TP 4 analyse de survie avec données sur animaux marqués

Olivier Gimenez

19/12/2020

On charge les packages `RMark` et `R2ucare`, ce dernier servant à tester les hypothèse des modèles de capture-recapture en population ouverte.

```
library(RMark)
library(R2ucare)
```

Partie 1 : Estimation de la survie, exemple du cincle plongeur

Les données.

```
cincle <- convert.inp("dat/cincle-plongeur.inp")
```

On jette un coup d'oeil.

```
head(cincle)
```

```
##        ch freq
## 1 0000010   23
## 2 0000011   23
## 3 0000100   16
## 4 0000110    9
## 5 0000111   16
## 6 0001000   16
```

On prépare les données.

```
cincle.proc <- process.data(cincle,
                              begin.time = 1,
                              model = "CJS")
cincle.ddl <- make.design.data(cincle.proc)
```

On inspecte la structure pour la survie.

```
head(cincle.ddl$Phi)
```

```
##   par.index model.index group cohort age time occ.cohort Cohort Age Time
## 1         1           1     1      1   0    1          1      0   0    0
```

```
## 2           2           2     1      1   1   2          1      0   1    1
## 3           3           3     1      1   2   3          1      0   2    2
## 4           4           4     1      1   3   4          1      0   3    3
## 5           5           5     1      1   4   5          1      0   4    4
## 6           6           6     1      1   5   6          1      0   5    5
```

Et la détection.

```r
head(cincle.ddl$p)
```

```
##   par.index model.index group cohort age time occ.cohort Cohort Age Time
## 1         1          22     1      1   1    2          1      0   1    0
## 2         2          23     1      1   2    3          1      0   2    1
## 3         3          24     1      1   3    4          1      0   3    2
## 4         4          25     1      1   4    5          1      0   4    3
## 5         5          26     1      1   5    6          1      0   5    4
## 6         6          27     1      1   6    7          1      0   6    5
```

On spécifie les effets sur les paramètres.

```r
phit <- list(formula=~time)
phi <- list(formula=~1)
pt <- list(formula=~time)
p <- list(formula=~1)
```

On ajuste le modèle Cormack-Jolly-Seber (CJS).

```r
cjs.cincle <- mark(cincle.proc,
                   cincle.ddl,
                   model.parameters = list(Phi = phit, p = pt))
```

```
##
## Output summary for CJS model
## Name : Phi(~time)p(~time)
##
## Npar :  12  (unadjusted=11)
## -2lnL:  656.9502
## AICc :  681.7057  (unadjusted=679.58789)
##
## Beta
##                   estimate           se           lcl          ucl
## Phi:(Intercept)  0.9354608    0.7685290    -0.5708561    2.4417777
## Phi:time2       -1.1982802    0.8706768    -2.9048067    0.5082464
## Phi:time3       -1.0228344    0.8049213    -2.6004801    0.5548113
## Phi:time4       -0.4198637    0.8091545    -2.0058065    1.1660791
## Phi:time5       -0.5361028    0.8031500    -2.1102769    1.0380713
## Phi:time6        0.2481345 1274.0678000 -2496.9247000 2497.4210000
## p:(Intercept)    0.8292795    0.7837387    -0.7068484    2.3654074
## p:time3          1.6556275    1.2913796    -0.8754765    4.1867315
## p:time4          1.5220955    1.0729180    -0.5808238    3.6250148
## p:time5          1.3767446    0.9884837    -0.5606836    3.3141728
## p:time6          1.7950938    1.0688799    -0.2999108    3.8900985
```

```
## p:time7         -0.0147544  973.0311800 -1907.1559000 1907.1264000
##
##
## Real Parameter Phi
##
##           1         2         3         4         5         6
## 1 0.7181818 0.4346708 0.4781705 0.6261176 0.5985334 0.7655936
## 2           0.4346708 0.4781705 0.6261176 0.5985334 0.7655936
## 3                     0.4781705 0.6261176 0.5985334 0.7655936
## 4                               0.6261176 0.5985334 0.7655936
## 5                                         0.5985334 0.7655936
## 6                                                   0.7655936
##
##
## Real Parameter p
##
##           2         3         4         5         6         7
## 1 0.6962026 0.9230769 0.9130435 0.9007892 0.9324138 0.6930729
## 2           0.9230769 0.9130435 0.9007892 0.9324138 0.6930729
## 3                     0.9130435 0.9007892 0.9324138 0.6930729
## 4                               0.9007892 0.9324138 0.6930729
## 5                                         0.9324138 0.6930729
## 6                                                   0.6930729
```

Inspectons les résultats.

```
cjs.cincle$results$real
```

```
##                   estimate          se           lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.7181818   0.1555477  3.610393e-01 0.9199581
## Phi g1 c1 a1 t2 0.4346708   0.0688290  3.075047e-01 0.5710588
## Phi g1 c1 a2 t3 0.4781705   0.0597091  3.643838e-01 0.5942685
## Phi g1 c1 a3 t4 0.6261176   0.0592656  5.048461e-01 0.7333741
## Phi g1 c1 a4 t5 0.5985334   0.0560517  4.855434e-01 0.7019411
## Phi g1 c1 a5 t6 0.7655936 228.6437300 1.816826e-308 1.0000000
## p g1 c1 a1 t2   0.6962026   0.1657643  3.302956e-01 0.9141511
## p g1 c1 a2 t3   0.9230769   0.0728778  6.161495e-01 0.9889758
## p g1 c1 a3 t4   0.9130435   0.0581758  7.140648e-01 0.9778505
## p g1 c1 a4 t5   0.9007892   0.0538330  7.360176e-01 0.9672855
## p g1 c1 a5 t6   0.9324138   0.0458025  7.684926e-01 0.9828579
## p g1 c1 a6 t7   0.6930729 206.9855000 1.256111e-308 1.0000000
```

Les PIM pour CJS.

```
PIMS(cjs.cincle,"Phi")
```

```
## group = Group 1
##    1  2  3  4  5  6
## 1  1  2  3  4  5  6
## 2     2  3  4  5  6
## 3        3  4  5  6
## 4           4  5  6
## 5              5  6
## 6                 6
```

On fait tourner le modèle avec paramètres constants.

```r
phip.cincle <- mark(cincle.proc,
                    cincle.ddl,
                    model.parameters = list(Phi = phi, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~1)p(~1)
##
## Npar :  2
## -2lnL:  666.8377
## AICc :  670.866
##
## Beta
##                  estimate        se       lcl       ucl
## Phi:(Intercept) 0.2421484 0.1020127 0.0422034 0.4420933
## p:(Intercept)   2.2262661 0.3251094 1.5890517 2.8634805
##
##
## Real Parameter Phi
##
##          1        2        3        4        5        6
## 1 0.560243 0.560243 0.560243 0.560243 0.560243 0.560243
## 2          0.560243 0.560243 0.560243 0.560243 0.560243
## 3                   0.560243 0.560243 0.560243 0.560243
## 4                            0.560243 0.560243 0.560243
## 5                                     0.560243 0.560243
## 6                                              0.560243
##
##
## Real Parameter p
##
##           2         3         4         5         6         7
## 1 0.9025835 0.9025835 0.9025835 0.9025835 0.9025835 0.9025835
## 2           0.9025835 0.9025835 0.9025835 0.9025835 0.9025835
## 3                     0.9025835 0.9025835 0.9025835 0.9025835
## 4                               0.9025835 0.9025835 0.9025835
## 5                                         0.9025835 0.9025835
## 6                                                   0.9025835
```

Les résultats.

```r
phip.cincle$results$real
```

```
##                  estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.5602430 0.0251330 0.5105493 0.6087577
## p g1 c1 a1 t2   0.9025835 0.0285857 0.8304826 0.9460113
```

Les PIM.

```r
PIMS(phip.cincle,"Phi")
```

```
## group = Group 1
##    1  2  3  4  5  6
## 1  1  1  1  1  1  1
## 2     1  1  1  1  1
## 3        1  1  1  1
## 4           1  1  1
## 5              1  1
## 6                 1
```

```r
PIMS(phip.cincle,"p")
```

```
## group = Group 1
##    2  3  4  5  6  7
## 1  2  2  2  2  2  2
## 2     2  2  2  2  2
## 3        2  2  2  2
## 4           2  2  2
## 5              2  2
## 6                 2
```

On ajoute les covariables environnementales.

```r
cov.cincle <- readxl::read_xls("dat/covariables-environnementales-cincle-plongeur.xls")
cov.cincle
```

```
## # A tibble: 7 x 3
##   année `débit (l/sec)` `temperature hiver (°C)`
##   <dbl>           <dbl>                    <dbl>
## 1  1981             443                     -2.3
## 2  1982            1114                     -0.4
## 3  1983             529                     -1.2
## 4  1984             434                     -4.2
## 5  1985             627                     -3
## 6  1986             466                     -2.8
## 7  1987             730                      0.1
```

On simplifie le nom des colonnes.

```r
cov.cincle <- janitor::clean_names(cov.cincle)
cov.cincle
```

```
## # A tibble: 7 x 3
##   annee debit_l_sec temperature_hiver_c
##   <dbl>       <dbl>               <dbl>
## 1  1981         443                -2.3
## 2  1982        1114                -0.4
## 3  1983         529                -1.2
## 4  1984         434                -4.2
## 5  1985         627                -3
## 6  1986         466                -2.8
## 7  1987         730                 0.1
```

