A User's Guide to the SAS Implementation of the Phylogenetic Regression, version 0.7

Alan Grafen (alan.grafen@sjc.ox.ac.uk)
http://users.ox.ac.uk/~grafen/phylo/

1. Introduction

The program implements the phylogenetic regression (A. Grafen, 1989, Philosophical Transactions of the Royal Society B, **326**:119-157 - henceforth "the source paper"), a statistical technique which allows the hypothesis testing facilities of general linear models to be applied validly to cross-species data. It offers substantially the same facilities as the GLIM implementation released in 1990, and will eventually replace it. Most of this manual assumes the reader is not familiar with the GLIM implementation, but there is a section below entitled 'Introduction for users of PHYLO.GLM'.

The user will benefit from an acquaintance with SAS statistical software in general and with PROC GLM in particular, although with determination a SAS novice might manage on the basis of this manual alone. One way to gain experience with SAS in a more guided fashion is to follow the textbook *Modern Statistics for the Life Sciences* (A. Grafen and R. Hails, 2002, Oxford University Press) using the online SAS supplement available at http://www.oup.co.uk/best.textbooks/biology/grafenhails/ . This will also provide an introduction to General Linear Models, and how to use model formulae, which will also be found useful in employing the software described here. More basic operations such as importing data are explained too.

Sections 2 to 7 set out to explain in order how to carry out phylogenetic regressions from scratch, and section 8 makes some fairly obvious points about using the results in publication. As well as learning from this instructional mode, you may find it useful to read the two worked examples in section 9 (or indeed run them on your computer). Section 10 is for reference, and section 11 gives the version history.

1.1 Introduction for users of PHYLO.GLM

It was always foreseen that an implementation in one of the major packages would be preferable to GLIM. Not only do more biologists use them, but the syntax of GLIM is very restrictive and made the user interface less than simple. This new SAS version makes it much easier to perform analyses, and also offers some new features. Because I no longer have access to the current version of GLIM, this SAS version will become the only supported version. The GLIM version will remain available, but is unlikely to be updated. The update in GLIM itself from 3.77 to 4 included a poorly documented failure of backward compatibility which necessitated changes to PHYLO.GLM – similar changes in GLIM in future are unlikely to be followed by the necessary changes to PHYLO.GLM.

The new features include (i) straightforward use of model formulae for continuous and categorical variables and interactions (ii) ability to use newick-style phylogenies (iii) no artificial limits on numbers of levels of categorical variables or interactions (iv) no artificial memory limits (v) the p-value is now given with the F-ratio (vi) categorical variables can have character levels (vii) coefficient estimates are now provided so as to allow the calculation of fitted values in the same way as for any GLM.

There is one small point on which my statistical opinion has changed since 1990, which concerns the extra degree of freedom to be subtracted when ρ is fitted. I no longer recommend that the fitting of ρ should require any alterations to the degrees of freedom. My new view is based on the following reasoning. ρ is fitted to the control variables only in the long regression, and there might be a case for subtracting a degree of freedom from the residual degrees of freedom in this model. However, the test is performed with fixed ρ, and by partitioning the error sum of squares of the model in which ρ is fitted. The fitting of ρ does not seem to me to affect the numerator any more or less than the denominator of the resulting F-ratio. Thus although the facility still exists to subtract a degree of freedom (parameter 'addDF' to %phyreg), my advice is to maintain the default value of zero. Users of PHYLO.GLM can achieve the same outcome by setting sc__(10) to zero.

2. Installation

The files phyprog.sas, phyexam.sas and phyman.pdf are available on http://users.ox.ac.uk/~grafen/phylo/. The PDF file contains this manual. Installation employs the SAS files as follows.

The steps you need to take just once are as follows.

(1) Create a SAS data-library called PHYLO. You can do this by selecting the Explorer window from within SAS and choosing File > New > Library. Specify a directory a disk to contain the SAS library, and check box to load at startup.

(2) Download the phyprog.sas file and know its location on your hard disk. The next instruction assumes it is at 'c:\phyprog.sas', but other locations will do equally well. (Though note that SAS doesn't easily handle directory names with special characters in, such as '.)

(3) Import the program. Move to an Editor Window in SAS, type or copy-and-paste the instruction

```
PROC CIMPORT L=Phylo file='c:\phyprog.sas'; quit;
```

Then select the whole instruction and choose Run > Submit

(4) Optional downloading of example library. Create a SAS data-library called PHYLOEXS, download the file phyexam.sas, and submit the instruction (assuming you have downloaded to 'c:\phyexam.sas')

```
PROC CIMPORT L=Phyloexs file='c:\phyexam.sas'; quit;
```

(5) Install IML modules. Using the Explorer window, find the library PHYLO, and open catalog 'Program', in which you should see two items. These are 'Install' and 'LoadMacros'. For initial installation, we use 'Install'. On a PC, right-click on 'Install', and choose Run from the pop-up menu, and you're done. The following

longer method should work on all systems. Open 'Install', which will appear as a textfile. Then choose Run > Submit, and SAS will run the file (it will create a catalog phylo.zkeepmacs, but the user doesn't need to know this). Now close the textfile to ensure you don't edit it by mistake!

(6) That's it! If you want to save space, you can delete the 'Install' entry in the catalog.

You cannot yet actually run analyses. What remains is that in each SAS session, you must carry out the following step so that the macros become available for the rest of the session.

(1) Load the macros. Using Explorer, find the library PHYLO and the catalog 'Program'. Open the catalog to find the item 'LoadMacros'. On a PC, simply right-click on 'LoadMacros' and choose Run from the pop-up menu. On all systems, you should be able to open 'LoadMacros' into a textfile, and then choose Run > Submit to run the file. Now close the textfile to make sure you don't edit it by mistake!

For the rest of the session the facilities of the phylogenetic regression are available to you. These are embodied in two macros, %newick and %phyreg, and their use is described in the following sections.

Section 9 explains how the examples library can be used.

3. Preparing to perform an analysis

A user must have a dataset in which each species is a row, and the variables of interest are columns. The dataset needs to be imported into a SAS dataset, and this can conveniently be done through an Excel worksheet. Some variables will be treated as categorical variables – these can be either text or numeric. Others, including the response variable, will be treated as continuous, and these must be numeric. At a suitable moment, the user must be able to specify which variables are to be treated as categorical.

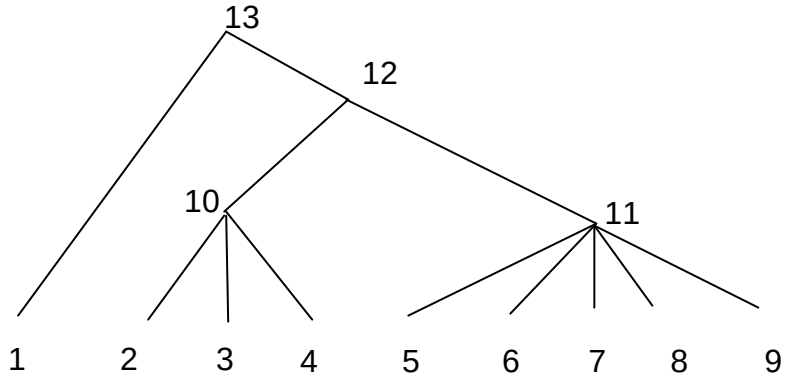
But the main preparation involves the phylogeny, both its topology and its branch lengths. We now consider in succeeding subsections how to specify the topology of the working phylogeny, how to specify the branch lengths, and then how to choose a topology in the first place.

3.1 Specifying the topology of the working phylogeny

A working phylogeny can be provided in one of three ways, which are considered in turn. We begin with the least familiar to the user, but the most natural to the program, and then move on to more familiar alternatives. After looking at the three methods, we will turn to what is meant by a *working phylogeny*.

The single variable method. One reason for explaining this method is that users may wish to use it. Another is that the software will convert a phylogeny provided by either of the two remaining methods into this form, and for some purposes it may be helpful in interpreting output to understand this method. In particular, higher nodes are represented by numbers, and these are the numbers in the internal representation of the phylogeny.

To implement this method, start with a drawing of the working phylogeny. The first task is to number each node using consecutive integers. Each species node should be numbered according to its position in the dataset. The higher nodes should be numbered in accordance with the principle that the number of each node must be lower than the number of its parent node. The root of the tree will therefore have the highest number. There are many ways of numbering the nodes compatibly with these principles, and all are equally good. Here is an example of a phylogeny numbered in line with these principles.



The second task is to write down a list of numbers which define the phylogeny. It is most convenient to write two lists in adjacent columns. The left hand column contains the integers 1, 2, and so on up to one less than the number of the root. The right hand column contains the number of the parent node of the node denoted on the left. The required variable is the right hand column, which now contains a full description of the working phylogeny. This method can represent any working phylogeny. For the example, the phylogeny drawn above looks like

Node	Parent	Comment
1	13	This species is connected directly to the root
2	10	These three species
3	10	belong to a genus
4	10	with node number 10
5	11	These five
6	11	species belong
7	11	to a genus
8	11	with node
9	11	number 11
10	12	This genus (“10”) belongs to node 12
11	12	So does this genus (“11”)
12	13	This node, containing genera 10 and 11, belongs to the root
		The root has no parent, and so has no entry here

I know from experience that this process is quite error-prone, even if in this mini-example it seems beguilingly simple. The result should be checked carefully.

The left hand column is simply the integers in order, and so carries no information. The program requires a SAS dataset with only a single variable, containing the second of these columns.

