QCB 508 – Week 10

John D. Storey

Spring 2017

Contents

OLS Goodness of Fit

Pythagorean Theorem

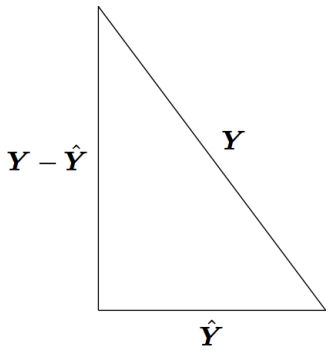


Figure 1: PythMod

Least squares model fitting can be understood through the Pythagorean theorem: $a^2 + b^2 = c^2$. However, here we have:

$$\sum_{i=1}^{n} Y_i^2 = \sum_{i=1}^{n} \hat{Y}_i^2 + \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$

where the $\hat{Y}_i$ are the result of a **linear projection** of the Y_i.

OLS Normal Model

In this section, let's assume that $(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)$ are distributed so that

$$\begin{aligned} Y_i &= \beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip} + E_i \\ &= \boldsymbol{X}_i \boldsymbol{\beta} + E_i \end{aligned}$$

where $\boldsymbol{E} | \boldsymbol{X} \sim \text{MVN}_n(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$. Note that we haven't specified the distribution of the $\boldsymbol{X}_i$ rv's.

Projection Matrices

In the OLS framework we have:

$$\hat{\boldsymbol{Y}} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}.$$

The matrix $\boldsymbol{P}_{n\times n} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$ is a projection matrix. The vector $\boldsymbol{Y}$ is projected into the space spanned by the column space of $\boldsymbol{X}$.

Project matrices have the following properties:

- $\boldsymbol{P}$ is symmetric
- $\boldsymbol{P}$ is idempotent so that $\boldsymbol{PP} = \boldsymbol{P}$
- If $\boldsymbol{X}$ has column rank p, then $\boldsymbol{P}$ has rank p
- The eigenvalues of $\boldsymbol{P}$ are p 1's and $n-p$ 0's
- The trace (sum of diagonal entries) is $\text{tr}(\boldsymbol{P}) = p$
- $\boldsymbol{I} - \boldsymbol{P}$ is also a projection matrix with rank $n-p$

Decomposition

Note that $\boldsymbol{P}(\boldsymbol{I}-\boldsymbol{P}) = \boldsymbol{P} - \boldsymbol{PP} = \boldsymbol{P} - \boldsymbol{P} = \boldsymbol{0}$.

We have

$$\begin{aligned}\|\boldsymbol{Y}\|_2^2 = \boldsymbol{Y}^T\boldsymbol{Y} &= (\boldsymbol{PY} + (\boldsymbol{I}-\boldsymbol{P})\boldsymbol{Y})^T(\boldsymbol{PY} + (\boldsymbol{I}-\boldsymbol{P})\boldsymbol{Y}) \\ &= (\boldsymbol{PY})^T(\boldsymbol{PY}) + ((\boldsymbol{I}-\boldsymbol{P})\boldsymbol{Y})^T((\boldsymbol{I}-\boldsymbol{P})\boldsymbol{Y}) \\ &= \|\boldsymbol{PY}\|_2^2 + \|(\boldsymbol{I}-\boldsymbol{P})\boldsymbol{Y}\|_2^2\end{aligned}$$

where the cross terms disappear because $\boldsymbol{P}(\boldsymbol{I}-\boldsymbol{P}) = \boldsymbol{0}$.

Note: The ℓ_p norm of an n-vector $\boldsymbol{w}$ is defined as

$$\|\boldsymbol{w}\|_p = \left(\sum_{i=1}^n |w_i|^p\right)^{1/p}.$$

Above we calculated

$$\|\boldsymbol{w}\|_2^2 = \sum_{i=1}^n w_i^2.$$

Distribution of Projection

Suppose that $Y_1, Y_2, \ldots, Y_n \overset{\text{iid}}{\sim} \text{Normal}(0, \sigma^2)$. This can also be written as $\boldsymbol{Y} \sim \text{MVN}_n(\boldsymbol{0}, \sigma^2\boldsymbol{I})$. It follows that

$$\boldsymbol{PY} \sim \text{MVN}_n(\boldsymbol{0}, \sigma^2\boldsymbol{PIP}^T).$$

where $\boldsymbol{PIP}^T = \boldsymbol{PP}^T = \boldsymbol{PP} = \boldsymbol{P}$.

Also, $(\boldsymbol{PY})^T(\boldsymbol{PY}) = \boldsymbol{Y}^T\boldsymbol{P}^T\boldsymbol{PY} = \boldsymbol{Y}^T\boldsymbol{PY}$, a **quadratic form**. Given the eigenvalues of $\boldsymbol{P}$, $\boldsymbol{Y}^T\boldsymbol{PY}$ is equivalent in distribution to p squared iid Normal(0,1) rv's, so

$$\frac{\boldsymbol{Y}^T\boldsymbol{PY}}{\sigma^2} \sim \chi^2_p.$$

Distribution of Residuals

If $\boldsymbol{PY} = \hat{\boldsymbol{Y}}$ are the fitted OLS values, then $(\boldsymbol{I}-\boldsymbol{P})\boldsymbol{Y} = \boldsymbol{Y} - \hat{\boldsymbol{Y}}$ are the residuals.

It follows by the same argument as above that

$$\frac{\boldsymbol{Y}^T(\boldsymbol{I}-\boldsymbol{P})\boldsymbol{Y}}{\sigma^2} \sim \chi^2_{n-p}.$$

It's also straightforward to show that $(\boldsymbol{I}-\boldsymbol{P})\boldsymbol{Y} \sim \text{MVN}_n(\boldsymbol{0}, \sigma^2(\boldsymbol{I}-\boldsymbol{P}))$ and $\text{Cov}(\boldsymbol{PY}, (\boldsymbol{I}-\boldsymbol{P})\boldsymbol{Y}) = \boldsymbol{0}$.

Degrees of Freedom

The degrees of freedom, p, of a linear projection model fit is equal to

- The number of linearly dependent columns of $\boldsymbol{X}$
- The number of nonzero eigenvalues of $\boldsymbol{P}$ (where nonzero eigenvalues are equal to 1)
- The trace of the projection matrix, $\text{tr}(\boldsymbol{P})$.

The reason why we divide estimates of variance by $n-p$ is because this is the number of effective independent sources of variation remaining after the model is fit by projecting the n observations into a p dimensional linear space.

Submodels

Consider the OLS model $\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{E}$ where there are p columns of $\boldsymbol{X}$ and $\boldsymbol{\beta}$ is a p-vector.

Let $\boldsymbol{X}_0$ be a subset of p_0 columns of $\boldsymbol{X}$ and let $\boldsymbol{X}_1$ be a subset of p_1 columns, where $1 \leq p_0 < p_1 \leq p$. Also, assume that the columns of $\boldsymbol{X}_0$ are a subset of $\boldsymbol{X}_1$.

We can form $\hat{\boldsymbol{Y}}_0 = \boldsymbol{P}_0\boldsymbol{Y}$ where $\boldsymbol{P}_0$ is the projection matrix built from $\boldsymbol{X}_0$. We can analogously form $\hat{\boldsymbol{Y}}_1 = \boldsymbol{P}_1\boldsymbol{Y}$.

Hypothesis Testing

Without loss of generality, suppose that $\boldsymbol{\beta}_0 = (\beta_1, \beta_2, \ldots, \beta_{p_0})^T$ and $\boldsymbol{\beta}_1 = (\beta_1, \beta_2, \ldots, \beta_{p_1})^T$.

How do we compare these models, specifically to test $H_0 : (\beta_{p_0+1}, \beta_{p_0+2}, \ldots, \beta_{p_1}) = \mathbf{0}$ vs $H_1 : (\beta_{p_0+1}, \beta_{p_0+2}, \ldots, \beta_{p_1}) \neq \mathbf{0}$?

The basic idea to perform this test is to compare the goodness of fits of each model via a pivotal statistic. We will discuss the generalized LRT and ANOVA approaches.

Generalized LRT

Under the OLS Normal model, it follows that $\hat{\boldsymbol{\beta}}_0 = (\boldsymbol{X}_0^T \boldsymbol{X}_0)^{-1} \boldsymbol{X}_0^T \boldsymbol{Y}$ is the MLE under the null hypothesis and $\hat{\boldsymbol{\beta}}_1 = (\boldsymbol{X}_1^T \boldsymbol{X}_1)^{-1} \boldsymbol{X}_1^T \boldsymbol{Y}$ is the unconstrained MLE. Also, the respective MLEs of σ^2 are

$$\hat{\sigma}_0^2 = \frac{\sum_{i=1}^n (Y_i - \hat{Y}_{0,i})^2}{n}$$

$$\hat{\sigma}_1^2 = \frac{\sum_{i=1}^n (Y_i - \hat{Y}_{1,i})^2}{n}$$

where $\hat{\boldsymbol{Y}}_0 = \boldsymbol{X}_0 \hat{\boldsymbol{\beta}}_0$ and $\hat{\boldsymbol{Y}}_1 = \boldsymbol{X}_1 \hat{\boldsymbol{\beta}}_1$.

The generalized LRT statistic is

$$\lambda(\boldsymbol{X}, \boldsymbol{Y}) = \frac{L\left(\hat{\boldsymbol{\beta}}_1, \hat{\sigma}_1^2; \boldsymbol{X}, \boldsymbol{Y}\right)}{L\left(\hat{\boldsymbol{\beta}}_0, \hat{\sigma}_0^2; \boldsymbol{X}, \boldsymbol{Y}\right)}$$

where $2 \log \lambda(\boldsymbol{X}, \boldsymbol{Y})$ has a $\chi^2_{p_1 - p_0}$ null distribution.

