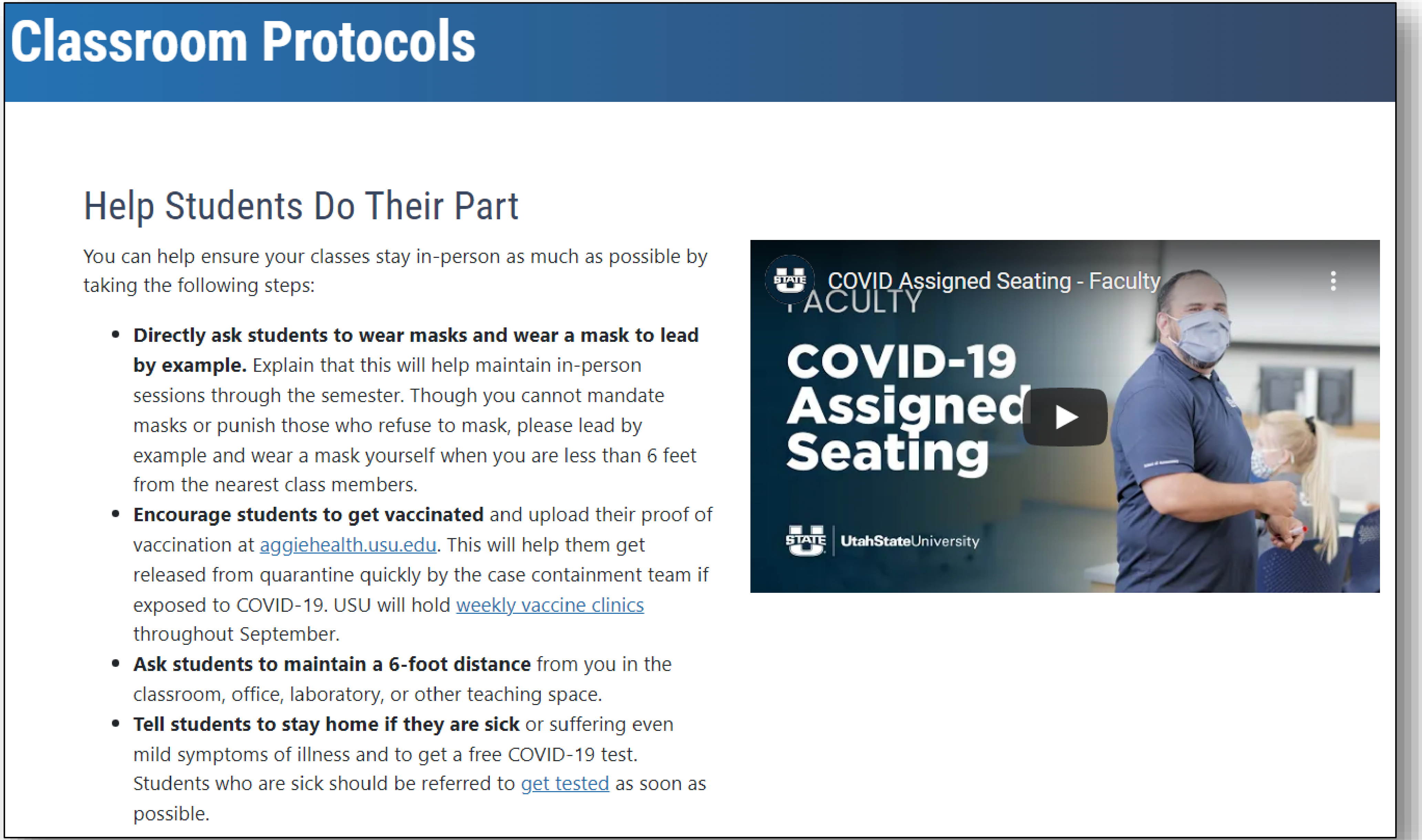
# Introduction to Systems Biology: Constraint-based Metabolic Reconstructions & Analysis



# Classroom Protocols

## Help Students Do Their Part

You can help ensure your classes stay in-person as much as possible by taking the following steps:

- **Directly ask students to wear masks and wear a mask to lead by example.** Explain that this will help maintain in-person sessions through the semester. Though you cannot mandate masks or punish those who refuse to mask, please lead by example and wear a mask yourself when you are less than 6 feet from the nearest class members.
- **Encourage students to get vaccinated** and upload their proof of vaccination at aggiehealth.usu.edu. This will help them get released from quarantine quickly by the case containment team if exposed to COVID-19. USU will hold weekly vaccine clinics throughout September.
- **Ask students to maintain a 6-foot distance** from you in the classroom, office, laboratory, or other teaching space.
- **Tell students to stay home if they are sick** or suffering even mild symptoms of illness and to get a free COVID-19 test. Students who are sick should be referred to get tested as soon as possible.

# Lecture Learning Objectives

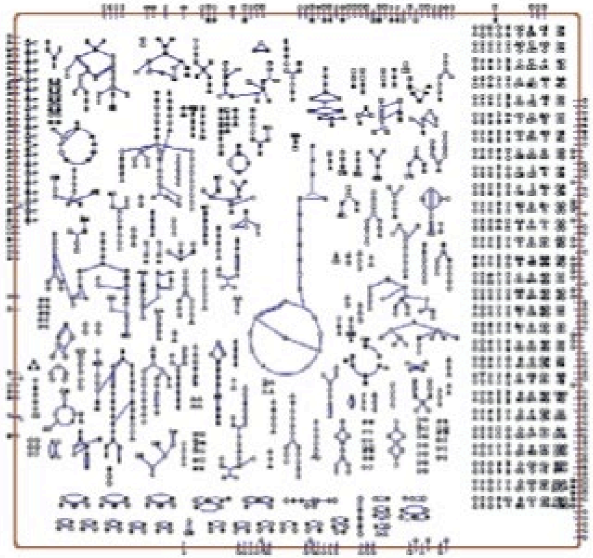
Each student should be able to:

- Explain the strengths and limitations of constraint-based modeling
- Understand how to use the course website
- Explain the course expectations
- Explain the grading process

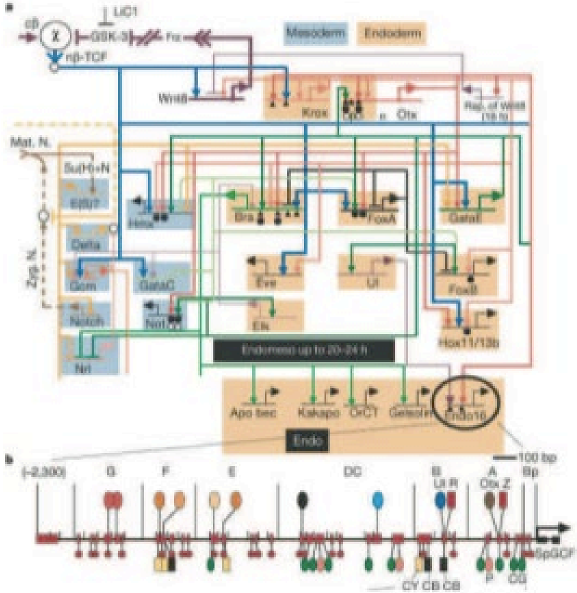
# Course Introduction

- Content Overview
- Course Website
- Course Learning Process
- Course Grading & Expectations