On a 7 occasions de capture, donc 6 paramètres de survie. Si on suppose que la première année de capture dans le jeu de données cincle est 1981, alors on peut estimer la survie entre 1981 et 1982, à laquelle on applique la valeur de covariable en 1981, etc… jusqu'à la survie entre 1986 et 1987 à laquelle s'applique la valeur de covariable de 1986, donc on n'a pas besoin de la dernière ligne dans le jeu de données.

```
cov.cincle <- cov.cincle[!(cov.cincle$annee == "1987"),]
```

Jetons un coup d'oeil à la structure sur la survie.

```
cincle.ddl$Phi
```

```
##    par.index model.index group cohort age time occ.cohort Cohort Age Time
## 1          1           1     1      1   0    1          1      0   0    0
## 2          2           2     1      1   1    2          1      0   1    1
## 3          3           3     1      1   2    3          1      0   2    2
## 4          4           4     1      1   3    4          1      0   3    3
## 5          5           5     1      1   4    5          1      0   4    4
## 6          6           6     1      1   5    6          1      0   5    5
## 7          7           7     1      2   0    2          2      1   0    1
## 8          8           8     1      2   1    3          2      1   1    2
## 9          9           9     1      2   2    4          2      1   2    3
## 10        10          10     1      2   3    5          2      1   3    4
## 11        11          11     1      2   4    6          2      1   4    5
## 12        12          12     1      3   0    3          3      2   0    2
## 13        13          13     1      3   1    4          3      2   1    3
## 14        14          14     1      3   2    5          3      2   2    4
## 15        15          15     1      3   3    6          3      2   3    5
## 16        16          16     1      4   0    4          4      3   0    3
## 17        17          17     1      4   1    5          4      3   1    4
## 18        18          18     1      4   2    6          4      3   2    5
## 19        19          19     1      5   0    5          5      4   0    4
## 20        20          20     1      5   1    6          5      4   1    5
## 21        21          21     1      6   0    6          6      5   0    5
```

On crée une survie qui depend du débit.

```
cincle.ddl$Phi$debit <- 0 # nv var mise a 0
for (i in 1:nrow(cov.cincle)){
   cincle.ddl$Phi$debit[cincle.ddl$Phi$time == i] <- as.numeric(cov.cincle[i, "debit_l_sec"])
}
```

On vérifie que ça a marché.

```
cincle.ddl$Phi
```

```
##    par.index model.index group cohort age time occ.cohort Cohort Age Time debit
## 1          1           1     1      1   0    1          1      0   0    0   443
## 2          2           2     1      1   1    2          1      0   1    1  1114
## 3          3           3     1      1   2    3          1      0   2    2   529
## 4          4           4     1      1   3    4          1      0   3    3   434
## 5          5           5     1      1   4    5          1      0   4    4   627
## 6          6           6     1      1   5    6          1      0   5    5   466
```

```
## 7          7           7     1      2   0    2          2      1   0    1  1114
## 8          8           8     1      2   1    3          2      1   1    2   529
## 9          9           9     1      2   2    4          2      1   2    3   434
## 10        10          10     1      2   3    5          2      1   3    4   627
## 11        11          11     1      2   4    6          2      1   4    5   466
## 12        12          12     1      3   0    3          3      2   0    2   529
## 13        13          13     1      3   1    4          3      2   1    3   434
## 14        14          14     1      3   2    5          3      2   2    4   627
## 15        15          15     1      3   3    6          3      2   3    5   466
## 16        16          16     1      4   0    4          4      3   0    3   434
## 17        17          17     1      4   1    5          4      3   1    4   627
## 18        18          18     1      4   2    6          4      3   2    5   466
## 19        19          19     1      5   0    5          5      4   0    4   627
## 20        20          20     1      5   1    6          5      4   1    5   466
## 21        21          21     1      6   0    6          6      5   0    5   466
```

Idem pour temperature.

```
cincle.ddl$Phi$temp <- 0 # nv var mise a 0
for (i in 1:nrow(cov.cincle)){
   cincle.ddl$Phi$temp[cincle.ddl$Phi$time == i] <- as.numeric(cov.cincle[i, "temperature_hiver_c"])
}
cincle.ddl$Phi
```

```
##    par.index model.index group cohort age time occ.cohort Cohort Age Time debit
## 1          1           1     1      1   0    1          1      0   0    0   443
## 2          2           2     1      1   1    2          1      0   1    1  1114
## 3          3           3     1      1   2    3          1      0   2    2   529
## 4          4           4     1      1   3    4          1      0   3    3   434
## 5          5           5     1      1   4    5          1      0   4    4   627
## 6          6           6     1      1   5    6          1      0   5    5   466
## 7          7           7     1      2   0    2          2      1   0    1  1114
## 8          8           8     1      2   1    3          2      1   1    2   529
## 9          9           9     1      2   2    4          2      1   2    3   434
## 10        10          10     1      2   3    5          2      1   3    4   627
## 11        11          11     1      2   4    6          2      1   4    5   466
## 12        12          12     1      3   0    3          3      2   0    2   529
## 13        13          13     1      3   1    4          3      2   1    3   434
## 14        14          14     1      3   2    5          3      2   2    4   627
## 15        15          15     1      3   3    6          3      2   3    5   466
## 16        16          16     1      4   0    4          4      3   0    3   434
## 17        17          17     1      4   1    5          4      3   1    4   627
## 18        18          18     1      4   2    6          4      3   2    5   466
## 19        19          19     1      5   0    5          5      4   0    4   627
## 20        20          20     1      5   1    6          5      4   1    5   466
## 21        21          21     1      6   0    6          6      5   0    5   466
##    temp
## 1  -2.3
## 2  -0.4
## 3  -1.2
## 4  -4.2
## 5  -3.0
## 6  -2.8
```

```
## 7  -0.4
## 8  -1.2
## 9  -4.2
## 10 -3.0
## 11 -2.8
## 12 -1.2
## 13 -4.2
## 14 -3.0
## 15 -2.8
## 16 -4.2
## 17 -3.0
## 18 -2.8
## 19 -3.0
## 20 -2.8
## 21 -2.8
```

On définit les effets.

```
phi.debitptemp <- list(formula =~ debit + temp)
```

On ajuste le modèle.

```
phicov.cincle <- mark(cincle.proc,
                      cincle.ddl,
                      model.parameters = list(Phi = phi.debitptemp, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~debit + temp)p(~1)
##
## Npar :  4
## -2lnL:  660.53
## AICc :  668.625
##
## Beta
##                      estimate           se        lcl       ucl
## Phi:(Intercept) -2.883125e-01 0.6383632000 -1.5395043 0.9628794
## Phi:debit        3.500799e-05 0.0006604623 -0.0012595 0.0013295
## Phi:temp        -2.095950e-01 0.1170475000 -0.4390081 0.0198181
## p:(Intercept)    2.235034e+00 0.3250918000  1.5978546 2.8722145
##
##
## Real Parameter Phi
##
##          1         2         3         4         5         6
## 1 0.552126 0.4587252 0.4954303 0.6472972 0.5896267 0.5780728
## 2          0.4587252 0.4954303 0.6472972 0.5896267 0.5780728
## 3                    0.4954303 0.6472972 0.5896267 0.5780728
## 4                              0.6472972 0.5896267 0.5780728
## 5                                        0.5896267 0.5780728
## 6                                                  0.5780728
##
##
```

```
## Real Parameter p
##
##           2         3         4         5         6         7
## 1 0.9033518 0.9033518 0.9033518 0.9033518 0.9033518 0.9033518
## 2           0.9033518 0.9033518 0.9033518 0.9033518 0.9033518
## 3                     0.9033518 0.9033518 0.9033518 0.9033518
## 4                               0.9033518 0.9033518 0.9033518
## 5                                         0.9033518 0.9033518
## 6                                                   0.9033518
```

Les paramètres estimés.

```r
phicov.cincle$results$real
```

```
##                  estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.5521260 0.0380925 0.4768510 0.6250857
## Phi g1 c1 a1 t2 0.4587252 0.0640473 0.3382614 0.5842149
## Phi g1 c1 a2 t3 0.4954303 0.0515130 0.3959964 0.5952270
## Phi g1 c1 a3 t4 0.6472972 0.0421823 0.5609558 0.7249834
## Phi g1 c1 a4 t5 0.5896267 0.0319296 0.5259226 0.6504599
## Phi g1 c1 a5 t6 0.5780728 0.0299046 0.5186304 0.6353361
## p g1 c1 a1 t2   0.9033518 0.0283829 0.8317183 0.9464557
```

Visualisons la relation survie vs. débit pour une valeur moyenne de température. On créé d'abord une grille pour le débit.

```r
min.debit <- min(cov.cincle$debit_l_sec)
max.debit <- max(cov.cincle$debit_l_sec)
debit.values <- seq(from = min.debit, to = max.debit, by = 5)
temp.values <- c(quantile(cov.cincle$temperature_hiver_c, 0.05),
                 mean(cov.cincle$temperature_hiver_c),
                 quantile(cov.cincle$temperature_hiver_c, 0.95))
```