Nested Projections

We can apply the Pythagorean theorem we saw earlier to linear subspaces to get:

$$\begin{aligned} \|\boldsymbol{Y}\|_2^2 &= \|(\boldsymbol{I} - \boldsymbol{P}_1)\boldsymbol{Y}\|_2^2 + \|\boldsymbol{P}_1 \boldsymbol{Y}\|_2^2 \\ &= \|(\boldsymbol{I} - \boldsymbol{P}_1)\boldsymbol{Y}\|_2^2 + \|(\boldsymbol{P}_1 - \boldsymbol{P}_0)\boldsymbol{Y}\|_2^2 + \|\boldsymbol{P}_0 \boldsymbol{Y}\|_2^2 \end{aligned}$$

We can also use the Pythagorean theorem to decompose the residuals from the smaller projection $\boldsymbol{P}_0$:

$$\|(\boldsymbol{I}-\boldsymbol{P}_0)\boldsymbol{Y}\|_2^2 = \|(\boldsymbol{I}-\boldsymbol{P}_1)\boldsymbol{Y}\|_2^2 + \|(\boldsymbol{P}_1-\boldsymbol{P}_0)\boldsymbol{Y}\|_2^2$$

$\boldsymbol{F}$ Statistic

The F statistic compares the improvement of goodness in fit of the larger model to that of the smaller model in terms of sums of squared residuals, and it scales this improvement by an estimate of σ^2:

$$\begin{aligned} F &= \frac{\left[\|(\boldsymbol{I}-\boldsymbol{P}_0)\boldsymbol{Y}\|_2^2 - \|(\boldsymbol{I}-\boldsymbol{P}_1)\boldsymbol{Y}\|_2^2\right]/(p_1-p_0)}{\|(\boldsymbol{I}-\boldsymbol{P}_1)\boldsymbol{Y}\|_2^2/(n-p_1)} \\ &= \frac{\left[\sum_{i=1}^n (Y_i-\hat{Y}_{0,i})^2 - \sum_{i=1}^n (Y_i-\hat{Y}_{1,i})^2\right]/(p_1-p_0)}{\sum_{i=1}^n (Y_i-\hat{Y}_{1,i})^2/(n-p_1)} \end{aligned}$$

Since $\|(\boldsymbol{I}-\boldsymbol{P}_0)\boldsymbol{Y}\|_2^2 - \|(\boldsymbol{I}-\boldsymbol{P}_1)\boldsymbol{Y}\|_2^2 = \|(\boldsymbol{P}_1-\boldsymbol{P}_0)\boldsymbol{Y}\|_2^2$, we can equivalently write the F statistic as:

$$\begin{aligned} F &= \frac{\|(\boldsymbol{P}_1-\boldsymbol{P}_0)\boldsymbol{Y}\|_2^2/(p_1-p_0)}{\|(\boldsymbol{I}-\boldsymbol{P}_1)\boldsymbol{Y}\|_2^2/(n-p_1)} \\ &= \frac{\sum_{i=1}^n (\hat{Y}_{1,i}-\hat{Y}_{0,i})^2/(p_1-p_0)}{\sum_{i=1}^n (Y_i-\hat{Y}_{1,i})^2/(n-p_1)} \end{aligned}$$

$\boldsymbol{F}$ Distribution

Suppose we have independent random variables $V \sim \chi_a^2$ and $W \sim \chi_b^2$. It follows that

$$\frac{V/a}{W/b} \sim F_{a,b}$$

where $F_{a,b}$ is the F distribution with (a, b) degrees of freedom.

By arguments similar to those given above, we have

$$\frac{\|(\boldsymbol{P}_1-\boldsymbol{P}_0)\boldsymbol{Y}\|_2^2}{\sigma^2} \sim \chi^2_{p_1-p_0}$$

$$\frac{\|(\boldsymbol{I}-\boldsymbol{P}_1)\boldsymbol{Y}\|_2^2}{\sigma^2} \sim \chi^2_{n-p_1}$$

and these two rv's are independent.

F Test

Suppose that the OLS model holds where $\boldsymbol{E}|\boldsymbol{X} \sim \text{MVN}_n(\boldsymbol{0}, \sigma^2\boldsymbol{I})$.

In order to test $H_0 : (\beta_{p_0+1}, \beta_{p_0+2}, \ldots, \beta_{p_1}) = \boldsymbol{0}$ vs $H_1 : (\beta_{p_0+1}, \beta_{p_0+2}, \ldots, \beta_{p_1}) \neq \boldsymbol{0}$, we can form the F statistic as given above, which has null distribution $F_{p_1-p_0,n-p_1}$. The p-value is calculated as $\Pr(F^* \geq F)$ where F is the observed F statistic and $F^* \sim F_{p_1-p_0,n-p_1}$.

If the above assumption on the distribution of $\boldsymbol{E}|\boldsymbol{X}$ only approximately holds, then the F test p-value is also an approximation.

Example: Davis Data

```
> library("car")
> data("Davis", package="car")
```

```
> htwt <- tbl_df(Davis)
> htwt[12,c(2,3)] <- htwt[12,c(3,2)]
> head(htwt)
# A tibble: 6 × 5
     sex weight height repwt repht
  <fctr>  <int>  <int> <int> <int>
1      M     77    182    77   180
2      F     58    161    51   159
3      F     53    161    54   158
4      M     68    177    70   175
5      F     59    157    59   155
6      M     76    170    76   165
```

Comparing Linear Models in R

Example: Davis Data

Suppose we are considering the three following models:

```
> f1 <- lm(weight ~ height, data=htwt)
> f2 <- lm(weight ~ height + sex, data=htwt)
> f3 <- lm(weight ~ height + sex + height:sex, data=htwt)
```

How do we determine if the additional terms in models `f2` and `f3` are needed?

ANOVA (Version 2)

A generalization of ANOVA exists that allows us to compare two nested models, quantifying their differences in terms of goodness of fit and performing a hypothesis test of whether this difference is statistically significant.

A model is *nested* within another model if their difference is simply the absence of certain terms in the smaller model.

The null hypothesis is that the additional terms have coefficients equal to zero, and the alternative hypothesis is that at least one coefficient is nonzero.

Both versions of ANOVA can be described in a single, elegant mathematical framework.

Comparing Two Models with `anova()`

This provides a comparison of the improvement in fit from model `f2` compared to model `f1`:

```
> anova(f1, f2)
Analysis of Variance Table

Model 1: weight ~ height
Model 2: weight ~ height + sex
  Res.Df   RSS Df Sum of Sq      F    Pr(>F)
1    198 14321
2    197 12816  1    1504.9 23.133 2.999e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

When There's a Single Variable Difference

Compare above `anova(f1, f2)` p-value to that for the `sex` term from the `f2` model:

```
> library(broom)
> tidy(f2)
         term    estimate   std.error statistic      p.value
1 (Intercept) -76.6167326 15.71504644 -4.875374 2.231334e-06
2      height   0.8105526  0.09529565  8.505662 4.499241e-15
3        sexM   8.2268893  1.71050385  4.809629 2.998988e-06
```

Calculating the F-statistic

```
> anova(f1, f2)
Analysis of Variance Table

Model 1: weight ~ height
Model 2: weight ~ height + sex
  Res.Df   RSS Df Sum of Sq      F    Pr(>F)
1    198 14321
2    197 12816  1    1504.9 23.133 2.999e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

How the F-statistic is calculated:

```
> n <- nrow(htwt)
> ss1 <- (n-1)*var(f1$residuals)
> ss1
[1] 14321.11
> ss2 <- (n-1)*var(f2$residuals)
> ss2
[1] 12816.18
> ((ss1 - ss2)/anova(f1, f2)$Df[2])/(ss2/f2$df.residual)
[1] 23.13253
```

Calculating the Generalized LRT

```
> anova(f1, f2, test="LRT")
Analysis of Variance Table

Model 1: weight ~ height
Model 2: weight ~ height + sex
  Res.Df   RSS Df Sum of Sq  Pr(>Chi)
1    198 14321
2    197 12816  1    1504.9 1.512e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
> library(lmtest)
> lrtest(f1, f2)
Likelihood ratio test

Model 1: weight ~ height
Model 2: weight ~ height + sex
  #Df LogLik Df  Chisq Pr(>Chisq)
```

```
1   3 -710.9
2   4 -699.8  1 22.205   2.45e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

These tests produce slightly different answers because `anova()` adjusts for degrees of freedom when estimating the variance, whereas `lrtest()` is the strict generalized LRT. See here.

ANOVA on More Distant Models

We can compare models with multiple differences in terms:

```
> anova(f1, f3)
Analysis of Variance Table

Model 1: weight ~ height
Model 2: weight ~ height + sex + height:sex
  Res.Df   RSS Df Sum of Sq      F    Pr(>F)
1    198 14321
2    196 12567  2      1754 13.678 2.751e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Compare Multiple Models at Once