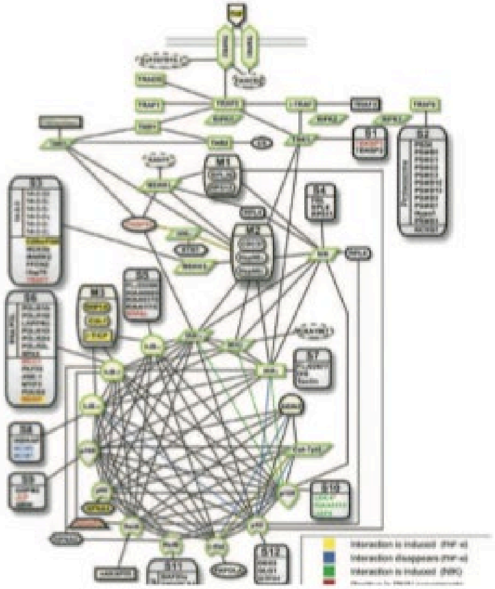
# Types of Biological Networks



Metabolism



Regulation

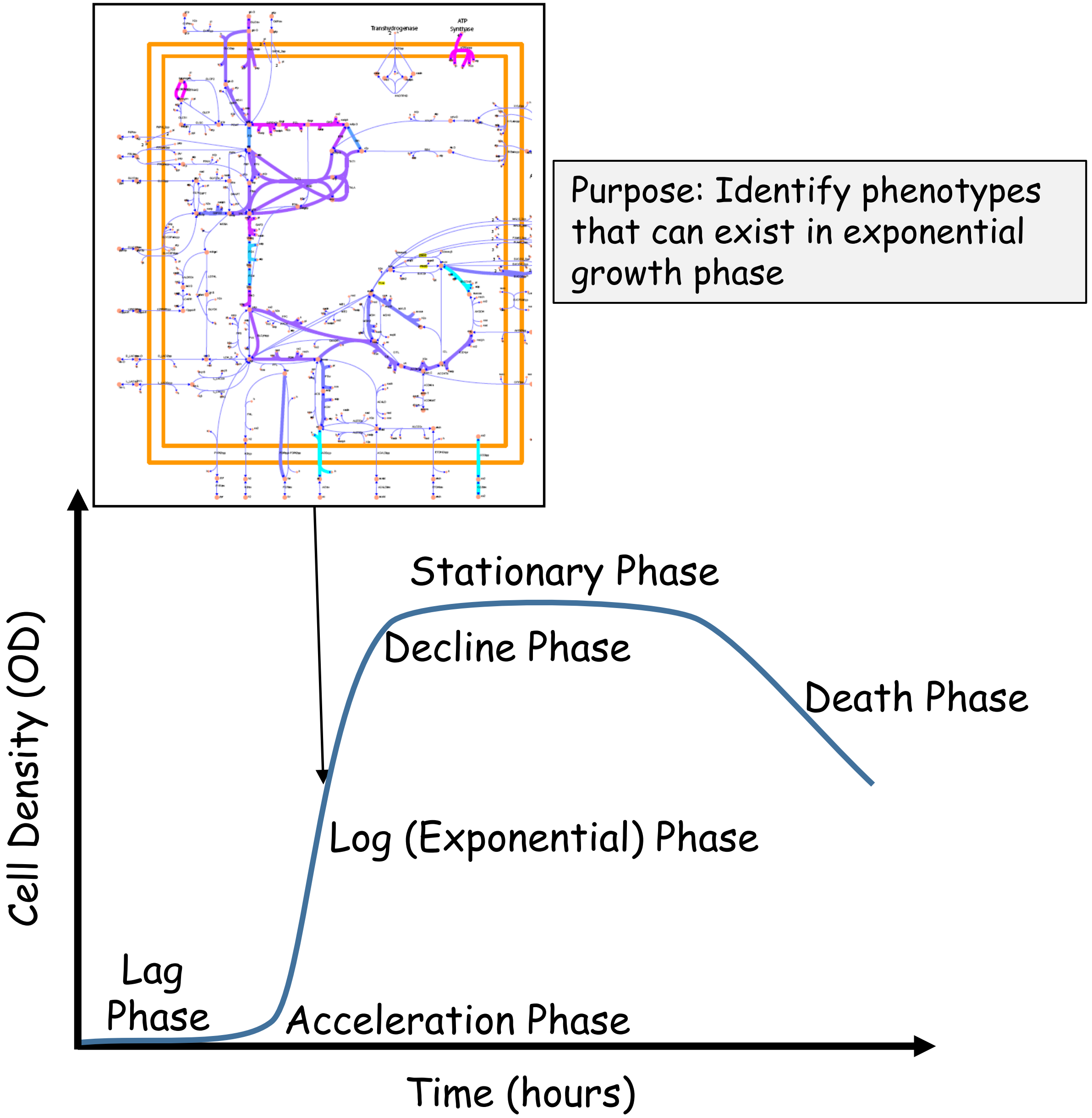


Signaling

B. Palsson, Lectures from Systems Biology: Simulation of Dynamic Network States, Chapter 1

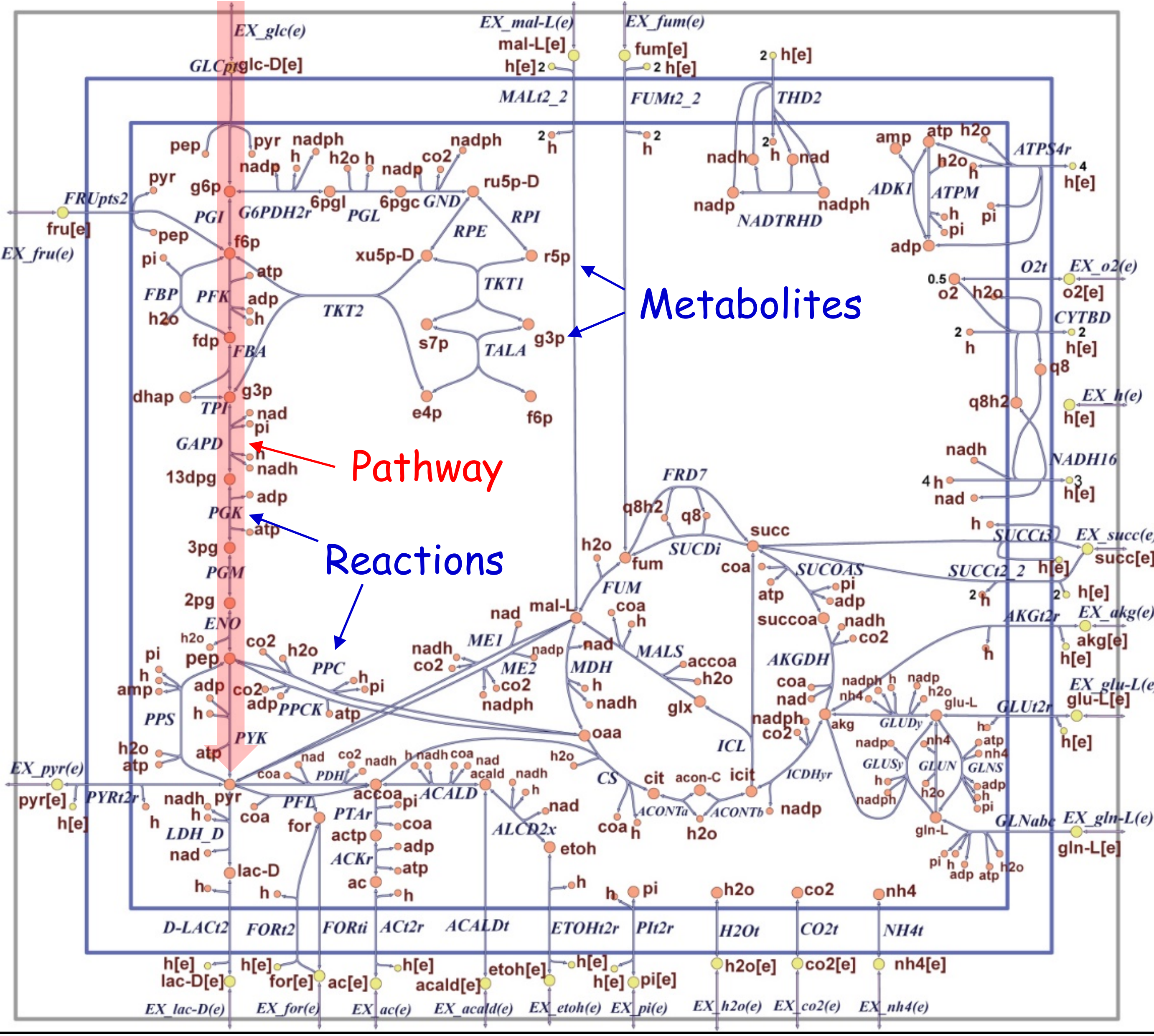
# Constraint-based Modeling

- Model cell steady-state phenotypes during exponential growth phase.
    - ✓ Can model the different phenotypes that can exist during the exponential growth phase.
    - ✓ Can understand the capabilities of each phenotype
    - ✓ Can identify and modify cellular pathways to favor specific bioproduct producing phenotypes
    - ✓ Constraint-based models do not model transitions between phenotypes
    - ✓ Most genome-scale models do not include the genes required for the stationary phase (proteases, etc.)
    - ✓ Most genome-scale models do not include the complete transcription and translation pathways
- The biomass function represents the average metabolic load required during exponential cell growth.
    - ✓ The biomass function represents the average percentages of the component parts (amino acids, nucleotides, energy, etc.) that are included in 1 gm of cell biomass.



Purpose: Identify phenotypes that can exist in exponential growth phase

# Metabolic Models



Metabolites

Pathway

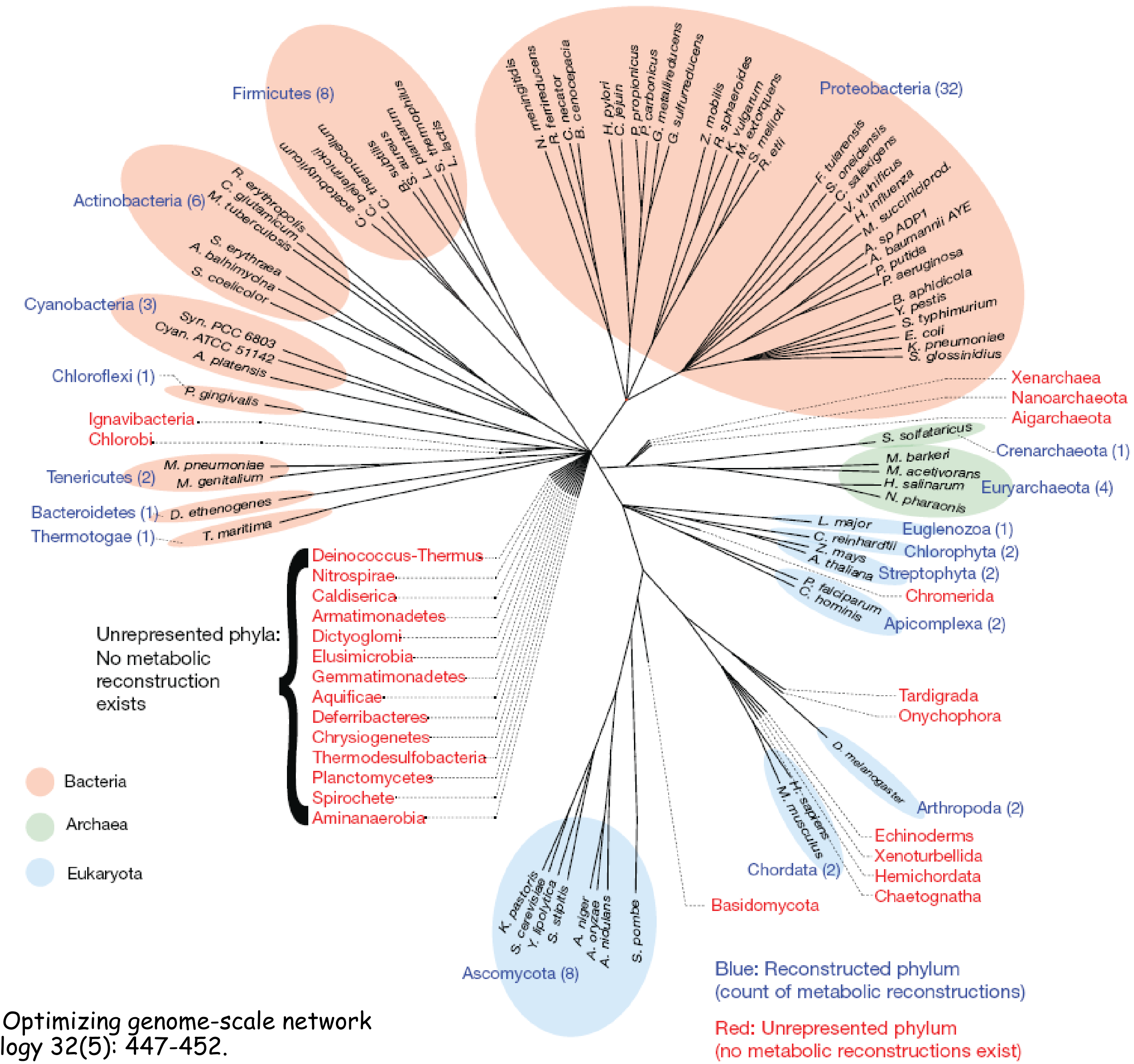
Reactions

Orth, J. D., I. Thiele, et al. (2010). "What is flux balance analysis?" Nature biotechnology **28(3): 245-248.**

# Metabolic Reconstructions

A GEnome scale Network Reconstructions (GENREs) serves as a structured knowledge base of established biochemical facts, while a GEnome scale Models (GEMs) is a model which supplements the established biochemical information with additional (potentially hypothetical) information to enable computational simulation and analysis.

Metabolic reconstructions for 78 species across the tree of life

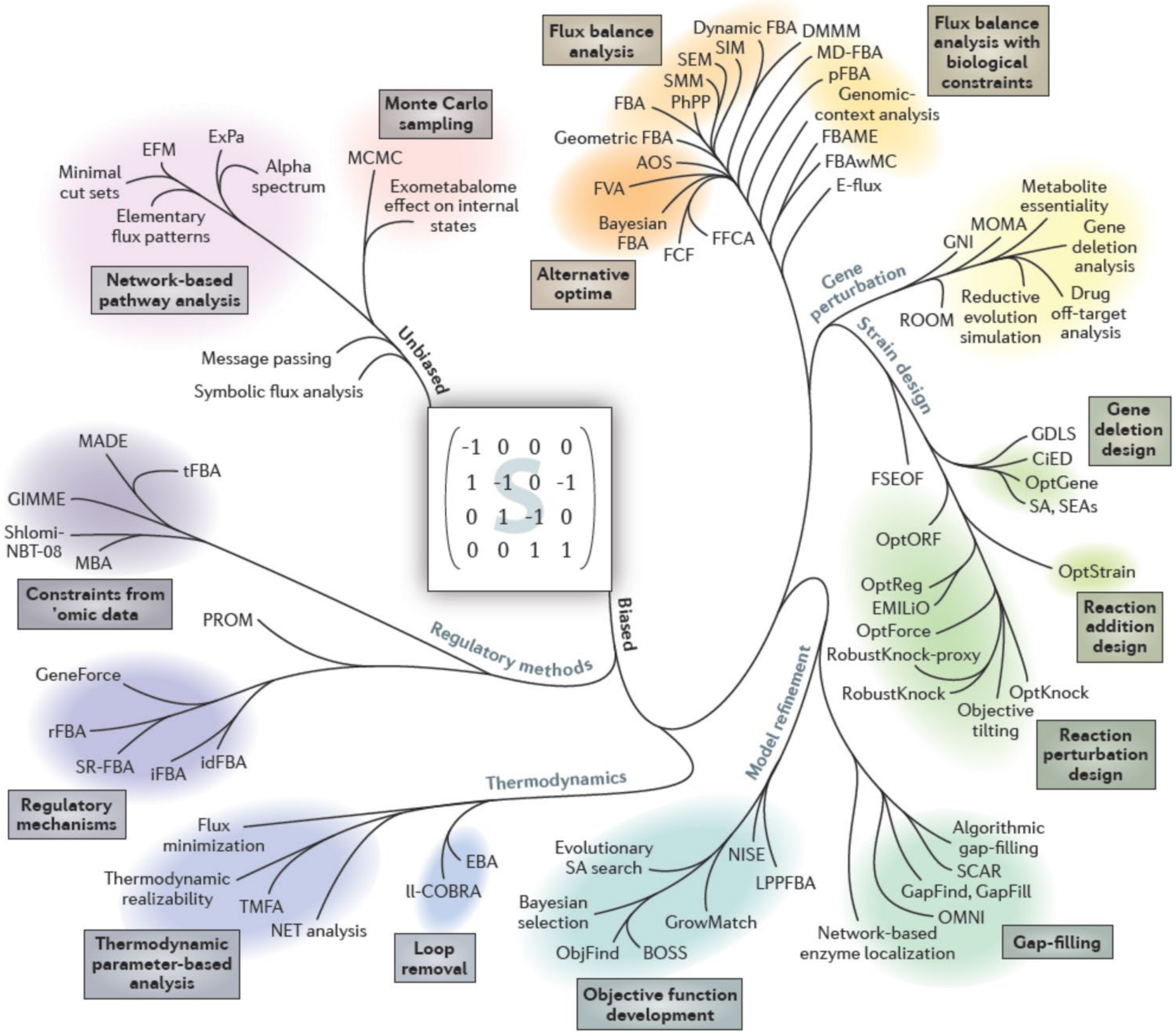


Blue: Reconstructed phylum (count of metabolic reconstructions)

Red: Unrepresented phylum (no metabolic reconstructions exist)

Monk, J., J. Nogales, et al. (2014). "Optimizing genome-scale network reconstructions." Nature biotechnology 32(5): 447-452.

