Introduction to the CGP-Library

Andrew James Turner

Overview

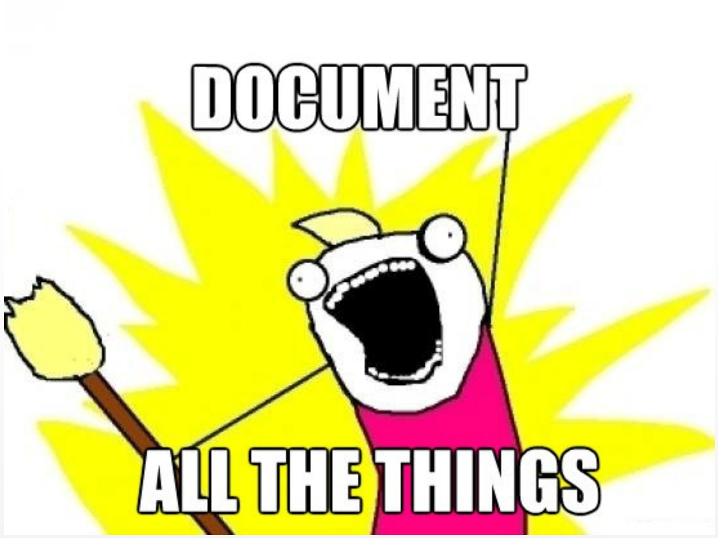
- Cross Platform Library
- Open Source (GNU LGPL)
- Well Documented
- Simple to use
- Simple to modify / extend
- Cartesian Genetic Programing
- Recurrent Cartesian Genetic Programing
- NeuroEvolution
- Visualization tools

What it is NOT

- It is **NOT** a stand-alone program
  - It is a library, like math.h
  - It is used **BY** other programs

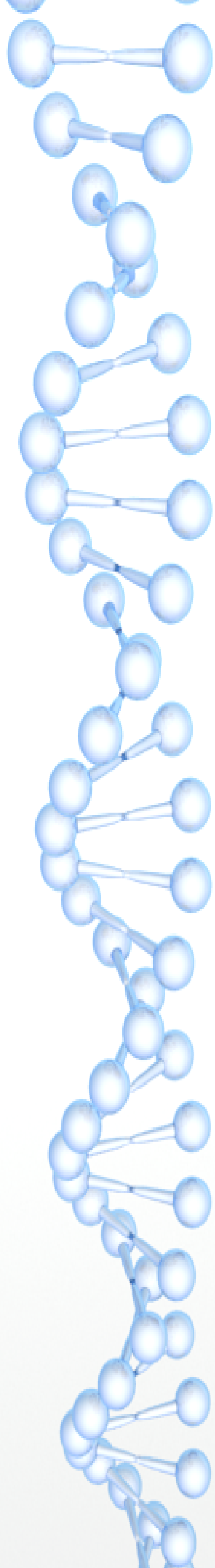
Documentation

- http://cgplibrary.co.uk/
- Full API description
- 10+ tutorials/examples
- Quick introduction to CGP



Installation

- As a system-wide library
  - Windows
    - 32bit, Copy CGP-Library.dll --> C:\Windows\System32
    - 64bit, Copy CGP-Library.dll --> C:\Windows\SysWOW64
  - Linux
    - Copy libcgp.so --> /usr/lib/
    - Copy cgp.h --> /usr/include/
- Compile into executable / binary
  - Place cgp.c and cgp.h in build path
  - #define NO_DILL (windows only)



Basic Use

```c
#include <stdio.h>
#include "../src/cgp.h"

int main(void){

    struct parameters *params = NULL;
    struct dataSet *trainingData = NULL;
    struct chromosome *chromo = NULL;

    int numInputs = 1;
    int numNodes = 15;
    int numOutputs = 1;
    int nodeArity = 2;

    int numGens = 10000;

    params = initialiseParameters(numInputs, numNodes, numOutputs, nodeArity);

    addNodeFunction(params, "add,sub,mul,div,sin");

    trainingData = initialiseDataSetFromFile("./dataSets/symbolic.data");

    chromo = runCGP(params, trainingData, numGens);

    printChromosome(chromo, 0);

    freeDataSet(trainingData);
    freeChromosome(chromo);
    freeParameters(params);

    return 0;
}
```