We can compare multiple models at once:

```
> anova(f1, f2, f3)
Analysis of Variance Table

Model 1: weight ~ height
Model 2: weight ~ height + sex
Model 3: weight ~ height + sex + height:sex
  Res.Df   RSS Df Sum of Sq       F    Pr(>F)
1    198 14321
2    197 12816  1   1504.93 23.4712 2.571e-06 ***
3    196 12567  1    249.04  3.8841   0.05015 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Generalized Linear Models

Definition

The generalized linear model (GLM) builds from OLS and GLS to allow for the case where $Y|\boldsymbol{X}$ is distributed according to an exponential family distribution. The estimated model is

$$g\left(\mathrm{E}[Y|\boldsymbol{X}]\right) = \boldsymbol{X}\boldsymbol{\beta}$$

where $g(\cdot)$ is called the **link function**. This model is typically fit by numerical methods to calculate the maximum likelihood estimate of $\boldsymbol{\beta}$.

Exponential Family Distributions

Recall that if Y follows an EFD then it has pdf of the form

$$f(y;\boldsymbol{\theta}) = h(y)\exp\left\{\sum_{k=1}^{d}\eta_k(\boldsymbol{\theta})T_k(y) - A(\boldsymbol{\eta})\right\}$$

where $\boldsymbol{\theta}$ is a vector of parameters, $\{T_k(y)\}$ are sufficient statistics, $A(\boldsymbol{\eta})$ is the cumulant generating function.

The functions $\eta_k(\boldsymbol{\theta})$ for $k = 1, \ldots, d$ map the usual parameters $\boldsymbol{\theta}$ (often moments of the rv Y) to the *natural parameters* or *canonical parameters*.

$\{T_k(y)\}$ are sufficient statistics for $\{\eta_k\}$ due to the factorization theorem.

$A(\boldsymbol{\eta})$ is sometimes called the *log normalizer* because

$$A(\boldsymbol{\eta}) = \log\int h(y)\exp\left\{\sum_{k=1}^{d}\eta_k(\boldsymbol{\theta})T_k(y)\right\}.$$

Natural Single Parameter EFD

A natural single parameter EFD simplifies to the scenario where $d = 1$ and $T(y) = y$

$$f(y;\eta) = h(y)\exp\left\{\eta(\theta)y - A(\eta(\theta))\right\}$$

where without loss of generality we can write $\mathrm{E}[Y] = \theta$.

Dispersion EFDs

The family of distributions for which GLMs are most typically developed are dispersion EFDs. An example of a dispersion EFD that extends the natural single parameter EFD is

$$f(y;\eta) = h(y,\phi)\exp\left\{\frac{\eta(\theta)y - A(\eta(\theta))}{\phi}\right\}$$

where ϕ is the dispersion parameter.

Example: Normal

Let $Y \sim \text{Normal}(\mu, \sigma^2)$. Then:

$$\theta = \mu, \eta(\mu) = \mu$$

$$\phi = \sigma^2$$

$$A(\mu) = \frac{\mu^2}{2}$$

$$h(y,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{y^2}{\sigma^2}}$$

EFD for GLMs

There has been a very broad development of GLMs and extensions. A common setting for introducting GLMs is the dispersion EFD with a general link function $g(\cdot)$.

See the classic text *Generalized Linear Models*, by McCullagh and Nelder, for such a development.

Components of a GLM

1. *Random*: The particular exponential family distribution.

$$Y \sim f(y;\eta,\phi)$$

2. *Systematic*: The determination of each η_i from $\boldsymbol{X}_i$ and $\boldsymbol{\beta}$.

$$\eta_i = \boldsymbol{X}_i\boldsymbol{\beta}$$

3. *Parametric Link*: The connection between $E[Y_i|\boldsymbol{X}_i]$ and $\boldsymbol{X}_i\boldsymbol{\beta}$.

$$g(E[Y_i|\boldsymbol{X}_i]) = \boldsymbol{X}_i\boldsymbol{\beta}$$

Link Functions

Even though the link function $g(\cdot)$ can be considered in a fairly general framework, the **canonical link function** $\eta(\cdot)$ is often utilized.

The canonical link function is the function that maps the expected value into the natural paramater.

In this case, $Y|\boldsymbol{X}$ is distributed according to an exponential family distribution with

$$\eta\left(\mathrm{E}[Y|\boldsymbol{X}]\right) = \boldsymbol{X}\boldsymbol{\beta}.$$

Calculating MLEs

Given the model $g\left(\mathrm{E}[Y|\boldsymbol{X}]\right) = \boldsymbol{X}\boldsymbol{\beta}$, the EFD should be fully parameterized. The Newton-Raphson method or Fisher's scoring method can be utilized to find the MLE of $\boldsymbol{\beta}$.

Newton-Raphson

1. Initialize $\boldsymbol{\beta}^{(0)}$. For $t = 1, 2, \ldots$
2. Calculate the score $s(\boldsymbol{\beta}^{(t)}) = \nabla\ell(\boldsymbol{\beta}; \boldsymbol{X}, \boldsymbol{Y}) \mid_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(t)}}$ and observed Fisher information
$$H(\boldsymbol{\beta}^{(t)}) = -\nabla\nabla^T\ell(\boldsymbol{\beta}; \boldsymbol{X}, \boldsymbol{Y}) \mid_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(t)}}$$
. Note that the observed Fisher information is also the negative Hessian matrix.
3. Update $\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} + H(\boldsymbol{\beta}^{(t)})^{-1}s(\boldsymbol{\beta}^{(t)})$.
4. Iterate until convergence, and set $\hat{\boldsymbol{\beta}} = \boldsymbol{\beta}^{(\infty)}$.

Fisher's scoring

1. Initialize $\boldsymbol{\beta}^{(0)}$. For $t = 1, 2, \ldots$
2. Calculate the score $s(\boldsymbol{\beta}^{(t)}) = \nabla \ell(\boldsymbol{\beta}; \boldsymbol{X}, \boldsymbol{Y}) \mid_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(t)}}$ and expected Fisher information
$$I(\boldsymbol{\beta}^{(t)}) = -\operatorname{E}\left[\nabla\nabla^T \ell(\boldsymbol{\beta}; \boldsymbol{X}, \boldsymbol{Y}) \mid_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(t)}}\right].$$
3. Update $\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} + I(\boldsymbol{\beta}^{(t)})^{-1} s(\boldsymbol{\beta}^{(t)})$.
4. Iterate until convergence, and set $\hat{\boldsymbol{\beta}} = \boldsymbol{\beta}^{(\infty)}$.

When the canonical link function is used, the Newton-Raphson algorithm and Fisher's scoring algorithm are equivalent.

Exercise: Prove this.

Iteratively Reweighted Least Squares

For the canonical link, Fisher's scoring method can be written as an iteratively reweighted least squares algorithm, as shown earlier for logistic regression. Note that the Fisher information is

$$I(\boldsymbol{\beta}^{(t)}) = \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X}$$

where $\boldsymbol{W}$ is an $n \times n$ diagonal matrix with (i, i) entry equal to $\operatorname{Var}(Y_i | \boldsymbol{X}; \boldsymbol{\beta}^{(t)})$.

The score function is

$$s(\boldsymbol{\beta}^{(t)}) = \boldsymbol{X}^T \left(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}^{(t)}\right)$$

and the current coefficient value $\boldsymbol{\beta}^{(t)}$ can be written as

$$\boldsymbol{\beta}^{(t)} = (\boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X} \boldsymbol{\beta}^{(t)}.$$

Putting this together we get

$$\boldsymbol{\beta}^{(t)} + I(\boldsymbol{\beta}^{(t)})^{-1} s(\boldsymbol{\beta}^{(t)}) = (\boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{z}^{(t)}$$

where

$$\boldsymbol{z}^{(t)} = \boldsymbol{X}\boldsymbol{\beta}^{(t)} + \boldsymbol{W}^{-1} \left(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}^{(t)}\right).$$

This is a generalization of the iteratively reweighted least squares algorithm we showed earlier for logistic regression.

Estimating Dispersion

For the simple dispersion model above, it is typically straightforward to calculate the MLE $\hat{\phi}$ once $\hat{\boldsymbol{\beta}}$ has been calculated.

CLT Applied to the MLE

Given that $\hat{\boldsymbol{\beta}}$ is a maximum likelihood estimate, we have the following CLT result on its distribution as $n \to \infty$:

$$\sqrt{n}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) \xrightarrow{D} \text{MVN}_p(\mathbf{0}, \phi(\boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X})^{-1})$$

Approximately Pivotal Statistics

The previous CLT gives us the following two approximations for pivtoal statistics. The first statistic facilitates getting overall measures of uncertainty on the estimate $\hat{\boldsymbol{\beta}}$.

$$\hat{\phi}^{-1}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})^T (\boldsymbol{X}^T \hat{\boldsymbol{W}} \boldsymbol{X})(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) \;\dot{\sim}\; \chi^2_1$$

This second pivotal statistic allows for performing a Wald test or forming a confidence interval on each coefficient, β_j, for $j = 1, \ldots, p$.

$$\frac{\hat{\beta}_j - \beta_j}{\sqrt{\hat{\phi}[(\boldsymbol{X}^T \hat{\boldsymbol{W}} \boldsymbol{X})^{-1}]_{jj}}} \;\dot{\sim}\; \text{Normal}(0, 1)$$

Deviance

Let $\hat{\boldsymbol{\eta}}$ be the estimated natural parameters from a GLM. For example, $\hat{\boldsymbol{\eta}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}$ when the canonical link function is used.

Let $\hat{\boldsymbol{\eta}}_n$ be the **saturated model** wwhere Y_i is directly used to estimate η_i without model constraints. For example, in the Bernoulli logistic regression model $\hat{\boldsymbol{\eta}}_n = \boldsymbol{Y}$, the observed outcomes.

The **deviance** for the model is defined to be

$$D\left(\hat{\boldsymbol{\eta}}\right) = 2\ell(\hat{\boldsymbol{\eta}}_n; \boldsymbol{X}, \boldsymbol{Y}) - 2\ell(\hat{\boldsymbol{\eta}}; \boldsymbol{X}, \boldsymbol{Y})$$

Generalized LRT

Let $\boldsymbol{X}_0$ be a subset of p_0 columns of $\boldsymbol{X}$ and let $\boldsymbol{X}_1$ be a subset of p_1 columns, where $1 \leq p_0 < p_1 \leq p$. Also, assume that the columns of $\boldsymbol{X}_0$ are a subset of $\boldsymbol{X}_1$.

Without loss of generality, suppose that $\boldsymbol{\beta}_0 = (\beta_1, \beta_2, \ldots, \beta_{p_0})^T$ and $\boldsymbol{\beta}_1 = (\beta_1, \beta_2, \ldots, \beta_{p_1})^T$.

Suppose we wish to test $H_0 : (\beta_{p_0+1}, \beta_{p_0+2}, \ldots, \beta_{p_1}) = \mathbf{0}$ vs $H_1 : (\beta_{p_0+1}, \beta_{p_0+2}, \ldots, \beta_{p_1}) \neq \mathbf{0}$

We can form $\hat{\boldsymbol{\eta}}_0 = \boldsymbol{X}\hat{\boldsymbol{\beta}}_0$ from the GLM model $g\left(\mathrm{E}[Y|\boldsymbol{X}_0]\right) = \boldsymbol{X}_0\boldsymbol{\beta}_0$. We can analogously form $\hat{\boldsymbol{\eta}}_1 = \boldsymbol{X}\hat{\boldsymbol{\beta}}_1$ from the GLM model $g\left(\mathrm{E}[Y|\boldsymbol{X}_1]\right) = \boldsymbol{X}_1\boldsymbol{\beta}_1$.

The 2 log generalized LRT can then be formed from the two deviance statistics

$$2\log\lambda(\boldsymbol{X}, \boldsymbol{Y}) = 2\log\frac{L(\hat{\boldsymbol{\eta}}_1; \boldsymbol{X}, \boldsymbol{Y})}{L(\hat{\boldsymbol{\eta}}_0; \boldsymbol{X}, \boldsymbol{Y})} = D\left(\hat{\boldsymbol{\eta}}_0\right) - D\left(\hat{\boldsymbol{\eta}}_1\right)$$

where the null distribution is $\chi^2_{p_1-p_0}$.

Example: Grad School Admissions

Let's revisit a logistic regression example now that we know how the various statistics are calculated.

```
> mydata <-
+   read.csv("http://www.ats.ucla.edu/stat/data/binary.csv")
> dim(mydata)
[1] 400   4
> head(mydata)
  admit gre  gpa rank
1     0 380 3.61    3
2     1 660 3.67    3
3     1 800 4.00    1
4     1 640 3.19    4
5     0 520 2.93    4
6     1 760 3.00    2
```

Fit the model with basic output. Note the argument `family = "binomial"`.

