`standard-GEM`: standardization of open-source genome-scale metabolic models

Mihail Anton[1,✉], Eivind Almaas[2], Rui Benfeitas[3], Sara Benito-Vaquerizo[4], Lars M. Blank[5], Andreas Dräger[6], John M. Hancock[7], Cheewin Kittikunapong[8], Matthias König[9], Feiran Li[8], Ulf W. Liebal[5], Hongzhong Lu[10], Hongwu Ma[11], Radhakrishnan Mahadevan[12], Adil Mardinoglu[13,14], Jens Nielsen[15,16,17], Juan Nogales[18], Marco Pagni[19], Jason A. Papin[20], Kiran Raosaheb Patil[21], Nathan D. Price[22,23], Jonathan L. Robinson[17], Benjamín J. Sánchez[24], Maria Suarez-Diez[4], Snorre Sulheim[25], L. Thomas Svensson[1], Bas Teusink[26], Wanwipa Vongsangnak[27], Hao Wang[8], Ahmad A. Zeidan[24], and Eduard J. Kerkhoven[8,✉]

[1]Department of Life Sciences, National Bioinformatics Infrastructure Sweden, Science for Life Laboratory, Chalmers University of Technology, Gothenburg SE-41258, Sweden
[2]Department of Biotechnology and Food Science, NTNU - Norwegian University of Science and Technology, Sem Sælands veg 6/8, N-7491 Trondheim, Norway
[3]National Bioinformatics Infrastructure Sweden (NBIS), Science for Life Laboratory, Department of Biochemistry and Biophysics, Stockholm University, Tomtebodavägen 23A, Solna SE-17165, Sweden
[4]Laboratory of Systems and Synthetic Biology, Wageningen University & Research, Stippeneng 4 6708 WE, Wageningen, The Netherlands
[5]iAMB - Institute of Applied Microbiology, ABBT, RWTH Aachen University, 52074 Aachen, Germany
[6]Computational Systems Biology of Infections and Antimicrobial-Resistant Pathogens, Institute for Bioinformatics and Medical Informatics (IBMI), Department of Computer Science, Eberhard Karl University of Tübingen, 72076 Tübingen, Germany
[7]Institute of Biochemistry, Faculty of Medicine, University of Ljubljana, Zaloška 4, 1000 Ljubljana, Slovenia
[8]Department of Life Sciences, Chalmers University of Technology, Kemivägen 10, SE-41258 Gothenburg, Sweden
[9]Institute for Biology, Institute for Theoretical Biology, Systems Medicine of the Liver Lab, Humboldt-University Berlin, Philippstraße 13, 10115 Berlin, Germany
[10]State Key Laboratory of Microbial Metabolism, School of Life Sciences and Biotechnology, Shanghai Jiao Tong University, Shanghai, China
[11]Biodesign Center, Tianjin Institute of Industrial Biotechnology, Chinese Academy of Sciences, Tianjin, 300308, China
[12]Department of Chemical Engineering and Applied Chemistry; Institute of Biomedical Engineering, University of Toronto, 200, College Street, University of Toronto
[13]Science for Life Laboratory, KTH - Royal Institute of Technology, Stockholm, Sweden
[14]Centre for Host-Microbiome Interactions, Faculty of Dentistry, Oral & Craniofacial Sciences, King's College London, London, United Kingdom
[15]Department of Life Sciences, Chalmers University of Technology, Gothenburg SE-412 96, Sweden
[16]Novo Nordisk Foundation Center for Biosustainability, Chalmers University of Technology, Gothenburg SE-412 96, Sweden
[17]BioInnovation Institute, Ole Maaløes Vej 3, Copenhagen DK-2200, Denmark
[18]Department of Systems Biology, Centro Nacional de Biotecnología, CSIC, C/ Darwin 3, Madrid, Spain
[19]Vital-IT group, SIB Swiss Institute of Bioinformatics, Lausanne, Switzerland
[20]Department of Biomedical Engineering, University of Virginia, Charlottesville, Virginia, USA
[21]Medical Research Council Toxicology Unit, University of Cambridge, Cambridge, UK
[22]Thorne HealthTech, New York, NY 10019, USA
[23] Institute for Systems Biology Seattle, WA 98109, USA
[24]Systems Biology, Discovery, R&D, Chr. Hansen A/S, 2970 Hørsholm, Denmark
[25]Department of Biotechnology and Nanomedicine, SINTEF Industry, Richard Birkelands vei 3, 7052 Trondheim, Norway
[26]Systems Biology Lab, Amsterdam Institute of Molecular and Life Science, VU University Amsterdam, O2 building, location code 2E51, De Boelelaan 1085, 1081HV Amsterdam, The Netherlands
[27]Department of Zoology, Faculty of Science, Kasetsart University, Bangkok 10900, Thailand

ABSTRACT. The field of metabolic modelling at the genome-scale continues to grow with more models being created and curated. This comes with an increasing demand for adopting common principles regarding transparency and versioning, in addition to standardisation efforts regarding file formats, annotation and testing. Here, we present a standardised template for git-based and GitHub-hosted genome-scale metabolic models (GEMs) supporting both new models and curated ones, following FAIR principles (findability, accessibility, interoperability, and reusability), and incorporating best-practices. `standard-GEM` facilitates the reuse of GEMs across web services and platforms in the metabolic modelling field and enables automatic validation of GEMs. The use of this template for new models, and its adoption for existing ones, paves the way for increasing model quality, openness, and accessibility with minimal effort.

Availability: `standard-GEM` is available from github.com/MetabolicAtlas/standard-GEM under the conditions of the CC BY 4.0 licence along with additional supporting material.

genome-scale metabolic model | open source | versioning

Correspondence: ***mihail.anton@chalmers.se, eduardk@chalmers.se***

Introduction

Genome-scale metabolic models (GEMs) have been used to understand and guide modifications of metabolism with a wide range of applications, from elucidating disease mechanisms and identifying biomarkers to optimising strains for producing valuable chemicals (Chen et al., 2019). However, the size of the GEMs and the iterative curation work required to build and refine them, as detailed in a 96-step standard operating procedure by Thiele and Palsson (2010), naturally leads modellers to desire a framework that borrows concepts commonly found in software development. Together with the code associated with performing model simulations and analyses, modellers need an accessible ensemble of standards and tools that is easy to work with, yet carries much of the burden of ensuring quality, while allowing the researcher to focus on the scientific aspects of their work.

