Vignette IndRSA

Guillaume Bastille-Rousseau

July 19, 2021

Individual and population-level RSF - *rsf_ind* and *pop_avg*

The function *rsf_ind* performs individual-level RSFs. The function takes a list of candidate models (using *as.formula()*). The function *aictab_ind* performs population-level AIC model selection (by adding up individual model loglikelihood). *pop_avg* performs population-level averaging. Two methods are available for calculating confidence intervals, the default (and recommended) is based on Murtaugh (2007). A bootstrap approach (based on Prokopenko 2016) is also available. *ind_coef* and *ind_se* also extract individual-level coefficients and standard errors. Murtaugh, P. Simplicity and complexity in ecological data anlysis. Ecology 88, 56–62 (2007). Prokopenko, C. M., Boyce, M. S. & Avgar, T. Characterizing wildlife behavioural responses to roads using integrated step selection analysis. J. Appl. Ecol. 1, (2016).

```
data(goats)
ls1<-list()
ls1[[1]]<-as.formula(STATUS~ELEVATION+SLOPE+ET+ASPECT+HLI+TASP)
ls1[[2]]<-as.formula(STATUS~ET+ASPECT+HLI+TASP)
out<-rsf_ind(goats$ID, data=goats, form_ls=ls1)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
aictab_ind(out) # Model 1 is the best
```

```
##   Modnames        LL K     AICc Delta_AICc     ModelLik       AICcWt
## 1        1 -8161.769 7 16393.54     0.0000  1.00000e+00  1.00000e+00
## 2        2 -8656.390 5 17337.78   944.2414 9.13254e-206 9.13254e-206
```

```
#Calculate population average for the first model of the list (m=1)
pop<-pop_avg(m=1, out, method="murtaugh")
pop[[1]] #Population average summary
```

```
##                       Mean         LCI          UCI
## X.Intercept.  0.3912143716 -4.00899424  4.791422979
## ELEVATION     0.0001629426 -0.00243771  0.002763596
## SLOPE         0.0040336976 -0.02465580  0.032723198
## ET           -0.0154533579 -0.02196717 -0.008939550
## ASPECT       -0.0010958503 -0.01839513  0.016203426
## HLI           2.1003857812  0.21454516  3.986226403
## TASP          0.2659307215 -0.94227732  1.474138762
```

```
pop[[2]] #Individual level coefficients
```

```
##    name X.Intercept.     ELEVATION       SLOPE           ET        ASPECT
## 1     1 -10.13342821  4.236729e-03 -0.03054204 -0.014568265  0.0140584927
## 2    10   3.85166898 -4.910948e-03  0.01030740 -0.010624612  0.0000303647
## 3     2  -1.70616513  2.009721e-03 -0.01810113 -0.009539741 -0.0034456556
## 4     3  -6.40055314  5.390393e-03  0.05824633 -0.008133348  0.0061258662
## 5     4   0.09941785 -1.681873e-03  0.01289947 -0.012552525  0.0033421256
## 6     5  -5.93245004 -5.092003e-06  0.02191594 -0.008507392  0.0251354208
## 7     6  12.37871241 -5.489341e-03 -0.02361020 -0.035424039 -0.0642285201
## 8     7  -5.51546257 -1.424490e-03  0.06593032 -0.008339853  0.0129314837
## 9     8  -1.39380208  2.960728e-03 -0.05535403 -0.022932242 -0.0088051689
## 10    9   2.03953303  5.435987e-04 -0.02831900 -0.023911562  0.0038970882
##           HLI        TASP ID Freq
## 1   7.7848091 -0.96253280  1    1
## 2   4.3830861 -1.00491280 10    1
## 3   0.6083625  0.95036169  2    1
## 4  -0.6543761  0.06491749  3    1
## 5  -1.0838909  1.56390026  4    1
## 6   1.6264778 -1.37636758  5    1
## 7   4.3060802  4.48403249  6    1
## 8   3.9934187 -1.11284986  7    1
## 9   2.5685909  0.09324346  8    1
## 10 -3.0630299  1.43735058  9    1
```

