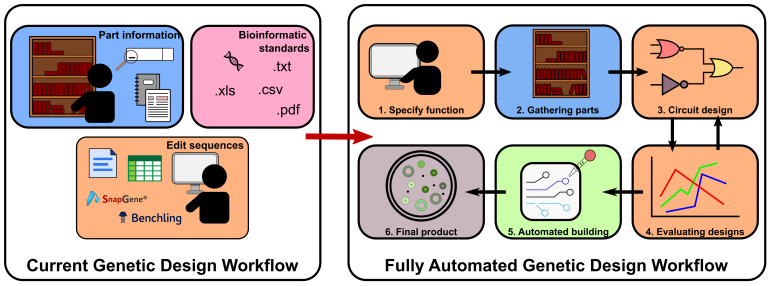
Graphical Abstract

Engineering Genetic Circuits: Advancements in Genetic Design Automation Tools and Standards for Synthetic Biology

Lukas Buecherl, Chris J. Myers



Highlights

Engineering Genetic Circuits: Advancements in Genetic Design Automation Tools and Standards for Synthetic Biology

Lukas Buecherl, Chris J. Myers

- Synthetic biology aims to apply engineering principles in biology
- Insufficient part libraries, standards, and software tools are challenges to the field
- Working on the challenges will move genetic design automation forward
- Ultimate goal is a fully automated workflow for design, build, and test of genetic circuits

Engineering Genetic Circuits: Advancements in Genetic Design Automation Tools and Standards for Synthetic Biology

Lukas Buecherl[a], Chris J. Myers[b]

[a]*Biomedical Engineering Program, University of Colorado Boulder, 1111 Engineering Drive, Boulder, 80309, Colorado, United States*
[b]*Department of Electrical, Computer, and Energy Engineering, University of Colorado Boulder, 425 UCB, Boulder, 80309, Colorado, United States*

Abstract

Synthetic biology is a field at the intersection of biology and engineering. Inspired by engineering principles, researchers use defined parts to build functionally defined biological circuits. *Genetic design automation* allows scientists to design, model, and analyze their genetic circuits *in silico* before building them in the lab, saving time and resources in the process. Establishing synthetic biology's future is dependent on genetic design automation, since the computational approach opens the field to a broad, interdisciplinary community. However, challenges with part libraries, standards, and software tools are currently stalling progress in the field. This review first covers recent advancements in genetic design automation, followed by an assessment of the challenges ahead, and a proposed automated genetic design workflow for the future.

Keywords: synthetic biology, genetic design automation, standards, genetic circuits, genetic part libraries

1. Introduction

Synthetic biology (SynBio) is a research field that is integrating biology with engineering principles, such as standards (reusable parts and design information), abstraction (analysis of behavior at a higher level), and decoupling (separation of the design process from the construction process) [1]. The field has a variety of applications including protein engineering, metabolic

engineering, artificial cell engineering [2], vaccine development, molecular diagnostics, cell-based therapeutics [3], and bio-manufacturing of food, biochemicals, and pharmaceuticals [4]. This review focuses on one aspect of the field, the design of *genetic circuits*, collections of biological components that perform a desired function within a cell.

Early examples of genetic circuits in SynBio include the construction of the genetic toggle switch [5] and the repressilator [6]. These early genetic circuits were designed in the laboratory using a largely manual and experimental process. However, the field of SynBio promises to deliver systematic, model-based design and automated construction of genetic circuits [7, 8, 9, 10, 11].

Circuits can be constructed on the transcriptional level by controlling gene expression, on the post-transcriptional level using metabolic perceptrons [12] or protein interactions [13], and in cell-free environments. This review focuses on transcriptional circuit design and *genetic design automation* (GDA) methods and tools that support the computational design, modeling, and analysis of these genetic circuits. More information on transcriptional and post-transcriptional circuits can be found in [14], while current cell-free SynBio platforms are reviewed in [15].

Despite recent advancements in GDA, a gap between computational and laboratory SynBio remains. Limited part reuse due to a lack of acceptance of standards and library part characterization processes within the community stalls the progress of GDA [16, 17, 18, 19]. This article first reviews the recent advancements in design efforts, followed by an assessment of challenges ahead, and a proposed workflow for the future.

2. Advancements

Applying principles from *electronic design automation* (EDA) to SynBio resulted in advances in GDA [20]. This work defines GDA as the development of an automated, seamless computational workflow, that aids scientist from an idea to a final design ready to be built in a laboratory. This includes the automated selection of biological parts, their combination and assembly within a DNA plasmid, and an evaluation of the final result. GDA enables the design of more complex circuits, such as the genetic circuit for programming the bacterium *Bacteroides thetaiotaomicron* for drug delivery in the human gut [21]. Figure 1 shows the three cornerstones of GDA: (a) software tools, (b) standards, and (c) part libraries.

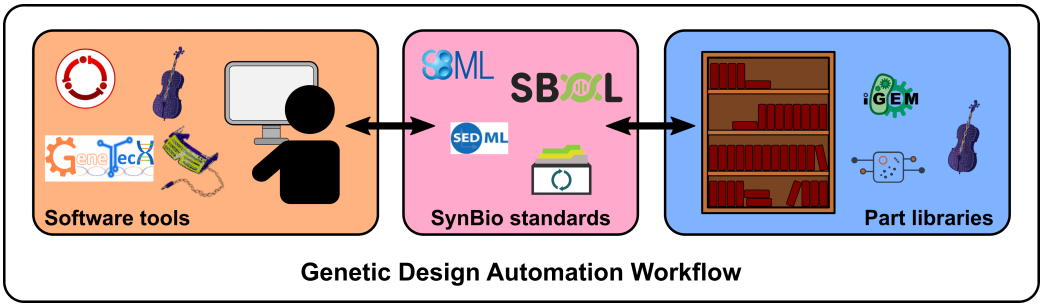


Figure 1: Software tools, SynBio standards, and part libraries as the cornerstones of GDA. Researchers can use standard-enabled software tools to seamlessly design genetic circuits using characterized part libraries encoded using data standards.

Part Libraries enable a reproducible and modular engineering approach to design. Community developed part repositories store characterized genetic parts, facilitating the exchange of genetic design information. These parts define a specific DNA sequence with known functionally, for example a *promoter* that is a specified DNA sequence that acts as a starting site for DNA transcription. However, the characterization of these parts happens in isolation and in a specified environment. Introducing those parts to a new environmental, cellular, or genetic context results in variation of the parts performance [22, 23]. Therefore, creating ones own, reliable parts requires substantial time and resources. Accessing information of previously used parts allows researchers to efficiently design and build new genetic circuits. These part libraries can then be shared using data repositories, such as the SynBioHub data repository [24] or the *Joint Bio-Energy Institute Inventory of Composable Elements* (JBEI-ICE) [25]. Sequences can also be found in public sequence databases, such as those available from the *National Center for Biotechnology Information* (NCBI) [26], or from plasmid repositories, such as *Addgene* [27]. Examples of part libraries include the *Standard European Vector Architecture* (SEVA) database [28] for plasmid vectors, the *International Genetically Engineered Machines* (iGEM) Registry of Standard Biological Parts [29, 30], and the collection of parts used by the Cello project [31, 32, 33].