The taxonomic variables method. This method uses variables in the dataset to represent taxa to which each species belongs. Thus for example we might have a column GE representing the genus, FA for the family and OR for the order. The user need only have ready to use at an appropriate moment the list of taxonomic variables beginning with the lowest and ending with the highest (so in our example the list would be "GE FA OR").

Note three points of detail.

(1) The columns may be numeric or character, but *all must be of the same type.*

(2) The tiniest spelling difference will be counted as a distinct level – so check the values very carefully.

(3) The values of one taxonomic variable need only be unique within the set of values that are equal for all higher variables. For example, using numeric codes, genera could be numbered from 1 within each family, and families could be numbered from 1 within each order.

The Newick method. One of the now standard methods of describing a phylogeny is the Newick method, for example

```
(((Struthio_camelus:17.9, (Apteryx_australis:9.5,
Dromaius_novaehollandiae:9.5):8.4):8., ((Colinus_virginianus:15.1,
(((Tetrao_tetrix:5.82, (Lagopus_lagopus:4.47,
Lagopus_mutus:4.47):1.35):4.86, Meleagris_gallopavo:10.68):0.42,
(Phasianus_colchicus:10.3, (Coturnix_coturnix:9.9,
(Alectoris_rufa:7.2, Perdix_perdix:7.2):2.7):0.4):0.8):4.):7.8,
((Branta_canadensis:2.5, ((Anser_anser:0.05,
Anser_anser_dom.:0.05):1.95, Anser_indicus:2.):0.5):4.2,
(Somateria_mollissima:3.7, (Anas_platyrhynchos:0.05,
Anas_platyrhynchos_dom.:0.05):3.65):3.):16.2):3.):2.1,
(Ceryle_rudis:25., (Psittacula_krameri:23.1,
(((Streptopelia_decaocto:0.9, Streptopelia_risoria:0.9):19.9,
((Chlamydotis_macqueenii:0.4, Chlamydotis_undulata:0.4):19.7,
((((Phalaropus_lobatus:1., Phalaropus_tricolor:1.):4.5,
(Actitis_macularia:5.2, Calidris_pusilla:5.2):0.3):10.1,
(Chionis_minor:12.8, Larus_occidentalis:12.8):2.8):3.1,
((Falco_tinnunculus:15.2, Parabuteo_unicinctus:15.2):1.2,
(Phaeton_rubricauda:14., ((((Sula_dactylatra:2.8, Sula_sula:2.8):0.5,
Sula_nebouxii:3.3):8.8, Phalacrocorax_capensis:12.1):1.2,
((((Eudyptes_chrysolophus:4., Megadyptes_antipodes:4.):1.,
(Pygoscelis_adeliae:1., Pygoscelis_papua:1.):4.):0.8,
(Aptenodytes_forsteri:1., Aptenodytes_patagonicus:1.):4.8):4.6,
(Diomedea_chrysostoma:2.2, (Diomedea_exulans:0.5,
Diomedea_melanophris:0.5):1.7):8.2):2.9):0.7):2.4):2.3):1.4):0.7):0.8
```

```
, (Hylophylax_naevioides:19.7, ((Ptilonorhynchus_violaceus:11.7,
(Malurus_cyaneus:11.4, (Lanius_collurio:9.1, (Corvus_frugilegus:3.3,
((Aphelocoma_coerulescens:0.1, Aphelocoma_californica:0.1):0.9,
Aphelocoma_ultramarina:1.):2.3):5.8):2.3):0.3):1.1,
((Cinclus_pallasii:9.7, ((Turdus_merula:8.8, (Saxicola_torquata:7.,
Ficedula_hypoleuca:7.):1.8):0.3, ((Lamprotornis_chalybaeus:3.7,
Sturnus_vulgaris:3.7):2., Toxostoma_curvirostre:5.7):3.4):0.6):2.,
((Campylorhynchus_brunneicapillus:10.8, (Acrocephalus_scirpaceus:8.9,
Cettia_diphone:8.9):1.9):0.3, (((Passer_domesticus:1.9,
Passer_rufocinctus:1.9):6.6, (Lonchura_striata:6.9,
(Plocepasser_mahali:4.8, (Ploceus_reichenowi:0.7,
Ploceus_philippensis:0.7):4.1):2.1):1.6):1.5,
(Carpodacus_mexicanus:6.9, (((Agelaius_phoeniceus:0.56,
Molothrus_ater:0.56):4.52, (((Melospiza_melodia:2.54,
Zonotrichia_leucophrys:2.54):0.3, Amphispiza_bilineata:2.84):0.3,
Spizella_arborea:3.14):1.94):1.52,
Calcarius_lapponicus:6.6):0.3):3.1):1.1):0.6):1.1):6.9):1.9):1.5):1.9
):3.)
```

The bracketed structure represents phylogeny in an obvious way, and the colon introduces extra data about the immediately preceding node. We will see shortly that in this case the extra data represents the lengths of path segments in the phylogeny.

The user needs to be able, at a suitable juncture either to (i) be able to copy-and-paste this phylogeny into a SAS program window or (ii) be able to provide the name of a textfile containing just this phylogeny. There should not be a semi-colon in the phylogeny. Carriage returns and spaces are ignored, and so can be used freely in formatting. The final bracket, representing the root, is permitted to have additional data following it, although it does not in the example.

3.2 Specifying path segment lengths of the phylogeny

The path segment lengths define the correlation structure of the error in the statistical model, and so lie at the heart of the whole enterprise of controlling for phylogeny. There are three methods of specifying path segment lengths, which are described in turn.

The default "Figure 2" method The name derives from Figure 2 in the source paper, and is the method that will be used if you take no special action. Each node is assigned a height equal to one fewer than the number of species among the descendants of the node. Thus each species node has a height of zero, and the root has a height of $n-1$ if there are n species. The length of a path segment between two nodes is calculated simply as the difference between the heights of the nodes at either end.

Taxonomic levels method. This method is available only if the phylogeny was created by the taxonomic variables method. If there were v taxonomic variables, then there are $v+2$ taxonomic levels. The extra two are the root, at the top of the tree, and the species level at the bottom. The method allocates the species level a height of zero, and so there are $v+1$ remaining heights to specify. You do this by placing $v+1$ values in a variable in a SAS dataset. The values should be positive, as they must be strictly above the species height of zero, and they should be increasing, as the heights are read

from the lowest level to the highest level. Multiplying the heights by a positive constant has no consequence for the analysis.

Fully general method. This method allows you to specify a different height for every single node. To do this you must provide a SAS dataset with a single variable, which contains in its *i*th row the height of the *i*th node. If you have used the single column method of entering the phylogeny, then you will already know the numbers that code for each of the higher nodes.

If you have used the taxonomic levels method, you can obtain the phylogeny as follows. Run an analysis using some other heights method, and then the SAS dataset phylo.zphy will contain a single variable with the names of the higher nodes. Using this, you can write the numbers of the nodes on a drawing of the phylogeny, and construct the required list of heights.

If you used the Newick-style method, then the most straightforward way to specify heights is to include them with the phylogeny as the extra data. Section 4 on using %newick explains how to handle both the possibility of specifying path segment lengths, as found in the phylogeny shown above, and the possibility of specifying a height for each node. In this second case, the root too would need to have additional data.

3.3 Choosing the topology and path segment lengths

Choice of topology is mainly a matter of deciding how certain you are about the order of splitting. Where there is complete certainty, then use the full binary phylogeny. Where there is complete uncertainty about the order of splitting of a group with four daughters, then assign a polytomy with a four-way split. There will obviously be intermediate cases.

The phylogeny we choose to use for the statistical analysis is the *working phylogeny*, and will frequently not be our best guess at the true phylogeny because it contains polytomies representing insufficient certainty to rely on. The main tension in principle is that on the one hand each higher node counts as a datapoint, and the more datapoints the more powerful the analysis; but on the other, if we make false assumptions about order of splitting, then we have a more powerful analysis on a false basis, and our results are invalid. We can check whether results are sensitive to particular topological choices by repeating the analysis with a more conservative and a less conservative topology. If the results are the same, then we needn't worry too much about the details of the topology.

What about path segment lengths? The phylogenetic regression will give different F-ratios depending on how you specify the path segment lengths. This may seem undesirable at first sight. The reason is that in reducing all the data to one number, decisions have to be made about how to combine data from different parts of the tree. No method can avoid this arbitrariness. But comparative data is clearly worth analyzing, and so this arbitrariness has to be accepted as one extra source of uncertainty, along with the unreliability of data, the fact that nature may have played a joke on you, the usually tenuous connection between the theory of interest and the

measurements actually to hand, and so on. Indeed, it is exactly the same kind of uncertainty as arises in ordinary regressions, in which there is an arbitrary choice to be made as to how to weight the datapoints against each other. Most people do not even realize there is a choice to be made, but use an unweighted regression because that's what the package offers by default, a choice that usually has no special justification. I am not recommending you to the same blindness, only arguing that comparative analyses suffer this fundamental problem just as other kinds of analysis do - this besetting sin, at least, is not unique.

The first point is that varying weights moderates will not make very much difference to the F-ratio. Second, the major dimension of variation is whether much weight should go near the root or near the species tips - and this dimension is automatically taken care of by the fitting of a parameter called ρ. Third, multiplying all the heights by a positive constant makes no difference, because the program automatically scales them so the root has a height of one. Fourth, adding a constant does make a difference because a different family of path segment lengths will be generated by varying ρ, even though the member with ρ=1 will be common to them all. The fitting of ρ is explained in the following section.

A sensible course of action would then seem to be as follows. If the taxonomic variables method of specifying the phylogeny is used, then use the taxonomic levels method. If dates of divergence for these levels are known roughly, then use those dates for the heights of the levels. If not, then use 1, 2, 3 …. If the single variable method of specifying the phylogeny is used, then use the default "Figure 2" method.

