Worksheet: Linear regression for the estimation of nonlinear receptive field models

Joaquin Rapela

February 12, 2025

Following Rapela et al. [2006], to model responses of visual cell in animals to stimulation with natural images, we project these images into a small number of basis functions (learned using the projection pursuit regression algorithm [Friedman and Stuetzle, 1981]), $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_L$. Below we call *relevant dimensions* to these basis functions. We denote the projections of the nth image, $\mathbf{v}_n$, into the lth relevant dimension, $\boldsymbol{\alpha}_l$, by $x_{nl} = \boldsymbol{\alpha}_l^\intercal \mathbf{v}_n$. Then we model the number of spikes produced by the cell in response to the nth image, y_n, with a multivariate polynomial of degree D. Equation 1 shows a model of the response of a cell to the nth image with two basis functions, $L = 2$, and a multivariate polynomial of degree $D = 3$.

$$
\begin{aligned}
y_n = & \beta_0 + \beta_1 x_{n1} + \beta_2 x_{n2} + \\
& \beta_{11} x_{n1}^2 + 2\beta_{12} x_{n1} x_{n2} + \beta_{22} x_{n2}^2 + \\
& \beta_{111} x_{n1}^3 + 3\beta_{112} x_{n1}^2 x_{n2} + 3\beta_{122} x_{n1} x_{n2}^2 + \beta_{222} x_{n2}^3 \\
= & [1, x_{n1}, x_{n2}, x_{n1}^2, 2x_{n1}x_{n2}, x_{n2}^2, x_{n1}^3, 3x_{n1}^2 x_{n2}, 3x_{n1}x_{n2}^2, x_{n2}^3]
\begin{bmatrix}
\beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_{11} \\ \beta_{12} \\ \beta_{22} \\ \beta_{111} \\ \beta_{112} \\ \beta_{122} \\ \beta_{222}
\end{bmatrix}
\end{aligned}
\tag{1}
$$

Thus the above model is nonlinear on the input images, but linear on the model parameters. To estimate these parameters we use the method of least squares.

In this worksheet you will

1. do cross-validation to learn the optimal number of relevant dimensions, L, and the optimal degree of the multivariate polynomial, D.
2. plot residuals to check the goodness of fit of the model.
3. computer bootstrap confidence intervals to test the significance of model coefficients.

It is a good practice to first test your methods with simulated data, for which you know the ground truth. Thus, we will first perform this exercise on data from a simulated cell and then on data recorded from a complex cell in primary visual cortex of an anaesthetised cat [Felsen et al., 2005].

1 Simulated complex cell

We simulated responses of a complex cell using the energy model [Adelson and Bergen, 1985] in Eq. 2. This model uses two identical Gabor relevant dimensions, $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$, that are 180° out of phase. The relevant dimensions used for this simulation appear in Figure 1.

$$y_n = (\boldsymbol{\alpha}_1^\intercal \mathbf{v}_n)^2 + (\boldsymbol{\alpha}_2^\intercal \mathbf{v}_n)^2 \tag{2}$$