Construit le jeu de données.

```r
pred.dat <- expand.grid(debit = debit.values,
                        temp = temp.values)
pred.dat <- cbind(1, pred.dat)
pred.dat <- as.matrix(pred.dat)
```

On fait la prédiction, sur l'échelle logit.

```r
betas.phi <- phicov.cincle$results$beta[1:3,1]
pred.surv.logit <- pred.dat %*% betas.phi
```

On back-transforme et on arrange.

```r
pred.surv <- plogis(pred.surv.logit)
pred.df <- cbind(pred.dat[,-1], pred.surv)
colnames(pred.df) <- c("debit", "temp", "survie")
pred.df <- as.data.frame(pred.df)
head(pred.df)
```

```
##   debit temp    survie
## 1   434 -3.9 0.6328125
## 2   439 -3.9 0.6328532
## 3   444 -3.9 0.6328938
## 4   449 -3.9 0.6329345
## 5   454 -3.9 0.6329752
## 6   459 -3.9 0.6330158
```

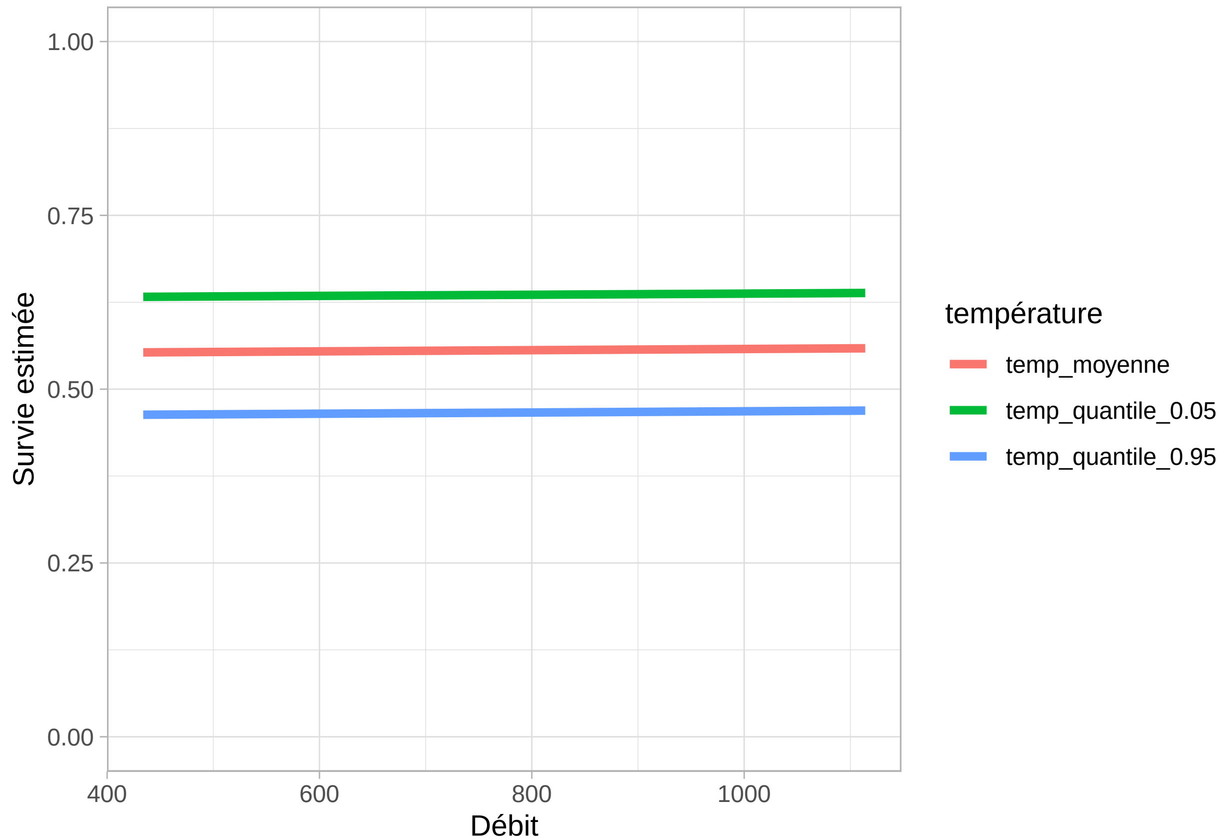
On prépare les données.

```
pred.df$temp <- ifelse(pred.df$temp == mean(cov.cincle$temperature_hiver_c),
                           "temp_moyenne",
                           ifelse(pred.df$temp == quantile(cov.cincle$temperature_hiver_c, 0.05),
                                  "temp_quantile_0.05",
                                  "temp_quantile_0.95"))
head(pred.df)
```

```
##   debit               temp    survie
## 1   434 temp_quantile_0.05 0.6328125
## 2   439 temp_quantile_0.05 0.6328532
## 3   444 temp_quantile_0.05 0.6328938
## 4   449 temp_quantile_0.05 0.6329345
## 5   454 temp_quantile_0.05 0.6329752
## 6   459 temp_quantile_0.05 0.6330158
```

On visualise.

```
library(ggplot2)
ggplot(pred.df, aes(x = debit, y = survie)) +
   geom_line(aes(color = temp), size = 1.5) +
   labs(x = "Débit",
        y = "Survie estimée",
        color = "température") +
    ylim(0, 1) +
   theme_light()
```



Partie 2 : Estimation de la survie, exemple du martinet noir

Les données.

```
martinet <- convert.inp("dat/martinet-noir.inp",
                        group.df = data.frame(colonie = c("nord", "sud")),
                        covariates = NULL)
head(martinet)
```

```
##            ch freq colonie
## 1:1 00000001    7    nord
## 1:2 00000010    6    nord
## 1:3 00000011    1    nord
## 1:4 00000100    1    nord
## 1:8 00001000    1    nord
## 1:9 00001110    1    nord
```

On prépare les données.

```
martinet.proc <- process.data(martinet,
                              begin.time = 1,
                              model = "CJS",
                              groups = ("colonie"))
martinet.ddl <- make.design.data(martinet.proc)
```

On spécifie les effets sur les paramètres.

```
phit <- list(formula=~time)
phi <- list(formula=~1)
pt <- list(formula=~time)
p <- list(formula=~1)
```

Fait tourner modèle CJS, et examine les paramètres estimés.

```
cjs.martinet <- mark(martinet.proc,
                     martinet.ddl,
                     model.parameters = list(Phi = phit, p = pt))
```

```
##
## Output summary for CJS model
## Name : Phi(~time)p(~time)
##
## Npar :  14  (unadjusted=13)
## -2lnL:  354.9445
## AICc :  385.1905  (unadjusted=382.88072)
##
## Beta
##                    estimate           se          lcl          ucl
## Phi:(Intercept)  1.7439684    0.8654869    0.0476141    3.4403227
## Phi:time2       -0.9669983    1.0306490   -2.9870704    1.0530738
## Phi:time3       -0.5738963    1.1624682   -2.8523340    1.7045414
## Phi:time4       -0.8957157    1.0338553   -2.9220722    1.1306408
## Phi:time5       -0.9809801    0.9802287   -2.9022283    0.9402682
## Phi:time6       -0.6912500    1.0551098   -2.7592653    1.3767653
## Phi:time7       -1.8256772 1057.4144000 -2074.3579000 2070.7065000
## p:(Intercept)    2.0030691    1.0495416   -0.0540324    4.0601707
## p:time3         -0.9689950    1.1967009   -3.3145288    1.3765388
## p:time4         -1.9340766    1.1630687   -4.2136912    0.3455380
## p:time5         -1.2041772    1.1750418   -3.5072591    1.0989048
## p:time6         -0.0882492    1.2916860   -2.6199538    2.4434554
## p:time7         -0.0861472    1.4799823   -2.9869125    2.8146182
## p:time8         -1.1127912 1890.7083000 -3706.9011000 3704.6755000
##
##
## Real Parameter Phi
## Group:colonienord
##           1         2        3         4         5         6         7
## 1 0.8511904 0.6850267 0.763158 0.7002005 0.6820022 0.7412966 0.4795841
## 2           0.6850267 0.763158 0.7002005 0.6820022 0.7412966 0.4795841
## 3                     0.763158 0.7002005 0.6820022 0.7412966 0.4795841
## 4                              0.7002005 0.6820022 0.7412966 0.4795841
## 5                                        0.6820022 0.7412966 0.4795841
## 6                                                  0.7412966 0.4795841
## 7                                                            0.4795841
##
## Group:coloniesud
##           1         2        3         4         5         6         7
## 1 0.8511904 0.6850267 0.763158 0.7002005 0.6820022 0.7412966 0.4795841
```

```
## 2           0.6850267 0.763158 0.7002005 0.6820022 0.7412966 0.4795841
## 3                     0.763158 0.7002005 0.6820022 0.7412966 0.4795841
## 4                              0.7002005 0.6820022 0.7412966 0.4795841
## 5                                        0.6820022 0.7412966 0.4795841
## 6                                                  0.7412966 0.4795841
## 7                                                            0.4795841
##
##
## Real Parameter p
## Group:colonienord
##           2        3         4         5         6         7         8
## 1 0.8811189 0.737705 0.5172413 0.6897374 0.8715597 0.8717948 0.7089475
## 2           0.737705 0.5172413 0.6897374 0.8715597 0.8717948 0.7089475
## 3                    0.5172413 0.6897374 0.8715597 0.8717948 0.7089475
## 4                              0.6897374 0.8715597 0.8717948 0.7089475
## 5                                        0.8715597 0.8717948 0.7089475
## 6                                                  0.8717948 0.7089475
## 7                                                            0.7089475
##
## Group:coloniesud
##           2        3         4         5         6         7         8
## 1 0.8811189 0.737705 0.5172413 0.6897374 0.8715597 0.8717948 0.7089475
## 2           0.737705 0.5172413 0.6897374 0.8715597 0.8717948 0.7089475
## 3                    0.5172413 0.6897374 0.8715597 0.8717948 0.7089475
## 4                              0.6897374 0.8715597 0.8717948 0.7089475
## 5                                        0.8715597 0.8717948 0.7089475
## 6                                                  0.8717948 0.7089475
## 7                                                            0.7089475
```

```
cjs.martinet$results$real
```

```
##                      estimate          se           lcl       ucl fixed note
## Phi gnord c1 a0 t1 0.8511904   0.1096271  5.119013e-01 0.9689412
## Phi gnord c1 a1 t2 0.6850267   0.1013890  4.640514e-01 0.8452711
## Phi gnord c1 a2 t3 0.7631580   0.1402705  4.131404e-01 0.9365019
## Phi gnord c1 a3 t4 0.7002005   0.1187097  4.353323e-01 0.8761681
## Phi gnord c1 a4 t5 0.6820022   0.0998051  4.653068e-01 0.8409044
## Phi gnord c1 a5 t6 0.7412966   0.1157330  4.675200e-01 0.9033959
## Phi gnord c1 a6 t7 0.4795841 263.9127500 5.126238e-309 1.0000000
## p gnord c1 a1 t2   0.8811189   0.1099378  4.864952e-01 0.9830463
## p gnord c1 a2 t3   0.7377050   0.1112487  4.768147e-01 0.8966881
## p gnord c1 a3 t4   0.5172413   0.1251483  2.863172e-01 0.7410289
## p gnord c1 a4 t5   0.6897374   0.1130732  4.410916e-01 0.8622989
## p gnord c1 a5 t6   0.8715597   0.0842866  6.080349e-01 0.9674088
## p gnord c1 a6 t7   0.8717948   0.1166266  4.679767e-01 0.9813322
## p gnord c1 a7 t8   0.7089475 390.1302700 1.354962e-308 1.0000000
```

