TP 4 analyse de survie avec données sur animaux marqués

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

Premiers modèles avec des effets temps

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

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

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)  0.9354584 0.7685263 -0.5708531  2.4417700
## Phi:time2       -1.1982775 0.8706742 -2.9047990  0.5082439
## Phi:time3       -1.0228320 0.8049184 -2.6004720  0.5548080
## Phi:time4       -0.4198614 0.8091524 -2.0058002  1.1660773
## Phi:time5       -0.5361005 0.8031476 -2.1102698  1.0380688
## Phi:time6        0.2481474 0.0000000  0.2481474  0.2481474
## p:(Intercept)    0.8292820 0.7837387 -0.7068458  2.3654098
## p:time3          1.6556247 1.2913816 -0.8754833  4.1867328
## p:time4          1.5220927 1.0729176 -0.5808259  3.6250113
## p:time5          1.3767419 0.9884841 -0.5606870  3.3141709
## p:time6          1.7950910 1.0688809 -0.2999156  3.8900975
## p:time7         -0.0147652 0.0000000 -0.0147652 -0.0147652
## 
## 
## Real Parameter Phi
## 
##            1         2         3         4         5         6
```

```
## 1 0.7181814 0.4346708 0.4781705 0.6261176 0.5985334 0.7655955
## 2           0.4346708 0.4781705 0.6261176 0.5985334 0.7655955
## 3                     0.4781705 0.6261176 0.5985334 0.7655955
## 4                               0.6261176 0.5985334 0.7655955
## 5                                         0.5985334 0.7655955
## 6                                                   0.7655955
##
##
## Real Parameter p
##
##           2         3         4         5         6         7
## 1 0.6962031 0.9230769 0.9130434 0.9007892 0.9324138 0.6930712
## 2           0.9230769 0.9130434 0.9007892 0.9324138 0.6930712
## 3                     0.9130434 0.9007892 0.9324138 0.6930712
## 4                               0.9007892 0.9324138 0.6930712
## 5                                         0.9324138 0.6930712
## 6                                                   0.6930712
```

Inspectons les résultats.

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

```
##                  estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.7181814 0.1555473 0.3610400 0.9199575
## Phi g1 c1 a1 t2 0.4346708 0.0688290 0.3075047 0.5710589
## Phi g1 c1 a2 t3 0.4781705 0.0597091 0.3643839 0.5942686
## Phi g1 c1 a3 t4 0.6261176 0.0592656 0.5048460 0.7333741
## Phi g1 c1 a4 t5 0.5985334 0.0560517 0.4855434 0.7019412
## Phi g1 c1 a5 t6 0.7655955 0.0000000 0.7655955 0.7655955
## p g1 c1 a1 t2   0.6962031 0.1657641 0.3302962 0.9141513
## p g1 c1 a2 t3   0.9230769 0.0728778 0.6161495 0.9889758
## p g1 c1 a3 t4   0.9130434 0.0581758 0.7140647 0.9778505
## p g1 c1 a4 t5   0.9007892 0.0538330 0.7360175 0.9672855
## p g1 c1 a5 t6   0.9324138 0.0458026 0.7684925 0.9828579
## p g1 c1 a6 t7   0.6930712 0.0000000 0.6930712 0.6930712
```

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 0.0422035 0.4420933
## p:(Intercept)   2.2262659 0.3251093 1.5890517 2.8634802
##
##
## 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 considère deux autres modèles, avec l'effet temps sur la survie mais pas sur la détection, et inversément.

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

```
##
## Output summary for CJS model
## Name : Phi(~time)p(~1)
##
## Npar :  7
## -2lnL:  659.7301
## AICc :  673.998
##
## Beta
##                    estimate        se        lcl       ucl
## Phi:(Intercept)   0.5143913 0.4767826 -0.4201026 1.4488851
## Phi:time2        -0.6981412 0.5537219 -1.7834360 0.3871537
## Phi:time3        -0.6009364 0.5301018 -1.6399360 0.4380632
## Phi:time4        -0.0061065 0.5334633 -1.0516946 1.0394817
## Phi:time5        -0.0757120 0.5276525 -1.1099110 0.9584870
## Phi:time6        -0.1780637 0.5265673 -1.2101355 0.8540082
## p:(Intercept)     2.2203957 0.3288850  1.5757810 2.8650104
##
##
## Real Parameter Phi
##
##           1         2         3         4         5         6
## 1 0.6258353 0.4541913 0.4783772 0.6244043 0.6079443 0.5832982
## 2           0.4541913 0.4783772 0.6244043 0.6079443 0.5832982
## 3                     0.4783772 0.6244043 0.6079443 0.5832982
## 4                               0.6244043 0.6079443 0.5832982
```

```
## 5                                         0.6079443 0.5832982
## 6                                                   0.5832982
##
##
## Real Parameter p
##
##           2         3         4         5         6         7
## 1 0.9020662 0.9020662 0.9020662 0.9020662 0.9020662 0.9020662
## 2           0.9020662 0.9020662 0.9020662 0.9020662 0.9020662
## 3                     0.9020662 0.9020662 0.9020662 0.9020662
## 4                               0.9020662 0.9020662 0.9020662
## 5                                         0.9020662 0.9020662
## 6                                                   0.9020662
```

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

```
##
## Output summary for CJS model
## Name : Phi(~1)p(~time)
##
## Npar :  7
## -2lnL:  664.4802
## AICc :  678.7481
##
## Beta
##                   estimate        se        lcl       ucl
## Phi:(Intercept) 0.2131641 0.1121136 -0.0065785 0.4329067
## p:(Intercept)   1.2955246 0.7437234 -0.1621732 2.7532225
## p:time3         0.8005298 1.1635483 -1.4800249 3.0810845
## p:time4         0.6512771 1.0018571 -1.3123628 2.6149171
## p:time5         0.9977282 0.9454476 -0.8553491 2.8508055
## p:time6         1.4658876 1.0303996 -0.5536955 3.4854708
## p:time7         1.9900717 3.0642128 -4.0157855 7.9959289
##
##
## Real Parameter Phi
##
##           1         2         3         4         5         6
## 1 0.5530901 0.5530901 0.5530901 0.5530901 0.5530901 0.5530901
## 2           0.5530901 0.5530901 0.5530901 0.5530901 0.5530901
## 3                     0.5530901 0.5530901 0.5530901 0.5530901
## 4                               0.5530901 0.5530901 0.5530901
## 5                                         0.5530901 0.5530901
## 6                                                   0.5530901
##
##
## Real Parameter p
##
##           2         3         4         5         6         7
## 1 0.7850808 0.8905191 0.8750975 0.9083167 0.9405546 0.9639314
## 2           0.8905191 0.8750975 0.9083167 0.9405546 0.9639314
## 3                     0.8750975 0.9083167 0.9405546 0.9639314
```

```
## 4                               0.9083167 0.9405546 0.9639314
## 5                                         0.9405546 0.9639314
## 6                                                   0.9639314
```

On affiche la sélection de modèles.

