Universidad de Castilla-La Mancha

SEGUIMIENTO DE LA DIVERSIDAD BIOLÓGICA

Modelos Cormack-Jolly-Seber (CJS)

José Jiménez García-Herrera (IREC-CSIC)

Universidad de Castilla-La Mancha

Este capítulo se va a centrar en uno de los principales componentes de la dinámica de poblaciones: la probabilidad supervivencia, que es un parámetro demográfico clave suele ser trascendente en la dinámica de poblaciones (Clobert y Lebreton, 1991; Saether y Bakke, 2000). Normalmente, el interés se centra en su estima, testar posibles covariables y hacer comparativas o predicciones.

El método estadístico más común para estimar conjuntamente las probabilidades de recaptura y supervivencia en poblaciones animales y vegetales es a través de un tipo de modelos abiertos de captura-recaptura, a la que pertenece el modelo Cormack-Jolly-Seber (CJS) (Cormack, 1964; Jolly, 1965; Seber, 1965). Los reencuentros pueden obtenerse por diferentes métodos (captura física, avistamientos, seguimiento genético), pero la clave es que los individuos se identifiquen sin error. Es decir, sólo tenemos falsos negativos, pero nunca falsos positivos. El análisis frecuentista del modelo CJS se describe con detalle en Lebreton et al. (1992). El modelo bayesiano que utilizaremos en el curso es el más básico (espacio de estados) y está extraído Bayesian population analysis using WinBUGS. A hierarchical perspective. Bayesian Population Analysis using WinBUGS (BPA). Academic Press / Elsevier. http://doi.org/10.1016/B978-0-12-387020-9.00014-6.

El diseño del muestreo es el siguiente: Se captura una muestra aleatoria de individuos de la población de estudio, se marcan todos individualmente y se liberan de nuevo en la población. Esto se repite varias veces (por ejemplo, varios años). Algunos individuos marcados se volverán a encontrar, con lo que se obtienen datos de captura-recaptura que pueden resumirse en historiales de captura individuales.

Repetid los análisis que se ejecutan aquí. Para ampliar conocimientos resulta adecuado consultar y estudiar BPA.

1. Simulación de datos

Vamos a simular datos en R(R Core Team, 2020) y a ejecutar el modelo CJS utilizando JAGS (Plummer, 2003). El nivel de precisión que podemos obtener con estas estimas va a depender en buena parte del número de recapturas, que a su vez depende de la detectabilidad (p) y el número de ocasiones de muestreo.

```
> # Modelo con parámetros constantes
> # Definimos los valores de parámetros
> set.seed(1960)
> n.occasions <- 10                  # Número de ocasiones captura
> marked <- rep(50, n.occasions-1)  # Número anual de nuevos individuos marcados
> phi <- rep(0.65, n.occasions-1)
> p <- rep(0.5, n.occasions-1)
> # Definimos matrices con probabilidades de supervivencia y recaptura
> PHI <- matrix(phi, ncol = n.occasions-1, nrow = sum(marked))
> P <- matrix(p, ncol = n.occasions-1, nrow = sum(marked))
> # Definimos funciones para simular la matriz de historias de captura (CH)
> simul.cjs <- function(PHI, P, marked){
+   n.occasions <- dim(PHI)[2] + 1
+   CH <- matrix(0, ncol = n.occasions, nrow = sum(marked))
+   # Definimos un vector con la ocasión de marcado
+   mark.occ <- rep(1:length(marked), marked[1:length(marked)])
+   # Llenamos la matriz CH
+   for (i in 1:sum(marked)){
+     CH[i, mark.occ[i]] <- 1       # Escribimos un 1 en la ocasión de suelta
+     if (mark.occ[i]==n.occasions) next
+     for (t in (mark.occ[i]+1):n.occasions){
+       # Bernoulli: ¿El individuo sobrevive en la ocasión?
+       sur <- rbinom(1, 1, PHI[i,t-1])
+       if (sur==0) break  # Si muere, nos vamos al siguiente individuo
+       # Bernoulli: ¿Se recaptura el individuo?
+       rp <- rbinom(1, 1, P[i,t-1])
+       if (rp==1) CH[i,t] <- 1
+     } #t
+   } #i
+   return(CH)
+ }
```

```
> # Ejecutamos la función
> CH <- simul.cjs(PHI, P, marked)
```

Vamos a ver los historiales de captura que hemos creado.

```
> head(CH)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    1    1    0    0    0    0    0    0    0     0
[2,]    1    0    0    0    0    0    0    0    0     0
[3,]    1    1    1    0    1    0    0    0    0     0
[4,]    1    0    0    0    0    0    0    0    0     0
[5,]    1    0    1    1    0    0    0    0    0     0
[6,]    1    0    0    0    0    0    0    0    0     0
```