PIM pour CJS.

```
PIMS(cjs.martinet,"Phi")
```

```
## group = colonienord
```

```
##    1  2  3  4  5  6  7
## 1  1  2  3  4  5  6  7
## 2     2  3  4  5  6  7
## 3        3  4  5  6  7
## 4           4  5  6  7
## 5              5  6  7
## 6                 6  7
## 7                    7
## group = coloniesud
##    1  2  3  4  5  6  7
## 1  1  2  3  4  5  6  7
## 2     2  3  4  5  6  7
## 3        3  4  5  6  7
## 4           4  5  6  7
## 5              5  6  7
## 6                 6  7
## 7                    7
```

Fait tourner modèle avec param constants.

```
phip.martinet <- mark(martinet.proc,
                      martinet.ddl,
                      model.parameters = list(Phi = phi, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~1)p(~1)
##
## Npar :  2
## -2lnL:  372.8533
## AICc :  376.9136
##
## Beta
##                  estimate        se       lcl      ucl
## Phi:(Intercept) 0.8524384 0.1753794 0.5086948 1.196182
## p:(Intercept)   0.8881232 0.2391869 0.4193170 1.356929
##
##
## Real Parameter Phi
## Group:colonienord
##           1         2         3         4         5         6         7
## 1 0.7010784 0.7010784 0.7010784 0.7010784 0.7010784 0.7010784 0.7010784
## 2           0.7010784 0.7010784 0.7010784 0.7010784 0.7010784 0.7010784
## 3                     0.7010784 0.7010784 0.7010784 0.7010784 0.7010784
## 4                               0.7010784 0.7010784 0.7010784 0.7010784
## 5                                         0.7010784 0.7010784 0.7010784
## 6                                                   0.7010784 0.7010784
## 7                                                             0.7010784
##
## Group:coloniesud
##           1         2         3         4         5         6         7
## 1 0.7010784 0.7010784 0.7010784 0.7010784 0.7010784 0.7010784 0.7010784
## 2           0.7010784 0.7010784 0.7010784 0.7010784 0.7010784 0.7010784
```

```
## 3                     0.7010784 0.7010784 0.7010784 0.7010784 0.7010784
## 4                               0.7010784 0.7010784 0.7010784 0.7010784
## 5                                         0.7010784 0.7010784 0.7010784
## 6                                                   0.7010784 0.7010784
## 7                                                             0.7010784
##
##
## Real Parameter p
## Group:colonienord
##           2         3         4         5         6         7         8
## 1 0.7085027 0.7085027 0.7085027 0.7085027 0.7085027 0.7085027 0.7085027
## 2           0.7085027 0.7085027 0.7085027 0.7085027 0.7085027 0.7085027
## 3                     0.7085027 0.7085027 0.7085027 0.7085027 0.7085027
## 4                               0.7085027 0.7085027 0.7085027 0.7085027
## 5                                         0.7085027 0.7085027 0.7085027
## 6                                                   0.7085027 0.7085027
## 7                                                             0.7085027
##
## Group:coloniesud
##           2         3         4         5         6         7         8
## 1 0.7085027 0.7085027 0.7085027 0.7085027 0.7085027 0.7085027 0.7085027
## 2           0.7085027 0.7085027 0.7085027 0.7085027 0.7085027 0.7085027
## 3                     0.7085027 0.7085027 0.7085027 0.7085027 0.7085027
## 4                               0.7085027 0.7085027 0.7085027 0.7085027
## 5                                         0.7085027 0.7085027 0.7085027
## 6                                                   0.7085027 0.7085027
## 7                                                             0.7085027
```

```
phip.martinet$results$real
```

```
##                      estimate        se       lcl       ucl fixed note
## Phi gnord c1 a0 t1 0.7010784 0.0367538 0.6245005 0.7678449
## p gnord c1 a1 t2   0.7085027 0.0493985 0.6033198 0.7952602
```

PIM pour CJS.

```
PIMS(phip.martinet,"Phi")
```

```
## group = colonienord
##    1  2  3  4  5  6  7
## 1  1  1  1  1  1  1  1
## 2     1  1  1  1  1  1
## 3        1  1  1  1  1
## 4           1  1  1  1
## 5              1  1  1
## 6                 1  1
## 7                    1
## group = coloniesud
##    1  2  3  4  5  6  7
## 1  1  1  1  1  1  1  1
## 2     1  1  1  1  1  1
## 3        1  1  1  1  1
```

```
## 4           1  1  1  1
## 5              1  1  1
## 6                 1  1
## 7                    1
```

Modèle avec 2 classes d'âge sur la survie.

```
# create 0, 1+ age variable
martinet.ddl <- add.design.data(martinet.proc,
                                martinet.ddl, # add 2 age-class structure to design matrix
                             "Phi",
                             type = "age",
                             bins = c(0, 1, 7),
                             name = "ageclass",
                             right = FALSE)
```

On spécifie une survie qui dépend de l'âge.

```
phi.age <- list(formula=~ageclass) # age effect on survival
```

