CNApy Users Guide

This manuscript (permalink) was automatically generated from cnapy-org/users-guide@ea90005 on July 8, 2021.

Authors

- **Sven Thiele**
  0000-0002-5812-6963 · sthiele
  Analysis and Redesign of Biological Networks, Max Planck Institute for Dynamics of Complex Technical Systems Magdeburg

- **Axel von Kamp**
  · axelvonkamp
  Analysis and Redesign of Biological Networks, Max Planck Institute for Dynamics of Complex Technical Systems Magdeburg

Abstract

Introduction

This is the user manual of CNApy. CNApy is a graphical interface for the modelling and analysis of metabolic networks on the basis of constraint-based (stoichiometric) modeling approaches. It allows

- the import/export of SBML models
- creating COBRApy/CNA models
- linking a graphical representation to the model data
- convenient exploration and editing of the model
- model analysis with standard and advanced constraint-based methods
- and saving everything as a CNApy *.cna project

The methods provided for model analysis (some of them are part of CNApy but most are functions from CNA and COBRApy) include:

- Flux balance analysis
- Parsimonious flux balance analysis
- Flux variability analysis
- Minimal cut sets
- Elementary modes
- Phase plane analysis
- Yield optimization

CNApy is available at https://github.com/cnapy-org/CNApy under the Apache-2.0 License.

We appreciate any comments or suggestions for improvements and we are greatly interested in your feedback which you can give at our User Forum.

For feature requests and bug reports please use our Issue tracker

Thank you for using CNApy!

Installation

The easiest way to install CNApy is using conda. First install conda and then:

- Create a conda environment with all dependencies

  ```
  conda create -n cnapy-1.0.0 -c conda-forge -c cnapy cnapy=1.0.0
  ```