These recommendations are for normal use, where the phylogeny is of no interest in itself, and the focus of interest is on the interrelationship of the variables. If there is reason to think that the phylogeny has a strong effect, or that there has been much faster evolution in one part of the tree than in another, then the program provides facilities for exploration. The fully general method allows each node height to be specified separately.

If there is serious disagreement between different phylogenies, then this means the conclusions about the interrelationships of variables are seriously in doubt. This is likely to arise, for example, when some parts of the phylogeny show a strong positive relationship between two variables, and other parts show a strong negative relationship. By choosing how to weight these parts, one can engineer either a net positive, or negative, or zero effect. In such a case, probably there is a relationship, but it may be non-linear, or it may involve an interaction with another variable. One caution should be given. Having found any significant relationship, it is almost certainly possible to find by systematic exploration some set of path segment lengths that removes it. The mere existence of such a set of lengths is no cause for concern. Finding such lengths by exploration is the logical equivalent of correlating hundreds of x-variables with y, picking the most highly significant, and claiming to have found a significant relationship. Although finding the lengths is a way of diminishing significance, rather than enhancing it, it is similarly cheating because of the data-dependent choices made. Serious doubt should be caused by phylogeny only if pre-specified and biologically reasonable sets of path segment lengths give substantially different answers.

It is worth making one final point about these lengths. One apparently reasonable point of view is that heights of nodes should be made proportional to the time before the present at which divergence occurred, or perhaps to some measure of the amount of neutral selection that has gone on. A strong implication of this kind of view is that the path segments should always be the same for a given phylogeny, no matter which variables are being analysed. The more flexible approach suggested here would expect different path segment lengths to be appropriate for different datasets, and even for different models in the same dataset. In most datasets some characters will vary more nearer the species level, and others vary more at higher levels; if we add such a variable to the model and it accounts for a substantial amount of the variability, it is quite likely that the distribution of error across the tree will change. The attitude taken in this package is that we must treat the path segment lengths pragmatically, and the fitting of ρ is provided in that spirit. The view of the error expounded in the source paper is that the error arises through relevant but omitted variables, and not as a result of neutral evolution; this view of the error logically justifies the more pragmatic approach.

3.4 Fitting of ρ

This section explains the fitting of the parameter ρ. A fuller account is given in the source paper. The heights of nodes, whether specified as heights or whether derived from the default "Figure 2" method, are used to construct path segment lengths for each node-to-node segment in the phylogeny, and these represent an assumption about the error variance associated with that segment.

A major dimension in which different sets of path segment lengths vary is whether species are fairly independent and have only weak phylogenetic associations (in which case the lengths of segments attached to species would be long, and the segments between the higher nodes would be short), or whether there are very strong phylogenetic effects and most of the variation occurs at high phylogenetic levels (when the reverse pattern would apply).

The object in fitting ρ is to allow the data to decide where most of the error variation occurs. The first stage is to scale heights so they lie between 0 (for the species) and 1 (for the root), but are still in proportion to those specified by the user. But before calculating the path segment lengths, we raise each height to the power ρ. This leaves 0 at 0 and 1 at 1. If ρ is less than one, then all the other heights are increased, which lengthens the segments near the species and shortens the segments near the root. If ρ is greater than one, all the other heights are decreased, which shortens the segments near the species and lengthens the segments near the root. Thus the family of sets of heights parametrised by ρ allows any strength of phylogenetic dependence from no dependence to very high dependence.

Anti-phylogenetic correlations, in which close nodes are more different than expected by chance, are not permitted by the general structure. This would show up when the program tried to reduce ρ to zero. A warning that the minimum value of ρ has been reached is given in such a case.

3.5 Missing values

For y-variables and variables in the model formulae for control and test, SAS missing values are permitted and are handled automatically. The 'opdf=1' option will give information about how many species are omitted because of missing values.

Missing values are not permitted elsewhere. Thus no missing values are permitted in the taxonomic variables, the single variable defining the topology of the tree, or the variables defining heights.

4. Using newick-style phylogenies

This section is relevant to users who are using a newick-style phylogeny for information on the topology and/or path segment. The macro %newick is included in the package, and is made available when LoadMacros is run. It takes as input either a string with the newick-style phylogeny, specified by for example 'phylstr=((a:3.4,b:3.4):1.2,c:4.6)' or the name of a textfile which contains only the newick-style phylogeny, specified by for example 'phylfile="c:\myphylogeny.txt"'. Notice that the string should not be in quotation marks (and should not contain them), but that the filename should be in quotation marks.

The simplest output is the SAS dataset containing the single variable representing the topology of the phylogeny, which is obtained by for example 'outphyfile=mylib.birdphy'. This is then suitable for supplying as the phylogeny to %phyreg by including 'inphydataset=mylib.birdphy'.

If the additional data are numeric and represent path segment lengths, then a SAS dataset containing the heights can be produced by for example 'outheightsfile=mylib.birdhts'. An alternative coding would be for each node to contain its height in the phylogeny, for example the number of millions of years ago at which the divergence is thought to have occurred. In this case, the root will also need to have additional data. The heights can then be produced by specifying the extra 'data' parameter in addition, for example 'data=HEIGHTS, outheightsfile=mylib.birdhts'. By whichever route it is produced, the file can then be supplied as the heights in an analysis by including 'inheightsdataset=mylib.birdhts' in the call of %phyreg.

Two further outputs are also possible, though they are of no direct use in conducting analyses. The species names along with their number and a third column for the additional data can be saved in a SAS dataset by for example 'outspecfile=mylib.specdata'; and the coding number for higher nodes along with their additional data can be saved by for example 'outhigherfile=mylib.highdata'.

Here are two example uses, assuming that the phylogeny shown in section 3.1 above is contained in the file 'c:\birdnewick.txt'.

```
%newick(phylfile= 'c:\birdnewick.txt', outphyfile=mylib.birdphy,
   outheightsfile=mylib.birdhts);
```

```
%newick(phylstr=((a:3.4,b:3.4):1.2,c:4.6),
   outphyfile=mylib.toyphy, outheightsfile= mylib.toyhts);

%newick(phylstr=((a:0,b:0):3.4,c:0):4.6, outphyfile=mylib.toyphy,
   data=HEIGHTS, outheightsfile= mylib.toyhts);
```

The second and third calls would have the same effect. Usually species would have height zero, but this is not required. If heights were found by estimated amounts of neutral evolution, rather than estimates of dates, then species need not all have the same height. Similarly, if fossil species were included, then species need not all be contemporaneous and their heights would likely be different.

Having provided a phylogeny, and perhaps heights as well, to %newick, they will have been converted into SAS datasets, and the user will be ready to carry out an analysis.

5. Performing one analysis

The command needs to made within a program window. A simple example would be to type

```
%phyreg(y=Y, control=X, test= A, class=A, dataset=range.data1,
   phydataset=range.phyl);
```

Then select it all and select Run > Submit from the menus. This will perform a phylogenetic regression in which the y-variable is Y, the continuous variable X is controlled for, and the test is of the categorical variable A. The regression data come from the SAS dataset ‘range.data1’ and the phylogeny is given in single-variable form in the SAS dataset ‘range.phyl’. The path segment lengths are derived from the default “Figure 2” method.

A complete list of the parameters of %phyreg for reference purposes is given in section 10.1. Here, we look at a few of the simpler complications of the use of %phyreg.

5.1 More complex models

The control and test parameters can be given values such that ‘control’ and ‘control test’ are both valid models in the GLM command. The macro first fits the model *control*, and fits ρ, simultaneously by maximum likelihood. Then it fits the model *"control test"* at the fitted value of ρ. The difference between these models is used to test the significance of *test* controlling for *control*. This manual will sometimes refer to the statistical test as being of the *test* variables, for simplicity, ignoring the facts that some of the contents of *test* are included in *control* and so not being tested for, and that the elements of *test* are model terms which may be variables but may also be design variables or interactions between variables.

Example values of combinations of control, test and class are

Control	Test	Class	Terms tested for
X	A	A	A
X A	X*A	A	X*A
X|A	B	A B	B
X|A	B Z	A B	B Z
X|A B	B|Z	A B	Z B*Z
A|B|C	X|Z	A B C	X Z X*Z

Adding two variables in a model is denoted by a space (unfortunately '+' is not allowed), an interaction is denoted by '*', and the shorthand for 'A B A*B' is 'A|B'. Nesting is also allowed.

5.2 Using taxonomic variables

If the topology of the phylogeny is to be specified by taxonomic variables, then they should be in the same dataset as the data variables. They must be specified in ascending taxonomic order, and there is no need for a 'species' variable. Suppose you have variables called Order Family SubFamily Genus. Then %phyreg would be called as follows

```
%phyreg(y=Y, control=X, test= A, class=A, dataset=range.data1,
   taxvars=Genus SubFamily Family Order);
```

An extra possibility of specifying heights arises here, which is to specify a height with each phylogenetic variable. See the section 5.3.

