Package `sra`: tutorial

Arnaud Le Rouzic

December 20, 2010

The package `sra` (Selection Response Analysis) proposes a set of functions implementing genetic models describing the evolution of populations submitted to artificial selection. This document is an informal tutorial describing the practical use of the software, as well as some basic concepts in quantitative genetics models. The theoretical concepts and the models on which the package is based are fully described in the following reference:

- A modelling framework for the analysis of artificial-selection time series. A. Le Rouzic, D. Houle, and T.F. Hansen (2011), Genetics Research (in press).

Additional information about similar models or alternative statistical frameworks can be found in the literature. The reader may want to have a look at e.g.:

- Evolution and Selection of Quantitative Traits. B. Walsh and M. Lynch, unpublished `http://nitro.biosci.arizona.edu/zbook/volume_2/vol2.html`.
- Estimating genetic architectures from artificial-selection responses: a random-effect framework. A. Le Rouzic, H.J. Skaug, and T.F. Hansen (2010b), Theor. Popul. Biol. 77:119–130.

The last version of the package can be downloaded on the official CRAN site (`http://cran.r-project.org/web/packages/sra/index.html`), where the reference manual is also available. More information about `R` in general, including manuals and tutorials, can be found on `http://www.r-project.org`.

For clarity, this tutorial is devoid of scientific citations: these can be found in the references above.

Contents

1 Context

Artificial selection expriments are common designs in both breeding sciences and evolutionary biology. The procedure consists in measuring a trait in all individuals of a population, and to select a subset of this population for breeding, based on the measured phenotype. Provided that the variation in the trait value relies on genetic bases, the selected character will be more frequent or more pronounced among the offspring of the selected individuals, and the process can be repeated for several generations, accumulating genetic change in the course of time. The trait can be discrete (such as presence/absence of a character) or continuous (e.g. size, weight...). The current statistical framework can handle only the latter, i.e. "classical" continuous quantitative characters.

The response to artificial selection, i.e. the rate of change under selection pressure, depends both on the selection strength and on the genetic architecture of the character. Most of the time, this genetic architecture is underlain by several genes of various effect on the trait of interest, and by complex interactions between these genes and the environment. Dissecting genetic architectures is in general costly and often not completely satisfactory (only a subset of the important genes can be mapped), and it is common to describe genetic architectures by a few parameters designed to summary their evolutionary properies. The additive genetic variance (σ^2_A) of the population is probably the most interesting of them. The additive variance is the part of the genetic variation that is directly heritable, transmitted from the parents to the offspring. Consequently, the additive variance determines precisey the genetic progress in the population. Another important quantity is the environmental variance σ^2_E which represents the residual variance of the genetic system, i.e. the part of the phenotypic variance (σ^2_P) that cannot be explained by genes. Often, this unexplained variance can be larger than the genetic variance.

The current package implements some of the models described in Le Rouzic et al. 2011. These models are designed to catch and analyze the changes in genetic architectures in the course of time in artificially-selected populations. Models are based on summary statistics (means and variances), so that knowing the pedigree structure is unnecessary. Two main families of models are proposed: "phenomenological" models catch the trends without proposing biological explanations, while "mechanistic" models implement known genetic mechanisms and attempt at interpreting the changes in terms of e.g. genetic drift, mutations, or epistasis. Models are fit by maximizing numerically their likelihood, and the functions described below provide user-friendly wrappers for the optimization routines `mle` and `optim` provided in the base distribution of `R`.

2 Model overview

The statistical setting and the most important models are summarized in the following paragraphs. For more information, the models and the likelihood functions are detailed in Le Rouzic et al. 2011.

2.1 General framework

The statistcal framework is based on a maximum-likelihood fit of non-linear models. Three variables describing the genetic architecture of a population are tracked during the selection process: the phenotypic mean μ, the additive genetic variance σ^2_A, and the environmental variance σ^2_E. The part of genetic variance that is not additive is thus considered along with environmental variance. Change in the mean is predicted by the Lande equation: $\mu_{t+1} = \mu_t + \beta_t \sigma^2_{A_t}$, where β_t is the selection gradient at generation t. The minimal model features three parameters ($\mathbf{\Theta} = \mu_1, \sigma^2_{A_1}, \sigma^2_{E_1}$), and additional parameters describing changes in σ^2_A and σ^2_E can be considered (see below).

