Curriculum

JMBE Journal of Microbiology & Biology Education

Biotechnology Data Analysis Training with Jupyter Notebooks

Ulf W. Liebal,[a] Rafael Schimassek,[a] Iris Broderius,[a] Nicole Maaßen,[a] Alina Vogelgesang,[b] Philipp Weyers,[b] and Lars M. Blank[a]

[a]*Institute of Applied Microbiology, ABBT, RWTH Aachen University, Aachen, Germany*
[b]*Center for Learning Services, RWTH Aachen University, Aachen, Germany*

Biotechnology has experienced innovations in analytics and data processing. As the volume of data and its complexity grow, new computational procedures for extracting information are being developed. However, the rate of change outpaces the adaptation of biotechnology curricula, necessitating new teaching methodologies to equip biotechnologists with data analysis abilities. To simulate experimental data, we created a virtual organism simulator (*silvio*) by combining diverse cellular and subcellular microbial models. With the silvio Python package, we constructed a computer-based instructional workflow to teach growth curve data analysis, promoter sequence design, and expression rate measurement. The instructional workflow is a Jupyter Notebook with background explanations and Python-based experiment simulations combined. The data analysis is conducted either within the Notebook in Python or externally with Excel. This instructional workflow was separately implemented in two distance courses for Master's students in biology and biotechnology with assessment of the pedagogic efficiency. The concept of using virtual organism simulations that generate coherent results across different experiments can be used to construct consistent and motivating case studies for biotechnological data literacy.

KEYWORDS biotechnology, data analysis, Jupyter Notebooks, Python, recombinant expression, systems biology, mathematical model, cloning, growth modeling, recombinant protein production

INTRODUCTION

Biotechnology is increasingly generating vast amounts of complex data that require advanced computational analyses (1). The experiments include, among others, multiomics investigations (2), high-throughput techniques, or multiplexed experiments (3), and computational modeling approaches demand advanced data skills (4). In parallel, new tools facilitate data analysis by simplified machine learning scripts, e.g., the Python SciKit learn library (5), and easily accessible data analysis environments, like Jupyter Notebooks (6), Galaxy (7), KBase (8), or KNIME (9). Data analysis with Jupyter Notebooks is becoming particularly popular (10) and has been used to guide metabolomic data analysis (11), metabolic engineering (12, 13), and gene expression (14). However, these developments in computational analysis have rarely managed to receive adequate attention in the curriculum of biotechnology.

Increasing the share of data analysis and bioinformatics in the biology and biotechnology curricula are excellent means for developing critical 21st-century skills by fostering inquiry-based interdisciplinary learning (15). Student motivation is particularly important, because the topic can deviate from many students' prime interests and skills. Several reports have discussed problem-based learning and flipped classrooms in computational biology (15–18). A particularly useful tool for motivation can be gamification and serious game elements (19, 20), which contribute more strongly to positive self-assessment than exam results (21).

Python is particularly popular because of the extensive community that supplies and maintains easily accessible packages to support general tasks from machine learning to specific biological solutions (22). The Jupyter Notebook technology is an interactive Web-based tool that allows one to combine programming, computational output, explanatory text, and multimedia resources in a single browser-accessible document. Jupyter Notebooks are developed to facilitate data analysis and code sharing for Python (6) and are popular teaching resources, e.g., in engineering (23, 24), with guidelines for their setup (19, 25). Jupyter Notebooks are run on a server, a JupyterHub, that provides the Python computational environment, the

Editor Catherine Vrentas, The Engaged Scientist
Address correspondence to Institute of Applied Microbiology, ABBT, RWTH Aachen University, Aachen, Germany. E-mail: ulf.liebal@rwth-aachen.de.
The authors declare no conflict of interest.
Received: 4 August 2022, Accepted: 20 December 2022, Published: 16 January 2023

TABLE 1
Overview of intended audience, prerequisite knowledge, learning time, and outcomes of the recombinant expression simulation Jupyter Notebook workflow[a]

Parameter	Details
Intended audience	Late Bachelor, early Master Biology, Biotechnology
Learning time	180 min
Prerequisite knowledge	Distinction of Eukaryotes, Prokaryotes and Archaea, DNA structure and principles of gene expression, basic linear algebra, Internet and file browsing
Teaching content	Background lecture on growth and gene expression (40 min)
	Reading and quiz of *Nature* article on Jupyter Notebooks (25 min)
	Python and Jupyter introduction for programming basics (25 min)
	Biotech simulation data analysis group activity (80 min)
	Result wrap-up and reflection (10 min)
Learning objectives	Informative
	• Recognize promoter architecture of −35 and −10 boxes
	• Explain strain specificity of growth and biomass
	• List factors that impact cloning efficiencies
	Performative
	• Distinguish Python variable types
	• Calculate DNA melting temp
	• Calculate growth rate and maximum biomass with linear regression

[a]The teaching focus was on biotechnology-related data analysis and was suited for all programming experience levels. A Python introduction was available for beginners, whereas advanced students were provided with complex programming tasks. All required background information regarding growth experiments and promoter architecture was covered by the preceding lecture. Calculations of melting temperature were provided with the Notebook in the associated documentation of the simulation.