DataSet

3 Bit Even Parity

```
3,1,8
0,0,0,0
0,0,1,1
0,1,0,1
0,1,1,0
1,0,0,1
1,0,1,0
1,1,0,0
1,1,1,1
```

Inputs, Outputs, Samples

Input_0, input_1, input_2, output_0

Defaults

Parameter	Value
Mu (μ)	1
Lambda (λ)	4
Evolutionary Strategy	(μ+λ)-ES
Mutation Type	Probabilistic
Mutation Rate	0.05 (5%)
Selection Scheme	Select Fittest
Reproduction Scheme	Mutate Random Parent
Fitness Function	Supervised Learning (ABS error)
Recurrent Connection Probability	0.00 (0%)
Connection Weight Range	1.00 (i.e. +/- 1)
Update Frequency	1

Changing Defaults

- setMu(...)
- setLambda(...)
- setEvolutionaryStrategy(...)
- setMutationRate(...)
- setMutationType(...)
- ...

Node Types

- addNodeFunction(...)

add	AND
sub	NAND
mul	OR
div	NOR
abs	XOR
sqrt	XNOR
sq	NOT
cube	sig
pow	gauss
exp	step
sin	softsign
cos	tanh
tan	wire

Mutation Types

- Probabilistic
  - Mutates each chromosome gene with a given probability
- Point
  - Always mutates the same percentage of randomly selected genes
- OnlyActive
  - Conducts probabilistic mutation on active nodes only
- Single
  - Keeps mutating randomly selected genes until an active gene is mutated to a new allele

Custom Evolutionary Stages

- setCustomFitnessFunction(...)
- setCustomSelectionScheme(...)
- setCustomReproductionScheme(...)
- addCustomNodeFunction(...)

- This is quite simple to do!
  - Tutorials provided

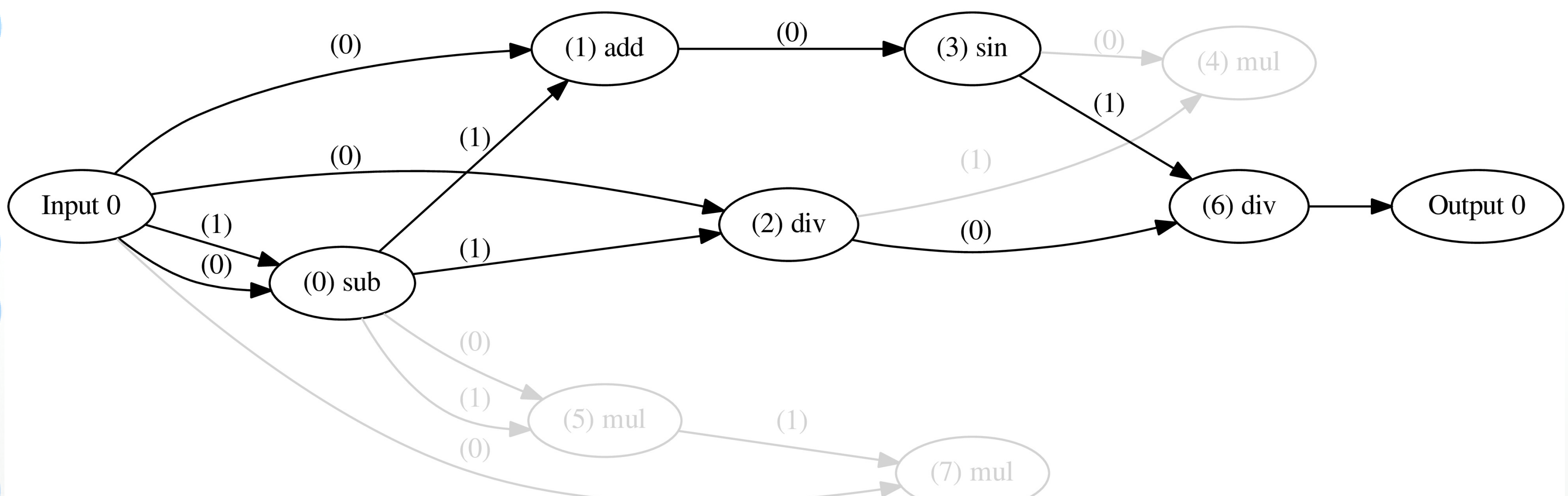
Visualization

- printChromosome(...)
  - Displays in terminal

```
(0):   input
(1):   sub   0 0   *
(2):   add   0 1   *
(3):   div   0 1   *
(4):   sin   2     *
(5):   mul   4 3
(6):   mul   1 1
(7):   div   3 4   *
(8):   mul   0 6
outputs: 7
```

