MEMOTE for standardized genome-scale metabolic model testing

To the Editor — Reconstructing metabolic reaction networks enables the development of testable hypotheses of an organism’s metabolism under different conditions. State-of-the-art genome-scale metabolic models (GEMs) can include thousands of metabolites and reactions that are assigned to subcellular locations. Gene–protein–reaction (GPR) rules and annotations using database information can add meta-information to GEMs. GEMs with metadata can be built using standard reconstruction protocols, and guidelines have been put in place for tracking provenance and enabling interoperability, but a standardized means of quality control for GEMs is lacking. Here we report a community effort to develop a test suite named MEMOTE (for metabolic model tests) to assess GEM quality.

Incompatible description formats and missing annotations limit GEM reuse. Moreover, numerical errors and omission of essential cofactors limit model predictions untrustworthy. Every year, increasing numbers of manually curated and automatically generated GEMs are published, including those for human and cancer tissue models. We believe that it is essential to optimize GEM reproducibility and reuse. Researchers need models that are software-agnostic, with components that have standardized, database-independent identifiers. Default conditions and mathematically specified modeling formulations must be precisely defined to allow reproduction of the original model predictions. Models must produce feasible phenotypes under various conditions. Finally, data used to build any model must be made available in a reusable format.

A dual approach could be used to improve GEM reuse and reproducibility. First, we advocate adoption of the latest version of the Systems Biology Markup Language (SBML) level 3 flux balance constraints (SBML3FBC) package as the primary description and exchange format. The SBML3FBC package adds structured, semantic descriptions for domain-specific model components such as flux bounds, multiple linear objective functions, GPR rules, metabolite chemical formulas, charge and annotations. The SBML and constraint-based modeling communities collaboratively develop this package, updating it based on user input. It has been adopted by a wide range of constraint-based modeling software and public model repositories (http://cbmpy.sourceforge.net/ and refs. [10–15]), and should therefore be considered the standard for encoding GEMs.

Second, we present MEMOTE (/’mi:moʊt/ in international phonetic alphabet notation), an open-source Python software that represents a unified approach to ensure the formally correct definition of SBML3FBC and provides quality control and continuous quality assurance of metabolic models with tools and best practices already used in software development. MEMOTE accepts stoichiometric models encoded in SBML3FBC and previous versions as input. In addition to structural validation analogous to the SBML validator, MEMOTE benchmarks metabolic models using consensus tests from four general areas: annotation, basic tests, biomass reaction and stoichiometry.

Annotation tests check that a model is annotated according to community standards with minimum information required in annotation of models (MIRIAM)-compliant cross-references, that all primary identifiers belong to the same namespace rather than being fractured across several namespaces, and that components are described using Systems Biology Ontology (SBO) terms. A lack of explicit, standardized annotations complicates the use, comparison and extension of GEMs, and thus strongly hampers collaboration.

Basic tests check the formal correctness of a model and verify the presence of components such as metabolites, compartments, reactions and genes. These tests also check for metabolite formula and charge information, and GPR rules. General quality metrics, such as the degree of metabolic coverage representing the ratio of reactions and genes, are also checked.

A model is tested for production of biomass precursors in different conditions, for biomass consistency, for nonzero growth rate and for direct precursors. The biomass reaction is based on the biomass composition of the modeled organism and expresses its ability to produce the necessary precursors for in silico cell growth and maintenance. Thus, an extensive, well-formed biomass reaction is crucial for accurate predictions with a GEM.

Stoichiometric inconsistency, erroneously produced energy metabolites and permanently blocked reactions are identified by MEMOTE. Errors in stoichiometries may result in the production of ATP or redox cofactors from nothing and are detrimental to the performance of the model when using flux-based analyses.