The iGEM part library is an important resource used by the iGEM competition [34], in which interdisciplinary student teams design, build, and test genetic circuit designs. The teams both use existing and contribute new registry parts. iGEM parts have been used, for example, to build an arsenic

biosensor [35] and to control bacterial cellulose production [36].

The Cello library used in the Cello project consists of 12 orthogonal *Tet repressors* that drive cassettes including promoters, coding sequences, and terminators to function as NOT/NOR gates. Since the behavior of the underlying parts has been well-characterized, they can be connected to build a predictable genetic circuit. The well-characterized parts in the Cello library allowed the successful construction of a NOR gate in *Pseudomonas putida* utilizing Cello's parts tailored to *Escherichia coli* [37], demonstrating the portability of the design to a different host.

Standards are a foundation of every engineering discipline and can be found in architecture, electronics, mechanical design, and chemical synthesis [38]. They are needed to describe properties precisely in order to enable reproducibility and predictability. Furthermore, they facilitate communication between different research groups and software tools.

For SynBio, the community has both leveraged existing standards, as well as created new ones specific to this domain. For example, bioinformatics standards, such as GenBank [39] and FASTA [40], are often leveraged to represent DNA sequences. BioBricks [41] and MoClo [42] are standards developed for DNA assembly and cloning, while SEVA [28] has been developed for plasmid vectors. In order to represent hierarchical genetic designs composed of DNA and other types of parts (RNA, proteins, small molecules, etc.), the SynBio community has developed the *synthetic biology open language* (SBOL) [43, 44, 45, 46]. The community has also developed SBOL Visual to visualize genetic designs using a standard set of glyphs [47, 48]. For modeling, the community widely uses the *systems biology markup language* (SBML) [49, 50]. The *simulation experiment description markup language* (SED-ML) [51] encodes simulations performed on these models, while *COMBINE archives* [52] packages design and modeling information together.

Software tools enable the design of genetic circuits in graphical user interfaces opening the field to researchers without expertise in laboratory work. These tools automate the design process by leveraging standards to access libraries of characterized parts. Table 1 shows a non-comprehensive overview of software tools for GDA. In this review, software tools for GDA are defined as those specifically developed for the design and modeling of genetic circuits and their sequences. This table includes their application domain, standard and part library support, and availability of support and documentation. The first group of tools are developed by academic researchers, while the second group are developed by commercial vendors. Furthermore, software

tools can be separated into sequence editors that constrain the user to work directly on the DNA sequence and *high-level* (HL) design and modeling tools that incorporate abstraction into the design process allowing researchers to work with parts instead of directly on the sequence.

While most tools only implement a part of the GDA workflow shown in Figure 1, there are two high-level design tools that support the entire workflow, leverage SynBio standards, and connect to part library repositories. First, iBioSim [58, 59, 60] is an actively developed, open-source, academic tool, for the model-based design of genetic circuits. Researchers can create and edit hierarchical genetic designs represented using SBOL and visualized with SBOL Visual, automatically generate computational models represented in SBML for simulation [73, 74], and fetch parts and store design information in SynBioHub. Xiang et al. used iBioSim to design and analyze a genetic circuit for the detection of early biological markers of Lung Cancer [75]. Second, Cello [31, 32, 33] specifies genetic circuit designs using the *hardware description language* (HDL) Verilog, and it maps them to parts from the Cello gate library. The Cello tool was tested on 60 genetic circuits in *Escherichia coli* [33]. Example circuits designed using Cello involve a drug delivery circuit for *Bacteroides thetaiotaomicron* [21] and the before mentioned NOR gate for *Pseudomonas putida* [37].

3. Challenges

While there have been significant advancements in GDA, the automated design of genetic circuits still faces many challenges. There is still a significant gap between the potential computational GDA workflows shown in Figure 1 and the actual laboratory practices commonly used by wet-lab synthetic biologists. As shown in Figure 2, there is currently limited utilization of part libraries, standards, and software tools in the laboratory workflow.

While part libraries like the iGEM and Cello libraries are valuable additions to SynBio, their current state is insufficient to be a reliable backbone for design automation. Parts submitted by the participating teams of the competition are not standardized and often only tested in defined lab conditions. A recent analysis by Mante et al. of the iGEM library identified the use of free text, insufficient part provenance, part duplication, lack of part removal, and insufficient continuous curation complicate its use as a library of reliable parts [77]. Indeed, only a small core set of well-defined and characterized parts is usually reused [16]. Other libraries like the Cello library are

Table 1: Non-comprehensive list of software tools specifically developed for genetic circuit design. The first group are from academia, and the second group are from industry.

Tool	Application	Standards	Part Library Repositories	Supported / Documented
ApE [53]	Sequence Editor	GenBank	-	Yes / Yes
Cello [31, 32, 33]	HL Design	SBOL	SynBioHub	Yes / Yes
Device Editor [54]	Sequence Editor	Genbank, FASTA SBOL, SBOL Visual	JBEI-ICE	Yes / Yes
Eugene [55]	Sequence Editor	FASTA, GenBank, SBOL	-	No / Yes
GeneTech [56]	HL Design	SBOL, SBOL Visual	-	Yes / Yes
GenoCAD [57]	Sequence Editor	GenBank, FASTA, SBML	-	No / Yes
iBioSim [58, 59, 60]	HL Design & Modeling	SBOL, SBOL Visual, SBML, SED-ML, COMBINE Archive, GenBank, FASTA	SynBioHub	Yes / Yes
j5 [61]	Sequence Editor	SBOL, SBOL Visual, FASTA, GenBank	JBEI-ICE	Yes / Yes
Mosec [62]	Modeling	SBML, CellML, GenBank, SBOL	-	No / No
Proto BioCompiler [63]	HL Design & Modeling	SBOL, SBML, CellML	-	No / No
SBOLCanvas [64]	Sequence Editor	SBOL, SBOL Visual	SynBioHub	Yes/ Yes
SBROME [65]	HL Design	-	-	No / Yes
Tellurium [66, 67]	Modeling	COMBINE archive, SBML, SED-ML, CellML, SBOL	-	Yes/ Yes
Tinkercell [68, 69, 70]	HL Design & Modeling	SBML, SBOL, SBOL Visual	-	No / Yes
Benchling *https://www.benchling.com*	Sequence Editor	GenBank, FASTA	AddGene, iGEM, NCBI JBEI-ICE	Yes / Yes
Doulix *https://getstarted.doulix.com*	Sequence Editor	SBOL, SBOL Visual GenBank, FASTA	Doulix	Yes / Yes
Geneious *https://www.geneious.com*	Sequence Editor	GenBank, FASTA	NCBI	Yes / Yes
Genetic Constructor [71]	Sequence Editor	SBOL, SBOL Visual, GenBank	iGEM, NCBI	No / Yes
Visual GEC [72]	HL Design & Modeling	SBML	-	No / Yes
Genome Compiler *https://designer.genomecompiler.com*	Sequence Editor	GenBank, FASTA	Addgene iGEM, NCBI	No / No
OpenVectorEditor *https://github.com/TeselaGen/openVectorEditor*	Sequence Editor	GenBank, FASTA	-	Yes / Yes
Snapgene *https://www.snapgene.com*	Sequence Editor	GenBank, FASTA	NCBI	Yes / Yes