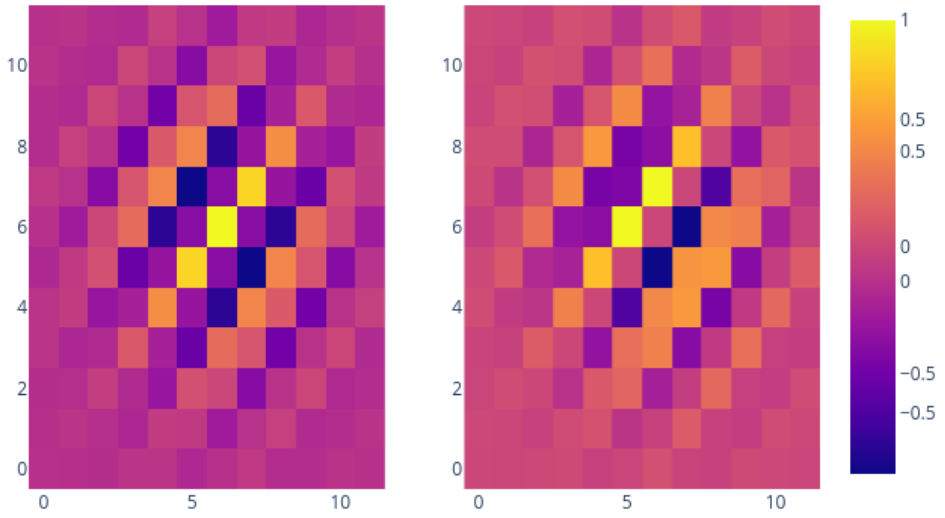

Figure 1: Relevant dimensions used for the complex cell simulation.

All the code for this worksheet appears in the subdirectory worksheets/06_linearRegression/code/scripts of the class repo. In particular, the code to simulate a complex cells is in script doSimulateComplexCell.py.

Prior to running this code install the required dependencies by running, from the subdirectory worksheets/06_linearRegression/code/scripts, the following commands:

```
pip install -r requirements.txt
pip install -r requirements2.txt
```

The script doLinearRegressionVisualCell.py implements all the functionality required for this worksheet, but it has a few lines that you will need to complete to make it work. In this exercise

you will perform all the mathematical operations required to estimate the regression coefficients (e.g., matrix inversions, tranposes, and multiplications). However, it is better to use numerical packages (e.g., scipy) as they take care of important numerical details.

The first section you will have to complete is on file doLinearRegressionVisualCell.py:

```
1    # calculate regression coefficients with train data using L2 regularisation
2    X = utils.buildDataMatrix(px=px, order=order, nRDs=n_RDs)
3    X_train, X_test, Y_train, Y_test = \
4        sklearn.model_selection.train_test_split(X, Y,
5                                                 test_size=test_percentage)
6    I = np.eye(X.shape[1])
7    coefs = ...
```

Lines 3-5 partition the data into train and test sets. We will use the train set to estimate model parameters and the test set to asses the model predictive power without overfitting.

You will need to complete line 7 to calculate the coefficients using the train set and the formula given in class to compute regression coefficients using L2 regularisation. You may want to use functions np.matmul for matrix-matrix or matrix-vector multiplication, np.transpose to transpose an array, np.linalg.inv for matrix inversion and/or np.linalg.solve to solve systems of equations.

The second section you will have to complete is on file doLinearRegressionVisualCell.py:

```
1      # calculate residuals
2      fitted_train = ...
3      residuals_train = Y_train - fitted_train
4
5      # compute correlation coefficient on test data
6      fitted_test = ...
7      rho_test = np.corrcoef(Y_test, fitted_test)[0, 1]
```

Given the coefficients estimated above, here you need to compute the fitted response (or dependent variable) for the train and test segments. We use the fitted train response to calculate the residuals for the train data. We use the fitted test response to calculate their correlation coefficients with the observed test dependent variable.

After you complete the missing parts of the code, you can run it with the parameters set for the simulated complex cell specified in the shell script doAnalyzeSimulatedComplexCell.csh. In this script you can specify the number of relevant dimensions (parameter `--nRDs`) and the polynomial order (parameter `--order`) that you want to use. Running this script with two relevant dimensions and with a polynomial order 2 (i.e., parameters `--order=2 --nRDs=2`) I obtained the estimated relevant dimensions in Figure 2, the coefficients in Figure 3, the histogram of residuals in Figure 4 and the fitted responses in Figure 5.

