Network Science **8** (4): 596–608, 2020
doi:10.1017/nws.2020.12

CAMBRIDGE
UNIVERSITY PRESS

COMMENTARY

Commentary: A network science summer course for high-school students

Florian Klimm[1*†] and Benjamin F. Maier[2†]

[1]Oxford Centre for Industrial and Applied Mathematics, Mathematical Institute, University of Oxford, Oxford OX2 6GG, UK; Department of Statistics, University of Oxford, 24-29 St Giles', Oxford OX1 3LB, UK; Department of Mathematics, Imperial College London, London, SW7 2AZ, UK; MRC Mitochondrial Biology Unit, University of Cambridge, Cambridge Biomedical Campus Hills Road, Cambridge, CB2 0XY, UK and [2]Department of Physics, Humboldt Universität zu Berlin, Newtonstraße 15, D-12489 Berlin, Germany; Robert Koch-Institute, Nordufer 20, D-13353 Berlin, Germany (e-mail: benjaminfrankmaier@gmail.com)
*Corresponding author. Email: f.klimm@gmail.com

Action Editor: Stanley Wasserman

Abstract

We discuss a two-week summer course on "Network Science" and "Complex Systems" that we taught for 15 German high-school pupils of ages 16–18. In this course, we covered topics in graph theory, applied network science, programming, and dynamic systems alike. We find that "Network Science" is a well-suited course for introducing students to university-level mathematics. We reflect on difficulties regarding programming exercises and the discussion of more advanced topics in dynamic systems. We make the course material available and encourage fellow network scientists to organize similar outreach events.

Keywords: outreach, mathematics, physics, summer course, teaching

1. Introduction and motivation

The science behind networks and complex systems is a prime example for a modern interdisciplinary field with rich applications and approachable real-world problems that are frequently treatable with basic mathematical concepts already taught in high school. Nevertheless, teaching network science is usually completely omitted in high-school curricula (Cramer et al., 2015a,b; Sayama et al., 2015).

Hence, scientists organized outreach events for teenagers to bring them into contact with the study of networks, notably in England (Harrington et al., 2013) and Spain (Sánchez & Brändle, 2014). These outreach events consisted of an introductory presentation, followed by work in smaller groups on topics such as "node importance," "disease spread and vaccination strategies," and "why your friends have more friends than you do." Based on the success of these half-day events, we decided to design a two-week summer course for 15 German high-school pupils with the topic "Networks and Complex Systems" (German: *Netzwerke und komplexe Systeme*). It was one of six courses taught in parallel within the framework of an established German summer school system called *Deutsche SchülerAkademie* (DSA, German Pupils Academy) which in our case took place in a monastery school in Roßleben, a small town in the Free State of Thuringia (Germany). The DSA concepts are described in more detail in Appendix A.

[†]Both authors contributed equally.

The longer duration of the course and the higher age range (16–18 years) of the participants allowed us to extend the material covered in the aforementioned shorter outreach events, go into more mathematical detail, and include some components of programming with Python. In particular, our goal was to show how problems from mathematics, physics, biology, and engineering can be easily connected and solved with simple mathematical and numerical frameworks, a point we feel is often neglected in high-school curricula. We wanted to provide future young scientists in a potential variety of fields with a "complex-systems perspective" on distinct scientific problems. Other techniques usually omitted in high school but covered in this course were the communication of scientific ideas in oral presentations and writing.

In Section 2, we introduce and motivate the didactical elements the course consisted of, some of which were given by the academy framework.

A reader solely interested in the content of the course might want to skip this part of the paper to continue with Section 3, where we summarize the course topics, describe the contents of smaller hands-on modules, and discuss our course material (i.e., problem sheets and programming exercises).

In Section 4, we discuss our approaches and evaluate to what extent they were successful or might need further improvements.

We make our teaching material available online (Klimm & Maier, 2017).

2. Course structure and didactical elements

In the following, we briefly outline in chronological order the didactical and organizational elements that we used for the development of the course.

Prior to the academy, we sent a questionnaire to all 15 participants to receive information about their background knowledge and personal interests, described in Subsection 2.1. Based on their replies, we compiled a *Reader* containing preparation material, the content of which we outline in Subsection 2.2. We further sent out personalized material to each student such that, based on their interests, they were to prepare individual presentations as described in Subsection 2.3.

The main material of the course was then taught in both teacher-centered lectures and more interactive pupil-centered modules. The ratio of these was carefully balanced to both ensure effective learning as well as an accurate depiction of how content is presented in scientific programmes at German universities. The structure of those units is described in Subsection 2.4, and their contents are outlined in brief in Section 3 and discussed in detail in Appendices B and C. During the modules, pupils at times deepened their understanding by solving mathematical and programming exercises that we prepared as problem sheets, which we describe in Subsection 2.5.

Subsection 2.6 discusses the *Rotation*, which entailed pupils giving presentations on their learnings to students and teachers from other courses. At the end of the academy, participants had to write a report on the course called *Dokumentation*, a process explained in Subsection 2.7.

The elements *Reader*, *Rotation*, and *Dokumentation* were given by the framework of the *Deutsche SchülerAkademie*.