```
ind_coef(m=1, out) #Another way of getting individual coefficients
```

```
##    name X.Intercept.     ELEVATION       SLOPE           ET        ASPECT
## 1     1 -10.13342821  4.236729e-03 -0.03054204 -0.014568265  0.0140584927
## 2    10   3.85166898 -4.910948e-03  0.01030740 -0.010624612  0.0000303647
## 3     2  -1.70616513  2.009721e-03 -0.01810113 -0.009539741 -0.0034456556
## 4     3  -6.40055314  5.390393e-03  0.05824633 -0.008133348  0.0061258662
## 5     4   0.09941785 -1.681873e-03  0.01289947 -0.012552525  0.0033421256
## 6     5  -5.93245004 -5.092003e-06  0.02191594 -0.008507392  0.0251354208
## 7     6  12.37871241 -5.489341e-03 -0.02361020 -0.035424039 -0.0642285201
## 8     7  -5.51546257 -1.424490e-03  0.06593032 -0.008339853  0.0129314837
## 9     8  -1.39380208  2.960728e-03 -0.05535403 -0.022932242 -0.0088051689
## 10    9   2.03953303  5.435987e-04 -0.02831900 -0.023911562  0.0038970882
##           HLI        TASP ID Freq
## 1   7.7848091 -0.96253280  1    1
## 2   4.3830861 -1.00491280 10    1
## 3   0.6083625  0.95036169  2    1
## 4  -0.6543761  0.06491749  3    1
## 5  -1.0838909  1.56390026  4    1
## 6   1.6264778 -1.37636758  5    1
## 7   4.3060802  4.48403249  6    1
## 8   3.9934187 -1.11284986  7    1
## 9   2.5685909  0.09324346  8    1
## 10 -3.0630299  1.43735058  9    1
```

```
ind_se(m=1, out) #Individual level standard errors
```

```
##    name X.Intercept.    ELEVATION       SLOPE          ET       ASPECT
## 1     1    1.5476203 0.0003683769 0.018260682 0.001295096 0.0043686987
## 2    10    0.9003622 0.0003765816 0.009155507 0.001298387 0.0011426357
## 3     2    0.7247551 0.0004064946 0.008499543 0.001045511 0.0018195899
## 4     3    1.0116713 0.0006325931 0.009435019 0.001257300 0.0044977709
## 5     4    1.5958551 0.0004173023 0.009569907 0.002363429 0.0062629399
## 6     5    0.9032989 0.0004851826 0.008816969 0.001060137 0.0043021825
## 7     6    1.4459139 0.0006131895 0.009734168 0.002300391 0.0081069777
## 8     7    0.8015992 0.0003565728 0.010539403 0.001283016 0.0020406725
## 9     8    0.7235709 0.0002638068 0.007866326 0.001691144 0.0018135862
## 10    9    0.7138810 0.0002663567 0.005883089 0.001562936 0.0005441483
##          HLI      TASP ID Freq
## 1  2.2132177 0.7749198  1    1
## 2  1.2291081 0.4568789 10    1
## 3  1.0280597 0.3318805  2    1
## 4  1.1829801 0.5922559  3    1
## 5  2.0474102 0.9933123  4    1
## 6  1.1758283 0.5938980  5    1
## 7  1.2088065 0.8539357  6    1
## 8  1.2209609 0.5122544  7    1
## 9  0.9278968 0.3813859  8    1
## 10 0.7797046 0.3098031  9    1
```

```
kfold_ind(m=1, out, ls=ls1, grph=F) #kfold cross validation for each individual
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##         1         2         3         4         5         6         7         8
## 0.9362622 0.9311141 0.6953561 0.8630876 0.7872327 0.6558205 0.8811079 0.9627288
##         9        10
## 0.9437966 0.9164559
```