```
> mydata$rank <- factor(mydata$rank, levels=c(1, 2, 3, 4))
> myfit <- glm(admit ~ gre + gpa + rank,
+              data = mydata, family = "binomial")
> myfit

Call:  glm(formula = admit ~ gre + gpa + rank, family = "binomial",
```

```
    data = mydata)

Coefficients:
(Intercept)          gre          gpa        rank2
  -3.989979     0.002264     0.804038    -0.675443
      rank3        rank4
  -1.340204    -1.551464

Degrees of Freedom: 399 Total (i.e. Null);  394 Residual
Null Deviance:      500
Residual Deviance: 458.5    AIC: 470.5
```

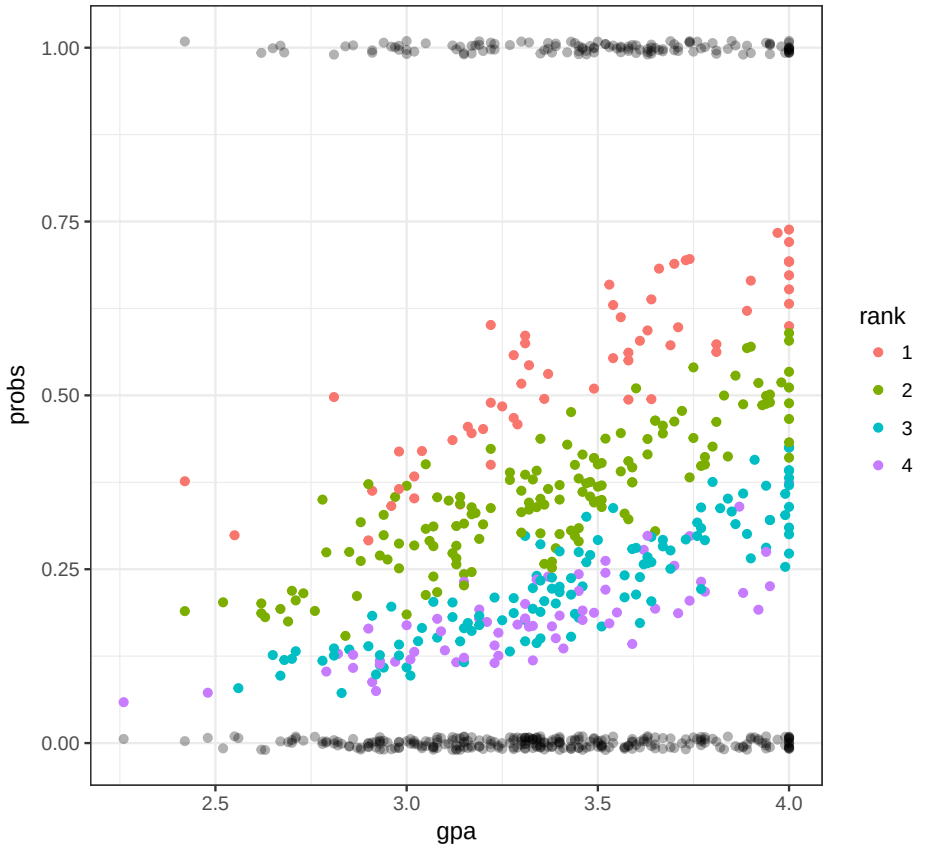
This shows the fitted coefficient values, which is on the link function scale – logit aka log odds here. Also, a Wald test is performed for each coefficient.

```
> summary(myfit)

Call:
glm(formula = admit ~ gre + gpa + rank, family = "binomial",
    data = mydata)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6268  -0.8662  -0.6388   1.1490   2.0790

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.989979   1.139951  -3.500 0.000465 ***
gre          0.002264   0.001094   2.070 0.038465 *
gpa          0.804038   0.331819   2.423 0.015388 *
rank2       -0.675443   0.316490  -2.134 0.032829 *
rank3       -1.340204   0.345306  -3.881 0.000104 ***
rank4       -1.551464   0.417832  -3.713 0.000205 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 458.52  on 394  degrees of freedom
AIC: 470.52

Number of Fisher Scoring iterations: 4
```

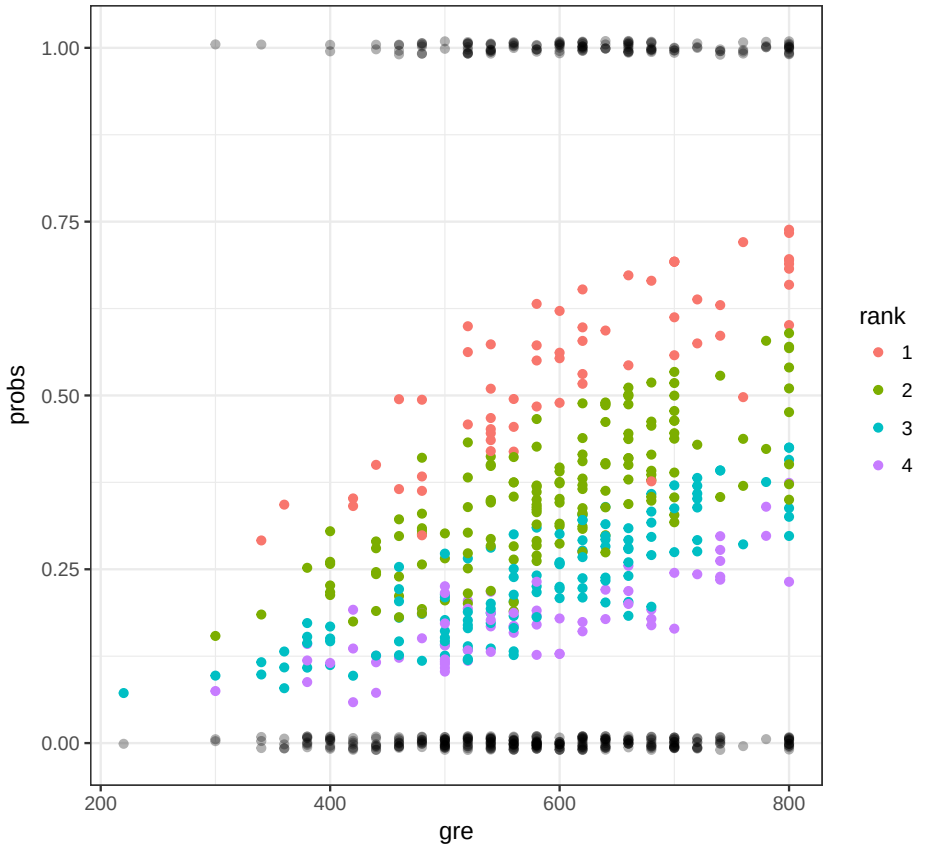
Here we perform a generalized LRT on each variable. Note the `rank` variable is now tested as a single factor variable as opposed to each dummy variable.

```
> anova(myfit, test="LRT")
Analysis of Deviance Table

Model: binomial, link: logit

Response: admit

Terms added sequentially (first to last)

     Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                   399     499.98
gre   1  13.9204       398     486.06 0.0001907 ***
gpa   1   5.7122       397     480.34 0.0168478 *
rank  3  21.8265       394     458.52 7.088e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

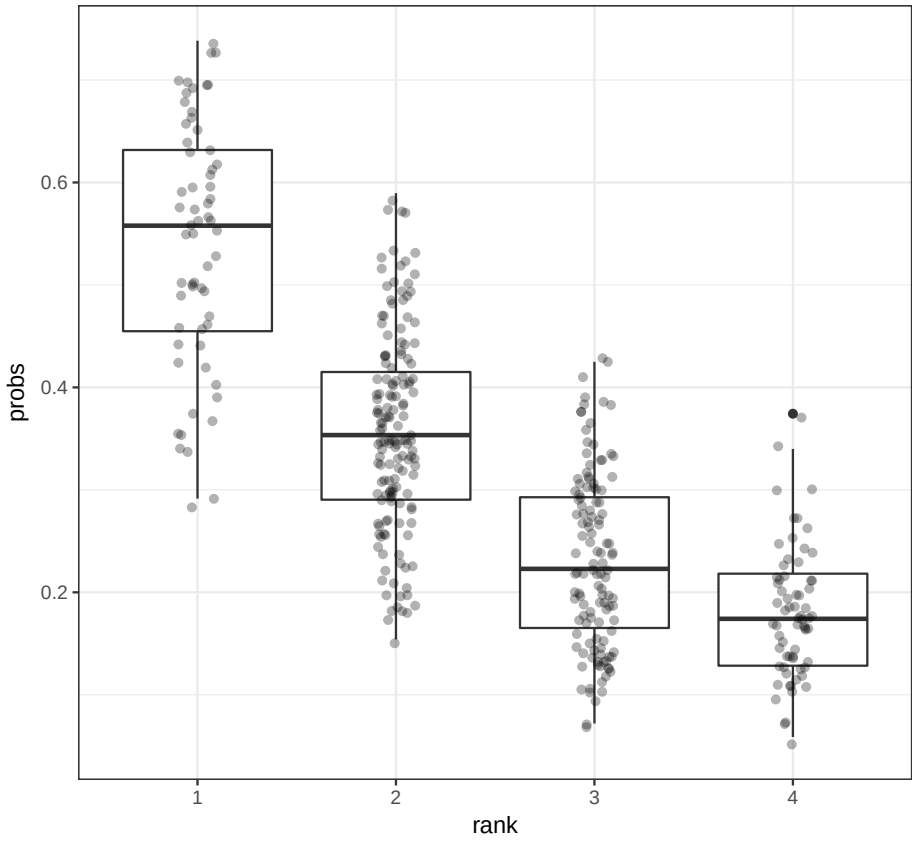
```
> mydata <- data.frame(mydata, probs = myfit$fitted.values)
> ggplot(mydata) + geom_point(aes(x=gpa, y=probs, color=rank)) +
+   geom_jitter(aes(x=gpa, y=admit), width=0, height=0.01, alpha=0.3)
```




```
> ggplot(mydata) + geom_point(aes(x=gre, y=probs, color=rank)) +
+   geom_jitter(aes(x=gre, y=admit), width=0, height=0.01, alpha=0.3)
```




```
> ggplot(mydata) + geom_boxplot(aes(x=rank, y=probs)) +
+   geom_jitter(aes(x=rank, y=probs), width=0.1, height=0.01, alpha=0.3)
```



`glm()` Function

The `glm()` function has many different options available to the user.

```
glm(formula, family = gaussian, data, weights, subset,
    na.action, start = NULL, etastart, mustart, offset,
    control = list(...), model = TRUE, method = "glm.fit",
    x = FALSE, y = TRUE, contrasts = NULL, ...)
```

To see the different link functions available, type:

```
help(family)
```

Nonparametric Regression

Simple Linear Regression

Recall the set up for simple linear regression. For random variables $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$, **simple linear regression** estimates the model

$$Y_i = \beta_1 + \beta_2 X_i + E_i$$

where $\mathrm{E}[E_i|X_i] = 0$, $\mathrm{Var}(E_i|X_i) = \sigma^2$, and $\mathrm{Cov}(E_i, E_j|X_i, X_j) = 0$ for all $1 \leq i, j \leq n$ and $i \neq j$.