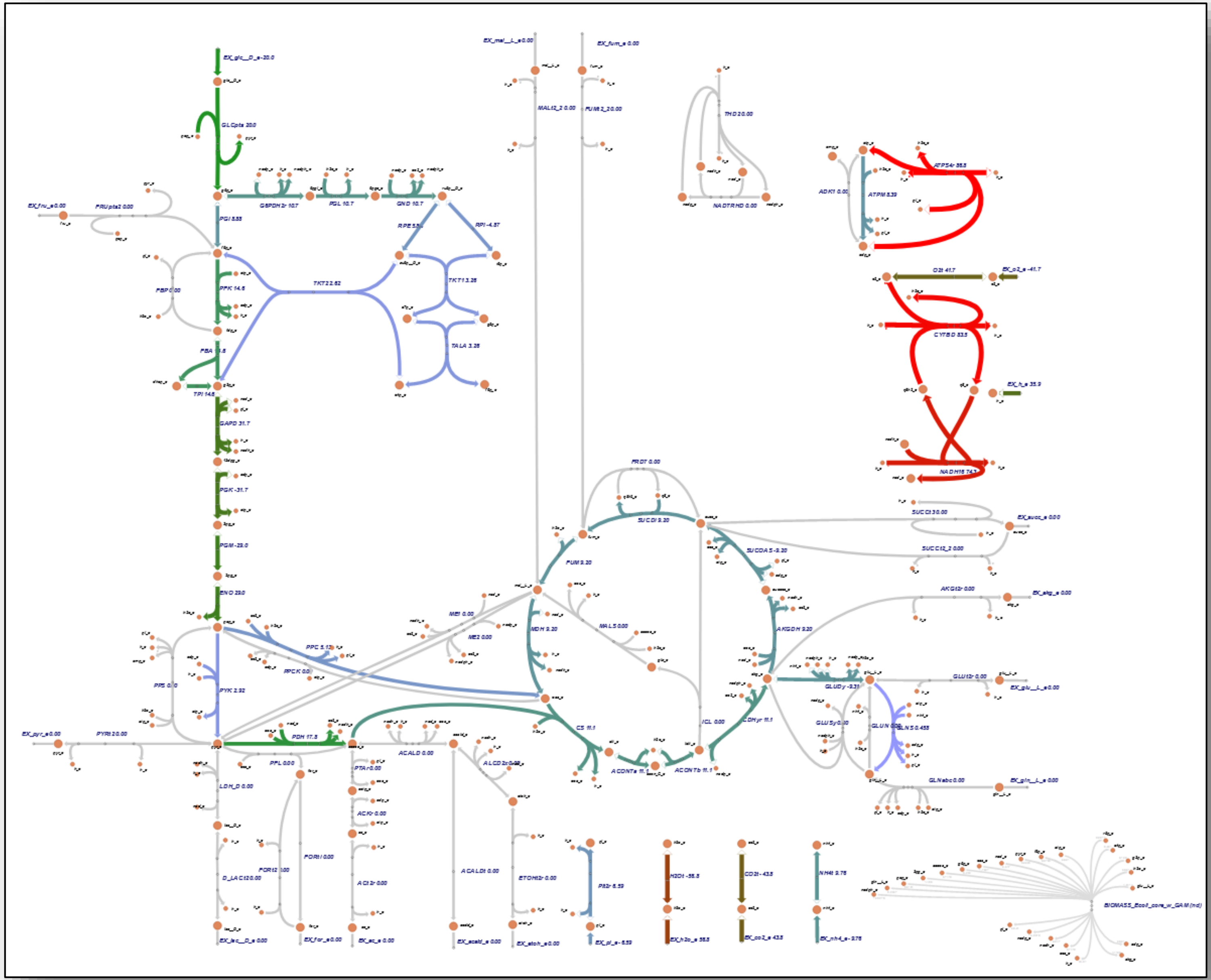
# The 'Phylogeny' of Constraint-based Modeling Methods



Lewis, N. E., H. Nagarajan, et al. (2012). "Constraining the metabolic genotype-phenotype relationship using a phylogeny of in silico methods." Nature reviews. Microbiology 10(4): 291-305.

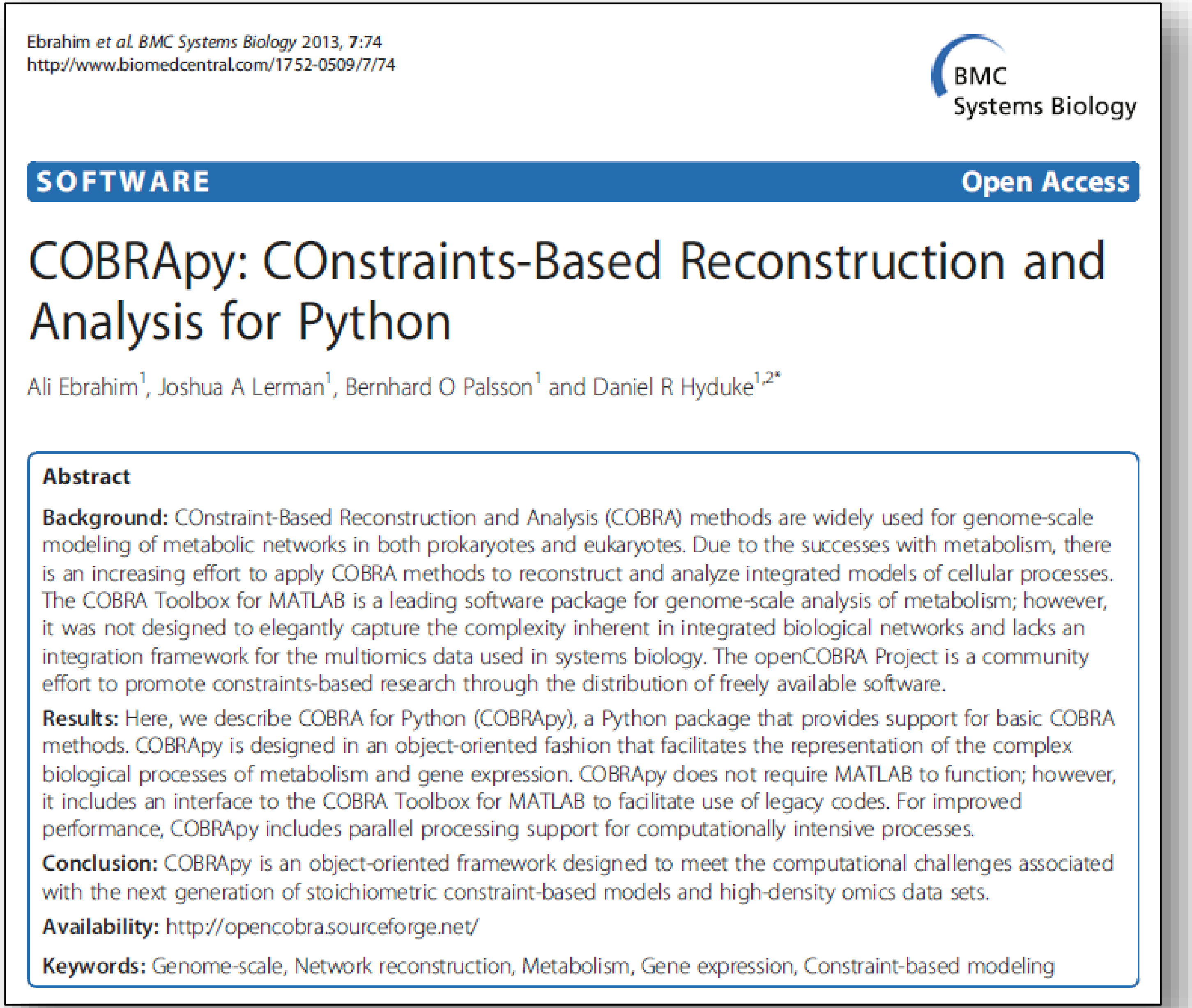
# Course Content

- Course Introduction
- COBRA Models
- Flux Balance Analysis
- *E.coli* Core Metabolic Model
- Flux Variability Analysis
- Randomized Sampling
- Model Interrogation
- Model Creation/Enhancement
- Production Envelopes
- Gene/Reaction Knockout Strategies
- Gene/Reaction Modulation
- Strain Design
- Advanced Topics



# COBRApy Toolbox Paper

Ebrahim, A., Lerman, J.A., Palsson, B.O. et al. COBRApy: COnstraints-Based Reconstruction and Analysis for Python. BMC Syst Biol 7, 74 (2013). https://doi.org/10.1186/1752-0509-7-74



Ebrahim *et al. BMC Systems Biology* 2013, **7**:74
http://www.biomedcentral.com/1752-0509/7/74

BMC Systems Biology

**SOFTWARE** **Open Access**

## COBRApy: COnstraints-Based Reconstruction and Analysis for Python

Ali Ebrahim[1], Joshua A Lerman[1], Bernhard O Palsson[1] and Daniel R Hyduke[1,2*]

**Abstract**

**Background:** COnstraint-Based Reconstruction and Analysis (COBRA) methods are widely used for genome-scale modeling of metabolic networks in both prokaryotes and eukaryotes. Due to the successes with metabolism, there is an increasing effort to apply COBRA methods to reconstruct and analyze integrated models of cellular processes. The COBRA Toolbox for MATLAB is a leading software package for genome-scale analysis of metabolism; however, it was not designed to elegantly capture the complexity inherent in integrated biological networks and lacks an integration framework for the multiomics data used in systems biology. The openCOBRA Project is a community effort to promote constraints-based research through the distribution of freely available software.

**Results:** Here, we describe COBRA for Python (COBRApy), a Python package that provides support for basic COBRA methods. COBRApy is designed in an object-oriented fashion that facilitates the representation of the complex biological processes of metabolism and gene expression. COBRApy does not require MATLAB to function; however, it includes an interface to the COBRA Toolbox for MATLAB to facilitate use of legacy codes. For improved performance, COBRApy includes parallel processing support for computationally intensive processes.

