Ideal Free Distribution

Inés Naya

3 de Octubre 2008

1. Introducción

Asumiendo una temporada de pesca (season) lo suficientemente corta como para obviar los procesos de mortalidad, crecimiento y reclutamiento, para entender la dinámica intra-season del esfuerzo en cada área según el modelo de distribución libre ideal del esfuerzo (*Ideal Free Distribution*)y se llevó a cabo su implementación en una hoja de cálculo (ver *IDF.xls*).

Se tomaron como variables de entrada a los siguientes parámetros:

1. Parametros globales fijos independientes del tiempo: h, B_s, P, y pr_0.
2. Parametros locales fijos independientes del tiempo: $N_{i,0}$, q_i, c_i.
3. Esfuerzo total a lo largo de toda la campaña de pesca (season): E_T

Donde,

- h: Tiempo necesario para pescar sistemáticamente toda la agregación.
- B_s: Biomasa mean explotable por agregación o captura por agregación atacada.
- P: Precio por captura.
- pr_0: Rentabilidad base en el equilibrio. Asumiendo que se llega a equilibrio.
- $N_{i,0}$: Tamaño inicial de la población en el área i.
- q_i: Capturabilidad en el área i.
- c_i: Costes asociados al área i.

2. Implementación de IFD según Walters&Martell 2003, pp.215-223

Si queremos calcular los distintos $N_{i,t}$ sabemos que:

$$\frac{dN_{i,t}}{dt} = -f_{i,t}CPUE_{i,t} = \frac{-f_{i,t}q_iN_{i,t}}{[1+\frac{q_ih}{B_s}N_{i,t}]} \quad (1)$$

donde $\int_0^T f_{i,t}dt = E_i$ que podemos calcular mediante,

$$E_i = \frac{\frac{q_ih}{B_s}(N_{i,0} - Ni,T) - ln(\frac{N_{i,T}}{N_{i,0}})}{q_i} \quad (2)$$

sabiendo que,

$$N_{i,T} = \frac{[pr_0 + c_i/P]}{[q_i - \frac{q_ih}{B_s}(pr_0 + c_i)/P} \quad (3)$$

Si no tenemos restricciones en el esfuerso total se fija la rentabilidad base (pr_0) por debajo de la cual se deja de pescar. Si tenemos restricciones en el esfuerzo total (E_T), partimos de una rentabilidad (*profitability*) inicial que se fija arbitrariamente dentro de unos valores razonables (para que la velocidad de convergencia del algoritmo sea mayor). A partir de esta estimación inicial se estima la rentabilidad a la cual se cumple que $\Sigma E_{i,T} = E_T$. Esto se consigue con Solver (en el caso de Excel), estimando el valor de pr_0 con $\Sigma E_{i,T} - E_T = 0$.

PROBLEMAS:

1. Si el nivel de esfuerzo es mucho mayor de lo que se podría llegar a alcanzar con valores positivos de pr_0, en Excel sale error y el algoritmo no converge bien. La razón es que a valores negativos de pr_0, $N_{i,T}$ también es negativa, por lo que al calcular E_i da error (no hay ln de números negativos).

2. La estimación de $pr0$ a partir de un E_T asume que los rendimientos se van a igualar al final de la temporada de pesca. Si este supuesto no se cumple, este planteamiento no es correcto. Para calcular la pr_0, se deben igualar las CPUEs de todas las áreas, si hay un área con CPUE muy baja asume que el **esfuerzo va a ser negativo**. Sobreestima el esfuerzo de las otras áreas. ¡- Se puede arreglar restringiendo los esfuerzos a valores positivos y ignorando las áreas con esfuerzos negativos a la hora del cálculo de pr_0.

3. IFD por asignación de esfuerzo al área de mayor rentabilidad instantánea: Método obsoleto según Walters&Martell 2003

Asumimos que el esfuerzo total va a estar repartido en el tiempo de forma homogénea de forma, $E_{T,t} = E_T/T$.