2.1 Participant questionnaire

We received a list with contact details of all 15 participants two months prior to the academy. We expected some heterogeneity in existing knowledge because the pupils were in one of the final two grades in German high school. This discrepancy is enhanced by the federal education system in Germany which gives different foci to subjects in different states (Lohmar & Eckhardt, 2014). To assess the knowledge of the pupils, we conducted a nonanonymous online survey which was answered by all 15 participants. We mainly assessed the mathematical knowledge in 11 brief questions, for example, "Do you know what a matrix is?". The results gave us an overview over the pupils' abilities: All pupils knew how to operate with vectors and two-thirds knew how to work with matrices, almost all of them were able to differentiate, but less knew about Riemann

https://doi.org/10.1017/nws.2020.12 Published online by Cambridge University Press

integration. The most heterogeneous knowledge was given for programming with a third of the participants having no programming experience and others knowing Python or C-family programming languages rudimentarily. Three quarters did not know LaTeX, which we intended to use for the *Dokumentation* (see Subsection 2.7). We also included an open-ended question about their motivation to choose this particular course. The pupils stated that they were especially interested because the course connects mathematical tools with real-world applications and targets questions from different sciences, such as physics and biology.

2.2 Reader

Based on the results from the questionnaire, we provided the participants with a so-called *Reader* as preparation for the course. The main purposes of this document were to give all participants a common level of mathematical background and to provide them with some programming experience. For this we drew from two sources, (i) the Python online course "Spielend programmieren lernen" (engl. "Learning Programming Effortlessly") by the Hasso-Plattner-Institute (Meinel & Schweiger, 2017; von Löwis, 2017) and one chapter of "Physik mit Bleistift" (engl. "Physics with a Pencil") (Schulz, 2004). The latter gave a brief introduction to vectors and differentiating. In the post-course evaluation, the pupils were, however, not satisfied with the material as it focused too much on mathematical tools that we used only briefly and not enough on learning programming.

2.3 Pupil presentations

Each of the participants gave a presentation during the course. In preparation for the course, we suggested a topic to each of them based on their interests as deduced from the survey and provided them with reading material. The talks were supposed to be 10–15 min long, and we advised the participants to approach the topics from a phenomenological side. For example, Turing patterns were introduced under the topic "how leopards get their patterns". Our aim was to discuss the mathematics behind the phenomena together after the introduction given by the pupils. Furthermore, we wanted to confront pupils with a situation in which they had to independently research and explain an academic topic.

2.4 Lectures and modules

Lectures were a fundamental part of our course. Usually, they lasted 1–1.5 h each and covered concepts that were introduced by the pupils earlier in their presentations. We subsequently formalized the mathematical description and went into more detail. Lectures were typically given early in the day and accompanied by modules in afternoon classes. These modules were usually of a duration of 2 h and had a stronger component of pupil involvement and interactive discussion. We received very positive feedback on those more hands-on tasks from the pupils. Being able to explore network concepts on their own, based on the world around them, was very motivating for them.

The content of these lectures and modules is described in Section 3.

2.5 Problem sheets and programming exercises

We created six problem sheets that gave the pupils the chance to apply some of the learned concepts. We initially planned to discuss one problem sheet per day but quickly realized that our schedule was too ambitious. The problem sheets usually consisted of a mixture of pen-and-paper analytical questions, as well as programming exercises. For those, we used Python as the main and only programming language because it is open source and relatively easy to learn and implement. In addition to the default Python functions, we mainly used three libraries, NETWORKX, MATPLOTLIB, and NUMPY. We also used Gephi (Bastian et al., 2009) for the analysis of social networks and as an easy way to graphically access them.

https://doi.org/10.1017/nws.2020.12 Published online by Cambridge University Press

We estimate that in order to cover all problems on a single sheet, a student would need at least three hours. The sheets, however, were designed in an open-ended way such that students that were already familiar with the subjects would also be challenged by additional, optional tasks.

We have made all problem sheets in the original German version and a translated English version available online (Klimm & Maier, 2017).

2.6 Rotation

The *Rotation* is an inherent part of each DSA academy. It occurs at half-way through the course, thus after approximately seven days of covered material. It is a half-day event in which each course has to organize 20-minute presentations for pupils of other courses of the academy.

The aim of this task was to simulate a pseudo-academic situation in which the pupils learn how to present obtained knowledge to somebody outside of their area.

We planned the talks in our course and discussed which topics were appropriate to present to the other pupils.

Our participants identified that the fundamental mathematical definitions should be covered and social networks would be a good example to make the topic accessible, while deeper mathematical discussions would be inappropriate given that the audience would consist of other academy participants that do not necessarily have a strong mathematical background. The feedback they received on the presentations was very positive. The participants as well as the lecturers of the other courses were astonished by how applicable and approachable mathematics can be. Feedback like this strengthens our view that network science itself can serve as a “figurehead” topic inside mathematics because it easily contradicts the usual stereotypes of mathematics being incomprehensible and overly complicated.

2.7 Dokumentation

The *Dokumentation* (German for documentation) is another inherent part of each DSA summer school. It is a written report in which each course summarizes the covered material. Each participant contributed a small one- to two-page part, while we served as editors.