Kernel. The JupyterHub is accessed with an Internet browser that renders the Jupyter Notebook on the client computer. Thus, users need no special software, hardware, or operating system except Internet connection with a browser.

The course was taught remotely with Zoom as well as in-person seminars with an optimal timing of 3 h and contained a background lecture (see the supplemental material), individual reading, and Jupyter Notebooks with a Python introduction and the biotechnology data simulation (Table 1). The course delivered learning objectives for Biology, namely, students could outline the box-based promoter-driven gene expression, calculate primer melting temperature, and analyze data to estimate growth rates, as well as computational objectives, namely, the students could learn Python code and can use Jupyter Notebooks. Student motivation is fostered by presenting the simulation in the context of a biotechnology-relevant narrative, Moodle quizzes, by social learning in student groups, varied learning media, and gamification of the simulations. A rubric is provided and details the outcome level for the different learning categories (Text S1). An evaluation of the course was performed by anonymous feedback that included prior knowledge, competence inventory and experience, and usability of the Jupyter Notebook. The teaching materials containing lecture, Python introduction Notebook, biotechnology simulation Notebook, and Moodle course implementation (see the supplemental material) are available with CC BY 4.0 license online at https://git.rwth-aachen.de/ulf.liebal/biolabsim.

PROCEDURE

The simulator of virtual organisms

The microbial phenotypes are simulated in Python with an ensemble of models connected with the *silvio* (simulator of virtual organisms) framework. *silvio* is a Python package containing multiple models to simulate microbial physiology, such as logistic growth and gene expression, and experimental parameters, such as primer annealing temperature. *silvio* generates surrogate molecular and microbiological data without the attempt to reproduce real phenotypes. Instead, at each start of *silvio*, a new virtual organism is initiated with unique parameters to generate realistic data regarding type, structure, and complexity. The data are simulated by models of the associated experiments, including mechanistic models, i.e., growth laws, and statistical models, i.e., random forest for gene expression prediction. Fig. 1 provides an overview of the architecture of *silvio*.

silvio is organized in a parent object (exp) from which all experiments and genetic manipulation is performed. Two types of functions can be distinguished, one for changing host parameters and another for performing analytical experiments. The host object class contains all biologically relevant information, for example, the host strain (e.g., *Escherichia coli*, *Pseudomonas putida*), the maximum biomass of cultivation (strain specific, 30 to 145 g

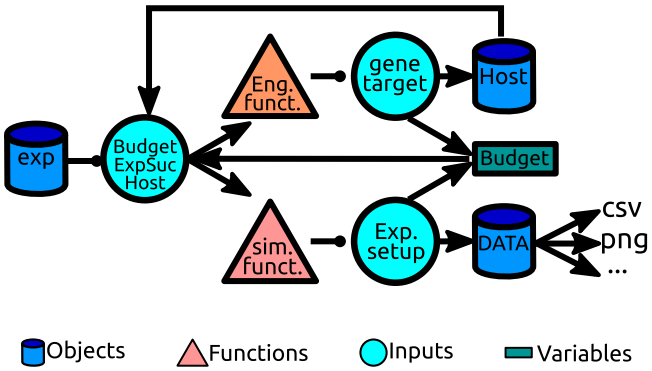

FIG 1. The *silvio* code logic. A global experiment object (exp) is the starting point of functions for host engineering and genetic manipulation (Eng. funct.) and for analytical experiments (sim. funct.). Both functions require input about the host for the experiment (Host), available money (Budget), and experiment success probability (ExpSuc). The host organism is defined as an object with user-defined properties (strain type) and randomly initiated variables (optimal temperature, maximum biomass). Simulation data are stored in data objects with associated functions for postprocessing.

of cell dry weight [CDW]/liter), the optimal temperature (25 to 40°C), the optimal primer length for cloning (16 to 28 nucleotides [nt]), and a correction factor for the expression strength (30 to 50), as well as budget information. The budget represents money which has to be invested in the initial lab equipment and in each experiment. The limited budget strengthens the importance of rational experiments and precludes a massive random search for the solution. The amount of investment in the initial lab equipment determines the probability of experiment failures. In addition to the budget resource, the time dimension is considered such that experimental functions are designed to last a few seconds. Host-changing functions alter the values stored in the host object. Cloning, the only current host-changing function, duplicates the host properties with changes in the genetic information. The data are stored in separate objects to facilitate processing, visualization, and export.

silvio contains several models to simulate biological processes which are measured or optimized during biotechnological strain engineering and include growth, the growth constant, DNA melting temperature, and gene expression (Table 2). A logistic growth model based on the Verhulst equation simulates the growth experiments to determine optimal growth temperature. The maximum biomass (K) is randomly initiated between 30 and 100 and 45 to 145 g CDW/liter for *E. coli* and *P. putida*, respectively. The temperature dependence of the growth constant (r) is calculated via a normal distribution, and the optimal temperature is randomly initiated between 25 and 40°C.