Sabemos que $pr_{i,t} = PCPUE_{i,t} - c_i$, y cómo,

$$CPUE_{i,t} = \frac{q_iN_{i,t}}{[1+\frac{q_ih}{B_s}N_{i,t}]} \quad (4)$$

Podemos calcular la CPUE por área y tiempo, calcular la rentabilidad instantánea por área y tiempo e intentar repartir el esfuerzo total en cada instante de tiempo en función de esas rentabilidades. Así, escogemos el área que tenga mayor rentabilidad y a esa es a la que le asignamos el esfuerzo en ese instante.

Para determinar el valor que toma $f_{i,t}$ hay que tener en cuenta varios casos:

1. Que todas las rentabilidades de área sean distintas: Se mira cúal es la mayor y a esa se le asigna todo el esfuerzo. A las otras se les asigna 0.
2. Que todas las rentabilidades sean iguales: Se divide el esfuerzo entre todas las áreas equitativamente.
3. Que haya dos o tres rentabilidades iguales: Se mira si son las más grandes y si lo son se reparte el esfuerzo entre ellas. (Hay que implementarlo. Aún no se ha hecho porque Excel no admite más de siete funciones anidadas, y como no tiene la estructura ifelse no permite hacerlo.)

4. Asignación del esfuerzo por el Método Gravitacional

Este método sólo funciona bien bajo niveles muy altos de esfuerzo.

5. Modificación en el método de Walters & Martell para casos de limitaciones en el esfuerzo

Se pueden hacer dos cosas:

1. Poner una restrición por la cual el esfuerzo no pueda tomar valores negativos y que calcule la pr_0 teniendo en cuenta esto. Método: IF(Esfuerzo¿0;Esfuerzo;0).
2. Hacerlo en dos pasos. Calculas pr_0, miras si hay algún esfuerzo negativo. Pasas esos esfuerzos a cero y calculas de nuevo pr_0 ignorando esas áreas.

En ambos casos, estaría bien prescindir de **Solver** para el cálculo de la pr_0. El algoritmo de Newton-Raphson se utiliza para encontrar aproximaciones de los ceros o raíces de una función.

5.1. Algoritmo de Newton-Raphson

Fuente: `http://es.wikipedia.org/wiki/M%C3%A9todo_de_Newton`

Descripción del método La idea de este método es la siguiente: se comienza con un valor razonablemente cercano al cero (denominado punto de arranque), entonces se reemplaza la función por la recta tangente en ese valor, se iguala a cero y se despeja (fácilmente, por ser una ecuación lineal). Este cero será, generalmente, una aproximación mejor a la raíz de la función. Luego, se aplican tantas iteraciones como se deseen.

Supóngase $f : [a, b] -> R$ función derivable definida en el intervalo real $[a, b]$. Empezamos con un valor inicial x_0 y definimos para cada número natural n

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{5}$$

Donde f' denota la derivada de f.

Algoritmo En nuestro caso nuestra función sería:

$$f(pr_0) = E_T - \sum_{i=1}^{n} E_i \tag{6}$$

donde,

$$E_i = \frac{1}{q_i}(\frac{q_i h}{Bs}(N_{i,0} - N_{i,T}) - ln(\frac{N_{i,T}}{N_{i,0}})) \tag{7}$$

y,

$$N_{i,T} = \frac{\frac{pr_0 + c_i}{P}}{q_i[1 - \frac{h}{B_s}\frac{(pr_0 + c_i)}{P}]} \tag{8}$$

Y su derivada:

$$\frac{\delta f}{\delta pr_0} = \sum_{i=1}^{n} \frac{1}{q_i(pr_0 + c_i)(1 - \frac{h}{B_s P}(pr_0 + c_i))^2} \tag{9}$$

