PeakPerformance - A software tool for fitting LC-MS/MS peaks including uncertainty quantification by Bayesian inference

Jochen Nießer [1], Michael Osthege [1], Eric von Lieres [1], Wolfgang Wiechert [1,2] and Stephan Noack [1,*]

[1] Institute of Bio- and Geosciences, IBG-1: Biotechnology, Forschungszentrum Jülich, Wilhelm-Johnen-Str., 52428 Jülich, Germany
[2] Computational Systems Biotechnology (AVT.CSB), RWTH Aachen University, Forckenbeckstr., 52074 Aachen, Germany

* To whom correspondence should be addressed.

Abstract

Summary: A major bottleneck of chromatography-based analytics has been the elusive fully automated identification and integration of peak data without the need of extensive human supervision. The presented Python package `PeakPerformance` applies Bayesian inference to chromatographic peak fitting, and provides an automated approach featuring model selection and uncertainty quantification. Currently, its application is focused on data from targeted liquid chromatography tandem mass spectrometry (LC-MS/MS), but its design allows for an expansion to other chromatographic techniques.

Availability and implementation: `PeakPerformance` is implemented in Python and the source code is available on `https://github.com/JuBiotech/peak-performance`. It is unit-tested on Linux and Windows and accompanied by general introductory documentation, as well as example notebooks.

Contact: s.noack@fz-juelich.de

1 Introduction

In biotechnological research and industrial applications, chromatographic techniques are ubiquitously used to analyze samples from fermentations, e.g. to determine the concentration of substrates and products. Over the course of a regular lab-scale bioreactor fermentation, dozens to hundreds of samples and subsequently – depending on the number of analytes per sample – hundreds to thousands of chromatographic peaks may accrue. This is exacerbated by the spread of microbioreactors causing a further increase in the amount of samples per time [Hemmerich et al., 2018, Kostov et al., 2001]. While the recognition and integration of peaks by vendor software is – in theory – automated, it typically requires visual inspection and occasional manual re-integration by the user due to a large number of false positives, false negatives or incorrectly determined baselines, ultimately downgrading it to a semi-automated process. Since this is a time-consuming, not to mention tedious, procedure and introduces the problem of comparability between purely manual and algorithm-based integration, as well as user-specific differences, we instead propose a peak fitting solution based on Bayesian inference.

The advantage of this approach is the complete integration of all relevant parameters – i.e. baseline, peak area and height, mean, signal-to-noise ratio etc. – into one model through which all parameters are estimated simultaneously. Furthermore, Bayesian inference comes with uncertainty quantification for all peak model parameters, and thus does not merely yield a point estimate as would commonly be the case. It also grants access to novel metrics for avoiding false positives and negatives by rejecting signals where a) a convergence criterion of the peak fitting was not fulfilled or b) the uncertainty of the estimated parameters exceeded a user-defined threshold.

2 Implementation

`PeakPerformance` is an open source Python package compatible with Windows and Linux/Unix platforms. At the time of manuscript submission, it features three modules: pipeline, models, and plotting. Due to its modular design, `PeakPerformance` can easily be expanded by adding e.g. additional models for deviating peak shapes or different plots. Currently, the featured peak models describe peaks in the shape of normal or skew normal distributions, as well as double peaks of normal or skewed normal shape. The normal distribution is regarded as the ideal peak shape and common phenomena like tailing and fronting can be expressed by the skew normal distribution [Azzalini, 1985]. A third distribution commonly used to describe peak shapes is the exponentially modified Gaussian [Grushka, 1972] for which an implementation in `PeakPerformance` is pending.

Bayesian inference is conducted utilizing the PyMC package [Abril-Pla et al., 2023] with the external sampler `nutpie` for improved performance. Both model selection and analysis of inference data objects were realized with the ArviZ package [Kumar et al., 2019]. Since the inference data is stored alongside graphs and report sheets, users may employ the ArviZ package or others for further analysis of the results if necessary.

© The Author(s) 2024.

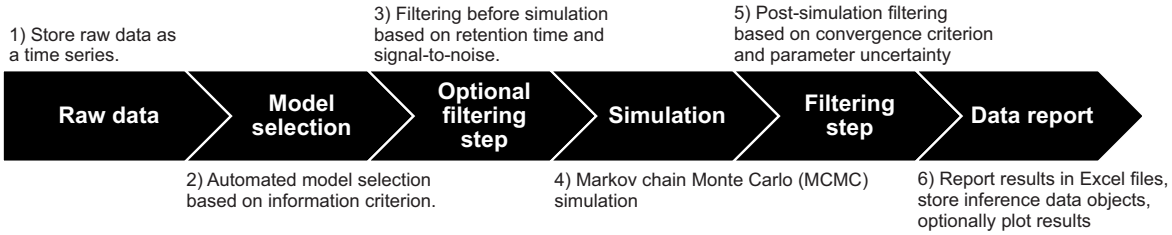

Fig. 1. Overview of the data analysis pipeline in `PeakPerformance`.