- Activate the cnapy conda environment

  ```
  conda activate cnapy-1.0.0
  ```

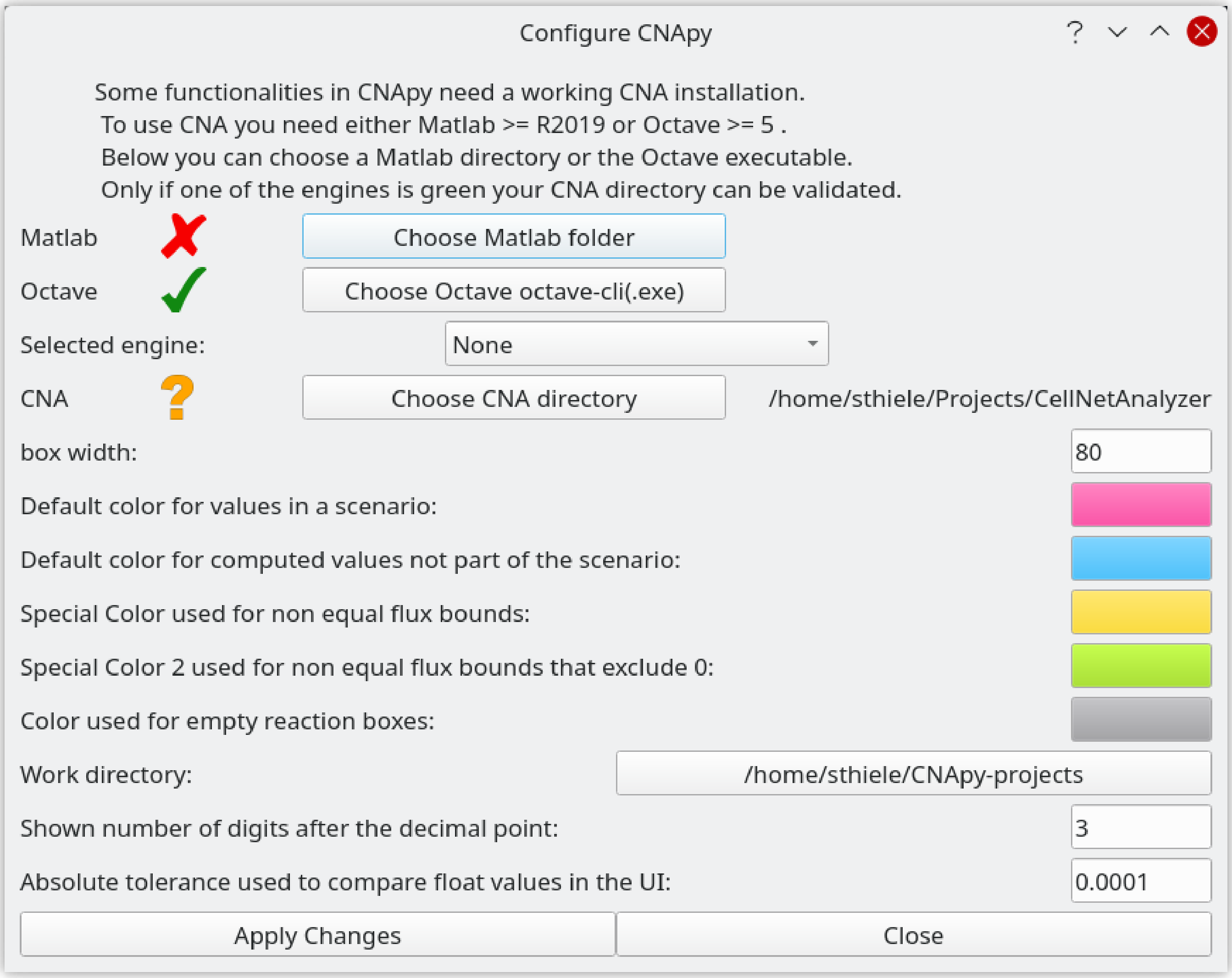
- Run CNApy

```
cnapy
```

On windows you may also use the graphical installer.

Configuration

When we start CNApy for the first time we are greeted with a dialog like this.



Some functionalities in CNApy need a working CNA installation.
To use CNA you need either Matlab >= R2019 or Octave >= 5 .
Below you can choose a Matlab directory or the Octave executable.
Only if one of the engines is green your CNA directory can be validated.

Matlab	✗	Choose Matlab folder	
Octave	✓	Choose Octave octave-cli(.exe)	
Selected engine:		None	
CNA	?	Choose CNA directory	/home/sthiele/Projects/CellNetAnalyzer
box width:			80
Default color for values in a scenario:			🟪
Default color for computed values not part of the scenario:			🟦
Special Color used for non equal flux bounds:			🟨
Special Color 2 used for non equal flux bounds that exclude 0:			🟩
Color used for empty reaction boxes:			⬜
Work directory:			/home/sthiele/CNApy-projects
Shown number of digits after the decimal point:			3
Absolute tolerance used to compare float values in the UI:			0.0001

Figure 1: CNApy Configuration dialog.

With this dialog you can configure which Matlab/Octave installation CNApy should use.

The minimal cut set computation in CNApy uses functions that are provided by CellNetAnalyzer. To be able to use these functions you have to provide CNApy with either a path to a Matlab installation >R2019 or the path to an Octave executable >4, and of course with a path to a recent CellNetAnalyzer installation. You can do this by clicking these buttons (1). If you change the settings CNApy performs some basic tests to make sure everything is working. If all checks are successful green check marks are shown, if a check fails a red cross is shown.

