Instruction to Evolutionary Constructor script language

The software package “Evolutionary constructor” (EC) provides modeling scenarios with the use of the special script language.

Any scenario of EC model consists of *two consequent sections*:

1) *Model declaration section;*
2) *Calculation section.*

The comment lines may begin with the following symbols: `//` (C-style), `!` (Fortran-style), and `%` (Matlab-style). The script language is *case insensitive*.

Model declaration section

This section starts with the `declare` statement and ends with the `end declare` statement. The other statements setting up model parameters in the declaration section are shown in the table below (synonyms are also shown if available):

Table 1. Model declaration statements

Statement	Description/model parameter set up	Example
`volume`	Environmental volume	`volume = 1e-1`
`flow`	Flow rate (from 0 to 1.0)	`flow = 0.1`
`nonspec (ns)`	Number of non-specific substrates in the model	`nonspec = 2`
`spec (ss)`	Number of specific substrates in the model	`spec = 3`
`init_ns (substrates_ns)`	Initial concentrations of non-specific substrates in environment (list)	`init_ns = 1.5e-3, 2e-3`
`init_ss (substrates_ss)`	Initial concentrations of specific substrates in environment (list of comma-separated values)	`init_ss = 0.1,0.2,0.3`
`inflow_ns (comsub)`	Concentration of non-specific substrates in the in-flow (list of comma-separated pairs – *substrate number:substrate concentration*)	`inflow_ss = 1:0.1, 4:0.2`
`mut_probability`	Mutation probability	`mut_prob = 1e-8`

`(mut_prob)`		
`hgt_probability` `(hgt_prob)`	Horizontal gene transfer probability	`hgt_prob = 1e-10`
`population` `(pop, p)`	Opens the population declaration section (see Table 2)	`pop = 1`
`end population` `(end pop, end p)`	Closes the population declaration section (see Table 2)	`end pop`

Each population in the declaration section is given by the particular subsection. That subsection is opened with `population` (`pop`, `p`) statement, and is closed with `end_population` (`end pop`, `end p`) statement. The only parameter of `pop` statement is the number of population. The statements setting up population's parameters are listed in the Table 2:

Table 2. Population declaration statements

Statement	Description/model parameter set up	Example
`size`	Initial size of a population	`size = 4e+8`

`increaser`
`(strategy,`
`trophic)`
	Trophic strategy of a population (argument is the name of strategy, for example, `rubel`, `liebig`, `inhibitory`, `chaotic` etc.)	`increaser = rubel`

`k_death`	Death parameter	`k_death = 1e-30`
`k_flow`	Flow parameter	`k_flow = 0.05`
`cprod`	Synthesis ratio for specific products	`cprod = 1e+5`
`ccons_nsp` `(ccons_ns)`	Utilization ratio for non-specific substrates	`ccons_nsp = 1e+7`
`ccons_sp` `(ccons_ss)`	Utilization ratio for specific substrates	`ccons_sp = 1e+5`

`genes_n`	Adds a new gene of non-specific substrate utilization to the generalized genome of a population. Arguments: *substrate number : list of alleles values : list of alleles frequencies in a population*	`genes_n = 1;`
		`allele_values:3,5,6;`
		`allele_concentrations:`
		`0.1,0.6,0.3`

`genes_s`	Adds a new gene of specific substrate utilization to the generalized genome of a population. Arguments: *substrate number : list of alleles values : list of alleles frequencies in a population*	`genes_s = 2;`
		`allele_values:3,5,6;`
		`allele_concentrations:`
		`0.1,0.6,0.3`

`genes_p`	Adds a new gene of specific substrate synthesis to the generalized genome of a population. Arguments: *substrate number : list of alleles values : list of alleles frequencies in a population*	`genes_p = 3;`
		`allele_values:3,5,6;`
		`allele_concentrations:`
		`0.1,0.6,0.3`

The actual values of traits determining corresponding utilization rates, i.e. r_i and c_i, (see Trophic strategies description) are calculated as a quotient of the following parameters: fixed component – *utilization ratio* (`ccons_ns`, `ccons_ss`) and variable components taken from `allele_values` list in `genes_n` and `genes_s` statements.

The actual values of traits determining corresponding synthesis rates, i.e. d_i (see Metabolism simulation description) are calculated as a product of analogical parameters: fixed component – *synthesis ratio* (`cprod`), and variable one set up with `genes_p` statement.