Note that in this model $\mathrm{E}[Y|X] = \beta_1 + \beta_2 X$.

Simple Nonparametric Regression

In **simple nonparametric regression**, we define a similar model while eliminating the linear assumption:

$$Y_i = s(X_i) + E_i$$

for some function $s(\cdot)$ with the same assumptions on the distribution of $\boldsymbol{E}|\boldsymbol{X}$. In this model, we also have

$$\mathrm{E}[Y|X] = s(X).$$

Smooth Functions

Suppose we consider fitting the model $Y_i = s(X_i) + E_i$ with the restriction that $s \in C^2$, the class of functions with continuous second derivatives. We can set up an objective function that regularizes how smooth vs wiggly s is.

Specifically, suppose for a given set of observed data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ we wish to identify a function $s \in C^2$ that minimizes for some λ

$$\sum_{i=1}^{n} (y_i - s(x_i))^2 + \lambda \int |s''(x)|^2 dx$$

Smoothness Parameter λ

When minimizing

$$\sum_{i=1}^{n}(y_i - s(x_i))^2 + \lambda \int |s''(x)|^2 dx$$

it follows that if $\lambda = 0$ then any function $s \in C^2$ that interpolates the data is a solution.

As $\lambda \to \infty$, then the minimizing function is the simple linear regression solution.

The Solution

For an observed data set $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ where $n \geq 4$ and a fixed value λ, there is an exact solution to minimizing

$$\sum_{i=1}^{n}(y_i - s(x_i))^2 + \lambda \int |s''(x)|^2 dx.$$

The solution is called a **natural cubic spline**, which is constructed to have knots at $x_1, x_2, \ldots, x_n$.

Natural Cubic Splines

Suppose without loss of generality that we have ordered $x_1 < x_2 < \cdots < x_n$. We assume all x_i are unique to simplify the explanation here, but ties can be deal with.

A **natural cubic spline** (NCS) is a function constructed from a set of piecewise cubic functions over the range $[x_1, x_n]$ joined at the knots so that the second derivative is continuous at the knots. Outside of the range ($< x_1$ or $> x_n$), the spline is linear and it has continuous second derivatives at the endpoint knots.

Basis Functions

Depending on the value λ, a different ncs will be constructed, but the entire family of ncs solutions over $0 < \lambda < \infty$ can be constructed from a common set of basis functions.

We construct n basis functions $N_1(x), N_2(x), \ldots, N_n(x)$ with coefficients $\theta_1(\lambda), \theta_2(\lambda), \ldots, \theta_n(\lambda)$. The NCS takes the form

$$s(x) = \sum_{i=1}^{n} \theta_i(\lambda) N_i(x).$$

Define $N_1(x) = 1$ and $N_2(x) = x$. For $i = 3, \ldots, n$, define $N_i(x) = d_{i-1}(x) - d_{i-2}(x)$ where

$$d_i(x) = \frac{(x - x_i)^3 - (x - x_n)^3}{x_n - x_i}.$$

Recall that we've labeled indices so that $x_1 < x_2 < \cdots < x_n$.

Calculating the Solution

Let $\boldsymbol{\theta}_\lambda = (\theta_1(\lambda), \theta_2(\lambda), \ldots, \theta_n(\lambda))^T$ and let $\boldsymbol{N}$ be the $n \times n$ matrix with (i, j) entry equal to $N_j(x_i)$. Finally, let $\boldsymbol{\Omega}$ be the $n \times n$ matrix with (i, j) entry equal to $\int N_i''(x) N_j''(x) dx$.

The solution to $\boldsymbol{\theta}_\lambda$ are the values that minimize

$$(\boldsymbol{y} - \boldsymbol{N}\boldsymbol{\theta})^T (\boldsymbol{y} - \boldsymbol{N}\boldsymbol{\theta}) + \lambda \boldsymbol{\theta}^T \boldsymbol{\Omega} \boldsymbol{\theta}.$$

which results in

$$\hat{\boldsymbol{\theta}}_\lambda = (\boldsymbol{N}^T \boldsymbol{N} + \lambda \boldsymbol{\Omega})^{-1} \boldsymbol{N}^T \boldsymbol{y}.$$

Linear Operator

Letting

$$\boldsymbol{S}_\lambda = \boldsymbol{N}(\boldsymbol{N}^T \boldsymbol{N} + \lambda \boldsymbol{\Omega})^{-1} \boldsymbol{N}^T$$

it folows that the fitted values are

$$\hat{\boldsymbol{y}} = \boldsymbol{S}_\lambda \boldsymbol{y}.$$

Thus, the fitted values from a NCS are contructed by taking linear combination of the response variable values $y_1, y_2, \ldots, y_n$.

Degrees of Freedom

Recall that in OLS, we formed projection matrix $\boldsymbol{P} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$ and noted that the number of columns p of $\boldsymbol{X}$ is also found in the trace of $\boldsymbol{P}$ where $\text{tr}(\boldsymbol{P}) = p$.

The effective degrees of freedom for a model fit by a linear operator is calculated as the trace of the operator.

Therefore, for a given λ, the effective degrees of freedom is

$$\text{df}_\lambda = \text{tr}(\boldsymbol{S}_\lambda).$$

Bayesian Intepretation

Minimizing

$$\sum_{i=1}^{n}(y_i - s(x_i))^2 + \lambda \int |s''(x)|^2 dx$$

is equivalent to maximizing

$$\exp\left\{-\frac{\sum_{i=1}^{n}(y_i - s(x_i))^2}{2\sigma^2}\right\} \exp\left\{-\frac{\lambda}{2\sigma^2}\int |s''(x)|^2 dx\right\}.$$

Therefore, the NCS solution can be interpreted as calculting the MAP where $Y|X$ is Normal and there's an Exponential prior on the smoothness of s.

Bias and Variance Trade-off

Typically we will choose some $0 < \lambda < \infty$ in an effort to balance the bias and variance. Let $\hat{Y} = \hat{s}(X;\lambda)$ where $\hat{s}(\cdot;\lambda)$ minimizes the above for some chosen λ on an independent data set. Then

$$\begin{aligned}
\text{E}\left[\left(Y - \hat{Y}\right)^2\right] &= \text{E}\left[\left(s(x) + E - \hat{s}(x;\lambda)\right)^2\right] \\
&= \text{E}\left[\left(s(x) - \hat{s}(x;\lambda)\right)^2\right] + \text{Var}(E) \\
&= \left(s(x) - \text{E}[\hat{s}(x;\lambda)]\right)^2 + \text{Var}\left(\hat{s}(x;\lambda)\right) + \text{Var}(E) \\
&= \text{bias}_\lambda^2 + \text{variance}_\lambda + \text{Var}(E)
\end{aligned}$$

where all of the above calculations are conditioned on $X = x$.

In minimizing

$$\sum_{i=1}^{n}(y_i - s(x_i))^2 + \lambda \int |s''(x)|^2 dx$$

the relationship is such that:

$$\uparrow \lambda \Longrightarrow \text{bias}^2 \uparrow, \text{variance} \downarrow$$

$$\downarrow \lambda \Longrightarrow \text{bias}^2 \downarrow, \text{variance} \uparrow$$

Choosing λ

There are several approaches that are commonly used to identify a value of λ, including:

- Scientific knowledge that guides the acceptable value of df_λ
- Cross-validation or some other prediction quality measure
- Model selection measures, such as Akaike information criterion (AIC) or Mallows C_p

Smoothers and Spline Models

We investigated one type of nonparametric regression model here, the NCS. However, in general there are many such "smoother" methods available in the simple nonparametric regression scenario.

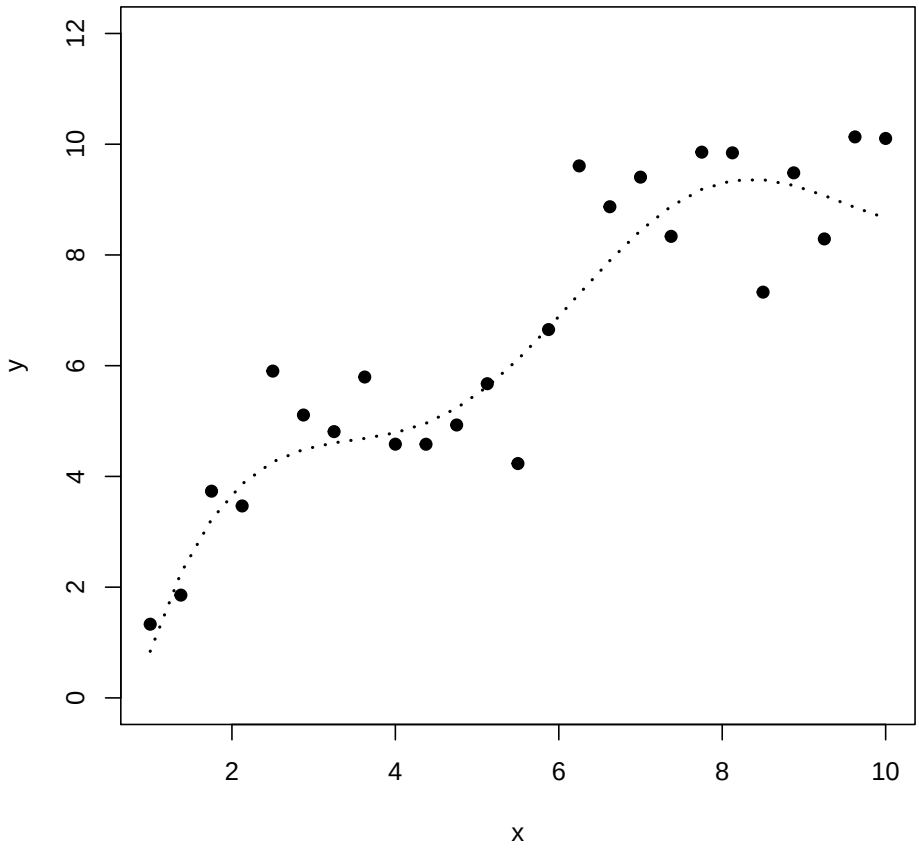
Splines are particularly popular since splines are constructed from putting together polynomials and are therefore usually tractable to compute and analyze.