Pseudo-código El pseudocódigo del algoritmo en nuestro caso es:

```
newtonIterationFunction <- function (x) {
for(i in 1:length(No)){
        coc<- h/Bs;
        N[i]<- (x+c[i])/(P*q[i]-q[i]*coc*(x+c[i]));
        E[i] <- (coc*(No[i]-N[i])-ln(N[i]/N[0]/q[i]));
        dE[i]<- 1/(q[i]*(x+c[i])*(1-coc/P*(x+c[i]))^2);
}
  return  x - (ET-sum(E))/ (-sum(dE));
}

pr0 <- 0.5

for (i in 0:99) {
    "Iteraciones: " + i
```

```
    "Valor aproximado: " + x
    pr0old <- pr0
    pr0 <- newtonIterationFunction(pr0)
    if (pr0 = pr0old) {
        "Solución encontrada!"
        break
    }
}
```

5.2. Algoritmo para el cálculo de la distribución del esfuerzo por áreas bajo condiciones de esfuerzo total limitado según la IDF

1. Ordenamos áreas por:

   a) $N_{i,0}$ si las q_i, c_i, y P son iguales para todas las áreas.

   b) $CPUE_{i,0}$ si los q_i son distintos entre áreas.

$$CPUE_{i,0} = \frac{q_i}{\frac{1}{N_{i,0}} + \frac{q_i.h}{B_s}} \tag{10}$$

   c) $pr_{i,0}$ si los c_i son distintos entre áreas.

$$pr_{i,0} = \frac{P.q_i}{\frac{1}{N_{i,0}} + \frac{q_i.h}{B_s}} - c_i \tag{11}$$

   Nosotros las ordenaremos por $pr_{i,0}$ por ser el caso más general (ver Ordenación de las áreas por productividad).

2. Calculamos los esfuerzos parciales (EP) que se requieren para ir llevando a la población a los niveles en los que se van igualando sucesivamente las $pr_{i,t}$ de las distintas áreas. Así, $EP1$ sería el esfuerzo que se requeriría para igualar las productividades de la población con productividad mayor y la siguiente en productividad. Esto se puede hacer mediante una función del tipo:

```
IgualarPops <- function(areas){
# Donde 'areas' es el número de áreas de la población
areasiguales <-0
for (j in 1:areas-1){
  EP[j] <-0;
  for (i in 1:(j+1)){ #Calcula los elementos del EP
   coc<-h/Bs;
   Nfin<-(pr[j+1]+c[i])/(P*q[i]-q[i]*coc*(pr[j+1]+c[i]));
   EP[j]=EP[j]+coc*(N0[i]-Nfin)-ln(Nfin/N0[i])/q[i];
  }
```

```
  # Mira si se sobrepaso el ET y para el bucle.
  if(sum(EP[1:j])>ET){break;} else {areasiguales=j+1}
 }
 return EP;
}
```

3. Mientras que el esfuerzo parcial sea menor que el esfuerzo total se calculan los esfuerzos parciales necesarios para ir igualando cada vez más áreas. En cuanto se sobrepasa ese esfuerzo total, sabemos que se han igualado i poblaciones.

4. Se calcula la pr_0 para esas i áreas sabiendo que va a tener un valor entre $pr_{i,0}$ y $pr_{i+1,0}$ (o 0 en el caso de que se hayan igualado todas las áreas). Al resto de las áreas se les otorga esfuerzo 0.

Ordenación de las áreas por productividad En Excel, hicimos una solución *provisional* al problema de ordenar las áreas por productividad. Creamos un vector **orden** que indicará en orden decreciente de productividad los indicadores de área. Es decir si **orden** $= (2, 3, 1, 4)$ el área con mayor productividad es la 2 y la de menor productividad es la 4. Este vector lo utilizamos como índices en los cálculos posteriores. Para crear este vector **orden** se utilizó el algoritmo Quicksort, pasandole una matriz con los datos de productividad que queremos ordenar y sus indices de area asociados, y una vez ordenados nos quedamos con los indices de área.