MEMOTE enables a quick comparison of any two given models, in which individual test results are quantified and condensed to calculate an overall score (Supplementary Note 1). In addition to these consensus tests, researchers can supply experimental data from growth and gene perturbation studies in a range of input formats (.csv, .tsv, .xls or .xlsx) in MEMOTE. To support reproducibility, researchers can configure MEMOTE to recognize specific data types as input to predefined experimental tests for model validation (Supplementary Note 2).

There are two main workflows for MEMOTE (Fig. 1a and Supplementary Figs. 1–3). For peer review, MEMOTE can produce either a ‘snapshot report’ or a ‘diff report’ that display MEMOTE test results of one single or multiple models, respectively. For model reconstruction, MEMOTE helps users to create a version-controlled repository of the model and to activate continuous integration toward building a ‘history report’ that records the results of each tracked edit of the model. Although a model repository can be used offline, we encourage community collaboration via distributed version control development platforms, such as GitHub (https://github.com), GitLab (https://gitlab.com) or BioModels (http://www.ebi.ac.uk/biomodels/). MEMOTE is tightly integrated with GitHub. Models generated and versioned in MEMOTE can easily be uploaded to GitLab and BioModels. Collaborative model reconstruction with MEMOTE as benchmark can occur using all three software platforms (Fig. 1b).

We validated MEMOTE using models from seven GEM collections (Fig. 2, Supplementary Table 1 and Supplementary Methods), that comprise manually and (semi)-automatically reconstructed GEMs (10,780 models in total). Most GEM collections have already made models available in SBML format. A nonlinear dimensional reduction of the normalized test results (Supplementary Methods) using t-distributed stochastic neighbor embedding
model. For this purpose, the model is written in a sorted YAML format after every change. Bold blue happens in the form of a comparative 'diff' report, whereas for file-based discrepancies MEMOTE gatekeepers, choosing to accept only high-quality contributions. Identification of functional differences external experts to improve specific model features. MEMOTE further enables model authors to act as which model builders could use to curate different parts of the model simultaneously or to invite.

This serves as a guide toward a functional, high-quality GEM (Supplementary Fig. 3). This workflow is accessible through the command line only. Both, Github and GitLab support a branching strategy, b

Fig. 1 Graphical summary of MEMOTE. a, Graphical representation of the two principal workflows in detail. For peer review, MEMOTE serves as a benchmark tool generating a comprehensive, human-readable report, which quantifies the model’s performance (Supplementary Figs. 1 and 2). With this information, a definitive assessment of model quality can be made by editors, reviewers and users. This workflow is accessible through a web interface (https://memote.io) or locally through a command line interface. For model reconstruction, MEMOTE helps users to create a version-controlled repository for the model (indicated by the blue asterisk), and to activate continuous integration. The model is tested using MEMOTE’s library of test cases, the results are saved, and an initial report of the model is generated. This constitutes the first iteration of the development cycle. Now, users may edit the model using their preferred reconstruction tool and subsequently export it to SBML3FBc, thus creating a generated. This constitutes the first iteration of the development cycle. Now, users may edit the model using their preferred reconstruction tool and subsequently export it to SBML3FBc, thus creating a generated. This constitutes the first iteration of the development cycle. Now, users may edit the model using their preferred reconstruction tool and subsequently export it to SBML3FBc, thus creating a generated. This constitutes the first iteration of the development cycle. Now, users may edit the model using their preferred reconstruction tool and subsequently export it to SBML3FBc, thus creating a generated. This constitutes the first iteration of the development cycle. Now, users may edit the model using their preferred reconstruction tool and subsequently export it to SBML3FBc, thus creating a generated. This constitutes the first iteration of the development cycle. Now, users may edit the model using their preferred reconstruction tool and subsequently export it to SBML3FBc, thus creating a generated. This constitutes the first iteration of the development cycle. Now, users may edit the model using their preferred reconstruction tool and subsequently export it to SBML3FBc, thus creating a generated.
and 11), and partial overlaps for models from KBase and BiGG (Supplementary Figs. 12–16), or AGORA and BiGG (Supplementary Figs. 17 and 18), but not KBase and AGORA. BiGG is the only collection with models using MetaNetX\(^{27}\) annotations (Supplementary Fig. 19). MetaNetX consolidates biochemical namespaces by establishing a mapping between them through a set of unique identifiers. Hence, knowing the MetaNetX identifier for a given entity often means also knowing the identifiers for other databases (Supplementary Methods).