This model predicts, for each generation, the phenoptypic mean of the population (μ_t) and the phenotypic variance ($\sigma^2_{A_t} + \sigma^2_{E_t}$). The vector of parameters $\mathbf{\Theta}$ is then adjusted numerically so that the likelihood of the predicted time series given the observed data (mean and variance at each generation) is minimized.

2.2 Phenomenological models

The first family of models implemented in the package `sra`, termed "phenomenological", aims at describing the changes in the genetic architecture without *a priori* assumptions on the genetic mechanisms that are involved. The model is inspired from an autoregressive setting, such as:

$$\sigma^2_{A_{t+1}} = k_{A_0} + k_{A_1}\sigma^2_{A_t},$$
$$\sigma^2_{E_{t+1}} = k_{E_0} + k_{E_1}\sigma^2_{E_t}.$$

Changing the number of autoregressive terms makes model selection possible, since it is very easy to construct simpler or more complex models (e.g. $\sigma^2_{A_{t+1}} = k_{A_0} + k_{A_1}\sigma^2_{A_t} + k_{A_2}\sigma^2_{A_{t-1}} \dots$). Such high-lag models do not necessarily mean that information is transmitted several generations ahead: additional lags may rather reflect hidden variables that are not accounted for explicitly in the models.

2.3 Mechanistic models

Mechanistic models implement known genetic mechanisms that can affect the genetic architecture during selection response. Indeed, it is unlikely that the additive genetic variance remains constant during long selection expriments. So far, the model can account for:

- The deterministic decrease of genetic variation due to genetic drift (the strength of this decrease depends on the parameter N_e, the effective population size).
- The regular increase of the genetic variance due to mutations, through the mutational variance σ^2_M.
- The influence of linkage disequilibrium generated by the selection (the "Bulmer effect"). No additional parameter is necessary.
- The fact that the number of loci is finite (the number of loci is n).
- The influence of directional epistasis, through the directional epistasis coefficient ε.
- Genetic and environmental canalization (decrease or increase of variance associated to a particular state of the initial population), described by parameters k_g and k_c respectively.
- The fact that natural selection may contradict artificial selection (parameter s, the strength of natural stabilizing selection).

2.4 Scaling issues

Scaling is an important issue in quantitative genetics. The simplest models generally rely on the assumption that alleles combine additively. However, for most traits, this assumption is dubious. Indeed, most traits such as weight or size are on a ratio scale, and are by nature multiplicative. It is thus necessary to change their scale from a multiplicative to an additive scale before fitting additive models, which can be achieved with a logarithm transformation.

However, logarithm transformation does not necessarily address any type of scaling issue. Two additional models are considered here, assuming different constraints on genetic architectures.

One of these models assume that additive and environmental variances change with the mean of the population, i.e. that the ratios $I_A = \sigma_A^2/\mu^2$ and $I_E = \sigma_E^2/\mu^2$ are the quantities that are tracked by the model. These are the "evolvability" models.

The second model assumes that environmental variance scales with the additive variance, so that the "heritability" $h^2 = \sigma_A^2/(\sigma_A^2 + \sigma_E^2)$ becomes the quantity of interest.

2.5 Model selection

One of the strengths of this approach is that models can be compared, so that some hypotheses can be discarded and others further tested. The fit itself (the likelihood of the model) cannot be used as such, because by definition more complex models always fit better due to overparameterization. The likelihood has to be penalized for the number of parameters, the easiest way to proceed being the calculation of the Akaike information criterion $\mathrm{AIC} = -2\log(L) + 2k$, where L is the likelihood of the model and k the number of parameters. Smallest AIC values correspond to the best models.

2.6 Parameter transformation

Most convergence algorithms work better when they are allowed to explore the parameter set freely, and models were implemented with transformed parameters, so that the transformed parameters can take any value between $-\infty$ and $+\infty$. After convergence on the maximum likelihood, parameters have to be back-transformed for interpretation. The two transformations that were used are:

- The logarithm transformation for parameters that should be strictly positive (e.g. additive variance): $Y = \log(X)$, reversed by $X = \exp(Y)$.
- The logistic transformation for parameters that should remain between 0 and 1 (e.g. heritability): $Y = \log(X/(1-X))$, reversed by $X = 1/(1+\exp(-Y))$.

