# lintegrate: A C/Python numerical integration library for working in log-space

Matthew Pitkin[1]

1 Department of Physics, Lancaster University, Lancaster, UK, LA1 4YB

Summary

There are many situations in which the integral of a function must be evaluated numerically between given limits. For C codes, there are a range of numerical integration (sometimes called numerical quadrature) functions provided within the GNU Scientific Library (GSL) (Galassi et al., 2009). However, in situations where the integrand has an extremely large dynamic range these GSL functions can fail due to numerical instability. One way to get around numerical instability issues is to work with the natural logarithm of the function. You cannot simply integrate the logarithm of the function as this will not produce the logarithm of the integral of the original function. lintegrate provides a range of C integration functions, equivalent to functions in GSL, that allow you to integrate a function when only working with the natural logarithm of the function is computationally practical. The result that is returned is the natural logarithm of the integral of the underlying function. lintegrate also provides a Python module for accessing some of these functions in Python.

Statement of need

A particular case where the natural logarithm of a function is generally numerically favourable is when evaluating likelihoods in statistical applications. For example, the Gaussian likelihood of a model $m(\vec{\theta})$, defined by a set of model parameters $\vec{\theta}$ and given a data set $\mathbf{d}$ consisting N points, is

$$L(\vec{\theta}) \propto \exp\left(-\sum_{i=1}^{N} \frac{(d_i - m_i(\vec{\theta}))^2}{2\sigma_i^2}\right),\tag{1}$$

where σ_i^2 is an estimate of the noise variance for point i. Evaluating the exponent for a range of $\vec{\theta}$ values will often lead to a numbers that breach the limits of values that are storable as double precision floating point numbers and/or have an extremely large dynamic range. In these cases, if you wanted to marginalise (i.e., integrate) over some subset of the parameters $\vec{\theta}$, e.g.,

$$Z = \int^{\theta_1} L(\vec{\theta})\pi(\theta_1)\mathrm{d}\theta_1,\tag{2}$$

where $\pi(\theta_1)$ is the prior probability distribution for the parameter θ_1, you cannot work directly with Equation 1. Instead, it helps to work with the natural logarithm of the likelihood:

$$\ln L(\vec{\theta}) = C - \sum_{i=1}^{N} \frac{(d_i - m_i(\vec{\theta}))^2}{2\sigma_i^2}.\tag{3}$$

30 lintegrate allows you to calculate the logarithm of Z in Equation 2, while working with the
31 natural logarthim of integrands such as in Equation 3.

32 lintegrate was originally developed to marginalise probability distributions for the hierarchical
33 Bayesian inference of pulsar ellipticity distributions in Pitkin et al. (2018). In Nash & Durkan
34 (2019) and Strauss & Oliva (2021), lintegrate has been used to calculate the "true" value of
35 integrals to compare against values learned or inferred through other methods as a form of
36 validation.

Availability and development

38 lintegrate is open source code, released under a GPL-3.0 license. The source code is publically
39 hosted on GitHub, while release versions of the Python library are availble through PyPI and
40 conda-forge. Documentation of the C library functions and Python interface can be found on
41 Read the Docs, including the API and example usage.

42 Contributions to the code are welcome via the GitHub issues tracker or Pull Requests. Contin-
43 uous integration is set up to run a test suite on the code when any changes are made.

Acknowledgements

References

48 Galassi, M., Davies, J., Theiler, J., Gough, B., Jungman, G., Alken, P., Booth, M., Rossi,
49   F., & Ulerich, R. (2009). *GNU Scientific Library Reference Manual (3rd Ed.)* (B. Gough,
50   Ed.). Network Theory Ltd. ISBN: 0954612078

51 Nash, C., & Durkan, C. (2019). Autoregressive energy machines. In K. Chaudhuri & R.
52   Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning*
53   (Vol. 97, pp. 1735–1744). PMLR. https://proceedings.mlr.press/v97/durkan19a.html

54 Pitkin, M., Messenger, C., & Fan, X. (2018). Hierarchical Bayesian method for detecting
55   continuous gravitational waves from an ensemble of pulsars. *Physical Review D*, *98*(6),
56   063001. https://doi.org/10.1103/PhysRevD.98.063001

57 Strauss, R. R., & Oliva, J. B. (2021). Arbitrary conditional distributions with energy.
58   *arXiv:2102.04426*. https://arxiv.org/abs/2102.04426