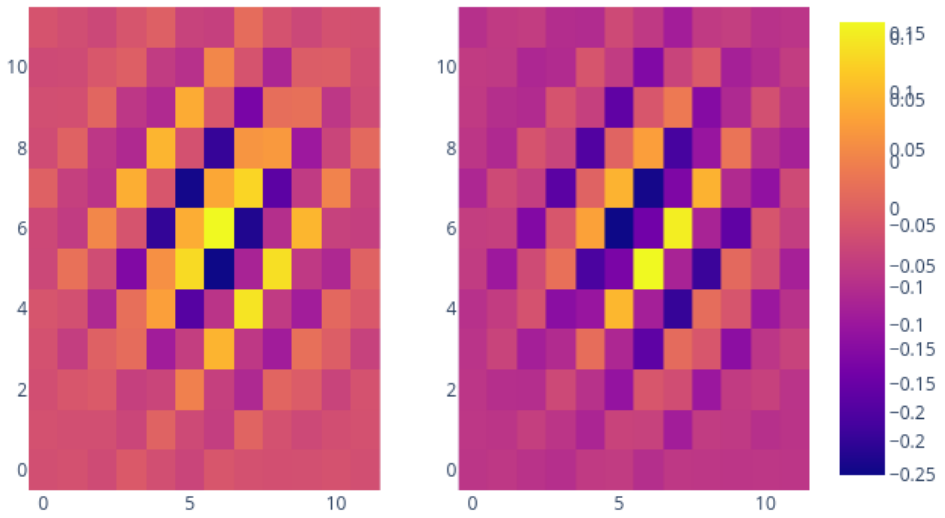

Figure 2: Relevant dimensions estimated for simulated data of a complex cell. Compare with Figure 1

.

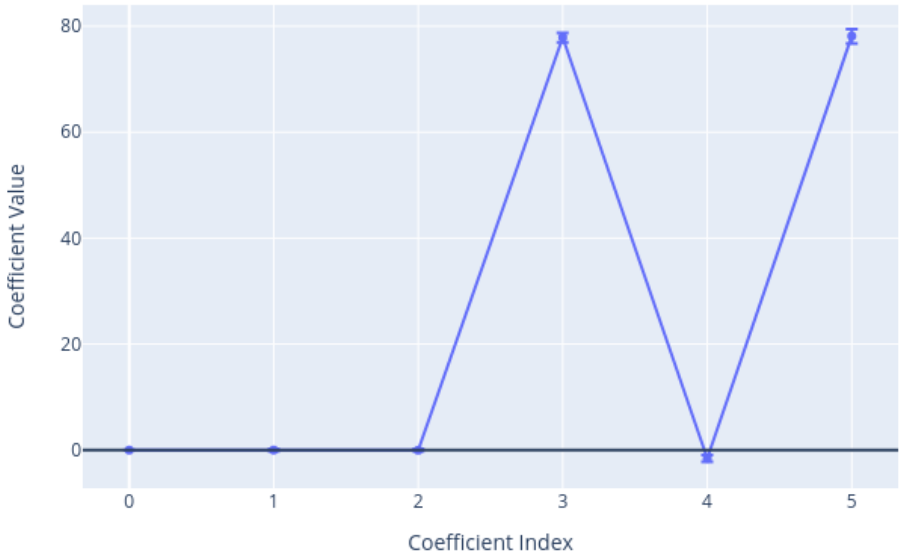

Figure 3: Coefficients estimated for simulated data of a complex cell. Only the coefficients with indices 2 and 4 in the plot, corresponding to β_{11} and β_{22} in Eq. 1, are significantly different from zero. This is the correct solution, as it can be seen from the model in Eq. 2

