

SEGUIMIENTO DE LA DIVERSIDAD BIOLÓGICA

Modelos de población integrados (IPM)

José Jiménez García-Herrera (IREC-CSIC)

Universidad de Castilla-La Mancha

Repetid primero los análisis que se ejecutan aquí. Pegad los resultados en un procesador de texto (resultados numéricos y gráficos) e incluid también vuestros propios comentarios sobre cuales serían las actuaciones de conservación más efectivas, y la probable evolución futura de la población. Enviadme vuestros trabajos por e-mail a: Jose.Jimenez@csic.es.

Información sobre códigos

Los códigos originales son del libro de Kéry & Schaub *Bayesian Population Analysis using WinBUGS - A hierarchical perspective* (2012). Editorial Elsevier. Estos códigos se han modificado y adaptado para su funcionalidad en Nimble. Se han añadido además scripts para ejecutar algunas funciones del paquete *popbio* y hacer proyecciones poblacionales.

Modelos de Población Integrados (IPM)

Los Modelos de Población Integrados (IPM) son modelos relativamente recientes que usan diferentes tipos de datos de forma combinada para estimar simultáneamente evoluciones de tamaños de población y parámetros demográficos, trabajando sobre matrices de población y bajo detección imperfecta.

Ventajas de su uso:

1. Se incrementa la precisión de la estima de los parámetros demográficos.
2. Aunque faltan algunos datos explícitos (como la productividad) es posible su estima como "variables latentes" (no observadas).
3. Un análisis combinado de datos para inferir parámetros demográficos y tamaño de población nos permite un estudio simultáneo de todos los procesos que determinan la población (parámetros demográficos) y el resultado de esos procesos (tamaño de población) lo cual es de extraordinario interés -por ejemplo, en el ámbito de la biología de la conservación- para investigar las causas demográficas de los cambios poblacionales.

Un interesante desarrollo de los IPM son los Modelos de Población Integrados Espacialmente Explícitos, desarrollados por Chandler & Clark (2014) que añaden la información espacial a la ya señalada. Ello permite aplicar los IPM, por ejemplo, al estudio de metapoblaciones, conectividad, etc. Hay tres pasos para construir el IPM:

- **Definición del vínculo entre la evolución de los tamaños de población y los parámetros demográficos.** Usamos para ello modelos matriciales de población (por edades o por estados), siempre basados en hembras.
- **Definir la probabilidad de cada conjunto de datos**
 1. Probabilidad de los datos de conteo; describe el proceso real -tamaño de población- y permite la separar de la variabilidad real (o ecológica) de la variabilidad del proceso de observación.
 2. Probabilidad de la captura-recaptura: aquí se usa un modelo Cormack-Jolly-Seber. En otros casos se puede usar un modelo multiestado (Margalida et al., 2020).
 3. Probabilidad del éxito reproductor. Usamos una regresión de Poisson.
- **Formular la probabilidad conjunta.** Para ello asumimos independencia entre los conjuntos de datos

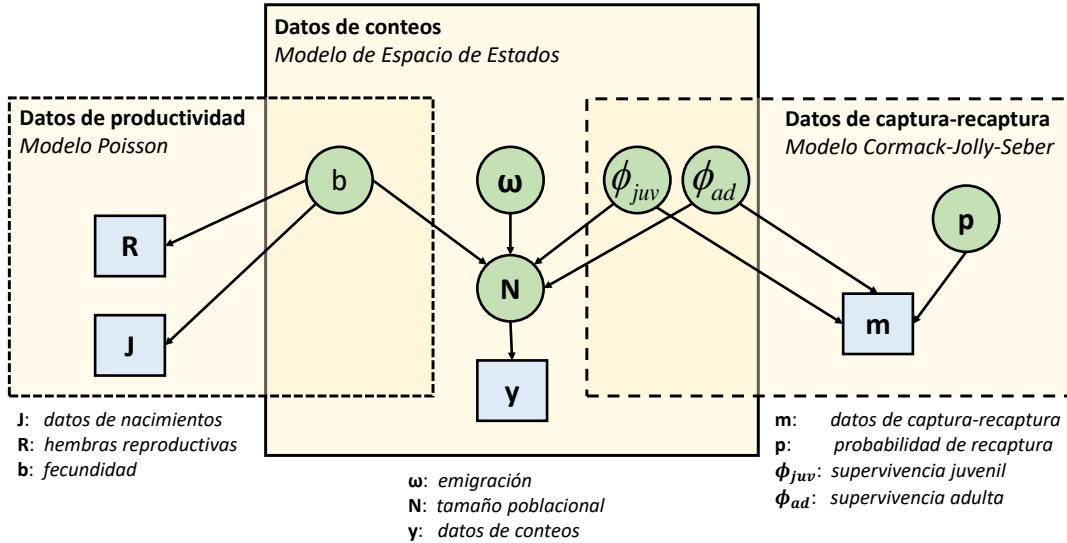


Figura 1: Estructura (DAG) de un IPM. Los cuadrados azules representan los datos utilizados, y los círculos verdes, los parámetros que se estiman.