Empirical formulas are used to calculate the DNA melting temperature and the gene expression strength. The optimal primer length is randomly initiated between 16 and 28 nucleotides (26). Two equations are used to calculate the optimal melting temperature: the first equation for primers of <25 nucleotides and the second equation for larger primers (Table 2). Gene expression is predicted based on the promoter sequence with 40 nucleotides upstream of the open reading frame. The regression is performed by a random forest machine learning module trained with measurements of a σ^{70}-dependent synthetic promoter library expressed in *E. coli* and *Pseudomonas taiwanensis* (14). The *silvio* simulation environment is available as a ready-to-install PyPI-package at https://pypi.org/project/silvio, and the Jupyter Notebook is available at https://github.com/uliebal/RecExpSim/. The Jupyter Notebook is provided as a HTML file (in the supplemental material) and can be cloned from Github and adapted as to the requirements of the respective courses. The biotechnology data analysis simulation can be run without any further software installation and without accounts using the public, GitHub-connected JupyterHub Binder. A link is provided in the README.md file in the GitHub repository.

General teaching setup

We developed an instructional workflow to convey biotechnological principles of recombinant expression within a modern data analysis environment. The course starts with a

TABLE 2
Models in silvio for biological processes

Process	Model	Details	Output
Growth	Verhulst logistic model $P(t) = K/\{1 + [(K - P_0)/P_0]e^{-rt}\}$	$P(t)$, biomass (g CDW/liter); K, max. biomass, random; P_0, 0.1 g (CDW)/liter (initial biomass); r, growth constant, temp dependent	CSV file with growth data, columns for time (h) and biomass (g CDW) for different temps
Growth constant	Normal distribution	μ, mean optimal temp, random; σ = 5 (variance)	No output
DNA melting	$T_m = 2(A + T) + 4(C + G)$	A, C, G, T are the no. of specific base present	No output
	$T_m = 64.9 + 0.41(GC) - (600/N_{Nt})$	GC content (%); N_{Nt}, number of nucleotides	
Gene expression	Random forest regression	*E. coli*, *P. taiwanensis* promoter library (unpublished)	CSV file with strain information, columns for host, gene, promoter, GC content, temp, biomass, expression

TABLE 3
Computational steps of the Jupyter Notebook workflow[a]

Step	Variable	Activity	Challenge
1. Host choice	• Seed: int	• Read general introduction	• Python variable assignment
	• myInvest: int		
	• myMutant: str		• Cost: free
2. Host characterization	• Temperatures: array of int	• Analyze Excel data	• Variance in biomass level
		• Estimate growth curve parameters	• Growth expt fails
			• Cost: 100 €
3. Promoter design	• mySequenceID: str	• Learn importance of promoter boxes	• Optimal primer length unknown
	• myPromoter: str	• Calculate DNA melting temp	• Cost: 200 €
	• myPrimer: str	• Derive primer complementary to promoter	
	• myTm: int		
4. Evaluation	• host_names: list	• Judge effectiveness of promoter	• Multiple rounds of promoter design
	• gene_name: str	• Test impact of GC content on expression	• Cost: 500 €
	• cult_temp: int		
	• growth_rate: float		
	• biomass: int		

[a]int, integer; str, string; list, list of strings; float, real number.

lecture on the theory of growth phases, equations, and gene expression to ensure a balanced competence level. The students proceed to an individual reading of a *Nature* article about the development, popularity, and caveats of Jupyter Notebooks (10), followed by an individual quiz to self-check understanding. After the theoretical background, the students start coding in a Python and Jupyter introduction with different levels for novices about variable types and functions, medium levels about loops, and professional experience about file input-output and list comprehension. In the last 90 min, the biotechnology simulation itself is conducted. A Moodle course room was our central navigation space for the students, along the different steps of the course, while Jupyter Notebooks were provided by external Jupyter Hubs as links from the Moodle (in our case, the RWTH JupyterHub and Binder).

The biotechnology simulation is embedded in a motivating narrative about a fictional biotech company that aims to engineer a strain for coronavirus vaccine production and encourages a competition for the highest production rates among student participants. The students are told that the company provides a budget for the strain development and they need to decide on an investment for general, unspecified laboratory equipment while bearing in mind future experiment expenditures. Too little investment into the general equipment leads to higher experimental failure rates, whereas too-generous investment limits the amount of experiments that can be performed (Table 3). The simulated steps are (i) host organism choice and characterization, (ii) promoter sequence design and cloning, and (iii) final expression rate measurement. The final rates depend only on the promoter sequence and the precision by which the students measure host growth properties. During the simulation, different experiments are performed by using predefined functions appropriately. Each experiment requires resources: the money budget is limited (Table 3), and some experiments are time-consuming to stimulate economic and effective use of the simulated experiments.

