STAT 704 Introduction to generalized additive models

Adapted from Timothy Hanson

Department of Statistics, University of South Carolina

Stat 704: Data Analysis I

Sections 11.4 and 11.5

Consider a linear regression problem:

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \epsilon_i,$$

where $e_1, \ldots, e_n \overset{iid}{\sim} N(0, \sigma^2)$.

- Diagnostics (residual plots, added variable plots) might indicate poor fit of the basic model above.
- Remedial measures might include transforming the response, transforming one or both predictors, or both.
- One also might consider adding quadratic terms and/or an interaction term.

When considering a transformation of one predictor, an added variable plot can suggest a transformation (e.g. $\log(x), 1/x$) that might work *if the other predictor is "correctly" specified*.

In general, a transformation is given by a function $x^* = g(x)$. Say we decide that x_{i1} should be log-transformed and the reciprocal of x_{i2} should be used. Then the resulting model is

$$\begin{aligned} Y_i &= \beta_0 + \beta_1 \log(x_{i1}) + \beta_2/x_{i2} + \epsilon_i \\ &= \beta_0 + g_{\beta_1}(x_{i1}) + g_{\beta_2}(x_{i2}) + \epsilon_i, \end{aligned}$$

where $g_{\beta_1}(x)$ and $g_{\beta_2}(x)$ are two functions specified by β_1 and β_2.

Here we are specifying forms for $g_1(x|\beta_1)$ and $g_2(x|\beta_2)$ based on exploratory data analysis, but we could from the outset specify *models* for $g_1(x|\boldsymbol{\theta}_1)$ and $g_2(x|\boldsymbol{\theta}_2)$ that are rich enough to capture interesting and predictively useful aspects of how the predictors affect the response and *estimate these functions from the data*.

One example of this is through an basis expansion; for the jth predictor the transformation is:

$$g_j(x) = \sum_{k=1}^{K_j} \theta_{jk} \psi_{jk}(x),$$

where $\{\psi_{jk}(\cdot)\}_{k=1}^{K_j}$ are sine/cosine functions, spline basis functions, etc. This approach is not the approach taken in SAS PROC GAM. PROC GAM makes use of *cubic smoothing splines*.

This is an example of "nonparametric regression," which ironically connotes the inclusion of *lots* of parameters rather than fewer.

For simple regression data $\{(x_i, y_i)\}_{i=1}^n$, a cubic spline smoother $g(x)$ minimizes

$$\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int_{-\infty}^{\infty} g''(x)^2 dx.$$

Good fit is achieved by minimizing the sum of squares $\sum_{i=1}^{n}(y_i - g(x_i))^2$. The $\int_{-\infty}^{\infty} g''(x)^2 dx$ term measures how wiggly $g(x)$ is and $\lambda \geq 0$ is how much we will penalize $g(x)$ for being wiggly.

So the spline trades off between goodness of fit and wiggliness.

Although not obvious, the solution to this minimization is a cubic spline: a piecewise cubic polynomial with the pieces joined at the unique x_i values.

Cubic spline basis

Cubic splines are piecewise cubic polynomial functions with *knots* $\{\tau_k\}$, where the knots are typically defined as unique values of X. Assume each of the n observed values of X are unique in the following discussion.

- The basis consists of truncated cubic polynomials “centered” at each of the knots, augmented by lower order polynomials.

$$\phi_1(t) = 1,\ \phi_2(t) = t,\ \phi_3(t) = t^2,\ \phi_4(t) = t^3,$$
$$\phi_{k+4}(t) = (t - \tau_k)_+^3,\ k = 1, \ldots, n$$

Cubic spline basis

- The piecewise cubic polynomial components are continuous and have continuous first and second derivatives.
- The cubic polynomial “joins” are smooth to the eye.
- *Natural* cubic spline bases are linear beyond the knots; PROC GAM estimates a natural cubic spline basis.
- The truncated power basis can result in a nearly singular X matrix; B-spline bases are more numerically stable.