Smoothers in R

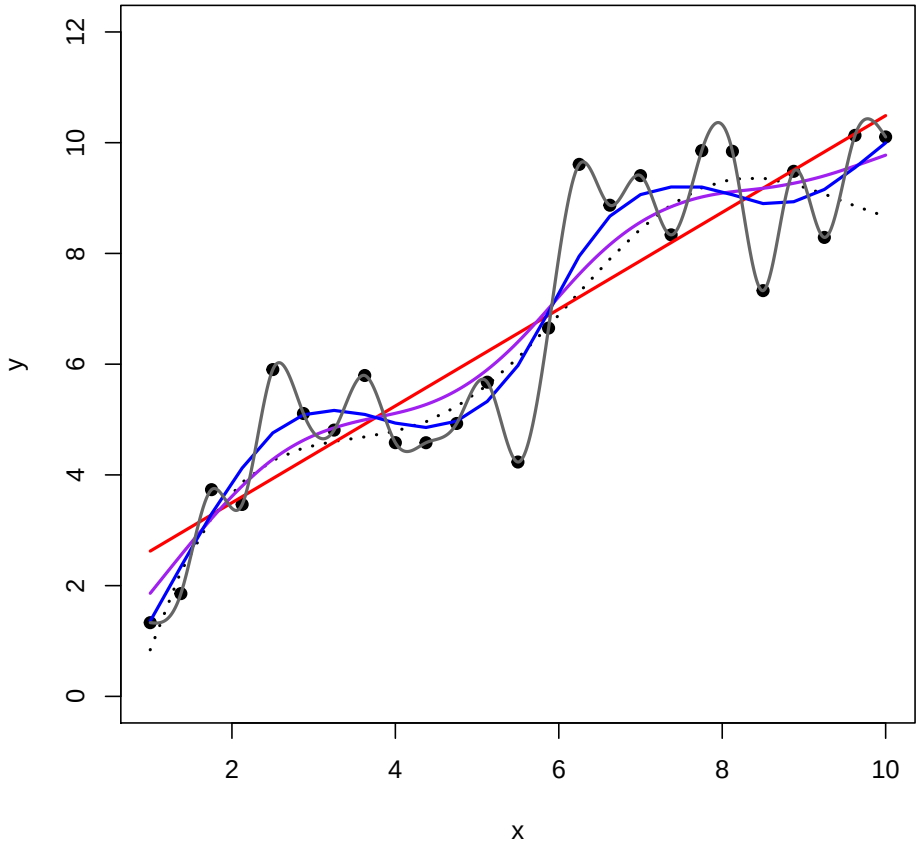
There are several functions and packages available in R for computing smoothers and tuning smoothness parameters. Examples include:

- `splines` library
- `smooth.spline()`
- `loess()`
- `lowess()`

Example




```
> y2 <- smooth.spline(x=x, y=y, df=2)
> y5 <- smooth.spline(x=x, y=y, df=5)
> y25 <- smooth.spline(x=x, y=y, df=25)
> ycv <- smooth.spline(x=x, y=y)
> ycv
Call:
smooth.spline(x = x, y = y)

Smoothing Parameter  spar= 0.5162045  lambda= 0.0002730906 (11 iterations)
Equivalent Degrees of Freedom (Df): 7.293673
Penalized Criterion: 14.80602
GCV: 1.180651
```



Generalized Additive Models

Ordinary Least Squares

Recall that OLS estimates the model

$$\begin{aligned} Y_i &= \beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip} + E_i \\ &= \boldsymbol{X}_i \boldsymbol{\beta} + E_i \end{aligned}$$

where $\mathrm{E}[\boldsymbol{E}|\boldsymbol{X}] = \boldsymbol{0}$ and $\mathrm{Cov}(\boldsymbol{E}|\boldsymbol{X}) = \sigma^2 \boldsymbol{I}$.

Additive Models

The **additive model** (which could also be called "ordinary nonparametric additive regression") is of the form

$$
\begin{aligned}
Y_i &= s_1(X_{i1}) + s_2(X_{i2}) + \ldots + s_p(X_{ip}) + E_i \\
&= \sum_{j=1}^{p} s_j(X_{ij}) + E_i
\end{aligned}
$$

where the $s_j(\cdot)$ for $j = 1, \ldots, p$ are a set of nonparametric (or flexible) functions. Again, we assume that $\mathrm{E}[\boldsymbol{E}|\boldsymbol{X}] = \mathbf{0}$ and $\mathrm{Cov}(\boldsymbol{E}|\boldsymbol{X}) = \sigma^2\boldsymbol{I}$.

Backfitting

The additive model can be fit through a technique called **backfitting**.

1. Intialize $s_j^{(0)}(x)$ for $j = 1, \ldots, p$.
2. For $t = 1, 2, \ldots$, fit $s_j^{(t)}(x)$ on response variable

$$y_i - \sum_{k \neq j} s_k^{(t-1)}(x_{ij}).$$

3. Repeat until convergence.

Note that some extra steps have to be taken to deal with the intercept.

GAM Definition

$Y|\boldsymbol{X}$ is distributed according to an exponential family distribution. The extension of additive models to this family of response variable is called **generalized additive models** (GAMs). The model is of the form

$$g\left(\mathrm{E}[Y_i|\boldsymbol{X}_i]\right) = \sum_{j=1}^{p} s_j(X_{ij})$$

where $g(\cdot)$ is the link function and the $s_j(\cdot)$ are flexible and/or nonparametric functions.

Overview of Fitting GAMs

Fitting GAMs involves putting together the following three tools:

1. We know how to fit a GLM via IRLS
2. We know how to fit a smoother of a single explanatory variable via a least squares solution, as seen for the NCS
3. We know how to combine additive smoothers by backfitting

GAMs in R

Three common ways to fit GAMs in R:

1. Utilize `glm()` on explanatory variables constructed from `ns()` or `bs()`
2. The `gam` library
3. The `mgcv` library

Example

```
> set.seed(508)
> x1 <- seq(1, 10, length.out=50)
> n <- length(x1)
> x2 <- rnorm(n)
> f <- 4*log(x1) + sin(x1) - 7 + 0.5*x2
> p <- exp(f)/(1+exp(f))
> summary(p)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.001842 0.074170 0.310700 0.436200 0.860400 0.944800
> y <- rbinom(n, size=1, prob=p)
> mean(y)
[1] 0.42
> df <- data.frame(x1=x1, x2=x2, y=y)
```

Here, we use the `gam()` function from the `mgcv` library. It automates choosing the smoothness of the splines.

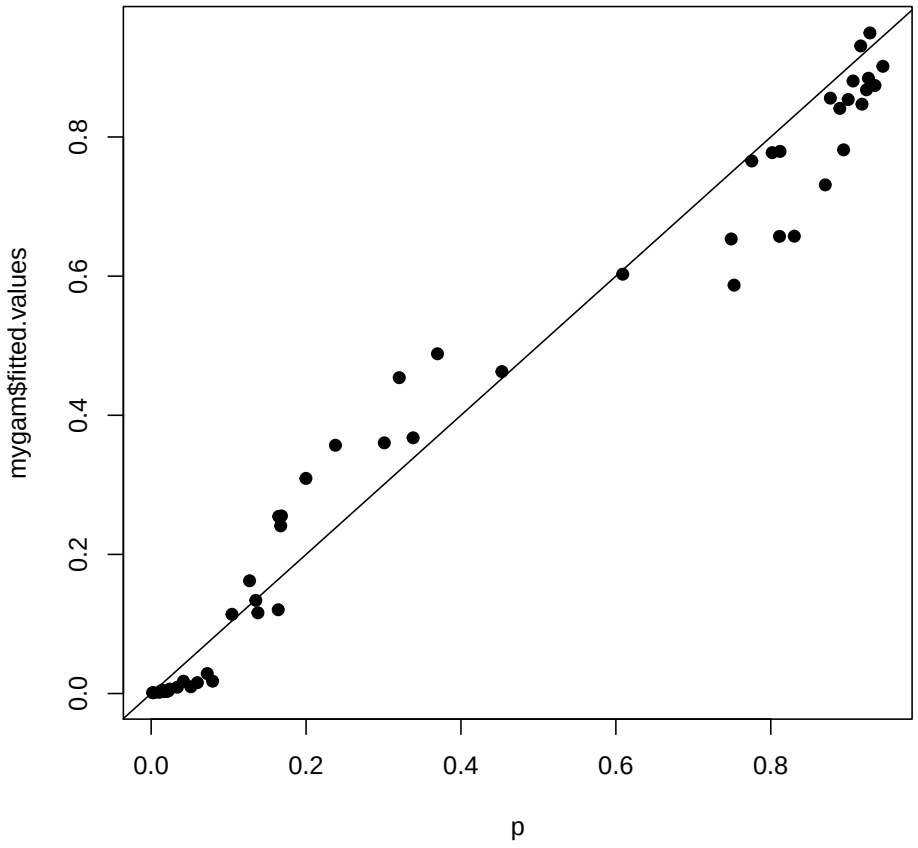
```
> library(mgcv)
> mygam <- gam(y ~ s(x1) + s(x2), family = binomial(), data=df)
> library(broom)
> tidy(mygam)
   term      edf   ref.df statistic     p.value
1 s(x1) 1.870282 2.374897 12.742930 0.005308795
2 s(x2) 1.000024 1.000048  1.163059 0.280836876
```

```
> summary(mygam)

Family: binomial
Link function: logit

Formula:
y ~ s(x1) + s(x2)

Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.1380     0.6723  -1.693   0.0905 .
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
       edf Ref.df Chi.sq p-value
s(x1) 1.87  2.375 12.743 0.00531 **
s(x2) 1.00  1.000  1.163 0.28084
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.488   Deviance explained =   47%
UBRE = -0.12392  Scale est. = 1         n = 50
```

