Note that the first column changes fastest and the last slowest. This is a single complete replicate.

Our example used three replicates, each split into two blocks so that the block comparison is confounded with the highest-order interaction. We can construct such a design in stages. First we find a half-replicate to be repeated three times and form the contents of three of the blocks. The simplest way to do this is to use `fac.design`:

```
blocks13 <- fac.design(levels = c(2, 2, 2),
     factor= list(N=mp, P=mp, K=mp), rep = 3, fraction = 1/2)
```

The first two arguments give the numbers of levels and the factor names and level labels. The third argument gives the number of replications (default 1). The `fraction` argument may only be used for 2^p factorials. It may be given either as a small negative power of 2, as here, or as a *defining contrast formula*. When `fraction` is numerical the function chooses a defining contrast that becomes the `fraction` attribute of the result. For half-replicates the highest-order interaction is chosen to be aliased with the mean. To find the complementary fraction for the remaining three blocks we need to use the defining contrast formula form for `fraction`:

```
blocks46 <- fac.design(levels = c(2, 2, 2),
    factor = list(N=mp, P=mp, K=mp), rep = 3, fraction = ~ -N:P:K)
```

(This is explained in the following.) To complete our design we put the blocks together, add in the `block` factor and randomize:

```
NPK <- design(block = factor(rep(1:6, each = 4)),
              rbind(blocks13, blocks46))
i <- order(runif(6)[NPK$block], runif(24))
NPK <- NPK[i,]  # Randomized
```

Using `design` instead of `data.frame` creates an object of class `design` that inherits from `data.frame`. For most purposes designs and data frames are equivalent, but some generic functions such as `plot`, `formula` and `alias` have useful `design` methods.

Defining contrast formulae resemble model formulae in syntax only; the meaning is quite distinct. There is no left-hand side. The right-hand side consists of colon products of factors only, separated by `+` or `-` signs. A plus (or leading blank) specifies that the treatments with *positive* signs for that contrast are to be selected and a minus those with *negative* signs. A formula such as `~A:B:C-A:D:E` specifies a quarter-replicate consisting of the treatments that have a positive sign in the ABC interaction and a negative sign in ADE.

Box, Hunter and Hunter (1978, §12.5) consider a 2^{7-4} design used for an experiment in riding up a hill on a bicycle. The seven factors are Seat (up or down), Dynamo (off or on), Handlebars (up or down), Gears (low or medium), Raincoat (on or off), Breakfast (yes or no) and Tyre pressure (hard or soft). A resolution III design was used, so the main effects are not aliased with each other. Such a design cannot be constructed using a numerical fraction in `fac.design`, so the defining contrasts have to be known. Box *et al.* use the design relations:

$$D = AB, \quad E = AC, \quad F = BC, \quad G = ABC$$

which mean that ABD, ACE, BCF and $ABCG$ are all aliased with the mean, and form the defining contrasts of the fraction. Whether we choose the positive or negative halves is immaterial here.

```
> lev <- rep(2, 7)
> factors <- list(S=mp, D=mp, H=mp, G=mp, R=mp, B=mp, P=mp)
> (Bike <- fac.design(lev, factors,
      fraction = ~ S:D:G + S:H:R + D:H:B + S:D:H:P))
  S D H G R B P
1 - - - - - - -
2 - + + + + - -
3 + - + + - + -
4 + + - - + + -
5 + + + - - - +
6 + - - + + - +
7 - + - + - + +
8 - - + - + + +

Fraction:  ~ S:D:G + S:H:R + D:H:B + S:D:H:P
```

(We chose `P` for pressure rather than `T` for tyres since `T` and `F` are reserved identifiers.)

We may check the symmetry of the design using `replications`:

```
> replications(~.^2, data = Bike)
 S D H G R B P S:D S:H S:G S:R S:B S:P D:H D:G D:R D:B D:P H:G
 4 4 4 4 4 4 4   2   2   2   2   2   2   2   2   2   2   2   2
 H:R H:B H:P G:R G:B G:P R:B R:P B:P
   2   2   2   2   2   2   2   2   2
```

Fractions may be specified either in a call to `fac.design` or subsequently using the `fractionate` function.

The third function, `oa.design`, provides some resolution III designs (also known as *main effect plans* or *orthogonal arrays*) for factors at two or three levels. Only low-order cases are provided, but these are the most useful in practice.