Despite most participants having no knowledge of LaTeX we decided to write the whole report with SHARELATEX, an online LaTeX editor that allows real-time collaboration and online compiling (Oswald et al., 2015)[1] since they will surely encounter it later on in their academic life.

The first step was the discussion of an overall structure of the *Dokumentation*. We did not impose any approach but discussed different potential approaches with the pupils. The final structure was similar but not identical to the one we chose for the course overall. The second step was the individual discussion with each pupil about the material they wanted to cover and what illustrations were necessary. All figures in this paper were created by pupils as part of the documentation.

The creation of the *Dokumentation* was a new challenge for the participants. One encountered problem was the space limitation, such that the presented information had to be condensed, while staying comprehensible. Furthermore, appropriate referencing was unfamiliar to the pupils. Most of them did not encounter these aspects before and thus found them challenging. Overall, however, we were impressed with the result and needed only two to three iterations of editing per pupil before the texts met the appropriate quality.

3. Course content

3.1 Choice of topics and motivation

The results of the participant questionnaires (see Subsection 2.1) determined that pupils had a decent knowledge of precalculus (functions, derivatives, and basic vector algebra) that we

https://doi.org/10.1017/nws.2020.12 Published online by Cambridge University Press

therefore assumed as a prerequisite for the course content. They further showed that participants had broad interests covering, e.g., mathematics, physics, biology, and computer science.

Our main goal as teachers was for pupils interested in STEM fields to be confronted with academic content and university-level problems for the first time. Typically, German high-school curriculae do not discuss science as a potentially interdisciplinary academic endeavor.

Because both network and complexity science have applications in all of the aforementioned fields, we expected them to be well-suited topics to show pupils how their seemingly different interests can be very well connected.

To this end, we began the first part of the course teaching basic graph theory as a foundation for more applied topics we discussed later on. It also allowed us to introduce pupils to basic proof techniques such as proof by contradiction and proof by induction.

As participants planned on joining various STEM university programmes after high school, we further hoped that a discussion of the principles of dynamic systems would make them sensitive for the discipline's implications within the pupils' respective future fields, for instance regarding the principles of stability and bifurcations in biology.

An overarching aim was also to teach them programming, a skill that is of ubiquitous necessity in modern science but usually omitted in German high school.

3.2 Lectures

We gave lectures on the following topics, each of which is described and motivated in more detail in Appendix B:

1. Introduction to Graph Theory (see Subsection B.1)
2. Graph Coloring (see Subsection B.2)
3. Social Networks and Friendship Paradox (see Subsection B.3)
4. Effective Resistance (see Subsection B.4)
5. Dynamic Systems and Fractals (see Subsection B.5)

While the lectures on graphs and networks seemed to be of an appropriate difficulty, the segments on dynamic systems and fractals were perceived as too difficult. Especially working with ODEs was a completely new subject for most of the pupils and hence frustrating at times. In a future course, we would refrain from discussing Lotka–Volterra systems, deterministic chaos, fractals, and Turing patterns. One could, however, still think about the discussion of susceptible-infected dynamics (and concurrently equilibria in simple one-dimensional systems), as those are easily applicable to networks.

As described in Section 2, lectures were often preceded by student presentations that they prepared prior to the summer school. An overview over all topics given by us and a brief assessment of their suitability is shown in Table 1.

3.3 Modules

In addition to longer lectures, we also used a couple of smaller modules, which had a stronger component of pupil involvement and interactive discussion. We will briefly introduce some of them in Appendix C:

1. How to Communicate a Graph (see Subsection C.1)
2. Analog Creation of Random Graphs with Dice (see Subsection C.2)
3. Creating a Floor Plan Network (see Subsection C.3)
4. Infectious Dynamics (see Subsection C.4)
5. Small-World Networks (see Subsection C.5)

https://doi.org/10.1017/nws.2020.12 Published online by Cambridge University Press

Table 1. Overview of some of the topics presented by pupils in talks.

Topic	Comment
Four color map problem	Good introduction to graph coloring. Allows a discussion about computer-assisted mathematical proofs
Seven bridges of Königsberg	Classic and easily accessible problem
Traveling salesman problem	Allows introduction to computational complexity
Kirchhoff's circuit laws	Well-suited to introduce pupils without a physics background to physical phenomena
Percolation theory	Easily accessible with, e.g., forest tree density and forest fires
Neuronal networks	Popular topic but too complex for deep discussion
Graph isomorphism problem	Allows discussion of $\mathscr{P}$ vs. $\mathscr{NP}$
Disease spreading	Excellent for tying together network phenomena and dynamics
Predator–prey interactions	Good introduction to dynamical systems but too complex for deep discussion
Turing patterns	Accessible with, e.g., animal fur patterns but too complex for deep discussion
Deterministic chaos	The logistic map is well-suited for simple analyses of chaotic phenomena
Fractals	Nicely accessible with, e.g., coast lines and Koch's curve
Fractal dimension	Counter-intuitive but easily accessible with box counting method

3.4 Problem sheets

Lectures and modules were at times complemented by one of six problem sheets that we make available online (Klimm & Maier, 2017). The content of those sheets is summarized as follows:

1. Simple graphs, deriving graph properties and graph generation with Python
2. Node and edge coloring, coding greedy coloring algorithm
3. Matrix algebra, graph Laplacian, resistor networks
4. Contagion dynamics on the floor plan network, centrality measures
5. Dynamic systems in discrete and continuous time, linear stability
6. Ordinary differential equations, Euler method, Lotka–Volterra systems, creating fractals with Lindenmayer systems

While students were appropriately challenged by the problems on sheets 1, 2, 4, and 5, they had considerable difficulties with the remaining problem sets. In particular, the final sheet was perceived as overwhelming and frustrating.