1. Datos del modelo

He simulado datos similares a los que hemos visto en Lab 2 sobre una población de *Strix occidentalis caurina*. Simulamos un estudio de 10 años de duración, con un intenso seguimiento. Cada año se registran los machos adultos usando reclamos, se localizan nidos con puestas y el número de pollos que vuelan por nido (solo se reproducen los adultos). Tanto los pollos como los adultos son capturados y anillados. Por lo tanto, tenemos tres conjuntos de datos que contienen información sobre la dinámica de esta población: conteos (número de machos adultos y), datos de fecundidad (nidos localizados R y número total de pollos que vuelan R por nido) y datos de supervivencia (se van a extraer de la captura-recaptura de juveniles y adultos). Para simplificar he utilizado sólo dos clases de edad, adultos y jóvenes, y los parámetros de la simulación son: 0.36 para la supervivencia de los juveniles, 0.93 para la supervivencia de los adultos, 0.8 para la probabilidad de recaptura y 0.24 para la

productividad. Asumimos que los parámetros son constantes a lo largo del tiempo.

Usamos un modelo pre-reproducción, y asumimos que los individuos se empiezan a reproducir al llegar a la edad adulta, y vamos a distinguir a efectos del modelo dos clases de edad: de 1 año (*juv*), y de más de 1 año (*ad*).

```
> library(nimble)
> library(popbio)
> library(lattice)
> library(coda)
```

Vamos a calcular la tasa de crecimiento poblacional:

```
> set.seed(1945)
> sjuv=0.36; sad=0.93; f11=0; f12=0.24; vr<-list(sjuv=0.36, sad=0.93, f12=0.48)
> stages <- c("juvenil","adulto")
> post <- expression( matrix2(c(
+           0,      sad*f12/2, # Aquí se usa la producción de hembras (50%
+           sjuv,    sad), stages ))
> (A <- eval(post, vr))

      juvenil   adulto
juvenil     0.00 0.2232
adulto      0.36 0.9300

> lambda(A)
[1] 1.009589
```

Vamos a ver ahora los datos que usaremos en el modelo. El primer conjunto de datos es el de conteo de machos localizados con reclamo (asumimos, por el conocimiento de la biología de la especie, que hay idéntico número de hembras reproductoras):

```
> # Conteo de machos adultos (desde el año 1 al 10)
> y <- c(42, 36, 35, 33, 36, 37, 39, 40, 43, 39)
```

El segundo conjunto de datos son los de productividad. Se han tomado en el seguimiento de nidos con puesta. Son el número de pollos que vuelan (*J*) y el número de nidos (*R*):

```
> # Datos de productividad (desde el año 1 al 9)
> J <- c(4, 12, 20, 12, 16, 18, 4, 26, 26, 10)  # Pollos volados
> R <- c(37, 38, 36, 32, 38, 35, 36, 48, 41, 35) # Nidos localizados
```

El tercer conjunto de datos es el de captura-recaptura:

```
> # Datos de captura-recaptura (en formato m-array, desde el año 1 a 10)
> m<-matrix(c(1, 0, 0, 0, 0, 0, 0, 0, 0, 2,
+           0, 3, 0, 0, 0, 0, 0, 0, 0, 2,
+           0, 0, 5, 0, 0, 0, 0, 0, 0, 5,
+           0, 0, 0, 1, 0, 0, 0, 0, 0, 2,
+           0, 0, 0, 0, 2, 0, 0, 0, 0, 3,
+           0, 0, 0, 0, 0, 1, 0, 0, 0, 5,
+           0, 0, 0, 0, 0, 0, 2, 1, 0, 3,
+           0, 0, 0, 0, 0, 0, 0, 1, 0, 5,
+           0, 0, 0, 0, 0, 0, 0, 0, 2, 6,
+           17, 7, 1, 0, 0, 0, 0, 0, 0, 2,
+           0, 18, 1, 0, 0, 0, 0, 0, 0, 4,
+           0, 0, 25, 2, 1, 0, 0, 0, 0, 1,
+           0, 0, 0, 24, 4, 1, 1, 0, 0, 2,
+           0, 0, 0, 0, 20, 1, 1, 0, 0, 5,
+           0, 0, 0, 0, 0, 19, 4, 1, 1, 2,
+           0, 0, 0, 0, 0, 0, 17, 1, 0, 4,
+           0, 0, 0, 0, 0, 0, 0, 18, 6, 1,
+           0, 0, 0, 0, 0, 0, 0, 0, 19, 3), ncol = 10, byrow = TRUE)
```

Nota: El formato *m-array* es una forma de organización de los datos en una matriz triangular, y sirve para agilizar mucho el proceso de cálculo. En las filas van las ocasiones de liberación, y en las columnas las ocasiones de recaptura. Por último, en la última columna se detallan los individuos que nunca fueron recapturados. Por ejemplo, las historias de captura por individuos:

1	0	1	0
1	1	0	0
1	1	0	0
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0

Se resumen transformándolas en un *m-array*:

Cuadro 1: m-array

Suelta (1)	Recap(2)	Recap(3)	Recap(4)	NoCap
1	2	1	0	1
2	-	1	0	3
3	-	-	1	2

La interpretación es simple. De los 4 individuos capturados y liberados en la primera ocasión Suelta(1) 2 se recapturan en Recap(2), 1 en Recap(3), ninguno en Recap(4), y 1 no se ha capturado en ninguna ocasión (NoCap). Con esta matriz, hemos agregado las cuatro primeras filas de los históricos en la primera fila del *m-array*. Para simplificar la construcción de los *m-array* podemos usar el script específico creado por Kéry & Schaub (2012).

