGE.

Simon HA (1982) *Models of Bounded Rationality: Empirically Grounded Economic Reason*. Cambridge, MA: MIT Press.

Star SL (1989) The structure of Ill-structured solutions: Boundary objects and heterogeneous problem solving. In: Gasser LM and Huhns N (eds) *Distributed Artificial Intelligence*. London: Pitman, pp. 37–54.

Steane A (1998) Quantum computing. *Reports on Progress in Physics* 61: 117–173.

Von Foerster H (2007) *Understanding Understanding: Essays on Cybernetics and Cognition*. New York: Springer Science+Business Media.

Uprichard E (2013) *Big Data, Little Questions?* Discover Society. Available at: http://www.discoversociety.org/focus-big-data-little-questions (accessed 11 January 2016).

Wassermann S and Faust K (1994) *Social Network Analysis: Methods and Applications*. Cambridge, MA: Cambridge University Press.

Watts DJ (2004) The "New" science of networks. *Annual Review of Sociology* 30: 243–270.

Watts DJ (2014) Common sense and sociological explanations. *American Journal of Sociology* 120: 313–351.

Watts DJ and Strogatz SH (1998) Collective dynamics of "small-world" networks. *Nature* 393: 440–442.

Wild F (2015) *LAS: Latent Semantic Analysis*. Available at: https://cran.r-project.org/web/packages/lsa/index.html

Wimmer A and Lewis K (2010) Beyond and below racial homophily: ERG models of a friendship network documented on Facebook. *American Journal of Sociology* 116: 583–642.

Author biographies

Raphael H. Heiberger is a postdoc at the Institute for Sociology at the University of Bremen. His primary research interests are in social network analysis, statistical computing, and econophysics.

Jan R. Riebling is a doctoral student at the chair of theoretical sociology at the university of Bamberg. His main research interests are computational models of social symbols, mathematical sociology and network analysis.