3.5 Programming exercises

Given the results from our initial questionnaire, we were aware that the pupils had very different levels of programming knowledge (cf. Subsection 2.1). To approach this problem, we created two groups of similar size, one working on their own on the programming exercises and the other receiving closer guidance.

The programming tasks were very diverse in their difficulty. We started with the creation of the synthetic graphs discussed earlier in the lecture. To make the task more feasible, we gave the code to create a path graph P_n of n nodes (see Listing 1 in Appendix D). After a discussion of the code and answering questions like "Why is the FOR loop progressing from 0 to $n-1$?," they had to adopt the code to create functions that construct other graphs such as the empty graph N_n, the cycle graph C_n, and the complete graph K_n. We registered that a subset of the pupils had problems with this task, and a careful discussion of the code was necessary. A line-by-line dismantling of the script on the blackboard was especially helpful to increase understanding.

Other tasks included the illustration of the networks with the NETWORKX `draw` function and the computation of network metrics such as degree and betweenness. For each exercise, we also had one or two rather complex tasks, such as the implementation of the Dijkstra-algorithm for finding shortest paths (Dijkstra, 1959) to keep even the most advanced participants engaged. We clearly communicated that we did not expect the pupils to solve all available tasks to ensure that they did not feel overwhelmed by the programming exercises. We discussed the findings of all tasks with the whole class.

4. Conclusions

We were overall content with the course, the pupils' involvement, and their progress. We think that network science is an appropriate topic to give pupils an introduction to university-level science and mathematics, as we noted from the feedback during the *Rotation*, even for those who are not naturally interested in such topics.

We encountered the disparity in background, especially concerning programming, as the most challenging part because it aggravated the task of keeping all the participants engaged. Since we were two instructors, we were able to split the class into a beginner and advanced half, which helped in some cases. A better, more detailed, discussion of programming for those who had never programmed before might have been fruitful.

A crucial part of our teaching was the flexibility in the covered topics. Sometimes the pupils had problems with concepts that we thought were taught at school, e.g., the binomial coefficient, while other new concepts took much less time to discuss than we anticipated.

One weakness of our preparation was the *Reader*. We should have tailored it more to the specific course content and given "homework" problems instead of reading material only. This would have allowed us to challenge the pupils with more advanced topics, such as more complicated differential equations, and thus go into more detail in the course itself.

Although the discussion of a broad range of topics was praised by the pupils, we felt that a focus on networks under omission of a majority of the dynamic systems part would have been beneficial. The latter is an interesting and complex branch of science and would provide enough material for a course by itself. In retrospect, we would definitely drop the discussion of Lotka–Volterra systems, Turing patterns, chaos, and fractals, while a discussion of stability and infection dynamics might still be interesting as an example for the application of network science to real-world problems.

The students' feedback on the course was strongly positive yet included some constructive criticism. The pupils especially liked the programming parts. A criticism mentioned by multiple participants was the *Reader*, which they found too challenging and not tailored enough for the course material. Some of the students would have preferred a stronger focus on the complex systems part and less emphasis on pure graph theory. Most students asserted that the course did not change their future choice of study because they already planned on attending a scientific university programme prior to the academy. They stated, however, that attending the course reinforced their decision.

We also want to note that the organization of such a course has been rather beneficial to ourselves. It allowed us to recapture many different topics in graph theory and network science and think about ways to motivate and connect them with each other.

Finally, as well as Harrington et al. and Sanchez et al., we encourage fellow network scientists to use the presented material to tailor their own course to spread the fascination about networks and dynamic systems among teenagers and to contact us about ways in which we can collaborate toward that goal.

Acknowledgments. We would like to thank all the pupils who participated in the course for their engagement and criticism. It was a great pleasure working with them for the two weeks of the program. Further thanks goes to the central organization of the DSA in Bonn, as well as the academy administration and fellow course organizers at the DSA Roßleben 2016. We thank our doctoral advisers Mason A. Porter and Dirk Brockmann for supporting us in this endeavor and for their fruitful discussion. We are grateful to Sarah M. Griffin for proofreading the manuscript.

Funding. This work is supported by the EPSRC and MRC (FK, grant numbers EP/L016044/1, EP/R513295/1, EP/N014529/1) and by the Joachim Herz Stiftung (BFM).

Conflict of interest. The authors have nothing to disclose.

Note

[1] There are plenty of other LaTeX online editors available, such as Overleaf, Papeeria, and Authorea.

