

# Tests loi de mortalité pour les particules Ichthyop

Amaël Dupaix

6/4/2021

## Contents

<b>Loi exponentielle</b>	<b>1</b>
<b>Simulation Ichthyop</b>	<b>2</b>
Mortalité appliquée à chaque particule . . . . .	2
Mortalité appliquée après, sur la matrice . . . . .	5
Comparaison des mortalités . . . . .	5

### Document pour comparer les résultats avec les deux types de mortalité

**Conclusion:** les deux méthodes de mortalité ne change pas beaucoup les résultats. Les différences sont observées au bout de longtemps, environ 2 fois la durée de vie moyenne. Mais comme on va relâcher des particules toutes les 4 semaines probablement, cela ne pose pas de problème parce que les différences seront “masquées” par les autres simulations.

## Loi exponentielle

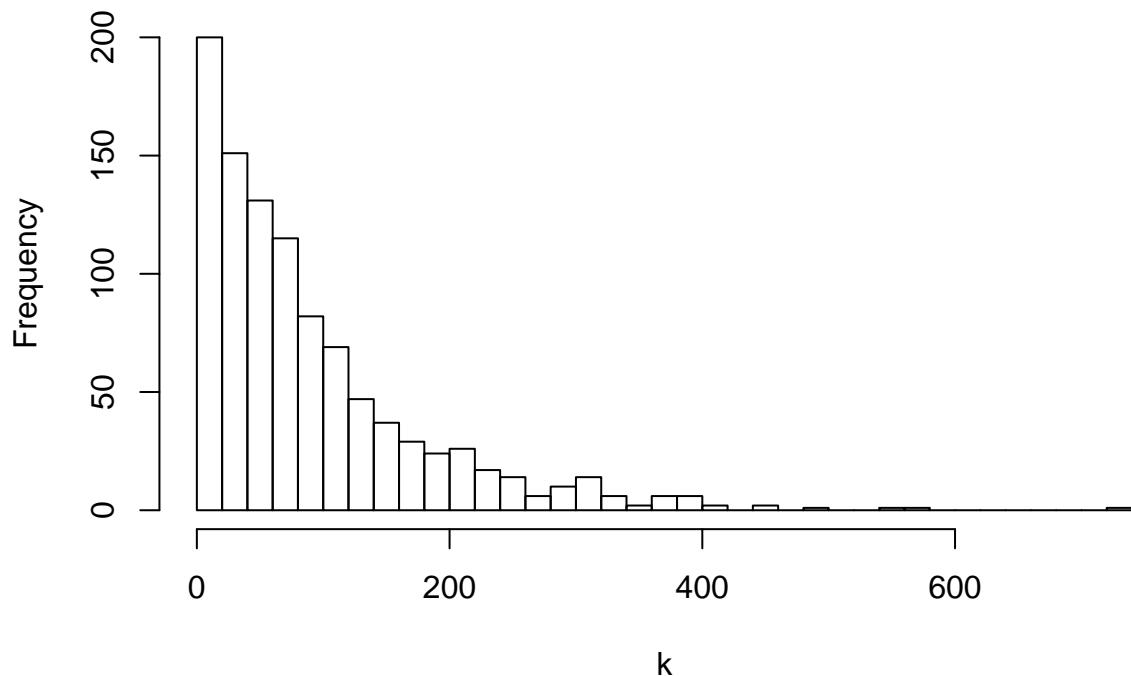
```
set.seed(123456)

k = rep(1, 1000)
p = 1/90 # durée de vie moyenne = 90 jours (pour le test)

for(i in 1:length(k)){ nbs = runif(2000) ; while(nbs[k[i]] > p){k[i] = k[i] + 1} }

hist(k, breaks = 50)
```

## Histogram of k



Si on ne veut pas supprimer de valeurs, une simulation de 90 jours, par exemple, demande de simuler la dérive des particules pendant presque 1000 jours... Pour une simulation de 180 jours, certaines durées de vie atteignent 1400 jours. Est-ce qu'on ne pourrait pas envisager de tronquer la queue de la distribution ?

