The Journal of Open Source Software

gaussplotR: Fit, Predict and Plot 2D-Gaussians in R

Vikram B. Baliga[1]

1 Department of Zoology, University of British Columbia, Vancouver, British Colombia, Canada V6T 1Z4

DOI: 10.21105/joss.03074

Software

- Review
- Repository
- Archive

Editor: Kristina Riemer

Reviewers:

- @cddesja
- @brunaw

Submitted: 31 January 2021
Published: 06 April 2021

Summary

Should the need to model the relationship between bivariate data and a response variable arise, two-dimensional (2D) Gaussian models are often the most appropriate choice. For example, Priebe et. al. characterized motion-sensitive neurons in the brains of macaques by fitting 2D-Gaussian functions to neurons' response rates as spatial and temporal frequencies of visual stimuli were varied (Priebe et al., 2003). The width and orientation of these fitted 2D-Gaussian surfaces provides insight on whether a neuron is "tuned" to particular spatial or temporal domains. Two-dimensional Gaussians are also used in other scientific disciplines such as physics (Kravtsov & Berczynski, 2004; Z. Wu & Guo, 1998), materials sciences (Riekel et al., 1999), and image processing (Hanumantharaju et al., 2013; Ketenci & Gencturk, 2013), particularly in medical imaging (Qadir et al., 2021; J. Wu et al., 2019).

Fitting 2D-Gaussian models to data is not always a straightforward process, as finding appropriate values for the model's parameters relies on complex procedures such as non-linear least-squares. `gaussplotR` is an R package that is designed to fit 2D-Gaussian surfaces to data. Should a user supply bivariate data (i.e., x-values and y-values) along with a univariate response variable, functions within `gaussplotR` will allow for the automatic fitting of a 2D-Gaussian model to the data. Fitting the model then enables the user to characterize various properties of the Gaussian surface (e.g., computing the total volume under the surface). Further, new data can be predicted from models fit via `gaussplotR`, which in combination with the package's plotting functions, can enable smoother-looking plots from relatively sparse input data. In principle, tools within `gaussplotR` have broad applicability to a variety of scientific disciplines.

Statement of Need

At the time of writing, we know of no other packages in the `R` ecosystem that automatically handle the fitting of 2D-Gaussians to supplied data. The `R` package `imagefx` (Witsil, 2020) does offer the capability to predict data from a 2D-Gaussian model, but only if the parameters of the model are known *a priori*. Further, although base `R` functions such as `stats::nls()` provide the capability to determine the non-linear least-squares estimates of the parameters for a non-linear model, the burden of determining the formula for a 2D-Gaussian falls upon the user.

To counter these issues, `gaussplotR` provides users with the capability to fit 2D-Gaussian models using one of three possible formulas, along with the ability to apply constraints to the amplitude and/or orientation of the fitted Gaussian, if desired. Coupled with the ability to characterize various properties of the fitted model, along with plotting functions (as the name of the package implies), `gaussplotR` is intended to be a feature-rich package for users interested in 2D-Gaussian modeling. These capabilities are briefly explained in the next section; vignettes supplied in the package delve into even further detail.

Overview and getting started

A series of vignettes that provides detailed guidance are available on gaussplotR's GitHub page.

The function `fit_gaussian_2D()` is the workhorse of `gaussplotR`. It uses `stats::nls()` to find the best-fitting parameters of a 2D-Gaussian fit to supplied data based on one of three formula choices. Each of these formula choices is designed for a specific use case. The most generic method (and the default) is `method = "elliptical"`. This allows the fitted 2D-Gaussian to take an ellipsoid shape, and this will likely be the best option for most use cases. A slightly-altered method to fit an ellipsoid 2D-Gaussian is available in `method = "elliptical_log"`. This method follows Priebe et al., (Priebe et al., 2003) and is geared towards use with log2-transformed data. A third option is `method = "circular"`. This produces a very simple 2D-Gaussian that is constrained to have to have a roughly circular shape (i.e. spread in X- and Y- are roughly equal). Rather than place the burden on the user to determine formula choice, the function `autofit_gaussian_2D()` can be used to automatically figure out the best formula choice and arrive at the best-fitting parameters.