Hastie and Tibshirani (1986, 1990) point out that the meaning of λ depends on the units x_i is measured in, but that λ can be picked to yield an "effective degrees of freedom" df or an "effective number of parameters" being used in $g(x)$. Then the complexity of $g(x)$ is equivalent to $(df - 1)$-degree polynomial, but with the coefficients "spread out" more yielding a more flexible function that fits data better.

Alternatively, λ can be picked through cross validation, by minimizing

$$CV(\lambda) = \sum_{i=1}^{n} (y_i - g_{\lambda}^{-i}(x_i))^2.$$

Both options are available in SAS.

We have $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $y_1, \dots, y_n$ are normal.

Each of $g_1(x), \dots, g_p(x)$ are modeled via cubic smoothing splines, each with their own smoothness parameters $\lambda_1, \dots, \lambda_p$ either specified as $df_1, \dots, df_p$ or estimated through cross-validation. The model is fit through *back-fitting* and *local scoring*. See Hastie and Tibshirani (1990) or the SAS documentation for details.

SAS actually fits $g_j(x_j) = \beta_j x_j + \tilde{g}_j(x_j)$, where $\tilde{g}_j(x_j)$ integrates to zero over the range of x_j. Thus one can test $H_0 : \tilde{g}_j(\cdot) = 0$, i.e. the usual linear predictor is sufficient for x_j.

Salary data

Let's fit a GAM to the salary data:

```
data salary;
input salary age educ pol$ @@;
datalines;
38 25 4 D 45 27 4 R 28 26 4 O 55 39 4 D 74 42 4 R 43 41 4 O
47 25 6 D 55 26 6 R 40 29 6 O 65 40 6 D 89 41 6 R 56 42 6 O
56 32 8 D 65 33 8 R 45 35 9 O 75 39 8 D 95 65 9 R 67 69 10 O
;

proc gam plots=all data=salary; *plots(unpack)=components(clm);
 class pol;
 model salary=param(pol) spline(age) spline(educ);
run;
```

I'll write the model on the board.

The Analysis of Deviance table gives a χ^2-test from comparing the deviance between the full model and the model with $\tilde{g}_j(x_j)$ dropped. We see that neither age nor education is nonlinear at the 5% level. The default $df = 3$ corresponds to a smoothing spline with the complexity of a cubic polynomial.

The plots of $\tilde{g}_j(x_j)$ are of the smoothing spline function with the linear effect removed. The plot includes a 95% confidence band for the whole curve. We visually inspect where this band does not include zero to get an idea of where significant nonlinearity occurs. This plot can suggest simpler transformations of predictor variables than use of the full-blown smoothing spline much like residual or added variable plots.

PROC GAM handles Poisson, Bernoulli, and gamma data as well as normal (more in STAT 705). If you only have normal data, PROC TRANSREG will fit a very general transformation model, for example

$$h(Y_i) = \beta_0 + g_1(x_{i1}) + g_2(x_{i2}) + \epsilon_i,$$

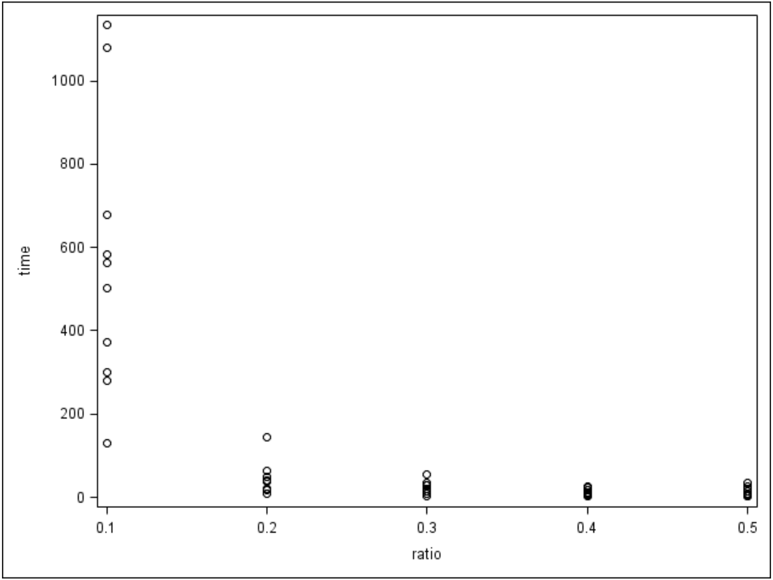
and provide estimates of $h(\cdot)$, $g_1(\cdot)$, and $g_2(\cdot)$.

