Chapter 7 Interval Beta Regression Model Code

The code for estimating a heteroscedastic Tobit model in R and Stata is provided in this document, along with an annotated demonstration of it using the reading accuracy scores example from chapters 6 and 7.

Code in R:

Functions and their usage

The log-likelihood function `betaint` receives a vector of starting values *h*, dependent vector *y2* ≥ *y1* representing the upper and lower bounds, an indicator variable *t* distinguishing precise from imprecise observations, a vector of predictors *x* for the location submodel, and a vector of predictors *z* for the dispersion submodel.

```
# This function computes the log-likelihood, and allows for a mix of
  precise and interval data.
betaint <- function(h, y1, y2, t, x, z)
{
hx = x%*%h[1: length(x[1,])]
mu = exp(hx)/(1+exp(hx))
gz = z%*%h[length(x[1,])+1: length(z[1,])]
phi = exp(gz)
      loglik = t*log(pbeta(y2,mu*phi,(1-mu)*phi) - pbeta(y1,mu*phi,(1-
mu)*phi)) + (1-t)*log(dbeta(y1,mu*phi,(1-mu)*phi))
      -sum(loglik, na.rm = TRUE)
}
```

The next set of commands illustrates a two-step estimation procedure that we recommend for estimating these models. First, we assume that commands have been issued in R that construct or identify the vectors and/or variables `start`, `xdata`, `y1data`, `y2data`, `tdata`, and `zdata`. Then these vectors are passed along with the function `htobit` to `optim` using the Nelder-Mead algorithm for estimation. The parameter estimates from this step are substituted for `start` in the second step, which re-estimates the model using the BFGS algorithm.

```
# The maximum likelihood estimation proceeds in two stages. The first is a
  “rough” but reliable estimator:
> betaopt0 <- optim(start, betaint, hessian = T, x = xdata, y1 = y1data, y2
= y2data, t = tdata, z = wdata, method = "Nelder-Mead")
# We use the parameter estimates from betaopt0 as new starting values, and
  then re-estimate the model with a more fine-tuned method:
> start <- betaopt0$par
> betaopt1 <- optim(start, betaint, hessian = T, x = xdata, y1 = y1data, y2
= y2data, t = tdata, z = wdata, method = "BFGS")
```

Probability judgment example

```
# Assuming that you’ve already loaded up the intervalbeta data file and
attached it.
library(MASS)
# Get starting values for a null model:
midy <- (y1+y2)/2
mumid <- mean(midy); varmid <- var(midy); premid <- mumid*(1-mumid)/varmid
- 1
start <- c(log(mumid/(1-mumid)), log(premid));
# Assign the variables.
y1data <- y1; y2data <- y2;
# This pair of commands tells which are precise and which are interval
  data:
tdata <- c(rep(1,length(y1data)));
```

```
for (i in 1:length(y1data)) if(y1data[i] == y2data[i]) tdata[i] = 0;
const <- rep(1,length(y1data));
xdata <- cbind(const); wdata <- cbind(const);
#
# Run null model:
> betaopt0 <- optim(start, betaint, hessian = T, x = xdata, y1 = y1data, y2
= y2data, t = tdata, z = wdata, method = "Nelder-Mead")
> betaopt0$convergence
[1] 0
> start <- betaopt0$par
> betaopt1 <- optim(start, betaint, hessian = T, x = xdata, y1 = y1data, y2
= y2data, t = tdata, z = wdata, method = "BFGS")
> betaopt1$convergence
[1] 0
> betaopt1$value
[1] 202.6898
> outopt1 <- rbind(estim <- betaopt1$par, serr <-
sqrt(diag(solve(betaopt1$hessian))), zstat <- estim/serr, prob <- 2*(1-
pnorm(abs(zstat)))); row.names(outopt1) <- c("estim", "serr", "zstat",
"prob"); outopt1
             [,1]       [,2]
estim  1.73317717  1.8408218
serr   0.08285678  0.1252639
zstat 20.91774823 14.6955437
prob   0.00000000  0.0000000
#
# Now include the experimental condition variables:
> xdata <- cbind(const, t, n, w);
> start <- c(1.73317717, 0.1, 0.1, -0.1, 1.8408218)
#
# Run the new model:

> betaopt0 <- optim(start, betaint, hessian = T, x = xdata, y1 = y1data, y2
= y2data, t = tdata, z = wdata, method = "Nelder-Mead")
> betaopt0$convergence
[1] 0
> start <- betaopt0$par
> betaopt1 <- optim(start, betaint, hessian = T, x = xdata, y1 = y1data, y2
= y2data, t = tdata, z = wdata, method = "BFGS")
> betaopt1$convergence
[1] 0
> betaopt1$value
[1] 199.122
> outopt1 <- rbind(estim <- betaopt1$par, serr <-
sqrt(diag(solve(betaopt1$hessian))), zstat <- estim/serr, prob <- 2*(1-
pnorm(abs(zstat)))); row.names(outopt1) <- c("estim", "serr", "zstat",
"prob"); outopt1
            [,1]      [,2]      [,3]      [,4]       [,5]
estim  1.5746681 0.1080359 0.5236790 0.1113009  1.8852989
serr   0.1378468 0.1861529 0.2089322 0.2023874  0.1260908
zstat 11.4233177 0.5803612 2.5064537 0.5499399 14.9519140
prob   0.0000000 0.5616710 0.0121949 0.5823606  0.0000000
#
# Model comparison:
> 1 - pchisq(2*(202.6898-199.122),3)
[1] 0.06769906
# Model improvement not quite significant.
#
# Try a model with predictors in the precision submodel:
wdata <- cbind(const, t, n, w); xdata <- cbind(const);
> start <- c(1.73317717, 1.8408218, 0.1, 0.1, -0.1)
```

```
#
# Run the new model:
> betaopt0 <- optim(start, betaint, hessian = T, x = xdata, y1 = y1data, y2
= y2data, t = tdata, z = wdata, method = "Nelder-Mead")
> betaopt0$convergence
[1] 0
> start <- betaopt0$par
> betaopt1 <- optim(start, betaint, hessian = T, x = xdata, y1 = y1data, y2
= y2data, t = tdata, z = wdata, method = "BFGS")
> betaopt1$convergence
[1] 0
> betaopt1$value
[1] 201.1213
# Clearly no significant improvement in model fit.
#
# Try model with just the narrow treatment effect:
> xdata <- cbind(const, n); wdata <- cbind(const);
> start <- c(1.5746681, 0.5236790, 1.8852989)
#
# Run the new model:
> betaopt0 <- optim(start, betaint, hessian = T, x = xdata, y1 = y1data, y2
= y2data, t = tdata, z = wdata, method = "Nelder-Mead")
> betaopt0$convergence
[1] 0
> start <- betaopt0$par
> betaopt1 <- optim(start, betaint, hessian = T, x = xdata, y1 = y1data, y2
= y2data, t = tdata, z = wdata, method = "BFGS")
> betaopt1$convergence
[1] 0
> betaopt1$value
[1] 199.3383
# Model comparison:
> 1 - pchisq(2*(202.6898-199.3383),1)
[1] 0.009625083
> outopt1 <- rbind(estim <- betaopt1$par, serr <-
sqrt(diag(solve(betaopt1$hessian))), zstat <- estim/serr, prob <- 2*(1-
pnorm(abs(zstat)))); row.names(outopt1) <- c("estim", "serr", "zstat",
"prob"); outopt1
             [,1]       [,2]       [,3]
estim  1.64588585 0.45268298  1.8852842
serr   0.08740499 0.18000828  0.1262281
zstat 18.83057073 2.51478982 14.9355398
prob   0.00000000 0.01191034  0.0000000
#
# The narrow condition resulted in a higher mean than the others.
```