References

Bastian, M., Heymann, S., & Jacomy, M. (2009). Gephi: An open source software for exploring and manipulating networks. In *International AAAI Conference on Weblogs and Social Media*.

Cramer, C., Sheetz, L., Sayama, H., Trunfio, P., Stanley, H. E., & Uzzo, S. (2015a). Netsci high: Bringing network science research to high schools. In G. Mangioni, F. Simini, S. M. Uzzo, & D. Wang (Eds.), *Proceedings of the 6th Workshop on Complex Networks CompleNet 2015* (pp. 209–218). Springer.

Cramer, C., Porter, M. A., Sayama, H., Sheetz, L., & Uzzo, S. (2015b). Network literacy – essential concepts and core ideas. Retrived fom https://sites.google.com/a/binghamton.edu/netscied/teaching-learning/network-concepts.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, *1*(1), 269–271.

Harrington, H. A., Beguerisse-Díaz, M., Rombach, M. P., Keating, L. M., & Porter, M. A. (2013). Commentary: Teach network science to teenagers. *Network Science*, *1*(02), 226–247.

Kaminski, J., Schober, M., Albaladejo, R., Zastupailo, O., & Hidalgo, C. (2012). Moviegalaxies-social networks in movies. Consulté sur http://moviegalaxies.com.

Klimm, F., & Maier, B. F. (2017). Teaching material for summer school on network science for high school students. https://github.com/floklimm/DSAMaterialNetworks.

Lohmar, B., & Eckhardt, T. (2014). The education system in the Federal Republic of Germany 2012/2013: A description of the responsibilities, structures and developments in education policy for the exchange of information in Europe. In *Bonn: Secretariat of the standing conference of the ministers of education and cultural affairs of the Länder in the federal republic of germany*.

Maier, B. F., & Brockmann, D. (2017). Cover time for random walks on arbitrary complex networks. *Physical Review E*, *96*(4), 042307.

Meinel, C., & Schweiger, S. (2017). openHPI–Das MOOC-Angebot des Hasso-Plattner-Instituts. In *Veranstaltungen 4.0* (pp. 195–226). Springer.

Newman, M. (2010). *Networks: An introduction*. Oxford: Oxford University Press.

Oswald, H., Allen, J., & Gough, B. (2015). ShareLaTeX, the online LaTeX editor.

Sánchez, A., & Brändle, C. (2014). More network science for teenagers. arxiv preprint arxiv:1403.3618.

Sander, M. 2010 (8). Photography of the monastery school Roßleben. Retrieved from https://de.wikipedia.org/wiki/Datei:Klosterschule_Ro%C3%9Fleben.JPG.

Sayama, H., Cramer, C., Porter, M. A., Sheetz, L., & Uzzo, S. (2015). What are essential concepts about networks? *Journal of Complex Networks*, cnv028.

Schulz, H. (2004). *Physik mit Bleistift*. Verlag Harri Deutsch.

Strogatz, S. H. (2012). *Friends you can count on*. New York Times.

Ulanowicz, R. E., & DeAngelis, D. L. (2005). Network analysis of trophic dynamics in south florida ecosystems. In *US geological survey program on the South Florida ecosystem*, vol. 114.

Völmicke, E. 2017 (10). Deutsche Schülerakademie.

von Löwis, M. 2017 (10). Spielend Programmieren lernen!

Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature*, *393*(6684), 440.

West, R., Pineau, J., & Precup, D. (2009). Wikispeedia: An online game for inferring semantic distances between concepts. In *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence* (pp. 1598–1603).

https://doi.org/10.1017/nws.2020.12 Published online by Cambridge University Press

Appendix A. Embedding in the "SchülerAkademie" framework

The course was organized and given as a part of the *Deutsche SchülerAkademie* (DSA), which is an extracurricular summer school for motivated and gifted pupils in the final two years of German high-school (Völmicke, 2017). For more than 20 years every summer, multiple of these academies have been organized where each of them consists of six courses for 15–16 pupils each. Our course was part of the academy "Roßleben 2016" and set in a monastery school in a small town in Thuringia. The courses concurrent with ours covered diverse academic topics from analyzing Richard Wagner's operas to String Theory. Approximately one hundred pupils participated in the courses and were encouraged to organize academic and non-academic extracurricular activities. Over the two weeks, the participants spent a total of approximately sixty hours (twelve days with five hours each) in their courses. In this time, they worked under guidance of the lecturers. Furthermore, they had to prepare talks for the participants of other courses explaining the topics of their course, a concept called *Rotation* (see Subsection 2.6), and create a written summary of the learned material, called *Dokumentation* (see Subsection 2.7).

Appendix B. Lectures

B.1 Introduction to graph theory

This lecture was built on the discussion about the representation of graphs as edge lists and adjacency matrices (see Subsection C.1). We proceeded to introduce simple network measures as the number of edges m, node degree k, path length L, and density ρ. We then discussed some deterministic synthetic networks such as the empty graph N_n, the path graph P_n, the complete graph K_n, and the complete bipartite graph $K_{n,n}$ and derived formulas of the network measures in dependence of the graph size n. Later in a programming exercise, the student validated these formulas numerically.