Scientific results, including those from model analysis, are increasingly scrutinised for reproducibility (Baker, 2016, Malik-Sheriff et al., 2020). Reproducibility can be viewed as part of a wider need for open science data management aimed at the reuse of what can generically be termed Digital Research Objects (DROs). The FAIR Principles (Findable, Accessible, Interoperable, Reusable (Wilkinson et al., 2016)) provide an easily accessible suite of recommendations for the sharing of DROs. These principles are conceived as broadly applicable not only to data but also to other classes of DROs including software tools, workflows and systems biology models. As highlighted by the Infrastructure for Systems Biology Europe (ISBE), "sharing [data and] models solely through supplementary material is still common practice" (Stanford et al., 2015), and they recommend that platforms should be used to disseminate assets. In line with this, EMBO Reports and other scientific journals have advocated using BioModels (Malik-Sheriff et al., 2020) as a default deposition database for sharing published models in standardised formats, such as the Systems Biology Markup Language (SBML) (Keating et al., 2020). ISBE in cooperation with ERASysAPP initiated a key step towards promoting FAIRification in systems biology with the development of FAIRDOM and FAIRDOMHub (Wolstencroft et al., 2016). FAIRDOMHub can be thought of as a publishing platform for systems modelling projects which links their various elements from data to model. These may be located at different physical locations and in different types of repository. By facilitating linking between resources such as SEEK (Wolstencroft et al., 2015) and OpenBIS (Barillari et al., 2016), FAIRDOMHub thus, facilitates the reproducible reuse of models. Here, we propose to continue this advancement in FAIRification through the implementation of open-source standards both during model creation and in its subsequent curation, not only for making the final model available in a reproducible format, but also enabling reproducibility throughout the model development and curation process (Tiwari et al., 2021).

The importance of code versioning has been well-recognised for biological models and repositories, especially for models that constantly evolve (Beard et al., 2009, Miller et al., 2011). Code versioning can improve the reproducibility and transparency of the curation process (Aite et al., 2018), particularly for GEMs that are continuously updated over the years by many researchers (Ravikrishnan and Raman, 2015). For example, the consensus GEM of *Saccharomyces cerevisiae* has been developed since 2008 (Herrgård et al.), and constantly updated with new curations, reaching version 8 in 2019 (Lu et al., 2019). At the same time, models hosted by databases such as the BiGG database (Norsigian et al., 2020) have been updated by applying the curation tool ModelPolisher (King et al., 2016). However, the changes that have been applied to the BiGG models are not traceable by the research community. Such missing provenance information makes it difficult to evaluate changes between versions. Similarly, in the latest version of BioModels, the authors note that it "transparently tracks changes to the model and associated files behind the scenes" (Malik-Sheriff et al., 2020), but such information is not readily available to researchers working with the models. `Standard-GEM`, therefore, aims at, but is not limited to, facilitating and streamlining the constant manual curation process takes place after publication by making all changes equally structured, explicit, and trackable, previously identified as desirable characteristics by Waltemath et al. (2013).

As noted by Scharm et al. (2018), "reuse of models is still impeded by a lack of trust and documentation". Within this work, we developed `standard-GEM`, a standardised git-based template for the development and curation of GEMs which addresses issues of reproducibility and availability of provenance information and applies FAIR principles to GEMs.

With `standard-GEM`, we aim to address this problem by imposing standards for model reuse, such as fixed folder structure and the use of a git-based workflow for version control, and by relying on repository-hosting solutions that provide open issues tracking and discussion forums to further boost the documentation process. We thus attempt to solve the need for standardisation by facilitating the creation of a community standard focused on simplicity. To this end, the standard introduced by `standard-GEM`, defined through the *.standard-GEM.md* file, is explicit and transparent.

Results

Template repository. As part of the repository template available at github.com/MetabolicAtlas/standard-GEM, the central file *.standard-GEM.md* (Supplementary Note 1) contains all the expectations of the standard, including steps for repository creation, git workflow, and model file formats. Additionally, using the `standard-GEM` template repository initiates a well-defined folder structure, including a structured *README* file that facilitates the presentation of the model, and a default licence file (CC BY 4.0).

As a template repository, there are no restrictions on making repositories public. This template can be used to create both private and public repositories identically in GitHub, or any git-hosting service. As some functionality of the template is leveraging the GitHub configuration via the *.github* folder, adopting it with another hosting service would entail additional work.

Automatic validation. GEMs must be maintained to maximise their value. This continuous effort is essential, and it also applies to GEMs created from the `standard-GEM` template. One of the costs associated with maintaining an open-source GEM is the burden of running checks, ranging from basic sanity checks to more complex evaluations of model content. To reduce this burden, we have created an automatic validation pipeline, publicly available at github.com/MetabolicAtlas/standard-GEM-validation. First, it uses the GitHub application programming interface (API) to identify all standard-GEM models by finding all repositories labelled "standard-GEM." The pipeline then proceeds to perform a validation of the repository's content by looking at the file tree. Then, after the repository has been deemed compliant, the pipeline verifies the formatting of the YAML and SBML files and runs the model testing framework MEMOTE (Lieven et al., 2020). Due to the inherent modularity, it is envisioned that more tests will be added. In addition, since previous releases of a model are all available in the same repository, tests can also be run retroactively on previous releases. All the results of the automatic pipeline are versioned in JSON format in the same repository as the pipeline to be easily reused or presented, e.g., on a website. A snapshot of the validation output is also provided as Supplementary Data. The pipeline regularly runs on its own, which means new results will come in as the models continue to be upgraded. This reduces the need for modellers to maintain identical validation pipelines separately, thus reducing the cost of maintenance, while keeping the benefit of validation checks.

Discussion

Generic versus specialised, and platform versus database. We advocate using `standard-GEM` through an independent platform that comes at no cost to the scientific community. While a git-based standard is a platform-independent solution, closer integration with GitHub specifically brings increased simplicity, for example, by instantiating a standard-compliant GEM with a single mouse click. Nonetheless, standard-GEM could alternatively be applied to other platforms of distributed version control and source code management such as GitLab or BitBucket.

A fundamental debate that underpins standard-GEM is the use of a generic and financially free infrastructure versus a specialised and non-free platform. An example of the latter has been implemented by MEMOsys (Pabinger et al., 2011), which proposed a centralised version control system. On the one hand, it offered "access to the complete development history" of a model, not unlike standard-GEM does through git and GitHub. Similar to this, `standard-GEM` opens the model for community curation. On the other hand, MEMOsys placed the version control platform under the responsibility of researchers, which exerts financial strains. More importantly, however, a platform like MEMOsys dissociates the code from the model, as it "has been designed to map and store all properties of a metabolic model in a database" (Pabinger et al., 2011), thus inadvertently limiting traceability and reproducibility of the changes.

Instead of storing all properties of a metabolic model in a database, `standard-GEM` provides a standardised format such that researchers can continue to work in an open-source way, directly combining model with code, whilst keeping the output file format easy to be reused by different model-centric websites such as BioModels, BiGG, FAIRDOMHub, Metabolic Atlas (Wang et al., 2021), and JWS Online (Olivier and Snoep, 2004). A more recent approach, ModelBricks, defines an alternative infrastructure of small models that requires the development of a database, tools, and content (Cowan et al., 2019). To some extent, standard-GEM facilitates the reuse of a model by forking the source repository. However, how to combine multiple standard-GEMs as proposed through ModelBricks remains an open question.

Suitability of a git-based workflow. The use of git and GitHub in computational biology has become increasingly popular. An introduction to these tools and how to use them in ten simple rules has been previously published (Perez-Riverol et al., 2016).