5.3 Using non-default heights

There are two ways to set heights that are not the default 'Figure 2' method. One is the fully general method, which involves specifying a height for each node, and this is done using the parameter 'heightsdataset'. If range.hts1 contains a single variable with a non-negative number for each node (each species, and each higher node including the root), then we might say, for example,

```
%phyreg(y=Y, control=X, test= A, class=A, dataset=range.data1,
   phydataset=range.phy1, heightsdataset=range.hts1);
```

The second method is possible when the taxonomic variables method has been used to determine the topology of the phylogeny, and it is to specify a height for each taxonomic variable plus in addition a height for the root. The 'heightsdataset' parameter is again used, relying on the program to use the length of the variable to infer which heights method is being used. Thus if range.hts2 contains a single variable with five increasing positive numbers, then we might say

```
%phyreg(y=Y, control=X, test= A, class=A, dataset=range.data1,
   taxvars=Genus SubFamily Family Order,
   heightsdataset=range.hts2);
```

5.4 Omitting species

It is commonly desirable for a user to carry out an analysis on only a subset of the data. The program allows a variable to be specified that indicates which species to include. This should be a variable in the dataset that contains the data for analysis, and so may require to be calculated perhaps in a DATA step. A value of 1 indicates inclusion and a value of 0 indicates exclusion – no other values should be used.

Suppose the dataset has a variable 'muchdata' that codes using 0 and 1, with a 1 indicating species with more than a certain amount of data supporting some classification. Then we could say

```
%phyreg(y=Y, control=X, test= A, class=A, spuse=muchdata,
   dataset=range.data1, phydataset=range.phy1,
   heightsdataset=range.hts1);
```

The special value _UNSET_ can be used to clear previous values and ensure all species are used (though of course missing data may still prevent some species being included in the analysis). Thus if we had previously used 'spuse=muchdata' and now wanted to ensure all species are used, we would say

```
%phyreg(y=Y, control=X, test= A, class=A, spuse=_UNSET_,
   dataset=range.data1, phydataset=range.phy1,
   heightsdataset=range.hts1);
```

5.5 Fixing ρ

By default, the value of ρ is fitted by maximum likelihood. To specify a value of ρ and prevent the fitting, the parameter 'rho' can be used, as follows:

```
%phyreg(y=Y, control=X, test= A, class=A, rho=0.7,
   dataset=range.data1, phydataset=range.phy1);
```

To return to fitting ρ, simply specify a negative value for rho, as follows

```
%phyreg(y=Y, control=X, test= A, class=A, rho=-1,
   dataset=range.data1, phydataset=range.phy1);
```

The most natural time to wish to set ρ is when we wish to assume that the path segment lengths represent the correlational structure exactly. In this case, we should specify 'rho=1', so that the heights are left unaltered by the rho-transformation. Though see section 3.3, where it is argued that this is not as logical an assumption as it might appear.

6. Performing a series of analyses

The macro %phyreg effectively remembers parameter values from one run to the next, within a session. Thus if the datasets and y-variable are not changing, you don't need to specify them after the first call. This can be helpful to save typing. It can also be

confusing if you're not sure quite what has been specified. Both these points are now illustrated.

Here is a possible sequence of macro calls to carry out the analyses in the table in section 5.1, in order.

```
%phyreg(y=Y, control=X, test=A, class=A, dataset=range.data1,
   phydataset=range.phyl);
%phyreg(control=X A, test=X.A);
%phyreg(control=X*A, test=B, class=A B);
%phyreg(test=B Z);
%phyreg(control=X*A B, test=B*Z);
%phyreg(control=A*B*C, test=X*Z, class=A B C);
```

Sometimes it may be useful to reset all the parameters to their defaults -- perhaps you aren't quite sure what state the parameter values are in after a long series of calls. In this case, include reset=YES in the parameter list. This unsets all the remembered parameter values, so you will have to include the datasets again too.

Thus to change y-variable to Y2, and make sure no lingering values are still influencing results, we could write

```
%phyreg(y=Y2, control=X, test=A, class=A, dataset=range.data1,
   phydataset=range.phyl, reset=YES);
```

It is also possible to cause all the parameter values to be included in the output by including 'opmacparm=1' when you use %phyreg. This will confirm all the parameter values, which could be useful for keeping records, checking on the current state of the parameter values, or reminding yourself what all the parameters are called.

Some parameters can be individually reset by using the special value _UNSET_ . These parameters are class, control, spuse, phydataset, heightsdataset and taxvars. In each case they revert to having no defined value. This can be useful as sometimes no value is wanted (e.g. control, class), the default action is required (e.g. spuse), or required information is being supplied another way (phydataset, taxvars, heightsdataset).

For example, suppose A is a class variable with numeric levels, and the order of the levels has a meaning. Then we can test a linear component of the categorical variable by treating A as a continuous variable. The first analysis treats A as a class variable, and the second as continuous:

```
%phyreg(y=Y, control=X, test=A, class=A, dataset=range.data1,
   phydataset=range.phyl);
%phyreg(class=_UNSET_);
```

The facilities are intended to make it easy to use %phyreg interactively, and easy to explore different models.

7. Interpreting output

Output confirming the parameter values is obtained by specifying 'opmacparm=1'. This section discusses the statistical output, which comes in three sections, which are controlled separately by the parameters opf (for F-ratio, p-value and related output), opdf (for an accounting of how the degrees of freedom of the final test are arrived at) and opparm (parameter estimates from the regression, and information about ρ). Let's look at them in turn.

7.1 Parameter estimates

The phylogenetic regression produces as its main result an F-ratio, as the primary problem with comparative data has been finding valid p-values. The values of slopes, and especially their signs, are important too. Two ways of finding a parameter estimate are possible in the program, one the fully correct method, and the other a quicker approximation. The quick approximation is to look at the parameter estimates of the long regression with the control and test model formulae. This method will usually be a good approximation to the fully correct method. The reason for its slight imperfection is that the value of ρ used in the long regression is the one found by fitting ρ simultaneously with the control model formula only. The best method is to use the value of ρ fitted simultaneously with the control and test model formulae. To achieve this, simply combine the control and test model formulae into a single model formula, and use it as the control model formula of a further analysis (and test for some arbitrary variable), and use the parameter estimates for the long regression with the control model formula only. The difference is likely always to be small, and if a variable is significant, it would be astonishing if the sign of the parameter estimates differed between the two methods. The whole point of the phylogenetic regression is that significance tests in the long regression cannot be trusted - this means that the standard errors of the parameter estimates cannot be trusted. They are therefore not supplied.

7.2 Calculating fitted values

Fitted values should be calculated in just the same way as would usually be done from SAS output, except that it must be done by hand. The design variables in SAS are chosen so that when adding a term for a categorical variable to construct a fitted value, the amount for the final level is always zero. This is shown explicitly in the output of the parameter values.

Section 7.1 explains why the long regression on control variables alone is the preferable analysis from which to calculate fitted values.

7.3 Phylogenetic degrees of freedom

These are discussed in §3(e) of the source paper. The basic idea of the phylogenetic regression is not to use species as independent datapoints, but instead to use the higher nodes in the phylogeny. This means that the number of independent datapoints is the

number of higher nodes, including the root. Apart from this basic reduction in numbers of degrees of freedom, compared to the number of species, there are two things that can happen to degrees of freedom because of the phylogenetic aspect. If these things do happen, they are reported by the program under the headings "Phylogenetic degrees of freedom in the numerator" and "Phylogenetic degrees of freedom in the denominator".

The "degrees of freedom in the numerator" refers to the numerator of the F-ratio, and is the number of degrees of freedom associated with the test model formula. If one of the test variables can be expressed as a linear combination of other test variables, then the number of degrees of freedom will be less than this. This situation is sometimes called multi-collinearity. If there is multi-collinearity in the species regression in the first place, then there will also be in the phylogenetic regression. But it can happen that variables which are not collinear in the species regression are collinear in the short regression. It is only these extra losses that are reported by the program. Such extra losses are likely to arise when variables differ from each other only at a few higher nodes. When degrees of freedom are lost in the numerator, it means that the variables involved are phylogenetically too restricted in their variation to deserve the full degrees of freedom.

"Degrees of freedom in the denominator" refers to the denominator of the F-ratio. The short regression is formed by condensing the information at a higher node into one datapoint, and to do the condensing it uses the residuals from the (long) regression on the control variables. If these residuals are all zero, then there is no unexplained variation at that node, and no condensing can be done. That higher node must therefore be dropped from the short regression. (Its parent and daughter nodes may still be retained.) When this occurs, the program reports which nodes have been dropped.

There are two ways loss of denominator degrees of freedom is likely to occur. What has to happen is that the fitted and observed values are identical for all the daughters of a higher node. The first reason is that a categorical variable has been included in the regression that takes different values for nearly all of a higher node's daughters and is unvarying elsewhere. Fitting this categorical variable can cause the observed and fitted values to be identical at that node. Any set of variables that differ only at a higher node can have the same effect. This is like, in an ordinary regression, including a categorical variable that is uniform except for one datapoint. Effectively, that datapoint is deleted from the regression.

The second way to lose a denominator degree of freedom is to have a y-variable that "just happens" to take the same value for all the species in a genus, or whose average values "just happen" to be the same for all the daughters of some higher node. If the error in the regression really were normally distributed, this "just happening" would never arise. Often, however, y-variables are treated as continuous, but in fact take a few discrete values. In this case, the "just happening" may arise quite often. Grafen and Ridley (1996, J. theor. Biol., 183, 255-267) discuss the case of a categorical y-variable, and report that the phylogenetic regression works well in the cases they studied compared to the other available methods.

Notice that the loss of a denominator degree of freedom means the loss of a datapoint. The second route introduces a new way for test variables to become collinear. They may become collinear when a datapoint that prevents collinearity is excluded.

The degrees of freedom possessed by the variables controlled for are subject to just the same considerations as the degrees of freedom possessed by the test variables. These changes are not reported by the program, as they will usually not be of major interest. They may cause puzzlement, however, as the degrees of freedom may not seem to add up correctly. There will seem to be too many altogether if the control variables are collinear - because these degrees of freedom will remain in the denominator instead of being soaked up by the control variables.