B.2 Graph coloring

Our aim was to teach the pupils network science and classical graph theoretical concepts alike. *Graph coloring* was an appropriate field for the latter because the underlying problems are easily accessible, while the proofs can be challenging. We introduced node coloring and edge coloring but focussed on the former. The pupils particularly enjoyed proving theorems and challenged our proofs if they were too sloppy. In the last part of this lecture, we also discussed coloring algorithms, particularly *greedy coloring*. One task the pupils enjoyed was to compute an upper bound of the chromatic number χ of a symmetrized food web (Ulanowicz & DeAngelis, 2005) consisting of $n = 63$ animals in the Everglades by applying greedy coloring many times to different orders of the nodes. We motivated this question by asking how many different cages would be necessary if a zoo director wanted to keep all the animals without one species preying on the other.

B.3 Social networks and friendship paradox

The discussion of social networks is well-suited to motivate the real-world applicability of network science to the pupils. The participants were rather astonished by the so-called *friendship paradox*. We followed Strogatz's excellent explanation in the New York Times (Strogatz, 2012).

We proceeded by discussing the existence of clusters and other mesoscale structures in networks. Unfortunately, for privacy reasons, Facebook does not allow downloading your own friendship network anymore. This used to be an excellent way to make pupils familiar with various network characteristics as communities and centrality measures. Therefore, we had to fall back on other available data, e.g., the social graphs from MovieGalaxies (Kaminski et al., 2012) and B. F. Maier's Facebook friends network from 2014 (Maier & Brockmann, 2017). As another example of a small-world network, we played the "Wikipedia-Game" to demonstrate short link paths between apparently unrelated articles. Its objective is to find the shortest link path between two random Wikipedia articles by only following links on the encountered Wikipedia pages (West et al., 2009).

B.4 Effective resistance

A lot of the pupils showed interest in physics. Therefore, we dedicated a day to the discussion of resistor networks. This topic was first introduced by a pupil's presentation on Kirchhoff's circuit laws. Subsequently, we aimed at computing the effective resistance of more complex networks where we followed Newman's discussion in Newman (2010). In particular, we proceeded as follows (see also Exercise Sheet 3; available under Klimm & Maier, 2017).

At first, we motivated eigenvalues and vectors of matrices with a rudimentary example of a growing population of linearly interacting species. Using this concept, we were able to motivate the inverse matrix and what it means when a matrix has eigenvalue zero (the concept of singularity), which was necessary to explain the reduction of the graph Laplacian to make it invertible.

https://doi.org/10.1017/nws.2020.12 Published online by Cambridge University Press

This topic was very difficult for some of the younger pupils who were not as versatile in working with matrices as the older ones. However, some of the more engaged participants particularly enjoyed those discussions.

B.5 Dynamic systems and fractals

In addition to traditional network science, we also discussed *dynamic systems*, e.g., simple exponential growth or more complex Lotka–Volterra equations. The pupils enjoyed the discussion, although an introduction of mathematical prerequisites, e.g., ordinary differential equations (ODEs), was necessary. Even though introducing those prerequisites took a rather long time, we think it was valuable for the pupils to learn about ODEs using very simple systems. In particular, we showed how to derive the equation of motion for logistic growth given the example of the mean-field model of the susceptible-infected infection model. Here, we started with discrete time and probabilistic considerations and arrived at the final equation by approaching continuous time using $\Delta t \to 0$.

We furthermore discussed chaotic systems by analyzing the logistic map in discrete time, as well as the stability of equilibria since those are very accessible with high-school-level calculus and graphical methods equally. Especially the discussion of stability was hence well-received, and the participants seemed to grasp the concept of phase transitions easily by analyzing the logistic growth equation in continuous time.

The lectures on dynamic systems ended with the discussion of fractals (e.g., as coast lines) and fractal dimensions (by introducing the box counting method). While the pupils seemed to understand the fundamental concepts, they had significant problems reproducing self-similar structures using Lindenmayer systems and Python's implementation of "turtle," a drawing engine that uses simple iterative instructions.

In retrospect, we feel that this part of the course was a bit out of scope and potentially overwhelming for the pupils.

Appendix C. Smaller teaching modules

C.1 How to communicate a graph

This module was one of the first elements of our course. We separated the pupils into two groups. Each of them was given a printout of a small labeled graph with the aim to communicate the graph in written form to the other group such that they were able to recreate the same graph.

The pupils quickly identified ways to achieve this, e.g., as an edge list. After joining both groups together, we discussed different approaches and their limitations, e.g., that edge lists do not allow the representation of isolated nodes but an additional node set is required. This naturally led to the introduction of other representations such as adjacency matrices. One of the graphs was dense and the other sparse, which lead to the discussion of space and memory considerations when working with large graphs in different representations.

This simple task was very successful because it clearly motivates the definition of a graph as node and edge set. Giving the mathematical definitions exclusively can intimidate pupils who are not familiar with the notion of sets.