The parameter names in the functions were chosen to reflect the corresponding tranformation.

3 Data

The models implemented in the package `sra` require several sources of data:

- The phenotypic mean of the population each generation, $\bar{y}$,
- The phenotypic variance each generation, s_y^2,
- The phenotypic mean of the selected breeders, $\bar{y}^*$,
- The phenotypic variance among breeders, $s_{y^*}^2$,
- The population size N.

Strincly speaking, only $\bar{y}$, s_y^2, and $\bar{y}^*$ are mandatory. If not provided, $s_{y^*}^2$ is calculated from the other information (mean and variance of the population, as well as the selection intensity), assuming a Gaussian distribution of phenotypes. The population size N is fixed arbitrarily at 100 when not provided (the results should not look too wrong, but confidence intervals will be irrelevant).

The mean of selected individuals is necessary to compute the selection gradient $\beta \simeq (\bar{y} - \bar{y}^*)/s_y^2$. The phenotypic variance of breeders conditions the strength of linkage disequilibrium, and may also be used in other specific models.

Data should be provided as vectors of lenght T, where T is the number of generations. The last element of $\bar{y}^*$ and $s_{y^*}^2$ can be empty (`NA`), because the last generation is not submitted to selection.

The prototype of the function `sraData` is:

```
sraData(phen.mean, phen.var, phen.sel, var.sel=NULL, N=NULL,
        gen=NULL, rep=NULL)
```

The five first arguments correspond to the data itself, and the paramaters `gen` and `rep` allow to specify the generation number and the repetition (in case of several time series). Note that gaps in the time series are not allowed, since the selection gradient must be known for each generation to predict the selection response of the population. The function performs basic checks to ensure that the data is suitable for analysis (no duplicated generations etc).

Data can be provided directly to the `sraData` function, such as in the fictive example below:

```
> dataset <- sraData(
  phen.mean=c(10, 12, 13, 16, 15, 17),
  phen.var =c(9, 11, 8, 14, 15, 13),
  phen.sel =c(14, 15, 16, 18, 18, NA),
  var.sel  =c(6, 6, 7, 8, 8, NA),
  N        =c(40, 38, 36, 41, 42, 35),
  gen      =c(1, 2, 3, 4, 5, 6),
  rep      =c("A", "A", "A", "A", "A", "A"))
```

The resulting object is a data frame of class `"sradata"`, which is recognized as such by the other functions of the package.

The `rep` field can be any character string (it will be interpreted as a factor) identifying unique time series. It is particularly useful when considering multiple time series. Large data sets can easily be imported from a file and provided column by column to the `sraData()` function.

4 Model fitting

4.1 Results of model fitting

4.1.1 Maximum-likelihood estimates and confidence intervals

The basic phenomenological model is implemented in the function `sraAutoreg`.

```
> modelP1 <- sraAutoreg(dataset)
```

The maximum-likelihood estimates of model parameters can be accessed with the `coef` function:

```
> coef(modelP1)
       mu0   logvarA0   logvarE0     logkA1     logkE1
9.98386946 1.51123075 1.54461689 0.03450894 0.15307379
```