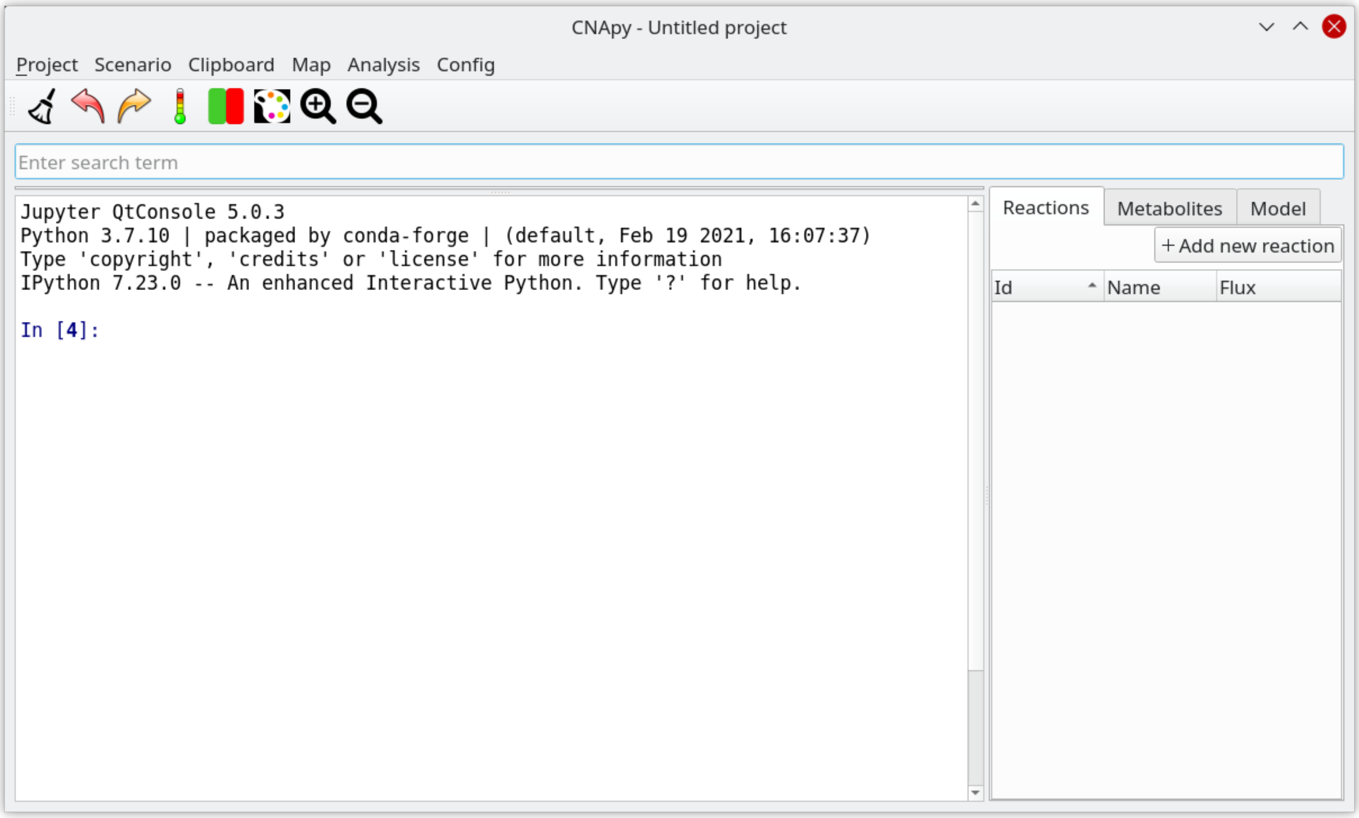
If no check can be performed, a yellow question mark is shown. For example, the check of the CNA directory needs a working Matlab or Octave installation.

You can also configure which colors CNApy should use to highlight reactions on the map or in the reaction list.

User interface overview

This section gives an introduction to the CNApy UI and its components and functions.

When we have finished our first configuration we are greeted with a window like this.



