Grammatical Evolution (GE) is a population-based evolutionary algorithm, where a BNF-style [[grammar|Grammars]] is used in the [[genotype-to-phenotype mapping process|Representation#genotype-phenotype-mapping-process]].

PonyGE2 is an implementation of GE in Python. It's intended as an advertisement and a starting-point for those new to GE, a reference for students and researchers, a rapid-prototyping medium for our own experiments, and as a Python workout.

The original version of PonyGE (https://github.com/jmmcd/ponyge) was originally designed to be a small single-file implementation of GE. However, over time this has grown to a stage where a more formal structured approach was needed. This has led to the development of PonyGE2 (https://github.com/PonyGE/PonyGE2), presented here.

A technical paper describing PonyGE2 has been published and been made freely available on arXiv at:

https://arxiv.org/abs/1703.08535

PonyGE2 can be referenced using the following citation:

```
Fenton, M., McDermott, J., Fagan, D., Forstenlechner, S., Hemberg, E., and
O'Neill, M. PonyGE2: Grammatical Evolution in Python. arXiv preprint,
arXiv:1703.08535, 2017.
```

The PonyGE2 development team can be contacted via GitHub.

PonyGE2 was reviewed in GPEM Volume 22:383-385 by Tuong Manh Vu. Thanks for the review!

Requirements

PonyGE2 requires Python 3.5 or higher. Using matplotlib, numpy, scipy, scikit-learn (sklearn), pandas.

All requirements can be satisfied with Anaconda.

Running PonyGE2

We don't provide any setup script. You can run an example problem (the default is [[regression|Example-Problems#regression]]) just by typing:

```
$ cd src
$ python ponyge.py
```

This will run an example problem and generate a results folder. The folder contains several files showing the run's stats, producing graphs and documenting the parameters used, as well as a file detailing the best individual. For a more verbose command line experience run the following:

```
$ cd src
$ python ponyge.py --verbose
```

Each line of the verbose output corresponds to a generation in the evolution, and prints out all statistics on the current run (only if `--verbose` is specified). Upon completion of a run, the best individual is printed to the command line, along with summary statistics.

There are a number of arguments that can be used for passing values via the command-line. To see a full list of these just run the following:

```
$ python ponyge.py --help
```

About PonyGE2

Grammatical Evolution (GE) [[[O'Neill & Ryan, 2003|References]]] is a grammar-based form of Genetic Programming [[[Koza, 1992|References]]]. It marries principles from molecular biology to the representational power of formal grammars. GE's rich modularity gives a unique flexibility, making it possible to use alternative search strategies, whether evolutionary, deterministic or some other approach, and to radically change its behaviour by merely changing the grammar supplied. As a grammar is used to describe the structures that are generated by GE, it is trivial to modify the output structures by simply editing the plain text grammar. This is one of the main advantages that makes the GE approach so attractive. The genotype-phenotype mapping also means that instead of operating exclusively on solution trees, as in standard GP, GE allows search operators to be performed on the genotype (e.g., integer or binary chromosomes), in addition to partially derived phenotypes, and the fully formed phenotypic derivation trees themselves.

PonyGE2 is primarily a Python implementation of canonical Grammatical Evolution, but it also includes a number of other popular techniques and EC aspects.