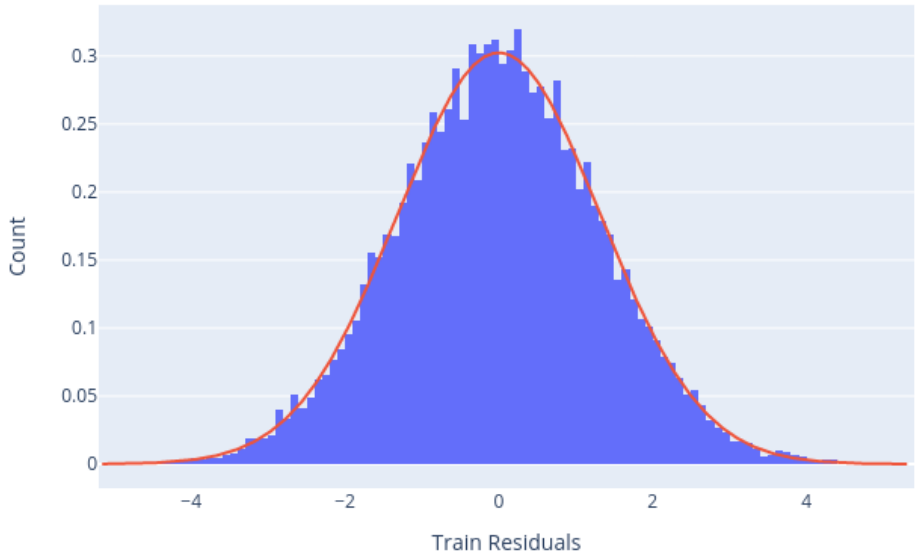

Figure 4: Histogram of train residuals for the simulated complex cell (blue) and density of Normal distribution with the same mean and variance as that of the train residuals (red).

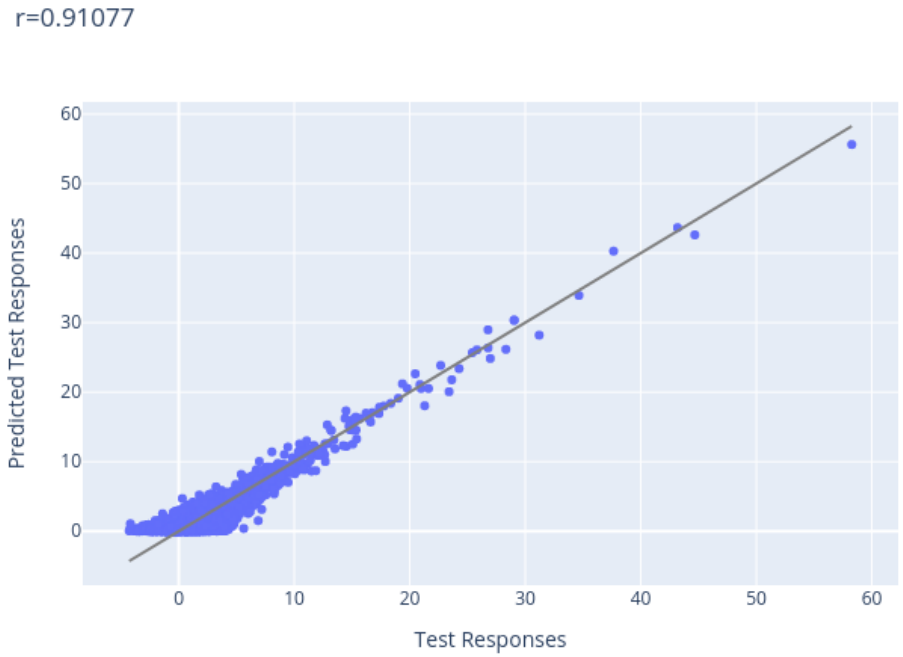

Figure 5: Cell responses versus predictions for the simulated complex cell. Their correlation coefficient appears in the title.

Vary the number of relevant dimensions (parameter `--nRDs` in doAnalyzeSimulatedComplexCell.csh) and model order (parameter `--order` in doAnalyzeSimulatedComplexCell.csh) to select their optimal values. Plot Figure 2-5 for the best and a poor set of parameters.

2 Real complex cell

The shell script doAnalyzeRealComplexCell.csh contains the parameters required to characterise a complex cell in primary visual cortex of an anaesthetised cat [Felsen et al., 2005]. Find the best set of parameters to characterise this cell. Plot Figure 2-5 for the best and a poor set of parameters.

References

E.H. Adelson and J.R. Bergen. Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America A*, 2:284–299, 1985.

G. Felsen, J. Touryan, F. Han, and Y. Dan. Cortical sensitivity to visual features in natural scenes. *PLoS Biology*, 3(10):e342, 9 2005. doi: 10.1371/journal.pbio.0030342. URL `http://dx.doi.org/10.1371%2Fjournal.pbio.0030342`.

J.H. Friedman and W. Stuetzle. Projection pursuit regression. *Journal of the American Statistical Association*, 76(376):817–823, 1981.

J. Rapela, J.M. Mendel, and N.M. Grzywacz. Estimating nonlinear receptive fields from natural images. *Journal of Vision*, 6(4):441–474, 2006.