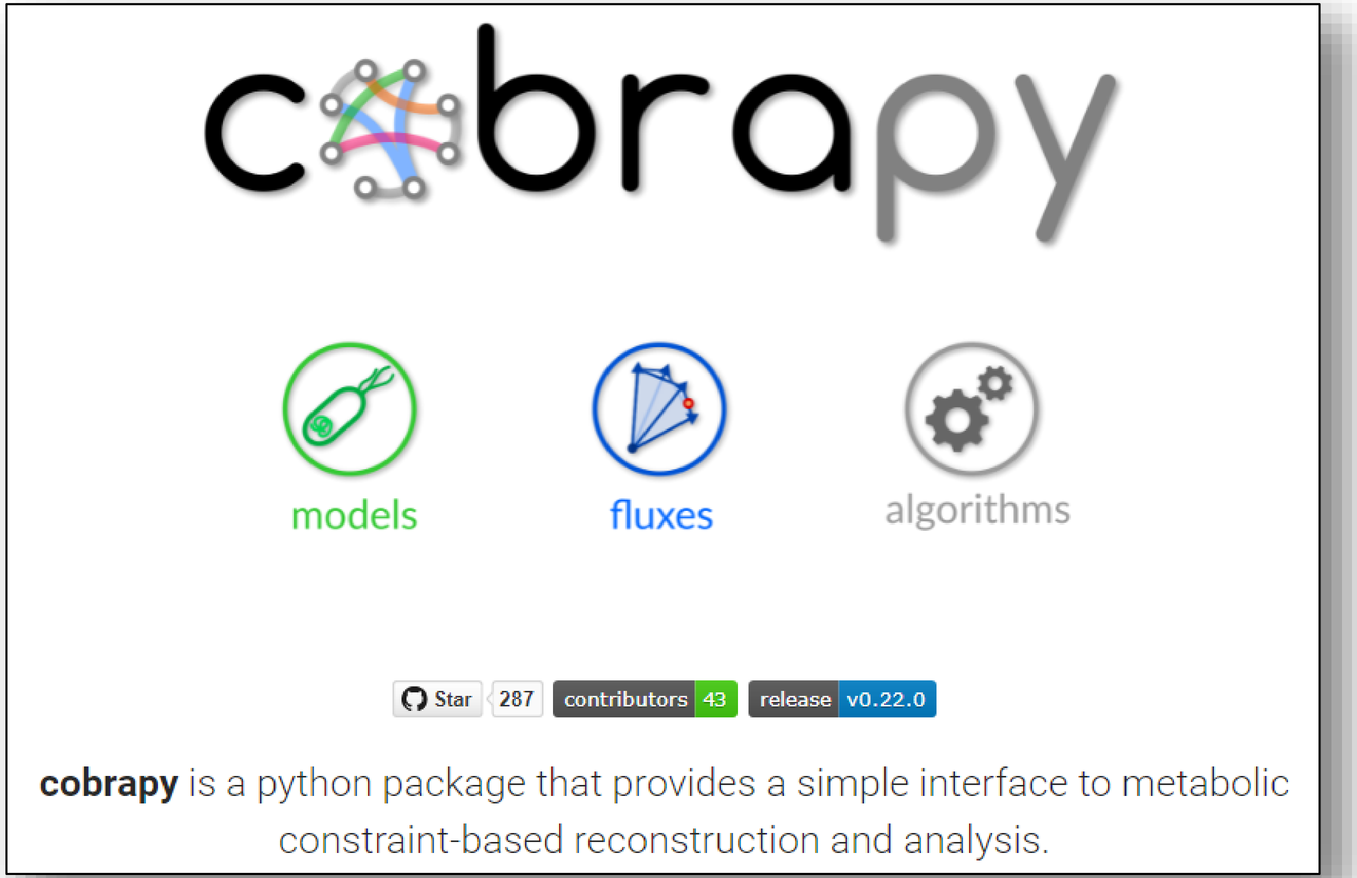
**Conclusion:** COBRApy is an object-oriented framework designed to meet the computational challenges associated with the next generation of stoichiometric constraint-based models and high-density omics data sets.

**Availability:** http://opencobra.sourceforge.net/

**Keywords:** Genome-scale, Network reconstruction, Metabolism, Gene expression, Constraint-based modeling

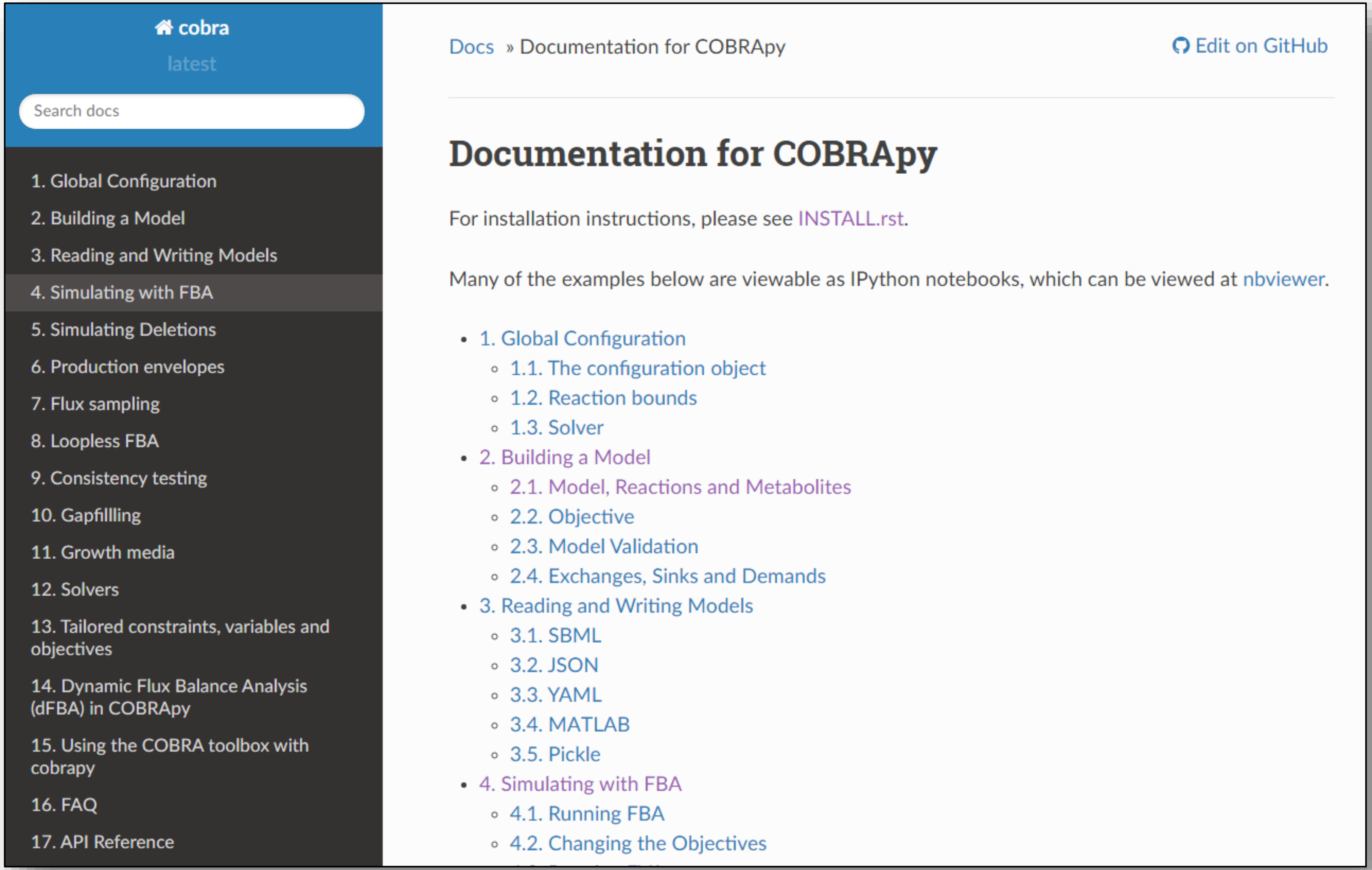
# COBRApy Toolbox Website

https://opencobra.github.io/cobrapy/



**cobrapy** is a python package that provides a simple interface to metabolic constraint-based reconstruction and analysis.

# COBRApy Documentation



1. Global Configuration
2. Building a Model
3. Reading and Writing Models
4. Simulating with FBA
5. Simulating Deletions
6. Production envelopes
7. Flux sampling
8. Loopless FBA
9. Consistency testing
10. Gapfillling
11. Growth media
12. Solvers
13. Tailored constraints, variables and objectives
14. Dynamic Flux Balance Analysis (dFBA) in COBRApy
15. Using the COBRA toolbox with cobrapy
16. FAQ
17. API Reference

## Documentation for COBRApy

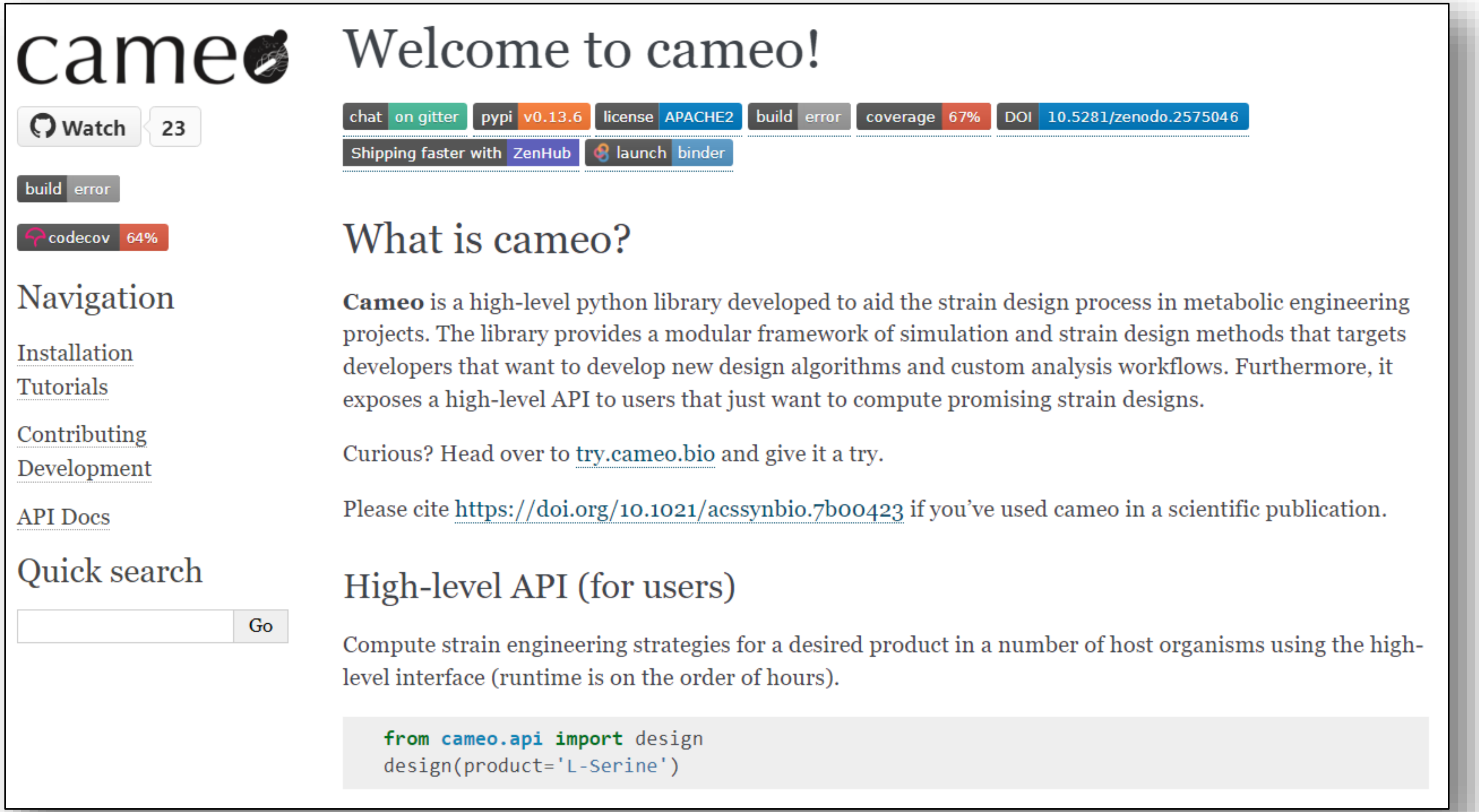
For installation instructions, please see INSTALL.rst.

Many of the examples below are viewable as IPython notebooks, which can be viewed at nbviewer.

- 1. Global Configuration
  - 1.1. The configuration object
  - 1.2. Reaction bounds
  - 1.3. Solver
- 2. Building a Model
  - 2.1. Model, Reactions and Metabolites
  - 2.2. Objective
  - 2.3. Model Validation
  - 2.4. Exchanges, Sinks and Demands
- 3. Reading and Writing Models
  - 3.1. SBML
  - 3.2. JSON
  - 3.3. YAML
  - 3.4. MATLAB
  - 3.5. Pickle
- 4. Simulating with FBA
  - 4.1. Running FBA
  - 4.2. Changing the Objectives

https://cobrapy.readthedocs.io/en/latest/index.html

# CAMEO



## Welcome to cameo!

### Navigation

Installation
Tutorials

Contributing
Development

API Docs

### Quick search

### What is cameo?

**Cameo** is a high-level python library developed to aid the strain design process in metabolic engineering projects. The library provides a modular framework of simulation and strain design methods that targets developers that want to develop new design algorithms and custom analysis workflows. Furthermore, it exposes a high-level API to users that just want to compute promising strain designs.

Curious? Head over to try.cameo.bio and give it a try.

Please cite https://doi.org/10.1021/acssynbio.7b00423 if you've used cameo in a scientific publication.