On ajuste le modèle avec survie âge-dépendante et prob de recapture constante.

```
CJSage.martinet <- mark(martinet.proc,
                        martinet.ddl,
                        model.parameters = list(Phi = phi.age, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~ageclass)p(~1)
##
## Npar :  3
## -2lnL:  372.846
## AICc :  378.9672
##
## Beta
##                     estimate        se        lcl       ucl
## Phi:(Intercept)    0.8749553 0.3191399  0.2494411 1.5004695
## Phi:ageclass[1,7] -0.0339140 0.3988106 -0.8155829 0.7477549
## p:(Intercept)      0.8823123 0.2487229  0.3948155 1.3698091
##
##
## Real Parameter Phi
## Group:colonienord
##           1         2         3         4         5         6         7
## 1 0.7057758 0.6986845 0.6986845 0.6986845 0.6986845 0.6986845 0.6986845
## 2           0.7057758 0.6986845 0.6986845 0.6986845 0.6986845 0.6986845
## 3                     0.7057758 0.6986845 0.6986845 0.6986845 0.6986845
## 4                               0.7057758 0.6986845 0.6986845 0.6986845
## 5                                         0.7057758 0.6986845 0.6986845
## 6                                                   0.7057758 0.6986845
## 7                                                             0.7057758
##
```

```
## Group:coloniesud
##           1         2         3         4         5         6         7
## 1 0.7057758 0.6986845 0.6986845 0.6986845 0.6986845 0.6986845 0.6986845
## 2           0.7057758 0.6986845 0.6986845 0.6986845 0.6986845 0.6986845
## 3                     0.7057758 0.6986845 0.6986845 0.6986845 0.6986845
## 4                               0.7057758 0.6986845 0.6986845 0.6986845
## 5                                         0.7057758 0.6986845 0.6986845
## 6                                                   0.7057758 0.6986845
## 7                                                             0.7057758
##
##
## Real Parameter p
## Group:colonienord
##           2         3         4         5         6         7         8
## 1 0.7073012 0.7073012 0.7073012 0.7073012 0.7073012 0.7073012 0.7073012
## 2           0.7073012 0.7073012 0.7073012 0.7073012 0.7073012 0.7073012
## 3                     0.7073012 0.7073012 0.7073012 0.7073012 0.7073012
## 4                               0.7073012 0.7073012 0.7073012 0.7073012
## 5                                         0.7073012 0.7073012 0.7073012
## 6                                                   0.7073012 0.7073012
## 7                                                             0.7073012
##
## Group:coloniesud
##           2         3         4         5         6         7         8
## 1 0.7073012 0.7073012 0.7073012 0.7073012 0.7073012 0.7073012 0.7073012
## 2           0.7073012 0.7073012 0.7073012 0.7073012 0.7073012 0.7073012
## 3                     0.7073012 0.7073012 0.7073012 0.7073012 0.7073012
## 4                               0.7073012 0.7073012 0.7073012 0.7073012
## 5                                         0.7073012 0.7073012 0.7073012
## 6                                                   0.7073012 0.7073012
## 7                                                             0.7073012
```

```
CJSage.martinet$results$real
```

```
##                       estimate        se       lcl       ucl fixed note
## Phi gnord c1 a0 t1 0.7057758 0.0662714 0.5620389 0.8176445
## Phi gnord c1 a1 t2 0.6986845 0.0463273 0.6010232 0.7811452
## p gnord c1 a1 t2   0.7073012 0.0514922 0.5974414 0.7973493
```

PIM pour CJS avec âge.

```
PIMS(CJSage.martinet,"Phi")
```

```
## group = colonienord
##     1  2  3  4  5  6  7
## 1   1  2  2  2  2  2  2
## 2      1  2  2  2  2  2
## 3         1  2  2  2  2
## 4            1  2  2  2
## 5               1  2  2
## 6                  1  2
## 7                     1
```

```
## group = coloniesud
##    1  2  3  4  5  6  7
## 1  1  2  2  2  2  2  2
## 2     1  2  2  2  2  2
## 3        1  2  2  2  2
## 4           1  2  2  2
## 5              1  2  2
## 6                 1  2
## 7                    1
```

Maintenant on passe au gros modèle $phi(a.g), p(g.t)$, avec interaction âge et groupe sur la survie, et groupe et temps sur la recapture.

On définit les paramètres.

```
phi.a.g <- list(formula=~ageclass*colonie) # age and colonie effect on survival
p.g.t <- list(formula=~colonie*time) # age and colonie effect on survival
```

On ajuste le modèle.

```
gros.mod <- mark(martinet.proc,
                        martinet.ddl,
                        model.parameters = list(Phi = phi.a.g, p = p.g.t))
```

```
##
## Output summary for CJS model
## Name : Phi(~ageclass * colonie)p(~colonie * time)
##
## Npar :  18  (unadjusted=16)
## -2lnL:  340.7324
## AICc :  380.4701  (unadjusted=375.67296)
##
## Beta
##                                 estimate          se          lcl         ucl
## Phi:(Intercept)                0.1691765   0.5256389   -0.8610757   1.1994287
## Phi:ageclass[1,7]              0.4792945   0.7462021   -0.9832618   1.9418507
## Phi:coloniesud                 1.4022500   0.7054861    0.0194972   2.7850027
## Phi:ageclass[1,7]:coloniesud  -1.0377751   0.9299054   -2.8603896   0.7848395
## p:(Intercept)                 16.1573860 173.9353700 -324.7559500 357.0707200
## p:coloniesud                 -14.2379800 173.9362800 -355.1530900 326.6771300
## p:time3                      -15.2604390 173.9381700 -356.1792600 325.6583800
## p:time4                      -16.4013190 173.9372800 -357.3184000 324.5157600
## p:time5                      -17.5585480 173.9395600 -358.4800900 323.3629900
## p:time6                      -16.1106360 173.9395400 -357.0321300 324.8108600
## p:time7                        9.8192080   0.0000000    9.8192080   9.8192080
## p:time8                      -16.7777280 173.9351400 -357.6906100 324.1351500
## p:coloniesud:time3            14.3175550 173.9400400 -326.6049400 355.2400500
## p:coloniesud:time4            14.6473630 173.9387700 -326.2726300 355.5673600
## p:coloniesud:time5            16.9697980 173.9415800 -323.9557100 357.8953000
## p:coloniesud:time6            16.5374510 173.9435100 -324.3918300 357.4667300
## p:coloniesud:time7           -10.0458270   0.0000000  -10.0458270 -10.0458270
## p:coloniesud:time8            14.7427580 173.9362800 -326.1723600 355.6578800
##
```

```
##
## Real Parameter Phi
## Group:colonienord
##           1         2         3         4         5         6         7
## 1 0.5421935 0.6566658 0.6566658 0.6566658 0.6566658 0.6566658 0.6566658
## 2           0.5421935 0.6566658 0.6566658 0.6566658 0.6566658 0.6566658
## 3                     0.5421935 0.6566658 0.6566658 0.6566658 0.6566658
## 4                               0.5421935 0.6566658 0.6566658 0.6566658
## 5                                         0.5421935 0.6566658 0.6566658
## 6                                                   0.5421935 0.6566658
## 7                                                             0.5421935
##
## Group:coloniesud
##           1         2         3         4         5         6         7
## 1 0.8279869 0.7335963 0.7335963 0.7335963 0.7335963 0.7335963 0.7335963
## 2           0.8279869 0.7335963 0.7335963 0.7335963 0.7335963 0.7335963
## 3                     0.8279869 0.7335963 0.7335963 0.7335963 0.7335963
## 4                               0.8279869 0.7335963 0.7335963 0.7335963
## 5                                         0.8279869 0.7335963 0.7335963
## 6                                                   0.8279869 0.7335963
## 7                                                             0.8279869
##
##
## Real Parameter p
## Group:colonienord
##           2         3         4         5         6 7         8
## 1 0.9999999 0.7103216 0.4393173 0.1976317 0.5116852 1 0.3497037
## 2           0.7103216 0.4393173 0.1976317 0.5116852 1 0.3497037
## 3                     0.4393173 0.1976317 0.5116852 1 0.3497037
## 4                               0.1976317 0.5116852 1 0.3497037
## 5                                         0.5116852 1 0.3497037
## 6                                                   1 0.3497037
## 7                                                     0.3497037
##
## Group:coloniesud
##           2         3         4        5         6         7         8
## 1 0.8720722 0.7264176 0.5412685 0.790949 0.9126334 0.8445904 0.4711413
## 2           0.7264176 0.5412685 0.790949 0.9126334 0.8445904 0.4711413
## 3                     0.5412685 0.790949 0.9126334 0.8445904 0.4711413
## 4                               0.790949 0.9126334 0.8445904 0.4711413
## 5                                        0.9126334 0.8445904 0.4711413
## 6                                                  0.8445904 0.4711413
## 7                                                            0.4711413
```

```
gros.mod$results$real
```

```
##                      estimate           se           lcl       ucl fixed note
## Phi gnord c1 a0 t1 0.5421935 1.304739e-01  2.971146e-01 0.7684231
## Phi gnord c1 a1 t2 0.6566658 1.044162e-01  4.355429e-01 0.8258093
## Phi gsud c1 a0 t1  0.8279869 6.701750e-02  6.568198e-01 0.9236989
## Phi gsud c1 a1 t2  0.7335963 5.103750e-02  6.227153e-01 0.8212446
## p gnord c1 a1 t2   0.9999999 1.672339e-05 9.126051e-142 1.0000000
## p gnord c1 a2 t3   0.7103216 2.128972e-01  2.439779e-01 0.9490630
## p gnord c1 a3 t4   0.4393173 2.616170e-01  8.901880e-02 0.8626892
```

```
## p gnord c1 a4 t5   0.1976317 1.847874e-01  2.447860e-02 0.7074119
## p gnord c1 a5 t6   0.5116852 2.865186e-01  9.968010e-02 0.9084025
## p gnord c1 a6 t7   1.0000000 0.000000e+00  1.000000e+00 1.0000000
## p gnord c1 a7 t8   0.3497037 2.284812e-01  6.981300e-02 0.7939467
## p gsud c1 a1 t2    0.8720722 1.169566e-01  4.662105e-01 0.9815520
## p gsud c1 a2 t3    0.7264176 1.196539e-01  4.492875e-01 0.8962834
## p gsud c1 a3 t4    0.5412685 1.239405e-01  3.072712e-01 0.7583782
## p gsud c1 a4 t5    0.7909490 1.016553e-01  5.313736e-01 0.9266036
## p gsud c1 a5 t6    0.9126334 8.004400e-02  5.935335e-01 0.9867948
## p gsud c1 a6 t7    0.8445904 1.129742e-01  5.014504e-01 0.9670664
## p gsud c1 a7 t8    0.4711413 1.002333e-01  2.882252e-01 0.6621506
```