The example below demonstrates the description of a monomorphic (with relation to utilization of the substrate S1) population, the actual c_1 parameter value of which is equal to $2{\cdot}10^{-5}$ (it means that one cell division takes $2{\cdot}10^{5}$ of substrate molecules):

```
pop = 1
     ccons_sp = 1e+5
     genes_s=1;allele_values:2;allele_concentrations:1.0
     // Other parameters of the population
end pop
```

Model calculation section

Calculation section starts immediately after `end declare` statement. The statements of this section are listed in the table below:

Table 3 Model calculation statements

Statement	Description/model parameter set up	Example
`iterate`	Iterate *n* simulation steps (*n* is argument). Stochastic operations (mutations, horizontal transfer etc.) are not permitted.	`iterate = 400`
`stochastic` `(s_iterate)`	Iterate *n* simulation steps (*n* is argument). Stochastic operations are permitted – possible events are regulated by corresponding probabilities.	`stochastic = 400`
`mutation` `(mut)`	Perform mutation in one (or more) cells in a population. Detailed description of this statement is in the text below.	`Description is in the text below`
`horizontal_transfer` `(ht, hgt)`	Perform horizontal transfer of a gene from a cell of donor population into a cell of acceptor population. Detailed description of this statement is in the text below.	`Description is in the text below`
`set_nssc` `(set_ns)`	Change the concentration of non-specific substrate in the in-flow. Arguments: *substrate number: new concentration*	`set_ns=1 : 0.5`
`set_flow` `(set_f)`	Change flow rate (the argument range is from 0 to 1)	`set_flow = 0.2`
`buffer_on`	Fix concentration(s) of some substrate(s) in environment on a certain level regardless of flow or populations action. Arguments: *substrate type* (possible values N,	`buffer_on=S:1:0.7`

	S): *substrate number* : *concentration*	
`buffer_off`	Cancel `buffer_on` command. Arguments: *substrate type* (possible values N, S): *substrate number*	`buffer_off=S:1`

`ns_pattern`	Set series of in-flow concentrations for a certain non-specific substrate alternating in time periods, and durations of those periods. Arguments: *substrate number; list of concentrations; list of duration periods*	`ns_pattern = 1;`
		`concentrations:0.1,2,0;`
		`durations:10,20,300`

Mutation statement

The statement `mutation` (`mut`) causes the mutation in cell(s) of a population:

```
mutation= pop:1; gene_type:n; sub_num:0; from:1.0; to:7.0; count:0.01
```

where

`pop` – population number;

`gene_type` – type of a gene to be mutated (possible values: *n, s, p*);

`sub_num` – number of substrate corresponding with the mutant gene;

`from` – initial trait value (only cells having trait of this particular value will mutate);

`to` – mutant trait value (mutant cells will have this trait value);

`count` (unnecessary parameter) – fraction of mutants in a whole population. If this parameter missed, the fraction is chosen automatically to get the minimal size of the mutant subpopulation (acceptable with regard to computational accuracy or special parameter).

Horizontal_transfer statement

The statement `horizontal_transfer` (`ht, hgt`) causes the horizontal transfer of gene(s) from a cell of donor population into a cell of acceptor population:

```
hgt=acceptor:2; donor:1; gene_type:n; sub_num:1,2; gene_type:s; sub_num:1,3,4,..; p_size:1e+1; allele:greatest|lowest:1
```

where

`acceptor` – acceptor population number;

`donor` – donor population number;

`gene_type` – type of transferred gene(s) (possible values: *n, s, p*);

`sub_num` – list of numbers of transferred genes;

`p_size` (unnecessary parameter) – subpopulation size (number of cells incurred horizontal gene transfer). If this parameter missed, the automatically set size is the minimal with regard to computational accuracy or special parameter;

`allele` (unnecessary parameter) – set the *rule* of allele *selection* (for the transfer) and the number of allele in cell(s) (in the order defined with selection rule) of the donor population. Possible values for selection rule: *greatest* or *lowest,* determine, whether maximal or minimum trait-valued alleles will be transferred. If this parameter missed, the most presented (in the donor population) allele will be selected for transfer.

Note: it is possible to transfer genes of all types simultaneously per one horizontal gene transfer. To achieve this it is necessary to specify lists of transferred genes in such a manner: point out gene type followed by list of genes numbers, then another gene type followed by list of genes numbers etc (see the example above).