**ec.simple.SimpleEvolutionState** holds the complete state of an evolution run. Performs a simple breed-evaluate loop on individuals in its population. Every object stored within this object is serializable, and so can be written out to a checkpoint file using **ec.util.Checkpoint**. The evolution state is also passed around to among objects a lot so they can access its random number generators, parameter database, and logs. This is done, of course, in a threadsafe manner.
The prototypical data object, copied once per thread, each time a generation of individuals needs to be evaluated, and passed between ec.gp.GPNodes to transfer data between then when they are executing. For Symbolic Regression, this object holds the floating-point value returned by functions.

A prototypical ADF Stack object, copied once per thread, each time a generation of individuals needs to be evaluated. ADF Stacks holds current contextual state of ADF function calls in two special stacks of ec.gp.ADFContext objects, one ADF Context per pending ADF function call. A prototypical ADF Context object is copied when new ADFContext objects are needed; old ADFContext objects are held in a Reserve to prevent excessive object construction. For more information on this complicated mechanism, see the class documentation. Since Symbolic regression doesn’t have ADFs, this facility is left unused.

An ADF Context holds the current execution context of an ADF function call. The ADF ("Automatically Defined Function", see Koza-I and Koza-II) called is stored in the context, as is the returned data from its children nodes. A prototypical data object is also held from which new children results are created if necessary. This data object is typically the same kind of prototype as the one used in the problem, in this case, ec.app.regression.Regression. ADF Contexts are also used for ADMs ("Automatically Defined Macros"). Since Symbolic Regression doesn’t have ADFs, this facility is left unused.
During population initialization, the number of times to retry generation of an individual if it is a duplicate of one already in the Subpopulation.
A table of GPFuncInfo objects, each containing a GPNode. These GPNodes are the prototypical nodes for each function (nonterminal or terminal) in the function set. In Symbolic Regression, this includes nodes for Sin, Cos, X, +, -, /, *, Log, and ERCs. These are stored in `ec.app.regression.func.*`.

- **Function Set Name**
- **Type name**
- **Type index**
- **Constraints name**
- **Constraints index**
- **Return type**
- **Argument types**
- **Probability of being picked as the parent’s breeding source**
- **Probability of being picked as the parent’s breeding source**
- **Tournament size**
- **Tournamentsize**
- **Array of sources feeding the pipeline (actually one source)**
- **How many times to try to breed a valid individual before giving up and just copying the old one.**
- **The maximum valid depth of a newly-bred individual**