Creamos un vector con la ocasión de marcaje y lo vemos:

```
> get.first <- function(x) min(which(x!=0))
> f <- apply(CH, 1, get.first)
> f
```

```
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [38] 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [75] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3
[112] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
[149] 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
[186] 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
[223] 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 6 6 6 6 6 6 6 6
[260] 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
[297] 6 6 6 6 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
[334] 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8
[371] 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 9 9 9 9 9 9
[408] 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
[445] 9 9 9 9 9 9
```

2. Preparación del código

Especificamos el modelo en BUGS

```
> cat(file = "model.txt",
+ "
+ model {
+
+   # Priori y restricciones
+   for (i in 1:nind){
+     for (t in f[i]:(n.occasions-1)){
+       phi[i,t] <- mean.phi
+       p[i,t] <- mean.p
+     } #t
+   } #i
+
+   mean.phi ~ dunif(0, 1)         # Prior para la supervivencia media
+   mean.p ~ dunif(0, 1)           # Prior para la recaptura media
+
+   # Definimos la probabilidad
+   for (i in 1:nind){
+     # Definimos el estado latente en la primera captura
+     z[i,f[i]] <- 1
+     for (t in (f[i]+1):n.occasions){
+       # Proceso de estado
+       z[i,t] ~ dbern(mu1[i,t])
+       mu1[i,t] <- phi[i,t-1] * z[i,t-1]
+       # Proceso de observación
+       y[i,t] ~ dbern(mu2[i,t])
+       mu2[i,t] <- p[i,t-1] * z[i,t]
+     } #t
+   } #i
+
+ }
+ ")
```

3. Preparación de datos e inicios

Hay información sobre z que la conocemos, y esa será tratado como un dato. Para ello la vamos a preparar con este script:

```
> known.state.cjs <- function(ch){
+   state <- ch
+   for (i in 1:dim(ch)[1]){
+     n1 <- min(which(ch[i,]==1))
+     n2 <- max(which(ch[i,]==1))
+     state[i,n1:n2] <- 1
+     state[i,n1] <- NA
+   }
+   state[state==0] <- NA
+   return(state)
+ }
```

Con eso ya podemos preparar los datos

```
> jags.data <- list(y = CH,    # Matriz de historias de recapturas
+                   f = f,     # vector con la ocasión de marcaje de cada animal
+                   nind = dim(CH)[1], # total de individuos marcados
+                   n.occasions = dim(CH)[2], # ocasiones de recaptura
+                   z = known.state.cjs(CH))  # estados conocidos de z
```

Preparación de inicios:

```
> # Función para crear una matriz de valores iniciales para el estado latente z
> cjs.init.z <- function(ch,f){
+   for (i in 1:dim(ch)[1]){
+     if (sum(ch[i,])==1) next
+     n2 <- max(which(ch[i,]==1))
+     ch[i,f[i]:n2] <- NA
+   }
+   for (i in 1:dim(ch)[1]){
+   ch[i,1:f[i]] <- NA
+   }
+   return(ch)
+ }
> # Valores iniciales
```

```
> inits <- function(){list(z = cjs.init.z(CH, f),
+                          mean.phi = runif(1, 0, 1),
+                          mean.p = runif(1, 0, 1))}
```

4. Compilación y ejecución

```
> # Parámetros a monitorizar
> parameters <- c("mean.phi", "mean.p")
> # Configuración MCMC
> ni <- 10000
> nt <- 5
> nb <- 5000
> nc <- 3
> library(jagsUI)
> # Llamamos a JAGS desde R
> cjs <- jags(jags.data, inits, parameters, "model.txt", n.chains = nc,
+   n.thin = nt, n.iter = ni, n.burnin = nb, parallel=TRUE)

Processing function input.......

Done.

Beginning parallel processing using 3 cores. Console output will be suppressed.

Parallel processing completed.

Calculating statistics.......

Done.
```

