iotc
ctoi
Indian Ocean Tuna Commission
Commission des Thons de l'Océan Indien

IOTC-2006-WPTT-INF06

IOTC Technical Report 02/01

Converting weight data into length data

IOTC Secretariat

IOTC – Seychelles, March 2002

Objective

The objective of this report is to document the statistical procedure applied to convert SF data reported in weight into length measurements.

Converting from Weight to Length

Some datasets, such as the Japanese SF data, may come as length or weight measurement. Therefore, there is a need for a standard procedure to convert all measurements into common units. For convenience, we have chosen to convert all measurements into length but a similar argument could be used to convert from length into weight.

A common procedure is to use the inverse of a length-weight (LW) relationship for the mapping. That is, given that we have a relationship:

$$W = aL^b \Rightarrow L = \left(\frac{W}{a}\right)^{\frac{1}{b}}$$

However, the parameters of a LW relationship are estimated in a way to optimize statistical properties of the expectation of weight conditional on length, $E(W \mid L)$. Instead, what we are interested is in obtaining the best, linear unbiased estimate of $E(L \mid W)$. This is a different estimation that will result in parameters with different numeric values. Therefore, it is better to actually estimate a new regression where length is the dependent variable.

A second, relatively minor, issue is that in many cases, the parameters of the LW have been estimated through a linear regression of log(W) vs log(L). The consequence of the logarithmic transformation is to increase the weight of the smallest observation at the expense of the larger values and, therefore, the parameters are somewhat biased (see Beauchamp, 1973, for a full treatment of this problem). This also implies assuming a lognormal error structure. A preferred approach is to use a non-linear estimation procedure on the original model without transforming L or W, obtaining in this way the $E(L \mid W)$.

Usually, we want to convert a distribution of weights into a distribution of lengths. At certain weights, for a fish of any given W, there is a distribution of possible lengths that extends a single length category. Therefore, we need to apportion the fish falling into a weight class to several length classes. This is easy to do considering that for a given weight w, the distribution of the standardized predicted length follows a Student's t distribution:

$$\frac{(\hat{l}_w - l_w)}{s_w} \sim t(1 - \alpha, n - 2); \hat{l}_w = cw^d$$

The estimate of the standard error (s_w) contains two components to account for the uncertainty in the expected value and the sampling distribution of values around the expected value.

Using this distributional assumption, we can estimate the proportion of fish of weight w that fall into the length interval $[l_a, l_b]$ by using the appropriate c.d.f., for the t distribution Φ_{l_w}

$$P(l_w \in [l_a, l_b]) = \Phi_{l_w}(l_b) - \Phi_{l_w}(l_a)$$

An implementation of this procedure in S is shown in Box 1.

```
function(val, wl.nls, freq = rep(1, length(val)), int = 2, nint = 20)
{
#--------------------------------------------------------------------
# This function takes a vector of values (val) of weight and converts it
# into an array of categories of predicted values, according to the function
# implied in wl.nls and the interval for category (int) and the number of
# categories for each predicted value (nint).
# The number of observations in each category is given by freq.
#
#
        prediction <- predict(wl.nls, list(Weight = val), se.fit = T)
        lt <- prediction$fit
        var.mean <- (prediction$se.fit)^2
        var.sampl <- (prediction$residual.scale)^2
        sd <- sqrt(var.mean + var.sampl)
        df <- prediction$df      #
#
# This is the low limit of the predicted lt class.
#
        lt.low <- int * trunc(lt/int)     #
#
# Two cases : either val is a scalar or a vector.
#
        n <- length(lt.low)
        if(n == 1) {
                ltt <- seq((lt.low - int * (nint/2)), by = int, length = nint)
                pp <- diff(pt((ltt - lt)/sd, df))
                pp <- pp * freq
                cat(ltt, "\n")
                names(pp) <- ltt[-1] - int/2
        }
        else if(n > 1) {
                prdmtx <- clssmtx <- matrix(0, nrow = n, ncol = nint - 1)
                for(i in 1:n) {
                        tgt <- seq((lt.low[i] - int * (nint/2)), by = int, length = nint)
                        prdmtx[i,  ] <- freq[i] * diff(pt((tgt - lt[i])/sd, df))
                        ss <- sum(prdmtx[i,  ])/freq[i]
                        if(ss < 0.95)
                                warning(" Coverage < 0.95. Increase number of intervals.")
                        clssmtx[i,  ] <- tgt[-1] - int/2
                }
                pp <- tapply(prdmtx, clssmtx, sum)
        }
        pp
 }
```

This function is implemented as function `predlt` in the IOTC S library.

Note that the application of this transformation will result in a length distribution that will appear smoothed out. Also note that the S code could be used with a small modification to the call to `predict` to get distributions for any non-linear function.