C.2 Analog creation of random graphs with dice

After introducing Erdős–Rényi random graphs $G(n, p)$, we gave the participants the task to create undirected random graphs with the help of six-sided and twenty-sided dice. We discussed how to generate graphs with different connection probabilities p and also identified an appropriate number n of nodes. We settled for $n = 10$ nodes given that 45 dice rolls per network seemed a reasonable time effort. We then split the pupils into four groups, each creating a graph of different connection probability $p \in \{1/20, 1/10, 1/6, 1/3\}$. Each group drew their final graph on the blackboard (see Figure C.1) and we subsequently discussed their properties, e.g., the size of their largest connected component and their degree distributions $P(k)$ and compared them with theoretical expectations.

This task was praised as "very fun" by the pupils. We evaluated it as successful from a didactical perspective because it playfully connects known material, such as dice rolls and probabilities, with new concepts, such as random graphs. Furthermore, it allows the transition to the creation of random graphs using pseudorandom number generators of a computer. However, the introduction of the binomial coefficient to explain the random graph's degree distribution was rather challenging for the majority of the pupils.

C.3 Creating a floor plan network

In addition to synthetic networks and the networks created from data that we provided, we wanted the participants to experience a more realistic experience of network creation. For this we gave them access to a printout of the floor plan of

https://doi.org/10.1017/nws.2020.12 Published online by Cambridge University Press

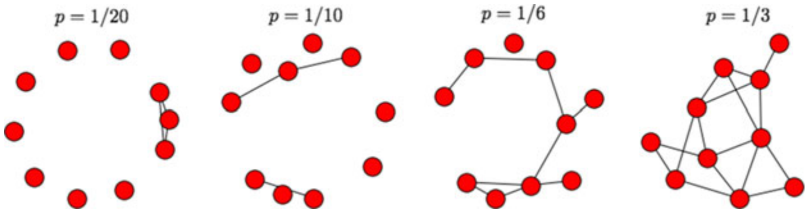

Figure C.1. Four $G(n = 10, p)$ random graphs as generated by the participants by rolling dice. A twenty-sided die was used for the probabilities $p = 1/20$ and $p = 10$ and a six-sided die for $p = 1/6$ and $p = 1/3$.

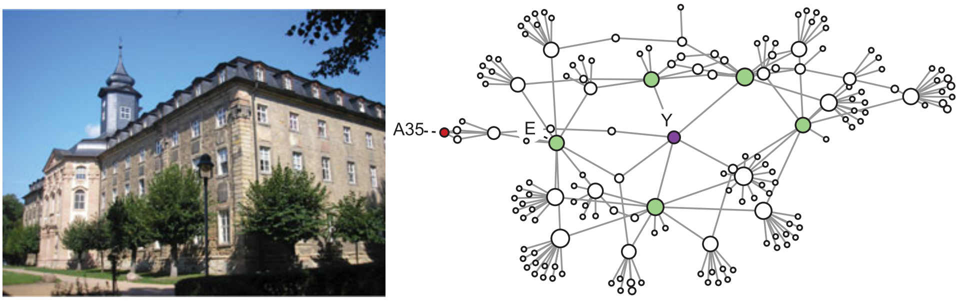

Figure C.2. Left: The monastery school building in Roßleben (Sander, 2010). Right: Network illustration of the floor plan of the former monastery school as created by the pupils. Marked are the eastern staircase "E" which has a high betweenness centrality and is close to the epidemic outbreak room "A35," as well as the yard "Y".

the *Klosterschule Roßleben*, which hosted the academy (see Figure C.2). They had the task to create a network in which nodes represent the $n = 161$ rooms and edges stand for direct connections between them.

We let the pupils organize among themselves to take on this task and to discuss how to overcome the problems they faced. They decided to split into groups, each of them covering one floor of the building. They labeled each room with a unique alphanumerical identifier consisting of a letter for the floor and a number for the room. The four stairwells, which connected floors, were given the label "East," "South,' "West," and 'North'. The unique labeling allowed the parallel creation of different parts of the network which were joined together later on. To avoid making mistakes in the network creation process, each floor's network was generated independently twice. The pupils then used their knowledge about necessary conditions of graph isomorphism, e.g., identical number of edges and degree distributions, to check whether both were potentially identical. If this was not the case they knew that there was a discrepancy between the created networks and they investigated which of the two creators made a mistake.

The generated network, which is given in the course material (Klimm & Maier, 2017) and shown in Figure C.2, was then further analyzed by the pupils. For example, they investigated the degree distribution and compared different centrality measures. They found a positive correlation between degree and betweenness, while some nodes contradicted that notion by having a much higher betweenness than expected from their degree. They identified these nodes to represent staircases that connect the hallways with each other.

C.4 Infectious dynamics

We used the created school building network for the simulation of compartmental models in epidemiology, specifically the susceptible-infected dynamics (SI), as shown in Figure C.3. We invented a scenario where another course of the academy, working with bacteria germs, let a hyperinfective strain escape which slowly infects the whole building room by room. The pupils then compared the results with the mean-field approximation introduced in the lectures.