```r
collect.models()
```

```
##                model npar     AICc DeltaAICc      weight Deviance
## 2       Phi(~1)p(~1)    2 670.8660  0.000000 0.811204642 58.15788
## 4    Phi(~time)p(~1)    7 673.9980  3.132004 0.169443317 51.05031
## 3    Phi(~1)p(~time)    7 678.7481  7.882084 0.015760053 55.80039
## 1 Phi(~time)p(~time)   12 681.7057 10.839629 0.003591988 48.27043
```

On a 7 occasions de capture, donc 6 paramètres de survie. La première année de capture dans le jeu de données est 1981, alors on peut estimer la survie entre 1981 et 1982, entre 1982 et 1983 etc. Une inondation eut lieu en 1982 et 1983, avec un effet possible sur la survie. On va comparer les modèles précédents à un modèle incorporant un effet inondation sur les survies sur les intervalles (1982-1983) et (1983-1984).

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

```r
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 ajoute un effet inondation sur la survie.

```r
cincle.ddl$Phi$Inondation <- 0
cincle.ddl$Phi$Inondation[cincle.ddl$Phi$time==2 | cincle.ddl$Phi$time==3] <- 1
```

On définit l'effet en question, et on fait tourner le modèle correspondant.

```
phiinondation <- list(formula=~Inondation)
phiinondationp.cincle <- mark(cincle.proc,
                              cincle.ddl,
                              model.parameters = list(Phi = phiinondation,
                                                      p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~Inondation)p(~1)
##
## Npar :  3
## -2lnL:  660.1028
## AICc :  666.1597
##
## Beta
##                    estimate        se        lcl        ucl
## Phi:(Intercept)   0.4351207 0.1297482  0.1808142  0.6894272
## Phi:Inondation   -0.5599740 0.2163758 -0.9840706 -0.1358774
## p:(Intercept)     2.1948811 0.3246088  1.5586478  2.8311145
##
##
## Real Parameter Phi
##
##           1         2         3         4         5         6
## 1 0.6070958 0.4688272 0.4688272 0.6070958 0.6070958 0.6070958
## 2           0.4688272 0.4688272 0.6070958 0.6070958 0.6070958
## 3                     0.4688272 0.6070958 0.6070958 0.6070958
## 4                               0.6070958 0.6070958 0.6070958
## 5                                         0.6070958 0.6070958
## 6                                                   0.6070958
##
##
## Real Parameter p
##
##           2         3         4         5         6         7
## 1 0.8997889 0.8997889 0.8997889 0.8997889 0.8997889 0.8997889
## 2           0.8997889 0.8997889 0.8997889 0.8997889 0.8997889
## 3                     0.8997889 0.8997889 0.8997889 0.8997889
## 4                               0.8997889 0.8997889 0.8997889
## 5                                         0.8997889 0.8997889
## 6                                                   0.8997889
```

On compare les modèles avec l'AIC.

```
collect.models()
```

```
##                    model npar     AICc DeltaAICc       weight Deviance
## 2 Phi(~Inondation)p(~1)    3 666.1596  0.000000 0.8951028662 51.42300
## 3          Phi(~1)p(~1)    2 670.8660  4.706387 0.0850930418 58.15788
## 5       Phi(~time)p(~1)    7 673.9980  7.838391 0.0177741183 51.05031
## 4       Phi(~1)p(~time)    7 678.7481 12.588471 0.0016531844 55.80039
## 1    Phi(~time)p(~time)   12 681.7057 15.546016 0.0003767892 48.27043
```

Pour aller plus loin avec les effets de l'environnement

On ajoute les covariables environnementales.

```
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.

```
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
```

Pour rappel, 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
##    Inondation
## 1           0
## 2           1
## 3           1
## 4           0
## 5           0
## 6           0
## 7           1
## 8           1
## 9           0
## 10          0
## 11          0
## 12          1
## 13          0
## 14          0
## 15          0
## 16          0
## 17          0
## 18          0
## 19          0
## 20          0
## 21          0
```

On crée une survie qui dépend de la température.

```
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"])
}
```

On vérifie que ça a marché.

```
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
##    Inondation temp
## 1           0 -2.3
## 2           1 -0.4
## 3           1 -1.2
## 4           0 -4.2
## 5           0 -3.0
## 6           0 -2.8
## 7           1 -0.4
## 8           1 -1.2
## 9           0 -4.2
## 10          0 -3.0
## 11          0 -2.8
## 12          1 -1.2
## 13          0 -4.2
## 14          0 -3.0
## 15          0 -2.8
## 16          0 -4.2
## 17          0 -3.0
## 18          0 -2.8
## 19          0 -3.0
## 20          0 -2.8
## 21          0 -2.8
```

On définit les effets.

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

On ajuste le modèle.