2. Modelo en BUGS utilizando Nimble

```
> code <- nimbleCode({  
+  
+ #-----  
+ # MODELO DE POBLACIÓN INTEGRADO  
+ # - Modelo estructurado por edades con 2 clases:  
+ #   joven (hasta 1 año) y adulto (al menos 2 años)  
+ # - Edad de primera reproducción: 1 año  
+ # - Conteos pre-reproductores (machos territoriales)  
+ # - Todos los ratios vitales se asumen constantes  
+ #-----  
+  
+ #-----  
+ # 1. Definimos la información a priori para los parámetros  
+ #-----  
+ # Error de observación  
+ tauy <- pow(sigma.y, -2)  
+ sigma.y ~ dunif(0, 50)  
+ sigma2.y <- pow(sigma.y, 2)  
+  
+ # Tamaños iniciales de población  
+ n1 ~ T(dnorm(3, tauy), 0, 1000)      # 1-año  
+ nad ~ T(dnorm(35, tauy), 0, 1000)     # Adultos  
+ N1[1] <- round(n1)  
+ Nad[1] <- round(nad)  
+  
+ # Supervivencia, productividad y probabilidad de recaptura  
+ for (t in 1:(nyears-1)){  
+   sjuv[t] <- mean.sjuv  
+   sad[t] <- mean.sad  
+   p[t] <- mean.p  
+   f[t] <- mean.fec  
+ }  
+  
+ mean.sjuv ~ dunif(0, 1)  
+ mean.sad ~ dunif(0, 1)
```

```
+ mean.p ~ dunif(0, 1)
+ mean.fec ~ dunif(0, 20)
+
+ #-----
+ # 2. Parámetros derivados
+ #-----
+ # Ratio de crecimiento poblacional
+ for (t in 1:(nyears-1)){
+   lambda[t] <- Ntot[t+1] / Ntot[t]
+ }
+
+ #-----
+ # 3. Probabilidad de los conjuntos de datos
+ #-----
+ # 3.1. Probabilidad de los datos de conteos
+   # 3.1.1 Proceso de sistema (realidad biológica)
+   for (t in 2:nyears){
+     mean1[t] <- f[t-1] / 2 * sjuv[t-1] * Ntot[t-1]
+     N1[t] ~ dpois(mean1[t])
+     Nad[t] ~ dbin(sad[t-1], Ntot[t-1])
+   }
+   for (t in 1:nyears){
+     Ntot[t] <- Nad[t] + N1[t]
+   }
+
+   # 3.1.2 Proceso de observación
+   for (t in 1:nyears){
+     y[t] ~ dnorm(Ntot[t], tauy)
+   }
+
+ # 3.2 Probabilidad de los datos de captura-recaptura:
+ # modelo CJS con dos clases de edad
+
+ # Probabilidad multinomial
+ for (t in 1:2*(nyears-1)){
+   m[t,1:nyears] ~ dmulti(pr[t,1:nyears], r[t])
+ }
```

```
+  
+ # Probabilidades del m-array para juveniles  
+ for (t in 1:(nyears-1)){  
+   # Diagonal principal  
+   q[t] <- 1-p[t]  
+   pr[t,t] <- sjuv[t] * p[t]  
+   # Por encima de la diagonal principal  
+   for (j in (t+1):(nyears-1)){  
+     pr[t,j] <- sjuv[t]*prod(sad[(t+1):j])*prod(q[t:(j-1)])*p[j]  
+   } #j  
+   # Bajo la diagonal principal  
+   for (j in 1:(t-1)){  
+     pr[t,j] <- 0  
+   } #j  
+   # Última columna: probabilidad de no-recaptura  
+   pr[t,nyears] <- 1-sum(pr[t,1:(nyears-1)])  
+ } #t  
+  
+ # Probabilidades del m-array para adultos  
+ for (t in 1:(nyears-1)){  
+   # Diagonal principal  
+   pr[t+nyears-1,t] <- sad[t] * p[t]  
+   # Por encima de la diagonal principal  
+   for (j in (t+1):(nyears-1)){  
+     pr[t+nyears-1,j] <- prod(sad[t:j])*prod(q[t:(j-1)])*p[j]  
+   } #j  
+   # Bajo la diagonal principal  
+   for (j in 1:(t-1)){  
+     pr[t+nyears-1,j] <- 0  
+   } #j  
+   # Última columna: probabilidad de no-recaptura  
+   pr[t+nyears-1,nyears] <- 1 - sum(pr[t+nyears-1,1:(nyears-1)])  
+ } #t  
+  
+ # 3.3. Probabilidad para datos de productividad: regresión de Poisson  
+ for (t in 1:(nyears-1)){  
+   J[t] ~ dpois(rho[t])
```

```
+      rho[t] <- R[t]*f[t]
+    }
+ })
>
```

Preparación de datos, constantes e inicios:

```
> str (data <- list(m = m,
+                      y = y,
+                      J = J))

List of 3
$ m: num [1:18, 1:10] 1 0 0 0 0 0 0 0 0 17 ...
$ y: num [1:10] 42 36 35 33 36 37 39 40 43 39
$ J: num [1:10] 4 12 20 12 16 18 4 26 26 10

> str(constants<-list(R = R,
+                      nyears = dim(m)[2],
+                      r = rowSums(m)))

List of 3
$ R      : num [1:10] 37 38 36 32 38 35 36 48 41 35
$ nyears: int 10
$ r      : num [1:18] 3 5 10 3 5 6 6 6 8 27 ...

> set.seed(1960)
> N1s<-rep(2, 10)
> Nads<-rep(35, 10)
> Ntots<-N1s+Nads
> str(inits <- list(mean.sjuv = runif(1, 0, 1),
+                      mean.sad = runif(1, 0, 1),
+                      mean.p = runif(1, 0, 1),
+                      mean.fec = runif(1, 0, 10),
+                      sigma.y = runif(1, 0, 1),
+                      n1 = rpois(1, 30),
+                      nad = rpois(1, 30),
+                      N1=N1s,
+                      Nad=Nads,
+                      Ntot=Ntots,
+                      mean1=rep(10,10),
```

```
+           lambda=runif(9,.5,1)))  
  
List of 12  
$ mean.sjuv: num 0.556  
$ mean.sad : num 0.557  
$ mean.p   : num 0.356  
$ mean.fec : num 2.02  
$ sigma.y  : num 0.000728  
$ n1       : int 27  
$ nad      : int 32  
$ N1       : num [1:10] 2 2 2 2 2 2 2 2 2 2  
$ Nad      : num [1:10] 35 35 35 35 35 35 35 35 35 35  
$ Ntot     : num [1:10] 37 37 37 37 37 37 37 37 37 37  
$ mean1    : num [1:10] 10 10 10 10 10 10 10 10 10 10  
$ lambda   : num [1:9] 0.656 0.971 0.548 0.78 0.625 ...
```

