Simulación de datos de ocupación, bajo el modelo estático (MacKenzie et al. 2002) para el venado de cola blanca en el Parque Nacional Machalilla

Entendiendo las simulaciones y el modelo básico. Gran parte del código y el texto han sido adaptados de los libros de Marc Kery (2010, 2011, y 2015).

Diego J. Lizcano

July, 2016

Contents

1 Porque simular?

1.1 Por que son útiles las simulaciones:

1. Al hacer simulaciones se conocen los parámetros verdaderos, así que podremos asegurarnos que el código que ejecutamos (R o BUGS) estima lo que queremos, y que los estimados son iguales o se acercan a los parámetros verdaderos, permitiendo depurar errores en el código.
2. Podemos calibrar un modelo derivado y/o más complejo más fácilmente. Las simulaciones pueden ser vistas como un experimento controlado, o como versiones simplificadas de un sistema real, en el cual podemos probar como varían ciertos parámetros que afectan los estimados de otros parámetros. Realizar experimentos controlados en el mundo real es muchas veces impractico o imposible en ecología, así que la simulación es la forma mas coherente de estudiar el sistema ecológico.
3. Se experimenta de primera mano el error de muestreo y se convierte en un fantástico proceso de aprendizaje.
4. Podemos verificar la calidad (frecuentista) de los estimados, así como la precisión y el efecto del tamaño muestral, computando la diferencia entre la media del estimado y el valor real (sesgo) y la varianza del estimado (la precisión).
5. Es la forma más flexible y directa de realizar un análisis de poder, resolviendo el gran problema de determinar el tamaño de la muestra necesario para detectar un efecto de cierta magnitud, con una probabilidad dada.

6. Podemos visualizar que tan identificables son los parámetros en modelos más complejos.
7. Podemos verificar que tan robusto es el modelo a violaciones de lo que se asume.
8. Al ser capaces de simular datos bajo cierto modelo, se garantiza que uno entiende el modelo, sus restricciones y limitaciones.

2 La ocupación de hábitat

Obtener datos para estudios de poblaciones animales, es costoso y dispendioso, y no siempre se puede medir la densidad poblacional o parámetros demográficos como natalidad o mortalidad. Es por eso que la estimación de ocupación de hábitat (ψ) es una buena herramienta de estudio, ya que es un reflejo de otros parámetros poblacionales importantes como la abundancia y densidad, que requieren de un elevado número de registros, con los costos económicos y logísticos que conlleva. Adicionalmente y debido a que la detectabilidad (p) en animales silvestres no es completa, el uso de los datos crudos genera subestimaciones de la ocupación de hábitat. Con el empleo de muestreos repetidos, es posible generar estimaciones de detectabilidad y, con esta estimación, obtener valores no sesgados de la ocupación del hábitat. Los métodos de análisis de la ocupación permiten realizar inferencias acerca de los efectos de variables continuas y categóricas sobre la ocupación de hábitat. Además, si los muestreos se realizan a través de períodos largos de tiempo, también es posible estimar tasas de extinción y recolonización, que son útiles en estudios de metapoblaciones. Este es un campo de gran desarrollo en bioestadística que ha producido una gran explosión de estudios que usan la ocupación teniendo en cuenta la detectabilidad.

3 Nuestro ejemplo:

El set de datos que vamos a simular, imita la forma espacial y temporal como imaginamos se originan las medidas repetidas de presencia ausencia en ecología. Las cuales son una combinación de un proceso ecológico y un proceso de observación. El primer proceso contiene los mecanismos bajos los cuales se originan patrones espacio-temporales de distribución, mientras que el segundo proceso contiene las diferentes facetas en las cuales se originan fuentes de error al tomar los datos.

Para ser más concretos vamos a llamar a nuestra especie imaginaria con un nombre real. La llamaremos el venado de cola blanca (*Odocoileus virginianus*; Fig. 1), un mamífero grande y común, ampliamente distribuido en el continente Americano, y de preocupación menor en términos de conservación.

Figura 1. Venado de cola blanca (*Odocoileus virginianus*) nuestra especie de interes para este ejemplo. Foto del proyecto fauna de Manabi http://faunamanabi.github.io

El set de datos contiene J datos replicados de detección o no detección de la especie en M sitios, teniendo en cuenta que asumimos que es una población cerrada ('closure' assumption). Es decir que durante el muestreo no ocurrieron cambios por nacimientos, muertes inmigración o emigración. En otras palabras, el muestreo fue corto en tiempo y la ocurrencia de la especie z no cambió por efectos demográficos.

Claramente debemos distinguir dos procesos, el primero es el proceso ecológico, el cual genera (parcialmente) un estado latente de la ocurrencia z. El segundo es el proceso de observación, el cual produce los datos observados de detección o no detección del venado. Aquí asumimos que el proceso de observación está gobernado por un mecanismo de detección imperfecta. Es decir algunos venado pudieron haberse escapado a mi observación, lo cual genera falsos negativos. También asumimos que los falsos positivos están ausentes, es decir que todo lo que identifico como venado, es efectivamente un venado. Para hacer más realista el ejemplo incluimos los efectos de la altitud y la cobertura de bosque en la ocurrencia, como factores que afectan la ocurrencia linealmente, disminuyendola para el caso de la elevación y que incrementan la ocupación linealmente para el caso de la cobertura de bosques. Al final las dos variables interactúan negativamente entre sí. Esos efectos se introducen en la ocurrencia en escala logarítmica como tradicionalmente se hace un modelo lineal generalizado (GLM).

En nuestra simulación vamos a hacer explícito que no es posible detectar a la totalidad de los venados de un sitio de muestreo, así que estamos enfrentando un tipo de error que nos hace sub-estimar la abundancia de la población. Hay muchas razónes por las cuales fallamos en detectar un individuo en la naturaleza, porque nos distrajimos mientras el vanado paso, porque los binoculares no tenían el aumento suficiente, o simplemente porque el venado se escondió detrás de un árbol al sentir nuestro olor, o por alguna otra razón. De esta forma nosotros vamos a registrar la presencia (z=1) con una probabilidad de detección p la cual también vamos a hacerla dependiente (en la escala logarítmica) de la altitud y de una co-variable que afecta la detección, la temperatura. En términos generales los animales son más difíciles de observar cuando la temperatura es mas alta y por lo general entre más altitud la temperatura disminuye. De esta forma asumimos que la detección está relacionada negativamente con a con la altitud y la temperatura. Pero también hay que tener en cuenta

que el efecto negativo en p también puede ser mediado por una disminución en la abundancia con la altitud, el cual también causa que la probabilidad de ocupación disminuya con la altitud. Tenga en cuenta que una co-variable, la altitud afecta a ambos procesos, el ecológico (la ocurrencia) y al proceso de observación (la probabilidad de detección). Esto tiene un propósito, y es probable que pase en la naturaleza muchas veces. Los modelos de ocupación tienen una base "mecanistica" produciendo variación espacial en la abundancia. Es decir tendremos sitios con mayor abundancia y otros con menor abundancia. Pero los modelos jerárquicos como el que estamos por construir, son capaces de desentrañar esas relaciones complejas entre ocurrencia y probabilidad de detección (e.g., Kéry, 2008; chaps. 12–13 in Kéry & Schaub, 2012). Finalmente para este ejemplo vamos a dejar por fuera el efecto de la interacción entre la altitud y la temperatura, ajustándolo a cero. Luego podremos variar este parámetro. En resumen vamos a generar datos bajo el siguiente modelo, donde los sitios son indexados como i y los conteos repetidos en el sitio van a ser referidos como j y α_3 va a ser igual a cero.

3.0.1 Modelo Ecológico:

$z_i = Bernoulli(\psi_i)$

$logit(\psi_i) = \beta_0 + \beta_1 * Altitud_i + \beta_2 * CovBosque_i + \beta_3 * Altitud_i * CovBosque_i$

3.0.2 Modelo de Observación:

$y_{ij} = Bernoulli(z_i * p_{ij})$

$logit(p_{ij}) = \alpha_0 + \alpha_1 * Altitud_i + \alpha_2 * Temperatura_{ij} + \alpha_3 * Altitud_i * Temperatura_{ij}$

Donde ψ es la ocupación y p la probabilidad de detección. Con β como el coeficiente de la regresión para las co-variables de la ocupación y α el coeficiente de regresión para las co-variables de la detección.

Vamos a generar datos desde "dentro hacia afuera" y desde arriba hacia abajo. Para esto, primero escogemos el tamaño de la muestra y creamos los valores para las co-variables. Segundo, seleccionamos los valores de los parámetros de del modelo ecológico (la ocupación) y ensamblamos la ocurrencia esperada (el parámetro ψ, o la ocupación) y posteriormente obtenemos la variable al azar z la cual tiene distribución Bernoulli. Tercero, seleccionamos los valores de los parámetros del modelo de observación (la detección), para ensamblar la probabilidad de detección p y obtener el segundo set de una variable al azar y (detección observada o no observada de un venado) la cual también tiene distribución Bernoulli.