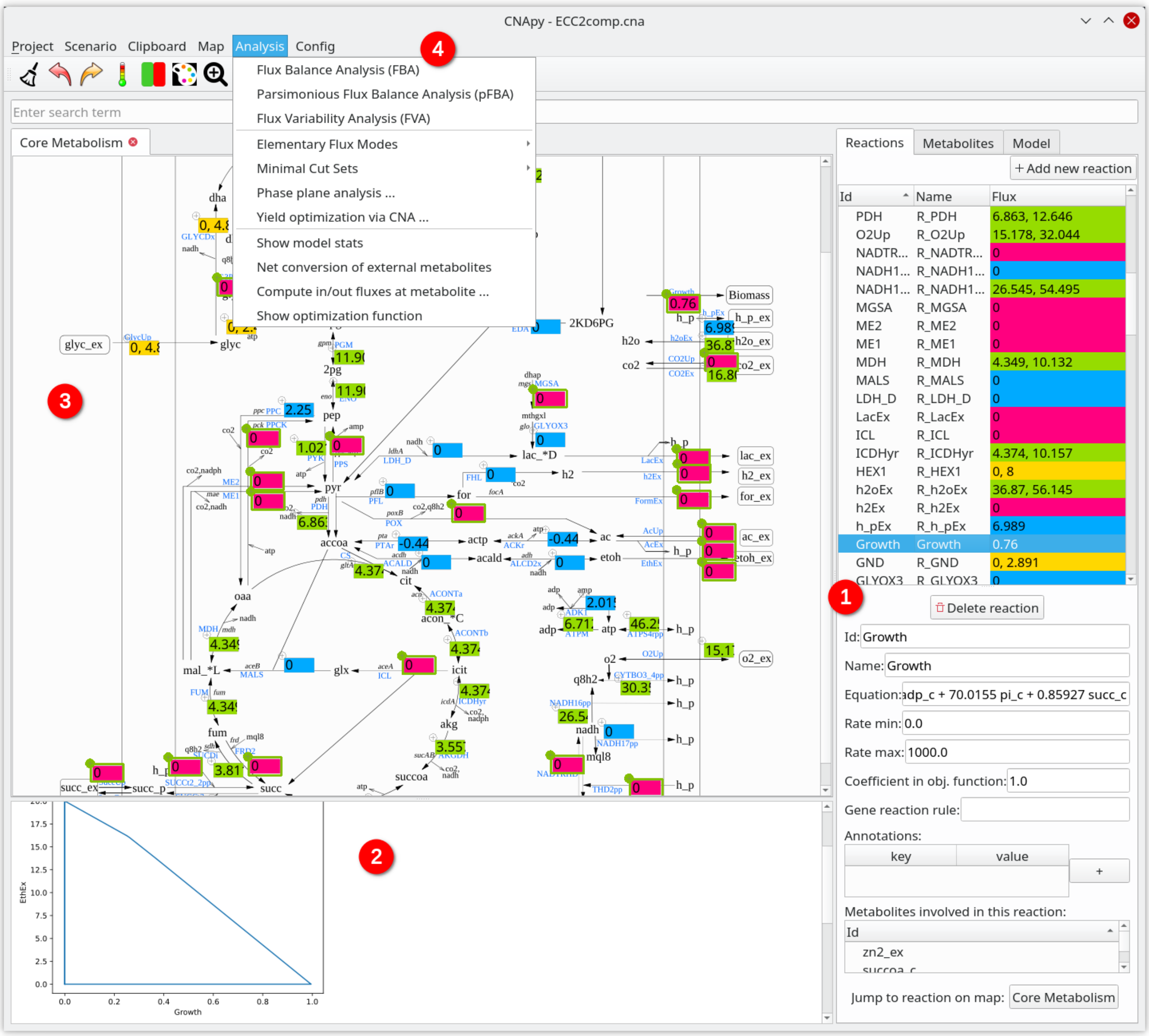
```
Jupyter QtConsole 5.0.3
Python 3.7.10 | packaged by conda-forge | (default, Feb 19 2021, 16:07:37)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.23.0 -- An enhanced Interactive Python. Type '?' for help.

In [4]:
```

Figure 2: CNApy with an empty project.

This is an empty project on the right hand side we see the empty lists of reactions and metabolites. On the left we see the embedded Jupyter Console where one can interact programmatically with the model and UI. We can create a new project by importing SBML models, editing the reactions and adding graphical maps. We can also load one of the projects included in our projects repository.

Let's go to Project in our menubar and open the `ECC2comp.cna` project.



Id	Name	Flux
PDH	R_PDH	6.863, 12.646
O2Up	R_O2Up	15.178, 32.044
NADTR...	R_NADTR...	0
NADH1...	R_NADH1...	0
NADH1...	R_NADH1...	26.545, 54.495
MGSA	R_MGSA	0
ME2	R_ME2	0
ME1	R_ME1	0
MDH	R_MDH	4.349, 10.132
MALS	R_MALS	0
LDH_D	R_LDH_D	0
LacEx	R_LacEx	0
ICL	R_ICL	0
ICDHyr	R_ICDHyr	4.374, 10.157
HEX1	R_HEX1	0, 8
h2oEx	R_h2oEx	36.87, 56.145
h2Ex	R_h2Ex	0
h_pEx	R_h_pEx	6.989
Growth	Growth	0.76
GND	R_GND	0, 2.891
GLYOX3	R_GLYOX3	0

Id: Growth
Name: Growth
Equation: adp_c + 70.0155 pi_c + 0.85927 succ_c
Rate min: 0.0
Rate max: 1000.0
Coefficient in obj. function: 1.0
Gene reaction rule:

Id
zn2_ex
succoa_c

Figure 3: CNApy with the ECC2comp project.