A good standard for GEM versioning should facilitate answering the 7W questions of provenance (*What*, *Who*, *When*, *Where*, *Which*, *Why*, *With/How*) that underpin each action in the curation process (Ram and Liu, 2012). Previously, one approach to address provenance post-curation would have been through an algorithm for difference detection in models of biological systems (Scharm et al., 2016), even though it was intended to compare finished models. The authors note that "there is scope for further extensions to provide hypotheses for the Why." Standard-GEM answers all the 7W questions by employing the version control tool git, which is widely used for tracking changes in software programming in a distributed way. Hosting a git-based project on an online platform such as GitHub makes these answers transparent by showing *What* the change is (a git commit), *Who* made the change (author/committer), *When* it occurred (git commit timestamp), *Where* it occurred (in which file or part of the model), *Which* changes were made (visible through a git diff). The *Why* and *With* are addressable through the functionality of online platforms, e.g., GitHub's Issues and Pull requests and commit messages for which `standard-GEM` provides templates. Discussions also occur openly on the GitHub platform, with changes to the standard being connected to issues, which can be raised by GitHub users. Through this approach, standard-GEM aims to document all changes of the standard and to equally invite contributors. Therefore, if used adequately, git-based versioning for models can bring benefits in terms of traceability. For these reasons, git was chosen to stand at the foundation of standard-GEM.

An advantage of using git versioned GEMs is that it makes comparisons between consecutive versions of the model straightforward. With git-flow, the standard development workflow on git-based platforms in the software industry, the differences across versions can be easily inspected via branch comparisons. In cases where git diff cannot work, as can be the case for SBML files with scrambled order, the tool *sbml-diff* can be used for comparisons between models or versions (Scott-Brown and Papachristodoulou, 2017).

However, one limitation inherent to the use of git is that line order changes matter, even if the lines' content is otherwise identical. The recommendation is that tools maintain the order of model components to provide a consistent order and robustness to the model files and make them suitable for git.

While promising, the use of code versioning principles when working with GEMs is not straightforward. One could look again at software programming, where best practices have been defined and widely spread during the past decade. However, many code conventions and configuration options can become overly complex for a metabolic modelling project. The GEM testing tool MEMOTE addresses this by providing `memote-cookiecutter`, a utility to instantiate a git-versioned project structure (Lieven et al., 2020). This tool brings significant benefits in terms of standardisation and time-savings. However, the rules and reasoning defining the internal file tree and *git* branch structure of a repository in-

Repository name	Country	Source URL
Anaerotignum_neopropionicum	Netherlands	gitlab.com/wurssb/Modelling/Anaerotignum_neopropionicum
Fruitfly-GEM	Sweden	github.com/SysBioChalmers/Fruitfly-GEM
Human-GEM	Sweden	github.com/SysBioChalmers/Human-GEM
Mouse-GEM	Sweden	github.com/SysBioChalmers/Mouse-GEM
Opol-GSMM	Germany	github.com/iAMB-RWTH-Aachen/Opol-GSMMM
Rat-GEM	Sweden	github.com/SysBioChalmers/Rat-GEM
Sco-GEM	Sweden	github.com/SysBioChalmers/Sco-GEM
vna-GEM	China	github.com/tibbdc/vna-GEM
Ustilago_maydis-GEM	Germany	github.com/iAMB-RWTH-Aachen/Ustilago_maydis-GEM
yeast-GEM	Sweden	github.com/SysBioChalmers/yeast-GEM
Worm-GEM	Sweden	github.com/SysBioChalmers/Worm-GEM
Zebrafish-GEM	Sweden	github.com/SysBioChalmers/Zebrafish-GEM

Table 1. A list of GEMs that have adopted standard-GEM and their geographical location, as fetched from github.com/topics/standard-gem and gitlab.com/explore/projects/topics/standard-gem.

stantiated with `memote-cookiecutter` are complex and not fully transparent.

Adhering to standards. `standard-GEM` is a standard as per the definition that it is only mutual agreement, and that “introducing standards should not be a goal in itself, but should help [. . .] solve problems” (Brazma et al., 2006).
As noted by Yurkovich et al. (2017), “one of the biggest hurdles for those attempting to link different pieces of software is standardisation”, to which they formulate “Lesson 6: Adopt or Develop Standards.” In line with this recommendation, standard-GEM limits the development of new standards to where they are strictly necessary, e.g., by promoting model file formats in the community. A prominent example is SBML as the de facto file format for GEMS. Other file formats exist, such as SBtab (Lubitz et al., 2016), which provides a spreadsheet-like interface to the model and its annotation. Different modelling formats have been reviewed by Dräger and Palsson (2014) and Schreiber et al. (2020), and usage of file formats has been polled by Carey et al. (2020), based on which of the file format recommendations in `standard-GEM` were made. Furthermore, `standard-GEM` is programming language agnostic, thus making it compatible with any systems biology software. It also leverages the GitHub application programming interface (API) for programmatic access to the entire repository.
With `standard-GEM`, we build on existing standards and reduce complexity. Recently, a set of features that defines a “gold standard” for metabolic network reconstruction has been compiled regarding content, annotation, and simulation capabilities (Carey et al., 2020). `Standard-GEM` aims to implement the proposed standards, being aware of the warning that a "consistent use of standards and resources remains challenging” (Carey et al., 2020). The proposed standards for the *de novo* reconstruction phase are implemented through a templated *README* file that is part of the default `standard-GEM` configuration. For the proposed standards of the curation process, `standard-GEM` leverages git-flow and the transparency of git-based platforms – commit messages, branches, issues, pull requests – to achieve the desired transparency in the reconstruction and curation process (Heavner and Price, 2015). Regarding the proposed standards for simulations, `standard-GEM` encourages the separation of concerns by using a fixed folder structure and *README* files. Moreover, `standard-GEM` aims to work towards full compatibility with the COMBINE archive standard (Bergmann et al., 2014) and RO-crate (Soiland-Reyes et al., 2021).
Compared to other standardisation efforts and initiatives, `standard-GEM` brings many, if not all, advantages at no financial cost to the scientific community.

FAIR. Adopting `standard-GEM`, via the reliance on mature technologies and platforms, elevates FAIRification in ways that are new to the field, and that have not been addressed before.
Findable. Using `standard-GEM` promotes findability. As mentioned previously, several established resources already exist for storing models that are considered finished, in the sense that they are ready for publication. Making models findable in this way, however, decreases the findability of other aspects. For example, it becomes much harder to find the rationale for specific choices modellers made. When a GEM is set up according to `standard-GEM`, it increases the findability of the information that accompanies an individual model file as a product.
Another aspect of findability to be considered is that of the repositories themselves. To emulate repository categories, GitHub uses searchable topics, and the newly introduced ”standard-GEM” topic makes it easy to find all such GEMs on GitHub (Table 1). Additionally, the recommended use of Zenodo ensures automatic DOI minting for each release in the GitHub repository (Supplementary Note 1).
Accessible. The main contribution of `standard-GEM` to FAIRification is via accessibility. When depositing a model file in a deposition database, only a minor part that goes into the development of the model becomes accessible at that resource. For example, data or code cannot be submitted to any resource. Moreover, thanks to the *git* versioning system and the online hosting of the repository, even previous versions of the files in the repository remain accessible. Furthermore, in anticipation of the low risk that the repository suddenly disappears, as part of the *.standard-GEM.md* file, we recommend the permanent archiving service Zenodo which integrates seamlessly with GitHub (Supplementary Note 1). In addition, GitHub allows easy switching between the private/public state of a repository, independently of it being instantiated from a repository template such as

standard-GEM.