Para simular los datos usaremos el lenguaje de programación estadística R, el cual provee una gran variedad de técnicas gráficas y estadísticas de modelación y un gran ecosistema de paquetes para análisis estadístico y ecológico. Si aún no lo ha hecho, baje e instale R en su computadora, posteriormente haga lo mismo con RStudio.

3.1 Pasos iniciales: tamaño de la muestra y valores de las co-variables

Inicie RStudio, copie, pegue y ejecute los comandos de la ventana gris.

Primero escogemos el tamaño de la muestra, el número de sitios y el número de medidas repetidas de presencia/ausencia en cada sitio.

```
M <- 60  # Número de réplicas espaciales (sitios)
J <- 15  # Número de réplicas temporales (conteos repetidos)
```

Luego creamos los valores para las co-variables. Tenemos altitud y cobertura de bosque como co-variables de cada sitio. Ellas difieren de sitio a sitio pero para cada muestreo son las mismas. Mientras que la temperatura es una co-variable de la observación, así que si varía en cada muestreo y también en cada sitio. Recuerde que

el sub índice i se refiere al sitio y el j a cada muestreo. Para simplificar las cosas nuestras co-variables van a tener una distribución normal con una media centrada en cero y no se van a extender muy lejos en cada lado del cero. En análisis de datos reales tendremos que estandarizar las co-variables para evitar problemas numéricos de diferencia en las escalas de las co-variables y poder calcular el valor de máxima verosimilitud (MLE), así como también para obtener convergencia en las cadenas de Markov del modelo bayesiano. Aquí vamos a ignorar un hecho de la vida real, y es que las co-variables no son totalmente independientes la una de la otra, es decir en la naturaleza la cobertura boscosa puede estar relacionada con la altitud, pero esto no va a ser relevante, por ahora.

Para inicializar el generador de números aleatorios y obtener siempre los mismos resultados podemos adicionar la siguiente línea:

```
set.seed(24) # Can choose seed of your choice
```

De esta forma podremos obtener siempre los mismos estimados. Pero luego cuando queramos obtener el error de muestreo deberemos remover esa línea. Para este ejemplo generaremos valores para las co variables centrados en cero y variando de -1 a 1.

```
elev <- runif(n = M, -1, 1)              # Scaled elevation of a site
forest <- runif(n = M, -1, 1)            # Scaled forest cover at each site
temp <- array(runif(n = M*J, -1, 1), dim = c(M, J)) # Scaled temperature
```

3.2 Simulando el proceso ecológico y su resultado: la ocurrencia del venado.

Para simular la ocurrencia del venado en cada sitio, escogemos los valores para los parámetros que gobiernan la variación espacial en la ocurrencia β_0 a β_3. El primer parámetro es la ocurrencia promedio esperada del venado (probabilidad de ocupación) cuando todas las co-variables tienen un valor de cero, en otras palabras el intercepto del modelo de ocurrencia. Preferimos pensar en los venados en términos de su ocurrencia en lugar de logit(ocurrencia). Aquí nosotros escogemos el intercepto de la ocupación primero y luego lo transformamos de la escala logarítmica con la función de enlace logit.

```
mean.occupancy <- 0.60          # Mean expected occurrence of deer
beta0 <- plogis(mean.occupancy) # Same on logit scale (= logit-scale intercept)
beta1 <- -2                     # Effect (slope) of elevation
beta2 <- 2                      # Effect (slope) of forest cover
beta3 <- 1                      # Interaction effect (slope) of elev and forest
```

Aquí aplicamos el modelo lineal (a la escala logarítmica) y obtenemos la transformación logit de la probabilidad de ocupación, la cual invertimos con la transformación logit para obtener la ocupación del venado y graficar todo.

```
logit.psi <- beta0 + beta1 * elev + beta2 * forest + beta3 * elev * forest
psi <- plogis(logit.psi)      # Inverse link transformation

# par()               # view current settings
opar <- par()       # make a copy of current settings
par(mfrow = c(2, 2), mar = c(5,4,2,2), cex.main = 1)
curve(plogis(beta0 + beta1*x), -1, 1, col = "red", frame.plot = FALSE, ylim = c(0, 1),
      xlab = "Altitud", ylab = "psi", lwd = 2)
text(0.9, 0.95, "A", cex = 1.5)
plot(elev, psi, frame.plot = FALSE, ylim = c(0, 1), xlab = "Altitud", ylab = "")
text(0.9, 0.95, "B", cex = 1.5)
```

```r
curve(plogis(beta0 + beta2*x), -1, 1, col = "red", frame.plot = FALSE, ylim = c(0, 1),
      xlab = "Forest cover", ylab = "psi", lwd = 2)
text(-0.9, 0.95, "C", cex = 1.5)
plot(forest, psi, frame.plot = FALSE, ylim = c(0, 1), xlab = "Forest cover", ylab = "")
text(-0.9, 0.95, "D", cex = 1.5)
```

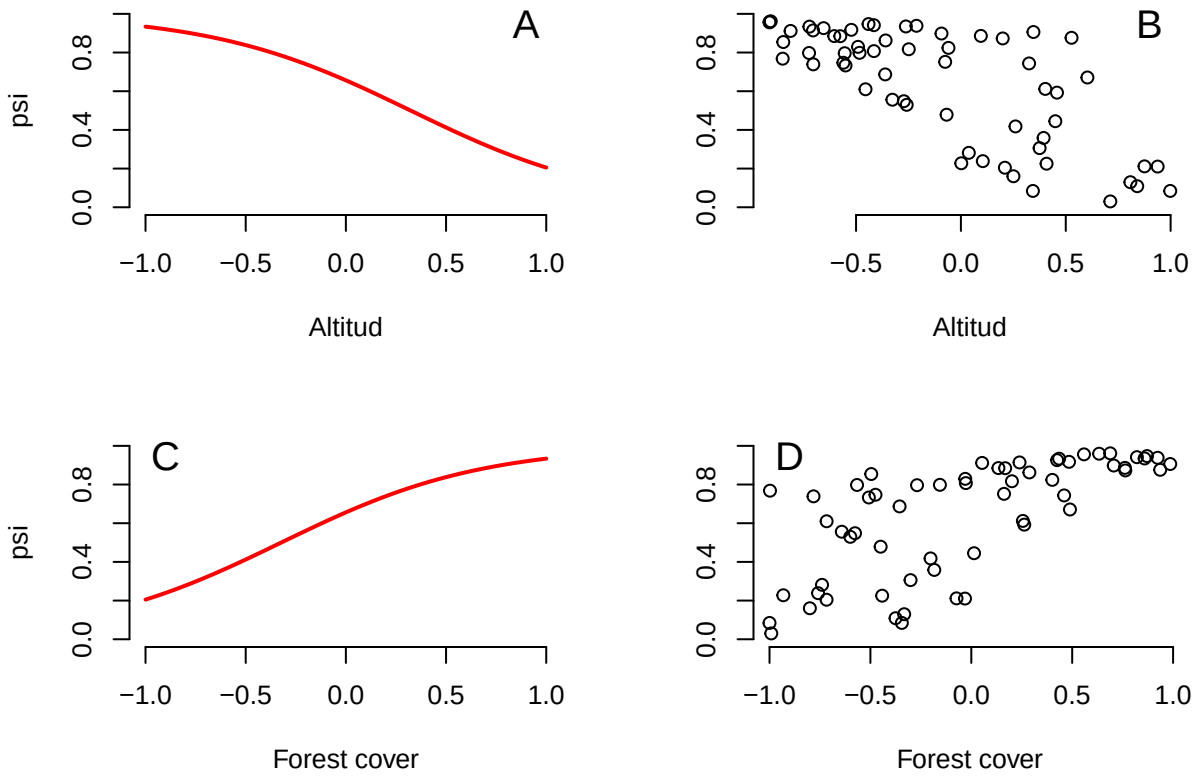


Figure 1: Figura 2. Dos formas de mostrar la relación entre la probabilidad de ocurrencia de los venados y las co-variables.

```r
# dev.off()
par(opar)           # restore original par settings
```

Figura 2. Dos formas de mostrar la relación entre la probabilidad de ocurrencia de los venados y las co-variables. (A) Relación entre psi y altitud para un valor constante (media igual a cero) de cobertura boscosa. (B) Relación entre psi y la altitud en un valor observado de cobertura boscosa. (C) relación psi cobertura boscosa para una altitud constante (en la media de cero). (D) Relación psi cobertura boscosa para el valor observado de altitud.