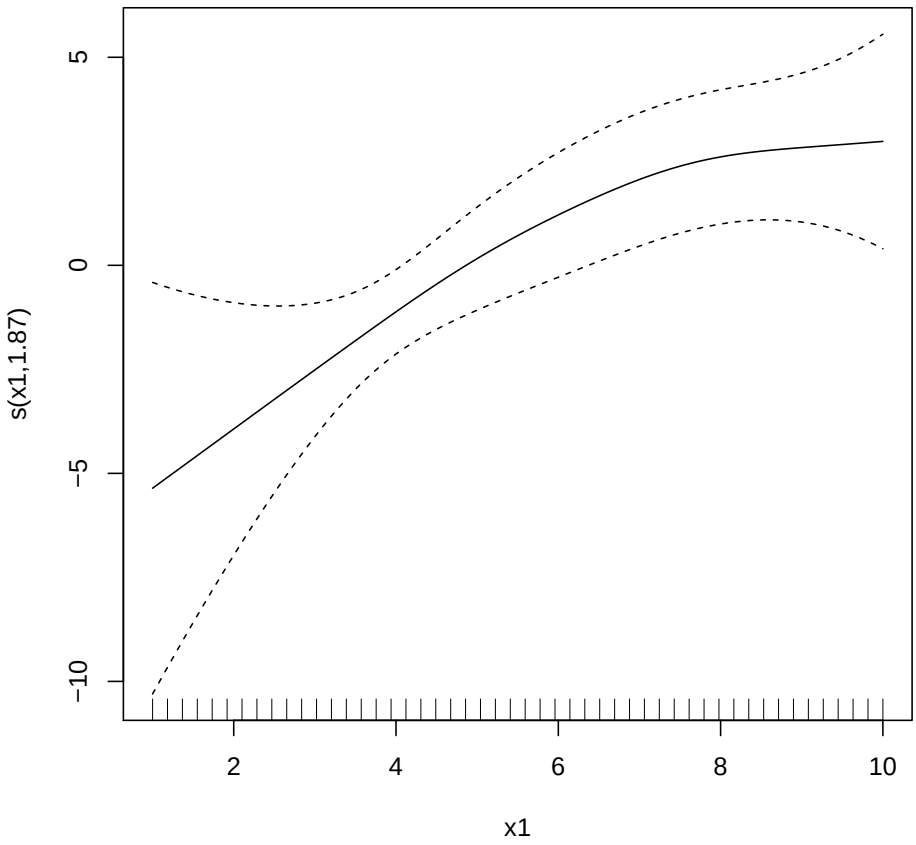
True probabilities vs. estimated probabilities.

```
> plot(p, mygam$fitted.values, pch=19); abline(0,1)
```



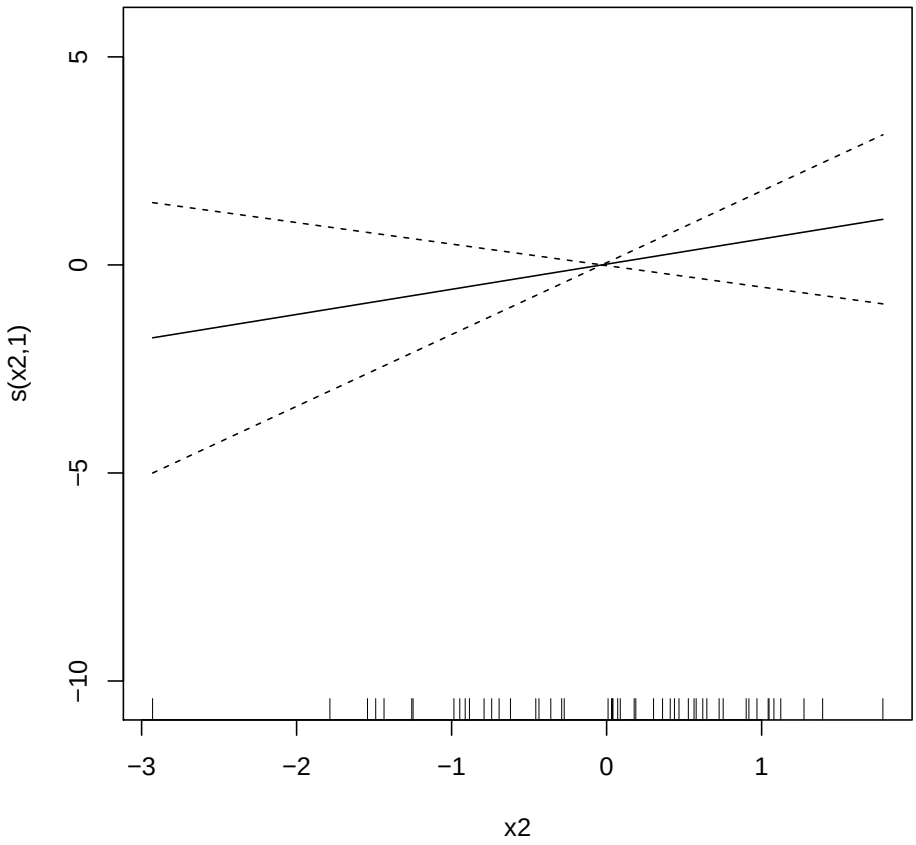
Smoother estimated for `x1`.

```
> plot(mygam, select=1)
```



Smoother estimated for x2.

```
> plot(mygam, select=2)
```



Here, we use the `glm()` function and include as an explanatory variable a NCS built from the `ns()` function from the `splines` library. We include a `df` argument in the `ns()` call.

```
> library(splines)
> myglm <- glm(y ~ ns(x1, df=2) + x2, family = binomial(), data=df)
> tidy(myglm)
              term    estimate  std.error statistic     p.value
1     (Intercept) -10.9228749  5.3078903 -2.057856 0.039603941
2 ns(x1, df = 2)1  21.3848301 10.1317747  2.110670 0.034800710
3 ns(x1, df = 2)2   6.3266262  2.1103294  2.997933 0.002718174
4              x2   0.7341812  0.6089442  1.205663 0.227947633
```

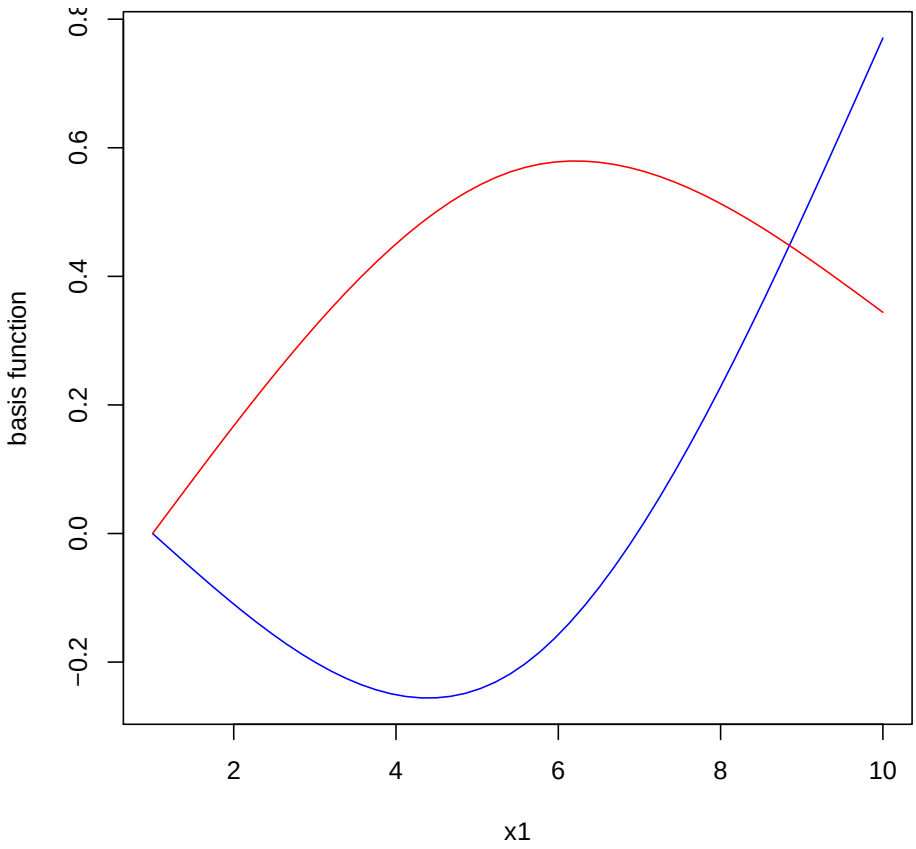
The spline basis evaluated at `x1` values.

```
> ns(x1, df=2)
                1           2
 [1,] 0.00000000  0.00000000
 [2,] 0.03114456 -0.02075171
 [3,] 0.06220870 -0.04138180
 [4,] 0.09311200 -0.06176867
```

```
 [5,] 0.12377405 -0.08179071
 [6,] 0.15411442 -0.10132630
 [7,] 0.18405270 -0.12025384
 [8,] 0.21350847 -0.13845171
 [9,] 0.24240131 -0.15579831
[10,] 0.27065081 -0.17217201
[11,] 0.29817654 -0.18745121
[12,] 0.32489808 -0.20151430
[13,] 0.35073503 -0.21423967
[14,] 0.37560695 -0.22550571
[15,] 0.39943343 -0.23519080
[16,] 0.42213406 -0.24317334
[17,] 0.44362840 -0.24933170
[18,] 0.46383606 -0.25354429
[19,] 0.48267660 -0.25568949
[20,] 0.50006961 -0.25564569
[21,] 0.51593467 -0.25329128
[22,] 0.53019136 -0.24850464
[23,] 0.54275927 -0.24116417
[24,] 0.55355797 -0.23114825
[25,] 0.56250705 -0.21833528
[26,] 0.56952943 -0.20260871
[27,] 0.57462513 -0.18396854
[28,] 0.57787120 -0.16253131
[29,] 0.57934806 -0.13841863
[30,] 0.57913614 -0.11175212
[31,] 0.57731586 -0.08265339
[32,] 0.57396762 -0.05124405
[33,] 0.56917185 -0.01764570
[34,] 0.56300897  0.01802003
[35,] 0.55555939  0.05563154
[36,] 0.54690354  0.09506722
[37,] 0.53712183  0.13620546
[38,] 0.52629468  0.17892464
[39,] 0.51450251  0.22310315
[40,] 0.50182573  0.26861939
[41,] 0.48834478  0.31535174
[42,] 0.47414005  0.36317859
[43,] 0.45929198  0.41197833
[44,] 0.44388099  0.46162934
[45,] 0.42798748  0.51201003
[46,] 0.41169188  0.56299877
[47,] 0.39507460  0.61447395
[48,] 0.37821607  0.66631397
[49,] 0.36119670  0.71839720
```

```
[50,] 0.34409692  0.77060206
attr(,"degree")
[1] 3
attr(,"knots")
50%
5.5
attr(,"Boundary.knots")
[1]  1 10
attr(,"intercept")
[1] FALSE
attr(,"class")
[1] "ns"     "basis"  "matrix"
```

Plot of basis function values vs `x1`.



