BestFitPoly.m: A MATLAB script to find best-fit polynomials

Ricardo Murphy (ricardo_murphy@fastmail.fm)
Edinburgh, UK

Introduction

The MATLAB script BestFitPoly.m accompanies the paper by Murphy (2024). It was designed to find 'best-fit' polynomials for data that might otherwise be analysed by one-way ANOVA. Thus the data consist of r treatment levels (x) and associated responses (y), where both x and y represent quantitative variables on a continuous scale. Furthermore, each treatment level must be represented by a group of at least three replicates, and all observations are independent (i.e. no repeated measures). The best-fit polynomial has the form:

$$y = \sum_{i=0}^{u} \beta_i x^i \qquad (0)$$

where in general one or more of the coefficients (β_i, $i \neq 0$) may be zero, and u is the degree of the polynomial. We must have $u \leq r-1$ if the polynomial is not to be underdetermined. In general it is also advisable to set $u \leq 3$ (Kutner *et al*. 2004).

Various criteria are used to determine 'best-fit'. These criteria are discussed in texts such as Kutner *et al*. (2004) and Miller (2002). For four of the criteria (MSE, AIC, BIC and PRESS) best-fit is indicated by a minimum value. For the remaining criterion (F-to-enter), model selection is based on an F test as described in the subsection `Critical F-to-enter value: 4.0` below. In general more than one polynomial model may be selected, because different criteria may select different models. The set of candidate best-fit models can then be compared according to various additional criteria such as statistical significance of parameters, number of parameters, width of confidence and prediction intervals, and the appearance of residual and normal plots. With regard to the latter, normality is not such an important issue with the present approach, because the nonparametric bootstrap is used to assess statistical significance and to construct confidence and prediction intervals (Efron and Tibshirami, 1993, Efron and Hastie, 2016). Moreover, since bootstrapping is done separately for each treatment level, there is no need to assume constant variance. In fact, even for normally distributed data, the use of the bootstrap is necessary with best-fit model selection because the F and t statistics used to assess statistical significance will not follow their expected distributions under the null hypothesis (NH) of no treatment effect (Miller, 2002).

Having extracted the file BestFitPoly.zip, you should now have a folder (BestFitPoly) containing two files (BestFitPoly.m and BestFitPoly_Config.txt) and two subfolders: test_data containing example data sets and test_data_output containing example output.

IMPORTANT NOTE: Items flagged with EXPERIMENTAL are still under development and need to be tested with Monte Carlo simulations. I currently don't have the resources to do this, but hopefully this will change in the next few months.

BestFitPoly_Config.txt

This is the BestFitPoly configuration file and it must be present in the same folder/directory as BestFitPoly.m. Set various options in this file before running BestFitPoly.m.

`Data folder/directory: test_data`

Set the name of the folder/directory (in this case `test_data`) which contains the input data file, and to which output will be sent. A path may be specified, including a drive letter if needed, e.g. `D:/mypath/myfolder` or in Windows `C:\mypath\myfolder`.

`Data text file name (excluding .txt): test_data_space`

Name of the text file containing the data to be analysed (here `test_data_space.txt`). The file name extension must be `.txt`, but do not include that extension here. The file must contain two columns of data, the predictor variable (*x*) followed by the response variable (*y*). Both variables must be quantitative and on a continuous scale. Since BestFitPoly.m was designed to analyse data that might otherwise be analysed by one-way ANOVA, the data should be arranged in groups (treatment levels), with at least three replicates per group. The first row must contain the variable names, which may not include spaces or quotation marks. The data may be whitespace, comma or tab delimited. See `test_data_space.txt`, `test_data_comma.txt` and `test_data_tab.txt` in the folder `BestFitPoly\test_data` for examples.

`New x values (for predictions; x1 x2 x3...): 0.25 0.5 0.75`

If predicted responses and effect sizes are required at *x* values not included in the original (training) data, specify those *x* values here. They may be whitespace or comma delimited.

`Center the x data (yes/no): yes`

Whether or not to center the *x* data. Centering is recommended to reduce the effects of multicolinearity (Kutner *et al*. 2004).

`Maximum polynomial degree: 3`

Polynomials will be fitted to the data up to and including the value given here. In general one should be cautious about specifying a value higher than three, since such high degree polynomials may exhibit erratic behaviour (Kutner *et al*. 2004). The program will limit the maximum degree to $r-1$, where r is the number of groups.