In this picture we see CNApy with the open ECC2comp project. On the right hand side (1) we see the populated reactions and metabolites lists with the color coded current values and corresponding details for the selected reaction/metabolite. The reactions list contains buttons that lets us add/delete reactions to/from the model. The console (2) now shows the output of some computations, and above the console we have a map view (3) with a graphical representation of our network. On top (4) we have the menu bar which gives us access to the various functionalities of CNApy and a Toolbar for quick access to often used functions.

We can add new maps via the Map menu, and drag reactions from the reaction list onto the desired position on the map.

Create a new project

In this Section we explain how to create a new network project. We also have a video on that topic https://youtu.be/bsNXZBmtyWw.

CNApy starts with an empty Project. You can directly add new reactions via the reaction mask on the right. By pressing the Add new reaction button.

Then simply type the reaction formula into the Equation field. For example `A + 2 B -> C`.

The associated metabolites `A`, `B` and `C` are then automatically added to the model.

Changing metabolite identifiers in the metabolite form instantly rewrites the reaction equation of the associated reactions.

Creating big models by adding single reactions is very cumbersome. Therefore CNApy allows you to create a new project from an SBML file. You find the point New project from SBML in the Project menu.

After importing an SBML model the reaction list is populated with all the reactions from the model.

You can add a map to our project, by going to the map menu and clicking add new map.

From the map menu we can also change the background image of our map.

Any SVG image can be used as a background.

You can add multiple maps that highlight different aspects of your model.

Reaction boxes can be put on the map by dragging over them from the reactions list.

You can adjust the box size with the keyboard shortcuts `Ctrl` + `+` and `Ctrl` + `-`. You can move the reaction box to the desired position by dragging them on their handles. If you want to move all reaction boxes together, press `Ctrl` while dragging.

Positioning a big number of reaction boxes can be tiresome. CNApy allows you to load and save reaction box positions. This facilitates reuse among similar projects.

The size of the background image can be adjusted with the shortcuts `Ctrl` + `Shift` + `+` and `Ctrl` + `Shift` + `-`.

To save a project click Save Project as in the Project menu. It is important to add the dot CNA extension to the file name. This way your files can be found by CNApy.

Scenarios

In CNApy you can define a scenario under which metabolic analysis like the FBA will be performed. In essence a scenario is a set of flux constraints for a set of reactions. These constraints can fix the flux of a reaction or constraint lower and upper bounds for the flux.

The scenario can be edited via the map, by entering values into the corresponding reaction boxes. Accepted values are either a single float like `1.2` or a pair of floats like `(-10, 1.2)`. A single float fixes the flux of the reaction to this value. While a pair sets the lower flux bound of the reaction to the first value and the upper flux bound to the second value.

Reactions boxes with scenario constraints are marked by the scenario color and a green frame. The frame turns yellow, if the scenario constraint contradicts the reaction constraints of the model.

To remove the constraints on a reaction simply delete the scenario value in the reaction box.

You can save and load scenarios as *.scen files. One scenario can be set as the default scenario of your project. This can be done via* set current scenario as default scenario* in the Scenario menu. The project must then be saved, and the next time you open the project the scenario is already set.

CNApy implements an edit history for the scenario with the tool buttons you can undo and redo your changes to the scenario. You can also import all the values that are currently in the reaction boxes into the scenario, and you can change the model's reaction bounds to the current scenario values.

The rest of this document is a full list of formatting elements/features supported by Manubot. Compare the input (`.md` files in the `/content` directory) to the output you see below.