In some cases, the researcher may be interested in characterizing the orientation of the fitted 2D-Gaussian and comparing it to theoretical predictions. For example, studies of visual neuroscience often describe the properties of individual motion-sensitive neurons based on whether they are "speed-tuned" or whether they show independence from the speed of visual stimuli. Assessing such properties can be done via fitting a 2D-Gaussian to the response rate of a neuron for a grid of investigated spatial (X-axis) and temporal frequencies (Y-axis). Should the orientation of the fitted 2D-Gaussian lie along the diagonal of the plot, the neuron can be classified as "speed-tuned." The function `characterize_gaussian_fits()` allows for such analysis within `gaussplotR`. Following methods used in studies of visual neuroscience (Levitt et al., 1994; Priebe et al., 2003; Winship et al., 2006), the orientation and partial correlations of 2D-Gaussian data are analyzed. Features include computation of partial correlations between response variables and independent and diagonally-tuned predictions, along with Z-difference scoring.

The `predict_gaussian_2D()` function can be used to predict values from the fitted 2D-Gaussian over a supplied grid of X- and Y-values (usually generated via `expand.grid()`). This is useful if the original data are relatively sparse and interpolation of values is desired, e.g. to attain smoother-looking contours in plots.

Plotting can then be achieved via `ggplot_gaussian_2D()`, but note that the `data.frame` created by `predict_gaussian_2D()` can be supplied to other plotting frameworks such as `lattice::levelplot()`. A 3D plot can also be produced via `rgl_gaussian_2D()`.

`gaussplotR` was designed for broad applicability; there are many disciplines in which a 2D-Gaussian surface would be a useful model for describing a response to a bivariate set of inputs. Functions in `gaussplotR` are being used in an in-prep article to determine the extent of spatiotemporal tuning of motion-sensitive neurons in hummingbirds and other avian species.

Acknowledgements

We thank Douglas R. Wylie, Douglas L. Altshuler, and Graham Smyth for help in working with 2D-Gaussian data.

References

Hanumantharaju, M. C., Ravishankar, M., & Rameshbabu, D. R. (2013). Design and FPGA implementation of an 2D gaussian surround function with reduced on-chip memory utilization. *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 604–609. https://doi.org/10.1109/ICACCI.2013.6637241

Ketenci, S., & Gencturk, B. (2013). Performance analysis in common color spaces of 2D gaussian color model for skin segmentation. *Eurocon 2013*, 1653–1657. https://doi.org/10.1109/EUROCON.2013.6625198

Kravtsov, Yu. A., & Berczynski, P. (2004). Description of the 2D gaussian beam diffraction in a free space in frame of eikonal-based complex geometric optics. *Wave Motion*, *40*(1), 23–27. https://doi.org/10.1016/j.wavemoti.2003.12.012

Levitt, J. B., Kiper, D. C., & Movshon, J. A. (1994). Receptive fields and functional architecture of macaque V2. *Journal of Neurophysiology*, *71*(6), 2517–2542. https://doi.org/10.1152/jn.1994.71.6.2517

Priebe, N. J., Cassanello, C. R., & Lisberger, S. G. (2003). The Neural Representation of Speed in Macaque Area MT/V5. *The Journal of Neuroscience*, *23*(13), 5650–5661. https://doi.org/10.1523/JNEUROSCI.23-13-05650.2003

Qadir, H. A., Shin, Y., Solhusvik, J., Bergsland, J., Aabakken, L., & Balasingham, I. (2021). Toward real-time polyp detection using fully CNNs for 2D gaussian shapes prediction. *Medical Image Analysis*, *68*, 101897. https://doi.org/10.1016/j.media.2020.101897

Riekel, C., Bränden, C., Craig, C., Ferrero, C., Heidelbach, F., & Müller, M. (1999). Aspects of x-ray diffraction on single spider fibers. *International Journal of Biological Macromolecules*, *24*(2), 179–186. https://doi.org/10.1016/S0141-8130(98)00084-1

Winship, I. R., Crowder, N. A., & Wylie, D. R. W. (2006). Quantitative Reassessment of Speed Tuning in the Accessory Optic System and Pretectum of Pigeons. *Journal of Neurophysiology*, *95*(1), 546–551. https://doi.org/10.1152/jn.00921.2005

Witsil, A. J. C. (2020). *Imagefx: Extract features from images*. https://CRAN.R-project.org/package=imagefx

Wu, J., Zhang, S., Xiao, Z., Zhang, F., Geng, L., Lou, S., & Liu, M. (2019). Hemorrhage detection in fundus image based on 2D gaussian fitting and human visual characteristics. *Optics & Laser Technology*, *110*, 69–77. https://doi.org/10.1016/j.optlastec.2018.07.049

Wu, Z., & Guo, L. (1998). Electromagnetic scattering from a multilayered cylinder arbitrarily located in a gaussian beam, a new recursive algorithms. *Progress in Electromagnetics Research*, *18*, 317–333. https://doi.org/10.2528/pier97071100