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

```
##
## Output summary for CJS model
```

```
## Name : Phi(~temp)p(~1)
##
## Npar :  3
## -2lnL:  660.5328
## AICc :  666.5896
##
## Beta
##                    estimate        se        lcl        ucl
## Phi:(Intercept) -0.2565831 0.2220600 -0.6918208  0.1786546
## Phi:temp        -0.2051771 0.0821952 -0.3662796 -0.0440745
## p:(Intercept)    2.2347028 0.3250019  1.5976991  2.8717064
##
##
## Real Parameter Phi
##
##          1         2         3         4         5         6
## 1 0.553624 0.4564823 0.4974074 0.6468361 0.5887858 0.5788155
## 2          0.4564823 0.4974074 0.6468361 0.5887858 0.5788155
## 3                    0.4974074 0.6468361 0.5887858 0.5788155
## 4                              0.6468361 0.5887858 0.5788155
## 5                                        0.5887858 0.5788155
## 6                                                  0.5788155
##
##
## Real Parameter p
##
##           2         3         4         5         6         7
## 1 0.9033228 0.9033228 0.9033228 0.9033228 0.9033228 0.9033228
## 2           0.9033228 0.9033228 0.9033228 0.9033228 0.9033228
## 3                     0.9033228 0.9033228 0.9033228 0.9033228
## 4                               0.9033228 0.9033228 0.9033228
## 5                                         0.9033228 0.9033228
## 6                                                   0.9033228
```

La sélection de modèles.

```
collect.models()
```

```
##                     model npar     AICc DeltaAICc       weight Deviance
## 3 Phi(~Inondation)p(~1)    3 666.1596  0.000000 0.5198230701 51.42300
## 2       Phi(~temp)p(~1)    3 666.5896  0.430000 0.4192588476 51.85299
## 4          Phi(~1)p(~1)    2 670.8660  4.706387 0.0494170312 58.15788
## 6       Phi(~time)p(~1)    7 673.9980  7.838391 0.0103221620 51.05031
## 5       Phi(~1)p(~time)    7 678.7481 12.588471 0.0009600722 55.80039
## 1    Phi(~time)p(~time)   12 681.7057 15.546016 0.0002188170 48.27043
```

Les paramètres estimés.

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

```
##                  estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.5536240 0.0255363 0.5031974 0.6029707
```

```
## Phi g1 c1 a1 t2 0.4564823 0.0480182 0.3649720 0.5510278
## Phi g1 c1 a2 t3 0.4974074 0.0355628 0.4282021 0.5667121
## Phi g1 c1 a3 t4 0.6468361 0.0412945 0.5623869 0.7230149
## Phi g1 c1 a4 t5 0.5887858 0.0277198 0.5335866 0.6418372
## Phi g1 c1 a5 t6 0.5788155 0.0264124 0.5263662 0.6295444
## p g1 c1 a1 t2   0.9033228 0.0283826 0.8316966 0.9464299
```

Visualisons la relation survie vs. température. On créé d'abord une grille pour la température

```r
min.temp <- min(cov.cincle$temperature_hiver_c)
max.temp <- max(cov.cincle$temperature_hiver_c)
temp.values <- seq(from = min.temp, to = max.temp, length = 50)
```

On fait la prédiction. Pour cela il nous faut l'ordonnée à l'origine (intercept) et la pente (slope) de l'effet température sur la survie.

```r
coef(phicov.cincle)
```

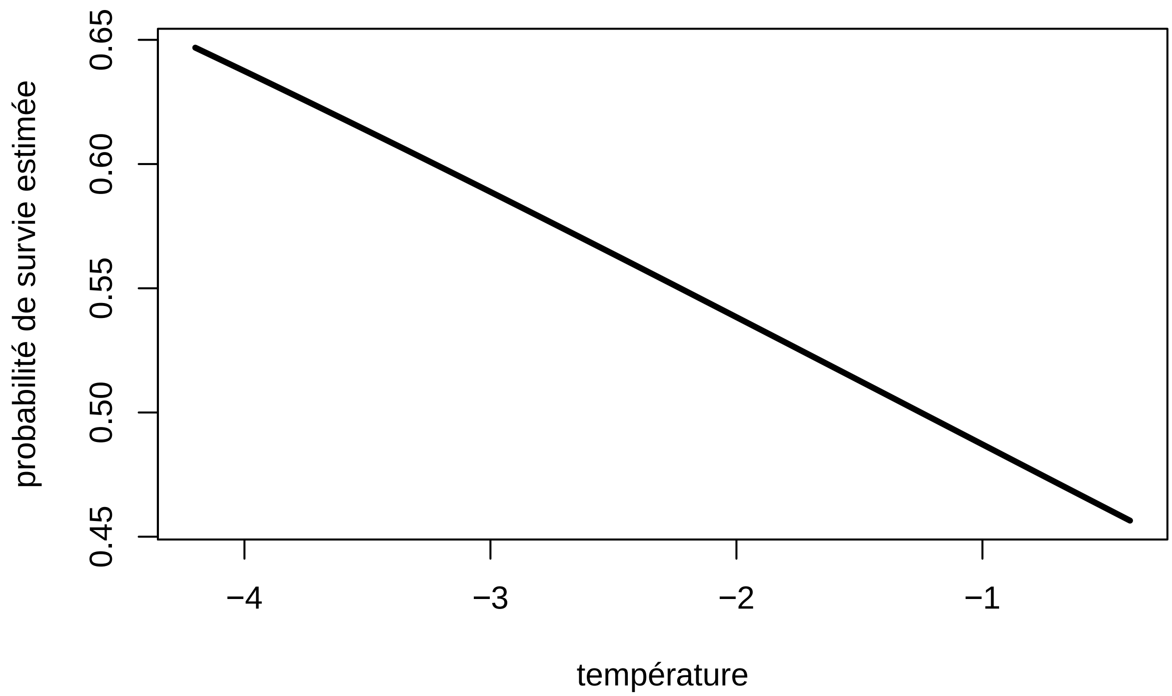
```
##                   estimate        se        lcl        ucl
## Phi:(Intercept) -0.2565831 0.2220600 -0.6918208  0.1786546
## Phi:temp        -0.2051771 0.0821952 -0.3662796 -0.0440745
## p:(Intercept)    2.2347028 0.3250019  1.5976991  2.8717064
```

On applique la relation aux valeurs de la grille, et on ramène les valeurs obtenues de survie sur l'échelle (0,1).

```r
phi.pred <- plogis(coef(phicov.cincle)[1,1] + coef(phicov.cincle)[2,1] * temp.values)
```