Basic formatting

Bold text

Semi-bold text

Centered text

Right-aligned text

Italic text

Combined *italics and **bold***

~~Strikethrough~~

1. Ordered list item
2. Ordered list item
   a. Sub-item
   b. Sub-item
      i. Sub-sub-item
3. Ordered list item
   a. Sub-item

- List item
- List item
- List item

subscript: H_2O is a liquid

superscript: 2^{10} is 1024.

unicode superscripts^{0123456789}

unicode subscripts$_{0123456789}$

A long paragraph of text. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Putting each sentence on its own line has numerous benefits with regard to editing and version control.

Line break without starting a new paragraph by putting
two spaces at end of line.

Document organization

Document section headings:

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

A heading centered on its own printed page

Horizontal rule:

`Heading 1`'s are recommended to be reserved for the title of the manuscript.

`Heading 2`'s are recommended for broad sections such as *Abstract*, *Methods*, *Conclusion*, etc.

`Heading 3`'s and `Heading 4`'s are recommended for sub-sections.

Links

Bare URL link: https://manubot.org

Long link with lots of words and stuff and junk and bleep and blah and stuff and other stuff and more stuff yeah

Link with text

Link with hover text

Link by reference

Citations

Citation by DOI [1].

Citation by PubMed Central ID [2].

Citation by PubMed ID [3].

Citation by Wikidata ID [4].

Citation by ISBN [5].

Citation by URL [6].

Citation by alias [7].

Multiple citations can be put inside the same set of brackets [1,5,7]. Manubot plugins provide easier, more convenient visualization of and navigation between citations [2,3,7,8].

Citation tags (i.e. aliases) can be defined in their own paragraphs using Markdown's reference link syntax:

Referencing figures, tables, equations

Figure 4

Figure 5

Figure 6

Figure 7

Table 1

Equation 1

Equation 2

Quotes and code

> Quoted text

> Quoted block of text
>
> Two roads diverged in a wood, and I—
> I took the one less traveled by,
> And that has made all the difference.

Code `in the middle` of normal text, aka `inline code`.

Code block with Python syntax highlighting:

```python
from manubot.cite.doi import expand_short_doi

def test_expand_short_doi():
    doi = expand_short_doi("10/c3bp")
    # a string too long to fit within page:
    assert doi == "10.25313/2524-2695-2018-3-vliyanie-enhansera-copia-i-
        insulyatora-gypsy-na-sintez-ernk-modifikatsii-hromatina-i-
        svyazyvanie-insulyatornyh-belkov-vtransfetsirovannyh-geneticheskih-
        konstruktsiyah"
```

Code block with no syntax highlighting:

```
Exporting HTML manuscript
Exporting DOCX manuscript
Exporting PDF manuscript
```

Figures

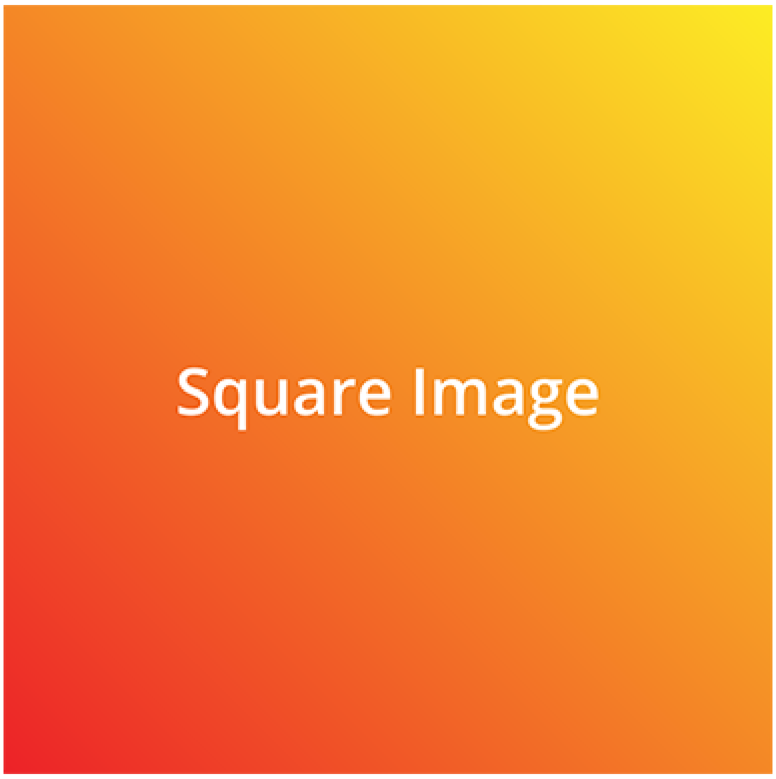


Figure 4: A square image at actual size and with a bottom caption. Loaded from the latest version of image on GitHub.