$h(\cdot)$ can simply be the Box-Cox family, indexed by λ, or a very general spline function.

- Consider time-to-failure in minutes of $n = 50$ electrical components.
- Each component was manufactured using a ratio of two types of materials; this ratio was fixed at 0.1, 0.2, 0.3, 0.4, and 0.5.
- Ten components were observed to fail at each of these manufacturing ratios in a designed experiment.
- It is of interest to model the failure-time as a function of the ratio, to determine if a significant relationship exists, and if so to describe the relationship simply.

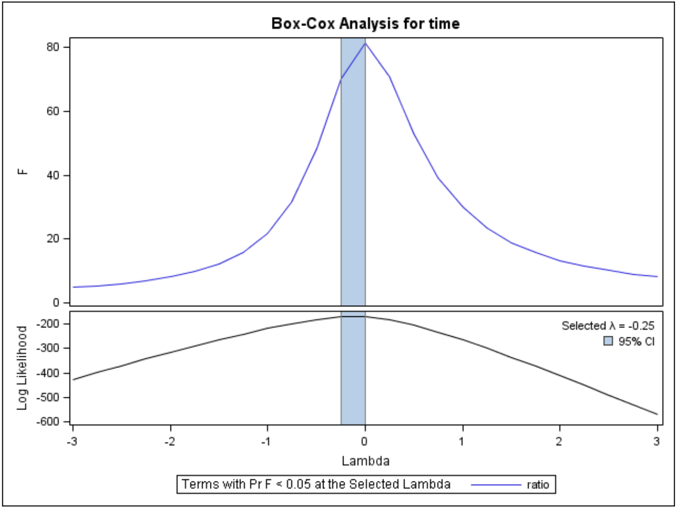
SAS code: data & plot

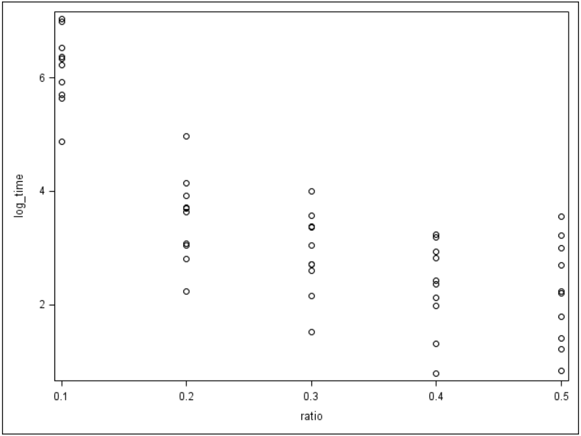
```
data elec;
input ratio time @@;
datalines;
0.5     34.9 0.5      9.3 0.5      6.0 0.5      3.4 0.5     14.9
0.5      9.0 0.5     19.9 0.5      2.3 0.5      4.1 0.5     25.0
0.4     16.9 0.4     11.3 0.4     25.4 0.4     10.7 0.4     24.1
0.4      3.7 0.4      7.2 0.4     18.9 0.4      2.2 0.4      8.4
0.3     54.7 0.3     13.4 0.3     29.3 0.3     28.9 0.3     21.1
0.3     35.5 0.3     15.0 0.3      4.6 0.3     15.1 0.3      8.7
0.2      9.3 0.2     37.6 0.2     21.0 0.2    143.5 0.2     21.8
0.2     50.5 0.2     40.4 0.2     63.1 0.2     41.1 0.2     16.5
0.1    373.0 0.1    584.0 0.1   1080.1 0.1    300.8 0.1    130.8
0.1    280.2 0.1    679.2 0.1    501.6 0.1   1134.3 0.1    562.6
;
proc sgscatter; plot time*ratio; run;
```