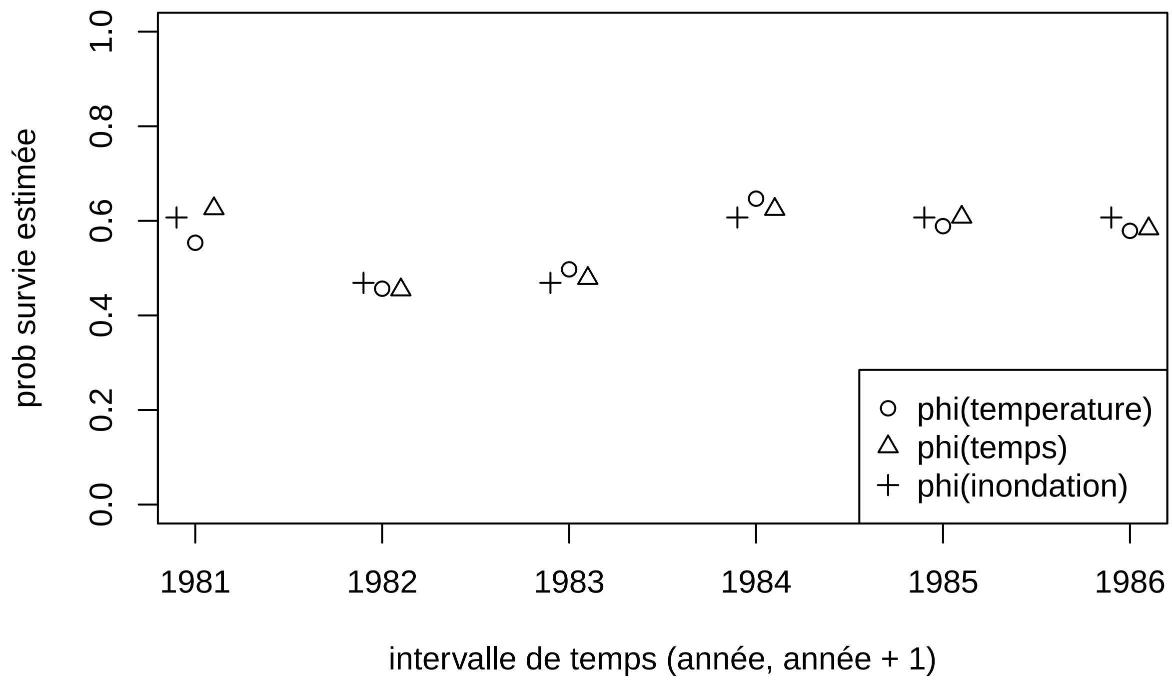
On visualise.

```r
plot(x = temp.values,
     y = phi.pred,
     lwd = 3,
     type = 'l',
     xlab = "température",
     ylab = "probabilité de survie estimée")
```



On représente maintenant les survies estimées pour les 4 meilleurs modèles, le modèle avec effet de l'inondation, de la température, le modèle avec survie constante et le modèle avec effet du temps sur la survie (la prob de recapture est constante dans les 4 modèles).

```
plot(x = 1981:1986,
     y = phicov.cincle$results$real[1:6, 1],
     ylim = c(0,1),
     xlab = "intervalle de temps (année, année + 1)",
     ylab = "prob survie estimée",
     pch = 1)
points(x = 1981:1986 + 0.1,
     y = phitp.cincle$results$real[1:6, 1],
     pch = 2)
points(x = 1981:1986 - 0.1,
     y = c(phiinondationp.cincle$results$real[1,1], # pas inondation
           phiinondationp.cincle$results$real[2,1], # inondation
           phiinondationp.cincle$results$real[2,1], # inondation
           phiinondationp.cincle$results$real[1,1], # pas inondation
           phiinondationp.cincle$results$real[1,1], # pas inondation
           phiinondationp.cincle$results$real[1,1]), # pas inondation
     pch = 3)
legend("bottomright",
       legend = c("phi(temperature)",
                  "phi(temps)",
                  "phi(inondation)"),
       pch = c(1,2,3))
```



Effet de l'âge

On définit l'effet âge (temps écoulé depuis la première capture).

```
phi.age <- list(formula =~ age)
```

On ajuste le modèle.

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

```
##
## Output summary for CJS model
## Name : Phi(~age)p(~1)
##
## Npar :  7  (unadjusted=6)
## -2lnL:  662.8206
## AICc :  677.0885  (unadjusted=675.02107)
##
## Beta
##                    estimate           se           lcl          ucl
## Phi:(Intercept)   0.2052267    0.1393353    -0.0678705    0.4783238
## Phi:age1          0.0214788    0.2537612    -0.4758931    0.5188508
## Phi:age2          0.3587524    0.3690996    -0.3646828    1.0821876
## Phi:age3         -0.0576612    0.5381900    -1.1125136    0.9971912
## Phi:age4          0.2395450    0.8866453    -1.4982799    1.9773699
## Phi:age5        -16.3275670 2200.8014000 -4329.8984000 4297.2433000
## p:(Intercept)     2.2552371    0.3288447     1.6107015    2.8997726
```

```
##
##
## Real Parameter Phi
##
##           1         2         3         4         5            6
## 1 0.5511273 0.5564349 0.6373727 0.5368246 0.6093954 9.957647e-08
## 2           0.5511273 0.5564349 0.6373727 0.5368246 6.093954e-01
## 3                     0.5511273 0.5564349 0.6373727 5.368246e-01
## 4                               0.5511273 0.5564349 6.373727e-01
## 5                                         0.5511273 5.564349e-01
## 6                                                   5.511273e-01
##
##
## Real Parameter p
##
##           2         3         4         5         6         7
## 1 0.9051013 0.9051013 0.9051013 0.9051013 0.9051013 0.9051013
## 2           0.9051013 0.9051013 0.9051013 0.9051013 0.9051013
## 3                     0.9051013 0.9051013 0.9051013 0.9051013
## 4                               0.9051013 0.9051013 0.9051013
## 5                                         0.9051013 0.9051013
## 6                                                   0.9051013
```