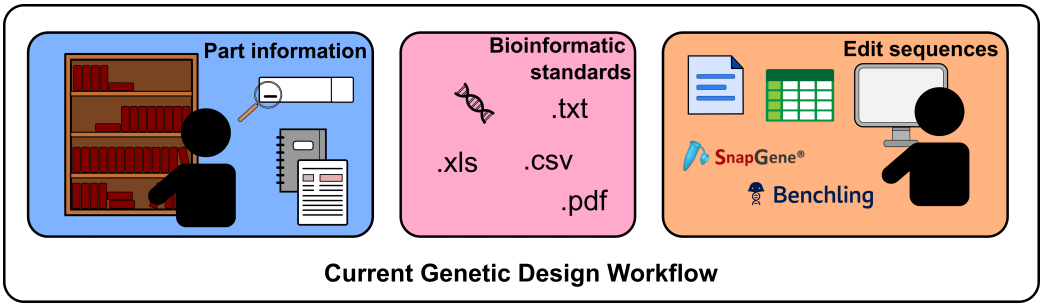


Figure 2: The current genetic design workflow in SynBio. There is no seamless integration between the elements. Part information is manually searched for in sequence databases and journal publications. The sequence information is often only available in bioinformatic standards, such FASTA or GenBank, or non-standard formats, such as spreadsheets, plain text, or PDF files [76]. Researchers do not use high-level GDA tools, but instead edit sequences directly using sequence editor GDA tools, spreadsheets, or text editors.

restricted due to its limited size and support of hosts. Overall, the limited usefulness of the current part registries sabotages the scalability of genetic designs, since researchers typically must manually search for part information, such as DNA sequences, either in sequence databases or in publications and their supplemental materials.

The SynBio community has been slow to develop and accept standards to encode genetic design information and part characterization data. Researchers predominantly work with bioinformatic standards for sequences without defined specifications, such as GenBank or FASTA, or represent their design information in non-standard formats, such as text files, spreadsheets, and PDF files, rather than using SynBio standards such as SBOL [76].

Finally, although software tools streamline the genetic design process, their usage in the community is limited. Researchers currently edit sequences directly using sequence editors, spreadsheets, or even text editors, instead of leveraging high-level design and modeling GDA software tools. As shown in Table 1, commercial tools that are still supported are sequence editors, although prior sequence editors both commercial and academic like the Genetic Constructor by Autodesk [71], Genome Compiler, and GenoCAD did not gain traction in the user community. Similarly, the high-level design tool Visual GEC [72] by Microsoft is now discontinued, possibly due to the limited pool of potential costumers. Further stalling the progress in the development of high-level design GDA software tools for industrial usage could be the limited awareness of the potential of such software tools, that are in

their early stages.

All high-level design and modeling tools and actively developed sequence editors using SBOL are being developed as academic open-source software. These tools often lack sufficient and sustained support and documentation. Additionally, the interfaces for these tools are often not intuitive to their targeted end-users. An inspection of papers citing the use of these tools finds that most genetic circuits designed are only constructed *in silico* to validate the software without real-world application. Cello was used to construct 60 circuits *in vivo* in *Escherichia coli*, but again only to validate the results of their tool [33]. Portability of genetic designs is an additional issue, since most genetic parts and GDA design tools are tailored around *E. Coli*. Although well-characterized parts help in addressing this issue, as shown in the previous example of a NOR gate for *Pseudomonas putida* [37]. Most academic tools are used within the research group that developed the software with little uptake from other researchers in the community. Finally, miscommunication between computational and laboratory-based SynBio leads to the development of features gratuitous for the laboratory workflow while beneficial or even necessary additions are overlooked by the software developers. However, recently, learning from the previous failure of other high-level design tools, start-up companies like Lattice Automation (`https://latticeautomation.com`) and Asimov (`https://asimov.com`) are trying to fill this gap by providing customized software and software solutions for SynBio laboratories instead of selling standalone software applications.

4. Discussion

Figure 3 shows a fully automated genetic design workflow. In this workflow, the researcher simply specifies a desired function in a GDA software tool that automatically identifies the parts needed to construct a robust circuit design. After *in silico* analysis to evaluate alternative designs, the selected design is constructed using microfluidic devices [78] and lab automation [79] provided by companies like Opentrons [80] (`https://opentrons.com`), for example. While OpenTrons and microfluidics allow some automation in the workflow, more complex robotics are required for a complete hands-free operation. Full automation, however, can be achieved by outsourcing the manufacturing to highly automated laboratories like Strateos (`https://strateos.com`) and the Edinburgh genome foundry (`https://www.ed.ac.uk/biology/research/facilities/edinburgh-genome-foundry`). Af-

ter construction and testing, the final product is produced without the need for special expertise or resources by the original designer. Yet, in order to create this fully automated genetic design workflow, allowing SynBio to fulfill its promise as a systematic, engineering-based, and automated design process, the challenges described above must be overcome.

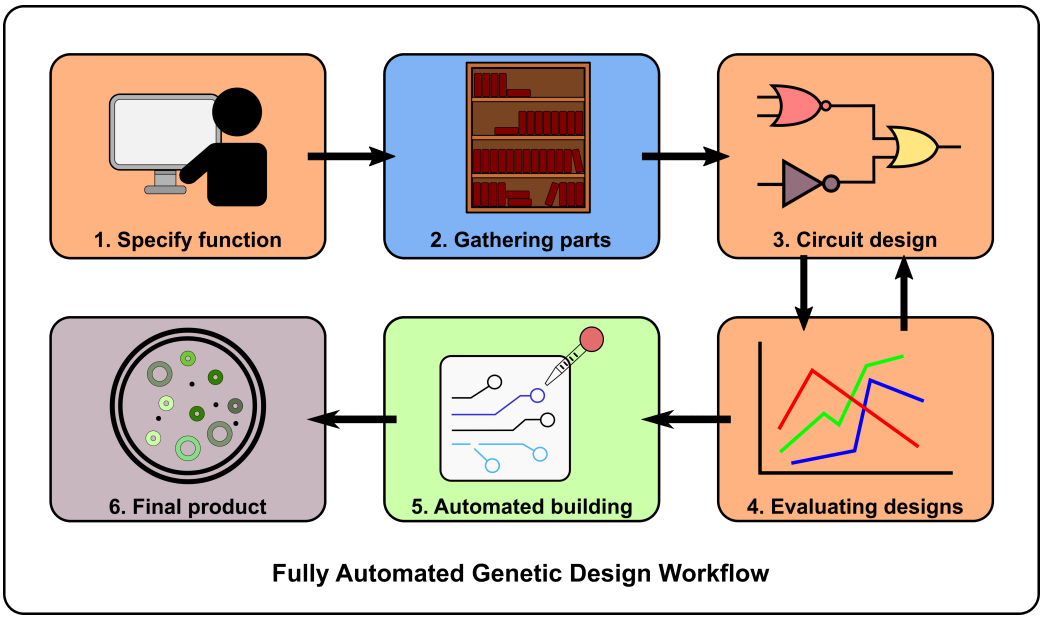


Figure 3: Fully automated genetic design workflow. Researchers specify their function using high-level design GDA tools. The software is integrated in the workflow and automatically finds applicable parts and devices stored in repositories. After analysis of the designs, the most suitable is selected and built in the lab using lab automation like microfluidics and robotics to create the final product.