PIM pour survie et détection dans le gros modèle.

```
PIMS(gros.mod,"Phi")
```

```
## group = colonienord
##    1  2  3  4  5  6  7
## 1  1  2  2  2  2  2  2
## 2     1  2  2  2  2  2
## 3        1  2  2  2  2
## 4           1  2  2  2
## 5              1  2  2
## 6                 1  2
## 7                    1
## group = coloniesud
##    1  2  3  4  5  6  7
## 1  3  4  4  4  4  4  4
## 2     3  4  4  4  4  4
## 3        3  4  4  4  4
## 4           3  4  4  4
## 5              3  4  4
## 6                 3  4
## 7                    3
```

```
PIMS(gros.mod,"p")
```

```
## group = colonienord
##    2  3  4  5  6  7  8
## 1  5  6  7  8  9 10 11
## 2     6  7  8  9 10 11
## 3        7  8  9 10 11
## 4           8  9 10 11
## 5              9 10 11
## 6                10 11
## 7                   11
## group = coloniesud
##    2  3  4  5  6  7  8
## 1 12 13 14 15 16 17 18
## 2    13 14 15 16 17 18
## 3       14 15 16 17 18
## 4          15 16 17 18
```

```
## 5              16 17 18
## 6                 17 18
## 7                    18
```

Partie 3 : Hypothèses des modèles de capture-recapture, hétérogénéité et tests d'ajustement

Le but de cet exercice est de se familiariser avec les données de capture-recapture en population ouverte, d'ajuster par maximum de vraisemblance quelques modèles simples, de comparer ces modèles entre eux pour déterminer celui qui fournit la meilleure description des données et de tester la qualité de l'ajustement de ces modèles.

Question 1

On simule 2 jeux de données de capture-recapture avec les paramètres de survie (ϕ) et recapture (p) suivants : * jeu de données G1 : $\phi = 0.8$, $p = 0.8$; * jeu de données G2 : $\phi = 0.8$, $p = 0.2$.

```
simul <- function(nind, nocc, phi, p){
   dat <- matrix(0, nrow = nind, ncol = nocc)
   dat[1:nind, 1] <- 1 # a single cohort
   for (i in 1:nind){
      # processus survie
      for (j in 2:nocc){
         alive.or.dead <- rbinom(1, 1, phi)
         # conditional on being alive at t, alive or dead at t+1
         dat[i, j] <- ifelse(dat[i, j - 1] == 0, 0, alive.or.dead)
      }
      # processus detection
      for (j in 2:nocc){
         detected.or.not <- rbinom(1, 1, p)
         # conditional on being alive at t, detected or not at t
         dat[i, j] <- ifelse(dat[i, j] == 0, 0, detected.or.not)
      }
   }
data.frame(y = dat)
}
```

```
set.seed(2021)
nind <- 500
nocc <- 8
G1 <- simul(nind = nind, nocc = nocc, phi = 0.8, p = 0.8)
G2 <- simul(nind = nind, nocc = nocc, phi = 0.8, p = 0.2)
```

Ajuster séparément à G1 et G2 le modèle $\Phi(t), p(t)$ appelé aussi le modèle de Cormack-Jolly-Seber (CJS). Que pouvez-vous vous dire sur l'estimation des paramètres ?

```
G1marked <- data.frame(ch = tidyr::unite(G1, col = "ch", sep = ""),
                       n = rep(1, nrow(G1)))
G2marked <- data.frame(ch = tidyr::unite(G2, col = "ch", sep = ""),
                       n = rep(1, nrow(G2)))
```

On prépare les données.

```
G1.proc <- process.data(G1marked)
G2.proc <- process.data(G2marked)
G1.ddl <- make.design.data(G1.proc)
G2.ddl <- make.design.data(G2.proc)
```

On spécifie les paramètres.

```
phi <- list(formula=~1)
p <- list(formula=~1)
```

On ajuste le modèle avec paramètres constants aux données G1.

```
cjs.G1 <- mark(G1.proc,
              G1.ddl,
              model.parameters = list(Phi = phi, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~1)p(~1)
##
## Npar :  2
## -2lnL:  3009.594
## AICc :  3013.601
##
## Beta
##                  estimate        se      lcl      ucl
## Phi:(Intercept) 1.349691 0.0584381 1.235152 1.464229
## p:(Intercept)   1.417211 0.0761598 1.267938 1.566484
##
##
## Real Parameter Phi
##
##          1         2         3         4         5         6         7
## 1 0.7940791 0.7940791 0.7940791 0.7940791 0.7940791 0.7940791 0.7940791
## 2           0.7940791 0.7940791 0.7940791 0.7940791 0.7940791 0.7940791
## 3                     0.7940791 0.7940791 0.7940791 0.7940791 0.7940791
## 4                               0.7940791 0.7940791 0.7940791 0.7940791
## 5                                         0.7940791 0.7940791 0.7940791
## 6                                                   0.7940791 0.7940791
## 7                                                             0.7940791
##
##
## Real Parameter p
##
##          2         3         4         5         6         7         8
## 1 0.8049008 0.8049008 0.8049008 0.8049008 0.8049008 0.8049008 0.8049008
## 2           0.8049008 0.8049008 0.8049008 0.8049008 0.8049008 0.8049008
## 3                     0.8049008 0.8049008 0.8049008 0.8049008 0.8049008
## 4                               0.8049008 0.8049008 0.8049008 0.8049008
## 5                                         0.8049008 0.8049008 0.8049008
## 6                                                   0.8049008 0.8049008
## 7                                                             0.8049008
```

```r
cjs.G1$results$real
```

```
##                    estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.7940791 0.0095556 0.7747190 0.8121787
## p g1 c1 a1 t2   0.8049008 0.0119598 0.7803896 0.8272818
```

Puis aux données G2.

```r
cjs.G2 <- mark(G2.proc,
               G2.ddl,
               model.parameters = list(Phi = phi, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~1)p(~1)
##
## Npar :  2
## -2lnL:  2091.359
## AICc :  2095.374
##
## Beta
##                    estimate        se       lcl       ucl
## Phi:(Intercept)  1.487792 0.1111557  1.269926  1.705657
## p:(Intercept)   -1.398919 0.0940821 -1.583320 -1.214518
##
##
## Real Parameter Phi
##
##           1         2         3         4         5         6         7
## 1 0.8157466 0.8157466 0.8157466 0.8157466 0.8157466 0.8157466 0.8157466
## 2           0.8157466 0.8157466 0.8157466 0.8157466 0.8157466 0.8157466
## 3                     0.8157466 0.8157466 0.8157466 0.8157466 0.8157466
## 4                               0.8157466 0.8157466 0.8157466 0.8157466
## 5                                         0.8157466 0.8157466 0.8157466
## 6                                                   0.8157466 0.8157466
## 7                                                             0.8157466
##
##
## Real Parameter p
##
##           2         3         4         5         6         7         8
## 1 0.1979877 0.1979877 0.1979877 0.1979877 0.1979877 0.1979877 0.1979877
## 2           0.1979877 0.1979877 0.1979877 0.1979877 0.1979877 0.1979877
## 3                     0.1979877 0.1979877 0.1979877 0.1979877 0.1979877
## 4                               0.1979877 0.1979877 0.1979877 0.1979877
## 5                                         0.1979877 0.1979877 0.1979877
## 6                                                   0.1979877 0.1979877
## 7                                                             0.1979877
```

```r
cjs.G2$results$real
```

```
##                    estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.8157466 0.0167072 0.7807302 0.8462721
## p g1 c1 a1 t2   0.1979877 0.0149392 0.1703258 0.2289026
```

Question 2

a) Grouper les jeux de données G1 et G2 pour obtenir le jeu de données G1+G2.

```
G1plusG2 <- rbind(G1, G2)
```

b) Ajuster le modèle CJS à G1+G2. Que remarquez-vous concernant l'estimation des paramètres ?

```
G1G2marked <- data.frame(ch = tidyr::unite(G1plusG2, col = "ch", sep = ""),
                         n = rep(1, nrow(G1plusG2)))
G1G2.proc <- process.data(G1G2marked)
G1G2.ddl <- make.design.data(G1G2.proc)
cjs.G1G2 <- mark(G1G2.proc,
                G1G2.ddl,
                model.parameters = list(Phi = phi, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~1)p(~1)
##
## Npar :  2
## -2lnL:  5825.357
## AICc :  5829.362
##
## Beta
##                  estimate        se       lcl       ucl
## Phi:(Intercept) 1.1639804 0.0436060 1.0785126 1.2494483
## p:(Intercept)   0.3450608 0.0495103 0.2480206 0.4421009
##
##
## Real Parameter Phi
##
##           1         2         3         4         5         6         7
## 1 0.7620552 0.7620552 0.7620552 0.7620552 0.7620552 0.7620552 0.7620552
## 2           0.7620552 0.7620552 0.7620552 0.7620552 0.7620552 0.7620552
## 3                     0.7620552 0.7620552 0.7620552 0.7620552 0.7620552
## 4                               0.7620552 0.7620552 0.7620552 0.7620552
## 5                                         0.7620552 0.7620552 0.7620552
## 6                                                   0.7620552 0.7620552
## 7                                                             0.7620552
##
##
## Real Parameter p
##
##           2         3         4         5         6         7         8
## 1 0.5854193 0.5854193 0.5854193 0.5854193 0.5854193 0.5854193 0.5854193
## 2           0.5854193 0.5854193 0.5854193 0.5854193 0.5854193 0.5854193
## 3                     0.5854193 0.5854193 0.5854193 0.5854193 0.5854193
## 4                               0.5854193 0.5854193 0.5854193 0.5854193
## 5                                         0.5854193 0.5854193 0.5854193
## 6                                                   0.5854193 0.5854193
## 7                                                             0.5854193
```

```r
cjs.G1G2$results$real
```

```
##                   estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.7620552 0.0079070 0.7462124 0.7772043
## p g1 c1 a1 t2   0.5854193 0.0120163 0.5616892 0.6087595
```