La sélection de modèles.

```
collect.models()
```

```
##                    model npar      AICc DeltaAICc       weight Deviance
## 4 Phi(~Inondation)p(~1)    3 666.1596  0.000000 0.5186813025 51.42300
## 3       Phi(~temp)p(~1)    3 666.5896  0.430000 0.4183379647 51.85299
## 5          Phi(~1)p(~1)    2 670.8660  4.706387 0.0493084889 58.15788
## 7       Phi(~time)p(~1)    7 673.9980  7.838391 0.0102994898 51.05031
## 2        Phi(~age)p(~1)    7 677.0885 10.928891 0.0021964543 54.14081
## 6       Phi(~1)p(~time)    7 678.7481 12.588471 0.0009579635 55.80039
## 1    Phi(~time)p(~time)   12 681.7057 15.546016 0.0002183364 48.27043
```

Les paramètres estimés.

```
phiage.cincle$results$real
```

```
##                      estimate          se           lcl       ucl fixed note
## Phi g1 c1 a0 t1 5.511273e-01 0.034469600  4.830389e-01 0.6173520
## Phi g1 c1 a1 t2 5.564349e-01 0.050265700  4.569908e-01 0.6515533
## Phi g1 c1 a2 t3 6.373727e-01 0.079480800  4.725208e-01 0.7752132
## Phi g1 c1 a3 t4 5.368246e-01 0.129373400  2.947856e-01 0.7626719
## Phi g1 c1 a4 t5 6.093954e-01 0.208961600  2.182570e-01 0.8970988
## Phi g1 c1 a5 t6 9.957647e-08 0.000219148 5.539126e-316 1.0000000
## p g1 c1 a1 t2   9.051013e-01 0.028245400  8.335088e-01 0.9478352
```

Ici l'effet d'âge est plein, autrement dit on estime une probabilité de survie pour chaque classe d'âge. On peut contraindre cet effet à un effet jeune (survie qui dure une année de 0 à 1 an) vs. adulte (la même survie pour les individus au-delà d'1 an).

Pour ce faire, on ajoute une variable ageclass.

```r
# create 0, 1+ age variable
cincle.ddl <- add.design.data(cincle.proc,
                              cincle.ddl,
                             "Phi",
                             type = "age",
                             bins = c(0, 1, 7), # 2 classes d'âge
                             name = "ageclass",
                             right = FALSE)
```

Jetons un coup d'oeil.

```r
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
##    Inondation temp ageclass
## 1           0 -2.3    [0,1)
## 2           1 -0.4    [1,7]
## 3           1 -1.2    [1,7]
## 4           0 -4.2    [1,7]
## 5           0 -3.0    [1,7]
## 6           0 -2.8    [1,7]
## 7           1 -0.4    [0,1)
## 8           1 -1.2    [1,7]
## 9           0 -4.2    [1,7]
## 10          0 -3.0    [1,7]
## 11          0 -2.8    [1,7]
## 12          1 -1.2    [0,1)
## 13          0 -4.2    [1,7]
## 14          0 -3.0    [1,7]
## 15          0 -2.8    [1,7]
## 16          0 -4.2    [0,1)
## 17          0 -3.0    [1,7]
```

```
## 18          0 -2.8    [1,7]
## 19          0 -3.0    [0,1)
## 20          0 -2.8    [1,7]
## 21          0 -2.8    [0,1)
```

On spécifie une survie qui dépend de l'âge.

```r
phi.age2 <- list(formula=~ageclass) # age effect on survival
```

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

```r
phiage2.cincle <- mark(cincle.proc,
                        cincle.ddl,
                        model.parameters = list(Phi = phi.age2, p = p))
```

```
##
## Output summary for CJS model
## Name : Phi(~ageclass)p(~1)
##
## Npar :  3
## -2lnL:  666.6804
## AICc :  672.7373
##
## Beta
##                      estimate        se        lcl       ucl
## Phi:(Intercept)     0.2041823 0.1390053 -0.0682681 0.4766326
## Phi:ageclass[1,7]   0.0841951 0.2119782 -0.3312822 0.4996724
## p:(Intercept)       2.2456920 0.3285355  1.6017623 2.8896217
##
##
## Real Parameter Phi
##
##          1         2         3         4         5         6
## 1 0.550869 0.5715988 0.5715988 0.5715988 0.5715988 0.5715988
## 2          0.5508690 0.5715988 0.5715988 0.5715988 0.5715988
## 3                    0.5508690 0.5715988 0.5715988 0.5715988
## 4                              0.5508690 0.5715988 0.5715988
## 5                                        0.5508690 0.5715988
## 6                                                  0.5508690
##
##
## Real Parameter p
##
##           2         3         4         5         6         7
## 1 0.9042783 0.9042783 0.9042783 0.9042783 0.9042783 0.9042783
## 2           0.9042783 0.9042783 0.9042783 0.9042783 0.9042783
## 3                     0.9042783 0.9042783 0.9042783 0.9042783
## 4                               0.9042783 0.9042783 0.9042783
## 5                                         0.9042783 0.9042783
## 6                                                   0.9042783
```

```
phiage2.cincle$results$real
```

```
##                   estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.5508690 0.0343916 0.4829396 0.6169524
## Phi g1 c1 a1 t2 0.5715988 0.0380167 0.4960218 0.6439770
## p g1 c1 a1 t2   0.9042783 0.0284377 0.8322646 0.9473310
```

Que donne la sélection de modèles?

```
collect.models()
```

```
##                    model npar     AICc DeltaAICc       weight Deviance
## 5 Phi(~Inondation)p(~1)    3 666.1596  0.000000 0.5088373938 51.42300
## 4       Phi(~temp)p(~1)    3 666.5896  0.430000 0.4103984444 51.85299
## 6          Phi(~1)p(~1)    2 670.8660  4.706387 0.0483726768 58.15788
## 3   Phi(~ageclass)p(~1)    3 672.7373  6.577620 0.0189787229 58.00061
## 8       Phi(~time)p(~1)    7 673.9980  7.838391 0.0101040186 51.05031
## 2        Phi(~age)p(~1)    7 677.0885 10.928891 0.0021547684 54.14081
## 7       Phi(~1)p(~time)    7 678.7481 12.588471 0.0009397825 55.80039
## 1    Phi(~time)p(~time)   12 681.7057 15.546016 0.0002141926 48.27043
```

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

On remet les compteurs à 0.

```
rm(list = ls())
```

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.

```r
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.