Individual and population-level SSF - *ssf_ind* and *pop_avg*

The function *ssf_ind* performs individual-level SSFs similar to *rsf_ind*. The function takes a list of candidate models (using *as.formula()*). The function *aictab_ind*, *pop_avg*, *ind_coef*, and *ind_se* works similarly than in example above. For the example below, we created a new column strata in the goat dataset that assigns two random points to each used points to create a conditional design similar to an SSF

```
data(goats)
goats<-goats[order(goats$ID),]
goats_use<-goats[goats$STATUS==1,]
```

```
goats_use$strata<-1:nrow(goats_use)
goats_rnd<-goats[goats$STATUS==0,]
goats_rnd$strata<-rep(1:nrow(goats_use), each=2)
goats_ssf<-rbind(goats_use, goats_rnd)
ls1<-list()
ls1[[1]]<-as.formula(STATUS~ELEVATION+SLOPE+ET+ASPECT+HLI+TASP+strata(strata))
ls1[[2]]<-as.formula(STATUS~ET+ASPECT+HLI+TASP+strata(strata))
out<-ssf_ind(goats_ssf$ID, data=goats_ssf, form_ls=ls1)
aictab_ind(out)
```

```
##   Modnames        LL K     AICc Delta_AICc      ModelLik        AICcWt
## 1        1 -3557.022 6 7154.044     0.0000  1.000000e+00  1.000000e+00
## 2        2 -3929.442 4 7874.884   720.8402 2.961691e-157 2.961691e-157
```

```
pop<-pop_avg(m=1, out, method="murtaugh")
pop[[1]] #Population average summary
```

```
##                    Mean          LCI          UCI
## ELEVATION  0.0002387237 -0.002127378  0.002604825
## SLOPE      0.0054329562 -0.025451896  0.036317809
## ET        -0.0146423565 -0.020585471 -0.008699242
## ASPECT     0.0007332586 -0.013140326  0.014606843
## HLI        2.0303081994  0.001572472  4.059043927
## TASP       0.1374410358 -0.838568472  1.113450544
```

```
pop[[2]] #Individual level coefficients
```

```
##    name     ELEVATION       SLOPE           ET        ASPECT        HLI
## 1     1  0.0042535122 -0.03956746 -0.013922777  0.0120851194 10.0080201
## 2    10 -0.0045373321  0.01857580 -0.009569205 -0.0002736992  4.7690322
## 3     2  0.0021011253 -0.01275551 -0.009215474 -0.0023106238  0.2044863
## 4     3  0.0048410465  0.05556333 -0.008044269  0.0080225171 -0.7486858
## 5     4 -0.0019231403  0.01672816 -0.012698985  0.0055535798 -1.6998418
## 6     5  0.0000632072  0.02160109 -0.008079944  0.0227318679  2.0333443
## 7     6 -0.0043200599 -0.04408664 -0.031916364 -0.0480950918  5.3337326
## 8     7 -0.0013199634  0.07219259 -0.007710489  0.0147231386  2.9712963
## 9     8  0.0028754110 -0.05147245 -0.021938475 -0.0090389675  2.1548643
## 10    9  0.0003534303 -0.02744158 -0.023327584  0.0039347458 -2.7349175
##           TASP ID Freq
## 1  -1.94597462  1    1
## 2  -1.09791964 10    1
## 3   0.96346547  2    1
## 4  -0.09422795  3    1
## 5   1.94682101  4    1
## 6  -1.25803161  5    1
## 7   2.77097060  6    1
## 8  -0.78124897  7    1
## 9   0.29098855  8    1
## 10  1.38156282  9    1
```

```
coef<-ind_coef(m=1, out)
se<-ind_se(m=1, out)
```

Metrics of variation in resource selection behavior (specialization, heterogeneity, consistency and reversal) - *simu_spe*, *simu_sd*, *simu_consX*, and *simu_revX*