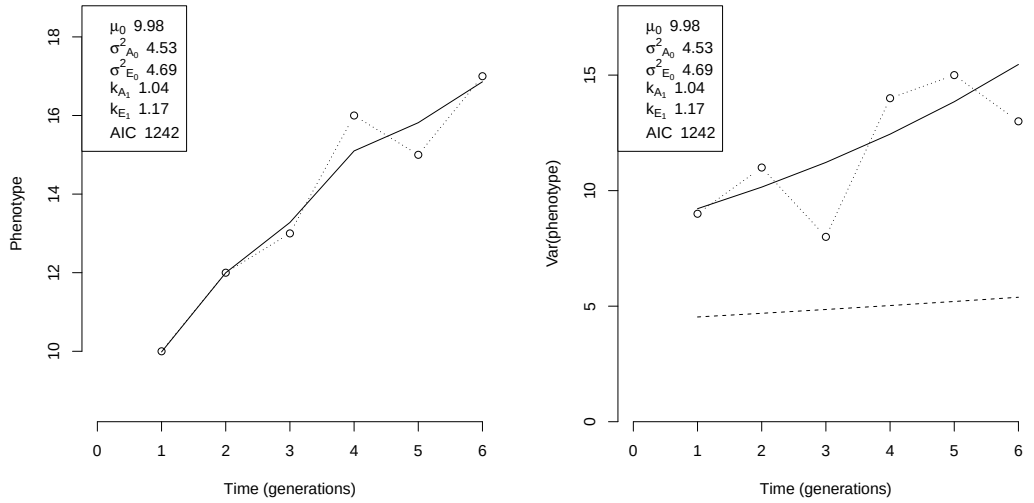

Figure 1: Fit of a simple phenomenological model. The plot on the left represents the phenotypic mean of the population, and on the right the phenotypic variance of the population. Symbols and dotted lines feature the data, and the plain line represents the model prediction. On the variance plot, the hyphenated line is the prediction for the additive variance.

The default model has 5 parameters: the phenotypic mean at the first generation, the initial additive and environmental variances, and the rates at which these variances change. Parameters that cannot be negative are log-transformed for fitting, and need to be back-transformed for interpretation. For instance, the estimate of the additive variance at the first generation is $\hat{\sigma}^2_{A_1} = \exp(1.511) = 4.53$, and its rate of change is $\hat{k}_{A_1} = \exp(0.0345) = 1.04$ (i.e. additive variance increases by 4% per generation).

The confidence intervals of the parameter estimates can be obtained calling `confint`:

```
> confint(modelP1)
                 [,1]       [,2]
mu0        9.08207396 10.8856650
logvarA0   1.08078407  1.9416774
logvarE0   0.84551509  2.2437187
logkA1    -0.22772017  0.2967380
logkE1    -0.09902335  0.4051709
```

The two columns correspond to the edges of the 95% confidence interval. The values also have to be transformed to be interpreted. Note that for performance reasons, confidence intervals are calculated from the diagonal of the variance-covariance matrix, instead of being derived from the profile likelihood.

4.1.2 Graphics

One of the best way to visualize the model is to use the specific `plot` function provided:

```
> plot(modelP1)
> plot(modelP1, var=TRUE)
```

Figure 1 presents the output of the `plot` function. The `var=TRUE` option plots the dynamics of the variance rather than the mean. Note that by default, the legend provides maximum-likelihood estimates of the parameters (on their original scale) as well as the AIC score of the model.

4.1.3 Changing the scale

The previous setting considers the dynamics of genetic architectures from a variance-centered view. Alternatives include:

- Log transformation

```
> coef(sraAutoregLog(dataset))
        mu0    logvarA0    logvarE0      logkA1      logkE1
 2.25629393 -3.07732727 -3.38491799 -0.25958069  0.02900387
Warning message:
In sraData(rep = sradata$rep, gen = sradata$gen, phen.mean = 2 *  :
  The phenotypic variance of breeders was not provided.
 An educated guess will be made, model fitting might be inaccurate.
```

  The parameters describe the dynamics of the log-transformed data set. The warning message is due to the fact that the variance of the breeders had to be recomputed based on a log-normality assumption (and truncation selection), because it cannot be directly deduced from the original data. Note that very similar results would have been obtained if the standard model was fed with mean and variances of log-transformed measurements. This, however, makes model selection impossible (the data set being different, AIC differences become irrelevant).

- Models focusing on mean-scaled variances ("evolvability")

```
> coef(sraAutoregEvolv(dataset))
        mu0      logIA0      logIE0      logkA1      logkE1
10.02200806 -3.15298944 -3.20918300 -0.21295393 -0.02444584
```

  I_A and I_E are respectively σ_A^2/μ^2 and σ_E^2/μ^2. k_A and k_E describe the dynamics of I_A and I_E in the same way as for the variances, i.e. $I_{A_{t+1}} = k_{A_0} + k_{A_1} I_{A_t} + k_{A_2} I_{A_{t-1}} + \ldots$.

- Models focusing on heritability

```
> coef(sraAutoregHerit(dataset))
        mu0    logith20    logvarP0      logkA1      logkE1
 9.96783661  0.01199777  2.21188316 -0.07920105  0.10467725
```

  In this setting, k_A parameters describe the changes in $h^2 = \sigma_A^2/(\sigma_A^2 + \sigma_E^2)$, and k_E describes the changes in $\sigma_P^2 = \sigma_A^2 + \sigma_E^2$.