`Compute t values based on parameters or maxeffect: maxeffect`

If set to `maxeffect`, t values are calculated at the value of x which maximises the magnitude of the predicted treatment effect (see Eq. (2) below). This was the method employed by Murphy (2024). If set to `parameters`, t is taken as the maximum t value for the model parameters (without regard to sign); this is an EXPERIMENTAL feature.

`Hierarchical polynomial fitting (yes/no): no`

With a hierarchical approach, if a polynomial of degree k is fitted, then all terms of lower order are retained. For example, if a quadratic is fitted, then there must also be a linear term; one would not fit the quadratic term but omit the linear term. Hierarchical fitting is commonly used, but in principle it might be somewhat restrictive.

`Always choose the minimum-parameter best-fit model (yes/no): yes`

In principle a best-fit criterion such as PRESS might exhibit more than one minimum. Setting this option to `no` ensures that the absolute minimum is chosen. On the other hand, `yes` means that the minimum corresponding to the model with the smallest number of parameters is chosen, even if it is not the absolute minimum. In other words, `yes` ensures that the most parsimonious best-fit model is selected.

`Critical level of significance (alpha): 0.05`

The critical level of significance (α) to use for confidence intervals and prediction intervals. These intervals will then aim at $100\times(1-\alpha)\%$ coverage (e.g. $\alpha = 0.05 \Rightarrow 95\%$).

`Print model-fitting details to screen and ...printout.txt (yes/no): yes`

Results of the analysis are always saved to a file *`filename`*`…polynomial-…analysis.txt` in the data folder/directory, where *`filename`* is the name of the input data file, excluding the `.txt` extension (e.g. `test_data`). Setting this option to `yes` will cause some information to be printed to the screen and saved to a file *`filename`*`_printout.txt`, specifically the output from MTALB function `fitlm`, which is used to fit

the polynomial models.

`Direction of F-to-enter search (forward/backward): backward`

The *F*-to-enter search may be performed forwards (increasing number of model parameters, *m*) or backwards (decreasing *m*). The latter may be preferred (Kutner *et al*. 2004).

`Critical F-to-enter value: 4.0`

For each *m*, the model with the smallest error sum of squares (*SSE*) is chosen. Then the model with *m* parameters is compared with the model with *m*−1 parameters by computing the following *F* statistic: $F = (n-m)(SSE_{m-1} - SSE_m)/SSE_m$, where *n* is the number of observations. If *F* exceeds the given critical value, the model with *m* parameters is retained. Four is a popular choice.

`Save model fitted values and residuals (yes/no): yes`

If `yes`, best-fit model predictions ($\hat{y}_i$, for the i^{th} observation; "yhat"), raw residuals ($y_i - \hat{y}_i$), standardised residuals ($\left(y_i - \hat{y}_i\right)/\mathrm{SD}\left(y_i - \hat{y}_i\right)$) and studentised residuals ($\left(y_i - \hat{y}_{i,-i}\right)/\mathrm{SD}\left(y_i - \hat{y}_{i,-i}\right)$, where $\hat{y}_{i,-i}$ is computed with the i^{th} observation omitted when fitting the model) are saved to a file *filename*_polynomial-…_resids_model-#.txt. # is a number assigned to the model (this number also appears in the …`analysis.txt` file).

`Number of bootstrap samples: 2000`

Nonparametric bootstrapping is used to estimate statistical significance, confidence intervals for model parameters and mean responses, and prediction intervals for new observation. Efron and Hastie (2016) recommend using 2000 bootstrap samples for the estimation of confidence intervals. Bootstrapping is performed on the residuals obtained by subtracting the group means. An alternative approach is to bootstrap (*x*,*y*) pairs (Efron and Tibshirami, 1993; Davison and Hinkley, 1997), but this is not implemented at present.

`Constant variance (yes/no): no`

If `no`, bootstrap samples will be generated for each treatment level (group) separately. If `yes`, residuals for all groups will be pooled prior to bootstrapping. The former is appropriate if there is evidence of heteroscedasticity (or one does not wish to assume homoscedasticity).