First, well characterized genetic part libraries must be developed and shared in standard formats via public repositories. Since genetic parts behave differently in different genetic contexts, a broader and standardized characterization would increase the reusability and predictability of genetic parts. Additionally, the part repositories need to be carefully curated to ensure quality. Improvements in characterization methods are also needed using high-throughput screening methods, such as described in [81, 82]. These new methods need to enable efficient collection and analysis of vast amounts of data collected leading to improved accuracy of part characterization. The use of modern standards is also not a complete solution, since researchers

must also agree on what minimum information must be encoded to reproduce and share genetic designs [77], and how to measure and what units to use to characterize the performance of parts, devices, and systems [38]. The creation of a minimum information standard and standardization of measurement protocols could enable the definition of part data sheets as suggested by Canton et al. [83].

Second, there needs to be an increase in the acceptance of standards in the community to improve data sharing and reproducibility of published results [84]. Not only laboratory procedures need to be standardized, but also standards for data sharing, visualization, and mathematical modeling in order to allow easy and efficient communication between different research groups [38]. Community developed standards, such as SBOL, SBOL Visual, and SBML, should be preferred over bioinformatic standards and non-standard formats to facilitate the sharing of complete design information among collaborators. Part libraries should be encoded using these standards, and software tools must support them to increase the ability for laboratory-based scientists to share their data and computational-based scientists to develop integrated software workflows. Finally, existing standards should continue to evolve and new standards developed, as needed, through open discourse between computational and laboratory-based researchers, as well as academic and industrial representatives.

Finally, software tools need to continue to be developed and adopted. The *in silico* analysis of genetic designs using software tools has the potential to enable efficient design space exploration. Most of the currently available GDA software tools are developed in academia. Although they are readily available as open-source software, they are often not intuitive, lack suitable interfaces and documentation, and are not consistently supported. Commercial software needs to be developed for high-level design and modeling that supports modern community developed standards and connects to public shared part repositories. Since the behavior of genetic circuits is stochastic and noisy [85], these software tools must employ stochastic analysis methods to evaluate design alternatives [73, 86]. Additionally, GDA software tools should assist the designer in finding the best parts for the genetic design from well characterized part libraries. Indeed, a genetic circuit's behavior can be adjusted by carefully considering the selection of genetic parts [87]. Furthermore, current design tools do not consider the circuit host interaction. While Cello provides toxicity data, a full analysis of the circuit host interaction is not readily available and needs to be taken into account in the

future. Better computational predictability of a circuit's toxicity is necessary to further improve design automation. Recent work presents the impact of using computational optimization to improve genetic design [88, 89, 90]. Overall, and most importantly, the gap between computational and laboratory SynBio needs to be closed by useful and improved computational tools that facilitate the design process and automate the workflow, saving time and resources making genetic design readily available.

Acknowledgements

The authors of this work are supported by the National Science Foundation under Grant No. 1856740. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agency.

References

[1] A. S. Khalil, J. J. Collins, Synthetic biology: Applications come of age, Nature Reviews Genetics 11 (2010) 367–379. doi:10.1038/nrg2775.

[2] Y. Lu, Cell-free synthetic biology: Engineering in an open world, Synthetic and Systems Biotechnology 2 (2017) 23–27. doi:10.1016/j.synbio.2017.02.003.

[3] X. Tan, J. H. Letendre, J. J. Collins, W. W. Wong, Synthetic biology in the clinic: Engineering vaccines, diagnostics, and therapeutics, Cell 184 (2021) 881–898. doi:10.1016/j.cell.2021.01.017.

[4] T. Shi, P. Han, C. You, Y.-H. P. J. Zhang, An in vitro synthetic biology platform for emerging industrial biomanufacturing: Bottom-up pathway design, Synthetic and Systems Biotechnology 3 (2018) 186–195. doi:10.1016/j.synbio.2018.05.002.

[5] T. S. Gardner, C. R. Cantor, J. J. Collins, Construction of a genetic toggle switch in Escherichia coli, Nature 403 (2000) 339–342. doi:10.1038/35002131.

[6] M. B. Elowitz, S. Leibler, A synthetic oscillatory network of transcriptional regulators, Nature 403 (2000) 335–338. doi:10.1038/35002125.

[7] E. Appleton, C. Madsen, N. Roehner, D. Densmore, Design Automation in Synthetic Biology, Cold Spring Harbor Perspectives in Biology 9 (2017) a023978. doi:10.1101/cshperspect.a023978.

[8] C. J. Myers, Computational Synthetic Biology: Progress and the Road Ahead, IEEE Transactions on Multi-Scale Computing Systems 1 (2015) 19–32. doi:10.1109/TMSCS.2015.2478442.

[9] M. W. Lux, B. W. Bramlett, D. A. Ball, J. Peccoud, Genetic design automation: Engineering fantasy or scientific renewal?, Trends in Biotechnology 30 (2012) 120–126. doi:10.1016/j.tibtech.2011.09.001.

[10] A. L. Slusarczyk, A. Lin, R. Weiss, Foundations for the design and implementation of synthetic genetic circuits, Nature Reviews Genetics 13 (2012) 406–420. doi:10.1038/nrg3227.

[11] S. Rollié, M. Mangold, K. Sundmacher, Designing biological systems: Systems Engineering meets Synthetic Biology, Chemical Engineering Science 69 (2012) 1–29. doi:10.1016/j.ces.2011.10.068.

[12] A. Pandi, M. Koch, P. L. Voyvodic, P. Soudier, J. Bonnet, M. Kushwaha, J.-L. Faulon, Metabolic perceptrons for neural computing in biological systems, Nature Communications 10 (2019) 3880. doi:10.1038/s41467-019-11889-0.

[13] D. Mishra, T. Bepler, B. Teague, B. Berger, J. Broach, R. Weiss, An engineered protein-phosphorylation toggle network with implications for endogenous network discovery, Science 373 (2021) eaav0780. doi:10.1126/science.aav0780.

[14] M. A. English, R. V. Gayet, J. J. Collins, Designing Biological Circuits: Synthetic Biology Within the Operon Model and Beyond, Annual Review of Biochemistry 90 (2021) 221–244. doi:10.1146/annurev-biochem-013118-111914.

[15] D. Jeong, M. Klocke, S. Agarwal, J. Kim, S. Choi, E. Franco, J. Kim, Cell-Free Synthetic Biology Platform for Engineering Synthetic Biological Circuits and Systems, Methods and Protocols 2 (2019) 39. doi:10.3390/mps2020039.