Losses of degrees of freedom in this way do not invalidate the F-ratio. They are detected by the program and appropriate action is taken. But they will happen in rather unusual circumstances, and may be caused by a mistake of some sort. So it is worthwhile working out where and why the degrees of freedom have been lost. Notice that sometimes degrees of freedom will be lost in the long regression before moving to the short regression. For example, when there is collinearity between some of the control variables, or between control and test variables, in the long regression itself. This has nothing special to do with phylogeny, and is not commented on by the program.

To track down the explanation for the degrees of freedom in the final F-test, it will help to include 'opdf=1' when %phyreg is used, for an account of the total degrees of freedom in the short regression, and how that total is divided up. (The degree of freedom allotted to ρ is discussed in §5(b) of the source paper. My view has changed since then for reasons given in section 1.1, and I no longer recommend subtracting a degree of freedom for the fitting of ρ.) Loss of degrees of freedom in the long and short regression, in control and test variables, can also be investigated by including 'opdf=1'. The variables excluded for collinearity will have an estimate of exactly '0' in the list of parameter estimates. Losses present in the long regression will occur in the short regression too, but are not related to phylogeny. Additional losses in the short compared to the long regression represent degrees of freedom lost for phylogenetic reasons.

7.4 Long and short regressions

This section explains what is meant by "long regression" and "short regression", terms which are used in the output of the program. The dataset begins as you enter it, with one datapoint for each species. Internally, it undergoes two transformations before the final F-ratio is produced, first into the long regression, then into the short regression.

The long dataset has one datapoint for each node in the phylogeny except the root, including each of the species. It has more datapoints than the species dataset, hence its name. Each datapoint has the average values of all the species below its corresponding node, expressed as a deviation from the corresponding average at its parent node. (The creation of the long regression is the process of "hanging on the tree" described in §3(a) of the source paper.). The long dataset and the long regression

are used to perform a regression that is valid in the face of *recognized phylogeny*, but not *unrecognized phylogeny*, that is equivalent to the *standard regression*. For explanations of the italicized terms, see the source paper. For present purposes, it is important that the long regression on the control variables alone is used to fit ρ, by maximum likelihood simultaneously with the regression parameters. It is the log-likelihood from this regression that is reported along with the maximizing value of ρ when opparm=1. (You can fix ρ by including 'rho=*value*' in the call of %phyreg. To restore automatic fitting, include 'rho=-1'.)

The path segment lengths implied by this value of ρ are then used to create the short dataset. This has one datapoint for each higher node (including the root). The idea is that each higher node should contribute one independent unit of information to the final statistical test. Theorem 3 of the source paper shows that the short regression on control variables alone has the same parameter estimates and deviance as the long regression on control variables alone. The F-ratio of the phylogenetic regression is the F-ratio for adding the test variables to the control variables in a regression on this short dataset.

The short regression provides biassed parameter estimates for test variables, because the linear contrasts used to form it from the long dataset are data-dependent. This is why they are not given. For parameter estimates, it is therefore best to use the estimates from the long regression, which are unbiassed. However, their standard errors are not given, as they cannot be trusted (this is the whole point of conducting a phylogenetic analysis).

7.5 The short regression dataset

Immediately after an analysis, the file phylo.short contains the short dataset. The variable names are just COL1, COL2 etc. The contents are as follows

The first column	The weighting variable, indicating whether the datapoint is included or not. See 'phylogenetic degrees of freedom' for an explanation of exclusion.
The second column	The contrasts for the y-variable
The third to penultimate columns	The contrasts for the x-variables. There is one for each parameter in the 'long regression of control and test variables', except the Constant, and they are in the same order as reported in the 'opparm=1' output.
The final column	Gives the node number of the higher node on the phylogeny as specified to the program. Some higher nodes may not be present, as omitted species may cause some higher nodes to have only one daughter node, and so 'disappear' from the phylogeny containing only the species included in a given analysis.

The linear contrasts can be inspected and connected to the phylogeny, to see which nodes contribute most to a relationship. They can also be plotted or subject to further calculation.

In interpreting the data, it is vital to remember that the short regression is fitted without a constant. Thus, if the signs of all the contrasts for a given higher node were reversed, it should make no difference to the interpretation, just as it would make no difference to the short regression.

The weighting variable is extremely important, and should be always be consulted, as the values for omitted higher nodes are not meaningful. When plotting, for example, it is obviously important to know which of the points contain meaningful data and deserve attention, and which of the points correspond to missing phylogenetic degrees of freedom in the denominator, and whose position is arbitrary. In general, omitted higher nodes will have zero values, but this value has been chosen arbitrarily, and does not correspond to any biological reality.

8. Publishing analyses

When reporting analyses using the phylogenetic regression, it is important to give details of which phylogeny was used, and how path segment lengths were assigned. Otherwise the analysis is unrepeatable even if the same dataset is available.

9. Example sessions

The example sessions illustrate many features of the program. If you are at a computer, it is probably best to run the examples rather than read most of this section. You do this by finding the PHYLOEXS library (see the installation section if you haven't got it), opening the 'Examples' catalog and opening 'Code'. This contains all the SAS code needed to run the examples, and all the same comments as you find below. You may select and Run > Submit (not forgetting you need to capture the final semicolon in the selection), or if you want to work hard, you could even type the commands into a separate editor window and run them from there.

If you aren't at a computer, then you could actually read this section.

Although the same comments are found here and in the online 'Code' item, here we give only excerpts from the output, to increase legibility.

Example 1 shows simple use, with all three methods of specifying a phylogeny (newick-style, taxonomic variables, and single variable) and all three methods of specifying path segment lengths (default "Figure 2", taxonomic levels, and completely general). A convenient way of exploring the effect of different sets of path segment lengths is illustrated. Use of %newick is demonstrated both for a phylogeny alone, and for phylogeny and heights together. The effect of missing values is shown. Polynomial regression is carried out. It shows how to check on the parameters that are being used by %phyreg, and how to track the number of species omitted through missing values. An analysis is conducted on a subset of species using a variable to indicate which should be included.

Example 2 shows categorical variables in use, including interactions between them. It shows how to convert from the taxonomic variables method to the single variable method through the saved file phylo.zphy. Phylogenetic degrees of freedom are lost when the y-variable is discrete.

9.1 Example 1

The first command contains 'reset=YES' so that the example will run properly even if you've been doing other analyses beforehand. It uses data in phyloexs.Example1, obtains the phylogeny from taxonomic variables ge fa and or, and tests whether alt influences conc. For this first analysis we give the whole pages of output including the headings.

```
%phyreg(reset=YES, dataset=phyloexs.Example1, taxvars=ge fa or,
   y=conc, test=alt);
```

```
                                  PHYLOGENETIC REGRESSION

                              y = conc; control = ; test = alt.

                                    PARAMETER ESTIMATES


Fitting of rho
    The likelihood was maximised by the value rho = 0.7058382
    The value of the log-likelihood was -50.09318



                                   Regression coefficients
                                   from long regression on
                                   control variables only

                                  Parameter    Estimate
                                  Constant     2.7070660722

Note: these are the estimates to use for interpretation of results



                                   Regression coefficients
                                   from long regression on
                                      control and test
                                           variables

                                  Parameter    Estimate
                                  Constant     2.6587084129
                                  alt          0.0017032668

Note: repeat with all variables as controls for better estimates
```

```
                                    PHYLOGENETIC REGRESSION

                                y = conc; control = ; test = alt.

Test statistic for
            yvariable = conc
        controlling for
            testing for alt

  F(1,11) = 1.8608968079    p = 0.1997810801



                                    SOURCE OF PATH LENGTHS

                        Path lengths derived from "Figure 2" default method

                                    SOURCE OF PHYLOGENY

                    Phylogeny constructed from taxonomic variables: ge fa or
```

Only the default two pages of output are given. The first gives parameter estimates, including of rho. The second gives the F-ratio for the test, and also confirmation of how path segment lengths were assigned and where the phylogeny came from. There is no evidence that alt influences conc.

Next we decide to set heights using the taxonomic levels method. We need a dataset with four numbers in it (one more than the number of taxonomic variables) in ascending order. This is a simple way to create such a dataset

```
data phyloexs.hts1a;
keep Height;
array b {4} (1,2,3,4);
do i=1 to dim(b);
 Height=b[i];
 output;
 end;
run;
```

The same analysis, but specifying that file as the source of the heights of nodes, is carried out simply by giving the heightsdataset parameter.

```
%phyreg(heightsdataset=phyloexs.hts1a);
```

```
Fitting of rho
    The likelihood was maximised by the value rho = 2.2368553
    The value of the log-likelihood was -48.66468
```

```
Test statistic for
            yvariable = conc
        controlling for
            testing for alt

  F(1,11) = 1.8413430045    p = 0.2019854397
```

The value of rho has changed quite dramatically, but this is presumably compensating for the different balance between top and bottom of the tree in the two sets of heights used. There is still no evidence that alt influences conc. What about some different weights? We put a different set of weights into the same dataset.

```
data phyloexs.hts1a;
keep Height;
array b {4} (4,5,6,8);
do i=1 to dim(b);
 Height=b[i];
 output;
 end;
run;
```

As none of the parameters has changed, we try the new weights just by repeating the same call -- and so no parameters need be given.

```
%phyreg();
```

```
Fitting of rho
    The likelihood was maximised by the value rho = 3.5062027
    The value of the log-likelihood was -49.28486
```

```
Test statistic for
            yvariable = conc
         controlling for
             testing for alt

  F(1,11) = 2.3466144628    p = 0.1537952043
```

The output shows that rho has changed again, but still no evidence that alt influences conc. Just to show what happens if we make a mistake in the number of levels in the heights dataset, let's put 5 heights in instead of the correct 4, and again repeat the analysis.

```
data phyloexs.hts1a;
keep Height;
array b {5} (1,2,3,4,8);
do i=1 to dim(b);
 Height=b[i];
 output;
 end;
run;

%phyreg();
```

```
Test statistic for
             yvariable = conc
         controlling for
             testing for alt

  F(1,11) = 1.8608968079    p = 0.1997810801



                                    SOURCE OF PATH LENGTHS

               Path lengths derived from "Figure 2" default method

               WARNING: A heightsdataset was specified. However, the number of rows
               did not match the number of levels or number of higher nodes

                                    SOURCE OF PHYLOGENY

                    Phylogeny constructed from taxonomic variables: ge fa or
```

The default Figure 2 heights are used and a warning is given. The output of parameter estimates and tests is therefore the same as that for the first analysis.