The teaching unit was conducted as part of a biotech training for biology and biotechnology Master students at the RWTH Aachen University and the Westfälische University of Applied Science, respectively. The courses were conducted as Zoom meetings and in-person seminars, starting with a lecture, individual reading, Python and Jupyter introduction, and biotech simulation. The biotech simulation was preceded by a walk-through of the simulation, including the solution to all steps (15 min). Next, participants were allotted to two-person groups in breakout rooms to work autonomously for 85 min on solving the simulation with regular support (every ~20 min) by a supervisor. We found that about 75% of the participants managed to test at least one successful vaccine expression. To increase success rate in the future, we will ask the student groups to use a group-specific seed number, to divide the growth and cloning sections among themselves, and to finally integrate their results. In the last ~10 min, all participants joined a conclusion round, during which the statistical relationship of promoter sequence and expression was examined, the production rates were compared, and the group with best production was honored.

Data analysis in recombinant expression

The recombinant engineering workflow is simulated with four selected steps and data analysis tasks (Table 3). The first input from the student is an integer seed, which determines the performance of the random number generator. The

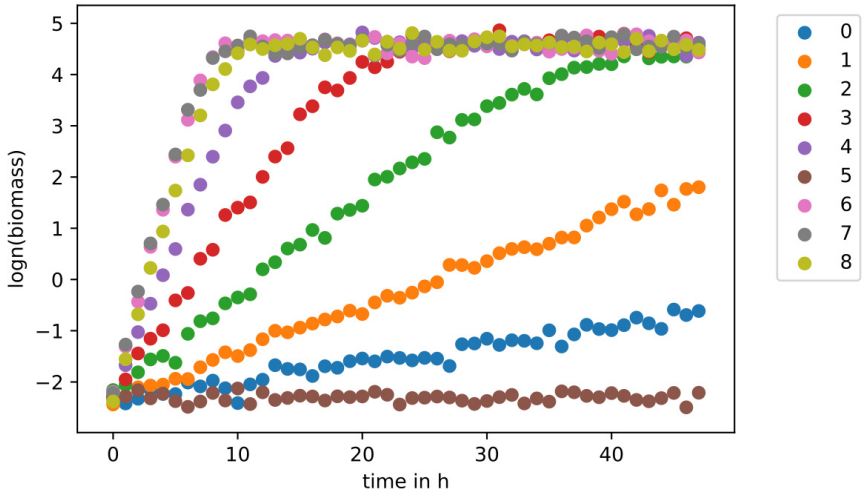

FIG 2. Example experiments (at 22 to 38°C) to determine optimal growth temperature. Five temperatures separated by 4°C were analyzed, which required ~15 s of simulation time. The experiment at 34°C resulted in the most robust linear growth increase to ~7 h. The experiment at 38°C failed, with an expected growth rate comparable to that for the experiment at 30 to 34°C. Very slow growth was measured at the suboptimal temperature of 22°C.

parameters in *silvio* are randomly initiated, and the seed ensures that students can continue with the same host characteristics over several course dates and can work in a group on an identically parameterized virtual host. The second parameter is the host bacterium choice, predefined as either *E. coli* or *P. putida*. Both are popular bacterial hosts and are related to important pathogens. The host choice does not affect the final rate, which is normalized to the maximum possible host-specific expression rate, but it does affect the maximum biomass concentration and the promoter activity. Thus, the host choice is rather a cosmetic feature which we included to have the opportunity to discuss two popular prokaryotic expression systems. Along with the host, the user also decides how much money is invested in the laboratory equipment. The experimental success rate is a hyperbolic function with variables of the total budget predefined by the lecturer and the investment of equipment chosen by the students, and it reaches an optimal plateau at ~20% of total budget. In each step, the students have to assign values of the right type (string, list, integer, float, array of integers) to variables to start the simulations. The generated data for analysis are stored in comma-separated value files online, which can be downloaded for further analysis.

Growth characterization

In the second step, the user identifies the optimal growth temperature and the associated growth rate and biomass. The optimal temperature and the maximum biomass are randomly initiated and the maximum growth rate is 1/h (see Procedure). The user inputs a vector with the test temperatures as an argument to the experiment function from *silvio* (simulate_growth in the experiment class) to simulate growth experiments to identify the optimal temperature. The experiment for each temperature costs 100 EUR and takes ~3 s of simulation time. With a low probability of ~10%, depending on investment in equipment, the growth fails and remains at the inoculum level. The outcome of this experiment is a CSV file with data formatting as generated by the instrument GrowthProfiler (Enzyscreen BV). The students could choose subsequent data analysis in Excel, LibreOffice, or Python. In Excel or LibreOffice, the data have to be correctly imported, followed by logarithmic operation and visualization. The temperature data with the strongest biomass increase are then subjected to linear regression within the linear regimen of the logarithmic data to determine the growth rate and the average biomass during the stationary phase in the original data. The procedure was explicitly presented to the students. In Excel, it involves using the functions for natural logarithm, the sloped function to estimate the growth rate in the initial linear regimen, and the average function to determine the mean maximum biomass level. To encourage Python-based analysis, we provided, disordered, the lines of code (Parsons puzzle). The students had to understand and rearrange the script. Two Python blocks exist: for visualization (Fig. 2), and for linear regression to extract growth rate and biomass. Analogous to the Excel procedure, the user identifies the optimal temperature along with growth rate and biomass.