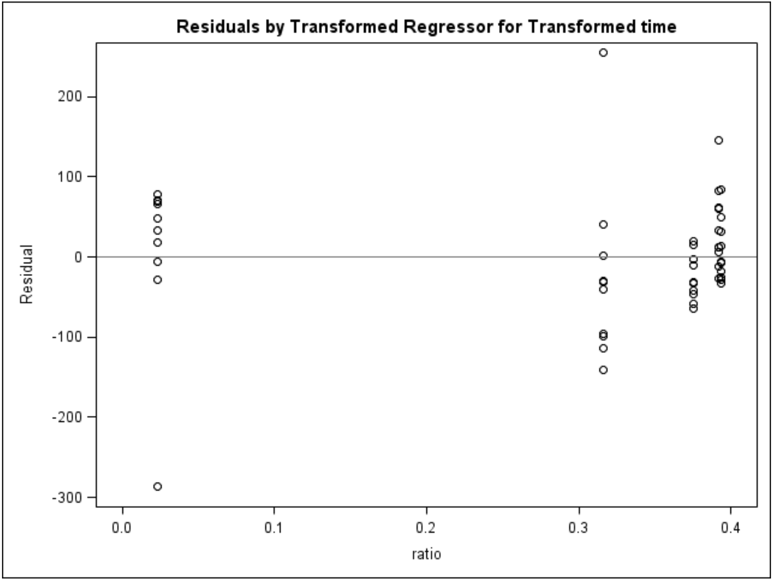

time
1000
800
600
400
200
0
0.1
0.2
0.3
0.4
0.5
ratio

SAS code: fit $h(Y_i) = \beta_0 + g_1(x_{i1}) + \epsilon_i$

```
proc transreg data=elec solve ss2 plots=(transformation obp residuals);
 model spline(time) = spline(ratio); run;
```