We subsequently discussed three vaccination strategies: (i) random vaccination, (ii) vaccination of nodes with high betweenness, and (iii) random next-neighbor vaccination. The temporal development of the number $I(t)$ of infected nodes

https://doi.org/10.1017/nws.2020.12 Published online by Cambridge University Press

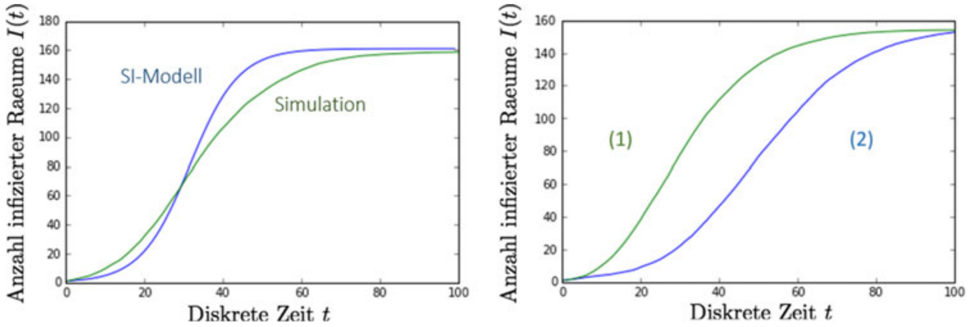

Figure C.3. Left: Simulated SI dynamics on the monastery room network (green) in comparison with a mean-field approximation (blue). Right: Spread of the disease as number of infected rooms over time for random vaccination of one room (green, (1)) and targeted vaccination of staircase "E" (blue, (2)). Both figures were produced by one of the participants and thus have German labels. Horizontal axes give the time steps, and vertical axes show the number $I(t)$ of infected rooms.

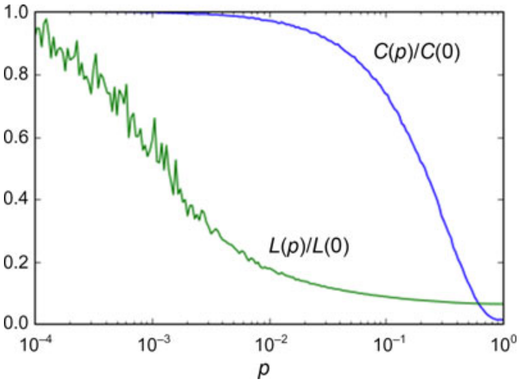

Figure C.4. Normalized characteristic path length $L(p)/L(0)$ and normalized clustering coefficient $C(p)/C(0)$ for the family of randomly rewired graphs with rewiring probability p, as discussed by Watts & Strogatz (1998). This figure was produced by a pupil.

for the former two is shown in Figure C.3, and the pupils were able to identify that the betweenness vaccination is slowing the spread of the disease. The final steady state, however, shows the same number of infected rooms.

This module successfully connected different aspects of the course, betweenness centrality, and SI dynamics, with a real-world application. Furthermore, it demonstrated the effectiveness of vaccination to the pupils, and we were able to discuss the aspect of herd immunity.

C.5 Small-world networks

Besides the introduction of essential concepts for networks, we also wanted to familiarize the pupils with scientific literature. For this we chose the first half of Watts' and Strogatz' famous paper *Collective Dynamics of "small-world" networks* (Watts & Strogatz, 1998). The task we gave them consisted of two parts: (i) read the paper and explain the definition of *clustering coefficient* and (ii) recreate Figure 2, which displays the characteristic path length and clustering coefficient of a lattice graph under rewiring.

This task was challenging but fruitful for the pupils. All of them quickly understood the contents of the paper. For the programming exercises they needed more help. We were, however, rather impressed by the pupils' abilities. Reproducing the results of an important paper, as shown in Figure C.4, led to a feeling of achievement as well as motivation for the upcoming lectures. Note that for time limitations we omitted averaging over a large number of realizations, which was used in the original paper. However, we discussed with the students the impact of and reason for such averaging procedures.

Appendix D. Code sample

This code sample creates the path graph P_{10} with ten nodes and returns the adjacency matrix to the terminal. We provided it to the students as starting point for the creation of other graphs, e.g., empty graphs and the complete graph C_8 with eight nodes.

```
# import necessary library
import numpy as np

# define a function
def path_graph( n ):

 #  Returns the (n x n) adjacency matrix
 #  of a path graph with n nodes.


    # empty adjacency matrix of shape (n x n)
    A = np.zeros((n,n))

    # loop over every node besides the last one
    for u in range(0, n-1):


        # connect this node to the subsequent one
        A[u,u+1] = 1

    # symmetrize matrix by adding its transpose
    # the original matrix
    # this ensures that not only u is conn-
    # ected to v, but the other way around, too.
    A = A + np.transpose(A)

    # the matrix will now be returned
    return A

# call the function to obtain an adjacency matrix
# for a path graph of 10 nodes
path_10 = path_graph(10)

# put the matrix on screen in text
print(path_10)
```

Listing 1 Python code for creating a path graph P_{10} with ten nodes as provided to the students as starting point for further programming, as the creation of a complete graph C_8 with eight nodes.

Cite this article: Klimm F. and Maier B. F. (2020). Commentary: A network science summer course for high-school students. *Network Science* **8**, 596–608. https://doi.org/10.1017/nws.2020.12