3 PeakPerformance workflow

The data analysis pipeline of `PeakPerformance` was designed in a user-friendly way and requires little to no programming knowledge (Fig. 1). As portrayed in an example notebook, only a few simple Python commands need to be executed. Instead of relying on these convenience functions, experienced users can also directly access the core functions of `PeakPerformance` for a more flexible application. Before using `PeakPerformance`, the user has to supply raw data files centered around the peak in question containing a NumPy array with time in the first and intensity in the second dimension. For each peak, such a file has to be provided according to the naming convention specified in the repository's ReadMe section and gathered in one directory. Subsequently, the raw data is parsed and listed in the automatically created Excel template file. Since targeted LC-MS/MS analyses cycle through a list of mass traces for every sample, a model type has to be assigned to each mass trace. This can be realized either by user input (directly into the Excel template) or via `PeakPerformance`'s automated model selection. The latter compares and ranks all implemented models based on either Pareto-smoothed importance sampling leave-one-out cross-validation (LOO-PIT) or the widely applicable information criterion (WAIC) [Watanabe, 2010, Vehtari et al., 2016]. Before sampling, an optional filtering step may weed out signals clearly lacking a peak to save computation time. After sampling, there is another more sophisticated filtering operation to remove false positives based on a convergence criterion – the potential scale reduction factor [Gelman and Rubin, 1992] – and the determined parameter uncertainties. Finally, the inference data object is saved alongside Excel report sheets as well as plots of the posterior probability distribution and the predictive posterior distribution against the raw data. Since most users are first and foremost interested in the peak area, a separate report sheet focusing only on that parameter is also created.

4 Validation

Several stages of validation were employed to prove the suitability of `PeakPerformance` for chromatographic peak data analysis. In the following, a short account of the methods and results are presented with a more thorough treatment in the Supplementary Information. In the first stage, fitting with normal and skew normal peak models was tested regarding the ability to reproduce the ground truth of randomly generated noisy synthetic data sets. The arithmetic means of the ratio of estimated parameter to ground truth are portrayed in Figure 2A and show values close to 1 for tests with a sample size of 500, implying a near identity between estimation and ground truth. Additionally, the normal peak model was paired with skew normally distributed noisy data and vice versa. In both cases, the location parameter σ was not reproduced well, especially by the normally distributed model. Nevertheless, the peak area and height were still identified correctly with the skew normal model and merely slightly underestimated by the normal model.

In the second stage, marginal cases in the form of slightly skewed peaks were investigated to observe whether their estimation with a normal or skew normal distribution would result in significant differences in terms of peak area and height. Here, a slight skew was defined as an α parameter of 1 resulting in distributions not visibly discernible as clearly normal or skew normal. With a sample size of 100 noisy, randomly generated data sets, we show that nearly identical estimates for peak area and height, as well as their respective uncertainties are obtained regardless of the utilized model (Fig. 2B).

In the third stage, experimental peak data was analyzed with both `PeakPerformance` (version 0.5.0) and Sciex MultiQuant (version 3.0.3) with human supervision. Beyond the comparability of the resulting peak areas portrayed in Figure 2C, it is relevant to state that 100 signals from MultiQuant (52 % of total signals) were manually modified. Of these, 22 % were false positives and 78 % were manually re-integrated. These figures are the result of a relatively high share of double peaks in the test sample which give a lot more cause for manual interference than single peaks. In contrast, however, the `PeakPerformance` pipeline was only started once and there was no cause for repetition or manual intervention after providing the initial settings for the pipeline.

5 Conclusion

`PeakPerformance` is a tool for automated LC-MS/MS peak data analysis employing Bayesian inference. Its design accommodates users with little programming experience by relying on Microsoft Excel for data input and reporting. Further, it allows strict control of model selection by the user but is not dependent on it due to the integrated automated model selection functionality. Regarding peak acceptance, it improves on vendor software solutions with more sophisticated, multi-layered metrics for decision making based on model convergence and parameter uncertainties. Finally, it allows the addition of new models with different probability distributions with just a few minor code changes, thus lending itself to expansion to data from other chromatographic techniques. In future releases of `PeakPerformance`, a feature allowing the free assembly of single peak models to obtain mixed multi peak models is planned as well as an expansion to HPLC data.

Author contributions

`PeakPerformance` was conceptualized by JN and MO. Software implementation was conducted by JN with code review by MO. The original draft was written by JN with review and editing by MO, SN, EvL, and WW. The work was supervised by SN and funding was acquired by SN, EvL, and WW.

Acknowledgments

The authors thank Tobias Latour for providing experimental LC-MS/MS data for the comparison with the vendor software MultiQuant.