The functions *simu_spe* and *simu_sd* calculates specialization and heterogeneity metrics as described in Bastille-Rousseau & Wittemyer (in review). To propagate uncertainty associated to individual coefficients, *simu_coefs* is first used to simulate individual coefficients based on the coefficient value and its standard error.

```
data(goats)
ls1<-list()
ls1[[1]]<-as.formula(STATUS~ELEVATION+SLOPE+ET+ASPECT+HLI+TASP)
out<-rsf_ind(goats$ID, data=goats, form_ls=ls1)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
coef<-ind_coef(m=1, out)
se<-ind_se(m=1, out)
simu<-simu_coefs(coef, se, n=100)

#Specialization
spe<-simu_spe(simu)
colnames(spe)<-names(coef)
head(spe)
```

```
##      name X.Intercept.   ELEVATION      SLOPE         ET     ASPECT      HLI
## [1,]  NaN     5.685987 0.003131398 0.02970778 0.01561434 0.01500891 2.914270
## [2,]  NaN     4.933090 0.002732592 0.03135758 0.01571801 0.01408842 3.141001
## [3,]  NaN     5.155552 0.002806515 0.03692438 0.01414097 0.01404773 3.037399
## [4,]  NaN     4.686901 0.002906583 0.03177717 0.01568819 0.01650757 3.366288
## [5,]  NaN     4.791894 0.002885848 0.03552904 0.01511553 0.01342007 2.516391
## [6,]  NaN     5.295584 0.002924374 0.03891887 0.01498334 0.01521567 2.934172
##          TASP  ID Freq
## [1,] 1.427029 NaN  NaN
## [2,] 1.567335 NaN  NaN
## [3,] 1.091918 NaN  NaN
## [4,] 1.660139 NaN  NaN
## [5,] 1.607578 NaN  NaN
## [6,] 1.334395 NaN  NaN
```

```
apply(spe, 2, quantile, na.rm=T) #Show variation around estimate of each covariate
```

```
##      name X.Intercept.   ELEVATION      SLOPE         ET     ASPECT      HLI
## 0%     NA     4.362545 0.002625538 0.02571872 0.01414097 0.01195264 2.197576
## 25%    NA     4.873133 0.002826341 0.03084652 0.01506311 0.01387240 2.891070
```

```
## 50%    NA     5.051423 0.002890441 0.03325192 0.01540014 0.01440641 3.138497
## 75%    NA     5.230011 0.002969025 0.03551870 0.01576504 0.01510030 3.415186
## 100%   NA     5.831183 0.003221333 0.04041247 0.01640400 0.01705602 4.122431
##          TASP ID Freq
## 0%   1.000031 NA   NA
## 25%  1.267981 NA   NA
## 50%  1.411219 NA   NA
## 75%  1.573891 NA   NA
## 100% 1.747967 NA   NA
```

```
colMeans(spe) #Calculate average specialization for each covariate
```

```
##         name X.Intercept.    ELEVATION        SLOPE           ET       ASPECT
##          NaN  5.072892533  0.002900413  0.033242618  0.015421154  0.014499684
##          HLI         TASP           ID         Freq
##  3.132356845  1.413921633          NaN          NaN
```

```
#Heterogeneity
sd<-simu_sd(simu)
colnames(sd)<-names(coef)
head(sd)
```

```
##      name X.Intercept.   ELEVATION      SLOPE          ET     ASPECT      HLI
## [1,]   NA     7.109468 0.003928912 0.03774199 0.010238524 0.02447672 3.212455
## [2,]   NA     6.389114 0.003533395 0.03957933 0.008975995 0.02148072 2.995797
## [3,]   NA     6.718919 0.003457206 0.04681968 0.008682079 0.02448959 3.213130
## [4,]   NA     5.577970 0.003662908 0.04463251 0.009559191 0.02840865 3.885276
## [5,]   NA     5.933837 0.003625301 0.04295000 0.009129926 0.02397845 2.828080
## [6,]   NA     6.792148 0.003722128 0.04737048 0.009862179 0.02446088 3.033219
##          TASP ID Freq
## [1,] 2.106938 NA   NA
## [2,] 2.090342 NA   NA
## [3,] 1.297120 NA   NA
## [4,] 2.239586 NA   NA
## [5,] 2.237342 NA   NA
## [6,] 1.718083 NA   NA
```

```
apply(sd, 2, quantile, na.rm=T) #Show variation around estimate of each covariate
```

```
##      name X.Intercept.   ELEVATION      SLOPE          ET     ASPECT      HLI
## 0%     NA     5.524011 0.003273688 0.03505007 0.007780620 0.01948679 2.273848
## 25%    NA     6.213879 0.003562324 0.03916736 0.008951928 0.02323065 2.995053
## 50%    NA     6.418689 0.003658145 0.04157174 0.009347474 0.02446880 3.376090
## 75%    NA     6.704991 0.003765746 0.04378089 0.009758936 0.02556537 3.750659
## 100%   NA     7.536486 0.004059013 0.04884258 0.010630410 0.02910282 4.924076
##          TASP ID Freq
## 0%   1.281287 NA   NA
## 25%  1.736921 NA   NA
## 50%  1.918881 NA   NA
## 75%  2.105534 NA   NA
## 100% 2.391654 NA   NA
```

```
colMeans(sd) #Calculate average heterogeneity for each covariate
```

```
##          name X.Intercept.    ELEVATION        SLOPE           ET       ASPECT
##            NA  6.467346217  0.003661548  0.041481196  0.009324034  0.024475402
##           HLI         TASP           ID         Freq
##  3.441173306  1.907426426           NA           NA
```