Para mostrar mejor la relación conjunta entre las dos co variables y psi, debemos realizar un diagrama de superficie. Aquí no hemos cambiado nada de la simulación, solo le hemos agregado más datos para visualizar mejor.

```r
# Compute expected occurrence for a grid of elevation and forest cover
cov1 <- seq(-1, 1, , 100)                          # Values for elevation
```

```r
cov2 <- seq(-1, 1, , 100)                          # Values for forest cover
psi.matrix <- array(NA, dim = c(100, 100)) # Prediction matrix, for every
 # combination of values of elevation and forest cover

for(i in 1:100){
   for(j in 1:100){
      psi.matrix[i, j] <- plogis(beta0 + beta1 * cov1[i] + beta2 * cov2[j] +
                                    beta3 * cov1[i] * cov2[j])
   }
}

mapPalette <- colorRampPalette(c("grey", "yellow", "orange", "red"))
image(x = cov1, y = cov2, z = psi.matrix, col = mapPalette(100), xlab = "Altitud",
      ylab = "Forest cover", cex.lab = 1.2)
contour(x = cov1, y = cov2, z = psi.matrix, add = TRUE, lwd = 1)
matpoints(elev, forest, pch="+", cex=0.8)
```

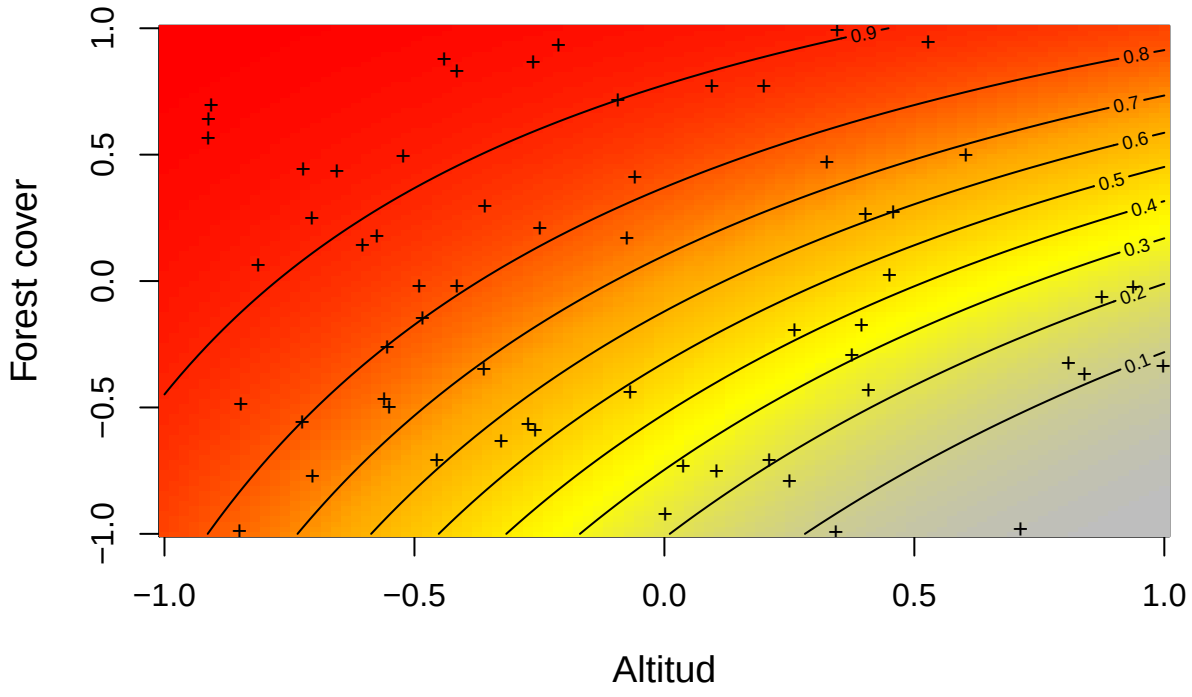


Figure 2: este se supone es el caption

Figura 3. Relación construida ente los datos simulados de la ocurrencia esperada (ocupación) del venado (psi) y la altitud y la cobertura boscosa simultáneamente.

Hasta ahora no hemos introducido ninguna variación estocástica en la relación entre la ocurrencia del venado y las co variables. Para hacer esto debemos hacer uso de algunos modelos estadísticos o distribuciones estadísticas, para describir la variabilidad al azar alrededor del valor esperado de psi. La forma típica de

introducir esta variación al azar es obtener la ocurrencia de venados en cada sitio i, z_i, de una distribución Bernoulli con los valores esperados (ψ_i).

```
z <- rbinom(n = M, size = 1, prob = psi)   # Realised occurrence
sum(z)                                     # Total number of occupied sites
```

```
## [1] 34
```

```
table(z)                                   # Frequency distribution of deer occurrence
```

```
## z
##  0  1
## 26 34
```

Aquí hemos creado el resultado del proceso ecológico: ocurrencia específica para cada sitio z_1. Vemos que 26 sitios no están ocupados y que los restantes 34 están ocupados.

3.3 Simulando el proceso de observación y su resultado: datos de detección/no detección de venados (o medidas de presencia/ausencia)

La ocurrencia z no es lo que normalmente vemos, ya que hay un chance de que fallemos en observar un individuo. De ahí que haya una medida binaria de error cuando medimos la ocurrencia (lo observamos o no lo observamos). Nosotros asumimos que podemos hacer únicamente una de las dos posibles observaciones (si, no), pero pudimos haber perdido la observación de un venado en algún sitio, entonces la probabilidad de detección es menor que uno y la medida de error es afectada por la cobertura de bosque y la temperatura. Hay que tener en cuenta que nunca vamos a registrar la presencia de un venado cuando en realidad no hay venados. En otras palabras estamos asumiendo que no tenemos falsos positivos. Para hacer explícito que tenemos un efecto de interacción entre dos co variables en nuestros datos, vamos a permitir un efecto de la interacción en el código, pero ajustado a cero y de esta forma sin efecto en el modelo que genera los datos. Primero seleccionamos los valores para α_0 hasta α_3, donde el primero es la probabilidad de detección para el venado, en la escala logit, cuando todas las co variables de la detección tienen un valor de cero. Hemos escogido el intercepto del modelo de detección y luego lo transformamos con la función de enlace plogis. Esto no es lo mismo que la probabilidad de detección media, la cual es más alta en nuestro modelo de simulación, como veremos más adelante.

```
mean.detection <- 0.3              # Mean expected detection
alpha0 <- qlogis(mean.detection) # same on logit scale (intercept)
alpha1 <- -1                       # Effect (slope) of elevation
alpha2 <- -3                       # Effect (slope) of temperature
alpha3 <- 0                        # Interaction effect (slope) of elevevation and temperature
```

Aplicando el modelo lineal, tenemos el logit de la probabilidad de detección del venado para cada sitio y muestreo, y aplicándole la transformación inversa (plogis), obtenemos una matriz de las dimensiones 267 por 3 con la probabilidad de detección para cada sitio i y muestreo j. Finalmente, graficamos las relaciones para la probabilidad de detección en los datos.

```
logit.p <- alpha0 + alpha1 * elev + alpha2 * temp + alpha3 * elev * temp
p <- plogis(logit.p)               # Inverse link transform
mean(p)                            # average per-site p is about 0.38
```

```
## [1] 0.3867913
```

```
par(mfrow = c(2, 2), mar = c(5,4,2,2), cex.main = 1)
curve(plogis(alpha0 + alpha1*x), -1, 1, col = "red", frame.plot = FALSE, ylim = c(0, 1.1),
      xlab = "Altitud", ylab = "p", lwd = 2)
text(-0.9, 1.05, "A", cex = 1.5)
matplot(elev, p, pch = "*", frame.plot = FALSE, ylim = c(0, 1.1), xlab = "Altitud",
        ylab = "")
text(-0.9, 1.05, "B", cex = 1.5)
curve(plogis(alpha0 + alpha2*x), -1, 1, col = "red", frame.plot = FALSE, ylim = c(0, 1.1),
      xlab = "Temperature", ylab = "p", lwd = 2)
text(-0.9, 1.05, "C", cex = 1.5)
matplot(temp, p, pch = "*", frame.plot = FALSE, ylim = c(0, 1.1), xlab = "Temperature",
        ylab = "p")
text(-0.9, 1.05, "D", cex = 1.5)
```