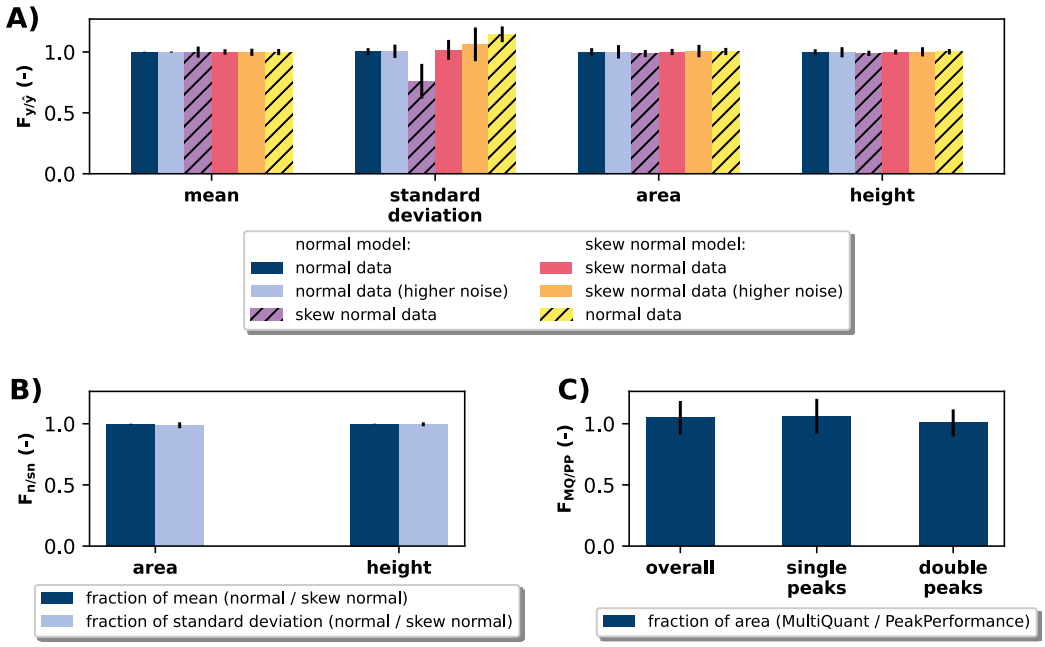

Fig. 2. Validation of results from `PeakPerformance`. **A)** Noisy synthetic data was randomly generated from one of the implemented distributions and the program's ability to infer the ground truth was observed. Portrayed are the fractions of estimated parameter to ground truth. **B)** The influence of model choice between normal and skew normal model in marginal cases with little to no skew was tested and the ratios between results from both models are plotted. **C)** Lastly, experimental data was analyzed with `PeakPerformance` version 0.5.0 and compared to results achieved with the commercial software Sciex MultiQuant version 3.0.3.

Funding

Funding was received from the German Federal Ministry of Education and Research (BMBF) (grant number 031B1134A) as part of the innovation lab "AutoBioTech" within the project "Modellregion, BioRevierPLUS: BioökonomieREVIER Innovationscluster Biotechnologie & Kunststofftechnik".

Competing interests

No competing interest is to be declared.

References

O. Abril-Pla, V. Andreani, C. Carroll, L. Dong, C. J. Fonnesbeck, M. Kochurov, R. Kumar, J. Lao, C. C. Luhmann, O. A. Martin, M. Osthege, R. Vieira, T. Wiecki, and R. Zinkov. Pymc: a modern, and comprehensive probabilistic programming framework in python. *PeerJ Comput Sci*, 9:e1516, 2023. doi: 10.7717/peerj-cs.1516.

A. Azzalini. A class of distributions which includes the normal ones. *Scand. J. Statist.*, 12:171–178, 1985.

A. Gelman and D. B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4), 1992.

E. Grushka. Characterization of exponentially modified gaussian peaks in chromatography. *Anal Chem*, 44(11):1733–8, 1972. doi: 10.1021/ac60319a011.

J. Hemmerich, S. Noack, W. Wiechert, and M. Oldiges. Microbioreactor systems for accelerated bioprocess development. *Biotechnol J*, 13(4): e1700141, 2018. doi: 10.1002/biot.201700141.

Y. Kostov, P. Harms, L. Randers-Eichhorn, and G. Rao. Low-cost microbioreactor for high-throughput bioprocessing. *Biotechnol Bioeng*, 72(3):346–52, 2001. doi: 10.1002/1097-0290(20010205)72:3<346::aid-bit12>3.0.co;2-x.

R. Kumar, C. Carroll, A. Hartikainen, and O. Martin. Arviz a unified library for exploratory analysis of bayesian models in python. *Journal of Open Source Software*, 4(33), 2019. doi: 10.21105/joss.01143.

A. Vehtari, A. Gelman, and J. Gabry. Practical bayesian model evaluation using leave-one-out cross-validation and waic. *Statistics and Computing*, 27(5):1413–1432, 2016. doi: 10.1007/s11222-016-9696-4.

S. Watanabe. Asymptotic equivalence of bayes cross validation and widely applicable information criterion in singular learning theory. *Journal of machine learning research*, 11:3571–3594, 2010.