Interoperable. While SBML has become the *de facto* file format, others focus on different use-cases where SBML is less optimal. For example, due to instrinsic properties of the XML format that SBML is based on, even small model curations can lead to more line changes than in other file formats. Instead of promoting a single solution as is usually the case with deposition databases, standard-GEM pools together many standards. This is achieved by recommending different file formats, thus increasing the interoperability of the models, which is a key pillar of standard-GEM. The RAVEN toolbox, for example, provides the wrapper function *exportForGit* that exports multiple formats at once, to reduce the chance of out-of-sync exports.

Reusable. Standard-GEM adds a new dimension to reusability. Whereas the use of standard file formats primarily covers reusability per its official interpretation, especially regarding annotation, the provenance aspect is largely overlooked. With a code versioning system, however, changes in a repository tend to appear gradually. In a standard-GEM, incremental curations have even more context than generic curation databases, such as APICURON (Hatos et al., 2021), focusing more on curation as a trackable activity than on the curation rationale. The provenance of individual curations in a GEM makes them more reusable to, e.g., another GEM, or it might even lead to annotation curations in other databases.

A different aspect of reusability inherent to a standard-GEM is that it is expected to contain code. Large curations, even if implemented manually, benefit from being code-aided. Even if such code is meant for a one-time use, versioning it in the repository makes it reusable for future curations.

Lastly, the reusability of a repository increases as soon as it becomes public on a git-hosting service, such as GitHub. Public repositories can be forked by other users, enabling them to duplicate the entire repository, including the change history, in their account, thus reusing it for their work. Later, their new changes can be conveniently incorporated in the original repository via a pull request, thus making even work outside the original repository reusable.

Future perspective. At the time of publication, standard-GEM has reached version 0.5. As the field matures, it is expected that the standard will do so as well. The discussions that have already taken place in the repository indicate potential future directions, such as expanding standard-GEM to include support of Community-GEMs and validation improvements. Moreover, instead of becoming a static standard, standard-GEM has been set up as an ”open system” to continue evolving and serving the community's future needs. Together with the standard-GEM template and resources, we will continue to support the conversion of models, as deposited in their respective publications, to the standard-GEM format, thus enabling community curation, expansion, versioning, and transparent annotation of these models.

Conclusion

By creating and disseminating standard-GEM, we are going much further than documenting the GEM reconstruction process — we are supporting open-source long-term model curation. Following standard-GEM is a way to make a model "reusable, extensible, and published open-source" (Medley et al., 2016). Upon this standard, further validation is achieved through an open-source automated pipeline. By adopting the workflow described by standard-GEM, the curation history of genome-scale and other metabolic models can be transparently reviewed in the git repository, thus facilitating continued community contributions and enabling GEMs to evolve from being research outputs to acting like an infrastructure.

ACKNOWLEDGEMENTS
The authors thank past and current members of the Systems and Synthetic Biology division at the Department of Life Sciences at Chalmers University of Technology for previous work that paved the way for standard-GEM and for valuable discussions.

References

M. Aite, M. Chevallier, C. Frioux, C. Trottier, J. Got, M. P. Cortés, S. N. Mendoza, G. Carrier, O. Dameron, N. Guillaudeux, M. Latorre, N. Loira, G. V. Markov, A. Maass, and A. Siegel. Traceability, reproducibility and wiki-exploration for "à-la-carte" reconstructions of genome-scale metabolic models. *PLoS computational biology*, 14:1–25, 2018. doi: 10.1371/journal.pcbi.1006146.

M. Baker. 1,500 scientists lift the lid on reproducibility. *Nature*, 533(7604):452–454, 2016. doi: 10.1038/533452a.

C. Barillari, D. S. Ottoz, J. M. Fuentes-Serna, C. Ramakrishnan, B. Rinn, and F. Rudolf. openbis eln-lims: an open-source database for academic laboratories. *Bioinformatics*, 32(4):638–640, 2016.

D. A. Beard, R. Britten, M. T. Cooling, A. Garny, M. D. Halstead, P. J. Hunter, J. Lawson, C. M. Lloyd, J. Marsh, A. Miller, D. P. Nickerson, P. M. Nielsen, T. Nomura, S. Subramanium, S. M. Wimalaratne, and T. Yu. CellML metadata standards, associated tools and repositories. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367(1895):1845–1867, May 2009. doi: 10.1098/rsta.2008.0310.

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, and N. Le Novère. Combine archive and omex format: One file to share all information to reproduce a modeling project. *BMC bioinformatics*, 15(1):1–9, 2014. doi: 10.1186/s12859-014-0369-z.

A. Brazma, M. Krestyaninova, and U. Sarkans. Standards for systems biology. *Nature Reviews Genetics*, 7:593–605, 2006. doi: 10.1038/nrg1922.

M. A. Carey, A. Dräger, M. E. Beber, J. A. Papin, and J. T. Yurkovich. Community standards to facilitate development and address challenges in metabolic modeling. *Molecular Systems Biology*, 16:1–9, 2020. doi: 10.15252/msb.20199235.

Y. Chen, G. Li, and J. Nielsen. *Genome-Scale Metabolic Modeling from Yeast to Human Cell Models of Complex Diseases: Latest Advances and Challenges*, pages 329–345. Springer New York, New York, NY, 2019. ISBN 978-1-4939-9736-7. doi: 10.1007/978-1-4939-9736-7_19.

C. Commons. Cc by 4.0, 2022. URL https://creativecommons.org/licenses/by/4.0/.

A. E. Cowan, P. Mendes, and M. L. Blinov. Modelbricks—modules for reproducible modeling improving model annotation and provenance. *NPJ Systems Biology and Applications*, 5, 2019. doi: 10.1038/s41540-019-0114-3.

A. Dräger and B. Ø. Palsson. Improving collaboration by standardization efforts in systems biology. *Frontiers in Bioengineering*, 2(61):1–20, Dec. 2014. doi: 10.3389/fbioe.2014.00061.

EMBO Reports. Author guidelines, 2022. URL https://www.embopress.org/page/journal/14693178/authorguide#:~:text=Authors%20are%20strongly%20encouraged%20to%20follow%20the%20set%20of%20guidelines%20provided%20by%C2%A0Combine%C2%A0and%20deposit%20their%20model%20in%20a%20public%20database%20such%20as%C2%A0Biomodels%C2%A0or%C2%A0JWS%20Online.

A. Hatos, F. Quaglia, D. Piovesan, and S. C. E. Tosatto. Apicuron: a database to credit and acknowledge the work of biocurators. *Database (Oxford)*, 2021, 2021. doi: 10.1093/database/baab019.