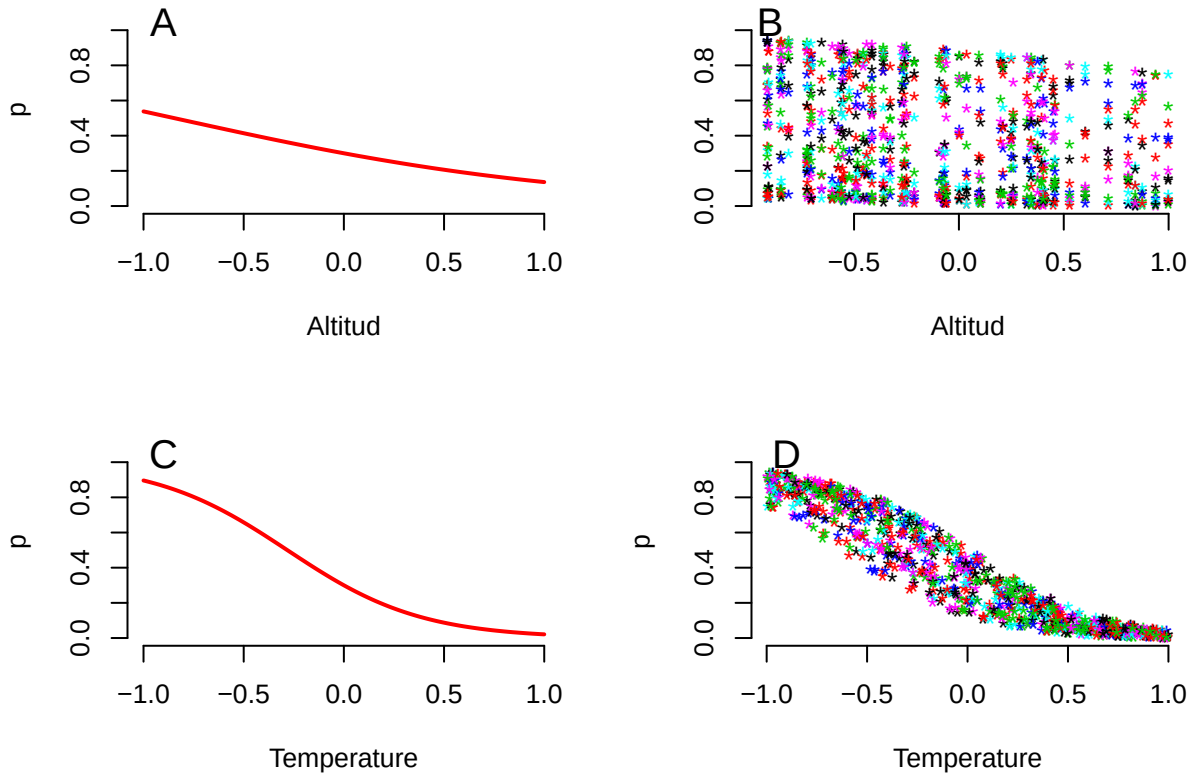


Figura 4. Dos formas de mostrar las relaciones entre la probabilidad de detección esperada del venado (p) y las dos variables altitud y temperatura. (A) Relación p y altitud para temperatura constante (en el valor medio, que es igual a cero). (B) Relación entre p y la altitud en el valor observado de cantidad de temperatura. (C) Relación entre p y temperatura para un valor constante de altitud (en la altitud media igual a cero). (D) Relación entre p y temperatura para un valor observado de altitud.

De forma similar vamos a producir una gráfica con la relación conjunta entre la altitud, la temperatura y la probabilidad de detección del venado (p), actuando simultáneamente, Pero sin el valor de interacción (recuerda que lo hemos ajustado a cero). La relación en la escala logarítmica es representada por un plano con pendiente constante en el eje x y y respectivamente.

```r
# Compute expected detection probability for a grid of elevation and temperature
cov1 <- seq(-1, 1,,100)                    # Values of elevation
cov2 <- seq(-1,1,,100)                     # Values of temperature
p.matrix <- array(NA, dim = c(100, 100)) # Prediction matrix which combines
# every value in cov 1 with every other in cov2
for(i in 1:100){
   for(j in 1:100){
      p.matrix[i, j] <- plogis(alpha0 + alpha1 * cov1[i] + alpha2 * cov2[j] +
                               alpha3 * cov1[i] * cov2[j])
   }
}
image(x = cov1, y = cov2, z = p.matrix, col = mapPalette(100), xlab = "Altitud",
      ylab = "Temperature", cex.lab = 1.2)
contour(x = cov1, y = cov2, z = p.matrix, add = TRUE, lwd = 1)
matpoints(elev, temp, pch="+", cex=0.7, col = "black")
```

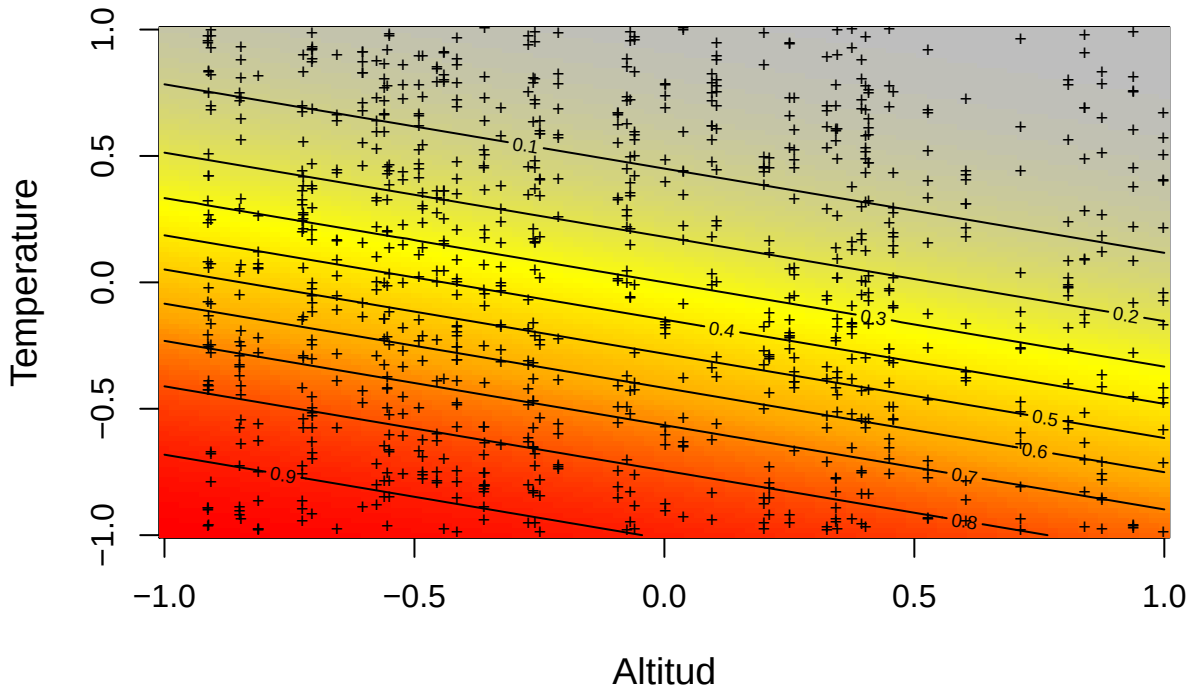


Figura 5. Relación entre la probabilidad de detección esperada del venado (p), la altitud y la temperatura simultáneamente.

3.3.1 Por que Bernoulli?

Cuando "medimos" la ocurrencia, la detección imperfecta, representa una fuente de error con una distribución de tipo Bernoulli (por ejemplo la presencia del venado en un sitio en que es detectado con una probabilidad p, o no es detectado como 1-p. Al aplicar este proceso de observación producimos medidas repetidas de la presencia o ausencia (1 o 0) del venado en cada sitio. La distribución Bernoulli es un caso especial de la

distribución binomial, y su mejor ejemplo es el lanzamiento de una moneda una sola vez. Si requiere una explicación más básica, detallada y con ejemplos le recomiendo visitar khanacademy.

```
y <- matrix(NA, nrow = M, ncol = J)       # Prepare array for counts
for (i in 1:J){                           # Generate counts
   y[,i] <- rbinom(n = M, size = 1, prob = z*p[,i])   # this is the Bernoulli
}
```

Hasta acá hemos simulado la presencia/ausencia del venado de cola blanca en 60 sitios durante 15 sesiones de muestreo. Veamos que contienen las tablas. Recuerde que los sitios están en las filas y los muestreos repetidos en las columnas. Para comparar, mostraremos la verdadera ocurrencia en la primera columna para 30 sitios y solo cinco muestreos.

```
library(knitr)
kable(as.data.frame(head(cbind("True Presence/Absence" = z,
           "1st survey" = y[,1],
           "2nd survey" = y[,2],
           "3rd survey" = y[,3],
           "4th survey" = y[,4],
           "5th survey" = y[,5]), 30)) )   # First 30 rows (= sites)
```

True Presence/Absence	1st survey	2nd survey	3rd survey	4th survey	5th survey
1	0	1	0	0	1
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
1	0	1	1	0	0
0	0	0	0	0	0
1	0	0	1	0	0
1	0	0	0	1	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
1	0	0	0	0	0
1	1	1	0	0	0
0	0	0	0	0	0
1	0	0	0	0	0
1	0	0	1	1	1
0	0	0	0	0	0
1	0	0	0	0	0
1	0	0	1	0	0
1	0	0	0	0	0
1	0	0	0	1	0
1	1	0	0	0	1
1	0	0	0	0	1
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
1	0	1	0	0	1

```
# library(DT)
# datatable(iris, options = list(pageLength = 5))
```

```
par(mfrow = c(2, 2), mar = c(5,4,2,2), cex.main = 1)
matplot(elev, jitter(y), pch = "*", frame.plot = FALSE, ylim = c(0, 1),
        xlab = "Altitud", ylab = "Detection/Nondetection (y)")
matplot(forest, jitter(y), pch = "*", frame.plot = FALSE, ylim = c(0, 1),
        xlab = "Forest cover", ylab = "Detection/Nondetection (y)")
matplot(temp, jitter(y), pch = "*", frame.plot = FALSE, ylim = c(0, 1),
        xlab = "Teperature", ylab = "Detection/Nondetection (y)")
hist(y, breaks = 50, col = "grey", ylim = c(0, 600), main = "",
     xlab = "Detection/Nondetection (y)")
```