### High-level API (for users)

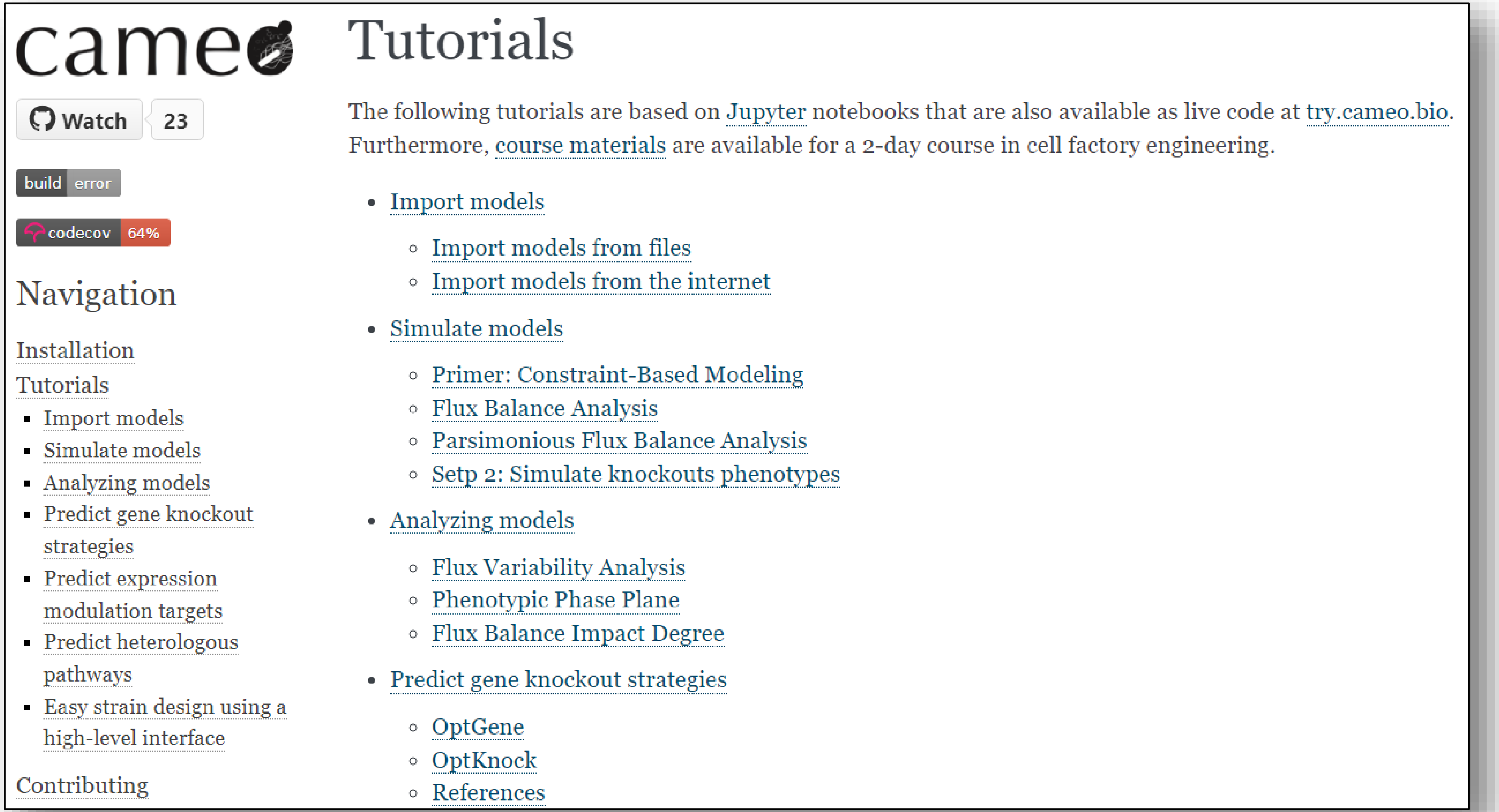
Compute strain engineering strategies for a desired product in a number of host organisms using the high-level interface (runtime is on the order of hours).

```
from cameo.api import design
design(product='L-Serine')
```

Cardoso, João GR, et al. "Cameo: a Python library for computer aided metabolic engineering and optimization of cell factories." *ACS synthetic biology* 7.4 (2018): 1163-1166.

# CAMEO Tutorials

https://cameo.bio/tutorials.html



## Navigation

Installation

Tutorials

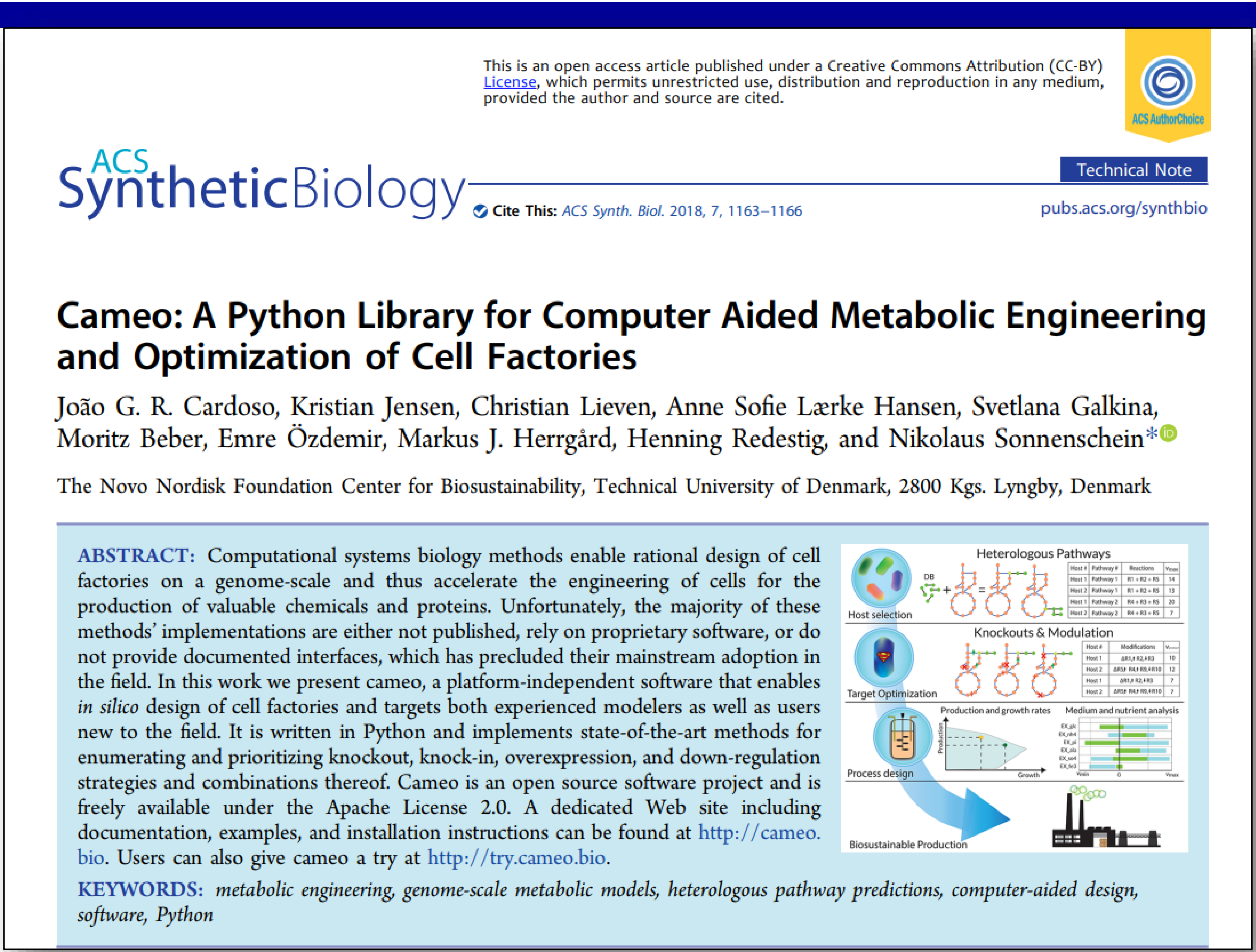
- Import models
- Simulate models
- Analyzing models
- Predict gene knockout strategies
- Predict expression modulation targets
- Predict heterologous pathways
- Easy strain design using a high-level interface

Contributing

# Tutorials

The following tutorials are based on Jupyter notebooks that are also available as live code at try.cameo.bio. Furthermore, course materials are available for a 2-day course in cell factory engineering.

- Import models
  - Import models from files
  - Import models from the internet
- Simulate models
  - Primer: Constraint-Based Modeling
  - Flux Balance Analysis
  - Parsimonious Flux Balance Analysis
  - Setp 2: Simulate knockouts phenotypes
- Analyzing models
  - Flux Variability Analysis
  - Phenotypic Phase Plane
  - Flux Balance Impact Degree
- Predict gene knockout strategies
  - OptGene
  - OptKnock
  - References



This is an open access article published under a Creative Commons Attribution (CC-BY) License, which permits unrestricted use, distribution and reproduction in any medium, provided the author and source are cited.

ACS AuthorChoice

ACS Synthetic Biology

Technical Note

Cite This: *ACS Synth. Biol.* 2018, 7, 1163−1166

pubs.acs.org/synthbio

# Cameo: A Python Library for Computer Aided Metabolic Engineering and Optimization of Cell Factories

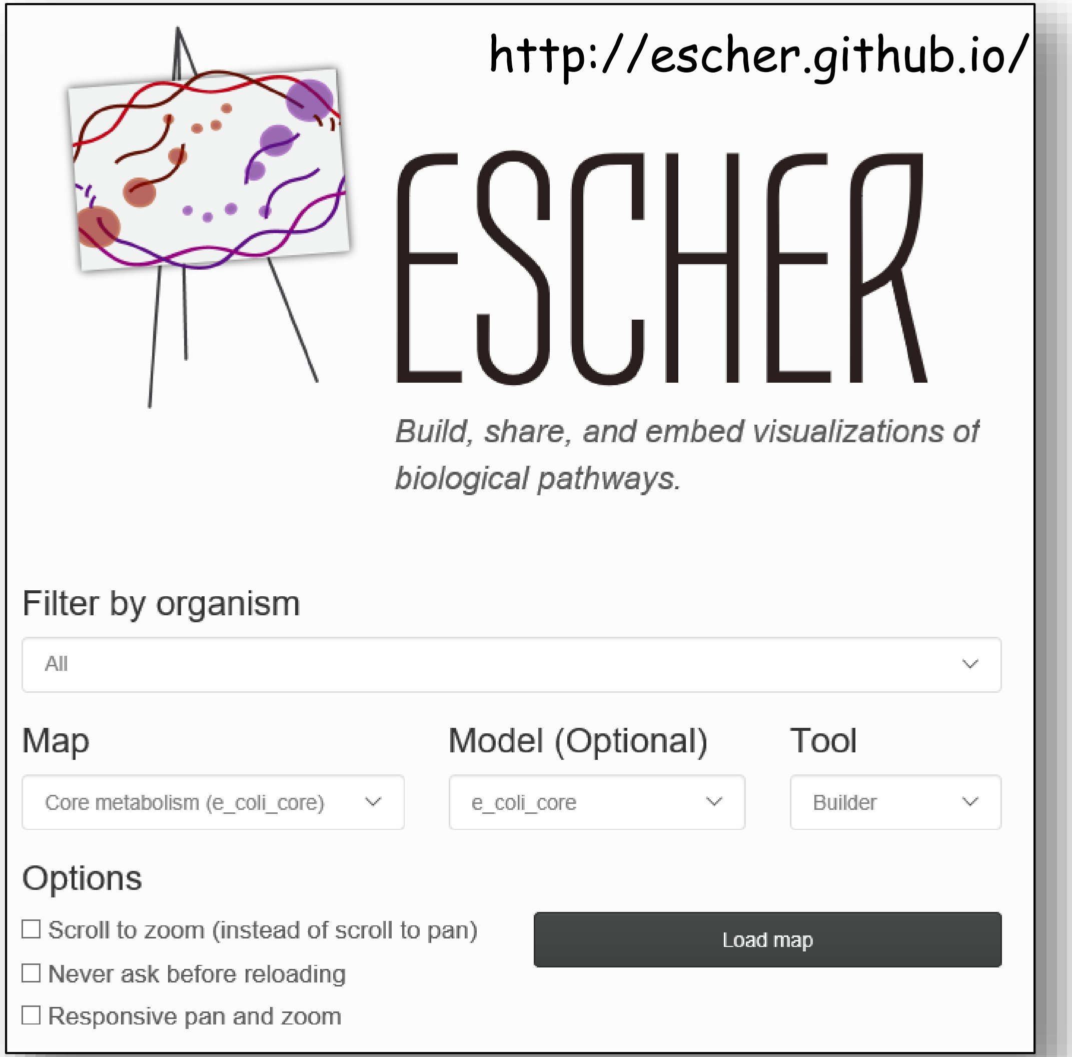
João G. R. Cardoso, Kristian Jensen, Christian Lieven, Anne Sofie Lærke Hansen, Svetlana Galkina, Moritz Beber, Emre Özdemir, Markus J. Herrgård, Henning Redestig, and Nikolaus Sonnenschein*

The Novo Nordisk Foundation Center for Biosustainability, Technical University of Denmark, 2800 Kgs. Lyngby, Denmark

**ABSTRACT:** Computational systems biology methods enable rational design of cell factories on a genome-scale and thus accelerate the engineering of cells for the production of valuable chemicals and proteins. Unfortunately, the majority of these methods' implementations are either not published, rely on proprietary software, or do not provide documented interfaces, which has precluded their mainstream adoption in the field. In this work we present cameo, a platform-independent software that enables *in silico* design of cell factories and targets both experienced modelers as well as users new to the field. It is written in Python and implements state-of-the-art methods for enumerating and prioritizing knockout, knock-in, overexpression, and down-regulation strategies and combinations thereof. Cameo is an open source software project and is freely available under the Apache License 2.0. A dedicated Web site including documentation, examples, and installation instructions can be found at http://cameo.bio. Users can also give cameo a try at http://try.cameo.bio.