`Save bootstrapped residuals (yes/no): yes`

For each group (treatment level) a set of centered residuals is formed by subtracting the mean *y* value for that group. Bootstrap samples (i.e. random samples with replacement) are

then taken from each set of centered residuals. Residuals from different groups are not mixed, thus allowing for the possibility of a nonconstant variance. Another advantage of bootstrapping is that normality is not assumed. Choose `yes` to save the bootstrap samples to a file *filename*…boot-resids.txt.

```
Save bootstrapped test statistics (yes/no): yes
```

To allow the assessment of statistical significance, *F* and *t* values are bootstrapped under the NH of no treatment effect. Since the bootstrapped residuals are centered, this NH is forced to be true. For each set of bootstrap samples (one sample per treatment level) and for each best-fit criterion (AIC etc.), a best-fit model is selected in the usual way. Then *F* and *t* values are computed, relative to a constant model in which the treatment level (x) has no effect on the response (y). Specifically, if the best-fit model has m parameters:

$$F = \frac{(SSE_1 - SSE_m)/(m-1)}{SSE_m/(n-m)} \tag{1}$$

where SSE_1 is the error sum of squares with a single mean fitted to all the data (no treatment effect). If "`Compute t values...`" is set to `maxeffect`, the *t* value is calculated at the value of x ($x_{\max}$) which maximises the magnitude of the predicted treatment effect, $\left|\hat{y}(x) - \hat{y}(x_1)\right|$, relative to the response of the first group:

$$t = \frac{\hat{y}(x_{\max}) - \hat{y}(x_1)}{\sqrt{\mathbf{d}'\mathbf{C}_{-1,-1}\mathbf{d}}} \tag{2}$$

where $\mathbf{C}_{-1,-1}$ is the variance-covariance matrix of model parameters with the first row and first column omitted (i.e. those elements corresponding to the intercept β_0, which does not contribute to the effect size), and $\mathbf{d}'$ is a row vector of differences in powers of x, e.g.

$$\mathbf{d}' = \begin{bmatrix} x_{\max} - x_1 & x_{\max}^2 - x_1^2 & x_{\max}^3 - x_1^3 \end{bmatrix} \tag{3}$$

for a third degree polynomial [based on Kutner *et al*. (2004)]. If "`Compute t values...`" is set to `parameters`, *t* is taken as the maximum *t* value for the best-fit model parameters (without regard to sign). The `parameters` option is EXPERIMENTAL.

An *F* value (F_{model}) is also bootstrapped for each best-fit model as determined for the original experimental data (not the bootstrap samples). In addition, an EXPERIMENTAL lack-of-fit (LOF) *F* value is bootstrapped for each such model, relative to a "full model" in which each of the r groups is fitted with its own mean y value. First, for each set of bootstrap samples, an initial LOF *F* value is computed as

$$F^*_{\text{LOF}} = \frac{(SSE_1 - SSE_r)/(r-1)}{SSE_r/(n-r)} \tag{3}$$

where SSE_r is the error sum of squares for the full model. Thus F^*_{LOF} measures the LOF when fitting a single mean to all the data, relative to fitting r means. But of course in reality there is no lack of fit, because the bootstrapped residuals are centered. Hence the NH of zero lack of fit is forced to be true. To obtain an LOF F value for each best-fit model (as determined for the original data), the df in the numerator of Eq. (3) is adjusted in accordance with the df of that best-fit model ($n - m$):

$$F_{\mathrm{LOF}} = F^*_{\mathrm{LOF}} \times \frac{r-1}{r-m} \tag{4}$$

i.e. numerator df = $(n - m) - (n - r) = r - m$.

The empirical distributions of these statistics (F, t, F_{model} and F_{LOF}) are used to compute the bootstrapped p values reported in the …`analysis.txt` file. Choose `yes` to save these statistics to files *filename*…boot_F/F_model/t/LOF.txt.

```
Save bootstrapped parameters and predictions (yes/no): yes
```

To generate confidence intervals for model parameters and mean responses, and prediction intervals for new observations, the bootstrapped residuals are augmented with the group means (so restoring the treatment effect, if there is one). Then for each best-fit model (#, as determined for the original data, not the bootstrap samples), BCa bootstrapped confidence intervals are computed for the model parameters, the error (residual) SD and mean responses using the MATLAB function `BCa_bootstrap` provided by Van Snellenberg (2018). BCa confidence intervals are also computed for the mean effect size for each group, calculated relative to the predicted mean response of the first group (x_1). Corresponding prediction intervals for new observations and effect sizes are computed using the percentile method (Efron and Tibshirami, 1993), as described in the Appendix. All these results are reported in the …`analysis.txt` file. Choose `yes` to save the bootstrapped parameter estimates and predictions to the following files: *filename*…boot-paras_model-#.txt, *filename*…boot-mean_response_model-#.txt, *filename*…boot-new_observation_model-#.txt, *filename*…boot-mean_effect_model-#.txt and *filename*…boot-new_effect_model-#.txt.

```
Random number generator: default
```

See the MATLAB function `rng` for other possible choices.

```
RNG seed (< 0 => shuffle): 0
```

The random number generator seed (0 is MATLAB's default). Picking a negative integer will result in a seed based on the current time ("shuffle").