B. D. Heavner and N. D. Price. Transparency in metabolic network reconstruction enables scalable biological discovery. *Current opinion in biotechnology*, 34:105–109, 2015. doi: 10.1016/j.copbio.2014.12.010.

M. J. Herrgård, N. Swainston, P. Dobson, W. B. Dunn, K. Y. Arga, M. Arvas, N. Büthgen, S. Borger, R. Costenoble, M. Heinemann, M. Hucka, N. Le Novère, P. Li, W. Liebermeister, M. L. Mo, A. P. Oliveira, D. Petranovic, S. Pettifer, E. Simeonidis, K. Smallbone, I. Spasić, D. Weichart, R. Brent, D. S. Broomhead, H. V. Westerhoff, B. Kirdar, M. Penttilä, E. Klipp, B. Palsson, U. Sauer, S. G. Oliver, P. Mendes, J. Nielsen, and D. B. Kell. A consensus yeast metabolic network reconstruction obtained from a community approach to systems biology. *Nature Biotechnology*, 26:1155–1160, 2008. doi: 10.1038/nbt1492.

ISO. How to write standards, 2016.

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, 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, et al. Sbml level 3: an extensible format for the exchange and reuse of biological models. *Molecular Systems Biology*, 16:1–21, 2020. doi: 10.15252/msb.20199110.

Z. A. King, J. Lu, A. Dräger, P. Miller, S. Federowicz, J. A. Lerman, A. Ebrahim, B. O. Palsson, and N. E. Lewis. Bigg models: A platform for integrating, standardizing and sharing genome-scale models. *Nucleic Acids Research*, 44:D515–D522, 2016. doi: 10.1093/nar/gkv1049.

C. Lieven, M. E. Beber, B. G. Olivier, F. T. Bergmann, M. Ataman, P. Babaei, J. A. Bartell, L. M. Blank, S. Chauhan, K. Correia, C. Diener, A. Dräger, B. E. Ebert, J. N. Edirisinghe, J. P. Faria, A. M. Feist, G. Fengos, R. M. Fleming, B. García-Jiménez, V. Hatzimanikatis, W. van Helvoirt, C. S. Henry, H. Hermjakob, M. J. Herrgård, A. Kaafarani, H. U. Kim, Z. King, S. Klamt, E. Klipp, J. J. Koehorst, M. König, M. Lakshmanan, D. Y. Lee, S. Y. Lee, S. Lee, N. E. Lewis, F. Liu, H. Ma, D. Machado, R. Mahadevan, P. Maia, A. Mardinoglu, G. L. Medlock, J. M. Monk, J. Nielsen, L. K. Nielsen, J. Nogales, I. Nookaew, B. O. Palsson, J. A. Papin, K. R. Patil, M. Poolman, N. D. Price, O. Resendis-Antonio, A. Richelle, I. Rocha, B. J. Sánchez, P. J. Schaap, R. S. Malik Sheriff, S. Shoaie, N. Sonnenschein, B. Teusink, P. Vilaça, J. O. Vik, J. A. Wodke, J. C. Xavier, Q. Yuan, M. Zakhartsev, and C. Zhang. Memote for standardized genome-scale metabolic model testing. *Nature Biotechnology*, 38:272–276, 2020. doi: 10.1038/s41587-020-0446-y.

H. Lu, F. Li, B. J. Sánchez, Z. Zhu, G. Li, I. Domenzain, S. Marcišauskas, P. M. Anton, D. Lappa, C. Lieven, M. E. Beber, N. Sonnenschein, E. J. Kerkhoven, and J. Nielsen. A consensus s. cerevisiae metabolic model yeast8 and its ecosystem for comprehensively probing cellular metabolism. *Nature Communications*, 10, 2019. doi: 10.1038/s41467-019-11581-3.

T. Lubitz, J. Hahn, F. T. Bergmann, E. Noor, E. Klipp, and W. Liebermeister. Sbtab: A flexible table format for data exchange in systems biology. *Bioinformatics*, 32:2559–2561, 2016. doi: 10.1093/bioinformatics/btw179.

R. S. Malik-Sheriff, M. Glont, T. V. N. Nguyen, K. Tiwari, M. G. Roberts, A. Xavier, M. T. Vu, J. Men, M. Maire, S. Kananathan, E. L. Fairbanks, J. P. Meyer, C. Arankalle, T. M. Varusai, V. Knight-Schrijver, L. Li, C. Dueñas-Roca, G. Dass, S. M. Keating, Y. M. Park, N. Buso, N. Rodriguez, M. Hucka, and H. Hermjakob. Biomodels—15 years of sharing computational models in life science. *Nucleic Acids Research*, 48:407–415, 2020. doi: 10.1093/nar/gkz1055.

J. K. Medley, A. P. Goldberg, and J. R. Karr. Guidelines for reproducibly building and simulating systems biology models. *IEEE Transactions on Biomedical Engineering*, 63:2015–2020, 2016. doi: 10.1109/TBME.2016.2591960.

A. K. Miller, T. Yu, R. Britten, M. T. Cooling, J. Lawson, D. Cowan, A. Garny, M. D. Halstead, P. J. Hunter, D. P. Nickerson, G. Nunns, S. M. Wimalaratne, and P. M. F Nielsen. Revision history aware repositories of computational models of biological systems. *BMC Bioinformatics*, 12 (1):22, Jan. 2011. ISSN 1471-2105. doi: 10.1186/1471-2105-12-22.

C. J. Norsigian, N. Pusarla, J. L. McConn, J. T. Yurkovich, A. Drager, B. O. Palsson, and Z. King. Bigg models 2020: multi-strain genome-scale models and expansion across the phylogenetic tree. *Nucleic Acids Res*, 48(D1):D402–D406, 2020. doi: 10.1093/nar/gkz1054.

B. G. Olivier and J. L. Snoep. Web-based kinetic modelling using jws online. *Bioinformatics*, 20 (13):2143–4, 2004. ISSN 1367-4803 (Print) 1367-4803 (Linking). doi: 10.1093/bioinformatics/bth200.

S. Pabinger, R. Rader, R. Agren, J. Nielsen, and Z. Trajanoski. Memosys: Bioinformatics platform for genome-scale metabolic models. *BMC Systems Biology*, 5:20, 2011. doi: 10.1186/1752-0509-5-20.

Y. Perez-Riverol, L. Gatto, R. Wang, T. Sachsenberg, J. Uszkoreit, V. Leprevost Fda, C. Fufezan, T. Ternent, S. J. Eglen, D. S. Katz, T. J. Pollard, A. Konovalov, R. M. Flight, K. Blin, and J. A. Vizcaino. Ten simple rules for taking advantage of git and github. *PLoS Comput Biol*, 12(7): e1004947, 2016. doi: 10.1371/journal.pcbi.1004947.

S. Ram and J. Liu. A semantic foundation for provenance management. *Journal on data semantics*, 1:11–17, 2012. doi: 10.1007/s13740-012-0002-0.