[16] P. Siyari, B. Dilkina, C. Dovrolis, Evolution of Hierarchical Structure and Reuse in iGEM Synthetic DNA Sequences, in: J. M. F. Rodrigues, P. J. S. Cardoso, J. Monteiro, R. Lam, V. V. Krzhizhanovskaya, M. H. Lees, J. J. Dongarra, P. M. Sloot (Eds.), Computational Science – ICCS 2019, Springer International Publishing, Cham, 2019, pp. 468–482.

[17] R. G. Curty, K. Crowston, A. Specht, B. W. Grant, E. D. Dalton, Attitudes and norms affecting scientists' data reuse, PLOS ONE 12 (2017) e0189288. doi:10.1371/journal.pone.0189288.

[18] P. E. Bourne, J. R. Lorsch, E. D. Green, Perspective: Sustaining the big-data ecosystem, Nature 527 (2015) S16–S17. doi:10.1038/527S16a.

[19] C. Tenopir, S. Allard, K. Douglass, A. U. Aydinoglu, L. Wu, E. Read, M. Manoff, M. Frame, Data Sharing by Scientists: Practices and Perceptions, PLoS ONE 6 (2011) e21101. doi:10.1371/journal.pone.0021101.

[20] T. Schladt, N. Engelmann, E. Kubaczka, C. Hochberger, H. Koeppl, Automated Design of Robust Genetic Circuits: Structural Variants and Parameter Uncertainty, ACS Synthetic Biology 10 (2021) 3316–3329. doi:10.1021/acssynbio.1c00193.

[21] M. Taketani, J. Zhang, S. Zhang, A. J. Triassi, Y.-J. Huang, L. G. Griffith, C. A. Voigt, Genetic circuit design automation for the gut resident species Bacteroides thetaiotaomicron, Nature Biotechnology 38 (2020) 1001–1001. doi:10.1038/s41587-020-0545-9.

[22] T. Nguyen, T. S. Jones, P. Fontanarrosa, J. V. Mante, Z. Zundel, D. Densmore, C. Myers, Design of Asynchronous Genetic Circuits, Proceedings of the IEEE 107 (2019) 1356–1368. doi:10.1109/JPROC.2019.2916057.

[23] P. Vaidyanathan, B. S. Der, S. Bhatia, N. Roehner, R. Silva, C. A. Voigt, D. Densmore, A Framework for Genetic Logic Synthesis, Proceedings of the IEEE 103 (2015) 2196–2207. doi:10.1109/JPROC.2015.2443832.

[24] J. A. McLaughlin, C. J. Myers, Z. Zundel, G. Mısırlı, M. Zhang, I. D. Ofiteru, A. Goñi-Moreno, A. Wipat, SynBioHub: A Standards-Enabled Design Repository for Synthetic Biology, ACS Synthetic Biology 7 (2018) 682–688. doi:10.1021/acssynbio.7b00403.

[25] T. S. Ham, Z. Dmytriv, H. Plahar, J. Chen, N. J. Hillson, J. D. Keasling, Design, implementation and practice of JBEI-ICE: An open source biological part registry platform and tools, Nucleic Acids Research 40 (2012) e141–e141. doi:10.1093/nar/gks531.

[26] E. W. Sayers, J. Beck, E. E. Bolton, D. Bourexis, J. R. Brister, K. Canese, D. C. Comeau, K. Funk, S. Kim, W. Klimke, A. Marchler-Bauer, M. Landrum, S. Lathrop, Z. Lu, T. L. Madden, N. O'Leary, L. Phan, S. H. Rangwala, V. A. Schneider, Y. Skripchenko, J. Wang, J. Ye, B. W. Trawick, K. D. Pruitt, S. T. Sherry, Database resources of the National Center for Biotechnology Information, Nucleic Acids Research 49 (2021) D10–D17. doi:10.1093/nar/gkaa892.

[27] J. Kamens, The Addgene repository: An international nonprofit plasmid and data resource, Nucleic Acids Research 43 (2015) D1152–D1157. doi:10.1093/nar/gku893.

[28] R. Silva-Rocha, E. Martínez-García, B. Calles, M. Chavarría, A. Arce-Rodríguez, A. de las Heras, A. D. Páez-Espino, G. Durante-Rodríguez, J. Kim, P. I. Nikel, R. Platero, V. de Lorenzo, The Standard European Vector Architecture (SEVA): A coherent platform for the analysis and deployment of complex prokaryotic phenotypes, Nucleic Acids Research 41 (2013) D666–D675. doi:10.1093/nar/gks1119.

[29] C. Vilanova, M. Porcar, iGEM 2.0—refoundations for engineering biology, Nature Biotechnology 32 (2014) 420–424. doi:10.1038/nbt.2899.

[30] C. D. Smolke, Building outside of the box: iGEM and the BioBricks Foundation, nature biotechnology 27 (2009) 4.

[31] T. S. Jones, S. M. D. Oliveira, C. J. Myers, C. A. Voigt, D. Densmore, Genetic circuit design automation with Cello 2.0, Nature Protocols (2022). doi:10.1038/s41596-021-00675-2.

[32] Y. Chen, S. Zhang, E. M. Young, T. S. Jones, D. Densmore, C. A. Voigt, Genetic circuit design automation for yeast, Nature Microbiology 5 (2020) 1349–1360. doi:10.1038/s41564-020-0757-2, ** This paper introduces Cello 2.0, a design automation software to facilitate engineers design of genetic by from high-level specifications to DNA sequences.

This updated version of Cello can be used to design genetic circuits for yeast.

[33] A. A. K. Nielsen, B. S. Der, J. Shin, P. Vaidyanathan, V. Paralanov, E. A. Strychalski, D. Ross, D. Densmore, C. A. Voigt, Genetic circuit design automation, Science 352 (2016) aac7341–aac7341. doi:10.1126/science.aac7341.

[34] J. Brown, The iGEM competition: Building with biology, IET Synth. Biol. 1 (2007) 4.

[35] F. Barone, F. Dorr, L. E. Marasco, Design and evaluation of an incoherent feed-forward loop for an arsenic biosensor based on standard iGEM parts, Synthetic Biology (2017) 10. doi:10.1093/synbio/ysx006.

[36] M. Florea, H. Hagemann, G. Santosa, J. Abbott, C. N. Micklem, X. Spencer-Milnes, L. d. A. Garcia, D. Paschou, C. Lazenbatt, D. Kong, H. Chughtai, K. Jensen, P. S. Freemont, R. Kitney, B. Reeve, T. Ellis, Engineering control of bacterial cellulose production using a genetic toolkit and a new cellulose-producing strain, PNAS (2016) 10.

[37] H. Tas, L. Grozinger, A. Goñi-Moreno, V. de Lorenzo, Automated design and implementation of a NOR gate in Pseudomonas putida, Synthetic Biology 6 (2021) ysab024. doi:10.1093/synbio/ysab024.

[38] J. Beal, A. Goñi-Moreno, C. Myers, A. Hecht, M. d. C. Vicente, M. Parco, M. Schmidt, K. Timmis, G. Baldwin, S. Friedrichs, P. Freemont, D. Kiga, E. Ordozgoiti, M. Rennig, L. Rios, K. Tanner, V. Lorenzo, M. Porcar, The long journey towards standards for engineering biosystems: Are the Molecular Biology and the Biotech communities ready to standardise?, EMBO reports 21 (2020). doi:10.15252/embr.202050521, ** This review focuses specifically on the adoption of standards for engineering biology. It presents an overview of the standards in life science and for engineering biology, and identifies the key actors needed for the conversation about SynBio standards.