Parámetros a monitorizar

```
> params <- c("mean.sjuv", "mean.sad", "mean.p", "mean.fec",  
+   "N1", "Nad", "Ntot", "lambda")
```

Compilación y ejecución

```
> # Preparamos el modelo para ejecución en Nimble  
> Rmodel <- nimbleModel(code=code, constants=constants, data=data,  
+                           inits=inits, check=FALSE)  
> Rmodel$initializeInfo()  
> Cmodel <- compileNimble(Rmodel)  
> # Establecemos los parámetros a monitorizar  
> mcmcSpec<-configureMCMC(Rmodel, monitors=params)  
  
===== Monitors =====  
thin = 1: mean.sjuv, mean.sad, mean.p, mean.fec, N1, Nad, Ntot, lambda  
===== Samplers =====  
slice sampler (18)  
  - Nad[]  (9 elements)  
  - N1[]  (9 elements)  
RW sampler (7)  
  - sigma.y
```


Resultados

```
> summary(outNim)

Iterations = 1:50000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 50000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:
```

	Mean	SD	Naive SE	Time-series SE
N1[1]	3.8748	1.57984	4.079e-03	0.0138890
N1[2]	1.8706	1.31095	3.385e-03	0.0071375
N1[3]	2.1512	1.35731	3.505e-03	0.0072459
N1[4]	2.2651	1.36883	3.534e-03	0.0083316
N1[5]	3.4074	1.53104	3.953e-03	0.0097253
N1[6]	3.3965	1.52146	3.928e-03	0.0092492
N1[7]	3.7044	1.57595	4.069e-03	0.0096256
N1[8]	3.6120	1.58720	4.098e-03	0.0096140
N1[9]	3.8152	1.63850	4.231e-03	0.0096266
N1[10]	2.4495	1.47923	3.819e-03	0.0073470
Nad[1]	35.6635	1.72320	4.449e-03	0.0169176
Nad[2]	34.8979	1.69935	4.388e-03	0.0103311
Nad[3]	33.2869	1.69834	4.385e-03	0.0114765
Nad[4]	32.3309	1.71942	4.440e-03	0.0135890
Nad[5]	32.5606	1.68683	4.355e-03	0.0134553
Nad[6]	33.7715	1.65097	4.263e-03	0.0119229
Nad[7]	35.0026	1.65800	4.281e-03	0.0115189
Nad[8]	36.2947	1.69672	4.381e-03	0.0116563
Nad[9]	37.4639	1.75278	4.526e-03	0.0116076
Nad[10]	37.3219	1.85927	4.801e-03	0.0102670
Ntot[1]	39.5383	1.60316	4.139e-03	0.0112225
Ntot[2]	36.7684	1.48728	3.840e-03	0.0077812
Ntot[3]	35.4381	1.48437	3.833e-03	0.0089573
Ntot[4]	34.5961	1.54767	3.996e-03	0.0121079
Ntot[5]	35.9681	1.44390	3.728e-03	0.0084474

