The Journal of Open Source Software

piecewise-regression (aka segmented regression) in Python

Charlie Pilgrim[1, 2]

1 Centre for Doctoral Training in Mathematics for Real-World Systems, University of Warwick, Coventry, UK **2** The Alan Turing Institute, London, UK

DOI: 10.21105/joss.03859

Software

- Review
- Repository
- Archive

Editor: Gabriela Alessio Robles

Reviewers:

- @vyasr
- @htjb
- @Ebedthan

Submitted: 04 October 2021
Published: 02 December 2021

Summary

Piecewise regression (also known as segmented regression, broken-line regression, or breakpoint analysis) fits a linear regression model to data that includes one or more breakpoints where the gradient changes. The `piecewise-regression` Python package uses the approach described by Muggeo (Muggeo, 2003), where the breakpoint positions and the straight line models are simultaneously fit using an iterative method. This easy-to-use package includes an automatic comprehensive statistical analysis that gives confidence intervals for all model variables and hypothesis testing for the existence of breakpoints.

Statement of Need

A common problem in many fields is to fit a continuous straight line model to data that includes some change(s) in gradient known as breakpoint(s). Examples include investigating medical interventions (Wagner et al., 2002), ecological thresholds (Toms & Lesperance, 2003), and geological phase transitions (Ryan et al., 2002). Fitting such models involves the global problem of finding estimates for the breakpoint positions and the local problem of fitting line segments given breakpoints. Possible approaches involve using linear regression to fit line segments together with a global optimisation algorithm to find breakpoints—for example, an evolutionary algorithm as in the `pwlf` python package (Jekel & Venter, 2019). Or one could take a non-linear least-squares approach using `scipy` (Virtanen et al., 2020) or the `lmfit` python package (Newville et al., 2016). Muggeo (Muggeo, 2003) derived an alternative method whereby the breakpoint positions and the line segment models are fitted simultaneously using an iterative method, which is computationally efficient and allows for robust statistical analysis. Many R packages implement this method, including the `segmented` R package written by Muggeo himself (Muggeo & others, 2008). However, before the `piecewise-regression` package, there were not comparable resources in Python.

Example

An example plot is shown in Figure 1. Data was generated with 3 breakpoints and some noise, and a model was then fit to that data. The plot shows the maximum likelihood estimators for the straight line segments and breakpoint positions. The package automatically carries out a Davies hypothesis test (Davies, 1987) for the existence of at least 1 breakpoint, in this example finding strong evidence for breakpoints with $p < 0.001$.

The Journal of Open Source Software

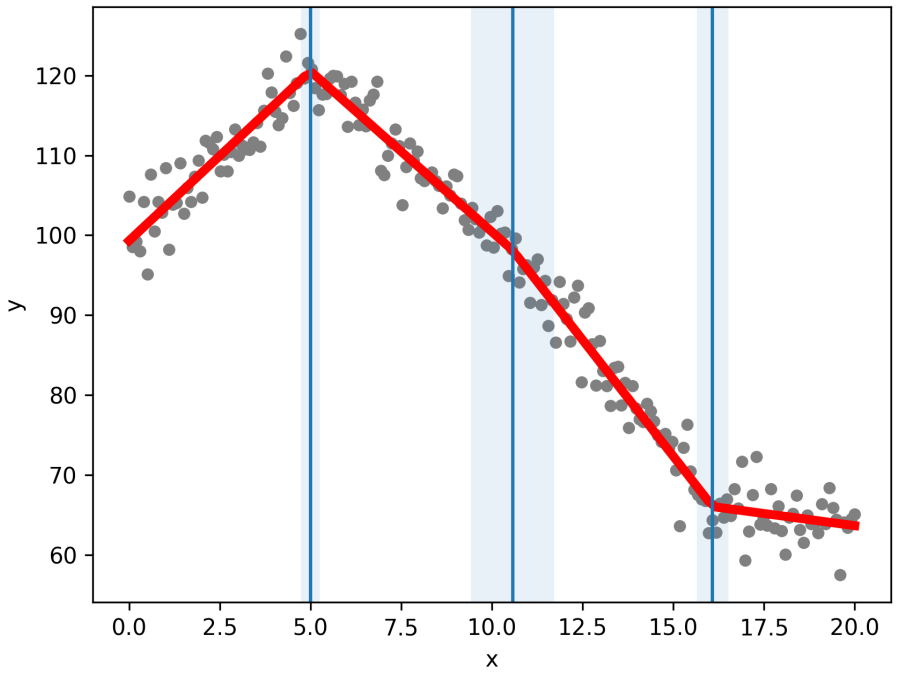

Figure 1: An example model fit (red line) to data (grey markers). The estimated breakpoint positions (blue lines) and confidence intervals (shaded blue regions) are shown. The data was generated using a piecewise linear model with a constant level of Gaussian noise. For example, this could represent observations with a sampling error of some physical process that undergoes phase transitions.