## Simulation Ichthyop

J'ai simulé 1000 particules, relachées d'un point unique, au nord de Madagascar, pendant 1000 jours. On simule une mortalité à 180 jours en moyenne, car c'est moins "lourd". Mais cela ne change rien aux résultats (par rapport à une mortalité moyenne à 360 jours par exemple), ce qu'on veut c'est comparer deux méthodes d'applications de la mortalité.

Certaines durées de vie simulées avec la première méthode sont supérieures à 1000 jours (4 particules sur 1000). Dans ce cas on coupe la trajectoire à 1000 jours.

## Mortalité appliquée à chaque particule

```
#'******  
# Adapté de nc.to.Array()  
#'******  
  
nc <- open.nc("/home/adupaix/Documents/These/Axe_1/Hist_FOB_env/3-Launch_Ichthyop_datarmor/ichthyop-out"  
gsize = 2
```

```

#~~ 1. Simulation de la mortalité pour chaque particule
mort <- var.get.nc(nc, "mortality")

k = rep(1, 1000)
p = 1/180 # durée de vie moyenne = 180 jours

for(i in 1:length(k)){ nbs = runif(2000) ; while(nbs[k[i]] > p){k[i] = k[i] + 1} }

# Pour avoir le pas de temps, on divise par 7 (une position tous les 7 jours)
k = round(k / 7) + 1
## On tronque la courbe exponentielle: les valeurs supérieures à 1000 jours sont ramenées à 1000
k[which(k > dim(mort)[2])] <- dim(mort)[2]

# met à jour la mortalité
for (i in 1:1000){mort[i, (k[i]:dim(mort)[2])] = 1}

#~~ 2. Recupere tous les pas de temps de la simulation
time <- var.get.nc(nc, variable = "time")

t0 <- "year 1900 month 01 day 01 at 00:00"

position_date <- as.POSIXct(t0, tz="GMT", format = "year %Y month %m day %d at %H:%M") +
  as.difftime(time/(3600*24),units = "days")

#~~ 3. Calcul le nombre de particules par pas de temps
nb_p_per_timestep <- foreach(i = seq(1, length(position_date),1),
                                .packages = c("raster"),
                                .combine = function(x,y) abind::abind(x,y, along = 3),
                                .export = c("create.raster")) %do% {

  longitude <- var.get.nc(nc, variable = "lon", start = c(1,i), count = c(1))
  latitude <- var.get.nc(nc, variable = "lat", start = c(1,i), count = c(1))
  mortality.i <- mort[,i] #lit la mortalité pour le pas de temps i

  # remove not released, filtered (discharge threshold or ltime filter 1)
  longitude <- longitude[which(mortality.i == 0)]
  latitude <- latitude[which(mortality.i == 0)]

  # create a raster of the area of interest
  r.i <- create.raster(gsize)

  # get the cell number where each particle is
  cell_pos <- cellFromXY(r.i, cbind(longitude,latitude))

  # count the number of particles in each cell, applying the weight
  nb_p_per_cell <- table(cell_pos)

  # fill the raster
  # the attribute of nb_p_per_cell used contains the cell number

  r.i[ as.numeric(attr(nb_p_per_cell, "dimnames")$cell_pos) ] <- nb_p_per_