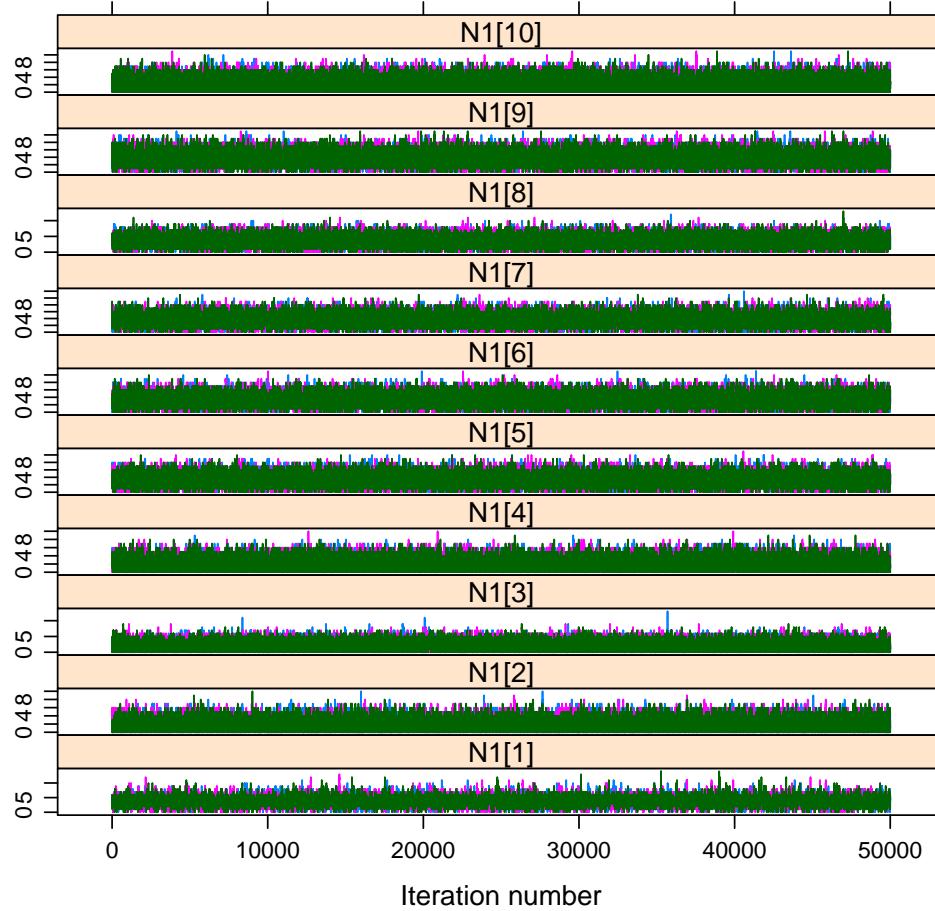
Ntot[6]	37.1681	1.43450	3.704e-03	0.0077911
Ntot[7]	38.7069	1.45716	3.762e-03	0.0077257
Ntot[8]	39.9067	1.48872	3.844e-03	0.0077721
Ntot[9]	41.2790	1.64607	4.250e-03	0.0100764
Ntot[10]	39.7714	1.75647	4.535e-03	0.0071001
lambda[1]	0.9310	0.04482	1.157e-04	0.0002878
lambda[2]	0.9647	0.04336	1.120e-04	0.0001542
lambda[3]	0.9771	0.04483	1.157e-04	0.0001913
lambda[4]	1.0410	0.04780	1.234e-04	0.0002478
lambda[5]	1.0344	0.04492	1.160e-04	0.0001793
lambda[6]	1.0424	0.04458	1.151e-04	0.0001881
lambda[7]	1.0319	0.04302	1.111e-04	0.0001650
lambda[8]	1.0352	0.04336	1.120e-04	0.0001786
lambda[9]	0.9644	0.04562	1.178e-04	0.0002387
mean.fec	0.4072	0.03378	8.721e-05	0.0002036
mean.p	0.8134	0.02514	6.491e-05	0.0001356
mean.sad	0.9211	0.01414	3.651e-05	0.0001216
mean.sjuv	0.3894	0.06129	1.582e-04	0.0004701

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
N1[1]	1.0000	3.0000	4.0000	5.0000	7.0000
N1[2]	0.0000	1.0000	2.0000	3.0000	5.0000
N1[3]	0.0000	1.0000	2.0000	3.0000	5.0000
N1[4]	0.0000	1.0000	2.0000	3.0000	5.0000
N1[5]	1.0000	2.0000	3.0000	4.0000	7.0000
N1[6]	1.0000	2.0000	3.0000	4.0000	7.0000
N1[7]	1.0000	3.0000	4.0000	5.0000	7.0000
N1[8]	1.0000	3.0000	4.0000	5.0000	7.0000
N1[9]	1.0000	3.0000	4.0000	5.0000	7.0000
N1[10]	0.0000	1.0000	2.0000	3.0000	6.0000
Nad[1]	32.0000	35.0000	36.0000	37.0000	39.0000
Nad[2]	31.0000	34.0000	35.0000	36.0000	38.0000
Nad[3]	30.0000	32.0000	33.0000	34.0000	37.0000
Nad[4]	29.0000	31.0000	32.0000	33.0000	36.0000
Nad[5]	29.0000	31.0000	33.0000	34.0000	36.0000

```
Nad[6]    30.0000 33.0000 34.0000 35.0000 37.0000
Nad[7]    32.0000 34.0000 35.0000 36.0000 38.0000
Nad[8]    33.0000 35.0000 36.0000 37.0000 39.0000
Nad[9]    34.0000 36.0000 38.0000 39.0000 41.0000
Nad[10]   33.0000 36.0000 37.0000 39.0000 41.0000
Ntot[1]   36.0000 39.0000 40.0000 41.0000 42.0000
Ntot[2]   34.0000 36.0000 37.0000 38.0000 40.0000
Ntot[3]   33.0000 34.0000 35.0000 36.0000 39.0000
Ntot[4]   32.0000 34.0000 34.0000 35.0000 38.0000
Ntot[5]   33.0000 35.0000 36.0000 37.0000 39.0000
Ntot[6]   34.0000 36.0000 37.0000 38.0000 40.0000
Ntot[7]   36.0000 38.0000 39.0000 40.0000 42.0000
Ntot[8]   37.0000 39.0000 40.0000 41.0000 43.0000
Ntot[9]   38.0000 40.0000 41.0000 42.0000 44.0000
Ntot[10]  36.0000 39.0000 40.0000 41.0000 43.0000
lambda[1] 0.8537 0.9000 0.9250 0.9512 1.0270
lambda[2] 0.8857 0.9444 0.9722 1.0000 1.0571
lambda[3] 0.8919 0.9444 0.9722 1.0000 1.0625
lambda[4] 0.9444 1.0000 1.0303 1.0625 1.1250
lambda[5] 0.9459 1.0000 1.0278 1.0571 1.1212
lambda[6] 0.9500 1.0250 1.0526 1.0789 1.1389
lambda[7] 0.9500 1.0000 1.0263 1.0526 1.1111
lambda[8] 0.9512 1.0000 1.0263 1.0571 1.1250
lambda[9] 0.8837 0.9286 0.9535 1.0000 1.0541
mean.fec 0.3442 0.3839 0.4061 0.4297 0.4756
mean.p   0.7616 0.7971 0.8143 0.8308 0.8603
mean.sad 0.8915 0.9119 0.9217 0.9310 0.9467
mean.sjuv 0.2744 0.3467 0.3882 0.4303 0.5138
```

```
> xyplot(outNim[,1:10])
```



```

> ipm<-as.data.frame(rbind(outNim$chain1,outNim$chain2,outNim$chain2))
> Ntot<-ipm[,21:30]
> head(Ntot)