Promoter construction and cloning

After characterizing the host organism, the students identify a suitable promoter sequence and an abstract cloning procedure is simulated. The key learning objectives are knowledge about the structure of a standard σ^{70} promoter, including boxes that control prokaryotic gene expression, computation of DNA melting temperature, and appreciation of the fickleness

of cloning. In the lab, cloning proceeds along numerous steps and the simulation only captures the first step of designing a promoter and cloning into a plasmid. The subsequent experimental steps require artful procedures that can be appreciated only in real lab courses. A 40-nt promoter reference sequence is provided (GCCCA**XXXXXX**AXGC*XXX*C*X*CGT*XXX*GG **XXXXXX**TGCACG; Xs in italics represent positions to be replaced by standard nucleotides, and the boldface Xs represent biologically important −35 and −10 boxes, respectively). The optimal composition of the box positions is explained in the accompanying text and literature links (27). The primers for cloning start with the first nucleotide of the promoter, whereas the optimal primer length is unknown and randomly initiated between 16 and 28 nt. The user then calculates the melting temperature according to the basic formula given by the equation in Table 2. The cloning experiment has a higher error rate and identical configurations can lead to different results with relatively frequent cloning failures. The cloning process is time-consuming and weakly predictive and aims to mirror the process of the learning of a beginner. The students are motivated to try several versions of promoters to improve expression.

Simulation results and cross-connection

Following promoter design, vaccine expression values are measured and compared among the group, and the possibility of correlations between GC content and expression are examined. The users first measure the promoter strength of each promoter construct. *silvio* contains an expression prediction algorithm for *E. coli* and *P. taiwanensis* promoters that was trained on libraries of the respective hosts with 40-nt promoter sequences, and it predicts green fluorescent protein expression in fluorescence units per gram of dry weight (14). The final simulated experiment reports the vaccine production, but only if correct values from the host characterization experiments are provided (optimal temperature, biomass, and growth rate are within 10% of the correct values) (see the variables in Table 3). As each group will only have tested 1 to 4 promoter variants, combining the results across the groups can lead to further insight. We investigated correlations between GC content and vaccine expression strength for all tested promoters, and we used the following two strategies to secure and integrate the students' simulations (Fig. 3). Strategy 1: during online Zoom courses the students reported their value pairs via the chat function and the lecturer copied the values into Python to generate a plot. Strategy 2: for in-person classes, the corresponding plot was developed on a blackboard, with groups sequentially naming their value pairs.

DISCUSSION

The course was supported with an evaluation to estimate the didactic benefit of the teaching concept. Established questionnaires in this study were used to obtain comparable, accurate, and trustworthy data. The teaching evaluation took place at the end of the course, between the 21 April and 25 May 2021, via an online questionnaire implemented on the platform SoSciSurvey (questions are reported in Text S1 of the supplemental material). In total, 16 students responded via the questionnaire, which amounted to an ~40% feedback rate; the students are registered in biology and biotechnology, with 60% females. Due to the small sample size, a statistically reliable analysis of the results was not guaranteed.

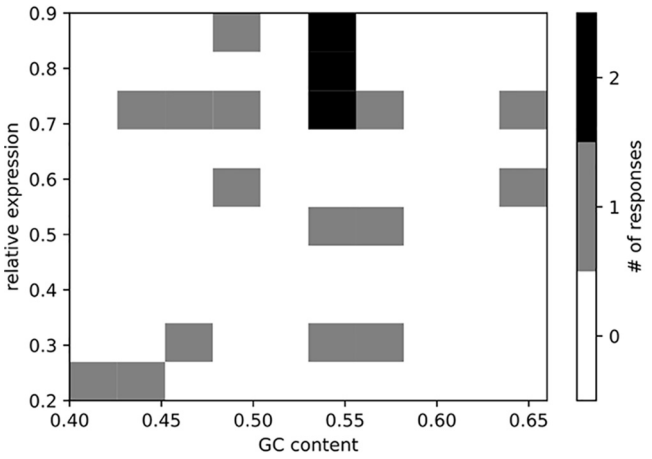

FIG 3. Joint simulation results for all groups for determining the relationship of GC content to expression strength. The chart shows the potential to combine results from all groups to arrive at new knowledge when the effort of all groups is integrated. We concluded that GC content and relative expression did not correlate linearly.