```

```

        as.matrix(r.i)

    }

dimnames(nb_p_per_timestep)[[3]] <- format(position_date, "%Y-%m-%d_%H:%M:%S")

df_list = list()
p_list = list()

for (i in 1:dim(nb_p_per_timestep)[3]){
  rast <- create.raster(2)
  rast[] <- nb_p_per_timestep[, , i]

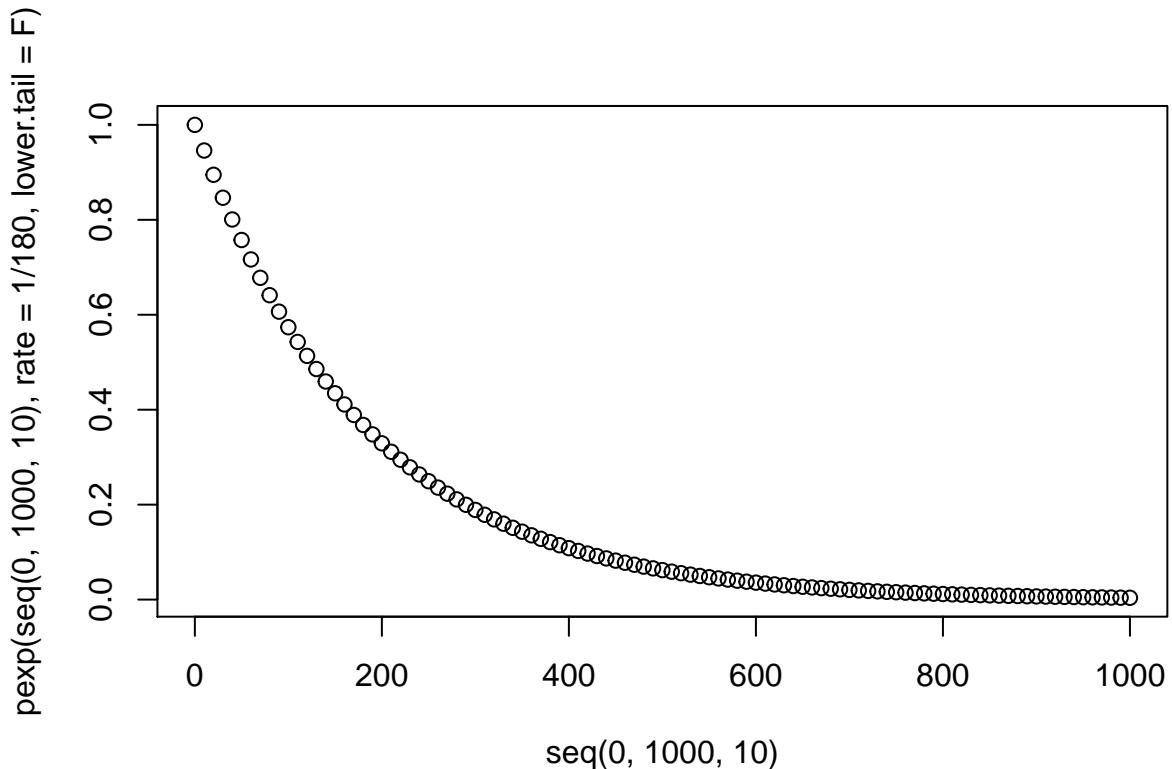
  df_list[[i]] <- as.data.frame(rast, xy = T)

  p_list[[2*i-1]] <- ggplot() +
    geom_raster(data = df_list[[i]],
                 aes(x=x, y=y, fill=occ))+
    scale_fill_gradient(high = "red")+
    geom_sf(data = coastline10)+ 
    coord_sf(xlim = xlm, ylim = ylm)+ 
    ggtitle(paste0("Week ", i, " - mortality per particule"))+
    theme(plot.title = element_text(hjust=0.5)
          )
}

}

```

## Mortalité appliquée après, sur la matrice



```
##  
## Attaching package: 'crayon'  
  
## The following object is masked from 'package:ggplot2':  
##  
##     %+%
```

## Comparaison des mortalités

Les deux méthodes collent plutôt bien jusqu'aux environs de la semaine 50 (soit 2 fois la durée de vie moyenne des particules). Ensuite, cela se différencie:

- avec la méthode de mortalité par particule, il reste seulement quelques particules, d'où quelques points très localisés.
- avec la méthode de mortalité appliquée directement sur les matrices, il reste l'ensemble des particules, mais pondérées de telle manière que les valeurs maximales sont extrêmement faibles (*cf.* les échelles des figures).