6.8 An Unbalanced Four-Way Layout

Aitkin (1978) discussed an observational study of S. Quine. The response is the number of days absent from school in a year by children from a large town in rural New South Wales, Australia. The children were classified by four factors, namely,

`Age`	4 levels: primary, first, second or third form
`Eth`	2 levels: aboriginal or non-aboriginal
`Lrn`	2 levels: slow or average learner
`Sex`	2 levels: male or female.

The dataset is included in the paper of Aitkin (1978) and is available as data frame `quine` in `MASS`. This has been explored several times already, but we now consider a more formal statistical analysis.

There were 146 children in the study. The frequencies of the combinations of factors are

```
> attach(quine)
> table(Lrn, Age, Sex, Eth)
, , F, A                    , , F, N
   F0 F1 F2 F3                 F0 F1 F2 F3
AL  4  5  1  9              AL  4  6  1 10
SL  1 10  8  0              SL  1 11  9  0

, , M, A                    , , M, N
   F0 F1 F2 F3                 F0 F1 F2 F3
AL  5  2  7  7              AL  6  2  7  7
SL  3  3  4  0              SL  3  7  3  0
```

(The output has been slightly rearranged to save space.) The classification is unavoidably very unbalanced. There are no slow learners in form F3, but all 28 other cells are non-empty. In his paper Aitkin considers a normal analysis on the untransformed response, but in the reply to the discussion he chooses a transformed response, $\log(\text{Days} + 1)$.

A casual inspection of the data shows that homoscedasticity is likely to be an unrealistic assumption on the original scale, so our first step is to plot the cell variances and standard deviations against the cell means.

```
Means <- tapply(Days, list(Eth, Sex, Age, Lrn), mean)
Vars  <- tapply(Days, list(Eth, Sex, Age, Lrn), var)
SD <- sqrt(Vars)
par(mfrow = c(1, 2))
plot(Means, Vars, xlab = "Cell Means", ylab = "Cell Variances")
plot(Means, SD, xlab = "Cell Means", ylab = "Cell Std Devn.")
```

Missing values are silently omitted from the plot. Interpretation of the result in Figure 6.5 requires some caution because of the small and widely different degrees of freedom on which each variance is based. Nevertheless the approximate linearity of the standard deviations against the cell means suggests a logarithmic transformation or something similar is appropriate. (See, for example, Rao, 1973, §6g.)

Some further insight on the transformation needed is provided by considering a model for the transformed observations

$$y^{(\lambda)} = \begin{cases} (y^\lambda - 1)/\lambda & \lambda \neq 0 \\ \log y & \lambda = 0 \end{cases}$$

where here $y = \text{Days} + \alpha$. (The positive constant α is added to avoid problems with zero entries.) Rather than include α as a second parameter we first consider Aitkin's choice of $\alpha = 1$. Box and Cox (1964) show that the profile likelihood function for λ is

$$\widehat{L}(\lambda) = \text{const} - \tfrac{n}{2} \log \text{RSS}(z^{(\lambda)})$$

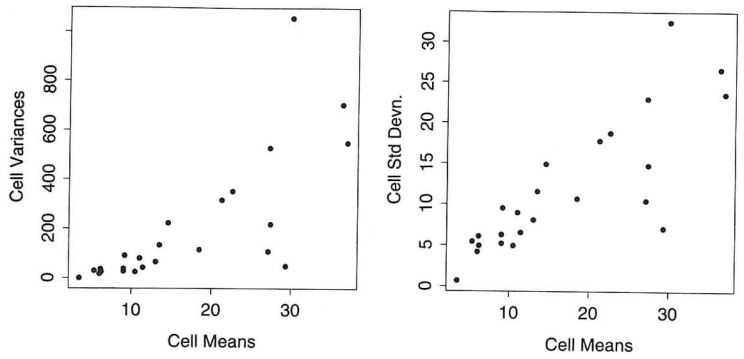

Figure 6.5: Two diagnostic plots for the Quine data.

where $z^{(\lambda)} = y^{(\lambda)}/\dot{y}^{\lambda-1}$, $\dot{y}$ is the geometric mean of the observations and $\text{RSS}(z^{(\lambda)})$ is the residual sum of squares for the regression of $z^{(\lambda)}$.