Running the BestFitPoly.m script

There are at least two ways to run the script:

(1) From a terminal window, type `matlab -r BestFitPoly`. Of course BestFitPoly.m must be present in the current folder/directory.

(2) Open and Run it from within a MATLAB session. **WARNING:** the MATLAB workspace will be cleared!

Program output

As mentioned above, BestFitPoly.m sends its output to a file *`filename`*`…polynomial-…analysis.txt` in the data folder/directory. The output is in three parts. The first part indicates the best-fit polynomial models according to the various criteria (MSE, AIC, BIC PRESS and *F*-to-enter). Such models are marked with an asterisk. The second part gives *p* values, parameter estimates, confidence intervals and prediction intervals based on the assumption that the residuals are drawn from a normal distribution ("NORMAL ERROR MODEL"). *F* values are calculated according to Eq. (1), and $p(F)$ values are determined from the central *F* distribution with $m-1$, $n-m$ df. Similarly F_{LOF} is calculated according to:

$$F_{\mathrm{LOF}} = \frac{\left(SSE_m - SSE_r\right)/\left(r-m\right)}{SSE_r/(n-r)} \tag{5}$$

and $p(F_{\mathrm{LOF}})$ is determined from the central *F* distribution with $r-m, n-r$ df. *t* values are calculated as described above, and $p(t)$ is determined from the central *t* distribution with $n-m$ df. Two $p(t)$ value are reported. p(t)I is the estimated Type I Error rate (rate of false positives) for a two-sided test. p(t)III can be regarded either as the estimated Type I Error rate for a one-sided test (assuming you predicted an effect in the direction actually observed), or as the estimated maximum Type III Error rate, i.e. the maximum probability of concluding that the true treatment effect is in the observed direction, when in reality it is in the opposite direction. The advantage of a Type III Error-based test is that it is as powerful as a one-sided Type I Error-based test, but it is nevertheless two-sided. The price to be paid for this advantage is the abandonment of the NH of zero treatment effect. That is, this NH (or more generally the NH of zero difference in population locations) is taken to be *a priori* false. For the normal error model, p(t)III is simply one-half p(t)I. See Murphy (2018; 2024) for further discussion.

Confidence intervals for parameter estimates and the parameter variance-covariance matrix are as reported by MATLAB's `fitlm` function, while the confidence interval for the error (residual) SD is estimated from the χ^2 distribution with $n-m$ df. Confidence intervals for the mean response and prediction intervals for new observations are as reported by `fitlm`. Effect sizes are calculated relative to $\hat{y}(x_1)$. A $100\times(1-\alpha)$ per cent confidence interval for the

mean effect size at $x = x_i$ is then given by $\hat{y}(x_i) - \hat{y}(x_1) \pm t_{\alpha/2,n-m}\sqrt{\mathbf{d}_i'\mathbf{C}_{-1,-1}\mathbf{d}_i}$, where $t_{\alpha/2,n-m}$ denotes the upper tail of the central t distribution with $n-m$ df and $\mathbf{d}_i'$ ′ is a row vector of differences in powers of x, e.g.

$$\mathbf{d}_i' = \begin{bmatrix} x_i - x_1 & x_i^2 - x_1^2 & x_i^3 - x_1^3 \end{bmatrix} \tag{6}$$

for a third degree polynomial ($m = 4$). Similarly, a prediction interval for a new estimate of effect size (involving new observations at x_1 and x_i) is given by $\hat{y}(x_i) - \hat{y}(x_1) \pm t_{\alpha/2,n-m}\sqrt{2MSE + \mathbf{d}_i'\mathbf{C}_{-1,-1}\mathbf{d}_i}$, where MSE is the mean square error, e.g. $MSE = SSE_m/(n-m)$ for a model with m parameters. These data are presented in tables, with the original x values in the first column, in ascending order. If predictions are to be made at new x values, these are given in the last rows of the table. According to the analysis of Murphy (2024) predictions are best made with the full model (i.e. the group means), which is presented after the best-fit polynomials. For the full model, linear interpolation is used for predictions at new x together with the associated confidence and prediction intervals.