```r
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)   1.7439687   0.8654857    0.0476167    3.4403207
## Phi:time2        -0.9669990   1.0306478   -2.9870687    1.0530708
## Phi:time3        -0.5738962   1.1624664   -2.8523303    1.7045378
## Phi:time4        -0.8957164   1.0338549   -2.9220721    1.1306393
## Phi:time5        -0.9809800   0.9802275   -2.9022260    0.9402659
## Phi:time6        -0.6912502   1.0551084   -2.7592627    1.3767623
## Phi:time7        -1.8256838 361.4618400  -710.2909000  706.6395300
## p:(Intercept)     2.0030692   1.0495408   -0.0540308    4.0601693
## p:time3          -0.9689953   1.1967002   -3.3145277    1.3765371
## p:time4          -1.9340767   1.1630677   -4.2136893    0.3455360
## p:time5          -1.2041767   1.1750411   -3.5072574    1.0989039
## p:time6          -0.0882494   1.2916848   -2.6199516    2.4434529
## p:time7          -0.0861473   1.4799849   -2.9869177    2.8146232
## p:time8          -1.1127807 646.3168000 -1267.8937000 1265.6682000
##
##
## Real Parameter Phi
## Group:colonienord
##           1         2         3         4         5         6         7
## 1 0.8511905 0.6850267 0.7631581 0.7002004 0.6820023 0.7412966 0.4795826
## 2           0.6850267 0.7631581 0.7002004 0.6820023 0.7412966 0.4795826
## 3                     0.7631581 0.7002004 0.6820023 0.7412966 0.4795826
## 4                               0.7002004 0.6820023 0.7412966 0.4795826
## 5                                         0.6820023 0.7412966 0.4795826
## 6                                                   0.7412966 0.4795826
## 7                                                             0.4795826
##
## Group:coloniesud
##           1         2         3         4         5         6         7
## 1 0.8511905 0.6850267 0.7631581 0.7002004 0.6820023 0.7412966 0.4795826
```

```
## 2           0.6850267 0.7631581 0.7002004 0.6820023 0.7412966 0.4795826
## 3                     0.7631581 0.7002004 0.6820023 0.7412966 0.4795826
## 4                               0.7002004 0.6820023 0.7412966 0.4795826
## 5                                         0.6820023 0.7412966 0.4795826
## 6                                                   0.7412966 0.4795826
## 7                                                             0.4795826
##
##
## Real Parameter p
## Group:colonienord
##          2        3         4         5         6         7         8
## 1 0.881119 0.737705 0.5172413 0.6897375 0.8715597 0.8717948 0.7089497
## 2          0.737705 0.5172413 0.6897375 0.8715597 0.8717948 0.7089497
## 3                   0.5172413 0.6897375 0.8715597 0.8717948 0.7089497
## 4                             0.6897375 0.8715597 0.8717948 0.7089497
## 5                                       0.8715597 0.8717948 0.7089497
## 6                                                 0.8717948 0.7089497
## 7                                                           0.7089497
##
## Group:coloniesud
##          2        3         4         5         6         7         8
## 1 0.881119 0.737705 0.5172413 0.6897375 0.8715597 0.8717948 0.7089497
## 2          0.737705 0.5172413 0.6897375 0.8715597 0.8717948 0.7089497
## 3                   0.5172413 0.6897375 0.8715597 0.8717948 0.7089497
## 4                             0.6897375 0.8715597 0.8717948 0.7089497
## 5                                       0.8715597 0.8717948 0.7089497
## 6                                                 0.8717948 0.7089497
## 7                                                           0.7089497
```

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

```
##                      estimate          se          lcl       ucl fixed note
## Phi gnord c1 a0 t1 0.8511905   0.1096270  5.119019e-01 0.9689412
## Phi gnord c1 a1 t2 0.6850267   0.1013890  4.640514e-01 0.8452710
## Phi gnord c1 a2 t3 0.7631581   0.1402705  4.131405e-01 0.9365020
## Phi gnord c1 a3 t4 0.7002004   0.1187097  4.353322e-01 0.8761681
## Phi gnord c1 a4 t5 0.6820023   0.0998051  4.653069e-01 0.8409045
## Phi gnord c1 a5 t6 0.7412966   0.1157331  4.675199e-01 0.9033960
## Phi gnord c1 a6 t7 0.4795826  90.2146610 1.915897e-308 1.0000000
## p gnord c1 a1 t2   0.8811190   0.1099377  4.864956e-01 0.9830463
## p gnord c1 a2 t3   0.7377050   0.1112487  4.768148e-01 0.8966881
## p gnord c1 a3 t4   0.5172413   0.1251483  2.863173e-01 0.7410289
## p gnord c1 a4 t5   0.6897375   0.1130732  4.410917e-01 0.8622990
## p gnord c1 a5 t6   0.8715597   0.0842865  6.080351e-01 0.9674088
## p gnord c1 a6 t7   0.8717948   0.1166268  4.679759e-01 0.9813323
## p gnord c1 a7 t8   0.7089497 133.3610400 1.354977e-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.

```r
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  0.2494412 1.5004695
## Phi:ageclass[1,7] -0.0339140 0.3988106 -0.8155827 0.7477547
## p:(Intercept)      0.8823122 0.2487229  0.3948154 1.3698090
##
##
## 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 0.7073011 0.7073011 0.7073011 0.7073011 0.7073011 0.7073011 0.7073011
## 2           0.7073011 0.7073011 0.7073011 0.7073011 0.7073011 0.7073011
## 3                     0.7073011 0.7073011 0.7073011 0.7073011 0.7073011
## 4                               0.7073011 0.7073011 0.7073011 0.7073011
## 5                                         0.7073011 0.7073011 0.7073011
## 6                                                   0.7073011 0.7073011
## 7                                                             0.7073011
##
## Group:coloniesud
##           2         3         4         5         6         7         8
## 1 0.7073011 0.7073011 0.7073011 0.7073011 0.7073011 0.7073011 0.7073011
## 2           0.7073011 0.7073011 0.7073011 0.7073011 0.7073011 0.7073011
## 3                     0.7073011 0.7073011 0.7073011 0.7073011 0.7073011
## 4                               0.7073011 0.7073011 0.7073011 0.7073011
## 5                                         0.7073011 0.7073011 0.7073011
## 6                                                   0.7073011 0.7073011
## 7                                                             0.7073011
```

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

```
##                      estimate        se       lcl       ucl fixed note
## Phi gnord c1 a0 t1 0.7057758 0.0662714 0.5620390 0.8176445
## Phi gnord c1 a1 t2 0.6986845 0.0463273 0.6010232 0.7811452
## p gnord c1 a1 t2   0.7073011 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.