4.2 Building more complex phenomenological models

4.2.1 Increasing complexity

The complexity of the model can be adjusted with the `active` option, by providing a vector of four booleans representing the "activity" of the four first lag parameters. For instance, the default model assumes that `active=c(FALSE, TRUE, FALSE, FALSE)`, i.e. that only lag 1 parameters (k_{A_1} and k_{E_1} are active. To activate both lags 0 and 1, the command is:

```
> modelP2 <- sraAutoreg(dataset, active=c(TRUE,TRUE,FALSE,FALSE))
> coef(modelP2)
        mu0    logvarA0    logvarE0 relativekA0 relativekE0      logkA1
 9.97687018  1.51483260  1.46671095  0.25755633  0.31564349 -0.24700546
     logkE1
-0.04102936
```

Note that for scaling reasons, k_{A_0} and k_{E_0} are expressed relative to $\sigma^2_{A_1}$ and $\sigma^2_{E_1}$ respectively.

Fine parameterization can be achieved by fixing some parameters using the `fixed` option, similar to the one provided by `mle`. For instance, one can force additive variance to be constant while environmental variance may vary:

```
> coef(sraAutoreg(dataset, active=c(TRUE,TRUE,FALSE,FALSE),
       fixed=list(relativekA0=0, logkA1=log(1))))
        mu0    logvarA0    logvarE0 relativekE0      logkE1
 9.92892263  1.56043513  1.42086473  0.37432962 -0.05101302
```

By default, k_A and k_E parameters are forced to be > 0. Allowing them to be negative allows more flexibility, including the possibility to describe cyclic behaviors, at the cost of a more complex likelihood function (and thus potential convergence issues).

```
> coef(sraAutoreg(dataset, active=c(TRUE,TRUE,FALSE,FALSE),
       negative.k=TRUE, rep=10))
        mu0    logvarA0    logvarE0 relativekA0 relativekE0         kA1
  10.054805    1.570043    1.384323    2.233791    3.970297   -1.693604
        kE1
  -1.103415
```

The option `negative.k=TRUE` changes the parameterization of the model and fits the model on e.g. `kA1` and `kE1` instead of `logkA1` and `logkE1`. `rep=10` is a way to increase the number of trials to find the maximum likelihood, reducing convergence problems (see next paragraph).

4.2.2 Dealing with convergence issues

The most complex models may generate convergence issues:

```
> confint(sraAutoreg(dataset, active=c(TRUE,TRUE,TRUE,FALSE)))
                     [,1]      [,2]
mu0             9.0524665 10.900256
logvarA0        0.9502235  2.070951
logvarE0        0.5020707  2.423552
relativekA0           NaN       NaN
logkA1                NaN       NaN
logkA2                NaN       NaN
relativekE0    -1.9226011  2.508723
logkE1         -3.6083929  3.504780
logkE2       -102.5519053 96.005756
Warning messages:
1: In sqrt(diag(object@vcov)) : NaNs produced
2: In sqrt(diag(object@vcov)) : NaNs produced
3: In sqrt(diag(object@vcov)) : NaNs produced
4: In sqrt(diag(object@vcov)) : NaNs produced
```

The warnings indicate that the `optim` routine did not manage to converge towards a maximum of the likelihood function, since the second derivative is negative for some parameters. This is likely to be due to the convergence towards a local maximum or a saddle point on a complex and rugged likelihood function. The best way to solve such a problem is to provide good stating values for every parameter (through the `start` option). However, in some cases, this may be quite difficult, and the `sra` package provides an algorithm that can generate random vectors of starting values, and pick the best model out of several convergence trials. For instance, the following is the best amongst `rep=100` convergence attemps.

```
> confint(sraAutoreg(dataset, active=c(TRUE,TRUE,TRUE,FALSE), rep=100))
                   [,1]      [,2]
mu0            9.074815 10.860446
logvarA0       1.062664  1.983493
logvarE0       0.357098  2.393155
relativekA0   -6.412921  6.443444
logkA1        -6.781094  6.735324
logkA2      -103.697783 96.814506
relativekE0   -2.786469  4.044168
logkE1       -34.706981 28.406871
logkE2        -2.068047  1.826611
```

The `rand` option can be adjusted to allow more or less randomness in the starting values. Although convenient, this approach remains problematic for several reasons, in particular because fitting the same model several times can be quite time consuming, and because one cannot be sure that the resulting model is really the maximum-likeihood (the real maximum can still be in a unexplored region of the parameter set).