Box & Cox suggest using the profile likelihood function for the largest linear model to be considered as a guide in choosing a value for λ, which will then remain fixed for any remaining analyses. Ideally other considerations from the context will provide further guidance in the choice of λ, and in any case it is desirable to choose easily interpretable values such as square-root, log or inverse.

Our `MASS` function `boxcox` calculates and (optionally) displays the Box–Cox profile likelihood function, together with a horizontal line showing what would be an approximate 95% likelihood ratio confidence interval for λ. The function is generic and several calling protocols are allowed but a convenient one to use here is with the same arguments as `lm` together with an additional (named) argument, `lambda`, to provide the sequence at which the marginal likelihood is to be evaluated. (By default the result is extended using a spline interpolation.)

Since the dataset has four empty cells the full model `Eth*Sex*Age*Lrn` has a rank-deficient model matrix. Hence in S-PLUS we must use `singular.ok = T` to fit the model. S+

```
boxcox(Days+1 ~ Eth*Sex*Age*Lrn, data = quine, singular.ok = T,
       lambda = seq(-0.05, 0.45, len = 20))
```

(Alternatively the first argument may be a fitted model object that supplies all needed information apart from `lambda`.) The result is shown on the left-hand panel of Figure 6.6 which suggests strongly that a log transformation is not optimal when $\alpha = 1$ is chosen. An alternative one-parameter family of transformations that could be considered in this case is

$$t(y, \alpha) = \log(y + \alpha)$$

Using the same analysis as presented in Box and Cox (1964) the profile log likelihood for α is easily seen to be

$$\widehat{L}(\alpha) = \text{const} - \tfrac{n}{2} \log \text{RSS}\{\log(y + \alpha)\} - \sum \log(y + \alpha)$$

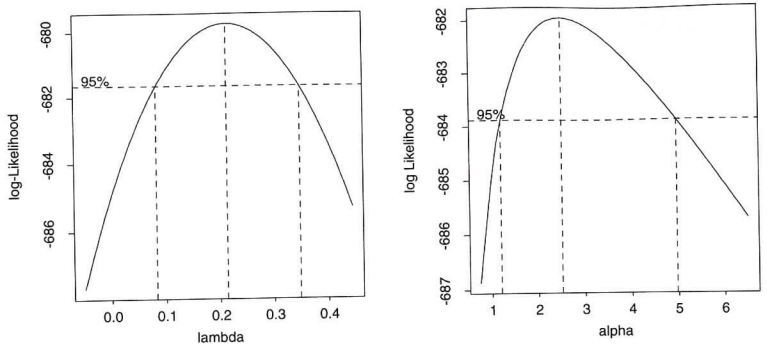

Figure 6.6: Profile likelihood for a Box–Cox transformation model with displacement $\alpha = 1$, left, and a displaced log transformation model, right.

It is interesting to see how this may be calculated directly using low-level tools, in particular the functions `qr` for the QR-decomposition and `qr.resid` for orthogonal projection onto the residual space. Readers are invited to look at our functions `logtrans.default` and `boxcox.default`.

```
logtrans(Days ~ Age*Sex*Eth*Lrn, data = quine,
    alpha = seq(0.75, 6.5, len = 20), singular.ok = T)
```

The result is shown in the right-hand panel of Figure 6.6. If a displaced log transformation is chosen a value $\alpha = 2.5$ is suggested, and we adopt this in our analysis. Note that $\alpha = 1$ is outside the notional 95% confidence interval. It can also be checked that with $\alpha = 2.5$ the log transform is well within the range of reasonable Box–Cox transformations to choose.

Model selection

The complete model, `Eth*Sex*Age*Lrn`, has a different parameter for each identified group and hence contains all possible simpler models for the mean as special cases, but has little predictive or explanatory power. For a better insight into the mean structure we need to find more parsimonious models. Before considering tools to prune or extend regression models it is useful to make a few general points on the process itself.

Marginality restrictions

In regression models it is usually the case that not all terms are on an equal footing as far as inclusion or removal is concerned. For example, in a quadratic regression on a single variable x one would normally consider removing only the highest-degree term, x^2, first. Removing the first-degree term while the second-degree one is still present amounts to forcing the fitted curve to be flat at $x = 0$, and unless there were some good reason from the context to do this it would be an arbitrary imposition on the model. Another way to view this is to note that if we write a polynomial regression in terms of a new variable $x^\star = x - \alpha$ the model remains in predictive terms the same, but only the highest-order coefficient remains invariant. If, as is usually the case, we would like our model selection procedure not to depend on the arbitrary choice of origin we must work only with the highest-degree terms at each stage.