Survey instruments for teaching evaluation

To evaluate the educational benefit of the Jupyter Notebook-based course, we asked the course participants to self-assess their prior knowledge, perceived competence, and competence gains, as well as the increase in topic-related interest. We were also interested in how the course participants perceived the user-friendliness of the Jupyter Notebook and how they rated the course overall. Perceived competence was assessed using the KIM questionnaire (a short scale of intrinsic motivation) (28). The questionnaire contained 12 perception-related statements (e.g., joyfulness, satisfaction, distress) and students rated the extent of agreement on a scale of 1 (full opposition) to 5 (full agreement). Competence gains were explored using the Graz evaluation model of competence acquisition (GEKO) for higher-education courses (29). The GEKO contains a list of six different areas, for example "the acquisition of theoretical, subject-related knowledge (e.g., theories and their contexts)" or "the ability to work in a team (e.g., coordination of cooperation with fellow students, division of tasks)," reported on a scale of 1 to 5. This scale of assessment was also used to evaluate increase in topic interest and the prior knowledge of students (novice to expert). A newly

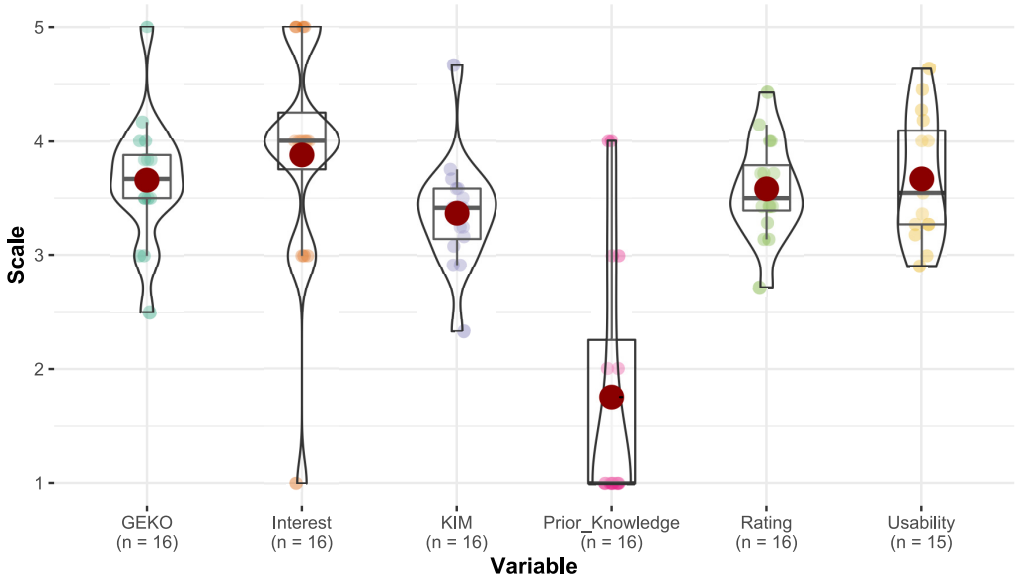

FIG 4. Distributions of total scores of scales regarding competence acquisition (GEKO), the students' assessments of the extent to which their interest in the topics was increased by the course (interest), the perceived competence experience (KIM), how the students assessed their prior knowledge before attending the event (prior knowledge), the overall evaluation of the event (rating), and the usability of the Jupyter Notebook.

designed question section assessed the usability: experienced independence, the user-friendliness of the interface, the topic-related knowledge, and the complementarity to the parallel lecture (scale of 1 to 5). The survey ended with the students' overall evaluation of the course by seven self-developed questions (scale of 1 to 5).

Evaluation of the survey

In the self-rated evaluation after the course (questions are provided in Text S2), the students reported on average little prior knowledge before attending the course (prior_knowledge; mean = 1.75, standard deviation [SD] = 1.13) and a relatively high competence gain (GEKO; mean = 3.66, SD = 0.56). The course also succeeded in raising interest in the topics and competencies covered (interest; mean = 3.88, SD = 1.02). In particular, the acquisition of competences in teamwork received the highest rating and testified to the effectiveness of digital collaborations (mean = 3.94, SD = 1.12). The evaluation of the KIM questionnaire showed that the course participants had a high intrinsic motivation and the experience of competence on average (KIM; mean = 3.37, SD = 0.49). Regarding the usability, the evaluation revealed that students found working with BioLabSim enjoyable and the simulations were rated overall as user-friendly (usability; mean = 3.67, SD = 0.55). The total score rating of the course also indicated that the students enjoyed the course (rating; mean= 3.58, SD = 0.43). (Fig. 4). Overall, the feedback demonstrated that the Jupyter Notebook-based course led to a generally positive learning experience.

Conclusions

We have developed a model ensemble to simulate biotechnological relevant experiments for microorganisms (*silvio*). These experiments were combined in an instructional Jupyter Notebook to teach selected data analysis steps in a recombinant gene expression project. By adding new models to *silvio*, new biotechnological experiments can be simulated and converted into instructional, motivating data analysis projects in the form of Jupyter Notebooks. Our approach decouples the computational data analysis from often lengthy, expensive, and sometimes unavailable experiments. We stress that our simulations are not meant to replace the physical experience of conducting the experiments in the laboratory, but rather to support analytical lectures by providing practice in data handling and evaluation. Finally, our survey documented the positive learning experience by the students.