A. Ravikrishnan and K. Raman. Critical assessment of genome-scale metabolic networks: the need for a unified standard. *Briefings in Bioinformatics*, 16(6):1057–1068, Feb. 2015. doi: 10.1093/bib/bbv003. URL http://dx.doi.org/10.1093/bib/bbv003.

M. Scharm, O. Wolkenhauer, and D. Waltemath. An algorithm to detect and communicate the differences in computational models describing biological systems. *Bioinformatics*, 32:563–570, 2016. doi: 10.1093/bioinformatics/btv484.

M. Scharm, T. Gebhardt, V. Touré, A. Bagnacani, A. Salehzadeh-Yazdi, O. Wolkenhauer, and D. Waltemath. Evolution of computational models in biomodels database and the physiome model repository. *BMC Systems Biology*, 12:1–10, 2018. doi: 10.1186/s12918-018-0553-2.

F. Schreiber, B. Sommer, T. Czauderna, M. Golebiewski, T. E. Gorochowski, M. Hucka, S. M. Keating, M. Konig, C. Myers, D. Nickerson, and D. Waltemath. Specifications of standards in systems and synthetic biology: status and developments in 2020. *J Integr Bioinform*, 17(2-3), 2020. ISSN 1613-4516 (Electronic) 1613-4516 (Linking). doi: 10.1515/jib-2020-0022.

J. Scott-Brown and A. Papachristodoulou. sbml-diff: A tool for visually comparing sbml models in synthetic biology. *ACS Synthetic Biology*, 6:1225–1229, 2017. doi: 10.1021/acssynbio.6b00273.

S. Soiland-Reyes, P. Sefton, M. Crosas, L. J. Castro, F. Coppens, J. M. Fernández, D. Garijo, B. Grüning, M. La Rosa, S. Leo, E. Carragáin, M. Portier, A. Trisovic, RO-Crate Community, P. Groth, and C. Goble. Packaging research artefacts with ro-crate. *Zenodo*, 2021. doi: 10.5281/ZENODO.5146228. URL https://zenodo.org/record/5146228.

N. J. Stanford, K. Wolstencroft, M. Golebiewski, R. Kania, N. Juty, C. Tomlinson, S. Owen, S. Butcher, H. Hermjakob, N. Le Novère, W. Mueller, J. Snoep, and C. Goble. The evolution of standards and data management practices in systems biology. *Molecular Systems Biology*, 11:851, 2015. doi: 10.15252/msb.20156053.

I. Thiele and B. O. Palsson. A protocol for generating a high-quality genome-scale metabolic reconstruction. *Nat Protoc*, 5(1):93–121, 2010. ISSN 1750-2799 (Electronic) 1750-2799 (Linking). doi: 10.1038/nprot.2009.203.

K. Tiwari, S. Kananathan, M. G. Roberts, J. P. Meyer, M. U. Sharif Shohan, A. Xavier, M. Maire, A. Zyoud, J. Men, S. Ng, T. V. N. Nguyen, M. Glont, H. Hermjakob, and R. S. Malik-Sheriff. Reproducibility in systems biology modelling. *Molecular Systems Biology*, 17:1–7, 2021. doi: 10.15252/msb.20209982.

D. Waltemath, R. Henkel, R. Hälke, M. Scharm, and O. Wolkenhauer. Improving the reuse of computational models through version control. *Bioinformatics*, 29(6):742–748, 01 2013. ISSN 1367-4803. doi: 10.1093/bioinformatics/btt018. URL https://doi.org/10.1093/bioinformatics/btt018.

H. Wang, J. L. Robinson, P. Kocabas, J. Gustafsson, M. Anton, P. E. Cholley, S. Huang, J. Gobom, T. Svensson, M. Uhlen, H. Zetterberg, and J. Nielsen. Genome-scale metabolic network reconstruction of model animals as a platform for translational research. *Proceedings of the National Academy of Sciences*, 118(30), 2021. doi: 10.1073/pnas.2102344118.

M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C. T. Evelo, R. Finkers, A. Gonzalez-Beltran, A. J. G. Gray, P. Groth, C. Goble, J. S. Grethe, J. Heringa, P. A. C. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, S. J. Lusher, M. E. Martone, A. Mons, A. L. Packer, B. Persson, P. Rocca-Serra, M. Roos, R. van Schaik, S.-A. Sansone, E. Schultes, T. Sengstag, T. Slater, G. Strawn, M. A. Swertz, M. Thompson, J. van der Lei, E. van Mulligen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao, and B. Mons. The fair guiding principles for scientific data management and stewardship. *Scientific Data*, 3(1):160018, 2016. ISSN 2052-4463. doi: 10.1038/sdata.2016.18.

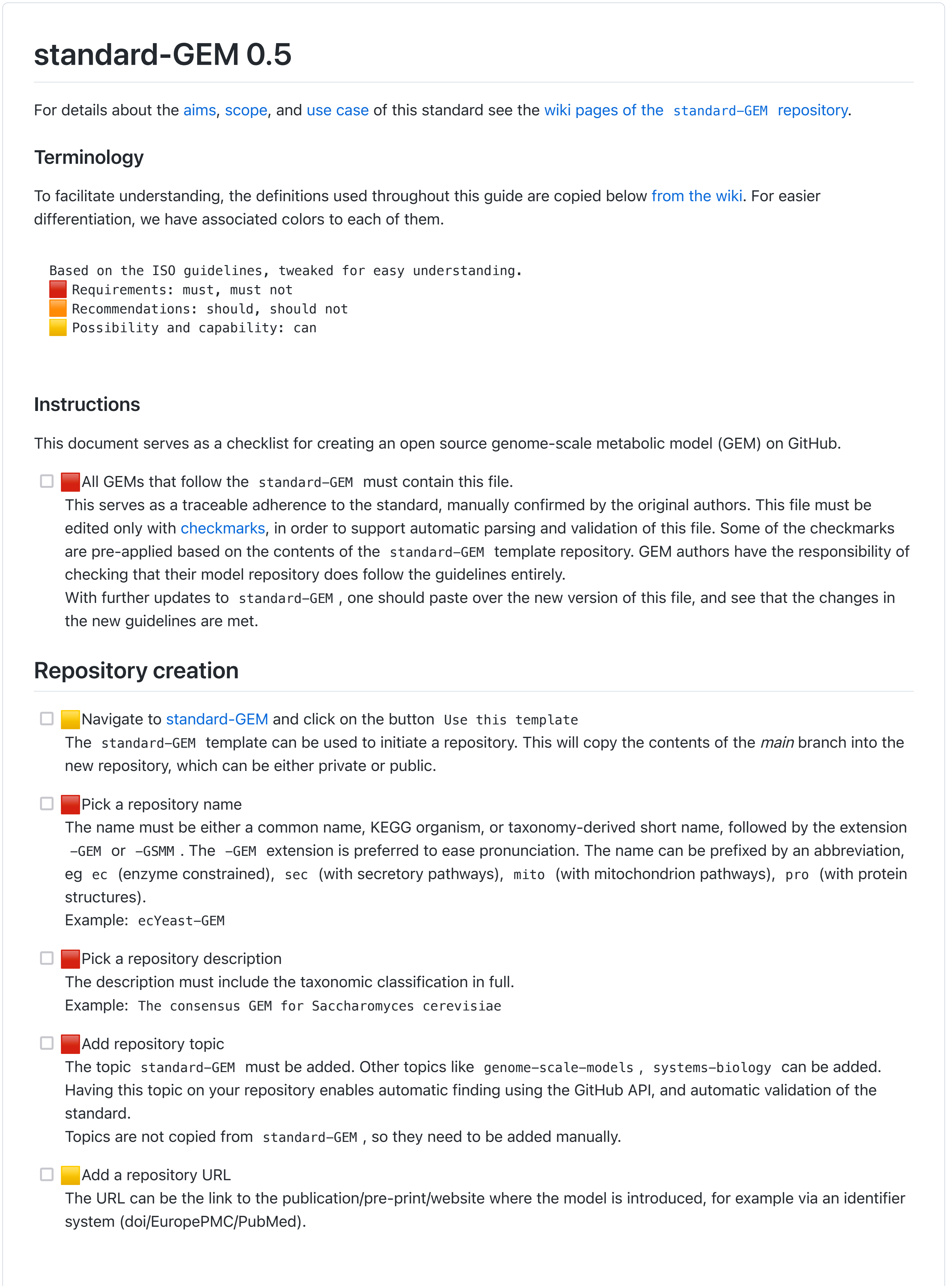
K. Wolstencroft, S. Owen, O. Krebs, Q. Nguyen, N. J. Stanford, M. Golebiewski, A. Weidemann, M. Bittkowski, L. An, D. Shockley, et al. Seek: a systems biology data and model management platform. *BMC Systems Biology*, 9(1):1–12, 2015.