The linear term in x is said to be *marginal* to the quadratic term, and the intercept term is marginal to both. In a similar way if a second-degree term in two variables, $x_1 x_2$, is present, any linear terms in either variable or an intercept term are marginal to it.

There are circumstances where a regression through the origin does make sense, but in cases where the origin is arbitrary one would normally only consider regression models where for each term present all terms marginal to it are also present.

In the case of factors the situation is even more clear-cut. A two-factor interaction `a:b` is marginal to any higher-order interaction that contains `a` and `b`. Fitting a model such as `a + a:b` leads to a model matrix where the columns corresponding to `a:b` are extended to compensate for the absent marginal term, `b`, and the fitted values are the same as if it were present. Fitting models with marginal terms removed such as with `a*b - b` generates a model with no readily understood statistical meaning[11] but updating models specified in this way using `update` changes the model matrix so that the absent marginal term again has no effect on the fitted model. In other words, removing marginal factor terms from a fitted model is either statistically meaningless or futile in the sense that the model simply changes its parametrization to something equivalent.

Variable selection for the Quine data

The `anova` function when given a single fitted-model object argument constructs a *sequential* analysis of variance table. That is, a sequence of models is fitted by expanding the formula, arranging the terms in increasing order of marginality and including one additional term for each row of the table. The process is order-dependent for non-orthogonal designs and several different orders may be needed to appreciate the analysis fully if the non-orthogonality is severe. For an orthogonal design the process is not order-dependent provided marginality restrictions are obeyed.

To explore the effect of adding or dropping terms from a model our two functions `addterm` and `dropterm` are usually more convenient. These allow the effect of, respectively, adding or removing individual terms from a model to be assessed, where the model is defined by a fitted-model object given as the first argument. For `addterm` a second argument is required to specify the scope of the terms considered for inclusion. This may be a formula or an object defining a formula for a larger model. Terms are included or removed in such a way that the marginality principle for factor terms is obeyed; for purely quantitative regressors this has to be managed by the user.

[11] Marginal terms are sometimes removed in this way in order to calculate what are known as 'Type III sums of squares' but we have yet to see a situation where this makes compelling statistical sense. If they are needed, they can be computed by `summary.aov` in S-PLUS.

Both functions are generic and compute the change in AIC (Akaike, 1974)

$$\text{AIC} = -2\,\text{maximized log-likelihood} + 2\,\#\,\text{parameters}$$

Since the log-likelihood is defined only up to a constant depending on the data, this is also true of AIC. For a regression model with n observations, p parameters and normally-distributed errors the log-likelihood is

$$L(\beta, \sigma^2; \boldsymbol{y}) = \text{const} - \tfrac{n}{2}\log\sigma^2 - \tfrac{1}{2\sigma^2}\|\boldsymbol{y} - X\boldsymbol{\beta}\|^2$$

and on maximizing over β we have

$$L(\widehat{\boldsymbol{\beta}}, \sigma^2; \boldsymbol{y}) = \text{const} - \tfrac{n}{2}\log\sigma^2 - \tfrac{1}{2\sigma^2}\text{RSS}$$

Thus if σ^2 is *known*, we can take

$$\text{AIC} = \frac{\text{RSS}}{\sigma^2} + 2p + \text{const}$$

but if σ^2 is *unknown*,

$$L(\widehat{\boldsymbol{\beta}}, \widehat{\sigma}^2; \boldsymbol{y}) = \text{const} - \tfrac{n}{2}\log\widehat{\sigma}^2 - \tfrac{n}{2}, \qquad \widehat{\sigma}^2 = \text{RSS}/n$$

and so

$$\text{AIC} = n\log(\text{RSS}/n) + 2p + \text{const}$$

For *known* σ^2 it is conventional to use Mallows' C_p,

$$C_p = \text{RSS}/\sigma^2 + 2p - n$$

(Mallows, 1973) and in this case `addterm` and `dropterm` label their output as C_p.