**KEYWORDS:** *metabolic engineering, genome-scale metabolic models, heterologous pathway predictions, computer-aided design, software, Python*

# Escher Visualization



http://escher.github.io/

ESCHER

*Build, share, and embed visualizations of biological pathways.*

Filter by organism

All

Map: Core metabolism (e_coli_core)

Model (Optional): e_coli_core

Tool: Builder

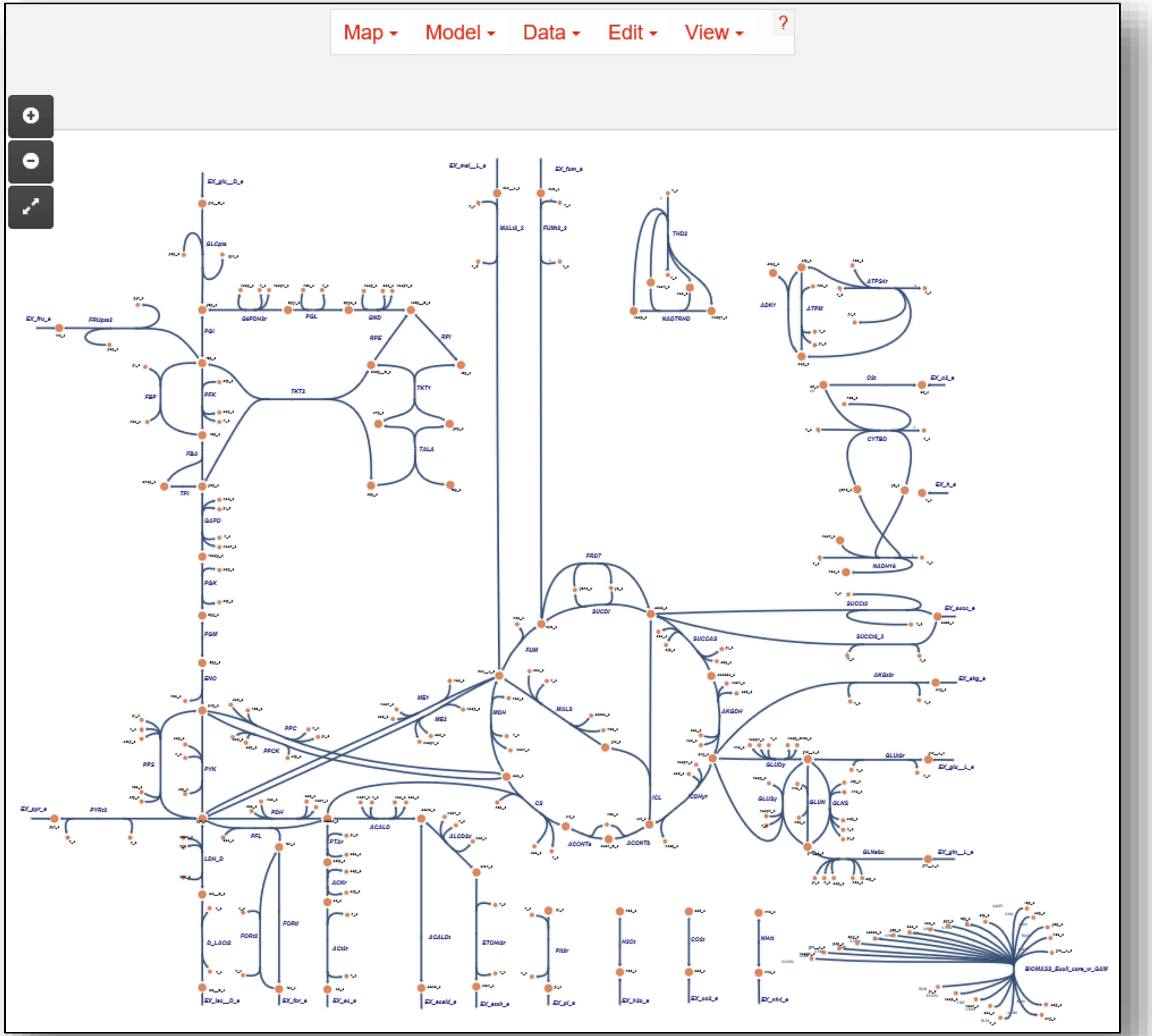
Options

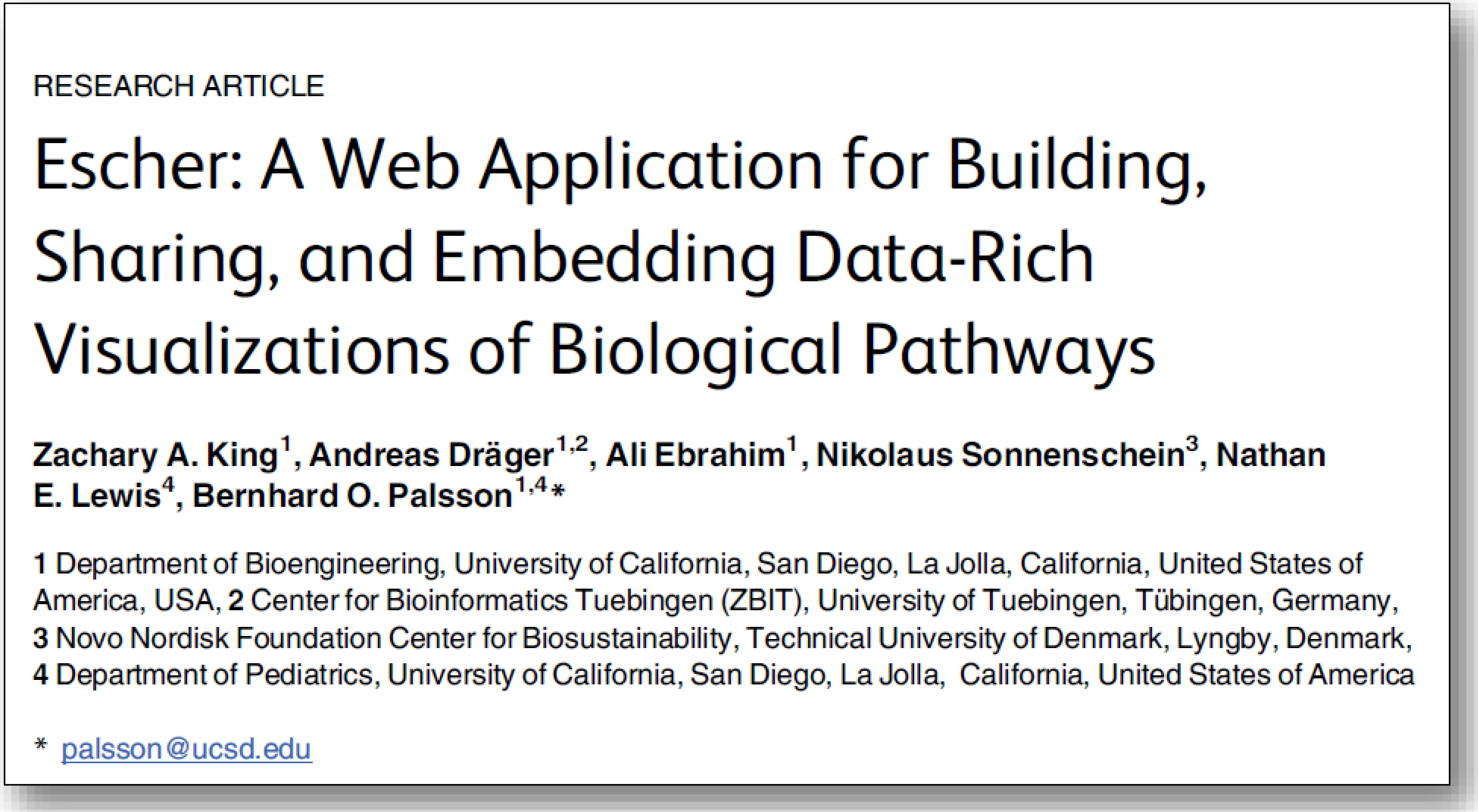
□ Scroll to zoom (instead of scroll to pan)

□ Never ask before reloading

□ Responsive pan and zoom

Load map





RESEARCH ARTICLE

# Escher: A Web Application for Building, Sharing, and Embedding Data-Rich Visualizations of Biological Pathways

**Zachary A. King[1], Andreas Dräger[1,2], Ali Ebrahim[1], Nikolaus Sonnenschein[3], Nathan E. Lewis[4], Bernhard O. Palsson[1,4]***