K. Wolstencroft, O. Krebs, J. L. Snoep, N. J. Stanford, F. Bacall, M. Golebiewski, R. Kuzyakiv, Q. Nguyen, O. Stuart, S. Soiland-Reyes, J. Straszewski, D. D. van Niekerk, A. R. Williams, L. Malmström, B. Rinn, W. Müller, and C. Goble. Fairdomhub: a repository and collaboration environment for sharing systems biology research. *Nucleic Acids Research*, 45(D1):D404–D407, 11 2016. doi: 10.1093/nar/gkw1032. URL https://doi.org/10.1093/nar/gkw1032.

J. T. Yurkovich, B. J. Yurkovich, A. Dräger, B. O. Palsson, and Z. A. King. A padawan programmer's guide to developing software libraries. *Cell Systems*, 5:431–437, 2017. doi: 10.1016/j.cels.2017.08.003.

Supplementary Note 1: .standard-GEM.md file

The `standard-GEM` template repository is centred around the *.standard-GEM.md* file. This file is intended to provide guidance similar to a standard operating procedure. In addition to being based on a simplified vocabulary concerning requirements, it also uses colour gradients to distinguish between the three levels of requirements. For user-friendliness, checkboxes have been added to support the user through the open-sourcing process.
Specific terminology has been used in the *.standard-GEM.md* file. The keywords defining the standard-GEM recommendations and requirements are *must*, *must not*, *should*, *should not*, and *can*. These have been based on keyword recommendations by ISO (2016) and RFC2119, and further simplified, favouring the use of *must* / *must not* over *shall* / *shall not* and avoiding the use of *may*, *need not* and *cannot*.
The *.standard-GEM.md* file is currently meant to be maintained manually. While this does include more work than using a programmatic tool which expert users might prefer, it lowers the barrier of entry for new users.
The file below was obtained from github.com/MetabolicAtlas/standard-GEM/blob/main/.standard-GEM.md. As it contains links, we recommend viewing the original file to browse these.



standard-GEM 0.5

For details about the aims, scope, and use case of this standard see the wiki pages of the `standard-GEM` repository.

Terminology

To facilitate understanding, the definitions used throughout this guide are copied below from the wiki. For easier differentiation, we have associated colors to each of them.

```
Based on the ISO guidelines, tweaked for easy understanding.
🟥 Requirements: must, must not
🟧 Recommendations: should, should not
🟨 Possibility and capability: can
```

Instructions

This document serves as a checklist for creating an open source genome-scale metabolic model (GEM) on GitHub.

- [ ] 🟥All GEMs that follow the `standard-GEM` must contain this file.
  This serves as a traceable adherence to the standard, manually confirmed by the original authors. This file must be edited only with checkmarks, in order to support automatic parsing and validation of this file. Some of the checkmarks are pre-applied based on the contents of the `standard-GEM` template repository. GEM authors have the responsibility of checking that their model repository does follow the guidelines entirely.
  With further updates to `standard-GEM`, one should paste over the new version of this file, and see that the changes in the new guidelines are met.

Repository creation

- [ ] 🟨Navigate to standard-GEM and click on the button `Use this template`
  The `standard-GEM` template can be used to initiate a repository. This will copy the contents of the *main* branch into the new repository, which can be either private or public.

- [ ] 🟥Pick a repository name
  The name must be either a common name, KEGG organism, or taxonomy-derived short name, followed by the extension `-GEM` or `-GSMM`. The `-GEM` extension is preferred to ease pronunciation. The name can be prefixed by an abbreviation, eg `ec` (enzyme constrained), `sec` (with secretory pathways), `mito` (with mitochondrion pathways), `pro` (with protein structures).
  Example: `ecYeast-GEM`

- [ ] 🟥Pick a repository description
  The description must include the taxonomic classification in full.
  Example: `The consensus GEM for Saccharomyces cerevisiae`

- [ ] 🟥Add repository topic
  The topic `standard-GEM` must be added. Other topics like `genome-scale-models`, `systems-biology` can be added. Having this topic on your repository enables automatic finding using the GitHub API, and automatic validation of the standard.
  Topics are not copied from `standard-GEM`, so they need to be added manually.

- [ ] 🟨Add a repository URL
  The URL can be the link to the publication/pre-print/website where the model is introduced, for example via an identifier system (doi/EuropePMC/PubMed).

Repository workflow

- [ ] 🟥Git branches
  The GEM repository must have at least two branches: *main* and *develop*.

- [ ] 🟥Releases
  Releases must use the tag format `X.X.X` where X are numbers, according to semantic versioning principles. The last field, also called "patch", can also be used to indicate changes to the repository that do not actually change the GEM itself. The use of a `v` before the version number (`v1.0`) is discouraged. For more information about releases see the documentation at GitHub.

- [ ] 🟨Commits
  Commit messages can follow the style of semantic commits.

File tree

`/` signifies the root of the repository.
`.keep` files are used to indicate that the empty folder should not be ignored by *git* - without it *git* would simply not want to version empty directories. Once folders are not empty, it is okay to remove these files.

- [x] 🟥 `/.gitignore`
  The repository must contain a `/.gitignore` file. This generic .gitignore was prepared for multiple programming languages. While it does not require modification, it can be further adapted to the needs of the repository.