```

	Ntot[1]	Ntot[2]	Ntot[3]	Ntot[4]	Ntot[5]	Ntot[6]	Ntot[7]	Ntot[8]	Ntot[9]
1	40	36	35	35	37	37	39	38	40
2	40	37	33	36	37	38	38	40	41
3	41	35	35	38	39	40	39	40	41
4	41	37	35	35	37	37	39	40	42
5	41	39	34	36	39	39	39	40	45
6	39	36	35	36	38	38	39	39	42

```
Ntot[10]
1      40
2      39
3      38
4      40
5      40
6      41

> lambda<-ipm[,31:39]
> head(lambda)

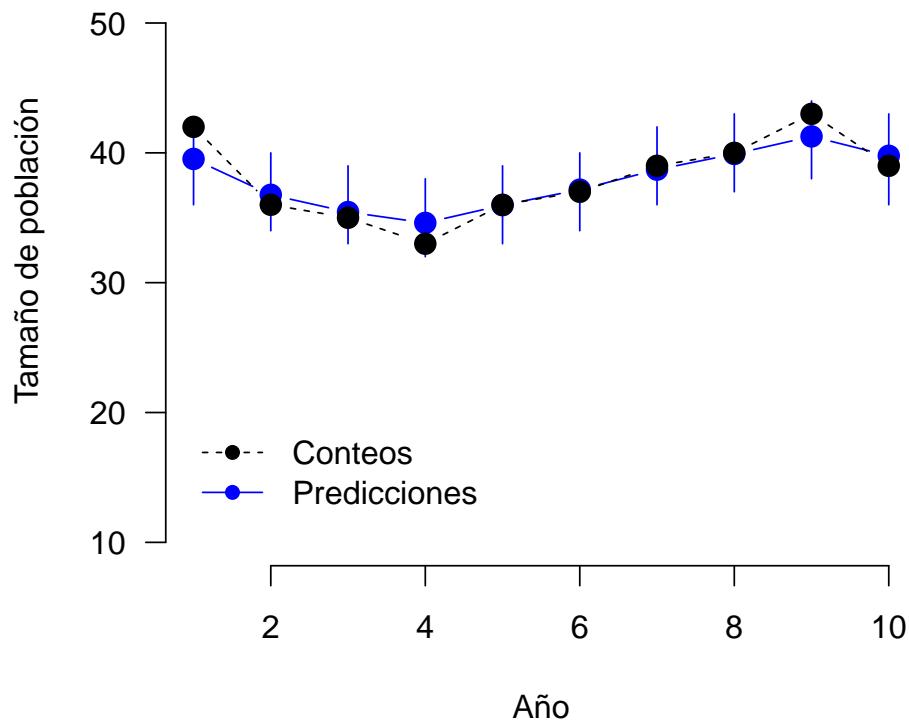
  lambda[1] lambda[2] lambda[3] lambda[4] lambda[5] lambda[6] lambda[7]
1 0.9000000 0.9722222 1.000000 1.057143 1.000000 1.054054 0.974359
2 0.9250000 0.8918919 1.090909 1.027778 1.027027 1.000000 1.052632
3 0.8536585 1.0000000 1.085714 1.026316 1.025641 0.975000 1.025641
4 0.9024390 0.9459459 1.000000 1.057143 1.000000 1.054054 1.025641
5 0.9512195 0.8717949 1.058824 1.083333 1.000000 1.000000 1.025641
6 0.9230769 0.9722222 1.028571 1.055556 1.000000 1.026316 1.000000

  lambda[8] lambda[9]
1 1.052632 1.0000000
2 1.025000 0.9512195
3 1.025000 0.9268293
4 1.050000 0.9523810
5 1.125000 0.8888889
6 1.076923 0.9761905
```

Conteos vs. Predicciones

```
> par(cex = 1.2)
> meanNtot<-apply(Ntot, 2, mean)
> lowerNt <- apply(Ntot, 2, quantile, p=0.025)
> upperNt <- apply(Ntot, 2, quantile, p=0.975)
> plot(meanNtot, type = "b", ylim = c(10, 55),
+       ylab = "Tamaño de población",
+       xlab = "Año",
+       las = 1, pch = 16, col = "blue", frame = F, cex = 1.5)
> segments(1:10, lowerNt, 1:10, upperNt, col = "blue")
> points(y, type = "b", col = "black", pch = 16, lty = 2, cex = 1.5)
> legend(x = 1, y = 20, legend = c("Conteos", "Predicciones"),
```