Modèle avec survie qui dépend du temps.

```r
phi.time <- list(formula=~time)
cjs.G1G2 <- mark(G1G2.proc,
                G1G2.ddl,
                model.parameters = list(Phi = phi.time, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~time)p(~1)
##
## Npar :  8
## -2lnL:  5792.723
## AICc :  5808.782
##
## Beta
##                   estimate        se        lcl       ucl
## Phi:(Intercept) 0.7598159 0.0909050  0.5816422 0.9379897
## Phi:time2       0.3979968 0.1933473  0.0190360 0.7769575
## Phi:time3       0.7072561 0.2137484  0.2883091 1.1262030
## Phi:time4       0.6334195 0.2291568  0.1842720 1.0825669
## Phi:time5       0.4558441 0.2336741 -0.0021572 0.9138454
## Phi:time6       0.8182710 0.3428341  0.1463162 1.4902257
## Phi:time7       1.0221780 0.5472995 -0.0505291 2.0948851
## p:(Intercept)   0.3870597 0.0505148  0.2880506 0.4860687
##
##
## Real Parameter Phi
##
##           1         2         3         4         5         6         7
## 1 0.6813138 0.7609351 0.8126119 0.8011083 0.7712989 0.8289334 0.8559429
## 2           0.7609351 0.8126119 0.8011083 0.7712989 0.8289334 0.8559429
## 3                     0.8126119 0.8011083 0.7712989 0.8289334 0.8559429
## 4                               0.8011083 0.7712989 0.8289334 0.8559429
## 5                                         0.7712989 0.8289334 0.8559429
## 6                                                   0.8289334 0.8559429
## 7                                                             0.8559429
##
##
## Real Parameter p
##
##           2         3         4         5         6         7         8
## 1 0.5955747 0.5955747 0.5955747 0.5955747 0.5955747 0.5955747 0.5955747
## 2           0.5955747 0.5955747 0.5955747 0.5955747 0.5955747 0.5955747
## 3                     0.5955747 0.5955747 0.5955747 0.5955747 0.5955747
## 4                               0.5955747 0.5955747 0.5955747 0.5955747
```

```
## 5                                         0.5955747 0.5955747 0.5955747
## 6                                                   0.5955747 0.5955747
## 7                                                             0.5955747
```

```r
cjs.G1G2$results$real
```

```
##                   estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.6813138 0.0197378 0.6414452 0.7186934
## Phi g1 c1 a1 t2 0.7609351 0.0259835 0.7063779 0.8081089
## Phi g1 c1 a2 t3 0.8126119 0.0294917 0.7479047 0.8637363
## Phi g1 c1 a3 t4 0.8011083 0.0335598 0.7271893 0.8588853
## Phi g1 c1 a4 t5 0.7712989 0.0379756 0.6886255 0.8372107
## Phi g1 c1 a5 t6 0.8289334 0.0470410 0.7166461 0.9027612
## Phi g1 c1 a6 t7 0.8559429 0.0668933 0.6723174 0.9450754
## p g1 c1 a1 t2   0.5955747 0.0121673 0.5715188 0.6191799
```

Question 3

A l'aide du package `R2ucare`, tester la qualité de l'ajustement du modèle CJS aux données G1, G2 et G1+G2. Quelles sont vos conclusions ?

G1

```r
overall_CJS(G1, rep(1,nrow(G1)))
```

```
##                             chi2 degree_of_freedom p_value
## Gof test for CJS model: 3.327                 9    0.95
```

G2

```r
overall_CJS(G2, rep(1,nrow(G2)))
```

```
##                              chi2 degree_of_freedom p_value
## Gof test for CJS model: 15.041                14   0.375
```

G1G2

```r
overall_CJS(G1plusG2, rep(1,nrow(G1plusG2)))
```

```
##                               chi2 degree_of_freedom p_value
## Gof test for CJS model: 150.342                15       0
```

Question 4

Il peut y avoir des animaux en transit sur la zone d'étude.

a) Pour créer artificiellement une telle situation, rajouter 50 individus en transit (i.e. possédant une histoire avec un seul événement de capture) à chaque date dans G1.

```
G1transit <- as.matrix(G1)
ntransients <- 50
for (j in 1:nocc){
   zeros <- matrix(0, nrow = ntransients, ncol = nocc)
   zeros[, j] <- 1
   G1transit <- rbind(G1transit, zeros)
}
G1transit <- data.frame(y = G1transit)
```

```
dim(G1transit)
```

```
## [1] 900   8
```

```
head(G1transit)
```

```
##   y.y.1 y.y.2 y.y.3 y.y.4 y.y.5 y.y.6 y.y.7 y.y.8
## 1     1     1     0     0     1     0     1     1
## 2     1     0     0     0     0     0     0     0
## 3     1     0     0     0     0     0     0     0
## 4     1     0     0     0     0     0     0     0
## 5     1     1     0     0     0     0     0     0
## 6     1     1     1     0     0     0     0     0
```

```
tail(G1transit)
```

```
##     y.y.1 y.y.2 y.y.3 y.y.4 y.y.5 y.y.6 y.y.7 y.y.8
## 895     0     0     0     0     0     0     0     1
## 896     0     0     0     0     0     0     0     1
## 897     0     0     0     0     0     0     0     1
## 898     0     0     0     0     0     0     0     1
## 899     0     0     0     0     0     0     0     1
## 900     0     0     0     0     0     0     0     1
```

b) Faire tourner le modèle CJS à ces nouvelles données avec `RMark`. Quelles sont vos conclusions concernant les estimations ?

```
G1transitmarked <- data.frame(ch = tidyr::unite(G1transit, col = "ch", sep = ""),
                              n = rep(1, nrow(G1transit)))
```

```
G1transit.proc <- process.data(G1transitmarked)
G1transit.ddl <- make.design.data(G1transit.proc)
```

Ajuste le modèle.

```
cjs.G1transit <- mark(G1transit.proc,
                      G1transit.ddl,
                      model.parameters = list(Phi = phi, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~1)p(~1)
##
## Npar :  2
## -2lnL:  3793.282
## AICc :  3797.288
##
## Beta
##                  estimate        se       lcl       ucl
## Phi:(Intercept) 0.7871844 0.0479482 0.6932058 0.8811629
## p:(Intercept)   1.1885539 0.0770665 1.0375036 1.3396042
##
##
## Real Parameter Phi
##
##          1         2         3         4         5         6         7
## 1 0.6872264 0.6872264 0.6872264 0.6872264 0.6872264 0.6872264 0.6872264
## 2           0.6872264 0.6872264 0.6872264 0.6872264 0.6872264 0.6872264
## 3                     0.6872264 0.6872264 0.6872264 0.6872264 0.6872264
## 4                               0.6872264 0.6872264 0.6872264 0.6872264
## 5                                         0.6872264 0.6872264 0.6872264
## 6                                                   0.6872264 0.6872264
## 7                                                             0.6872264
##
##
## Real Parameter p
##
##          2         3         4         5         6         7         8
## 1 0.7664823 0.7664823 0.7664823 0.7664823 0.7664823 0.7664823 0.7664823
## 2           0.7664823 0.7664823 0.7664823 0.7664823 0.7664823 0.7664823
## 3                     0.7664823 0.7664823 0.7664823 0.7664823 0.7664823
## 4                               0.7664823 0.7664823 0.7664823 0.7664823
## 5                                         0.7664823 0.7664823 0.7664823
## 6                                                   0.7664823 0.7664823
## 7                                                             0.7664823
```

```
cjs.G1transit$results$real
```

```
##                  estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.6872264 0.0103063 0.6666797 0.7070632
## p g1 c1 a1 t2   0.7664823 0.0137939 0.7383680 0.7924248
```