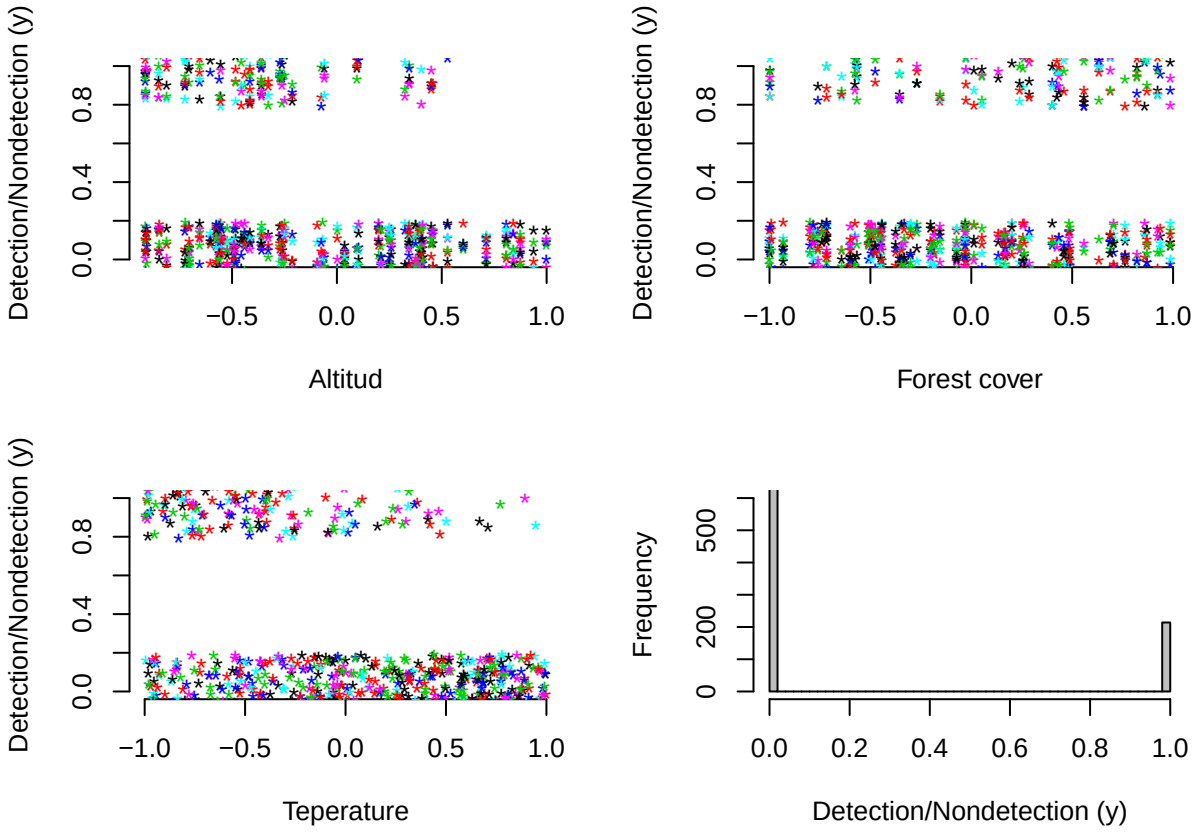


Figura 6. Relación entre la (jittered) ocupación observada de venados (y) y las tres co variables estandarizadas. Altitud (A). Cobertura de bosque (B). Temperatura (C) y la frecuencia de distribución de la ocurrencia observada (y) en un set de datos de 60 sitios con 15 muestreos cada uno (D).

Hasta aquí hemos creado un set de datos donde la deteccion/no deteccion del venado esta negativamente correlacionado con la temperatura y positivamente relacionado con la cobertura de bosque. Hay una razón por la cual esta correlacione entre variables es diferente. La ocurrencia por sitio, el objetivo de la inferencia ecológica, está afectado por la cobertura de bosque y la altitud, pero no por la temperatura, mientras que la probabilidad de detección, el parámetro caracterizando la medida del proceso de error cuando tomamos medidas de ocurrencia, es también afectado por la altitud y adicionalmente por la temperatura. Por lo tanto, como se puede notar, hay un gran reto en desenredar la razón de la variación espacio-temporal en la observación de los datos de detección/no detección, dado que pueden ser afectados por dos procesos totalmente diferentes: el ecológico y el observacional.

3.4 Empacando todo en una función.

Podría ser de mucha utilidad empacar todo lo que hemos hecho en una sola función que nos permita hacer lo mismo muchas veces repetidamente. Esto hará que podamos diseñar simulaciones de una forma más concisa y flexible y hace mas transparente la generación de parámetros usados para generar datos. Asi que vamos a definir una función para generar el mismo tipo de datos que acabamos de crear, asignando argumentos a la función, tales como tamaño de la muestra, efectos de las co variables y magnitudes de la interacción de los términos del error de detección. Esto hará que nuestro código sea más flexible y eficiente.

```
# Function definition with set of default values
data.fn <- function(M = 267, J = 3, mean.occupancy = 0.6,
                    beta1 = -2, beta2 = 2, beta3 = 1, mean.detection = 0.3,
                    alpha1 = -1, alpha2 = -3, alpha3 = 0, show.plot = TRUE){
# Function to simulate occupancy measurements replicated at M sites during J occasions.
# Population closure is assumed for each site.
# Expected occurrence may be affected by elevation (elev),
# forest cover (forest) and their interaction.
# Expected detection probability may be affected by elevation,
# temperature (temp) and their interaction.
# Function arguments:
#     M: Number of spatial replicates (sites)
#     J: Number of temporal replicates (occasions)
#     mean.occupancy: Mean occurrence at value 0 of occurrence covariates
#     beta1: Main effect of elevation on occurrence
#     beta2: Main effect of forest cover on occurrence
#     beta3: Interaction effect on occurrence of elevation and forest cover
#     mean.detection: Mean detection prob. at value 0 of detection covariates
#     alpha1: Main effect of elevation on detection probability
#     alpha2: Main effect of temperature on detection probability
#     alpha3: Interaction effect on detection of elevation and temperature
#     show.plot: if TRUE, plots of the data will be displayed;
#                set to FALSE if you are running simulations.

# Create covariates
elev <- runif(n = M, -1, 1)                          # Scaled elevation
forest <- runif(n = M, -1, 1)                        # Scaled forest cover
temp <- array(runif(n = M*J, -1, 1), dim = c(M, J)) # Scaled temperature

# Model for occurrence
beta0 <- qlogis(mean.occupancy)                # Mean occurrence on link scale
psi <- plogis(beta0 + beta1*elev + beta2*forest + beta3*elev*forest)
z <- rbinom(n = M, size = 1, prob = psi)    # Realised occurrence

# Plots
if(show.plot){
par(mfrow = c(2, 2), cex.main = 1)
devAskNewPage(ask = TRUE)
curve(plogis(beta0 + beta1*x), -1, 1, col = "red", frame.plot = FALSE,
      ylim = c(0, 1), xlab = "Elevation", ylab = "psi", lwd = 2)
plot(elev, psi, frame.plot = FALSE, ylim = c(0, 1), xlab = "Elevation",
     ylab = "")
curve(plogis(beta0 + beta2*x), -1, 1, col = "red", frame.plot = FALSE,
      ylim = c(0, 1), xlab = "Forest cover", ylab = "psi", lwd = 2)
plot(forest, psi, frame.plot = FALSE, ylim = c(0, 1), xlab = "Forest cover",
```

```
     ylab = "")
}

# Model for observations
y <- p <- matrix(NA, nrow = M, ncol = J)     # Prepare matrix for y and p
alpha0 <- qlogis(mean.detection)        # mean detection on link scale
for (j in 1:J){                         # Generate counts by survey
   p[,j] <- plogis(alpha0 + alpha1*elev + alpha2*temp[,j] + alpha3*elev*temp[,j])
   y[,j] <- rbinom(n = M, size = 1, prob = z * p[,j])
}

# True and observed measures of 'distribution'
sumZ <- sum(z)                      # Total occurrence (all sites)
sumZ.obs <- sum(apply(y,1,max))     # Observed number of occ sites
psi.fs.true <- sum(z) / M           # True proportion of occ. sites in sample
psi.fs.obs <- mean(apply(y,1,max)) # Observed proportion of occ. sites in sample

# More plots
if(show.plot){
par(mfrow = c(2, 2))
curve(plogis(alpha0 + alpha1*x), -1, 1, col = "red",
      main = "Relationship p-elevation \nat average temperature",
      xlab = "Scaled elevation", frame.plot = F)
matplot(elev, p, xlab = "Scaled elevation",
        main = "Relationship p-elevation\n at observed temperature",
        pch = "*", frame.plot = F)
curve(plogis(alpha0 + alpha2*x), -1, 1, col = "red",
      main = "Relationship p-temperature \n at average elevation",
      xlab = "Scaled temperature", frame.plot = F)
matplot(temp, p, xlab = "Scaled temperature",
        main = "Relationship p-temperature \nat observed elevation",
        pch = "*", frame.plot = F)
}

# Output
return(list(M = M, J = J, mean.occupancy = mean.occupancy,
            beta0 = beta0, beta1 = beta1, beta2 = beta2, beta3 = beta3,
            mean.detection = mean.detection,
            alpha0 = alpha0, alpha1 = alpha1, alpha2 = alpha2, alpha3 = alpha3,
            elev = elev, forest = forest, temp = temp,
            psi = psi, z = z, p = p, y = y, sumZ = sumZ, sumZ.obs = sumZ.obs,
            psi.fs.true = psi.fs.true, psi.fs.obs = psi.fs.obs))
}
```