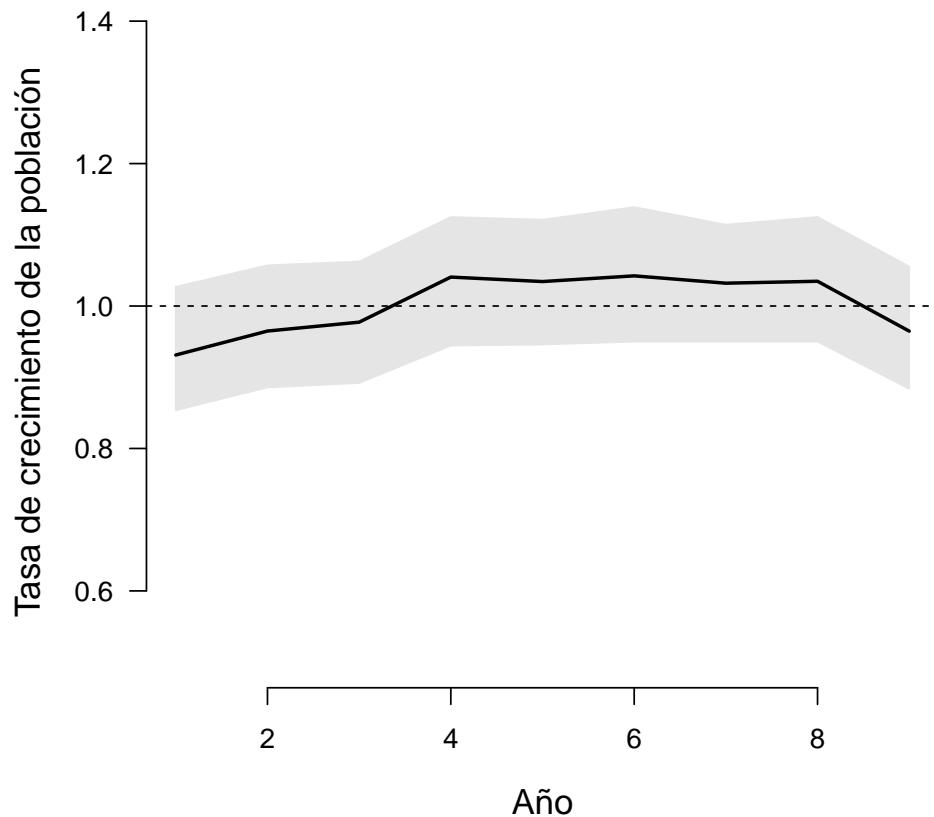
```
+ pch = c(16, 16), col = c("black", "blue"), lty = c(2, 1), bty = "n")
```



Ratio de crecimiento (λ)

```
> meanlambda<-apply(lambda, 2, mean)
> lowerlambda <- apply(lambda, 2, quantile, p=0.025)
> upperlambda <- apply(lambda, 2, quantile, p=0.975)
> plot(1:9, meanlambda, type = "l", ylim = c(0.5, 1.4),
+       ylab = "Tasa de crecimiento de la población",
+       xlab = "Año", las = 1,
+       pch = 16, col = "black",
```

```
+ frame = F, cex.lab=1.25)
> polygon(x = c(1:9, 9:1), y = c(lowerlambda, upperlambda[9:1]),
+ col = "gray90", border = "gray90")
> points(1:9, meanlambda, type = "l", pch = 16, col = "black", cex = 1, lwd=2)
> abline(h=1, lty=2)
```



Vamos a hacer las comprobaciones comparando con la población de partida:

```
> sjuv=0.36; sad=0.93; f=0.48
> A <- matrix(c(
+           0,      sad*f/2,
```

```
+           sjuv,      sad), nrow = 2, byrow = TRUE)
> lambda(A)
```

```
[1] 1.009589
```