4.3 Fitting mechanistic models

Mechanistic models implement quantitative genetics models and attempt at estimating the value of parameters of interest.

The simplest mechanistic model is the constant-variance model, which is generally associated to an infinitesimal genetic architecture (infinite number of loci of very small effect). The corresponding function is `sraCstvar` and is used in the same way as for the phenomenological approach:

```
> coef(sraCstvar(dataset))
     mu0 logvarA0 logvarE0
9.863218 1.627514 2.012143
```

The results of this model do not coincide exactly with the corresponding phenomenological model, because mechanistic models include by default the expected effect of linkage disequilibrium due to selection (Bulmer effect). This effect can be disabled easily, falling back to the result of the phenomenological model with constant variances:

```
> coef(sraCstvar(dataset, Bulmer=FALSE))
     mu0 logvarA0 logvarE0
9.919126 1.561737 1.982280
> coef(sraAutoreg(dataset, active=c(FALSE,FALSE,FALSE,FALSE)))
     mu0 logvarA0 logvarE0
9.919126 1.561737 1.982280
```

Alternative mechanistic models can include genetic drift, mutational variance, or directional epistasis:

```
> coef(sraDrift(dataset))
     mu0 logvarA0 logvarE0    logNe
9.863321 1.627620 2.012259 8.353398
> coef(sraMutation(dataset, start=list(logvarM=log(0.1))))
        mu0    logvarA0    logvarE0       logNe     logvarM
10.17368391  1.32857000  1.85082825  5.03612374 -0.08604935
> coef(sraDirepistasis(dataset, fixed=list(logNe=log(10))))
      mu0   logvarA0   logvarE0 logepsilon
10.177836   1.326353   1.801145  -2.646717
> coef(sraDrift(dataset, start=list(logn=0),
       fixed=list(logNe=log(1000))))
     mu0 logvarA0 logvarE0     logn
9.862883 1.628056 2.012711 6.168046
> coef(sraCanalization(dataset))
       mu0   logvarA0   logvarE0         kc         kg
9.94813929 1.54815006 1.43875897 0.12330431 0.03741515
> coef(sraSelection(dataset))
       mu0    logvarA0    logvarE0       logNe           s           o
9.87727530  1.93290039  1.79771914 11.29513495 -0.01292434 18.69352316
```

This illustrates how starting and fixed values can be used to help model convergence and customize the parameter set. Indeed, it is sometimes difficult to reach convergence when two parameters having a similar effect on the dynamics are activated at the same time. This is particularly true for N_e, the effective size of the population, and n, the effective number of loci. The meaning of the different parameters is detailed in Le Rouzic et al. 2011.

Note that all the mechanistic model functions (except `sraDirEpistasis`) are based on the same underlying model, and can thus be interchanged provided that the proper combination of parameters is activated:

```
> coef(sraCanalization(dataset))
       mu0   logvarA0   logvarE0         kc         kg
9.94813929 1.54815006 1.43875897 0.12330431 0.03741515
> coef(sraDrift(dataset, start=list(kc=0, kg=0), fixed=list(logNe=1e10)))
       mu0   logvarA0   logvarE0         kc         kg
9.94813929 1.54815006 1.43875897 0.12330431 0.03741515
```

The set of active parameters for each function is detailed in the table 1.

