Discussion Notes 3 – Stepwise Regression and Model Selection

Stepwise Regression

There are many different commands for doing stepwise regression. Here we introduce the command "step". There are many arguments in this command, and we will learn some of the important arguments here. In general, the syntax is as follow:

```
> step(model, scope, data, direction, scale, k, trace)
```

If we just type the above command, R shows every model it goes over, with the degree of freedom, SS, RSS and requested criterion of each predictor.

First, we consider different criteria. There are three major criteria in the stepwise regression: Cp, AIC and BIC. Here are their implementations: Given data, for any models and step directions:

For Cp:

```
> step(model, data, direction, scale=(summary(full)$sigma)^2, trace)
```

For AIC:

```
> step(model, data, direction, k=2, trace) # k=2 is by default
```

For BIC:

```
> step(model, data, direction, k=log(nrow(data)), trace)
```

where "`full`" is the full model.

Then we consider the direction of the stepwise regression. In general, it is either forward, backward or both directions. Here are their implementations: Given data, for any criterion:

For FORWARD stepwise:

```
> step(null, scope=list(lower=null, upper=full), data,
    direction="forward", scale, k, trace)
```

For BACKWARD stepwise:

```
> step(full, data, direction="backward", scale, k, trace)
```

For stepwise in BOTH direction:

```
> step(full, data, direction="both", scale, k, trace)
```

where "`null`" is the null model and "`full`" is the full model. Therefore, before we do the stepwise regression, it is good to generate a null model and a full model.

```
> null=lm(y~1, dat)
> full=lm(y~., dat)
```

Up to here, we learn how to write the command for specific stepwise direction and specific criteria respectively. If we combine the above two techniques, we may get whatever results we want. For example, if we want to perform a BACKWARD stepwise regression under the criteria of BIC, the command will be:

```
> step(full, data, direction="backward", k=log(nrow(data)))
```

If we want to perform a stepwise regression in BOTH direction under the criteria of AIC, the command will be:

```
> step(full, data, direction="both", k=2)
```

There are some other arguments that we may interest in. First, about the argument "`trace`", if we do not specify it (i.e. using default value), it will return every model it goes over together with the coefficients of the final model. However, if you just want to know the information of the final model, we may set "`trace=0`".

Another argument is the "`scope`". It gives the range of the predictors we want to include in the stepwise procedure. For example, we set "`scope=list(lower=null, upper=full)`"in the forward stepwise. This setting means we start from the null model and add predictors one-by-one up to the largest possible model, which is the full model. "`lower`" specifies the minimal model the stepwise procedure may go over, and "`upper`" specifies the maximal model the stepwise procedure may go over.

Finally, if we save the output of "`step`" into a variable, it will be a linear model type variable. Standard method of extracting information from a linear model can be used in this variable. For example, `summary(model)` returns the summary statistics of the final model, and so on.

Model Selection Criteria

If the number of predictors is small, we may also be interested in looking at the values of different criteria for every sub-model. In general, we are looking at the following 5 criteria: Adjusted R^2, Cp, AIC, BIC and PRESS.

We know how to obtain the adjusted R^2 given a linear model M.

```
> rsq.a = summary(M)$adj.r.squared
```

How about others? First, for Cp, by definition, (p.218 equation 10.9)

$$Cp = \frac{RSS}{\hat{\sigma}^2} + 2p - n$$

where p = number of predictors in the model and n = number of data.

Therefore, if we want to compute Cp in R, we may try

```
> rss = sum((N$residual)^2)
> sigma.full=summary(N)$sigma
> Cp = rss/sigma.full^2 - nrow(M$model) + 2*ncol(M$model)
```

For AIC and BIC, we consider a command "`extractAIC`". The syntax is:

```
> extractAIC(model, k) # k=2 by default
```

If we set the argument "`k=2`", it outputs the traditional AIC. If we set the argument "`k=log(nrow(M$model))`", it outputs the BIC. In general, R follows this formula:

$$AIC = -2\log L + k(edf)$$

where "L" is the likelihood, "edf" is the effective degree of freedom and "k" is the multiple in the penalty. For more details about the formula, please refer to the p.217 – 218 equation (10.7) and (10.8).