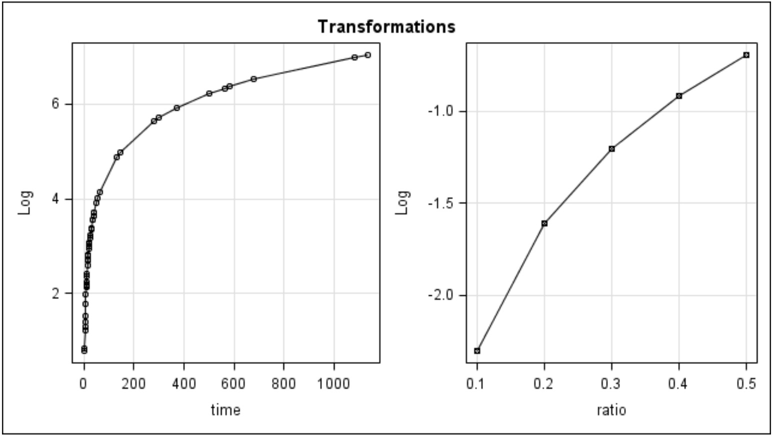


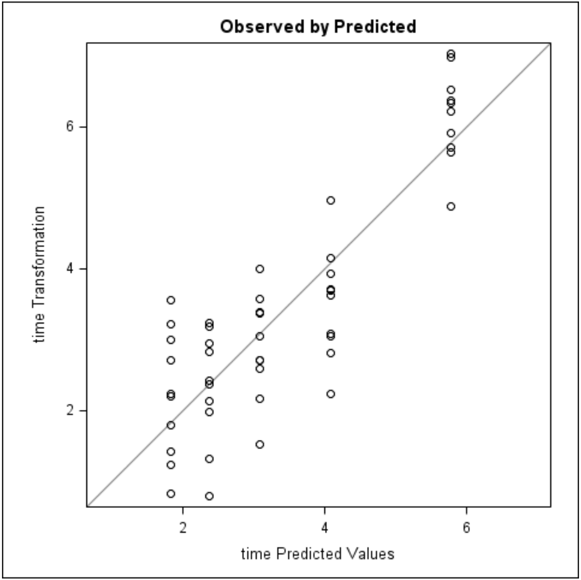

log_time
2
4
6
0.1
0.2
0.3
0.4
0.5
ratio

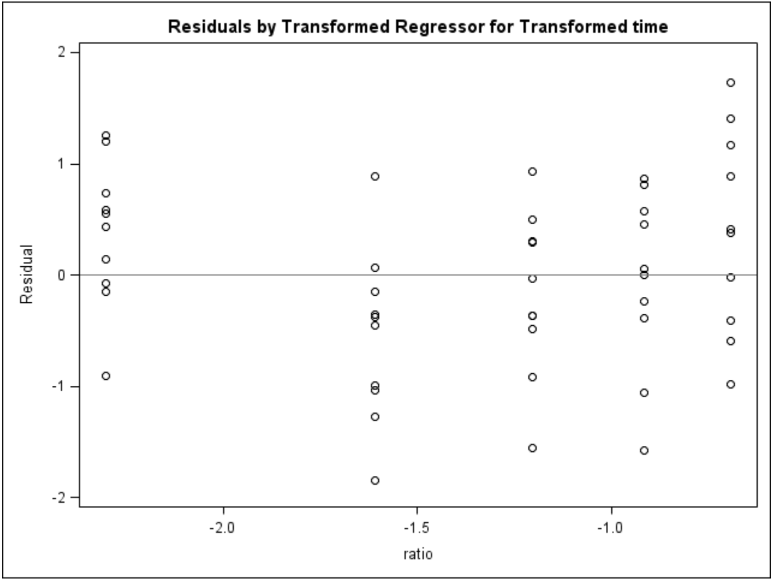

Residuals by Transformed Regressor for Transformed time
Residual
200
100
0
-100
-200
-300
0.0
0.1
0.2
0.3
0.4
ratio

What to do?

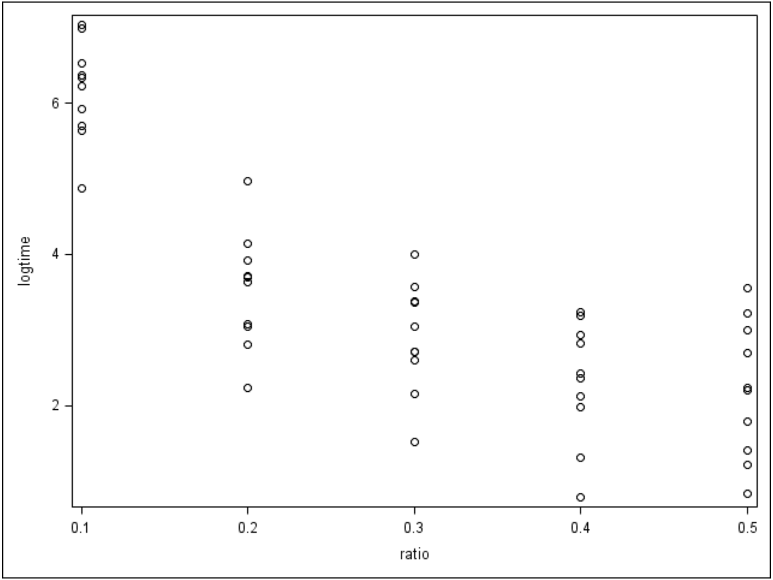
- The “best” fitted transformations look like log or square roots for both time and ratio.
- The log is also suggested by Box-Cox for time (not shown). Code: `model boxcox(time) = spline(ratio)`
- Refit the model with these simple functions:
- `model log(time) = log(ratio)`


Transformations
Log
6
4
2
0
200
400
600
800
1000
time
Log
-1.0
-1.5
-2.0
0.1
0.2
0.3
0.4
0.5
ratio


Observed by Predicted
time Transformation
time Predicted Values
2
4
6
2
4
6


Residuals by Transformed Regressor for Transformed time
Residual
2
1
0
-1
-2
-2.0
-1.5
-1.0
ratio

- Better, but not perfect.
- What if we transform Y_i first, then look at a simple scatterplot of the data?
- Here is plot of log(time) versus ratio...what transformation would you suggest for ratio?


logtime
6
4
2
0.1
0.2
0.3
0.4
0.5
ratio