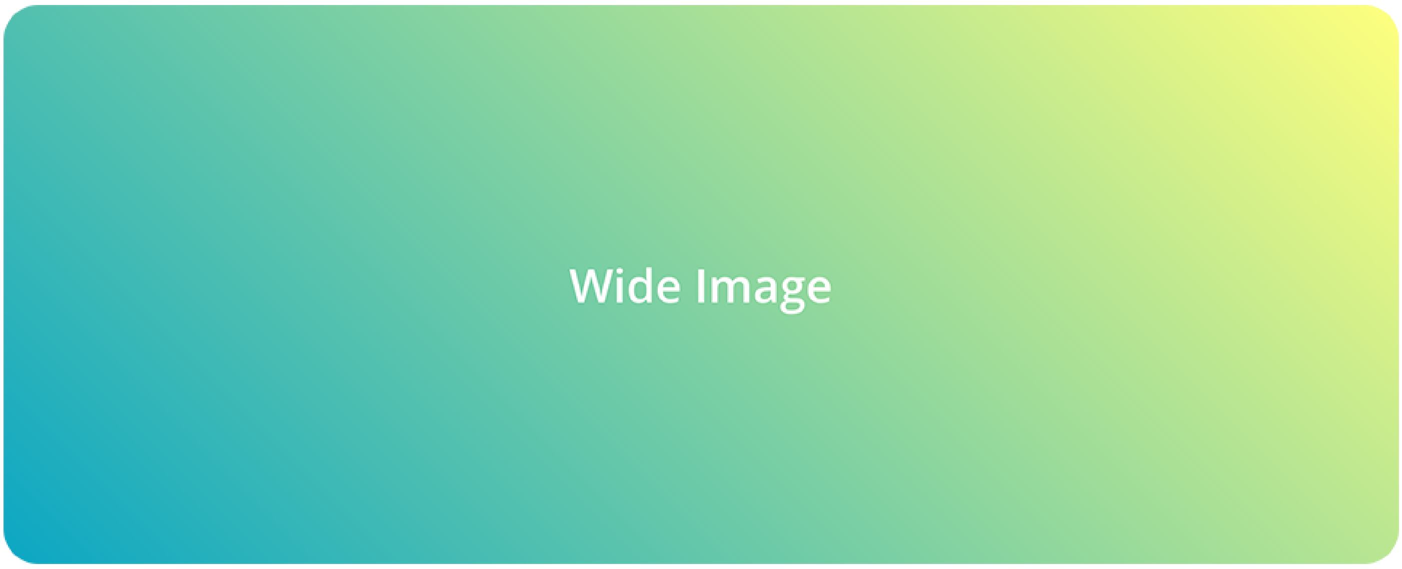


Figure 5: An image too wide to fit within page at full size. Loaded from a specific (hashed) version of the image on GitHub.

Figure 6: A tall image with a specified height. Loaded from a specific (hashed) version of the image on GitHub.

Figure 7: A vector `.svg` image loaded from GitHub. The parameter `sanitize=true` is necessary to properly load SVGs hosted via GitHub URLs. White background specified to serve as a backdrop for transparent sections of the image.

Tables

Table 1: A table with a top caption and specified relative column widths.

Bowling Scores	**Jane**	**John**	**Alice**	**Bob**
Game 1	150	187	210	105
Game 2	98	202	197	102
Game 3	123	180	238	134

Table 2: A table too wide to fit within page.

	Digits 1-33	**Digits 34-66**	**Digits 67-99**	**Ref.**
pi	3.14159265358979323 846264338327950	28841971693993751 0582097494459230	78164062862089986 2803482534211706	`piday.org`
e	2.71828182845904523 536028747135266	24977572470936999 5957496696762772	40766303535475945 7138217852516642	`nasa.gov`

Table 3: A table with merged cells using the `attributes` plugin.