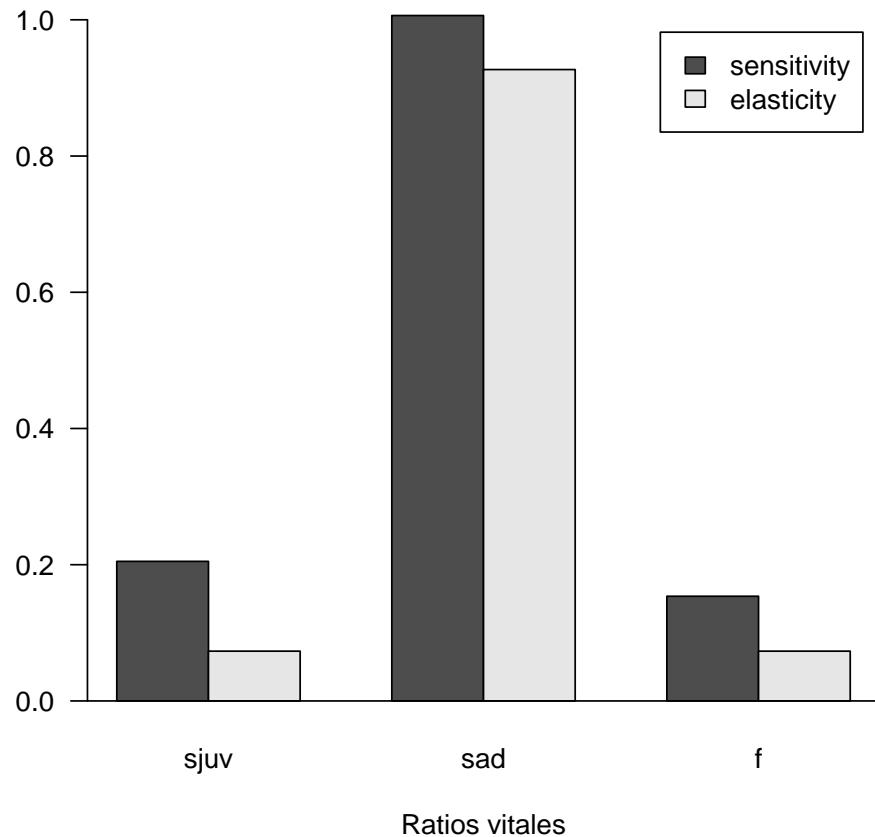
Sensibilidad y elasticidad

```
> vr<-list(sjuv=sjuv, sad=sad, f=f)
> # Creamos la matriz del modelo de población pre-reproducción:
> sp.A <- expression(
+           0,      sad*f/2,
+           sjuv,      sad)
> # vemos la sensibilidad y elasticidad
> x <- vitalsens(sp.A, vr)
> x

      estimate sensitivity elasticity
sjuv      0.36    0.2049252 0.07307241
sad       0.93    1.0062535 0.92692759
f         0.48    0.1536939 0.07307241

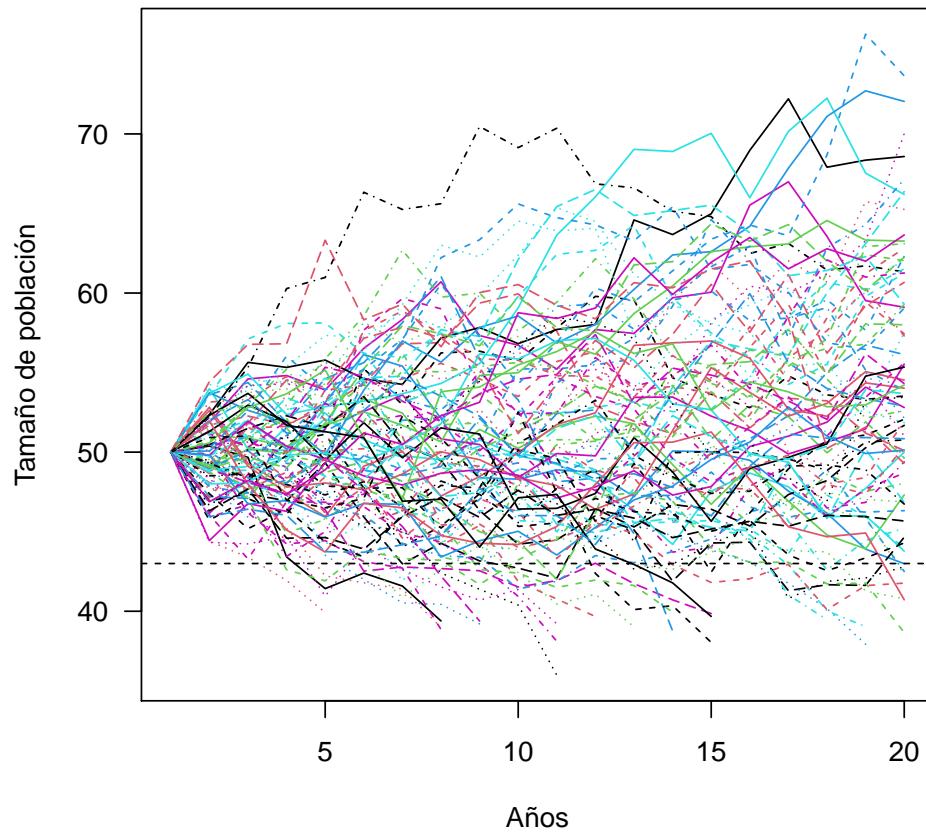
> barplot(t(x[,2:3]), beside=TRUE, legend=TRUE, las=1,
+   xlab="Ratios vitales",
+   main="Sensibilidad y elasticidad de los ratios vitales")
> abline(h=0)
```

Sensibilidad y elasticidad de los ratios vitales



```
> # Vamos a ver que podría pasar con esa población, añadiendo la propia
> # variación que ya presenta lambda, y suponiendo que esa variación sea
> # aleatoria
>
> # Múltiples simulaciones
> sims <- 100
> years<- 20
> lam<-c(0.931,0.9647,0.9771,1.041,1.0344,1.0424,1.0319,1.0352,0.9644)
> meanlog<-mean(log(lam))
> sdlog<-sd(lam)
```

```
> outmat<-matrix(NA,years,sims)
> outmat[1,]<-50
> umbral<- 40
> set.seed(1960)
> for (i in 1:sims){
+   for (t in 2:years){
+     lam<-exp(rnorm(1,meanlog,sdlog))
+     outmat[t,i]<-outmat[(t-1),i]*lam
+     if(outmat[t,i]<=umbral) break
+   }
+ }
> matplot(1:years, outmat, type = "l", las = 1, ylab = "Tamaño de población",
+ +         xlab = "Años")
> abline(h = 43, lty = 2)
```



```
> time<-NULL # creamos un vector contenedor
> for (i in 1:sims){ # bucle sobre las simulaciones
+   t<-max(which(outmat[,i]>0))
+   time<-c(time,t)
+ }
> time.under<-time[which(time<years)]
> (length(time.under)*100/sims)

[1] 19
```

3. REFERENCIAS

- Chandler, R. B., & Clark, J. D. (2014). Spatially explicit integrated population models. *Methods in Ecology and Evolution*, 5(12), 1351–1360. doi:10.1111/2041-210X.12153
- Kéry, M., & Schaub, M. (2012). Bayesian population analysis using WinBUGS. A hierarchical perspective. Academic Press / Elsevier. doi:10.1016/B978-0-12-387020-9.00014-6
- Margalida, A., J. Jiménez, J. M. Martínez, J. A. Sesé, D. García-Ferré, A. Llamas, M. Razin, M. Colomer, and B. Arroyo. 2020. An assessment of population size and demographic drivers of the Bearded Vulture using integrated population models. *Ecological Monographs* 90:e01414; page 1–17.