Visualization

- saveChromosomeDot(...)
  - Produces .dot file
  - Used by Graphviz (www.graphviz.org)



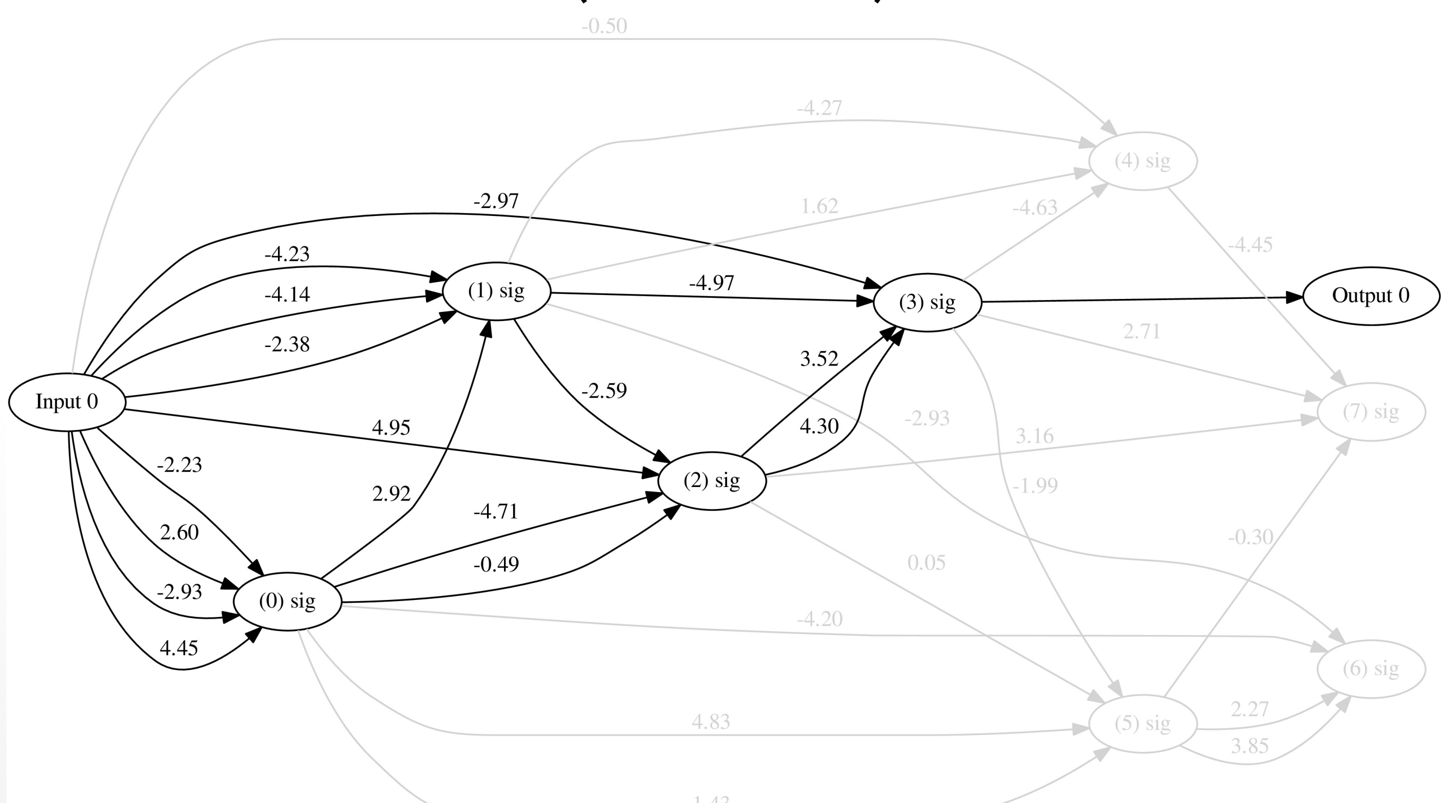
Visualization

- saveChromosomeLatex(...)
  - Produces .tex file
  - Used by LaTeX (or pdfLaTeX)