How It Works

We follow here the derivation by Muggeo (Muggeo, 2003). The general form of the model with one breakpoint is

$$y = \alpha x + c + \beta(x - \psi)H(x - \psi) + \zeta\,, \tag{1}$$

where given some data, x, y, we are trying to estimate the gradient of the first segment, α, the intercept of the first segment, c, the change in gradient from the first to second segments, β, and the breakpoint position, ψ. H is the Heaviside step function and ζ is a noise term. This cannot be solved directly through linear regression as the relationship is non-linear. We can take a linear approximation by a Taylor expansion around some initial guess for the breakpoint, $\psi^{(0)}$,

$$y \approx \alpha x + c + \beta(x - \psi^{(0)})H(x - \psi^{(0)}) - \beta(\psi - \psi^{(0)})H(x - \psi^{(0)}) + \zeta\,. \tag{2}$$

This is now a linear relationship and we can find a new breakpoint estimate, $\psi^{(1)}$, through ordinary linear regression using the `statsmodels` python package (Seabold & Perktold, 2010). We iterate in this way until the breakpoint estimate converges, at which point we stop the algorithm. If considering multiple breakpoints, the same approach is followed using a multivariate Taylor expansion around an initial guess for each of the breakpoints.

Muggeo's iterative algorithm is not guaranteed to converge on a globally optimal solution. Instead, it can converge to a local optimum or diverge. To address this limitation, we also implement bootstrap restarting (Wood, 2001), again following Muggeo's approach (Muggeo & others, 2008). The bootstrap restarting algorithm generates a non-parametric bootstrap of the data through resampling, which is then used to find new breakpoint values that may find a better global solution. This is repeated several times to escape local optima.

Model Selection

The standard algorithm finds a good fit with a given number of breakpoints. In some instances we might not know how many breakpoints to expect in the data. We provide a tool to compare models with different numbers of breakpoints based on minimising the Bayesian Information Criterion (Wit et al., 2012), which takes into account the value of the likelihood function while including a penalty for the number of model parameters, to avoid overfitting. When applied to the example in Figure 1, a model with 3 breakpoints is the preferred choice.

Features

The package includes the following features:

- Standard fit using the iterative method described by Muggeo.
- Bootstrap restarting to escape local optima.
- Bootstrap restarting with randomised initial breakpoint guesses.
- Calculation of standard errors and confidence intervals.
- Davies hypothesis test for the existence of a breakpoint.
- Customisable plots of fits.
- Customisable plots of algorithm iterations.
- Printable summary.
- Summary data output.
- Comprehensive tests.
- Model comparision with an unknown number of breakpoints, with the best fit based on the Bayesian information criterion.

The package can be downloaded through the Python Package Index. The full code is publicly available on github. Documentation, including an API reference, can be found at Read The Docs.

Acknowledgements

I acknowledge support from Thomas Hills. The work was funded by the EPSRC grant for the Mathematics for Real-World Systems CDT at Warwick (grant number EP/L015374/1).

References

Davies, R. B. (1987). Hypothesis testing when a nuisance parameter is present only under the alternative. *Biometrika*, *74*(1), 33–43.

Jekel, C. F., & Venter, G. (2019). PWLF: A Python library for fitting 1D continuous piecewise linear functions. *URL: Https://Github. Com/Cjekel/Piecewise_linear_fit_py*.

Muggeo, V. M. (2003). Estimating regression models with unknown break-points. *Statistics in Medicine*, *22*(19), 3055–3071.

Muggeo, V. M., & others. (2008). Segmented: An R package to fit regression models with broken-line relationships. *R News*, *8*(1), 20–25.

Newville, M., Stensitzki, T., Allen, D. B., Rawlik, M., Ingargiola, A., & Nelson, A. (2016). LMFIT: Non-linear least-square minimization and curve-fitting for Python. *Astrophysics Source Code Library*, ascl–1606.

Ryan, S. E., Porth, L. S., & Troendle, C. (2002). Defining phases of bedload transport using piecewise regression. *Earth Surface Processes and Landforms*, *27*(9), 971–990.

Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and statistical modeling with Python. *Proceedings of the 9th Python in Science Conference*, *57*, 61.

Toms, J. D., & Lesperance, M. L. (2003). Piecewise regression: A tool for identifying ecological thresholds. *Ecology*, *84*(8), 2034–2041.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., & others. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*(3), 261–272.

Wagner, A. K., Soumerai, S. B., Zhang, F., & Ross-Degnan, D. (2002). Segmented regression analysis of interrupted time series studies in medication use research. *Journal of Clinical Pharmacy and Therapeutics*, *27*(4), 299–309.

Wit, E., Heuvel, E. van den, & Romeijn, J.-W. (2012). 'All models are wrong...': An introduction to model uncertainty. *Statistica Neerlandica*, *66*(3), 217–236.

Wood, S. N. (2001). Minimizing model fitting objectives that contain spurious local minima by bootstrap restarting. *Biometrics*, *57*(1), 240–244.