Idem avec survie qui dépend du temps.

```
cjs.G1transit <- mark(G1transit.proc,
                      G1transit.ddl,
                      model.parameters = list(Phi = phi.time, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~time)p(~1)
```

```
##
## Npar :  8
## -2lnL:  3776.066
## AICc :  3792.138
##
## Beta
##                   estimate        se        lcl        ucl
## Phi:(Intercept)  1.1097474 0.1223381  0.8699648  1.3495301
## Phi:time2       -0.3581909 0.1908886 -0.7323325  0.0159508
## Phi:time3       -0.2686696 0.1890295 -0.6391674  0.1018283
## Phi:time4       -0.4461156 0.1934614 -0.8253000 -0.0669312
## Phi:time5       -0.4810602 0.2040143 -0.8809281 -0.0811922
## Phi:time6       -0.3850171 0.2250948 -0.8262029  0.0561686
## Phi:time7       -0.8490591 0.2307664 -1.3013613 -0.3967569
## p:(Intercept)    1.1866979 0.0786387  1.0325661  1.3408297
##
##
## Real Parameter Phi
##
##          1         2         3         4         5         6         7
## 1 0.752082 0.6795178 0.6986922 0.6600758 0.6521917 0.6736478 0.5648055
## 2          0.6795178 0.6986922 0.6600758 0.6521917 0.6736478 0.5648055
## 3                    0.6986922 0.6600758 0.6521917 0.6736478 0.5648055
## 4                              0.6600758 0.6521917 0.6736478 0.5648055
## 5                                        0.6521917 0.6736478 0.5648055
## 6                                                  0.6736478 0.5648055
## 7                                                            0.5648055
##
##
## Real Parameter p
##
##         2       3       4       5       6       7       8
## 1 0.76615 0.76615 0.76615 0.76615 0.76615 0.76615 0.76615
## 2         0.76615 0.76615 0.76615 0.76615 0.76615 0.76615
## 3                 0.76615 0.76615 0.76615 0.76615 0.76615
## 4                         0.76615 0.76615 0.76615 0.76615
## 5                                 0.76615 0.76615 0.76615
## 6                                         0.76615 0.76615
## 7                                                 0.76615
```

```
cjs.G1transit$results$real
```

```
##                   estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.7520820 0.0228105 0.7047384 0.7940528
## Phi g1 c1 a1 t2 0.6795178 0.0270283 0.6244072 0.7300381
## Phi g1 c1 a2 t3 0.6986922 0.0305397 0.6356995 0.7549906
## Phi g1 c1 a3 t4 0.6600758 0.0337817 0.5911055 0.7228667
## Phi g1 c1 a4 t5 0.6521917 0.0371981 0.5762202 0.7211350
## Phi g1 c1 a5 t6 0.6736478 0.0419754 0.5867403 0.7500642
## Phi g1 c1 a6 t7 0.5648055 0.0489543 0.4676275 0.6572466
## p g1 c1 a1 t2   0.7661500 0.0140892 0.7374131 0.7926264
```

c) Tester l'ajustement du modèle CJS à ces mêmes données avec `R2ucare`. Interpréter en particulier la composante 3.SR du test.

```r
overall_CJS(G1transit, rep(1,nrow(G1transit)))
```

```
##                              chi2 degree_of_freedom p_value
## Gof test for CJS model: 543.606                15       0
```

```r
test2ct(G1transit, rep(1,nrow(G1transit)))
```

```
## $test2ct
##      stat        df     p_val sign_test
##     1.135     5.000     0.951     0.600
##
## $details
##   component dof  stat p_val signed_test  test_perf
## 1         2   1 0.112 0.737      -0.335 Chi-square
## 2         3   1 0.003 0.953       0.055 Chi-square
## 3         4   1 0.721 0.396       0.849 Chi-square
## 4         5   1 0.139 0.709       0.373 Chi-square
## 5         6   1  0.16  0.69         0.4     Fisher
```

```r
test3sr(G1transit, rep(1,nrow(G1transit)))
```

```
## $test3sr
##      stat        df     p_val sign_test
##   540.279     6.000     0.000    23.140
##
## $details
##   component    stat p_val signed_test  test_perf
## 1         2  96.827     0        9.84 Chi-square
## 2         3 103.329     0      10.165 Chi-square
## 3         4  88.333     0       9.399 Chi-square
## 4         5   94.62     0       9.727 Chi-square
## 5         6 100.743     0      10.037 Chi-square
## 6         7  56.427     0       7.512 Chi-square
```

d) Faire tourner un modèle à 2 classes d'âge sur la survie $\phi(a2 * t)$ avec `RMark`. Vos conclusions ?

```r
G1transit.ddl <- make.design.data(G1transit.proc)
# create 0, 1+ age variable
G1transit.ddl <- add.design.data(G1transit.proc,
                             G1transit.ddl, # add 2 age-class structure to design matrix
                             "Phi",
                             type = "age",
                             bins = c(0, 1, nocc - 1),
                             name = "ageclass",
                             right = FALSE)
```

On spécifie une survie qui dépend de l'âge.

```r
phi.age <- list(formula=~ageclass) # age effect on survival
```

On ajuste le modèle.

```r
cjsage.G1transit <- mark(G1transit.proc,
                      G1transit.ddl,
                      model.parameters = list(Phi = phi.age, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~ageclass)p(~1)
##
## Npar :  3
## -2lnL:  3604.772
## AICc :  3610.784
##
## Beta
##                      estimate        se        lcl       ucl
## Phi:(Intercept)    -0.1000538 0.0738121 -0.2447255 0.0446178
## Phi:ageclass[1,7]   1.4656703 0.1046905  1.2604769 1.6708636
## p:(Intercept)       1.3783584 0.0769186  1.2275979 1.5291189
##
##
## Real Parameter Phi
##
##           1         2         3         4         5         6         7
## 1 0.4750074 0.7966710 0.7966710 0.7966710 0.7966710 0.7966710 0.7966710
## 2           0.4750074 0.7966710 0.7966710 0.7966710 0.7966710 0.7966710
## 3                     0.4750074 0.7966710 0.7966710 0.7966710 0.7966710
## 4                               0.4750074 0.7966710 0.7966710 0.7966710
## 5                                         0.4750074 0.7966710 0.7966710
## 6                                                   0.4750074 0.7966710
## 7                                                             0.4750074
##
##
## Real Parameter p
##
##           2         3         4         5         6         7         8
## 1 0.7987272 0.7987272 0.7987272 0.7987272 0.7987272 0.7987272 0.7987272
## 2           0.7987272 0.7987272 0.7987272 0.7987272 0.7987272 0.7987272
## 3                     0.7987272 0.7987272 0.7987272 0.7987272 0.7987272
## 4                               0.7987272 0.7987272 0.7987272 0.7987272
## 5                                         0.7987272 0.7987272 0.7987272
## 6                                                   0.7987272 0.7987272
## 7                                                             0.7987272
```

```r
cjsage.G1transit$results$real
```

```
##                   estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.4750074 0.0184069 0.4391222 0.5111526
## Phi g1 c1 a1 t2 0.7966710 0.0113847 0.7734445 0.8180764
## p g1 c1 a1 t2   0.7987272 0.0123656 0.7733979 0.8218774
```

D'une autre façon.

```r
G1transit.ddl <- make.design.data(G1transit.proc)
#max age 4
G1transit.ddl$Phi$max.age <- as.factor((G1transit.ddl$Phi$Age < 1) * G1transit.ddl$Phi$Age + (G1transit
phi.max.age <- list(formula=~max.age)
cjsaget.G1transit <- mark(G1transit.proc,
                     G1transit.ddl,
                     model.parameters = list(Phi = phi.max.age, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~max.age)p(~1)
##
## Npar :  3
## -2lnL:  3604.772
## AICc :  3610.784
##
## Beta
##                    estimate        se        lcl      ucl
## Phi:(Intercept) -0.1000537 0.0738121 -0.2447253 0.044618
## Phi:max.age1     1.4656701 0.1046905  1.2604767 1.670863
## p:(Intercept)    1.3783584 0.0769186  1.2275979 1.529119
##
##
## Real Parameter Phi
##
##           1         2         3         4         5         6         7
## 1 0.4750074 0.7966710 0.7966710 0.7966710 0.7966710 0.7966710 0.7966710
## 2           0.4750074 0.7966710 0.7966710 0.7966710 0.7966710 0.7966710
## 3                     0.4750074 0.7966710 0.7966710 0.7966710 0.7966710
## 4                               0.4750074 0.7966710 0.7966710 0.7966710
## 5                                         0.4750074 0.7966710 0.7966710
## 6                                                   0.4750074 0.7966710
## 7                                                             0.4750074
##
##
## Real Parameter p
##
##           2         3         4         5         6         7         8
## 1 0.7987272 0.7987272 0.7987272 0.7987272 0.7987272 0.7987272 0.7987272
## 2           0.7987272 0.7987272 0.7987272 0.7987272 0.7987272 0.7987272
## 3                     0.7987272 0.7987272 0.7987272 0.7987272 0.7987272
## 4                               0.7987272 0.7987272 0.7987272 0.7987272
## 5                                         0.7987272 0.7987272 0.7987272
## 6                                                   0.7987272 0.7987272
## 7                                                             0.7987272
```

```r
PIMS(cjsaget.G1transit,"Phi")
```

```
## group = Group 1
##    1  2  3  4  5  6  7
## 1  1  2  2  2  2  2  2
## 2     1  2  2  2  2  2
```

```
## 3         1  2  2  2  2
## 4            1  2  2  2
## 5               1  2  2
## 6                  1  2
## 7                     1
```

```
cjsaget.G1transit$results$real
```

```
##                   estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.4750074 0.0184069 0.4391222 0.5111526
## Phi g1 c1 a1 t2 0.7966710 0.0113847 0.7734445 0.8180764
## p g1 c1 a1 t2   0.7987272 0.0123656 0.7733979 0.8218774
```

Supprime fichiers créés en cours de route.

```
cleanup(ask = FALSE)
```