Ethics statement

The study was conducted according to institutional teaching guidelines of the Center for Learning Services, RWTH Aachen University. No sensitive personal data were collected and no medical research with human subjects was conducted.

SUPPLEMENTAL MATERIAL

Supplemental material is available online only.

SUPPLEMENTAL FILE 1, PDF file, 0.3 MB.

ACKNOWLEDGMENTS

We thank Frank Eiden, Jonathan Sturm, Lars Lauterbach, and Paul Cordero for teaching opportunities and support.

U.W.L. received funding from the Excellence Initiative of the German federal and state governments (DE-82:ETS488). This work is funded by the Deutsche Forschungsgemeinschaft (German Research Foundation) under Germany's Excellence Strategy, Exzellenzcluster 2186, The Fuel Science Center (ID 390919832).

We declare that no conflict of interest exist and that the funders had no role in the course material and setup.

REFERENCES

1. Oliveira AL. 2019. Biotechnology, big data and artificial intelligence. Biotechnol J 14:1800613. https://doi.org/10.1002/biot.201800613.
2. Liebal UW, Phan ANT, Sudhakar M, Raman K, Blank LM. 2020. Machine learning applications for mass spectrometry-based metabolomics. Metabolites 10:243. https://doi.org/10.3390/metabo10060243.
3. Wehrs M, de Beaumont-Felt A, Goranov A, Harrigan P, de Kok S, Lieder S, Vallandingham J, Tyner K. 2020. You get what you screen for: on the value of fermentation characterization in high-throughput strain improvements in industrial settings. J Ind Microbiol Biotechnol 47:913–927. https://doi.org/10.1007/s10295-020-02295-3.
4. Fang X, Lloyd CJ, Palsson BO. 2020. Reconstructing organisms in silico: genome-scale models and their emerging applications. Nat Rev Microbiol 18:731–743. https://doi.org/10.1038/s41579-020-00440-4.
5. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, Burovski E, Peterson P, Weckesser W, Bright J, Van Der Walt SJ. 2020. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature Methods 17:261–272. https://doi.org/10.1038/s41592-019-0686-2.
6. Kluyver T, Ragan-Kelley B, Pérez F, Granger B, Bussonnier M, Frederic J, Kelley K, Hamrick J, Grout J, Corlay S, Ivanov P, Avila D, Abdalla S, Willing C, Jupyter Development Team. 2016. Jupyter notebooks – a publishing format for reproducible computational workflows, p 87–90. Proceedings of the 20th International Conference on Electronic Publishing. IOS Press. https://doi.org/10.3233/978-1-61499-649-1-87.
7. Afgan E, Baker D, Batut B, van den Beek M, Bouvier D, Cech M, Chilton J, Clements D, Coraor N, Grüning BA, Guerler A, Hillman-Jackson J, Hiltemann S, Jalili V, Rasche H, Soranzo N, Goecks J, Taylor J, Nekrutenko A, Blankenberg D. 2018. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update. Nucleic Acids Res 46:W537–W544. https://doi.org/10.1093/nar/gky379.
8. Arkin AP, Cottingham RW, Henry CS, Harris NL, Stevens RL, Maslov S, Dehal P, Ware D, Perez F, Canon S, Sneddon MW, Henderson ML, Riehl WJ, Murphy-Olson D, Chan SY, Kamimura RT, Kumari S, Drake MM, Brettin TS, Glass EM, Chivian D, Gunter D, Weston DJ, Allen BH, Baumohl J, Best AA, Bowen B, Brenner SE, Bun CC, Chandonia J-M, Chia J-M, Colasanti R, Conrad N, Davis JJ, Davison BH, DeJongh M, Devoid S, Dietrich E, Dubchak I, Edirisinghe JN, Fang G, Faria JP, Frybarger PM, Gerlach W, Gerstein M, Greiner A, Gurtowski J, Haun HL, He F, Jain R, et al. 2018. KBase: the United States Department of Energy Systems Biology Knowledgebase. Nat Biotechnol 36:566–569. https://doi.org/10.1038/nbt.4163.
9. Berthold MR, Cebreon N, Dill F, Gabriel TR, Kötter T, Meinl T, Ohl P, Thiel K, Wiswedel B. 2009. KNIME - the Konstanz information miner: version 2.0 and beyond. ACM SIGKDD Explor Newsl 11:26–31. https://doi.org/10.1145/1656274.1656280.
10. Perkel JM. 2018. Why Jupyter is data scientists' computational notebook of choice. Nature 563:145–146. https://doi.org/10.1038/d41586-018-07196-1.
11. Mendez KM, Pritchard L, Reinke SN, Broadhurst DI. 2019. Toward collaborative open data science in metabolomics using Jupyter Notebooks and cloud computing. Metabolomics 15:125. https://doi.org/10.1007/s11306-019-1588-0.
12. Birkel GW, Ghosh A, Kumar VS, Weaver D, Ando D, Backman TWH, Arkin AP, Keasling JD, Martín HG. 2017. The JBEI quantitative metabolic modeling library (jQMM): a python library for modeling microbial metabolism. BMC Bioinformatics 18:205. https://doi.org/10.1186/s12859-017-1615-y.
13. Cardoso JGR, Jensen K, Lieven C, Lærke Hansen AS, Galkina S, Beber M, Özdemir E, Herrgård MJ, Redestig H, Sonnenschein N. 2018. Cameo: a Python library for computer aided metabolic engineering and optimization of cell factories. ACS Synth Biol 7:1163–1166. https://doi.org/10.1021/acssynbio.7b00423.
14. Liebal UW, Kobbing S, Netze L, Schweidtmann AM, Mitsos A, Blank LM. 2021. Insight to gene expression from promoter libraries with the machine learning workflow Exp2Ipynb. Front Bioinform 1:74728. https://doi.org/10.3389/fbinf.2021.747428.
15. Fahnert B. 2019. Be prepared: learning for the future. FEMS Microbiol Lett 366:fnz200. https://doi.org/10.1093/femsle/fnz200.
16. Compeau P. 2019. Establishing a computational biology flipped classroom. PLoS Comput Biol 15:e1006764. https://doi.org/10.1371/journal.pcbi.1006764.
17. Li Z, Liu B, Li SH-J, King CG, Gitai Z, Wingreen NS. 2020. Modeling microbial metabolic trade-offs in a chemostat. PLoS Comput Biol 16:e1008156. https://doi.org/10.1371/journal.pcbi.1008156.
18. Dillon MR, Bolyen E, Adamov A, Belk A, Borsom E, Burcham Z, Debelius JW, Deel H, Emmons A, Estaki M, Herman C, Keefe CR, Morton JT, Oliveira RRM, Sanchez A, Simard A, Vázquez-Baeza Y, Ziemski M, Miwa HE, Kerere TA, Coote C, Bonneau R, Knight R, Oliveira G, Gopalasingam P, Kaehler BD, Cope EK, Metcalf JL, Robeson Ii MS, Bokulich NA, Caporaso JG. 2021. Experiences and lessons learned from two virtual, hands-on microbiome bioinformatics workshops. PLoS Comput Biol 17:e1009056. https://doi.org/10.1371/journal.pcbi.1009056.
19. Brown NCC, Wilson G. 2018. Ten quick tips for teaching programming. PLoS Comput Biol 14:e1006023. https://doi.org/10.1371/journal.pcbi.1006023.
20. Koivisto J, Hamari J. 2019. The rise of motivational information systems: a review of gamification research. Int J Infect Manage 45:191–210. https://doi.org/10.1016/j.ijinfomgt.2018.10.013.
21. McEvoy JP. 2017. Interactive problem-solving sessions in an