For consistency and reversal, calculations are done one covariate at a time. *simu_cons2* and *simu_rev2* are used for calculations when there are two time periods and *simu_cons3* and *simu_rev3* are used when there is three temporal periods. For the example below, we created an artificial Season column to the goat dataset to estimate temporal consistency and reversal.

```
data(goats)
goats$Season<-c("1", "2") #Adding a fake season column
ls1<-list()
ls1[[1]]<-as.formula(STATUS~(ELEVATION+SLOPE+ET+ASPECT+HLI+TASP):Season)
out<-rsf_ind(goats$ID, data=goats, form_ls=ls1)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
coef<-ind_coef(m=1, out)
se<-ind_se(m=1, out)
simu<-simu_coefs(coef, se, n=100)

#Consistency for elevation
head(coef)
```

```
##   name X.Intercept. ELEVATION.Season1 ELEVATION.Season2 SLOPE.Season1
## 1    1  -10.1969189      4.872827e-03      3.788620e-03  -0.014344052
## 2   10    3.8614975     -5.091304e-03     -4.782372e-03   0.009929352
## 3    2   -1.6713766      2.252606e-03      1.755898e-03  -0.009301698
## 4    3   -6.3690662      5.205354e-03      5.606441e-03   0.054617460
## 5    4    0.7593941     -1.974396e-03     -1.401416e-03   0.004315306
## 6    5   -5.9348935     -8.467312e-05      6.639854e-05   0.023361158
##   SLOPE.Season2   ET.Season1   ET.Season2 ASPECT.Season1 ASPECT.Season2
## 1   -0.04243608 -0.014572266 -0.014520662   0.0093726078   1.851435e-02
## 2    0.01053813 -0.011594815 -0.009779521   0.0001143456  -9.853356e-05
## 3   -0.02713040 -0.009539203 -0.009601034  -0.0027131463  -4.267519e-03
## 4    0.06192616 -0.009630821 -0.007007847   0.0046601008   6.968818e-03
## 5    0.02078136 -0.013264766 -0.011976711  -0.0068558202   3.785424e-03
## 6    0.01998914 -0.009000024 -0.008088588   0.0267847019   2.370175e-02
##   HLI.Season1 HLI.Season2 TASP.Season1 TASP.Season2 ID Freq
## 1   8.3437525    7.107471  -1.37063362   -0.5643677  1    1
## 2   4.9088467    3.980008  -1.23189587   -0.8195229 10    1
## 3  -0.3288005    1.504011   0.94516107    0.9638896  2    1
## 4   0.1182481   -1.404355   0.06388014    0.1052738  3    1
## 5   0.3792886   -2.890543   2.36942823    1.9024422  4    1
## 6   1.3229182    1.938761  -1.43818622   -1.3346918  5    1
```

```
#Calculate specialization for elevation covariate, column 3 and 4 contains
#coefficients for elevation for each season
cons_elevation<-simu_cons2(simu, 3, 4)
quantile(cons_elevation) #Show variation around estimate of elevation covariate
```

```
##           0%          25%          50%          75%         100%
## 0.0003849605 0.0006409278 0.0007937824 0.0009031580 0.0013144232
```

```
mean(cons_elevation) #Calculate average consistency for elevation covariate
```

```
## [1] 0.0007830646
```

```
#Reversal for elevation
head(coef)
```

```
##   name X.Intercept. ELEVATION.Season1 ELEVATION.Season2 SLOPE.Season1
## 1    1  -10.1969189      4.872827e-03      3.788620e-03  -0.014344052
## 2   10    3.8614975     -5.091304e-03     -4.782372e-03   0.009929352
## 3    2   -1.6713766      2.252606e-03      1.755898e-03  -0.009301698
## 4    3   -6.3690662      5.205354e-03      5.606441e-03   0.054617460
## 5    4    0.7593941     -1.974396e-03     -1.401416e-03   0.004315306
## 6    5   -5.9348935     -8.467312e-05      6.639854e-05   0.023361158
##   SLOPE.Season2   ET.Season1   ET.Season2 ASPECT.Season1 ASPECT.Season2
## 1   -0.04243608 -0.014572266 -0.014520662   0.0093726078   1.851435e-02
## 2    0.01053813 -0.011594815 -0.009779521   0.0001143456  -9.853356e-05
## 3   -0.02713040 -0.009539203 -0.009601034  -0.0027131463  -4.267519e-03
## 4    0.06192616 -0.009630821 -0.007007847   0.0046601008   6.968818e-03
## 5    0.02078136 -0.013264766 -0.011976711  -0.0068558202   3.785424e-03
## 6    0.01998914 -0.009000024 -0.008088588   0.0267847019   2.370175e-02
##   HLI.Season1 HLI.Season2 TASP.Season1 TASP.Season2 ID Freq
## 1   8.3437525    7.107471  -1.37063362   -0.5643677  1    1
## 2   4.9088467    3.980008  -1.23189587   -0.8195229 10    1
## 3  -0.3288005    1.504011   0.94516107    0.9638896  2    1
## 4   0.1182481   -1.404355   0.06388014    0.1052738  3    1
## 5   0.3792886   -2.890543   2.36942823    1.9024422  4    1
## 6   1.3229182    1.938761  -1.43818622   -1.3346918  5    1
```

```
rev_elevation<-simu_rev2(simu, 3, 4) #Calculate specialization for elevation covariate
quantile(rev_elevation) #Show variation around estimate of elevation covariate
```

```
##   0%  25%  50%  75% 100%
##  0.0  0.0  0.1  0.1  0.2
```

```
mean(rev_elevation) #Calculate average reversal for elevation covariate
```

```
## [1] 0.082
```