Code in Stata:

Functions and their usage

The log-likelihood program `betaint` receives a vector of starting values *h*, dependent vector *y2* ≥ *y1* representing the upper and lower bounds, a vector of predictors *Xb* for the location submodel, and a vector of predictors *Xd* for the dispersion submodel.

```
// This program computes the log-likelihood.
// It also allows for a mix of precise and interval-valued data.
capture program drop betaint
program define betaint
args lnf Xb Xd
tempvar phi mu
quietly {
gen double `phi' = exp(`Xd')
```

```
gen double `mu' = exp(`Xb')/(1+exp(`Xb'))
replace `lnf' = ln(ibeta(`mu'*`phi',(1-`mu')*`phi',$ML_y2) -
ibeta(`mu'*`phi',(1-`mu')*`phi',$ML_y1)) if $ML_y2-$ML_y1 > 0
replace `lnf' = ln(betaden(`mu'*`phi',(1-`mu')*`phi',$ML_y1)) if $ML_y2-
$ML_y1 == 0
  }
end
```

Probability judgment example

```
// Try a null model first:
ml model lf betaint (muvar: y1 y2 = ) (phivar: )
ml search
initial:       log likelihood = -338.97394
alternative:   log likelihood = -280.85834
rescale:       log likelihood = -208.01541
rescale eq:    log likelihood = -208.01541
ml max
                                                Number of obs   =        220
                                                Wald chi2(0)    =          .
Log likelihood = -202.68978                     Prob > chi2     =          .
------------------------------------------------------------------------------
             |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
muvar        |
       _cons |   1.733142   .0828565    20.92   0.000     1.570746    1.895538
-------------+----------------------------------------------------------------
phivar       |
       _cons |    1.84083   .1252654    14.70   0.000     1.595315    2.086346
------------------------------------------------------------------------------
// This agrees with the R routine's output.
// Now try the Narrow condition only model:
ml model lf betaint (muvar: y1 y2 = n) (phivar: )
ml search
ml max
                                                Number of obs   =        220
                                                Wald chi2(1)    =       6.33
Log likelihood = -199.33828                     Prob > chi2     =     0.0119
------------------------------------------------------------------------------
             |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
muvar        |
           n |   .4528326   .1800152     2.52   0.012     .1000094    .8056559
       _cons |   1.645862   .0874051    18.83   0.000     1.474551    1.817173
-------------+----------------------------------------------------------------
phivar       |
       _cons |   1.885269   .1262282    14.94   0.000     1.637866    2.132672
------------------------------------------------------------------------------
// This also agrees with the R routine output.
```