introductory bioscience course engaged students and gave them feedback, but did not increase their exam scores. FEMS Microbiol Lett 364:182. https://doi.org/10.1093/femsle/fnx182.
22. Lubbock ALR, Lopez CF. 2021. Programmatic modeling for biological systems. Curr Opin Syst Biol 27:100343. https://doi.org/10.1016/j.coisb.2021.05.004.
23. Suárez-García A, Arce-Fariña E, Álvarez Hernández M, Fernández-Gavilanes M. 2021. Teaching structural analysis theory with Jupyter Notebooks. Comput Appl Eng Educ 29:1257–1266. https://doi.org/10.1002/cae.22383.
24. Seddighi M, Allanson D, Rothwell G, Takrouri K. 2020. Study on the use of a combination of IPython Notebook and an industry-standard package in educating a CFD course. Comput Appl Eng Educ 28:952–964. https://doi.org/10.1002/cae.22273.
25. Rule A, Birmingham A, Zuniga C, Altintas I, Huang S-C, Knight R, Moshiri N, Nguyen MH, Rosenthal SB, Pérez F, Rose PW. 2019. Ten simple rules for writing and sharing computational analyses in Jupyter Notebooks. PLoS Comput Biol 15:e1007007. https://doi.org/10.1371/journal.pcbi.1007007.
26. Chuang LY, Cheng YH, Yang CH. 2013. Specific primer design for the polymerase chain reaction. Biotechnol Lett 35:1541–1549. https://doi.org/10.1007/s10529-013-1249-8.
27. Paget M. 2015. Bacterial sigma factors and anti-sigma factors: structure, function and distribution. Biomolecules 5:1245–1265. https://doi.org/10.3390/biom5031245.
28. Wilde M, Bätz K, Kovaleva A, Urhahne D. 2009. Überprüfung einer Kurzskala intrinsischer Motivation (KIM) [german]. Zeitschrift Didakt Naturwissenschaften 15:31–45.
29. Paechter M, Maier B, Grabensberger E. 2007. Evaluation medienbasierter Lehre mittels der Einschätzung des Kompetenzerwerbs. Zeitschrift Medien 19:68–75. https://doi.org/10.1026/1617-6383.19.2.68.