Una vez que hayamos definido la función y ejecutado su código, podremos llamarla repetidamente y enviar los resultados a la pantalla o asignarlos a un objeto en R. De forma tal que podamos usar el set de datos almacenado en el objeto para un análisis.

```
data.fn()                   # Execute function with default arguments
data.fn(show.plot = FALSE) # same, without plots
data.fn(M = 267, J = 3, mean.occupancy = 0.6, beta1 = -2, beta2 = 2, beta3 = 1,
        mean.detection = 0.3, alpha1 = -1,
        alpha2 = -3, alpha3 = 0, show.plot = TRUE) # Explicit defaults
```

```
datos <- data.fn(show.plot = FALSE)          # Assign results to an object called 'datos'
```

Tal vez el uso más sencillo posible para esta función es experimentar de primera mano el error de muestreo: el cuál es la variabilidad natural de realizaciones repetidas (varios sets de datos) de nuestro proceso estocástico por el cual calculamos los set de datos. Vamos a simular 10.000 sets de datos del venado para ver como varían en términos del verdadero número de sitios ocupados (sumZ en el codigo) y el número de sitios en los que los venados fueron observados al menos una vez.

```
simrep <- 10000
trueSumZ <- obsSumZ <- numeric(simrep)
for(i in 1:simrep){
   if(i %% 1000 ==0 )                        # report progress
   cat("iter", i, "\n")
   data <- data.fn(M = 60,J = 3,show.plot = FALSE) # 60 sitios, 3 muestreos
   trueSumZ[i] <- data$sumZ
   obsSumZ[i] <- sum(apply(data$y, 1, max))
}
```

```
## iter 1000
## iter 2000
## iter 3000
## iter 4000
## iter 5000
## iter 6000
## iter 7000
## iter 8000
## iter 9000
## iter 10000
```

```
plot(sort(trueSumZ), ylim = c(min(obsSumZ), max(trueSumZ)), ylab = "", xlab = "Simulation",
     col = "red", main = "True (red) and observed (blue) number of occupied sites")
points(obsSumZ[order(trueSumZ)], col = "blue")
```

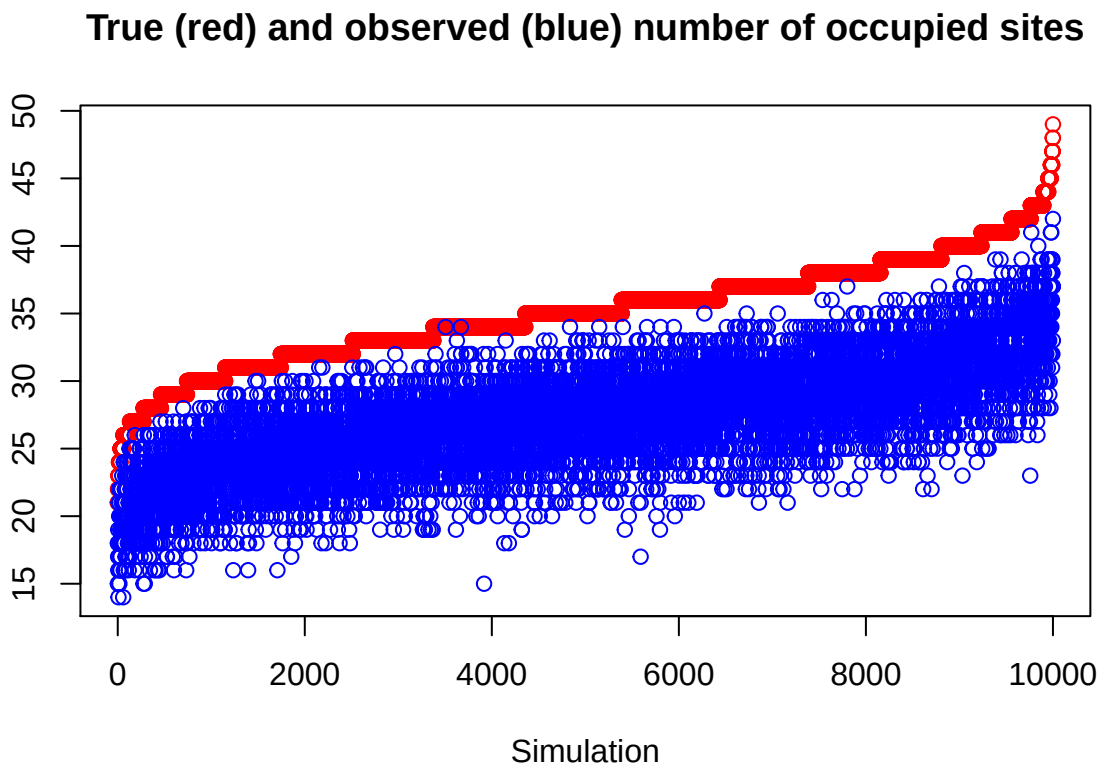


Figura 7. Variabilidad natural (error de muestreo) del verdadero número de sitios ocupados (ordenados por tamaño) en color rojo y el número observado de sitios ocupados (en azul), para un muestreo simulado de venados. El número de sitios observados también se conoce como la ocupación ingenua o detección "naïve" (observable) de la ocurrencia de los venados en 60 sitios en la simulación. El ancho del área azul representa el error inducido por la detección imperfecta. Note la importancia de tener en cuenta este error para tener una mejor idea de la ocupación.

Ahora podemos usar esta función para generar datos bajo diferentes esquemas de muestreo, variando el número de sitios y el número de muestreos repetidos. Así como también bajo diferentes características ecológicas y de detección.

```
data.fn(J = 2)                  # Only 2 surveys
data.fn(J = 1)                  # Only 1 survey (no temporal replicate)
data.fn(M = 1, J = 100)         # No spatial replicates, but 100 counts
data.fn(beta3 = 1)              # With interaction elev-wind on p
data.fn(M = 267, J = 3, mean.occupancy = 0.6, beta1 = -2, beta2 = 2, beta3 = 1,
        mean.detection = 1)             # p = 1 (i.e., perfect detection)
data.fn(mean.occupancy = 0.96) # Really common species
data.fn(mean.occupancy = 0.05) # Really rare species

data.fn(M = 267, J = 3, mean.occupancy = 0.6, beta1 = 0, beta2 = 0, beta3 = 0,
        mean.detection = 0.3, alpha1 = 0, alpha2 = 0, alpha3 = 0, show.plot = TRUE)
# Simplest case with occupancy and detection constant, no covariate effects
```

Felicitaciones!, si llego hasta acá y si entendió la simulación de datos y su procedimiento, entonces Ud. entendió totalmente el modelo básico de ocupación, el cual es la piedra angular del muestreo y monitoreo biológico moderno.

4 Analisis de ocupación

Ya que hemos entendido como funcionan e interactúan los dos procesos; el ecológico y el observacional para producir los datos de ocupación y luego de generar varios sets de datos, ahora solo nos resta analizarlos. La forma más directa e intuitiva es usar la función occu del paquete unmarked (cita). Posteriormente podremos usar un modelo de tipo bayesiano en el lenguaje BUGS para analizar los datos.

4.1 Generando los datos

Esta vez recurriremos a un diseño tipo TEAM (http://www.teamnetwork.org) con 60 sitios de muestreo y treinta días de muestreos repetidos. De nuevo nuestra especie es el venado de cola blanca. Para este ejemplo asumiremos que la detección es 0.5 y las mismas interacciones con las co variables altitud, cobertura de bosque y temperatura.

```
# data generation
datos2<-data.fn(M = 60, J = 30, beta1 = -2, beta2 = 2, beta3 = 1,
                mean.occupancy = 0.6, mean.detection = 0.5,
                    alpha1 = -1, alpha2 = -3, alpha3 = 0, show.plot = FALSE)

#To make the objects inside the list directly accessible to R, without having to address
#them as data$C for instance, you can attach datos2 to the search path.

attach(datos2)          # Make objects inside of 'datos2' accessible directly
```

```
## The following objects are masked _by_ .GlobalEnv:
##
##     alpha0, alpha1, alpha2, alpha3, beta0, beta1, beta2, beta3,
##     elev, forest, J, M, mean.detection, mean.occupancy, p, psi,
##     temp, y, z
```

```
#Remember to detach the data after use, and in particular before attaching a new data
#object, because more than one data set attached in the search path will cause confusion.

# detach(datos2)          # Make clean up
```