MEMOTE tests cover semantic and conceptual requirements, which are fundamental to SBML3FBC and constraint-based modeling, respectively. They are extensible to allow the validation of a model’s performance against experimental data and can be executed as a stand-alone tool or integrated into existing reconstruction pipelines. Capitalizing on robust workflows established in modern software development, MEMOTE promotes openness and collaboration by granting the community tangible metrics to support their research and to discuss assumptions or limitations openly.

Application of a set of defined metabolic model tests is not dependent on implementation in MEMOTE, and for some users it may be more desirable to implement each test separately to streamline the user experience.

We propose that an independent, central testing service is an alternative to quality control because the tests are continuously reviewed by the community. This resource will be maintained under stewardship of Nikolaus Sonnenschein by the openCOBRA consortium (https://github.com/openCOBRA). To encourage integration as opposed to duplication, MEMOTE provides a Python application programming interface (API) as well as being available as a web service. MEMOTE has already been integrated in several services and tools (Supplementary Note 3). We discuss alternatives and future perspectives of MEMOTE in Supplementary Notes 4 and 5, respectively.

We recommend that MEMOTE users reach out to GEM authors to report any errors and thereby enable community improvement of models as resources. Using inconsistent GEMs for hypothesis generation could lead researchers down blind alleys, so we weighed the influence of ‘consistency’ and ‘stoichiometric consistency’ and SBO terms higher than tests for metabolite, reaction and gene annotations.

We are committed to keeping MEMOTE open to support community principles. Robust benchmarking will only work if it is actively supported by the whole community, and we call on any interested experts to join this endeavor and enable its continual improvement.

**Data availability**

The model collection is available at https://doi.org/10.5281/zenodo.2636858. Individual results and aggregated tables, as well as analysis code, are available at https://doi.org/10.5281/zenodo.2638234.

**Code availability**

MEMOTE source code is available at https://github.com/opencobra/memote under the Apache license, version 2.0. Supporting documentation is available at https://memote.readthedocs.io. The MEMOTE web interface is hosted at https://memote.io. A detailed list of all tests in MEMOTE is available at https://memote.readthedocs.io/en/latest/autoapi/index.html.
The nf-core framework for community-curated bioinformatics pipelines

To the Editor — The standardization, portability and reproducibility of analysis pipelines are key issues within the bioinformatics community. Most bioinformatics pipelines are designed for use on-premises; as a result, the associated software dependencies and execution logic are likely to be tightly coupled with proprietary computing environments. This can make it difficult or even impossible for others to reproduce the ensuing results, which is a fundamental requirement for the validation of scientific findings. Here, we introduce the nf-core framework as a means for the development of collaborative, peer-reviewed, best-practice analysis pipelines (Fig. 1). All nf-core pipelines are written in Nextflow and so inherit the ability to be executed on most computational infrastructures, as well as having native support for container technologies such as Docker and Singularity. The nf-core community (Supplementary Fig. 1) has developed a suite of tools that automate pipeline creation, testing, deployment and synchronization. Our goal is to provide a framework for high-quality bioinformatics pipelines that can be used across all institutions and research facilities.

Being able to reproduce scientific results is the central tenet of the scientific method. However, moving toward FAIR (findable, accessible, interoperable and reusable) research methods in data-driven science is complex. Central repositories, such as bio.tools, omictools and the Galaxy toolshed, make it possible to find existing pipelines and their associated tools. However, it is still notoriously challenging to develop analysis pipelines that are fully reproducible and interoperable across multiple systems and institutions — primarily because of differences in hardware, operating systems and software versions.

