

A short course about fitting models with the `scipy.optimize` module

Ariel Rokem¹

¹ The University of Washington eScience Institute

DOI: [10.21105/jose.00016](https://doi.org/10.21105/jose.00016)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Submitted: 26 April 2018

Published: 04 July 2018

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC-BY](#)).

Summary

Fitting models and testing the match of the models to the measured data is a fundamental activity in many fields of science. This short (approximately 3-hour) course (available at: <https://github.com/arokem/scipy-optimize>) aims to teach participants to use the Scipy library's `optimize` module to fit models to data (Jones et al. 2001). Using data from a psychology experiment (Rokem and Landau 2016) as an example, the course motivates the use of explicit mathematical models to explain and predict data and compares linear models and non-linear models. The core of the lesson focuses on fitting a curve with the `curve_fit` function. The course also introduces the idea of model comparison with cross-validation for evaluation and selection between non-nested non-linear models.

Statement of need

Model fitting is useful in many different fields of research, but optimization for model fitting is not a topic that is usually covered in introductory statistics or computing classes in many fields (e.g., psychology). This course fills an existing need for hands-on curriculum that goes beyond the topics taught in introductory computing workshops, such as Software Carpentry, providing material for follow-up workshops on advanced/intermediate topics. The target audience for this course is researchers or students with some programming knowledge (e.g., having participated in a Software Carpentry workshop beforehand).

Learning objectives

In addition to the general objectives of this lesson, specific learning objectives are defined for each part of the lesson:

Part 1:

- Learners can define what a model is.
- Learners can define model parameters and model fitting.
- Learners can restate the benefits of modeling
- Learners can explain the utility of modeling applied to their data.

Part 2:

- Learners can identify a linear model.
- Learners can use `numpy` to fit a linear model to data
- Learners can evaluate a model using model residuals.

Part 3:

- Learners can identify a nonlinear model, and discuss the differences between linear and nonlinear models
- Learners can use `scipy.optimize` to fit a nonlinear model to data.
- Learners can calculate and display model residuals for nonlinear models

Part 4:

- Learners can define and identify overfitting.
- Learners can implement split-half cross-validation to evaluate model error.

Description of the module

This course originated from a [blog post](#) that extended the final exercise for the Software Carpentry instructor training (2013). It was then also taught as a remote online workshop, as part of a [series of remote workshops organized by the UC Davis Data Intensive Biology Lab](#).

The data used in the course comes from an experiment in visual neuroscience (Rokem and Landau 2016), but follows a format that is similar to data formats in many fields of science: repeated observations of a binary response (dependent) variable, for which the probability of a particular response depends on a known (independent) input variable. At the outset of the course, learners follow along as we read the data from a comma-separated file, plot the data, and transform it into a plot of probability of response as a function of input. The core of the course is a series of lessons on modeling these data: initially using a linear model, and then using non-linear models. This allows learners to understand and describe the distinction between these. The course introduces a functional form for these data and then uses the `curve_fit` function from the `scipy.optimize` module to fit these functional forms, with a sum-of-squared-errors objective function. The course also briefly introduces model comparison using cross-validation and motivates this with an example of over-fitting.

Usage

To use these instructional materials, it is recommended that the instructor type out the code in an interactive environment, such as a [Jupyter](#) notebook, while learners follow along on their own machines. For this purpose, prerequisites and setup instructions are provided on the [first page](#) of the lesson.

Example of instruction

A video of an example of instruction using this lesson is available on [YouTube](#)

References

Jones, Eric, Travis Oliphant, Pearu Peterson, and others. 2001. “SciPy: Open Source Scientific Tools for Python.” <http://www.scipy.org/>.

Rokem, Ariel, and Ayelet Nina Landau. 2016. “The Interaction of Orientation-Specific Surround Suppression and Visual-Spatial Attention.” *bioRxiv*. <https://doi.org/10.1101/091553>.