synergy-maps Documentation

Release 1.0.0

Richard P. I. Lewis

July 20, 2015
## 1 Introduction
1.1 Software .............................................. 1
1.2 Paper ..................................................... 1

## 2 Contents
2.1 Introduction ............................................. 3
2.2 Frontend ................................................. 4
2.3 Backend .................................................. 4
2.4 Documentation ......................................... 5
2.5 Git Structure ............................................ 6
2.6 synergy_maps package ................................. 6

## 3 Indices and tables ........................................ 7
Synergy maps are a novel visualization technique for exploring pairwise combination datasets, developed in the Centre for Molecular Informatics of the Chemistry Department of the University of Cambridge. This is the documentation for an interactive implementation, written by Richard Lewis, whilst a PhD student under the supervision of Andreas Bender.

1.1 Software

Briefly, the data processing is done on the Python backend, and the visualization is produced by the JavaScript frontend:

There is a demo of the interactive frontend for some example datasets hosted on GitHub.

1.2 Paper

The accompanying paper has been preliminarily been accepted into the Journal of Cheminformatics as part of the Mixtures and Combination Modelling thematic issue.
2.1 Introduction

Synergy maps are a novel visualization technique for exploring pairwise combination datasets, developed in the Centre for Molecular Informatics of the Chemistry Department of the University of Cambridge. This is the documentation for an interactive implementation, written by Richard Lewis, whilst a PhD student under the supervision of Andreas Bender.

2.1.1 Software

Briefly, the data processing is done on the Python backend, and the visualization is produced by the JavaScript frontend:

There is a demo of the interactive frontend for some example datasets hosted on GitHub.

2.1.2 Paper

The accompanying paper has been preliminarily been accepted into the Journal of Cheminformatics as part of the Mixtures and Combination Modelling thematic issue.
2.2 Frontend

The frontend is written in JavaScript, and runs in modern (HTML5 enabled) web browsers. It is an angularjs app, and was scaffolded using yeoman, using the angular generator. The visualization uses the D3.js library to render the images.

2.2.1 Dependencies

The project requires the node runtime to build the app, Grunt to run tasks, and the package managers npm and bower to install development and production dependencies respectively.

2.2.2 Building

To build and serve the project (having installed node and npm), change into the frontend directory and run:

```
$ npm install -g grunt bower  // install grunt and bower
$ npm install  // install node modules
$ bower install  // install bower modules
$ grunt serve  // build and run the server
```

Your default browser should open by default to `localhost:9000`, where the app is being served.

2.3 Backend

The backend is written in Python, and is where data is processed into the JSON representation. The API is detailed in `synergy_maps` package.

2.3.1 Installation

Change into the backend directory and run the `setup.py` script.

```
$ python setup.py install
```

2.3.2 Quickstart

A prebuilt script, `synergy_map.py` for analysing pairwise combination data in a standard format is provided in the `scripts` folder. This provides minimum functionality - the software is designed for extensibility, so consider writing your own script such as those in the example datasets included in `backend/synergy_maps/examples/`.

The script can be run using:

```
$ python synergy_map.py example_compounds.smiles example_combinations.csv > map.json
```

Data structure

Compounds

The example script takes compounds as a smiles file. The smiles should be provided as a first column, and the identifier as the second column. The individual activities are supplied as extra columns.
A sample of a (made up) compound dataset would be:

```bash
$ head example_compounds.smiles
SMILES name    pIC50
CC ethane 1.2
CCC propane 2.1
CN methylamine 4.3
... ... ...
```

Showing 3 compounds, with pIC50 as activity.

**Combinations**

The example script takes combinations as a csv file. The combination identifier should be given as the first column, and the component compounds ids should be provided as the second and third column. The example combinations should provide their synergy metrics as extra columns.

```bash
$ head example_combinations.csv
id RowId ColId pGamma ExcessOverBliss
0  ethane propane 0.001  1.0
1  ethane methylamine 0.2  4.0
2  propane methylamine -0.1  1.2
... ... ...
```

**Running the Script**

The script will use the default representations (Structural and Targets) and reduction methods (TSNE, PCA and MDS) to generate the synergy map data, and print out the JSON to the console. This can be redirected to a file:

```bash
$ python synergy_map.py example_compounds.smiles example_combinations.csv > synergy_map.json
```

**Viewing the Map**

The generated map can then be viewed by putting the json file into the data directory of the Frontend and building the application.

```bash
$ mkdir $SYNERGY_BASE/frontend/app/data/new_dataset/
$ mv synergy_map.json > $SYNERGY_BASE/frontend/app/data/new_dataset/data.json
```

You will also need to add it to the metadata.json file:

```bash
$ less $SYNERGY_BASE/frontend/app/data/metadata.json
{"datasets": ["DREAM-Lymphoma", "NCATS-Malaria", "new_dataset"]}
```

This will now be selectable from the table.

Please consider looking at synergy_maps/examples where there are two example scripts using the declarative API for generating the datasets.

### 2.4 Documentation

The documentation you are reading is included in the docs subdirectory of the project repository. It is build using [sphinx](https://www.sphinx-doc.org/en/master/) and hosted using [readthedocs](https://readthedocs.org/).
2.4.1 Dependencies

To get it running locally, install sphinx and the readthedocs theme:

```
$ pip install sphinx readthedocs
```

2.4.2 Building

Documentation can then be built using the Makefile (or the make.bat for Windows):

```
$ make html
```

2.4.3 Live editing

For quick editing, we use sphinx-autodoc.

```
$ pip install sphinx-autodoc
$ make livehtml
```

2.5 Git Structure

All code in the project is in one repository. In order to use GitHub Pages, `git subtree` was used to put the master `frontend/dist` directory as the root of the gh-pages branch, as suggested in this post.

Thus in order to update the gh-pages example, one must perform this subtree merge each time, to get the current master.

2.6 synergy_maps package

2.6.1 Module contents

2.6.2 Submodules

2.6.3 synergy_maps.reduction_methods module

2.6.4 synergy_maps.synergy_map module

2.6.5 synergy_maps.types module
Indices and tables

- genindex
- modindex
- search