4.2 Poniendo los datos en unmarked

Unmarked (http://cran.r-project.org/web/packages/unmarked/index.html) es el paquete de R que usamos para analizar los datos de ocupacion. Para lograr esto debemos primero preparar los datos y juntarlos en un objeto de tipo unmarkedFrame. En este caso usamos la funcion unmarkedFrameOccu que es especifica para analisis de ocupacion de una sola epoca o estacion. Mas sobre unkarked en: https://sites.google.com/site/unmarkedinfo/home

```
library(unmarked)
```

```
## Loading required package: reshape
```

```
## Loading required package: lattice
```

```
## Loading required package: Rcpp
```

```
siteCovs <- as.data.frame(cbind(forest,elev))
obselev<-matrix(rep(elev,J),ncol = J) #make elevetion per observation
obsCovs <- list(temp= temp,elev=obselev)
umf <- unmarkedFrameOccu(y = y, siteCovs = siteCovs, obsCovs = obsCovs)
```

4.3 Ajustando los modelos

El siguiente paso es ajustar los modelos que se requerían variando las co variables. Esto se logra con la función occu().

```
fm0 <- occu(~1 ~1, umf) #detection first, occupancy next
fm1 <- occu(~ temp + elev ~ 1, umf)
fm2 <- occu(~ temp + elev ~ elev, umf)
fm3 <- occu(~ temp + elev ~ elev + forest, umf)
# fm4 <- occu(~ temp + elev ~ elev + forest + elev*forest, umf)
```

4.4 Model selection

Unmarked permite hacer selección de modelos basándose en el AIC de cada uno. De forma tal que el menor AIC es el modelo más parsimonioso de acuerdo a nuestros datos.

```
library(xtable)
models <- fitList(
  'psi(.)p(.)'                       = fm0,
  'psi(.)p(temp + elev)'             = fm1,
  'psi(elev)p(temp + elev)'          = fm2,
  'psi(elev + forest)p(temp + elev)' = fm3)
#  'psi(elev + forest + elev*forest)p(temp + elev)'  = fm4)

modSel(models)
```

```
##                                nPars    AIC  delta   AICwt cumltvWt
## psi(elev + forest)p(temp + elev)     6 545.43   0.00 9.8e-01     0.98
## psi(elev)p(temp + elev)              5 553.52   8.09 1.7e-02     1.00
## psi(.)p(temp + elev)                 4 563.68  18.25 1.1e-04     1.00
## psi(.)p(.)                           2 779.86 234.43 1.2e-51     1.00
```

4.5 Predicción y graficas

El modelo con menor AIC puede ser usados para predecir resultados esperados de acuerdo a un nuevo set de datos. Por ejemplo, uno podría preguntar la abundancia de venados que se espera encontrar en un sitio con 50% de cobertura forestal. La predicciones también otra forma de presentar los resultados de un análisis. Aquí ilustraremos como se ve la predicción de p sobre el rango de las co variables estudiadas. Note que estamos usando co variables estandarizadas. Si estuviéramos usando co variables en su escala real, tendríamos que tener en cuenta que hay que transformarlas usando la media y la desviación estándar.

Antes de usar el modelo para predecir es buena idea verificar que el modelo ajusta bien con la función parboot.

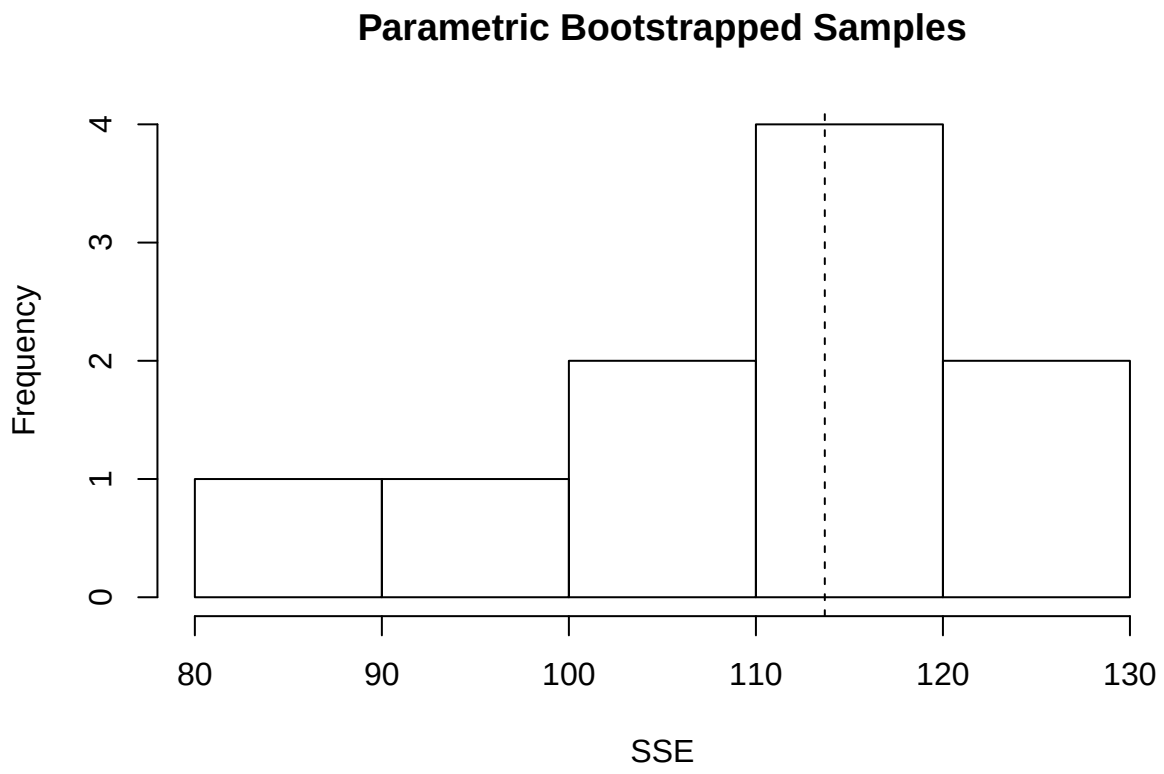
```
library(raster)
```

```
## Loading required package: sp

##
## Attaching package: 'sp'

## The following object is masked from 'package:unmarked':
##
##     coordinates

##
## Attaching package: 'raster'

## The following objects are masked from 'package:unmarked':
##
##     getData, projection
```

```
parboot(fm3) # goodness of fit
```

```
##
## Call: parboot(object = fm3)
##
## Parametric Bootstrap Statistics:
##      t0 mean(t0 - t_B) StdDev(t0 - t_B) Pr(t_B > t0)
## SSE 114           5.37             9.13        0.182
##
## t_B quantiles:
##     0% 2.5% 25% 50% 75% 97.5% 100%
## SSE 97   98 103 106 111   126  129
##
## t0 = Original statistic compuated from data
## t_B = Vector of bootstrap samples
```

```
plot(parboot(fm3)) # goodness of fit
```

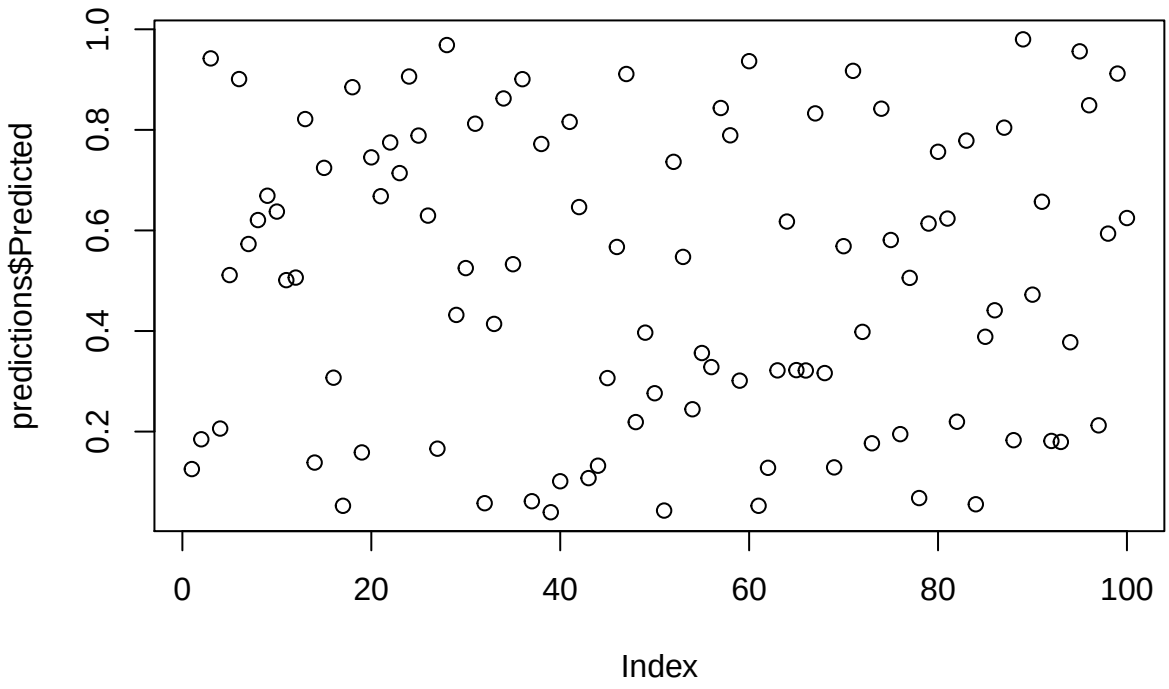



```
#makes a new dataset:

elev <- runif(n = 100, -1, 1)                          # Scaled elevation
forest <- runif(n = 100, -1, 1)                        # Scaled forest cover
temp <- array(runif(n = M*J, -1, 1), dim = c(M, J)) # Scaled temperature
mapdata.g<-as.data.frame(cbind(elev,forest))

library(spatstat)
```

```
## Loading required package: nlme

##
## Attaching package: 'nlme'

## The following object is masked from 'package:raster':
##
##     getData

## The following objects are masked from 'package:unmarked':
##
##     getData, ranef

## Loading required package: rpart
```

```
##
## spatstat 1.45-2       (nickname: 'Caretaker Mode')
## For an introduction to spatstat, type 'beginner'


##
## Attaching package: 'spatstat'

## The following objects are masked from 'package:raster':
##
##     area, rotate, shift

## The following object is masked from 'package:lattice':
##
##     panel.histogram
```

```
set.seed(2002)

# CONSTRUCT ANALYSIS WINDOW USING THE FOLLOWING:
xrange=c(-2.5, 1002.5)
yrange=c(-2.5, 502.5)
window<-owin(xrange, yrange)

# DISPERSE SEEDS FOR SPECIES 1:
#elev <- density(rpoispp(lambda=0.001, win=window)) # should generate ~505 seeds
#forest <- density(rpoispp(lambda=0.02, win=window)) # should generate ~505 seeds
# mapdata.m<-stack(elev,forest)

#elev<-raster(matrix(ncol = 50, nrow = 100, runif(100)))
#forest<-as.raster(matrix(hcl(0, 80, seq(50, 80, 10)),
                  #nrow = 4, ncol = 5))

predictions <- predict(fm3, type="state", newdata=mapdata.g)
# levelplot(Predicted ~ x.coord+y.coord, data=predictions)
plot(predictions$Predicted)
```