The next command used the macro %newick for the first time. We provide a phylogeny in newick style. The species are represented just by their numbers (to save space here), but could be represented by their names. Spaces and carriage returns are ignored by the program, and so can be used freely for formatting.

```
%newick(phylstr=(1,2,3,4,(5,(6,7,8),9),((10,11),12,13,
((14,15),16),(17,18),(19,20,(21,22,23),24),(25,26,27,28,29,30),
(31,32))),
       outphyfile=phyloexs.phy1a);
```

The output file phyloexs.phy1a now contains the phylogeny in single variable format, and so can be used in the following command. We use the single variable method instead of taxonomic variables. It is necessary to unset 'taxvars' because if they are present they override the phydataset. And we unset heightsdataset too, to avoid the warnings.

```
%phyreg(taxvars=_UNSET_, phydataset=phyloexs.phy1a,
   heightsdataset=_UNSET_);
```

```
Test statistic for
            yvariable = conc
        controlling for
            testing for alt

  F(1,11) = 1.8608968079    p = 0.1997810801



                                    SOURCE OF PATH LENGTHS

                        Path lengths derived from "Figure 2" default method

                                    SOURCE OF PHYLOGENY

                            Phylogeny read in from file: phyloexs.phy1a
```

Notice that the numerical output is the same. This is because the phylogeny we have created is the same as that produced by the taxonomic variables. On the F-ratio page, the method of specifying the phylogeny is confirmed.

The next command again uses %newick, but this time we specify the height of each node with a colon following the name (for species) or closing bracket (for higher nodes). We need to say 'data=HEIGHTS', because otherwise the program assumes that the additional data represents the length of the path segment leading to the node from its parent, and not its height.

```
%newick(phylstr=(1:0,2:0,3:0,4:0,(5:0,(6:0,7:0,8:0):1,9:0):3,
   ((10:0,11:0):2,12:0,13:0, ((14:0,15:0):1,16:0):2,(17:0,18:0):2,
   (19:0,20:0,(21:0,22:0,23:0):1,24:0):2,
   (25:0,26:0,27:0,28:0,29:0,30:0):2,(31:0,32:0):2):3):4,
   outphyfile=phyloexs.phy1b,
   outheightsfile=phyloexs.hts1b,data=HEIGHTS);
```

The phylogeny file and heights file are used in the next call of %phyreg. The output is the same as in an earlier analysis, because the heights are just the same as those obtained by the taxonomic levels method. Thus again the diversity of possible methods is illustrated with a uniformity of numerical results.

```
%phyreg(phydataset=phyloexs.phy1b, heightsdataset=phyloexs.hts1b,
   taxvars=_UNSET_);
```

```
Test statistic for
            yvariable = conc
        controlling for
            testing for alt

  F(1,11) = 1.8413430045    p = 0.2019854397



                                    SOURCE OF PATH LENGTHS

                  Heights of each higher node read from file: phyloexs.hts1b

                                    SOURCE OF PHYLOGENY

                          Phylogeny read in from file: phyloexs.phy1b
```

Now a sequence of analyses that in turn (i) control for alt and test for dura (ii) control for alt and dura and test for a quadratic term in dura (iii) control for alt and the linear and quadratic terms in dura, and test for the cubic term in dura.

```
%phyreg(control=alt,test=dura);
```

```
Test statistic for
            yvariable = conc
        controlling for alt
            testing for dura

  F(1,8) = 138.9679102     p = 2.4547658E-6
```

```
%phyreg(control=alt dura,test=dura*dura);
```

```
Test statistic for
            yvariable = conc
        controlling for alt dura
            testing for dura*dura

  F(1,7) = 2.1417316144    p = 0.1867455476
```

```
%phyreg(control=alt dura dura*dura,test=dura*dura*dura);
```

```
Test statistic for
            yvariable = conc
        controlling for alt dura dura*dura
            testing for dura*dura*dura

  F(1,6) = 0.2161951295    p = 0.6583501557
```

Before attending to the statistical significances, look at the total degrees of freedom for the F-test. There are two degrees of freedom fewer than expected, compared to the analyses that don't involve dura, because of missing data. This missing data reduces by two the number of higher nodes in the phylogeny, and so the degrees of freedom in the short regression. We will see later how to get an accounting of all these things in the output by specifying 'opdf=1'.

There is no evidence that more than the linear term is required, though the linear term is overwhelmingly significant. So we control for alt and dura, and test for terr. We need to specify that terr is a class variable. Notice in the parameter output how the parameters are named. The variable name terr is followed by an underscore and then by the name of the level.

```
%phyreg(control=alt dura, test=terr, class=terr);
```

```
        Regression coefficients
        from long regression on
           control and test
               variables

       Parameter    Estimate
       Constant      -0.17537093
       alt          0.0006311214
       dura         0.4560820629
       terr_N        -0.28624954
       terr_S       0.2600281086
       terr_W                  0
```

```
Test statistic for
             yvariable = conc
         controlling for alt dura
             testing for terr

  F(2,6) = 13.937949789    p = 0.0055562431
```

Next we control for nterr, which is a numerical coding of terr as 1, 2, 3. This effectively looks for a 'non-linear' effect of terr. Notice that two of the terr levels are now aliased, as terr has only one degree of freedom once nterr is controlled for. This shows that SAS behaves nicely in the face of collinearity, and we can safely include linear contrasts, letting SAS handle the degrees of freedom.

```
%phyreg(control=alt dura nterr, test=terr);
```

```
       Regression coefficients
       from long regression on
          control and test
              variables

      Parameter    Estimate
      Constant     -0.730722209
      alt          0.0005899324
      dura         0.4554004697
      nterr        0.2768878517
      terr_N       0.0074394101
      terr_S                  0
      terr_W                  0
```

```
Test statistic for
             yvariable = conc
          controlling for alt dura nterr
             testing for terr

  F(1,6) = 0.0021494105    p = 0.9645265137
```

The program warns that rho became too small. This probably means that there seemed to be anti-phylogenetic correlations, in which close species were too different from each other. We could try reducing minrho, in case that did help, but won't pursue that here.

Finally, we decide to repeat the final analysis on a subset of species. The variable conf in the dataset contains a 1 for some species and a 0 for others. Those with a 1 are included in the next analysis, and those with a 0 are excluded. There are many reasons why such an analysis might be of interest -- perhaps some species have less reliable data, or we want to know whether omitting a particular taxon will alter the result.

We also ask for the page of output that details how the degrees of freedom are calculated. It shows how many species are omitted by SPUSE, and how many for missing values, as well as many of the other small additions of degrees of freedom that are necessary to make the test work properly.

```
%phyreg(spuse=conf, opdf=1);
```

```
BREAKDOWN OF SPECIES NUMBERS

Total number of species    =        32
Omitted by SPUSE           =         7
Omitted for missing values =         3
Included in analysis       =        22


BREAKDOWN OF NUMBERS OF HIGHER NODES

                                                    Total:       12
                             Lost through omitted species:        2
                         Lost through lack of variability:        0
 Remainder as total degrees of freedom for short regression:      10


BREAKDOWN OF DEGREES OF FREEDOM IN THE SHORT REGRESSION

Control variables.    In long regression:         3
           Lost for phylogenetic reasons:         0
                     Net, by subtraction:         3

Test variables.        In long regression:        1
            Lost for phylogenetic reasons:        0
                      Net, by subtraction:        1

Hence, the total degrees of freedom break down as follows:

                                   Total:       10
                                 Control:        3
  Additional fitted parameters (e.g. rho):        0
                                    Test:        1
                Residual, by subtraction:        6

The last two numbers are the numerator and denominator
degrees of freedom in the F-test of the short regression.
```

The rho problem disappears here. This completes the first example.