Finally, for PRESS (predicted residual sum of square), by definition, (p.220 equation 10.10)

$$PRESS = \sum_{i=1}^{n}\left(\frac{e_i}{1-h_{ii}}\right)^2$$

where e_i is the residual of data I and h_{ii} is the leverage for data i. Therefore, if we want to compute PRESS in R, we may try

```
> hii=ls.diag(M)$hat
> PRESS=sum((M$residual/(1-hii))^2)
```

In professor's note, the function `lm.info` has included all of the above criteria. For more details about the command "`step`", the following pages are the documentation of this command.

```
step                package:stats                R Documentation
Choose a model by AIC in a Stepwise Algorithm
Description:
    Select a formula-based model by AIC.
Usage:
    step(object, scope, scale = 0,
         direction = c("both", "backward", "forward"),
         trace = 1, keep = NULL, steps = 1000, k = 2, ...)
```

```
Arguments:
  object: an object representing a model of an appropriate class
        (mainly '"lm"' and '"glm"'). This is used as the initial
        model in the stepwise search.
   scope: defines the range of models examined in the stepwise search.
        This should be either a single formula, or a list containing
        components 'upper' and 'lower', both formulae.  See the
        details for how to specify the formulae and how they are
        used.
   scale: used in the definition of the AIC statistic for selecting the
        models, currently only for 'lm', 'aov' and 'glm' models.
direction: the mode of stepwise search, can be one of '"both"',
        '"backward"', or '"forward"', with a default of '"both"'.  If
        the 'scope' argument is missing the default for 'direction'
        is '"backward"'.
   trace: if positive, information is printed during the running of
        'step'. Larger values may give more detailed information.
    keep: a filter function whose input is a fitted model object and
        the  associated 'AIC' statistic, and whose output is
        arbitrary.  Typically 'keep' will select a subset of the
        components of  the object and return them. The default is not
        to keep anything.
   steps: the maximum number of steps to be considered.  The default is
        1000 (essentially as many as required).  It is typically used
        to stop the process early.
      k: the multiple of the number of degrees of freedom used for the
        penalty. Only 'k = 2' gives the genuine AIC: 'k = log(n)' is
        sometimes referred to as BIC or SBC.
    ...: any additional arguments to 'extractAIC'.
Details:
    'step' uses 'add1' and 'drop1' repeatedly; it will work for any
    method for which they work, and that is determined by having a
    valid method for 'extractAIC'. When the additive constant can be
    chosen so that AIC is equal to Mallows' Cp, this is done and the
    tables are labelled appropriately.
    The set of models searched is determined by the 'scope' argument.
    The right-hand-side of its 'lower' component is always included in
    the model, and right-hand-side of the model is included in the
```

'upper' component. If 'scope' is a single formula, it specifes the 'upper' component, and the 'lower' model is empty. If 'scope' is missing, the initial model is used as the 'upper' model.

Models specified by 'scope' can be templates to update 'object' as used by 'update.formula'. So using '.' in a 'scope' formula means 'what is already there', with '.^2' indicating all interactions of existing terms.

There is a potential problem in using 'glm' fits with a variable 'scale', as in that case the deviance is not simply related to the maximized log-likelihood. The '"glm"' method for function 'extractAIC' makes the appropriate adjustment for a 'gaussian' family, but may need to be amended for other cases. (The 'binomial' and 'poisson' families have fixed 'scale' by default and do not correspond to a particular maximum-likelihood problem for variable 'scale'.)

Value:

the stepwise-selected model is returned, with up to two additional components. There is an '"anova"' component corresponding to the steps taken in the search, as well as a '"keep"' component if the 'keep=' argument was supplied in the call. The '"Resid. Dev"' column of the analysis of deviance table refers to a constant minus twice the maximized log likelihood: it will be a deviance only in cases where a saturated model is well-defined (thus excluding 'lm', 'aov' and 'survreg' fits, for example).

Warning:

The model fitting must apply the models to the same dataset. This may be a problem if there are missing values and R's default of 'na.action = na.omit' is used. We suggest you remove the missing values first.

Note:

This function differs considerably from the function in S, which uses a number of approximations and does not compute the correct AIC.

This is a minimal implementation. Use 'stepAIC' in package 'MASS' for a wider range of object classes.