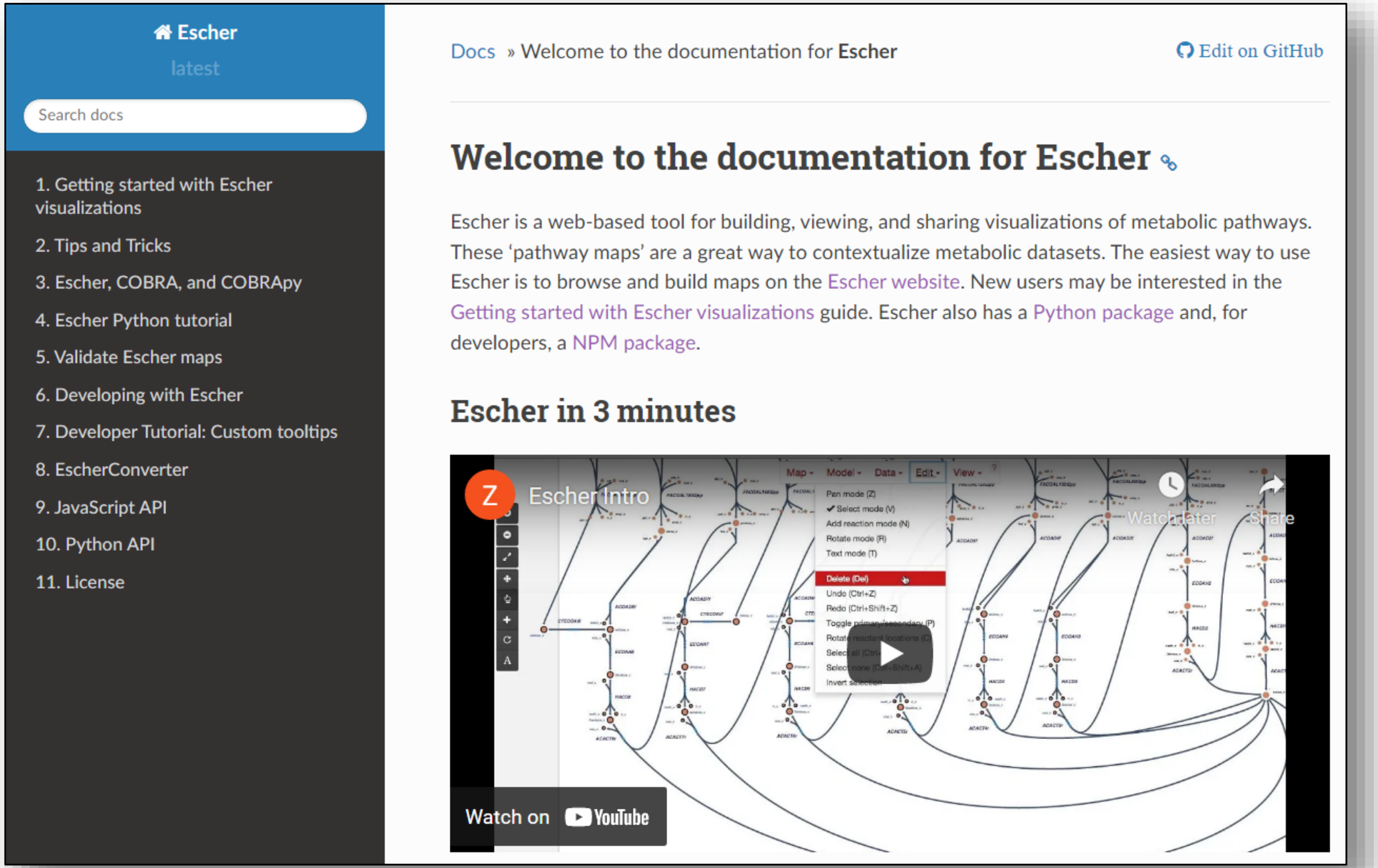
**1** Department of Bioengineering, University of California, San Diego, La Jolla, California, United States of America, USA, **2** Center for Bioinformatics Tuebingen (ZBIT), University of Tuebingen, Tübingen, Germany, **3** Novo Nordisk Foundation Center for Biosustainability, Technical University of Denmark, Lyngby, Denmark, **4** Department of Pediatrics, University of California, San Diego, La Jolla, California, United States of America

* palsson@ucsd.edu

King, Zachary A., et al. "Escher: a web application for building, sharing, and embedding data-rich visualizations of biological pathways." PLoS computational biology 11.8 (2015): e1004321.

# Escher Documentation

https://escher.readthedocs.io/en/latest/



Escher
latest

Search docs

1. Getting started with Escher visualizations
2. Tips and Tricks
3. Escher, COBRA, and COBRApy
4. Escher Python tutorial
5. Validate Escher maps
6. Developing with Escher
7. Developer Tutorial: Custom tooltips
8. EscherConverter
9. JavaScript API
10. Python API
11. License

Docs » Welcome to the documentation for Escher    Edit on GitHub

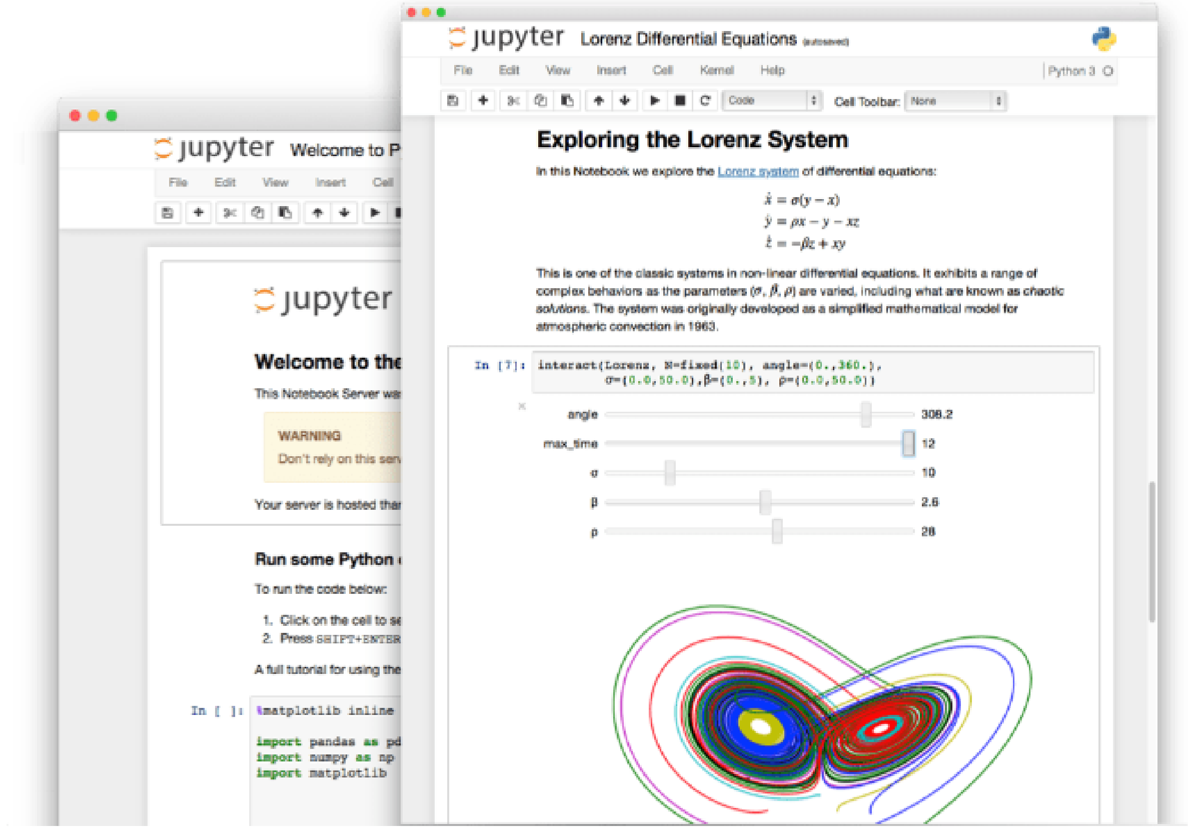
## Welcome to the documentation for Escher

Escher is a web-based tool for building, viewing, and sharing visualizations of metabolic pathways. These 'pathway maps' are a great way to contextualize metabolic datasets. The easiest way to use Escher is to browse and build maps on the Escher website. New users may be interested in the Getting started with Escher visualizations guide. Escher also has a Python package and, for developers, a NPM package.

## Escher in 3 minutes

# Jupyter Notebooks

https://jupyter.org/


jupyter Welcome to P
File Edit View Insert Cell
jupyter
jupyter Lorenz Differential Equations (autosaved)
File Edit View Insert Cell Kernel Help
Python 3
Code
Cell Toolbar: None

Welcome to the

This Notebook Server wa

WARNING

Don't rely on this ser

Your server is hosted tha

Run some Python

To run the code below:

1. Click on the cell to se
2. Press SHIFT+ENTER

A full tutorial for using the

```
In [ ]: %matplotlib inline

        import pandas as pd
        import numpy as np
        import matplotlib
```

## Exploring the Lorenz System

In this Notebook we explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters ($\sigma$, $\beta$, $\rho$) are varied, including what are known as *chaotic solutions*. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

```
In [7]: interact(Lorenz, N=fixed(10), angle=(0.,360.),
                 σ=(0.0,50.0),β=(0.,5), ρ=(0.0,50.0))
```

angle 308.2

max_time 12

σ 10

β 2.6

ρ 28

## Jupyter Notebook: The Classic Notebook Interface

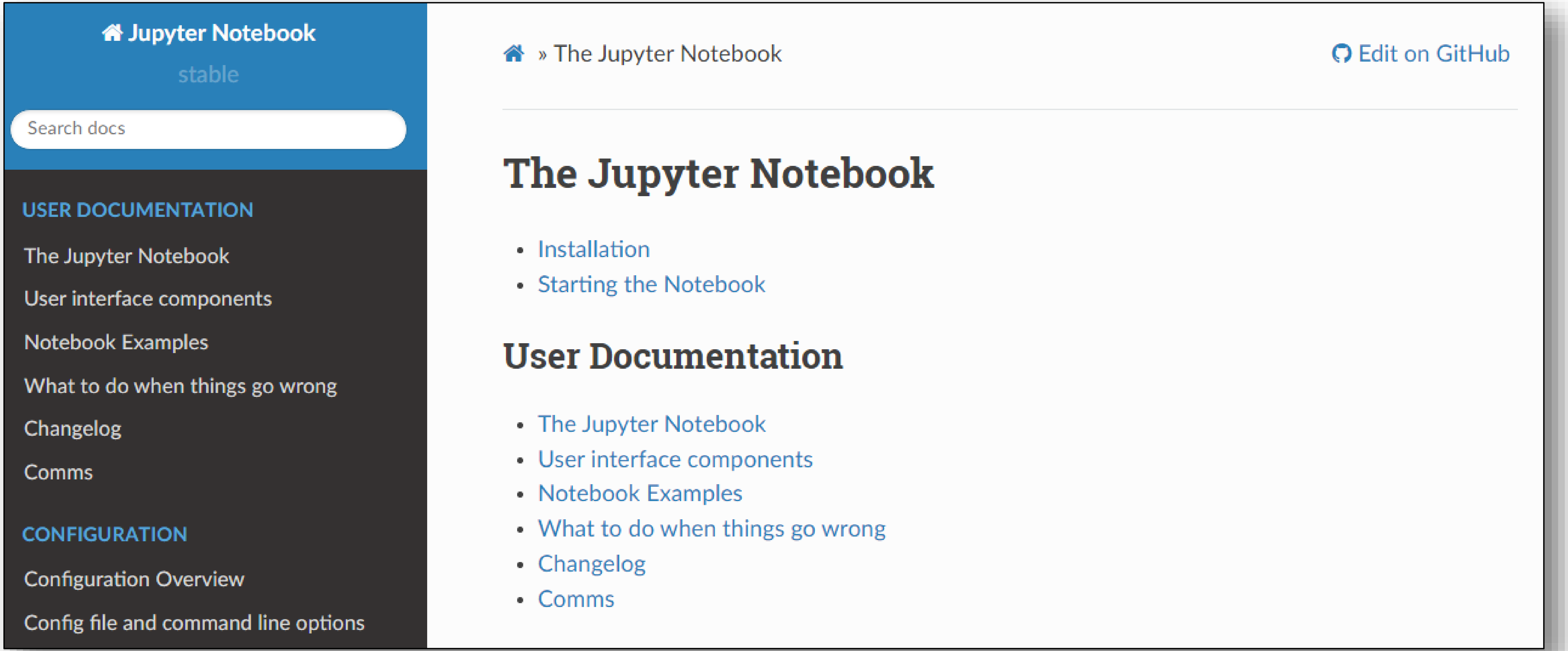
The Jupyter Notebook is a web application for creating and sharing documents that contain code, visualizations, and text. It can be used for data science, statistical modeling, machine learning, and much more.