[39] D. A. Benson, M. Cavanaugh, K. Clark, I. Karsch-Mizrachi, J. Ostell, K. D. Pruitt, E. W. Sayers, GenBank, Nucleic Acids Research 46 (2018) D41–D47. doi:10.1093/nar/gkx1094.

[40] W. R. Pearson, Using the FASTA Program to Search Protein and DNA Sequence Databases, volume 24, Humana Press, New Jersey, 1994, pp. 307–332. doi:10.1385/0-89603-246-9:307.

[41] R. P. Shetty, D. Endy, T. F. Knight, Engineering BioBrick vectors from BioBrick parts, Journal of Biological Engineering 2 (2008) 5. doi:10.1186/1754-1611-2-5.

[42] S. V. Iverson, T. L. Haddock, J. Beal, D. M. Densmore, CIDAR MoClo: Improved MoClo Assembly Standard and New *E. coli* Part Library Enable Rapid Combinatorial Design for Synthetic and Traditional Biology, ACS Synthetic Biology 5 (2016) 99–103. doi:10.1021/acssynbio.5b00124.

[43] J. A. McLaughlin, J. Beal, G. Mısırlı, R. Grünberg, B. A. Bartley, J. Scott-Brown, P. Vaidyanathan, P. Fontanarrosa, E. Oberortner, A. Wipat, T. E. Gorochowski, C. J. Myers, The Synthetic Biology Open Language (SBOL) Version 3: Simplified Data Exchange for Bioengineering, Frontiers in Bioengineering and Biotechnology 8 (2020) 1009. doi:10.3389/fbioe.2020.01009, ** This paper presents the third major iteration of the SBOL standard, SBOL3. The paper summarizes the factors that drove the evolution of SBOL to meet the data sharing needs for the SynBio community.

[44] C. Madsen, A. Goñi Moreno, U. P, Z. Palchick, N. Roehner, C. Atallah, B. Bartley, K. Choi, R. S. Cox, T. Gorochowski, R. Grünberg, C. Macklin, J. McLaughlin, X. Meng, T. Nguyen, M. Pocock, M. Samineni, J. Scott-Brown, Y. Tarter, M. Zhang, Z. Zhang, Z. Zundel, J. Beal, M. Bissell, K. Clancy, J. H. Gennari, G. Misirli, C. Myers, E. Oberortner, H. Sauro, A. Wipat, Synthetic Biology Open Language (SBOL) Version 2.3, Journal of Integrative Bioinformatics 16 (2019). doi:10.1515/jib-2019-0025.

[45] N. Roehner, J. Beal, K. Clancy, B. Bartley, G. Misirli, R. Grünberg, E. Oberortner, M. Pocock, M. Bissell, C. Madsen, T. Nguyen, M. Zhang, Z. Zhang, Z. Zundel, D. Densmore, J. H. Gennari, A. Wipat, H. M. Sauro, C. J. Myers, Sharing Structure and Function in Biological Design with SBOL 2.0, ACS Synthetic Biology 5 (2016) 498–506. doi:10.1021/acssynbio.5b00215.

[46] M. Galdzicki, K. P. Clancy, E. Oberortner, M. Pocock, J. Y. Quinn, C. A. Rodriguez, N. Roehner, M. L. Wilson, L. Adam, J. C. Anderson, B. A. Bartley, J. Beal, D. Chandran, J. Chen, D. Densmore, D. Endy, R. Grünberg, J. Hallinan, N. J. Hillson, J. D. Johnson, A. Kuchinsky, M. Lux, G. Misirli, J. Peccoud, H. A. Plahar, E. Sirin, G.-B. Stan, A. Villalobos, A. Wipat, J. H. Gennari, C. J. Myers, H. M. Sauro, The Synthetic Biology Open Language (SBOL) provides a community standard for communicating designs in synthetic biology, Nature Biotechnology 32 (2014) 545–550. doi:10.1038/nbt.2891.

[47] H. Baig, P. Fontanarossa, V. Kulkarni, J. McLaughlin, P. Vaidyanathan, B. Bartley, S. Bhakta, S. Bhatia, M. Bissell, K. Clancy, R. S. Cox, A. Goñi Moreno, T. Gorochowski, R. Grunberg, J. Lee, A. Luna, C. Madsen, G. Misirli, T. Nguyen, N. Le Novere, Z. Palchick, M. Pocock, N. Roehner, H. Sauro, J. Scott-Brown, J. T. Sexton, G.-B. Stan, J. J. Tabor, L. Terry, M. Vazquez Vilar, C. A. Voigt, A. Wipat, D. Zong, Z. Zundel, J. Beal, C. Myers, Synthetic biology open language visual (SBOL Visual) version 2.3, Journal of Integrative Bioinformatics 0 (2021) 20200045. doi:10.1515/jib-2020-0045.

[48] J. Y. Quinn, R. S. Cox, A. Adler, J. Beal, S. Bhatia, Y. Cai, J. Chen, K. Clancy, M. Galdzicki, N. J. Hillson, N. Le Novère, A. J. Maheshwari, J. A. McLaughlin, C. J. Myers, U. P, M. Pocock, C. Rodriguez, L. Soldatova, G.-B. V. Stan, N. Swainston, A. Wipat, H. M. Sauro, SBOL Visual: A Graphical Language for Genetic Designs, PLOS Biology 13 (2015) e1002310. doi:10.1371/journal.pbio.1002310.

[49] S. M. Keating, D. Waltemath, M. König, F. Zhang, A. Dräger, C. Chaouiya, F. T. Bergmann, A. Finney, C. S. Gillespie, T. Helikar, S. Hoops, R. S. Malik-Sheriff, S. L. Moodie, I. I. Moraru, C. J. Myers, A. Naldi, B. G. Olivier, S. Sahle, J. C. Schaff, L. P. Smith, M. J. Swat, D. Thieffry, L. Watanabe, D. J. Wilkinson, M. L. Blinov, K. Begley, J. R. Faeder, H. F. Gómez, T. M. Hamm, Y. Inagaki, W. Liebermeister, A. L. Lister, D. Lucio, E. Mjolsness, C. J. Proctor, K. Raman, N. Rodriguez, C. A. Shaffer, B. E. Shapiro, J. Stelling, N. Swainston, N. Tanimura, J. Wagner, M. Meier-Schellersheim, H. M. Sauro, B. Palsson, H. Bolouri, H. Kitano, A. Funahashi, H. Hermjakob, J. C. Doyle, M. Hucka, SBML Level 3 Community members, R. R. Adams, N. A. Allen, B. R. Angermann, M. Antoniotti, G. D. Bader, J. Červený, M. Courtot, C. D.