9.2 Example 2

We begin again by resetting parameters, so no hangovers from the previous example or your previous work interfere with the macro. We use the second example file and specify twenty-four taxonomic variables. We test for the effect of q on p, and set the lower boundary of the initial search region for rho to be 0.01. The reason we set addDF=1 is that this example was used in the GLIM implementation using an additional degree of freedom, and it is good to check that this version gives the same results.

```
%phyreg(reset=YES, dataset=phyloexs.example2,
        taxvars=t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12 t13 t14 t15
                t16 t17 t18 t19 t20 t21 t22 t23 t24,
        y=p,test=q, addDF=1,lorho=0.01);
```

```
                                  PHYLOGENETIC REGRESSION

                                y = p; control = ; test = q.

                                     PARAMETER ESTIMATES


Fitting of rho
    The likelihood was maximised by the value rho = 0.0001667
    The value of the log-likelihood was -98.38011



                                   Regression coefficients
                                   from long regression on
                                   control variables only

                                  Parameter    Estimate
                                  Constant     0.5055706672

Note: these are the estimates to use for interpretation of results



                                   Regression coefficients
                                   from long regression on
                                      control and test
                                           variables

                                  Parameter    Estimate
                                  Constant     0.4264927101
                                  q            0.1628833974

Note: repeat with all variables as controls for better estimates
```

```
                                  PHYLOGENETIC REGRESSION

                                 y = p; control = ; test = q.

Test statistic for
            yvariable = p
         controlling for
            testing for q

  F(1,71) = 2.2491337221    p = 0.138122954

WARNING: Test invalid because Rho-fitting ran up against the minimum value
         Try lowering minrho.


                                     SOURCE OF PATH LENGTHS

                        Path lengths derived from "Figure 2" default method


                                     SOURCE OF PHYLOGENY

Phylogeny constructed from taxonomic variables: t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12
t13 t14 t15 t16 t17 t18 t19 t20 t21 t22 t23 t24
```

Again the low rho message, but we continue as an exercise. We define f and gap to be class variables, and test first for f, then for gap controlling for f, and then for the interaction f*gap controlling for f and gap.

```
%phyreg(test=f,class=f gap);
```

```
Test statistic for
            yvariable = p
         controlling for
            testing for f

  F(4,68) = 1.7650267658    p = 0.1460277239
```

```
%phyreg(control=f,test=gap)
```

```
Test statistic for
            yvariable = p
         controlling for f
            testing for gap

  F(4,64) = 2.3275783353    p = 0.0656095153
```

```
%phyreg(control=f gap,test=f*gap);
```

```
       Regression coefficients
       from long regression on
       control variables only

      Parameter    Estimate
      Constant     0.4839295904
      f_A           -0.04796857
      f_B          0.0255950159
      f_C          0.0319534508
      f_D          0.1429970076
      f_E                     0
      gap_U        0.1231277148
      gap_V         -0.19579976
      gap_W        0.0090303048
      gap_X        0.0341361759
      gap_Y                   0

Note: these are the estimates to use for interpretation of results



       Regression coefficients
       from long regression on
          control and test
              variables

      Parameter    Estimate
      Constant     0.5649350163
      f_A          -0.190767584
      f_B          -0.056171442
      f_C          -0.032659121
      f_D          -0.000115083
      f_E                     0
      gap_U        0.0704397682
      gap_V        -0.335438294
      gap_W        -0.080572118
      gap_X        -0.137795493
      gap_Y                   0
      f*gap_A_U    0.1009033029
      f*gap_A_V    0.2812146987
      f*gap_A_W    0.1939650222
      f*gap_A_X    0.2483883975
      f*gap_A_Y               0
      f*gap_B_U    -0.157668629
      f*gap_B_V    0.3354761002
      f*gap_B_W    -0.079246631
      f*gap_B_X     0.359543411
      f*gap_B_Y               0
      f*gap_C_U    0.0557710091
      f*gap_C_V    0.1055157382
      f*gap_C_W    0.0773299262
      f*gap_C_X    0.1164199681
      f*gap_C_Y               0
      f*gap_D_U    0.2724956615
      f*gap_D_V    0.0419737239
      f*gap_D_W     0.288613326
      f*gap_D_X    0.2228033679
      f*gap_D_Y               0
      f*gap_E_U               0
```

```
      f*gap_E_V              0
      f*gap_E_W              0
      f*gap_E_X              0
      f*gap_E_Y              0
```

```
Test statistic for
            yvariable = p
        controlling for f gap
            testing for f*gap

  F(16,48) = 0.6084652819    p = 0.8612710462
```

f isn't significant, gap is almost significant, and the interaction isn't. Notice the parameter names for the interaction. They have f*gap followed by an underscore and the level of f and then another underscore and the level of gap.

Next a discrete variable is set as the y-variable. This is a reasonable thing to do (Grafen and Ridley, 1996, J. theor. Biol., 183, 255-267), and introduces a new phenomenon -- losing phylogenetic degrees of freedom in the denominator. See section 7.3 and the source paper for more details. Here we do an analysis to see what the output looks like. gn is a version of gap that is coded 1 to 5 instead of U to Y.

```
%phyreg(y=gn, control=_UNSET_, test=q);
```

```
Test statistic for
            yvariable = gn
        controlling for
            testing for q

  F(1,55) = 0.6549672197    p = 0.4218297408



                                    SOURCE OF PATH LENGTHS

                        Path lengths derived from "Figure 2" default method


                                    SOURCE OF PHYLOGENY

Phylogeny constructed from taxonomic variables: t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12
t13 t14 t15 t16 t17 t18 t19 t20 t21 t22 t23 t24


WARNINGS ABOUT PHYLOGENETIC DEGREES OF FREEDOM

Number of missing phylogenetic degrees of freedom in the denominator:        16
Numbers of the missing nodes: 103, 106, 109, 112, 115, 118, 121, 124, 127, 130, 133,
136, 139, 142, 145, 148
```

The F-ratio page has the extra output. It warns of the loss of degrees of freedom, and gives the identifying numbers of the higher nodes. If you have a drawing of the phylogeny with the nodes numbered, you can look to see which have been omitted. The opdf=1 page will show the accounting of degrees of freedom including this loss.

The next feature is to capture the phylogeny file. The existing analysis uses taxonomic variables, but in such cases %phyreg stores the single variable in a dataset called phylo.zphy. The following DATA step copies it into the phyloexs library, so it won't be overwritten in later runs of %phyreg.

```
data phyloexs.phy2;
set phylo.zphy;
run;
```

Then we use that phydataset in the next analysis. It should show no change in the output (apart from recognising that the phylogeny came from a file instead of taxonomic variables) because it is the same phylogeny as just used. We also ask for the parameters of the macro to be printed with opmacparm=1. This can be useful if we are unsure what values are taken by all the remembered items.

```
%phyreg(phydataset=phyloexs.phy2,opmacparm=1,opdf=1);
```

```
                    Confirmation of values of input parameters used in analysis



y = gn;  class = f gap; spuse = ; taxvars = t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12 t13
t14 t15 t16 t17 t18 t19 t20 t21 t22 t23 t24; addDF = 1; control = ; test = q; dataset
= phyloexs.example2; phydataset = phyloexs.phy2; heightsdataset = ; rho = -1; lorho =
0.01; hirho = 0.6; errrho = 0.02; minrho = 0.001; tolerance = 0.000001; opf = 1; opdf
= 1; opparm = 1; opmacparm = 1
```

```
Test statistic for
             yvariable = gn
         controlling for
             testing for q

  F(1,55) = 0.6549672197    p = 0.4218297408
```

In this case it explains why we *didn't* manage to use the phydataset. taxvars was still set, and so overrode phydataset. Thus we unset taxvars and get the result we want.

```
%phyreg(taxvars=_UNSET_);
```

```
                    Confirmation of values of input parameters used in analysis



y = gn;  class = f gap; spuse = ; taxvars = ; addDF = 1; control = ; test = q; dataset
= phyloexs.example2; phydataset = phyloexs.phy2; heightsdataset = ; rho = -1; lorho =
0.01; hirho = 0.6; errrho = 0.02; minrho = 0.001; tolerance = 0.000001; opf = 1; opdf
= 1; opparm = 1; opmacparm = 1
```

```
Test statistic for
             yvariable = gn
         controlling for
             testing for q

  F(1,55) = 0.6549672197    p = 0.4218297408



                                    SOURCE OF PATH LENGTHS

                        Path lengths derived from "Figure 2" default method

                                    SOURCE OF PHYLOGENY

                            Phylogeny read in from file: phyloexs.phy2


WARNINGS ABOUT PHYLOGENETIC DEGREES OF FREEDOM

Number of missing phylogenetic degrees of freedom in the denominator:        16
Numbers of the missing nodes: 103, 106, 109, 112, 115, 118, 121, 124, 127, 130, 133,
136, 139, 142, 145, 148
```

This ends the examples.

10. Reference

10.1 Complete list of the parameters of %phyreg with their default values

Name	Meaning	Status	Default (if any)
y	The y-variable in the regression.	C	
control	A model formula representing the variables and interactions to be controlled for in the analysis. See SAS documentation for PROC GLM for permissible model formulae. No quotation marks are required.		
test	A model formula representing the variables and interactions to be tested for in the analysis. . See SAS documentation for PROC GLM for permissible model formulae. No quotation marks are required.	C	
class	A list of variables to be treated as ‘class variables’ (also called factors or categorical variables). The items should be separated by spaces and the list requires no quotation marks.	O	
spuse	The name of a variable which contains a 1 for each species to include and a 0 for each species to exclude.	O	All species are included
dataset	A SAS dataset that contains all the data variables	C	

	used for the y-variable, used in the control and test model formulae. It must also contain, if they are used in an analysis, the taxonomic variables used to specify the phylogeny (i.e. the variables specified in the parameter 'taxvars'), and the variable that specifies which species to include (i.e. the variable specified in the parameter 'spuse'		
phydataset	If the single variable method of specifying the phylogeny is used, this parameter must contain the name of a SAS dataset containing a single numeric variable, which encodes the phylogeny.	O*	
heightsdataset	If the taxonomic levels or fully general heights methods of specifying path segment lengths is used, then this parameter must contain the name of a SAS dataset containing a single numeric variable which contains the heights.	O	
addDF	The original implementation of the phylogenetic regression permitted a degree of freedom to be subtracted from the short regression to allow for the fitting of ρ. This subtraction is no longer recommended, but the facility is included to allow that choice to the user, and to allow comparison with previous analyses which did subtract a degree of freedom. The outdated advice was to set addDF to 0 if a value of ρ was specified by the user, and to 1 if the ρ was fitted.	O	0
rho	If this value is positive, it is used as the fixed value of ρ. If zero or negative, then ρ is fitted from the data using maximum likelihood.	O	-1
lorho, hirho	The minimum and maximum values in the initial search grid for ρ. It will speed up the search if the value of ρ is between these values, and if these values are relatively close together.	O	0.3, 0.6
errrho	The accuracy to which ρ is fitted (perhaps somewhat illogically in additive terms). A smaller value will increase the time taken in the search.	O	0.02
minrho	The value below which the search for ρ is stopped. If ρ is very close to zero, then numerical errors may make the results unreliable. ρ =0 would correspond to an analysis without phylogenetic structure i.e. independent species	O	0.001
tolerance	Collinearity must be decided by whether certain numerical values are zero or not. Numerical errors cause even 'truly zero' values not to be exactly zero. This parameter is used to decide how close to zero should count as a true zero. It is used to decide which (if any) higher nodes must	O	0.000001