4.6 Análisis bayesiano

```
### Generate one data set
# ***************************
set.seed(148)
data <- data.fn(show.plot = T)     # Default arguments
str(data)                          # Look at the object


### Create values for two factors
# ******************************
### They are not correlated to response
# Create factors: time and habitat
time <- matrix(rep(as.character(1:data$J), data$M), ncol = data$J, byrow = TRUE)
hab <- c(rep("A", 90), rep("B", 90), rep("C", 87)) # for a total of M=267 sites



#### Skip model selection, which comes up with this AIC-best model
# ***************************************************************
library(unmarked)
umf <- unmarkedFrameOccu(y = data$y,
          siteCovs = data.frame(elev = data$elev, forest = data$forest, hab = hab),
          obsCovs = list(wind = data$wind, time = time))
```

```
summary(umf)
summary(fm22 <- occu(~elev+wind ~elev*forest*hab, data=umf))




### Fit same model with JAGS, using library jagsUI
# ************************************************
# Bundle data
HAB <- as.numeric(as.factor(hab))  # Get numeric habitat factor
win.data <- list(y = data$y,
                 M = nrow(data$y),
                 J = ncol(data$y),
                 elev = data$elev,
                 forest = data$forest,
                 wind = data$wind, HAB = HAB)
str(win.data)


# Specify model in BUGS language
sink("model22.txt")
cat("
model {

# Priors
mean.p ~ dunif(0, 1)         # Detection intercept on prob. scale
alpha0 <- logit(mean.p)      # same on logit scale
mean.psi ~ dunif(0, 1)       # Occupancy intercept on prob. scale
beta0 <- logit(mean.psi)     # same on logit scale
for(k in 1:2){               # 3 detection covariates
   alpha[k] ~ dnorm(0, 0.01) # Covariates on logit(detection)
#   alpha[k] ~ dnorm(0, 0.05) # Covariates on logit(detection)
#   alpha[k] ~ dunif(-10, 10) # Covariates on logit(detection)
}

for(k in 1:11){              # 11 occupancy covariate terms in design mat
   beta[k] ~ dnorm(0, 0.01)  # Covariates on logit(occupancy)
#   beta[k] ~ dnorm(0, 0.05)  # Covariates on logit(occupancy)
#   beta[k] ~ dunif(-10, 10)  # Covariates on logit(occupancy)
}

# Translation of the occupancy parameters in unmarked into those for BUGS:
# (Intercept)         (beta0 in BUGS)
# elev                (beta[1])
# forest              (beta[2])
# habB                (beta[3])
# habC                (beta[4])
# elev:forest         (beta[5])
# elev:habB           (beta[6])
# elev:habC           (beta[7])
# forest:habB         (beta[8])
# forest:habC         (beta[9])
# elev:forest:habB    (beta[10])
```

```
# elev:forest:habC    (beta[11])

# Likelihood
for (i in 1:M) {
  # True state model for the partially observed true state
  z[i] ~ dbern(psi[i])                      # True occupancy z at site i
  logit(psi[i]) <- beta0 +                  # occupancy (psi) intercept
    beta[1] * elev[i] +                     # elev
    beta[2] * forest[i] +                   # forest
    beta[3] * equals(HAB[i],2) +            # effect of habitat 2 (= B)
    beta[4] * equals(HAB[i],3) +            # effect of habitat 3 (= C)
    beta[5] * elev[i] * forest[i] +         # elev:forest
    beta[6] * elev[i] * equals(HAB[i],2) +       # elev:habB
    beta[7] * elev[i] * equals(HAB[i],3) +       # elev:habC
    beta[8] * forest[i] * equals(HAB[i],2) +     # forest:habB
    beta[9] * forest[i] * equals(HAB[i],3) +     # forest:habC
    beta[10] * elev[i] * forest[i] * equals(HAB[i],2) + # elev:forest:habB
    beta[11] * elev[i] * forest[i] * equals(HAB[i],3)   # elev:forest:habC

   for (j in 1:J) {
      # Observation model for the actual observations
      y[i,j] ~ dbern(p.eff[i,j])    # Detection-nondetection at i and j
      p.eff[i,j] <- z[i] * p[i,j]
      logit(p[i,j]) <- alpha0 +     # detection (p) intercept
         alpha[1] * elev[i] +       # effect of elevation on p
         alpha[2] * wind[i,j]       # effect of wind on p
   }
}

# Derived quantities
sumZ <- sum(z[])      # Number of occupied sites among those studied
occ.fs <- sum(z[])/M  # proportion of occupied sites among those studied
logit.psi <- beta0    # For comparison with unmarked
logit.p <- alpha0     # ditto
}
",fill = TRUE)
sink()


# Initial values
zst <- apply(data$y, 1, max)
inits <- function(){list(z = zst,
                         mean.psi = runif(1),
                         mean.p = runif(1),
                         alpha = rnorm(2),
                         beta = rnorm(11))}


# Parameters monitored
params <- c("sumZ", "occ.fs", "logit.psi", "logit.p", "alpha", "beta")

# MCMC settings
ni <- 50000   ;   nt <- 10   ;   nb <- 10000   ;   nc <- 3
```

```
#ni <- 600   ;   nt <- 1   ;   nb <- 100   ;   nc <- 3

# Call JAGS from R (ART 260 sec with norm(), 480 with unif(-10,10)) and summarize posteriors
system.time(out22 <- jags(win.data, inits, params, "model22.txt",
                          n.chains = nc,
                          n.thin = nt,
                          n.iter = ni,
                          n.burnin = nb,
                          parallel = T))
traceplot(out22)
print(out22, 3)

#### Compare estimates from ML method (first two columns) and posterior sampling (column 3-4).
tmp <- summary(fm22)
cbind(rbind(tmp$state[1:2], tmp$det[1:2]),
      Post.mean = out22$summary[c(3, 7:17, 4:6), 1],
      Post.sd = out22$summary[c(3, 7:17, 4:6), 2])
```

```
sessionInfo()
```

```
## R version 3.3.0 (2016-05-03)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] spatstat_1.45-2 rpart_4.1-10    nlme_3.1-127    raster_2.5-8
##  [5] sp_1.2-3        xtable_1.8-2    unmarked_0.11-0 Rcpp_0.12.5
##  [9] lattice_0.20-33 reshape_0.8.5   knitr_1.13
##
## loaded via a namespace (and not attached):
##  [1] tensor_1.5       magrittr_1.5     stringr_1.0.0    highr_0.6
##  [5] plyr_1.8.4       tools_3.3.0      grid_3.3.0       mgcv_1.8-12
##  [9] deldir_0.1-12    htmltools_0.3.5  abind_1.4-3      yaml_2.1.13
## [13] goftest_1.0-3    digest_0.6.9     Matrix_1.2-6     formatR_1.4
## [17] codetools_0.2-14 evaluate_0.9     rmarkdown_0.9.6  polyclip_1.5-6
## [21] stringi_1.1.1
```