Although the recommended guidelines for some analysis pipelines have become standardized (for example, GATK best practices), the actual implementations are usually developed on a case-by-case basis. As such, there is often little incentive to test, document and implement pipelines in a way that permits their reuse by other researchers. This can hamper sustainable sharing of data and tools, and results in a proliferation of heterogeneous analysis pipelines, making it difficult for newcomers to find what they need to address a specific analysis question.

As the scale of -omics data and their associated analytical tools has grown, the scientific community is increasingly moving toward the use of specialized workflow management systems to build analysis pipelines. They separate the requirements of the underlying compute infrastructure from the analysis and workflow description, introducing a higher degree of portability as compared to custom in-house scripts. One such popular tool is Nextflow. Using Nextflow, software packages can be bundled with analysis pipelines using built-in integration for package managers, such as Conda, and containerization platforms, such as Docker and Singularity. Moreover, support for most common high-performance-computing batch schedulers and cloud providers allows simple deployment of analysis pipelines on almost any infrastructure. The opportunity to run pipelines locally during initial development and then to proceed seamlessly to large-scale computational resources in high-performance-computing or cloud settings provides users and developers with great flexibility.

The nf-core community project collects a curated set of best-practice analysis pipelines built using Nextflow. Similar projects include the ‘awesome-pipelines’ repository, which provides an extensive list of pipelines developed by the Nextflow community (https://github.com/pditommaso/awesome-pipeline), although these pipelines are variable in terms of development status and design. High-level approaches to facilitate the creation of end-to-end analysis pipelines are also available: Flowcraft (https://github.com/assembleflow/flowcraft) and Pipeliner are
Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Research policies, see Authors & Referees and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

- The exact sample size (n) for each experimental group/condition, given as a discrete number and unit of measurement
- A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
- The statistical test(s) used AND whether they are one- or two-sided
- Only common tests should be described solely by name; describe more complex techniques in the Methods section.
- A description of all covariates tested
- A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
- A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
- For null hypothesis testing, the test statistic (e.g. F, t, r) with confidence intervals, effect sizes, degrees of freedom and P value noted
- Give P values as exact values whenever suitable.
- For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
- For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
- Estimates of effect sizes (e.g. Cohen's d, Pearson's r), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection

Data (models) was download from publicly available websites.

Data analysis

All code relevant to the manuscript is available at https://github.com/opencobra/memote, https://github.com/opencobra/memote-webserver, https://github.com/opencobra/cookiecutter-memote, https://github.com/opencobra/memote-docker, and https://github.com/biosustain/memote-meta-study/. At submission of the revised manuscript, memote had version 0.9.6 and cobrapy version 0.14.2 (the versions of all software dependencies).

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors/reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

The collected data (metabolic models) and code behind Figure 4 are available at https://github.com/biosustain/memote-meta-study/ and furthermore deposited to zenodo.org (https://doi.org/10.5281/zenodo.2638233)
Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

| Sample size | We aimed for the broadest possible assessment of publicly available genome-scale metabolic models. Sample size was therefore determined by the availability of data. |
|-------------|-------------------------------------------------------------------------------------------------|
| Data exclusions | Models that could not be parsed by memote because they were either not available in SBML or because the SBML files were invalid have been excluded from the study. |
| Replication | Not applicable. |
| Randomization | Not applicable. |
| Blinding | Not applicable. |

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

| Materials & experimental systems | Methods |
|--------------------------------|---------|
| n/a | Involved in the study |
| ☒ | Antibodies |
| ☒ | Eukaryotic cell lines |
| ☒ | Palaeontology |
| ☒ | Animals and other organisms |
| ☒ | Human research participants |
| ☒ | Clinical data |
| n/a | Involved in the study |
| ☒ | ChIP-seq |
| ☒ | Flow cytometry |
| ☒ | MRI-based neuroimaging |