	Colors	
Size	**Text Color**	**Background Color**
big	blue	orange
small	black	white

Equations

A LaTeX equation:

$$\int_0^\infty e^{-x^2} dx = \frac{\sqrt{\pi}}{2} \tag{1}$$

An equation too long to fit within page:

$$\begin{split} x = a + b + c + d + e + f + g + h + i + j + k + l + m + n + o + p + q + r + s + t \\ + u + v + w + x + y + z + 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 \end{split} \tag{2}$$

Special

⚠ WARNING *The following features are only supported and intended for `.html` and `.pdf` exports. Journals are not likely to support them, and they may not display correctly when converted to other formats such as `.docx`.*

LINK STYLED AS A BUTTON

Adding arbitrary HTML attributes to an element using Pandoc's attribute syntax:

> Manubot Manubot Manubot Manubot Manubot. Manubot Manubot Manubot Manubot. Manubot Manubot Manubot. Manubot Manubot. Manubot.

Adding arbitrary HTML attributes to an element with the Manubot `attributes` plugin (more flexible than Pandoc's method in terms of which elements you can add attributes to):

> Manubot Manubot Manubot Manubot Manubot. Manubot Manubot Manubot Manubot. Manubot Manubot Manubot. Manubot Manubot. Manubot.

Available background colors for text, images, code, banners, etc:

`white` `lightgrey` `grey` `darkgrey` `black` `lightred` `lightyellow` `lightgreen` `lightblue` `lightpurple` `red` `orange` `yellow` `green` `blue` `purple`

Using the Font Awesome icon set:

✓ ? ★ 🔔 ⊗ ⋯

Light Grey Banner
useful for *general information* - manubot.org

Blue Banner
useful for *important information* - manubot.org

Light Red Banner
useful for *warnings* - manubot.org

References

1. **Sci-Hub provides access to nearly all scholarly literature**
Daniel S Himmelstein, Ariel Rodriguez Romero, Jacob G Levernier, Thomas Anthony Munro, Stephen Reid McLaughlin, Bastian Greshake Tzovaras, Casey S Greene
eLife (2018-03-01) https://doi.org/ckcj
DOI: 10.7554/elife.32822 · PMID: 29424689 · PMCID: PMC5832410

2. **Reproducibility of computational workflows is automated using continuous analysis**
Brett K Beaulieu-Jones, Casey S Greene
Nature biotechnology (2017-04) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6103790/
DOI: 10.1038/nbt.3780 · PMID: 28288103 · PMCID: PMC6103790

3. **Bitcoin for the biological literature.**
Douglas Heaven
Nature (2019-02) https://www.ncbi.nlm.nih.gov/pubmed/30718888
DOI: 10.1038/d41586-019-00447-9 · PMID: 30718888

4. **Plan S: Accelerating the transition to full and immediate Open Access to scientific publications**
cOAlition S
(2018-09-04) https://www.wikidata.org/wiki/Q56458321

5. **Open access**
Peter Suber
MIT Press (2012)
ISBN: 9780262517638

6. **Open collaborative writing with Manubot**
Daniel S Himmelstein, Vincent Rubinetti, David R Slochower, Dongbo Hu, Venkat S Malladi, Casey S Greene, Anthony Gitter
Manubot (2020-05-25) https://greenelab.github.io/meta-review/

7. **Opportunities and obstacles for deep learning in biology and medicine**
Travers Ching, Daniel S Himmelstein, Brett K Beaulieu-Jones, Alexandr A Kalinin, Brian T Do, Gregory P Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M Hoffman, … Casey S Greene
Journal of The Royal Society Interface (2018-04-04) https://doi.org/gddkhn
DOI: 10.1098/rsif.2017.0387 · PMID: 29618526 · PMCID: PMC5938574

8. **Open collaborative writing with Manubot**
Daniel S Himmelstein, Vincent Rubinetti, David R Slochower, Dongbo Hu, Venkat S Malladi, Casey S Greene, Anthony Gitter
PLOS Computational Biology (2019-06-24) https://doi.org/c7np
DOI: 10.1371/journal.pcbi.1007128 · PMID: 31233491 · PMCID: PMC6611653