HETAGE Workshop, 5 July 2010

Canterbury, ISEC 2010.

Topics in the Workshop

- Introduction
- Background - the Jolly-Seber and Schwarz-Arnason models
- Practical Session 1
- Age-structured models - the discrete case
- Practical Session 2
- Age-structured models - the continuous case
- Practical Session 3
- Models with heterogeneity
- Practical Session 4
- Incorporating extra information
- Combining heterogeneity and age structure
- Practical Session 5
- User-defined functions
- Great crested newts example

Structure of the hetage R code

Likelihood function

The cornerstone is one function, mLL, which evaluates minus the log likelihood, based on a list of parameters:

`N` = superpopulation size

`beta` = a vector of entry parameters

`phi` = an array of survival probabilities

`p` = an array of capture probabilities

`pi` = a vector of proportions in each group of a finite mixture.

There is only the one likelihood function, and for fast evaluation it was written in C. It has been compiled, and is labelled `hetageLL.dll`

Different models

There is a suite of models which may be fitted to the data. Each model has two wrapper functions. Suppose we have a generic model called "thismodel". (They are actually called (phic,pc), (phic.pt) etc.)

The function `v.to.f.thismodel` takes a minimal vector of independent parameters and expands it into a full list of parameters (hence v.to.f for vector to full). This wraps up the parameters in a way which is usable by the likelihood function.

The function `start.thismodel` is used in obtaining suitable starting points. It unwraps a list of parameters and turns them into a minimal vector of independent parameters needed by that model.

Preparing the data

The function `hetage.process.data` takes the data frame, derives assorted constants, and turns the data into an x matrix (`x.mat`) with one row per unique capture history, and a y vector (`y.vect`) which specifies the number of copies of each row of `x.mat`.

Model fitting function

The function `hetage.fit.model` fits one model. It uses the current data set in `x.mat` and `y.vect`, and the user names the model to be fitted. Optionally, the user may provide a starting list of parameters, but if this is not done the function starts with a primitive set of constants for the parameters.

Methods used

A likelihood is maximised, using the R function `optim` on the set of independent parameters. For speed and computational accuracy, N is fitted on the log scale, and all proportions and probabilities are on the logistic scale. Within the function `optim`, the appropriate v.to.f function is used to expand the input vector in order to feed it to the likelihood for evaluation. The hessian matrix is calculated, in order to provide standard errors for N and other parameters.

Troubleshooting

Warnings:

For some models, the Hessian may not be able to be inverted. This is only a warning, not an error. It does not halt the sequence of calculations. The model has been fitted, but the variance-covariance matrix VC is NULL. Possible causes include gaps in the data - not enough different capture histories.

Errors:

If a model fails to fit, try starts from a different point. The default starts may be fairly hopeless.

Starting from a similar model which has successfully converged may be useful.

Example: To fit the model phi(t)p(h) we could use the parameter list output from model phi(c)p(h). The code might be:

phit.ph.out $<-$ hetage.fit.model("phit.ph",start=phic.ph.out$parameters)