Try it in your browser

Install the Notebook

# The Jupyter Notebook Documentation

https://jupyter-notebook.readthedocs.io/en/stable/



USER DOCUMENTATION

- The Jupyter Notebook
- User interface components
- Notebook Examples
- What to do when things go wrong
- Changelog
- Comms

CONFIGURATION

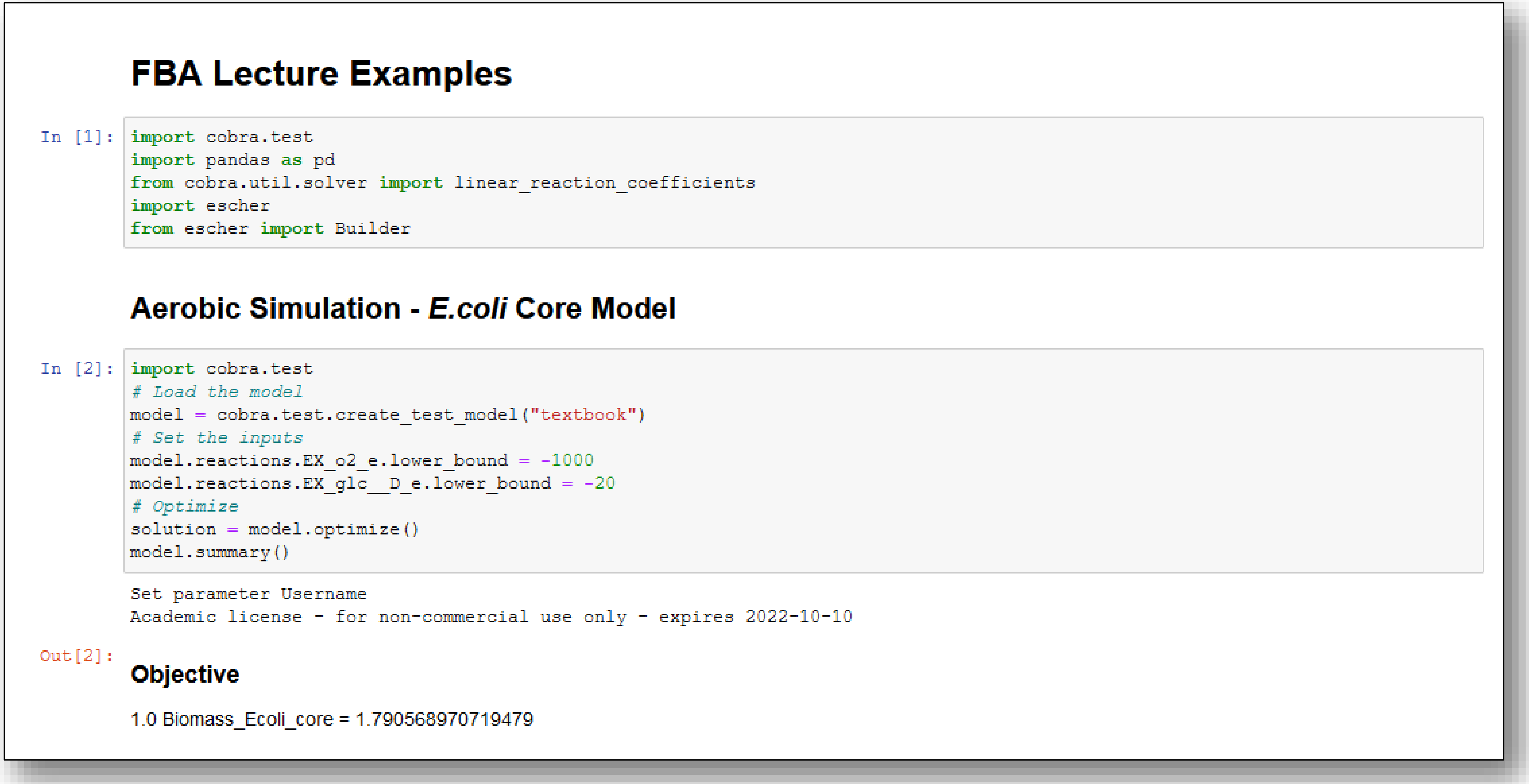
- Configuration Overview
- Config file and command line options

## The Jupyter Notebook

- Installation
- Starting the Notebook

## User Documentation

- The Jupyter Notebook
- User interface components
- Notebook Examples
- What to do when things go wrong
- Changelog
- Comms



# FBA Lecture Examples

```
In [1]: import cobra.test
        import pandas as pd
        from cobra.util.solver import linear_reaction_coefficients
        import escher
        from escher import Builder
```

## Aerobic Simulation - *E.coli* Core Model

```
In [2]: import cobra.test
        # Load the model
        model = cobra.test.create_test_model("textbook")
        # Set the inputs
        model.reactions.EX_o2_e.lower_bound = -1000
        model.reactions.EX_glc__D_e.lower_bound = -20
        # Optimize
        solution = model.optimize()
        model.summary()
```

```
Set parameter Username
Academic license - for non-commercial use only - expires 2022-10-10
```

Out[2]:

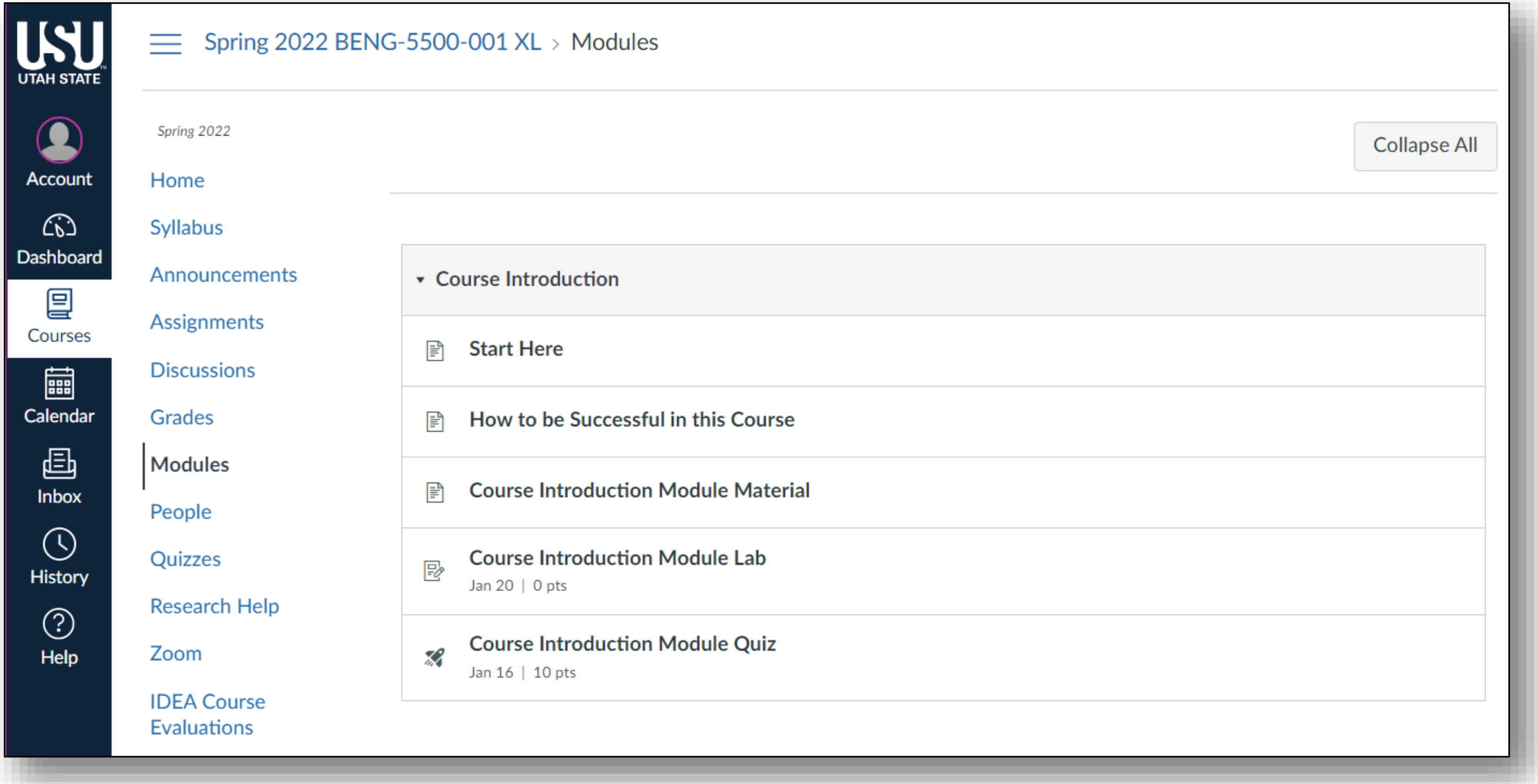
### Objective

1.0 Biomass_Ecoli_core = 1.790568970719479

# Course Introduction

- Content Overview
- ➡ Course Website
- Course Learning Process
- Course Grading & Expectations

# Canvas Website



Spring 2022 BENG-5500-001 XL › Modules

Spring 2022

Home
Syllabus
Announcements
Assignments
Discussions
Grades
Modules
People
Quizzes
Research Help
Zoom
IDEA Course Evaluations

Collapse All

▾ Course Introduction

- Start Here
- How to be Successful in this Course
- Course Introduction Module Material
- Course Introduction Module Lab
  Jan 20 | 0 pts
- Course Introduction Module Quiz
  Jan 16 | 10 pts

# Course Introduction

- Content Overview
- Course Website
- ➡ Course Learning Process
- Course Grading & Expectations

# Course Learning Objectives

Each student should be able to:

- Explain flux balance analysis
- Explain the basic *E.coli* core metabolic model
- Demonstrate the ability to effectively use the COBRApy Toolbox
- Explain and demonstrate flux variability analysis
- Explain and demonstrate randomized sampling
- Explain and demonstrate model interrogation
- Explain and demonstrate the model creation and enhancement
- Explain and demonstrate production envelopes
- Explain and demonstrate gene/reaction knockout strategies
- Explain and demonstrate gene/reaction modulation
- Explain and demonstrate strain design

# Course Learning Process

**Optional Textbook**

Systems Biology: Constraint-based Reconstruction and Analysis, Bernhard O. Palsson, Cambridge University Press, 2015

**Weekly Schedule**

- Tuesdays – Lecture and discussion.
- Thursdays – Lab:
    - Complete a lab that is associated with the content discussed on Tuesday.
    - Labs will be due at the beginning of the next week's lab
- Friday – Quiz
    - On-line through the Canvas website
    - Only cover the material associated with Tuesday's lecture and Thursday's lab
    - Will be a subset of the reflective questions associated with each lecture
    - Maximum time allowed per quiz is 15 minutes

# Course Introduction

- Content Overview
- Course Website
- Course Learning Process
- ➡ Course Grading & Expectations

# Grading

- Quizzes (you can drop one) - 40%
- Labs (you can drop one) – 50%
- Student participation – 10%

# Teacher Expectations

- Estimated homework for a B student
  - ✓ *Approximately 6-9 hours per week*
- All assignments and materials will be provided through the course website.
- Computer compatibility is your responsibility.
- Students are expected to attend every class.
- Students will check the course website at least two times per week.
- Students are expected to know *(or re-learn on their own)* material covered in prerequisite courses.

# Your Choice

- You will only get out of this course what you put into it!
- If you just try to get by, at the end of the course you will be totally confused and walk away with nothing.
- If you work hard and try to understand everything that is covered, at the end of the course you will walk away with a new understanding of the future of the life sciences.
- Since this is primarily a self-learning course, the battle is not between you and the professor, it will be an internal battle between your priorities.

# Course Introduction

- Content Overview
- Course Website
- Course Learning Process
- Course Grading & Expectations

# Reflective Questions

1. What is constraint-based modeling?
2. What is the biomass function?
3. How many labs can you drop?
4. What percentage of your grade will be based on quiz scores?
5. What is the maximum allowed time for each on-line quiz?
6. How many quizzes can you drop?
7. What percentage of your grade will be based on lab scores?
8. When are the labs due?
9. Who is responsible for your computer compatibility and management?
10. Are you responsible for material learned in the prerequisite courses?
11. What is the difference between a GEnome scale Network Reconstructions (GENREs) and a GEnome scale Models (GEMs)?
12. What is the purpose of Escher?
13. What is a Jupyter notebook?
14. How often should you review the course website?

# References

1. COBRA Overviews

   a. Lewis, N. E., H. Nagarajan, et al. (2012). "Constraining the metabolic genotype-phenotype relationship using a phylogeny of *in silico* methods." Nature reviews. Microbiology 10(4): 291-305.

   b. Terzer, M., N. D. Maynard, et al. (2009). "Genome-scale metabolic networks." Wiley Interdiscip Rev Syst Biol Med 1(3): 285-297.

   c. Feist, A. M. and B. O. Palsson (2008). "The growing scope of applications of genome-scale metabolic reconstructions using Escherichia coli." Nature biotechnology 26(6): 659-667.

2. Documentation for Course Tools

   a. Ebrahim, A., Lerman, J.A., Palsson, B.O. et al. COBRApy: COnstraints-Based Reconstruction and Analysis for Python. BMC Syst Biol 7, 74 (2013). https://doi.org/10.1186/1752-0509-7-74

   b. Cardoso, João GR, et al. "Cameo: a Python library for computer aided metabolic engineering and optimization of cell factories." ACS synthetic biology 7.4 (2018): 1163-1166.

   c. King, Zachary A., et al. "Escher: a web application for building, sharing, and embedding data-rich visualizations of biological pathways." PLoS computational biology 11.8 (2015): e1004321.