4.4 Model selection

Model selection consists in sorting different models according to their relevance, and thus defining the ones that describe the best the data out of a set of potential models. Model selection has to consider both the fit of the model, which can be quantified by its likelihood, and the possibility of overparameterization. Indeed, the likelihood of the models increases when adding parameters:

```
> logLik(sraAutoreg(dataset, active=c(FALSE,FALSE,FALSE,FALSE)))
'log Lik.' -617.6975 (df=3)
> logLik(sraAutoreg(dataset, active=c(TRUE,FALSE,FALSE,FALSE)))
'log Lik.' -615.8893 (df=5)
> logLik(sraAutoreg(dataset, active=c(FALSE,TRUE,FALSE,FALSE)))
'log Lik.' -615.9386 (df=5)
> logLik(sraAutoreg(dataset, active=c(TRUE,TRUE,FALSE,FALSE)))
'log Lik.' -615.883 (df=7)
```

		Code	C	D	M	Cn	Cno	S	De
Initial phenotypic mean	μ_1	`mu0`	✔	✔	✔	✔	✔	✔	✔
Initial additive variance	$\sigma^2_{A_1}$	`logvarA0`	✔	✔	✔	✔	✔	✔	✔
Initial environmental variance	$\sigma^2_{E_1}$	`logvarE0`	✔	✔	✔	✔	✔	✔	✔
Effective population size	N_e	`logNe`		✔	✔	✔			✔
Effective number of loci	n_e	`logn`							✘
Mutational variance	σ^2_M	`logvarM`			✔				
Environmental canalization strength	k_c	`kc`				✔	✔		✘
Genetic canalization strength	k_g	`kg`				✔	✔		✘
Optimum phenotype	o	`o`				=`mu0`[1]	✔	=`mu0`[1]	✘
Selection strength	s	`s`						✔	✘
Directional epistasis	ε	`logepsilon`[2]	✘	✘	✘	✘	✘	✘	✔

Table 1: Availability of the parameters and default setting for the seven default mechanistic model functions. Checks (✔) mark default parameters, crosses (✘) denote unavailable parameters. Blanks mean that the parameter can be activated by the user. Function codes: C: `sraCstvar`, D: `sraDrift`, M: `sraMutation`, Cn: `sraCanalization`, Cno: `sraCanalizationOpt`, S: `sraSelection`, De `sraDirEpistasis`. [1]: If unspecified, the optimum is considered as the same as the initial phenotypic mean. [2] Two models are actually fitted for directional epistasis: one model with a positive directionality (`logepsilon`) and another with a negative directionality (`minuslogepsilon`). The best model is then determined according the the likelihood and only one of these parameters is returned.

However, this does not necessarily mean that more complex models are better, since adding parameters also increases the risk of overparameterization. Comparing the AIC gives some hints on whether or not it is worth including additional parameters:

```
> AIC(sraAutoreg(dataset, active=c(FALSE,FALSE,FALSE,FALSE)))
[1] 1241.395
> AIC(sraAutoreg(dataset, active=c(TRUE,FALSE,FALSE,FALSE)))
[1] 1241.779
> AIC(sraAutoreg(dataset, active=c(FALSE,TRUE,FALSE,FALSE)))
[1] 1241.877
> AIC(sraAutoreg(dataset, active=c(TRUE,TRUE,FALSE,FALSE)))
[1] 1245.766
```

The lowest AIC remains the one in which variances do not change (AIC=1241.395), but the three first models remain very close in term of explanatory power (AIC differences above 2 units correspond to p-values below 0.05). Although the previous examples are based on phenomenological models, it is possible to perform model selection among mechanistic models as well, based on the same procedure.

5 Getting involved

5.1 Questions and bug reports

General questions about `R` and the base packages can be asked on various mailing lists, such as R-help (`https://stat.ethz.ch/mailman/listinfo/r-help`).

Specific questions about the SRA model and its applications to data should be sent to the corresponding authors of the papers cited in the introduction.

Questions, bug reports, and suggestions about the `sra` package, the implementation of the functions, and the official documentation should be sent to the official maintainers of the package (type `?sra` in `R` after having loaded the `sra` package).

Remarks or comments about this tutorial should be sent to the author, Arnaud Le Rouzic <`lerouzic@legs.cnrs-gif.fr`>.

5.2 Contribution

The `sra` package is released under a free (GLP-2) license and it is possible to install, study, execute, modify and publish the modifications as long as the GPL-2 license is respected (in brief: cite the authors and keep the license unchanged in the modified versions, read the full conditions for more details `http://www.gnu.org/licenses/gpl-2.0.html`).

Contributions, in terms of suggestions, bug reports, bug fixes, and new features are welcome.

Acknowledgements

Many thanks to Thomas Hansen and David Houle for their contribution to this project. I also acknowledge Hans J. Skaug and José M. Álvarez-Castro for helpful discussion.