The third part of the output has the same format as the second, but the results are derived from the analysis of bootstrap samples as described above and in the Appendix. For assessing the statistical significance of the best-fit models, the bootstrapped estimates of $p(F)$ and $p(t)$ are to be preferred, since even if the normal error model is valid, the p values resulting from the model-selection procedure will be too low (Miller, 2002). In general bootstrapped confidence intervals and, in principle, bootstrapped prediction intervals should also be regarded as more reliable than those based on the normal error model, unless there is good evidence that the residuals are drawn from a common normal distribution. In this regard, normal plots and plots of the residuals vs $\hat{y}$ are provided, together with plots of confidence intervals for the mean response and prediction intervals for new observations vs x. These graphs are plotted to the screen and saved to .png files. Note however that bootstrapped prediction intervals are currently an EXPERIMENTAL feature, not yet fully tested; the same goes for the bootstrapped LOF tests (see above).

Appendix: Bootstrapped prediction intervals (EXPERIMENTAL)

The method used here to generate bootstrapped prediction intervals for new observations and effects is based on the method described by Nielsen (2020), which in turn is based on the work of Kumar and Srivistava (2012). For the b^{th} bootstrap sample, and for the j^{th} observation in the i^{th} group, we form the residual

$$\varepsilon_{b,i,j} = q_{b,i} + v_{b,i,j} \tag{7}$$

where $q_{b,i}$ represents variation in the model prediction for y_i, and $v_{b,i,j}$ reflects the intrinsic variability of the y observations plus any model bias. An estimate of $q_{b,i}$ is given by

$$q_{b,i} = \sum\nolimits_b \hat{y}_b(x_i) \big/ B - \hat{y}_b(x_i) \tag{8}$$

where B is the number of bootstrap samples, and $\hat{y}_b(x_i)$ is the model estimate of the mean response obtained with the b^{th} bootstrap sample. The estimate of $v_{b,i,j}$ used here is the validation error:

$$v_{b,i,j} = y_{i,j} - \hat{y}_b(x_i) \tag{9}$$

where the observation $y_{i,j}$ is *not* in that bootstrap sample ("leave one out"). Specifically, for each b and i, j is selected at random from the set of observations for the i^{th} group that are not in the b^{th} bootstrap sample. In the unlikely event that there are no such observations, j is chosen at random from the full set of observations for the i^{th} group. Having formed the set of residuals given by Eq. (7), a prediction interval for y_i is obtained by taking the $100\times\alpha/2\%$ lower and upper percentiles of their distribution, and augmenting the intervening interval with the mean response predicted by the model fitted to the original data ($\hat{y}(x_i)$). A similar approach is used to generate prediction intervals for effect size, but instead of Eqs (8) and (9) we have

$$q_{b,i} = \sum\nolimits_b \left(\hat{y}_b(x_i) - \hat{y}_b(x_1)\right) \big/ B - \left(\hat{y}_b(x_i) - \hat{y}_b(x_1)\right) \tag{10}$$

$$v_{b,i,j,k} = \left(y_{i,j} - y_{1,k}\right) - \left(\hat{y}_b(x_i) - \hat{y}_b(x_1)\right) \tag{11}$$

where k is selected at random from the set of observations in the 1st group that are not in the b^{th} bootstrap sample, or from the full set of observations in the 1st group in the unlikely event that all these observations are in the b^{th} bootstrap sample. Linear interpolation is used to estimate prediction intervals for x values that are not in the original data (i.e. not one of the groups).

References

Davison, A. C. and D. V. Hinkley (1997). *Bootstrap Methods and their Application*. New York, Cambridge University Press.

Efron, B. and T. Hastie (2016). *Computer Age Statistical Inference*, Cambridge University Press.

Efron, B. and R. T. Tibshirami (1993). *An introduction to the bootstrap*. New York, Chapman & Hall.

Kumar, S. and A. N. Srivistava. (2012). Bootstrap Prediction Intervals in Non-Parametric Regression with Applications to Anomaly Detection. https://ntrs.nasa.gov/citations/20130014367.

Kutner, M. H., C. J. Nachtsheim, E. L. Newman and W. Li (2004). *Applied Linear Statistical Models.* New York, McGraw-Hill/Irwin.

Miller, A. (2002). *Subset Selection in Regression*. Boca Raton, Chapman & Hall/CRC.

Murphy R. (2018). On the use of one-sided statistical tests in biomedical research. *Clinical and Experimental Pharmacology and Physiology*, **45**, 109-114.

Murphy R. (2024). A comparison of one-way ANOVA and best-fit polynomials for the analysis of quantitative data. *Submitted*.

Nielsen, D. S. (2020). Bootstrapping prediction intervals. https://saattrupdan.github.io/2020-03-01-bootstrap-prediction/.

Van Snellenberg, Jared X. (2018). BCa_bootstrap (https://se.mathworks.com/matlabcentral/fileexchange/69119-bca_bootstrap), MATLAB Central File Exchange. Retrieved 7 Nov. 2020.