Cox, P. Dalle Pezze, E. Demir, W. S. Denney, H. Dharuri, J. Dorier, D. Drasdo, A. Ebrahim, J. Eichner, J. Elf, L. Endler, C. T. Evelo, C. Flamm, R. M. Fleming, M. Fröhlich, M. Glont, E. Gonçalves, M. Golebiewski, H. Grabski, A. Gutteridge, D. Hachmeister, L. A. Harris, B. D. Heavner, R. Henkel, W. S. Hlavacek, B. Hu, D. R. Hyduke, H. Jong, N. Juty, P. D. Karp, J. R. Karr, D. B. Kell, R. Keller, I. Kiselev, S. Klamt, E. Klipp, C. Knüpfer, F. Kolpakov, F. Krause, M. Kutmon, C. Laibe, C. Lawless, L. Li, L. M. Loew, R. Machne, Y. Matsuoka, P. Mendes, H. Mi, F. Mittag, P. T. Monteiro, K. N. Natarajan, P. M. Nielsen, T. Nguyen, A. Palmisano, J.-B. Pettit, T. Pfau, R. D. Phair, T. Radivoyevitch, J. M. Rohwer, O. A. Ruebenacker, J. Saez-Rodriguez, M. Scharm, H. Schmidt, F. Schreiber, M. Schubert, R. Schulte, S. C. Sealfon, K. Smallbone, S. Soliman, M. I. Stefan, D. P. Sullivan, K. Takahashi, B. Teusink, D. Tolnay, I. Vazirabad, A. Kamp, U. Wittig, C. Wrzodek, F. Wrzodek, I. Xenarios, A. Zhukova, J. Zucker, SBML Level 3: An extensible format for the exchange and reuse of biological models, Molecular Systems Biology 16 (2020). doi:10.15252/msb.20199110, ** This paper reviews the latest iteration of the Systems Biology Markup Language (SBML), SBML Level 3. The standard is used for representing mathematical models of biochemical circuits and systems. The paper provides a detailed explanation of the data standards' structure including its support for constraint-based models, reaction-diffusion models, logical network models, and rule-based models.

[50] M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, and the rest of the SBML Forum:, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, A. A. Cuellar, S. Dronov, E. D. Gilles, M. Ginkel, V. Gor, I. I. Goryanin, W. J. Hedley, T. C. Hodgman, J.-H. Hofmeyr, P. J. Hunter, N. S. Juty, J. L. Kasberger, A. Kremling, U. Kummer, N. Le Novere, L. M. Loew, D. Lucio, P. Mendes, E. Minch, E. D. Mjolsness, Y. Nakayama, M. R. Nelson, P. F. Nielsen, T. Sakurada, J. C. Schaff, B. E. Shapiro, T. S. Shimizu, H. D. Spence, J. Stelling, K. Takahashi, M. Tomita, J. Wagner, J. Wang, The systems biology markup language (SBML): A medium for representation and exchange of biochemical network models, Bioinformatics 19 (2003) 524–531. doi:10.1093/bioinformatics/btg015.

[51] L. Smith, F. Bergmann, A. Garny, T. Helikar, J. Karr, D. Nickerson,

H. Sauro, M. König, Simulation experiment description markup language (SED-ML): Level 1 version 4, J Integr Bioinform 18 (2021) 2021–0021.

[52] F. T. Bergmann, R. Adams, S. Moodie, J. Cooper, M. Glont, M. Golebiewski, M. Hucka, C. Laibe, A. K. Miller, D. P. Nickerson, B. G. Olivier, N. Rodriguez, H. M. Sauro, M. Scharm, S. Soiland-Reyes, D. Waltemath, F. Yvon, N. Le Novère, COMBINE archive and OMEX format: One file to share all information to reproduce a modeling project, BMC Bioinformatics 15 (2014) 369. doi:10.1186/s12859-014-0369-z.

[53] M. W. Davis, ApE—a plasmid editor, Website (2012).

[54] J. Chen, D. Densmore, T. S. Ham, J. D. Keasling, N. J. Hillson, DeviceEditor visual biological CAD canvas, Journal of Biological Engineering 6 (2012) 1. doi:10.1186/1754-1611-6-1.

[55] P. Gaspar, J. L. Oliveira, J. Frommlet, M. A. S. Santos, G. Moura, EuGene: Maximizing synthetic gene design for heterologous expression, Bioinformatics 28 (2012) 2683–2684. doi:10.1093/bioinformatics/bts465.

[56] H. Baig, J. Madsen, A top-down approach to genetic circuit synthesis and optimized technology mapping, in: 9th International Workshop on Bio-Design Automation (Pittsburgh, PA), 2017, pp. 1–2.

[57] M. J. Czar, Y. Cai, J. Peccoud, Writing DNA with GenoCADTM, Nucleic Acids Research 37 (2009) W40–W47. doi:10.1093/nar/gkp361.

[58] L. Watanabe, T. Nguyen, M. Zhang, Z. Zundel, Z. Zhang, C. Madsen, N. Roehner, C. Myers, IBIOSIM 3: A Tool for Model-Based Genetic Circuit Design, ACS Synthetic Biology 8 (2019) 1560–1563. doi:10.1021/acssynbio.8b00078.

[59] C. J. Myers, N. Barker, K. Jones, H. Kuwahara, C. Madsen, N.-P. D. Nguyen, iBioSim: A tool for the analysis and design of genetic circuits, Bioinformatics 25 (2009) 2848–2849. doi:10.1093/bioinformatics/btp457.

[60] N. Roehner, C. J. Myers, Directed Acyclic Graph-Based Technology Mapping of Genetic Circuit Models, ACS Synthetic Biology 3 (2014) 543–555. doi:10.1021/sb400135t.

[61] N. J. Hillson, R. D. Rosengarten, J. D. Keasling, J5 DNA Assembly Design Automation Software, ACS Synthetic Biology 1 (2012) 14–21. doi:10.1021/sb2000116.

[62] G. Misirli, J. S. Hallinan, T. Yu, J. R. Lawson, S. M. Wimalaratne, M. T. Cooling, A. Wipat, Model annotation for synthetic biology: Automating model to nucleotide sequence conversion, Bioinformatics 27 (2011) 973–979. doi:10.1093/bioinformatics/btr048.

[63] J. Beal, T. Lu, R. Weiss, Automatic Compilation from High-Level Biologically-Oriented Programming Language to Genetic Regulatory Networks, PLoS ONE 6 (2011) e22490. doi:10.1371/journal.pone.0022490.

[64] L. Terry, J. Earl, S. Thayer, S. Bridge, C. J. Myers, SBOLCanvas: A Visual Editor for Genetic Designs, ACS Synthetic Biology 10 (2021) 1792–1796. doi:10.1021/acssynbio.1c00096.

[65] L. Huynh, A. Tsoukalas, M. Köppe, I. Tagkopoulos, SBROME: A Scalable Optimization and Module Matching Framework for Automated Biosystems Design, ACS Synthetic Biology 2 (2013) 263–273. doi:10.1021/sb300095m.

[66] J. K. Medley, K. Choi, M. König, L. Smith, S. Gu, J. Hellerstein, S. C. Sealfon, H. M. Sauro, Tellurium notebooks—An environment for reproducible dynamical modeling in systems biology, PLOS Computational Biology 14 (2018) e1006220. doi:10.1371/journal.pcbi.1006220.