- [x] 🟥 `/.github`
  The repository must contain a `/.github` folder, in which the contributing guidelines, code of conduct, issue templates and pull request templates must be placed. Defaults are provided and they do not require any modification.

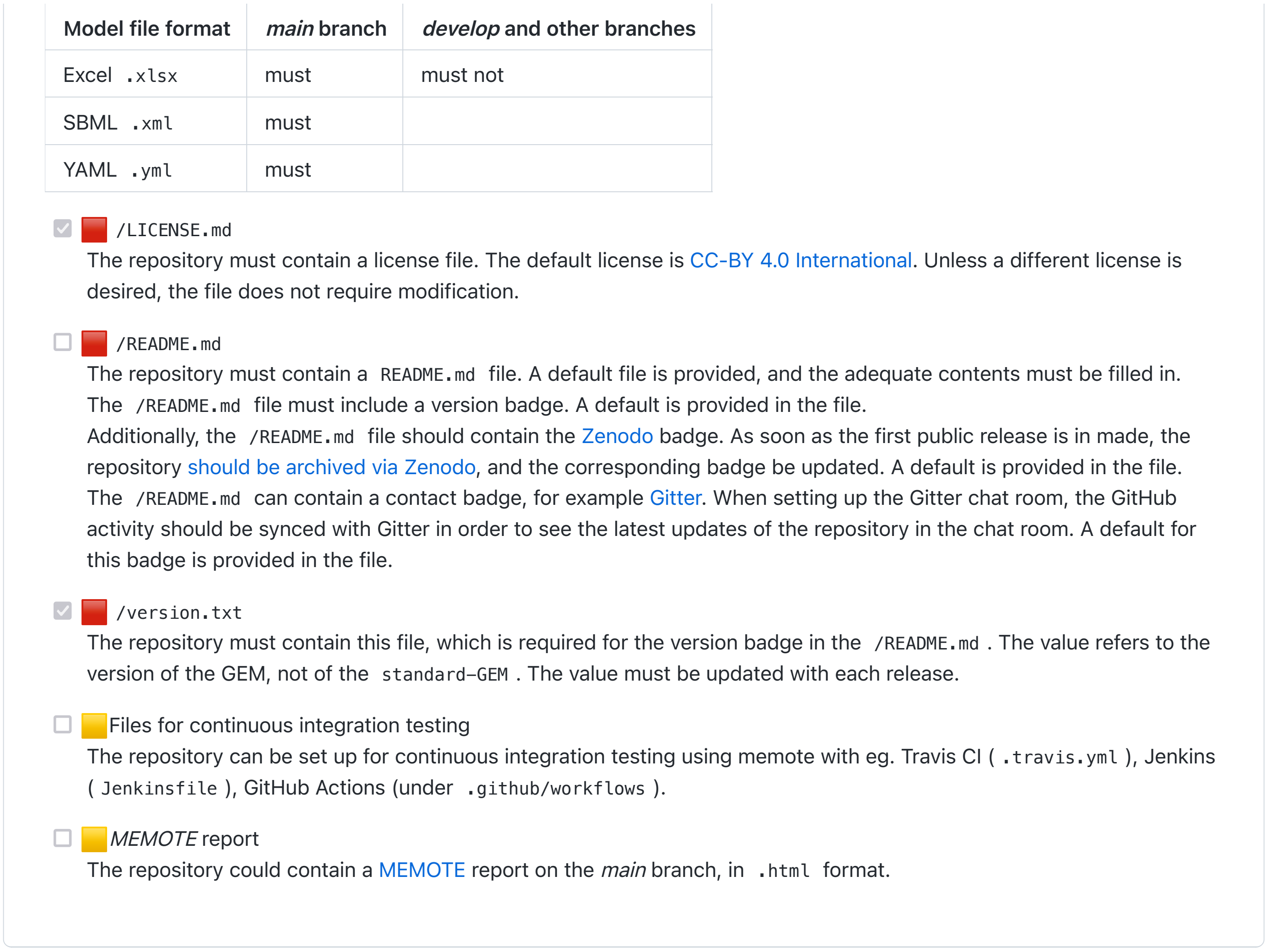
- [ ] 🟥 `/.github/CONTRIBUTING.md`
  This file is provided by the template, but it is empty. It must be filled in with the adequate contributing guideline instructions; a good example is https://github.com/SysBioChalmers/yeast-GEM/blob/main/.github/CONTRIBUTING.md.

- [ ] 🟥 `/code/README.md`
  The repository must contain a `/code` folder. This folder must contain all the code used in generating the model. It must also include a `README.md` file that describes how the folder is organized.

- [ ] 🟥 `/data/README.md`
  The repository must contain a `/data` folder. This folder contains the data used in generating the model. It must also include a `README.md` file that describes how the folder is organized.

- [ ] 🟥 `/model`
  The repository must contain `/model` folder.
  This folder must contain the model files, in multiple formats, according to the table below. As a general guideline, binary formats (`.mat` , `.xlsx`) must not exist on any other branches than *main*. The main reason for this is that binary files cannot be diff'ed, which means changes cannot be compared to previous versions, thus increasing the chance of errors. Moreover, with time, the size of the repository can create difficulties, and we cannot yet recommend storing these files with Git LFS, as it introducs complexity.
  For more information on the `sbtab` file format, see sbtab.net.
  All model files must be named the same as the repository, and with the appropriate extension.
  Example: `yeast-GEM.mat`

Model file format	*main* branch	*develop* and other branches
JSON `.json`	can	
Matlab `.mat`	should	must not
sbtab `.tsv`	can	
Text file `.txt`	must	



Model file format	*main* branch	*develop* and other branches
Excel `.xlsx`	must	must not
SBML `.xml`	must	
YAML `.yml`	must	

- [x] 🟥 `/LICENSE.md`
  The repository must contain a license file. The default license is CC-BY 4.0 International. Unless a different license is desired, the file does not require modification.

- [ ] 🟥 `/README.md`
  The repository must contain a `README.md` file. A default file is provided, and the adequate contents must be filled in.
  The `/README.md` file must include a version badge. A default is provided in the file.
  Additionally, the `/README.md` file should contain the Zenodo badge. As soon as the first public release is in made, the repository should be archived via Zenodo, and the corresponding badge be updated. A default is provided in the file.
  The `/README.md` can contain a contact badge, for example Gitter. When setting up the Gitter chat room, the GitHub activity should be synced with Gitter in order to see the latest updates of the repository in the chat room. A default for this badge is provided in the file.

- [x] 🟥 `/version.txt`
  The repository must contain this file, which is required for the version badge in the `/README.md`. The value refers to the version of the GEM, not of the `standard-GEM`. The value must be updated with each release.

- [ ] 🟨 Files for continuous integration testing
  The repository can be set up for continuous integration testing using memote with eg. Travis CI (`.travis.yml`), Jenkins (`Jenkinsfile`), GitHub Actions (under `.github/workflows`).

- [ ] 🟨 *MEMOTE* report
  The repository could contain a MEMOTE report on the *main* branch, in `.html` format.