	be dropped from the short regression, thus losing a phylogenetic degree of freedom in the denominator.		
taxvars	If the taxonomic variable method is used to specify the phylogeny, then this parameter should contain the list of variable names. The list should not include a species level, and should begin with the lowest level. The variables must be either all numeric, or all character. The items in the list should be separated by spaces, and the list requires no quotation marks.	O*	
opf	Set to 1 to obtain output including the F-ratio and p-value; set to 0 to suppress it.	O	1
opdf	Set to 1 to obtain output that shows how the numbers of species and higher nodes, the number of omitted species, the control and test model formulae, the value of 'addDF' and phylogenetic considerations produce the degrees of freedom of the final statistical test	O	0
opparm	Set to 1 to obtain output on the value of ρ (and if fitted, the maximum likelihood), the parameters of the General Linear Model, and the weighted means of the model terms.	O	1
opmacparm	Set to 1 to obtain output that gives the value of every parameter of %phyreg.	O	0
reset	NO (and indeed any value except YES) will 'remember' previous parameter values until they are altered. YES (case sensitive!) will set all values to their defaults apart from those actually specified in that particular call of the macro. A value of YES works essentially by resetting the 'remembered' values to their defaults and so affects subsequent calls too.	O	NO

Note: the variable specified by the parameter y, the variables used to define the models test and control, the taxonomic variables specified in taxvars, and the variable specified by spuse, must all belong to the dataset specified by the parameter 'dataset'.

Status is C for compulsory and O for optional. O* indicates a set of parameters (phydataset and taxvars) each of which is optional but at least one of which must be specified.

10.2 Complete list of the parameters of %newick with their default values

The heading of the definition of %newick is

```
%macro newick(phylstr=,phylfile=,outphyfile=,outspecfile=,
              outhigherfile=,outheightsfile=,data=);
```

The three input parameters, of which exactly one of the first two must be set, are

Name	Meaning
phylstr	A string to contain the newick-style phylogeny, which should *not* be enclosed in quotation marks. Carriage returns and spaces are ignored in the string and so may be used freely for formatting.
phylfile	A textfile to contain the newick-style phylogeny.
data	If this variable is set to HEIGHTS, then the macro assumes that the additional data is the height of the corresponding node (and not just the path segment length). The root must have a value if data is set to HEIGHTS. The default value is simply blank, but any value other than HEIGHTS will have the same effect.

The output parameters, all of which are optional and all of which will be created as SAS datasets, are

Name	Meaning
outphyfile	To contain the phylogeny according to the single variable method, which can then be supplied to %phyreg as the value of the parameter 'phydataset'
outheightsfile	To contain the heights of each node, which can be supplied to %phyreg as the value of the parameter 'heightsdataset'
outspecfile	Contains three variables (i) the species name (ii) the species number (but actually a text variable) and (iii) the additional data for the species (as a text variable)
outhigherfile	Contains two variables (i) the number of each higher node in turn (but actually a text variable) (ii) the additional data for that higher node (as a text variable)

The first two output parameters are designed so that they can be supplied to %phyreg. The idea is that after specifying 'outphyfile=mylib.phylog' to %newick, one can then specify 'inphydataset=mylib.phylog' to %phyreg. Equally, after specifying 'outheightsfile=mylib.heights' to %newick, one can then specify 'inheightsdataset=mylib.heights' to %phyreg. The other two output parameters provide a service for users, but have no special role in conducting analyses.

10.3 Leftover datasets and catalogs

The library phylo is set up initially to contain one catalog, 'Program', whose use is described in the installation section 2. After an analysis has been run, it will contain many further files. This section lists them and explains what they do. This will usually be of no importance to the user, except that phylo.zshort contains the linear contrasts used in the final regression. See section 7.5.

1	PROGRAM	CATALOG	Contains (i) Install, to be submitted at initial installation, which places modules in the catalog zKeepMacs (ii) LoadMacros, which allows use of %newick and %phyreg for the remainder of the SAS session

2	ZDESIGNX	DATA	Contains the design output of PROC GLMMOD for *y=control*
3	ZDESIGNXZ	DATA	Contains the design output of PROC GLMMOD for *y=control test*
4	ZDMISSING	DATA	A dataset used in discovering which species have missing values
5	ZKEEPMACS	CATALOG	Contains modules referred to by %phyreg. It is created in the initial installation
6	ZLGPARMX	DATA	Output from a PROC REG on the long regression on *control*. It contains information on parameter values
7	ZLGPARMXZ	DATA	Output from a PROC REG on the long regression on *control test*. It contains information on parameter values
8	ZLONGSTATX	DATA	Output from a PROC GLM on the long regression on *control*. It contains information on degrees of freedom and sums of squares
9	ZLONGSTATXZ	DATA	Output from a PROC GLM on the long regression on *control test*. It contains information on degrees of freedom and sums of squares
10	ZLONGXZ	DATA	Dataset for the long regressions
11	ZMERC	DATA	Dataset used to transfer column names computed by the second IML into the ordinary SAS code in between the second and third IMLs
12	ZOPARX	DATA	Contains the parameter output of PROC GLMMOD for *y=control*
13	ZOPARXZ	DATA	Contains the parameter output of PROC GLMMOD for *y=control test*
14	ZPHY	DATA	Contains the single-variable phylogeny from the most recent analysis in which the taxonomic variable method was used (and will not exist until such an analysis is run). This file can be copied and then used in subsequent analyses instead of the taxonomic variable method.
15	ZSHORSTATX	DATA	Output from a PROC GLM on the short regression on *control*. It contains information on degrees of freedom and sums of squares
16	ZSHORSTATXZ	DATA	Output from a PROC GLM on the short regression on *control test*. It contains information on degrees of freedom and sums of squares
17	ZSHORT	DATA	Dataset for the short regression. Users may wish to use this dataset, as it contains the linear contrasts. See section 7.5.
18	ZSHPARMX	DATA	Output from a PROC REG on the short regression on *control*. It contains information on parameter values
19	ZSHPARMXZ	DATA	Output from a PROC REG on the short regression on *control test*. It contains information on parameter values
20	ZTEMP	CATALOG	Contains IML variables that need to be carried over from the second IML to the third IML. May be read from within IML using 'STORAGE PHYLO.ZTEMP; LOAD *list of names*;'

10.4 Choosing variable names

For most uses of the program, there are no real difficulties about variable names. Characters such as forward slash, comma, and percent should be avoided. Standard SAS names are fine. If you use a name with a nonstandard character in a call of %phyreg, you may well find strange errors being reported in the SAS log.

The one situation in which conflicts can arise between the internal workings of %phyreg and the user's own workings is if users define their own macro variables. If you don't know what a macro variable is, then it is very unlikely that you are using them. %phyreg stores values from one call to the next, to allow values to be carried over. It does so in macro variables, and so there is a potential for name conflicts if users happen to choose the same names for their own macro variables. To avoid conflicts, it will suffice in naming macro variables to avoid names that end with an underscore. It is important to state again that this restriction does not apply to variable names in SAS datasets, or to the names of datasets, but only to macro variables.

11. Revision History

The first version of the manual was released in March 2004, with the first released version of the software, and was version number 0.5.

Version 0.6 was released in April 2005, and contained one bug fix. Version 0.5 misbehaved by crashing when (i) a heights dataset was used AND (ii) not all species were included in the analysis AND (iii) this resulted in the phylogeny of species included in the analysis having at least one fewer higher node than the phylogeny of all species specified in the original phylogeny. The nature of the bug meant that if an analysis did work in version 0.5, then it gave the correct result. The bug has now been mended, and I am grateful to Mark Harris for bringing the problem to my attention.

Two small points were expanded in the manual in Release 0.6. The 'YES' in 'reset=YES' must be upper case. Every other value (including 'yes') counts as 'NO'. This may be changed in later versions of the program. When creating a library, and in particular PHYLO during installation, it will usually be desirable, in the dialog box in which it is created and named, to check the box that causes it to be enabled at startup.

The current version as of September 2006 is Version 0.7. Versions 0.5 and 0.6 misbehaved when using the taxonomic variables method of specifying the phylogeny, in special circumstances. These circumstances could not arise if the data were sorted in a phylogenetic order i.e. each taxon consisted of a consecutive set of species. They were likely to arise if the phylogeny was quite well defined, and if the data were sorted by some other characteristic. Fortunately, the nature of the bug means that if it struck, it would crash the program – thus analyses that did work gave the correct result. The bug has now been mended, and the species can be in any order. I am grateful to Yetta Jager for contacting me about this problem.

I have taken the opportunity of the revision to add the version number to the title 'Phylogenetic Regression' in the output, and to alter slightly the note in the output of parameter estimates.

I would be grateful for reports of further bugs or inaccuracies. Please e-mail me at alan.grafen@sjc.ox.ac.uk.