```r
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.

```r
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)                0.1691814   0.5256397   -0.8610724   1.1994353
## Phi:ageclass[1,7]              0.4793027   0.7462046   -0.9832583   1.9418638
## Phi:coloniesud                 1.4022292   0.7054828    0.0194828   2.7849755
## Phi:ageclass[1,7]:coloniesud  -1.0377648   0.9299041   -2.8603769   0.7848473
## p:(Intercept)                 16.5833710 170.3871000 -317.3753600 350.5421000
## p:coloniesud                 -14.6638560 170.3880200 -348.6243800 319.2966600
## p:time3                      -15.6864410 170.3899600 -349.6507700 318.2778900
## p:time4                      -16.8273220 170.3890500 -350.7898700 317.1352200
## p:time5                      -17.9845410 170.3913800 -351.9516500 315.9825700
## p:time6                      -16.5365940 170.3913700 -350.5036900 317.4305000
## p:time7                       10.0639790   0.0000000   10.0639790  10.0639790
## p:time8                      -17.2037140 170.3868400 -351.1619300 316.7545000
## p:coloniesud:time3            14.7434430 170.3918600 -319.2246100 348.7115000
## p:coloniesud:time4            15.0732480 170.3905600 -318.8922500 349.0387500
## p:coloniesud:time5            17.3956480 170.3934300 -316.5754900 351.3667800
## p:coloniesud:time6            16.9633270 170.3954200 -317.0117100 350.9383600
## p:coloniesud:time7           -10.2907280   0.0000000  -10.2907280 -10.2907280
## p:coloniesud:time8            15.1686380 170.3879900 -318.7918400 349.1291100
##
```

```
##
## Real Parameter Phi
## Group:colonienord
##           1         2         3         4         5         6         7
## 1 0.5421948 0.6566688 0.6566688 0.6566688 0.6566688 0.6566688 0.6566688
## 2           0.5421948 0.6566688 0.6566688 0.6566688 0.6566688 0.6566688
## 3                     0.5421948 0.6566688 0.6566688 0.6566688 0.6566688
## 4                               0.5421948 0.6566688 0.6566688 0.6566688
## 5                                         0.5421948 0.6566688 0.6566688
## 6                                                   0.5421948 0.6566688
## 7                                                             0.5421948
##
## Group:coloniesud
##           1         2         3         4         5         6         7
## 1 0.8279846 0.7335968 0.7335968 0.7335968 0.7335968 0.7335968 0.7335968
## 2           0.8279846 0.7335968 0.7335968 0.7335968 0.7335968 0.7335968
## 3                     0.8279846 0.7335968 0.7335968 0.7335968 0.7335968
## 4                               0.8279846 0.7335968 0.7335968 0.7335968
## 5                                         0.8279846 0.7335968 0.7335968
## 6                                                   0.8279846 0.7335968
## 7                                                             0.8279846
##
##
## Real Parameter p
## Group:colonienord
##           2         3         4         5         6 7         8
## 1 0.9999999 0.7103183 0.4393129 0.1976305 0.5116921 1 0.3497035
## 2           0.7103183 0.4393129 0.1976305 0.5116921 1 0.3497035
## 3                     0.4393129 0.1976305 0.5116921 1 0.3497035
## 4                               0.1976305 0.5116921 1 0.3497035
## 5                                         0.5116921 1 0.3497035
## 6                                                   1 0.3497035
## 7                                                     0.3497035
##
## Group:coloniesud
##           2         3         4         5         6         7        8
## 1 0.8720843 0.7264167 0.5412661 0.7909434 0.9126355 0.8445876 0.471142
## 2           0.7264167 0.5412661 0.7909434 0.9126355 0.8445876 0.471142
## 3                     0.5412661 0.7909434 0.9126355 0.8445876 0.471142
## 4                               0.7909434 0.9126355 0.8445876 0.471142
## 5                                         0.9126355 0.8445876 0.471142
## 6                                                   0.8445876 0.471142
## 7                                                             0.471142
```

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

```
##                       estimate           se           lcl       ucl fixed note
## Phi gnord c1 a0 t1 0.5421948 1.304741e-01  2.971153e-01 0.7684243
## Phi gnord c1 a1 t2 0.6566688 1.044160e-01  4.355456e-01 0.8258115
## Phi gsud c1 a0 t1  0.8279846 6.701740e-02  6.568188e-01 0.9236970
## Phi gsud c1 a1 t2  0.7335968 5.103750e-02  6.227158e-01 0.8212450
## p gnord c1 a1 t2   0.9999999 1.069966e-05 1.464311e-138 1.0000000
## p gnord c1 a2 t3   0.7103183 2.128974e-01  2.439772e-01 0.9490616
## p gnord c1 a3 t4   0.4393129 2.616158e-01  8.901790e-02 0.8626864
```

```
## p gnord c1 a4 t5   0.1976305 1.847867e-01  2.447840e-02 0.7074109
## p gnord c1 a5 t6   0.5116921 2.865197e-01  9.968160e-02 0.9084056
## p gnord c1 a6 t7   1.0000000 0.000000e+00  1.000000e+00 1.0000000
## p gnord c1 a7 t8   0.3497035 2.284807e-01  6.981320e-02 0.7939461
## p gsud c1 a1 t2    0.8720843 1.169522e-01  4.662154e-01 0.9815555
## p gsud c1 a2 t3    0.7264167 1.196539e-01  4.492870e-01 0.8962827
## p gsud c1 a3 t4    0.5412661 1.239404e-01  3.072694e-01 0.7583762
## p gsud c1 a4 t5    0.7909434 1.016560e-01  5.313691e-01 0.9266002
## p gsud c1 a5 t6    0.9126355 8.004330e-02  5.935336e-01 0.9867955
## p gsud c1 a6 t7    0.8445876 1.129742e-01  5.014515e-01 0.9670649
## p gsud c1 a7 t8    0.4711420 1.002335e-01  2.882256e-01 0.6621514
```

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
```

La sélection de modèles.