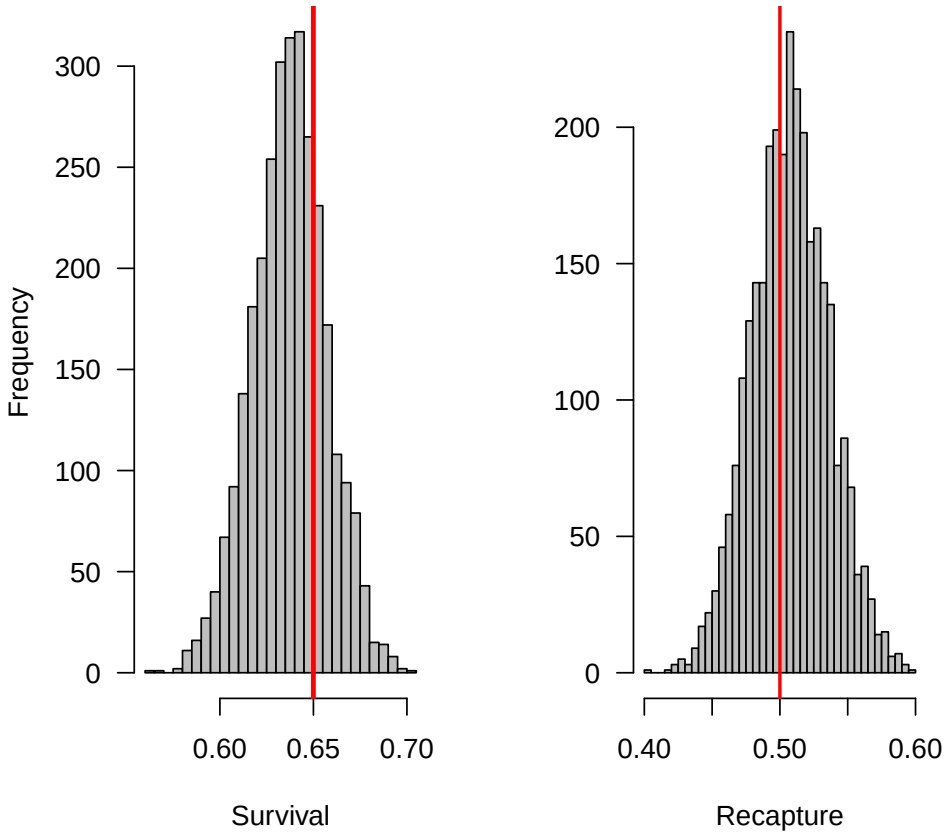
Vemos el resumen

```
> print(cjs, digits = 3)

JAGS output for model 'model.txt', generated by jagsUI.
Estimates based on 3 chains of 10000 iterations,
adaptation = 100 iterations (sufficient),
burn-in = 5000 iterations and thin rate = 5,
yielding 3000 total samples from the joint posterior.
```

```
MCMC ran in parallel for 0.469 minutes at time 2024-02-08 20:05:42.940658.

             mean     sd     2.5%      50%    97.5% overlap0 f  Rhat n.eff
mean.phi    0.637  0.020    0.597    0.638    0.676    FALSE 1 1.000  3000
mean.p      0.508  0.029    0.453    0.508    0.565    FALSE 1 1.001  1400
deviance 1293.466 28.950 1236.611 1292.821 1350.001    FALSE 1 1.000  3000

Successful convergence based on Rhat values (all < 1.1).
Rhat is the potential scale reduction factor (at convergence, Rhat=1).
For each parameter, n.eff is a crude measure of effective sample size.

overlap0 checks if 0 falls in the parameter's 95% credible interval.
f is the proportion of the posterior with the same sign as the mean;
i.e., our confidence that the parameter is positive or negative.

DIC info: (pD = var(deviance)/2)
pD = 419.3 and DIC = 1712.766
DIC is an estimate of expected predictive error (lower is better).
```

Veamos el resultado en histogramas:

```
> par(mfrow = c(1, 2), las = 1)
> hist(cjs$sims.list$mean.phi, nclass = 30, col = "gray", main = "",
+      xlab = "Survival", ylab = "Frequency")
> abline(v = phi, col = "red", lwd = 2)
> hist(cjs$sims.list$mean.p, nclass = 30, col = "gray", main = "",
+      xlab = "Recapture", ylab = "")
> abline(v = p[1], col = "red", lwd = 2)
```


Frequency
300
250
200
150
100
50
0
0.60
0.65
0.70
Survival
200
150
100
50
0
0.40
0.50
0.60
Recapture

5. REFERENCIAS

Clobert, J., Lebreton, J.D., 1991. Estimation of demographic parameters in bird populations. In: Perrins, C.M. (Ed.), Bird Population Studies. Oxford University Press, Oxford, pp. 75–104.

Cormack, R.M., 1964. Estimates of survival from the sighting of marked animals. Biometrika 51, 429–438.

Jolly, G.M., 1965. Explicit estimates from capture-recapture data with both death and immigration-stochastic model. Biometrika 52, 225–247.

Kéry M, Schaub M (2012) Bayesian population analysis using WinBUGS. A hierarchical perspective. Academic Press / Elsevier

Lebreton, J.D., Burnham, K.P., Clobert, J., Anderson, D.R., 1992. Modeling survival and testing biological hypothesis using marked animals: a unified approach with case studies. Ecol. Monogr. 62, 67–118.

Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. 3rd International Workshop on Distributed Statistical Computing (DSC 2003). Vienna, Austria. Retrieved from http://www.ci.tuwien.ac.at/Conferences/DSC-2003/Proceedings/

R Core Team. (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing. Vienna, Austria. Retrieved from https://www.r-project.org/.

Saether, B.E., Bakke, O., 2000. Avian life history variation and contribution of demographic traits to the population growth rate. Ecology 81, 642–653.

Seber, G.A.F., 1965. A note on the multiple recapture census. Biometrika 52, 249–259.