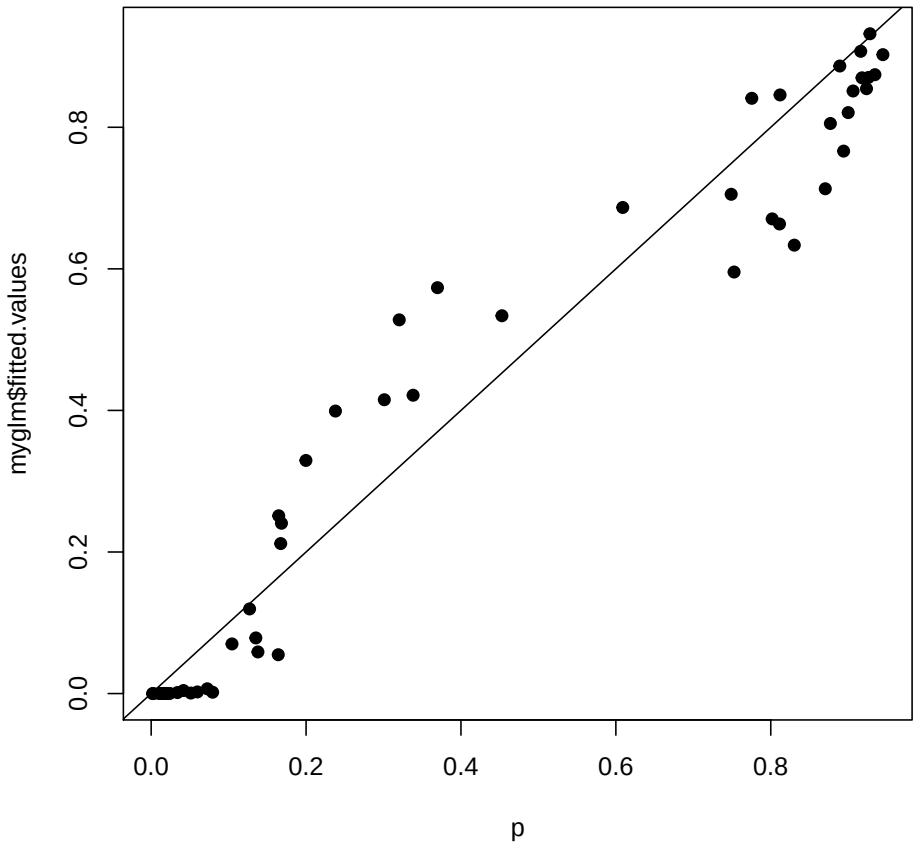

```
> summary(myglm)

Call:
glm(formula = y ~ ns(x1, df = 2) + x2, family = binomial(), data = df)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0214  -0.3730  -0.0162   0.5762   1.7616

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)     -10.9229     5.3079  -2.058  0.03960 *
ns(x1, df = 2)1  21.3848    10.1318   2.111  0.03480 *
ns(x1, df = 2)2   6.3266     2.1103   2.998  0.00272 **
x2                0.7342     0.6089   1.206  0.22795
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 68.029  on 49  degrees of freedom
Residual deviance: 35.682  on 46  degrees of freedom
AIC: 43.682

Number of Fisher Scoring iterations: 7
```

```
> anova(myglm, test="LRT")
Analysis of Deviance Table

Model: binomial, link: logit

Response: y

Terms added sequentially (first to last)

              Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                             49     68.029
ns(x1, df = 2)  2   30.755        47     37.274 2.097e-07 ***
x2              1    1.592        46     35.682     0.207
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

True probabilities vs. estimated probabilities.

```
> plot(p, myglm$fitted.values, pch=19); abline(0,1)
```



Bootstrap for Statistical Models

Homoskedastic Models

Let's first discuss how one can utilize the bootstrap on any of the three homoskedastic models:

- Simple linear regression
- Ordinary least squares
- Additive models

Residuals

In each of these scenarios we sample data $(\boldsymbol{X}_1, Y_1), (\boldsymbol{X}_2, Y_2), \ldots, (\boldsymbol{X}_n, Y_n)$. Let suppose we calculate fitted values $\hat{Y}_i$ and they are unbiased:

$$\mathrm{E}[\hat{Y}_i|\boldsymbol{X}] = \mathrm{E}[Y_i|\boldsymbol{X}].$$

We can calculate residuals $\hat{E}_i = Y_i - \hat{Y}_i$ for $i = 1, 2, \ldots, n$.

Studentized Residuals

One complication is that the residuals have a covariance. For example, in OLS we showed that

$$\mathrm{Cov}(\hat{\boldsymbol{E}}) = \sigma^2(\boldsymbol{I} - \boldsymbol{P})$$

where $\boldsymbol{P} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$.

To correct for this induced heteroskedasticity, we studentize the residuals by calculating

$$R_i = \frac{\hat{E}_i}{\sqrt{1 - P_{ii}}}$$

which gives $\mathrm{Cov}(\boldsymbol{R}) = \sigma^2\boldsymbol{I}$.

Confidence Intervals

The following is a bootstrap procedure for calculating a confidence interval on some statistic $\hat{\theta}$ calculated from a homoskedastic model fit. An example is $\hat{\beta}_j$ in an OLS.

1. Fit the model to obtain fitted values $\hat{Y}_i$, studentized residuals R_i, and the statistic of interest $\hat{\theta}$.
   For $b = 1, 2, \ldots, B$.
2. Sample n observations with replacement from $\{R_i\}_{i=1}^n$ to obtain bootstrap residuals $R_1^*, R_2^*, \ldots, R_n^*$.
3. Form new response variables $Y_i^* = \hat{Y}_i + R_i^*$.
4. Fit the model to obtain $\hat{Y}_i^*$ and all other fitted parameters.
5. Calculate statistic of interest $\hat{\theta}^{*(b)}$.

The bootstrap statistics $\hat{\theta}^{*(1)}, \hat{\theta}^{*(2)}, \ldots, \hat{\theta}^{*(B)}$ are then utilized through one of the techniques discussed earlier (percentile, pivotal, studentized pivotal) to calculate a bootstrap CI.

Hypothesis Testing

Suppose we are testing the hypothesis $H_0 : \mathrm{E}[Y|\boldsymbol{X}] = f_0(\boldsymbol{X})$ vs $H_1 : \mathrm{E}[Y|\boldsymbol{X}] = f_1(\boldsymbol{X})$. Suppose it is possible to form unbiased estimates $f_0(\boldsymbol{X})$ and $f_1(\boldsymbol{X})$ given $\boldsymbol{X}$, and f_0 is a restricted version of f_1.

Suppose also we have a statistic $T(\hat{f}_0, \hat{f}_1)$ for performing this test so that the larger the statistic, the more evidence there is against the null hypothesis in favor of the alternative.

The big picture strategy is to bootstrap studentized residuals from the unconstrained (alternative hypothesis) fitted model and then add those to the constrained (null hypothesis) fitted model to generate bootstrap null data sets.

1. Fit the models to obtain fitted values $\hat{f}_0(\boldsymbol{X}_i)$ and $\hat{f}_1(\boldsymbol{X}_i)$, studentized residuals R_i from the fit $\hat{f}_1(\boldsymbol{X}_i)$, and the observed statistic $T(\hat{f}_0, \hat{f}_1)$. For $b = 1, 2, \ldots, B$.
2. Sample n observations with replacement from $\{R_i\}_{i=1}^n$ to obtain bootstrap residuals $R_1^*, R_2^*, \ldots, R_n^*$.
3. Form new response variables $Y_i^* = \hat{f}_0(\boldsymbol{X}_i) + R_i^*$.
4. Fit the models on the response variables Y_i^* to obtain $\hat{f}_0^*$ and $\hat{f}_1^*$.
5. Calculate statistic $T(\hat{f}_0^{*(b)}, \hat{f}_1^{*(b)})$.

The p-value is then calculated as

$$\frac{\sum_{b=1}^{B} 1\left(T(\hat{f}_0^{*(b)}, \hat{f}_1^{*(b)}) \geq T(\hat{f}_0, \hat{f}_1)\right)}{B}$$

Parametric Bootstrap

For more complex scenarios, such as GLMs, GAMs, and heteroskedastic models, it is typically more straightforward to utilize a parametric bootstrap.

Extras

Source

License

Source Code

Session Information

```
> sessionInfo()
R version 3.3.2 (2016-10-31)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: macOS Sierra 10.12.4

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] splines   stats     graphics  grDevices utils     datasets
[7] methods   base

other attached packages:
 [1] mgcv_1.8-17     nlme_3.1-131    lmtest_0.9-35
 [4] zoo_1.8-0       car_2.1-4       broom_0.4.2
 [7] dplyr_0.5.0     purrr_0.2.2     readr_1.0.0
[10] tidyr_0.6.1     tibble_1.2      ggplot2_2.2.1
[13] tidyverse_1.1.1 knitr_1.15.1    magrittr_1.5
[16] devtools_1.12.0

loaded via a namespace (and not attached):
 [1] reshape2_1.4.2     haven_1.0.0        lattice_0.20-34
 [4] colorspace_1.3-2   htmltools_0.3.5    yaml_2.1.14
 [7] nloptr_1.0.4       foreign_0.8-67     withr_1.0.2
[10] DBI_0.5-1          modelr_0.1.0       readxl_0.1.1
[13] plyr_1.8.4         stringr_1.1.0      MatrixModels_0.4-1
[16] munsell_0.4.3      gtable_0.2.0       rvest_0.3.2
[19] codetools_0.2-15   psych_1.6.12       memoise_1.0.0
[22] evaluate_0.10      labeling_0.3       forcats_0.2.0
[25] SparseM_1.74       quantreg_5.29      pbkrtest_0.4-6
[28] parallel_3.3.2     Rcpp_0.12.9        scales_0.4.1
[31] backports_1.0.5    jsonlite_1.2       lme4_1.1-12
[34] mnormt_1.5-5       hms_0.3            digest_0.6.12
[37] stringi_1.1.2      grid_3.3.2         rprojroot_1.2
[40] tools_3.3.2        lazyeval_0.2.0     MASS_7.3-45
[43] Matrix_1.2-8       xml2_1.1.1         lubridate_1.6.0
[46] assertthat_0.1     minqa_1.2.4        rmarkdown_1.3
[49] httr_1.2.1         R6_2.2.0           nnet_7.3-12
```