```r
collect.models()
```

```
##                                         model npar     AICc DeltaAICc     weight
## 4                                 Phi(~1)p(~1)    2 376.9136  0.000000 0.64807823
## 2                          Phi(~ageclass)p(~1)    3 378.9672  2.053631 0.23210645
## 3 Phi(~ageclass * colonie)p(~colonie * time)   18 380.4701  3.556533 0.10948031
## 1                           Phi(~time)p(~time)   14 385.1905  8.276948 0.01033501
##   Deviance
## 4 133.6472
## 2 133.6399
## 3 101.5263
## 1 115.7384
```

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 : $\phi = 0.8$, $p = 0.2$.

```r
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)
}
```

```r
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 ?

```r
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.

```r
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.

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

On ajuste le modèle avec paramètres constants aux données G1.

```r
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
```

```
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 0.7803895 0.8272818
```

Puis aux données G2.

```
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  1.269927  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
```

```
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) 1.1639805 0.0436060 1.0785126 1.2494483
## p:(Intercept)   0.3450607 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
```

```
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.

```
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) 0.7598158 0.0909051  0.5816419 0.9379898
## Phi:time2       0.3979968 0.1933475  0.0190357 0.7769579
## Phi:time3       0.7072561 0.2137485  0.2883090 1.1262032
## Phi:time4       0.6334194 0.2291570  0.1842718 1.0825671
## Phi:time5       0.4558440 0.2336746 -0.0021582 0.9138462
## Phi:time6       0.8182723 0.3428348  0.1463161 1.4902285
```

```
## Phi:time7       1.0221786 0.5473029 -0.0505351 2.0948923
## p:(Intercept)   0.3870597 0.0505148  0.2880506 0.4860688
##
##
## Real Parameter Phi
##
##           1        2         3         4         5         6        7
## 1 0.6813137 0.760935 0.8126119 0.8011082 0.7712989 0.8289336 0.855943
## 2           0.760935 0.8126119 0.8011082 0.7712989 0.8289336 0.855943
## 3                    0.8126119 0.8011082 0.7712989 0.8289336 0.855943
## 4                              0.8011082 0.7712989 0.8289336 0.855943
## 5                                        0.7712989 0.8289336 0.855943
## 6                                                  0.8289336 0.855943
## 7                                                            0.855943
##
##
## 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
```

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

```
##                   estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.6813137 0.0197378 0.6414451 0.7186934
## Phi g1 c1 a1 t2 0.7609350 0.0259835 0.7063778 0.8081089
## Phi g1 c1 a2 t3 0.8126119 0.0294917 0.7479047 0.8637363
## Phi g1 c1 a3 t4 0.8011082 0.0335599 0.7271892 0.8588853
## Phi g1 c1 a4 t5 0.7712989 0.0379757 0.6886252 0.8372108
## Phi g1 c1 a5 t6 0.8289336 0.0470411 0.7166460 0.9027615
## Phi g1 c1 a6 t7 0.8559430 0.0668937 0.6723161 0.9450757
## 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

```
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.

```r
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)
```

```r
dim(G1transit)
```

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

```r
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
```

```r
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)   1.1885540 0.0770665 1.0375037 1.3396043
##
##
## 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
```

```r
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 0.7383681 0.7924249
```

Idem avec survie qui dépend du temps.

```r
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)  1.1097471 0.1223381  0.8699643  1.3495299
## Phi:time2       -0.3581906 0.1908887 -0.7323323  0.0159512
## Phi:time3       -0.2686693 0.1890297 -0.6391674  0.1018289
## Phi:time4       -0.4461149 0.1934615 -0.8252995 -0.0669304
## Phi:time5       -0.4810596 0.2040144 -0.8809279 -0.0811914
## Phi:time6       -0.3850170 0.2250948 -0.8262027  0.0561688
## Phi:time7       -0.8490585 0.2307665 -1.3013608 -0.3967562
## 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 0.6521918 0.6736478 0.5648056
## 2          0.6795178 0.6986922 0.6600758 0.6521918 0.6736478 0.5648056
## 3                    0.6986922 0.6600758 0.6521918 0.6736478 0.5648056
## 4                              0.6600758 0.6521918 0.6736478 0.5648056
## 5                                        0.6521918 0.6736478 0.5648056
## 6                                                  0.6736478 0.5648056
## 7                                                            0.5648056
##
##
## 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
```

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

```
##                    estimate        se       lcl       ucl fixed note
## Phi g1 c1 a0 t1 0.7520820 0.0228105 0.7047383 0.7940528
## Phi g1 c1 a1 t2 0.6795178 0.0270283 0.6244072 0.7300381
## Phi g1 c1 a2 t3 0.6986922 0.0305397 0.6356994 0.7549907
## Phi g1 c1 a3 t4 0.6600758 0.0337817 0.5911056 0.7228668
## Phi g1 c1 a4 t5 0.6521918 0.0371981 0.5762202 0.7211351
## Phi g1 c1 a5 t6 0.6736478 0.0419754 0.5867403 0.7500641
## Phi g1 c1 a6 t7 0.5648056 0.0489543 0.4676276 0.6572466
## p g1 c1 a1 t2   0.7661500 0.0140892 0.7374131 0.7926263
```

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 ?

```
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.

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

On ajuste le modèle.

```
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)   -0.1000537 0.0738121 -0.2447253 0.044618
## Phi:ageclass[1,7]  1.4656701 0.1046905  1.2604767 1.670863
## p:(Intercept)      1.3783585 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
```

```
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.

```
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 -0.2447254 0.044618
## Phi:max.age1     1.4656701 0.1046905  1.2604767 1.670864
## p:(Intercept)    1.3783585 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
```

```r
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.

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