For an example consider removing the four-way interaction from the complete model and assessing which three-way terms might be dropped next.

```
> quine.hi <- aov(log(Days + 2.5) ~ .^4, quine)
> quine.nxt <- update(quine.hi, . ~ . - Eth:Sex:Age:Lrn)
> dropterm(quine.nxt, test = "F")
Single term deletions
....
             Df Sum of Sq    RSS     AIC F Value   Pr(F)
<none>                    64.099 -68.184
Eth:Sex:Age  3    0.9739 65.073 -71.982  0.6077 0.61125
Eth:Sex:Lrn  1    1.5788 65.678 -66.631  2.9557 0.08816
Eth:Age:Lrn  2    2.1284 66.227 -67.415  1.9923 0.14087
Sex:Age:Lrn  2    1.4662 65.565 -68.882  1.3725 0.25743
```

Clearly dropping `Eth:Sex:Age` most reduces AIC but dropping `Eth:Sex:Lrn` would increase it. Note that only non-marginal terms are included; none are significant in a conventional F-test.

Alternatively we could start from the simplest model and consider adding terms to reduce C_p; in this case the choice of scale parameter is important, since the simple-minded choice is inflated and may over-penalize complex models.

```
> quine.lo <- aov(log(Days+2.5) ~ 1, quine)
> addterm(quine.lo, quine.hi, test = "F")
Single term additions
....
       Df Sum of Sq    RSS     AIC F Value   Pr(F)
<none>              106.79 -43.664
Eth    1    10.682  96.11 -57.052  16.006 0.00010
Sex    1     0.597 106.19 -42.483   0.809 0.36981
Age    3     4.747 102.04 -44.303   2.202 0.09048
Lrn    1     0.004 106.78 -41.670   0.006 0.93921
```

It appears that only `Eth` and `Age` might be useful, although in fact all factors are needed since some higher-way interactions lead to large decreases in the residual sum of squares.

Automated model selection

Our function `stepAIC` may be used to automate the process of stepwise selection. It requires a fitted model to define the starting process (one somewhere near the final model is probably advantageous), a list of two formulae defining the upper (most complex) and lower (most simple) models for the process to consider and a scale estimate. If a large model is selected as the starting point, the `scope` and `scale` arguments have generally reasonable defaults, but for a small model where the process is probably to be one of adding terms, they will usually need both to be supplied. (A further argument, `direction`, may be used to specify whether the process should only add terms, only remove terms, or do either as needed.)

By default the function produces a verbose account of the steps it takes which we turn off here for reasons of space, but which the user will often care to note. The `anova` component of the result shows the sequence of steps taken and the reduction in AIC or C_p achieved.

```
> quine.stp <- stepAIC(quine.nxt,
    scope = list(upper = ~Eth*Sex*Age*Lrn, lower = ~1),
    trace = F)
> quine.stp$anova
....
          Step Df Deviance Resid. Df Resid. Dev     AIC
1                                 120     64.099 -68.184
2 - Eth:Sex:Age  3   0.9739       123     65.073 -71.982
3 - Sex:Age:Lrn  2   1.5268       125     66.600 -72.597
```

At this stage we might want to look further at the final model from a significance point of view. The result of `stepAIC` has the same class as its starting point argument, so in this case `dropterm` may be used to check each remaining non-marginal term for significance.

```
> dropterm(quine.stp, test = "F")
           Df Sum of Sq    RSS     AIC F Value   Pr(F)
<none>                 66.600 -72.597
Sex:Age    3   10.796 77.396 -56.663  6.7542 0.00029
Eth:Sex:Lrn 1    3.032 69.632 -68.096  5.6916 0.01855
Eth:Age:Lrn 2    2.096 68.696 -72.072  1.9670 0.14418
```