$$f_0(x_0) = \frac{\frac{x_0}{(x_0 - x_0)}}{\sin((x_0 + (x_0 - x_0)))}$$

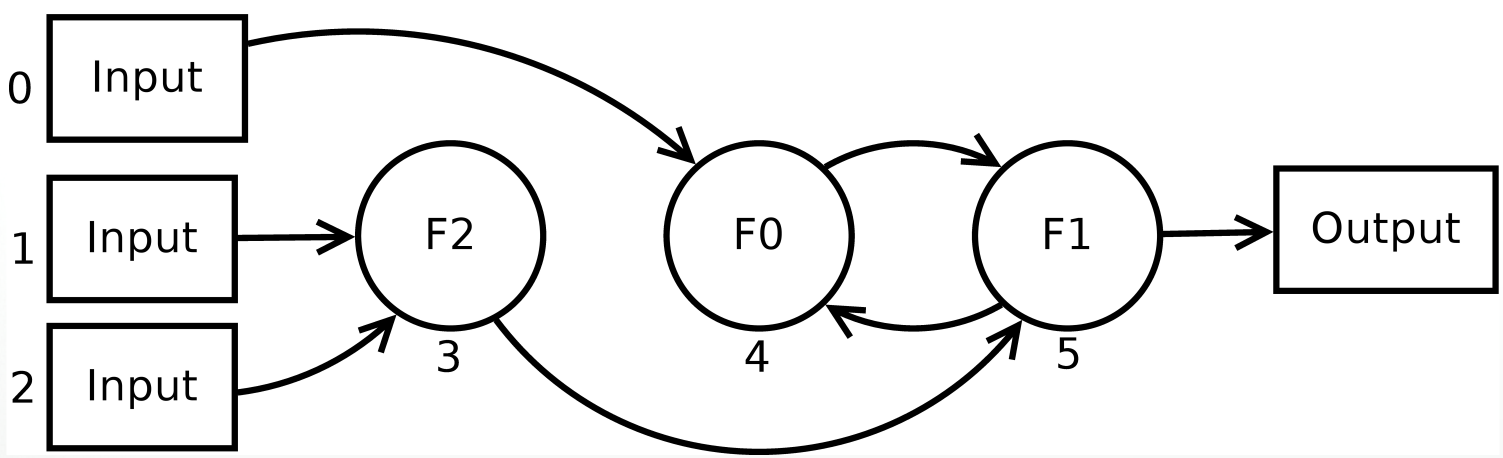
NeuroEvolution

- Applications of Evolutionary Computation to Artificial Neural Networks
- Cartesian Genetic Programming of Artificial Neural Networks (CGPANN)



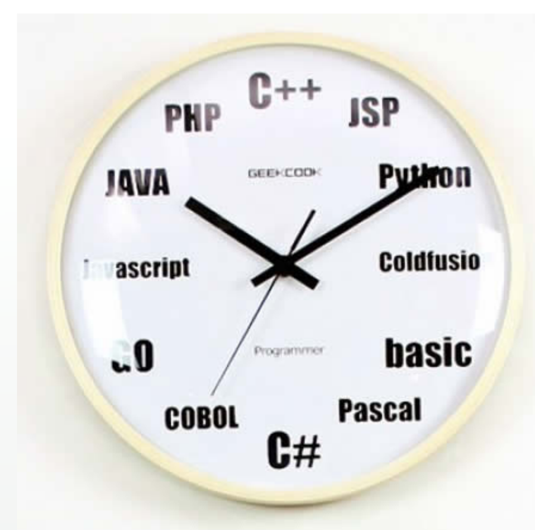
Recurrent Cartesian Genetic Programming

- Allows for feed-back / recurrent / cyclic connections
- Enables CGP to have memory / states
- Also works with NeuroEvolution



Language Bindings

- C / C++
- Python soon!
- Possible other languages on request:
  - **Java**, **C#**, D, Go, Octave, R, Javascript, Perl, PHP, Ruby, Lisp, Lua, Modula-3, OCAML ...



References

- A. J. Turner and J. F. Miller. **Introducing A Cross Platform Open Source Cartesian Genetic Programming Library**. The Journal of Genetic Programming and Evolvable Machines, to appear
- Miller, J.F.: **Cartesian Genetic Programming**. Springer (2011)
- Turner, A.J., Miller, J.F.: **Cartesian Genetic Programming encoded Artificial Neural Networks: A Comparison using Three Benchmarks**. In: Proceedings of the Conference on Genetic and Evolutionary Computation (GECCO'13), pp.1005-1012 (2013)
- Turner, A.J., Miller, J.F.: **Recurrent Cartesian Genetic Programming**. In:PPSN'14. pp.476-486 (2014)
- Goldman, B.W., Punch, W.F.: **Reducing Wasted Evaluations in Cartesian Genetic Programming**. In: Proceedings of the 16th European Conference on Genetic Programming (EuroGP'13). vol.7831, pp.61-72. Springer Verlag (2013)