[67] K. Choi, J. K. Medley, M. König, K. Stocking, L. Smith, S. Gu, H. M. Sauro, Tellurium: An extensible python-based modeling environment for systems and synthetic biology, Biosystems 171 (2018) 74–79. doi:10.1016/j.biosystems.2018.07.006.

[68] D. Chandran, F. T. Bergmann, H. M. Sauro, Computer-aided design of biological circuits using tinkercell, Bioengineered Bugs 1 (2010) 276–283. doi:10.4161/bbug.1.4.12506.

[69] D. Chandran, H. M. Sauro, Hierarchical Modeling for Synthetic Biology, ACS Synthetic Biology 1 (2012) 353–364. doi:10.1021/sb300033q.

[70] D. Chandran, F. T. Bergmann, H. M. Sauro, TinkerCell: Modular CAD tool for synthetic biology, Journal of Biological Engineering 3 (2009) 19. doi:10.1186/1754-1611-3-19.

[71] M. Bates, J. Lachoff, D. Meech, V. Zulkower, A. Moisy, Y. Luo, H. Tekotte, C. J. Franziska Scheitz, R. Khilari, F. Mazzoldi, D. Chandran, E. Groban, Genetic Constructor: An Online DNA Design Platform, ACS Synthetic Biology 6 (2017) 2362–2365. doi:10.1021/acssynbio.7b00236.

[72] B. Yordanov, N. Dalchau, P. K. Grant, M. Pedersen, S. Emmott, J. Haseloff, A. Phillips, A Computational Method for Automated Characterization of Genetic Components, ACS Synthetic Biology 3 (2014) 578–588. doi:10.1021/sb400152n.

[73] P. Fontanarrosa, H. Doosthosseini, A. E. Borujeni, Y. Dorfan, C. A. Voigt, C. Myers, Genetic Circuit Dynamics: Hazard and Glitch Analysis, ACS Synthetic Biology (2020) 15.

[74] G. Misirli, T. Nguyen, J. A. McLaughlin, P. Vaidyanathan, T. S. Jones, D. Densmore, C. Myers, A. Wipat, A Computational Workflow for the Automated Generation of Models of Genetic Designs, ACS Synthetic Biology 8 (2019) 1548–1559. doi:10.1021/acssynbio.7b00459.

[75] W. Xiang, W. Xiao-Cui, J. Jing-Bo, S. Jin-Jing, P. Shu-Song, H. Han-Hong, W. Wei-Ke, Genetic Circuit for the Early Warning of Lung Cancer using iBioSim, ITM Web of Conferences (2016) 5.

[76] J. Mante, Y. Hao, J. Jett, U. Joshi, K. Keating, X. Lu, G. Nakum, N. E. Rodriguez, J. Tang, L. Terry, X. Wu, E. Yu, J. S. Downie, B. T. McInnes, M. H. Nguyen, B. Sepulvado, E. M. Young, C. J. Myers, Synthetic Biology Knowledge System, ACS Synthetic Biology 10 (2021) 2276–2285. doi:10.1021/acssynbio.1c00188.

[77] J. Mante, N. Roehner, K. Keating, J. A. McLaughlin, E. Young, J. Beal, C. J. Myers, Curation Principles Derived from the Analysis of the SBOL iGEM Data Set, ACS Synthetic Biology (2021) acssynbio.1c00225.

doi:10.1021/acssynbio.1c00225, ** This paper presents a study of the iGEM registry of parts using statistical analysis, SnapGene auto-annotation, SynBict auto-annotation, and expert analysis. This paper identifies issues of the part library, including the use of free text, insufficient part provenance, part duplication, lack of part removal, and insufficient continuous curation. Finally, the paper suggests possible solutions.

[78] F. Zhang, Y. Sun, C. Luo, Microfluidic approaches for synthetic gene circuits' construction and analysis, Quantitative Biology 9 (2021) 47. doi:10.15302/J-QB-021-0235.

[79] J. Zhang, Y. Chen, L. Fu, E. Guo, B. Wang, L. Dai, T. Si, Accelerating strain engineering in biofuel research via build and test automation of synthetic biology, Current Opinion in Biotechnology 67 (2021) 88–98. doi:10.1016/j.copbio.2021.01.010, * This review provides further information on build and test automation in synthetic biology. It presents automation methods for part preparation, multi-part assembly, and construct verification in biofuel research in synthetic biology.

[80] M. May, A DIY approach to automating your lab, Nature 569 (2019) 587–588. doi:10.1038/d41586-019-01590-z.

[81] A. Espah Borujeni, J. Zhang, H. Doosthosseini, A. A. K. Nielsen, C. A. Voigt, Genetic circuit characterization by inferring RNA polymerase movement and ribosome usage, Nature Communications 11 (2020) 5001. doi:10.1038/s41467-020-18630-2.

[82] T. E. Gorochowski, I. Chelysheva, M. Eriksen, P. Nair, S. Pedersen, Z. Ignatova, Absolute quantification of translational regulation and burden using combined sequencing approaches, Molecular Systems Biology 15 (2019). doi:10.15252/msb.20188719.

[83] B. Canton, A. Labno, D. Endy, Refinement and standardization of synthetic biological parts and devices, Nature Biotechnology 26 (2008) 787–793. doi:10.1038/nbt1413.

[84] Y. Matsuoka, S. Ghosh, H. Kitano, Consistent design schematics for biological systems: Standardization of representation in biological engineering, Journal of The Royal Society Interface 6 (2009). doi:10.1098/rsif.2009.0046.focus.

[85] M. B. Elowitz, Stochastic Gene Expression in a Single Cell, Science 297 (2002) 1183–1186. doi:10.1126/science.1070919.

[86] L. Buecherl, R. Roberts, P. Fontanarrosa, P. J. Thomas, J. Mante, Z. Zhang, C. J. Myers, Stochastic Hazard Analysis of Genetic Circuits in iBioSim and STAMINA, ACS Synthetic Biology (2021) acssynbio.1c00159. doi:10.1021/acssynbio.1c00159.

[87] V. Bartoli, M. di Bernardo, T. E. Gorochowski, Self-adaptive biosystems through tunable genetic parts and circuits, Current Opinion in Systems Biology 24 (2020) 78–85. doi:10.1016/j.coisb.2020.10.006.

[88] M. Cotner, J. Zhan, Z. Zhang, A Computational Metabolic Model for Engineered Production of Resveratrol in Escherichia coli, ACS Synthetic Biology 10 (2021) 1992–2001. doi:10.1021/acssynbio.1c00163.

[89] G. Naseri, M. A. G. Koffas, Application of combinatorial optimization strategies in synthetic biology, Nature Communications 11 (2020) 2446. doi:10.1038/s41467-020-16175-y, * This review article covers the importance of automatic optimization in the computational workflow. It highlights computational and machine learning methods to help generate optimal constructs to aid in *in silico* design.

[90] M. Gómez-Schiavon, G. Dods, H. El-Samad, A. H. Ng, Multidimensional Characterization of Parts Enhances Modeling Accuracy in Genetic Circuits, ACS Synthetic Biology 9 (2020) 2917–2926. doi:10.1021/acssynbio.0c00288.