The term `Eth:Age:Lrn` is not significant at the conventional 5% significance level. This suggests, correctly, that selecting terms on the basis of AIC can be somewhat permissive in its choice of terms, being roughly equivalent to choosing an F-cutoff of 2. We can proceed manually

```
> quine.3 <- update(quine.stp, . ~ . - Eth:Age:Lrn)
> dropterm(quine.3, test = "F")
           Df Sum of Sq    RSS     AIC F Value   Pr(F)
<none>                  68.696 -72.072
Eth:Age     3     3.031 71.727 -71.768  1.8679 0.13833
Sex:Age     3    11.427 80.123 -55.607  7.0419 0.00020
Age:Lrn     2     2.815 71.511 -70.209  2.6020 0.07807
Eth:Sex:Lrn 1     4.696 73.391 -64.419  8.6809 0.00383
> quine.4 <- update(quine.3, . ~ . - Eth:Age)
> dropterm(quine.4, test = "F")
           Df Sum of Sq    RSS     AIC F Value    Pr(F)
<none>                  71.727 -71.768
Sex:Age     3    11.566 83.292 -55.942   6.987 0.000215
Age:Lrn     2     2.912 74.639 -69.959   2.639 0.075279
Eth:Sex:Lrn 1     6.818 78.545 -60.511  12.357 0.000605
> quine.5 <- update(quine.4, . ~ . - Age:Lrn)
> dropterm(quine.5, test = "F")

Model:
log(Days + 2.5) ~ Eth + Sex + Age + Lrn + Eth:Sex + Eth:Lrn
                  + Sex:Age + Sex:Lrn + Eth:Sex:Lrn
           Df Sum of Sq    RSS     AIC F Value     Pr(F)
<none>                  74.639 -69.959
Sex:Age     3    9.9002 84.539 -57.774   5.836 0.0008944
Eth:Sex:Lrn 1    6.2988 80.937 -60.130  11.140 0.0010982
```

or by setting `k = 4` in `stepAIC`. We obtain a model equivalent to `Sex/(Age + Eth*Lrn)` which is the same as that found by Aitkin (1978), apart from his choice of $\alpha = 1$ for the displacement constant. (However, when we consider a negative binomial model for the same data in Section 7.4 a more extensive model seems to be needed.)

Standard diagnostic checks on the residuals from our final fitted model show no strong evidence of any failure of the assumptions, as the reader may wish to verify.

It can also be verified that had we started from a very simple model and worked forwards we would have stopped much sooner with a much simpler model, even using the same scale estimate. This is because the major reductions in the residual sum of squares only occur when the third-order interaction `Eth:Sex:Lrn` is included.

There are other tools in S-PLUS for model selection called `stepwise` and `leaps`,[12] but these only apply for quantitative regressors. There is also no possibility of ensuring that marginality restrictions are obeyed.

[12] There are equivalent functions in the R package `leaps` on CRAN.

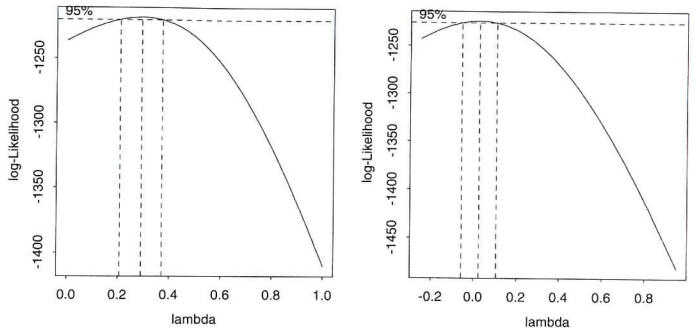

Figure 6.7: Box–Cox plots for the `cpus` data. Left: original regressors. Right: discretized regressors.

6.9 Predicting Computer Performance

Ein-Dor and Feldmesser (1987) studied data on the performance on a benchmark of a mix of minicomputers and mainframes. The measure was normalized relative to an IBM 370/158–3. There were six machine characteristics: the cycle time (nanoseconds), the cache size (Kb), the main memory size (Kb) and number of channels. (For the latter two there are minimum and maximum possible values; what the actual machine tested had is unspecified.) The original paper gave a linear regression for the square root of performance, but log scale looks more intuitive.

We can consider the Box–Cox family of transformations, Figure 6.7.

```
boxcox(perf ~ syct + mmin + mmax + cach + chmin + chmax,
       data = cpus, lambda = seq(0, 1, 0.1))
```

which tends to suggest a power of around 0.3 (and excludes both 0 and 0.5 from its 95% confidence interval). However, this does not allow for the regressors to be transformed, and many of them would be most naturally expressed on log scale. One way to allow the variables to be transformed is to discretize them; we show a more sophisticated approach in Section 8.8.

```
cpus1 <- cpus
attach(cpus)
for(v in names(cpus)[2:7])
  cpus1[[v]] <- cut(cpus[[v]], unique(quantile(cpus[[v]])),
                    include.lowest = T)
detach()
boxcox(perf ~ syct + mmin + mmax + cach + chmin + chmax,
       data = cpus1, lambda = seq(-0.25, 1, 0.1))
```

which does give a confidence interval including zero.

The purpose of this study is to predict computer performance. We randomly select 100 examples for fitting the models and test the performance on the remaining 109 examples.