Project 2

Alex Cochrane

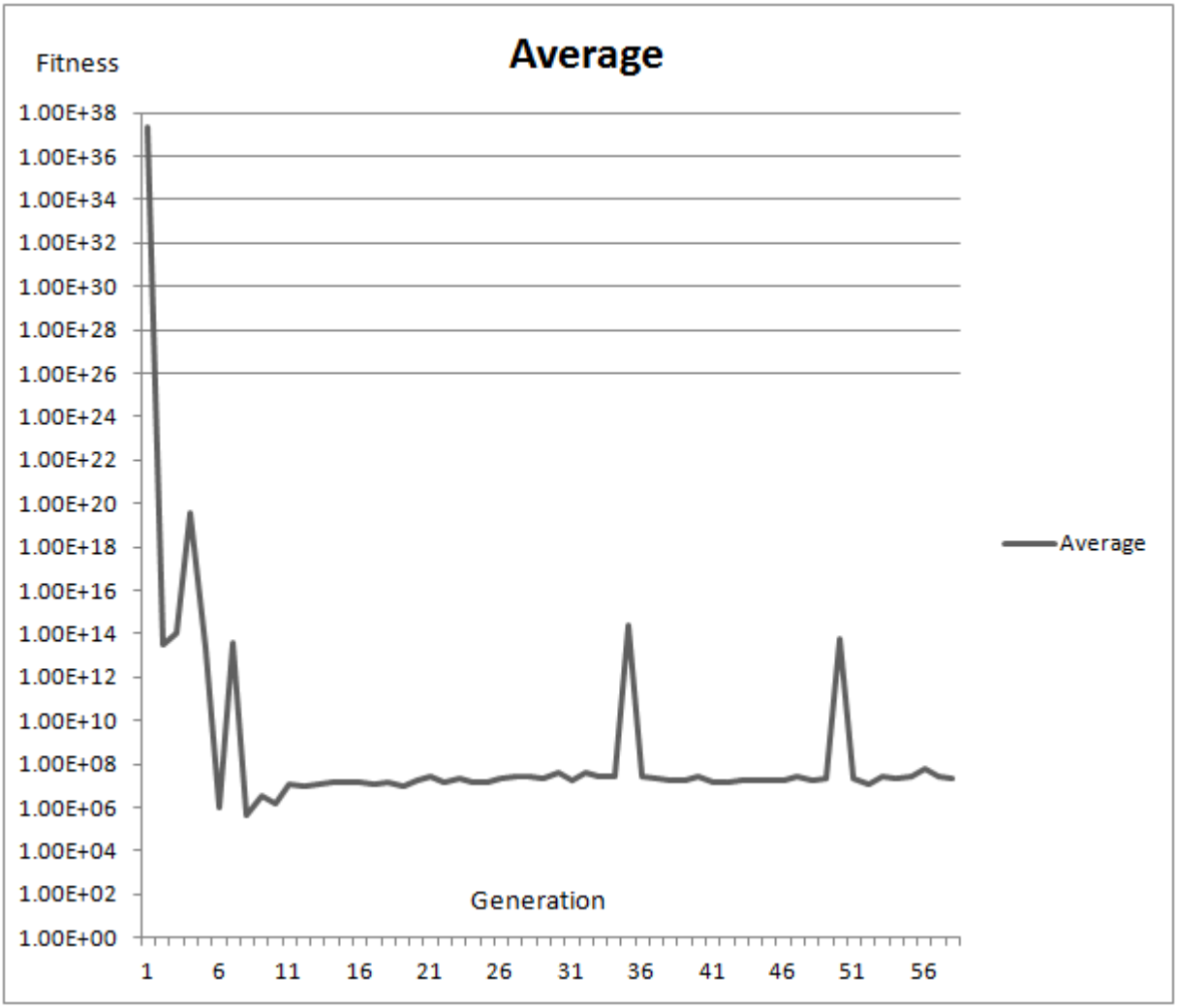
March 25, 2014

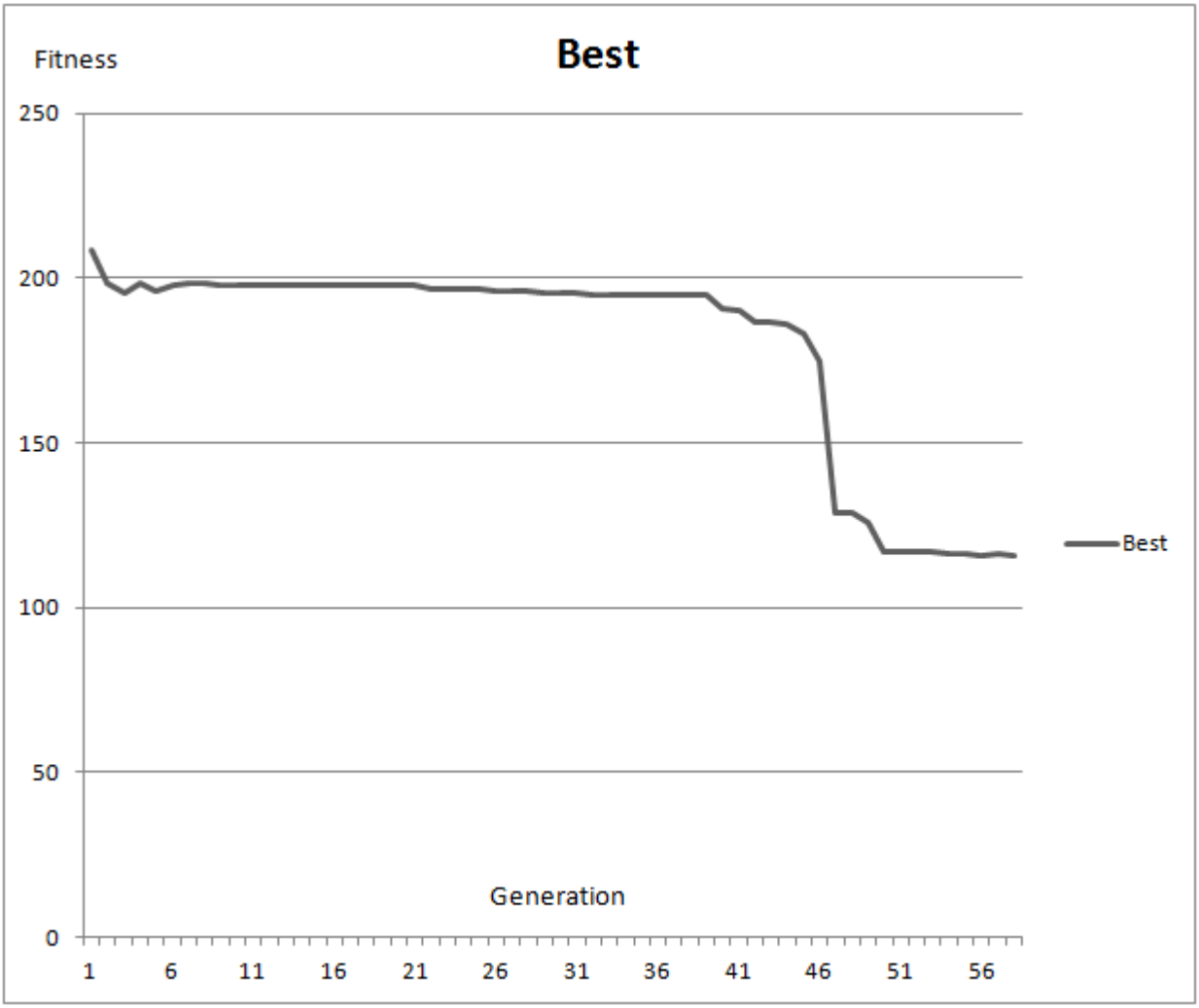
1 Abstract

Surprisingly this was a simple and fun project to work on. Creating a Genetic Program to fit a graph that has random noise seemed daunting at first, however splitting the project into simple processes for evolution, made everything very simple. The table below outlines a lot of this process. Following the table are three graphs, two showing the average and best fitnesses of a run and one showing the best graph that was found. The fitnesses are interesting to look at as they are completely different in their behaviours. The last thing I will talk about are my challenges and concerns about this project as a whole.

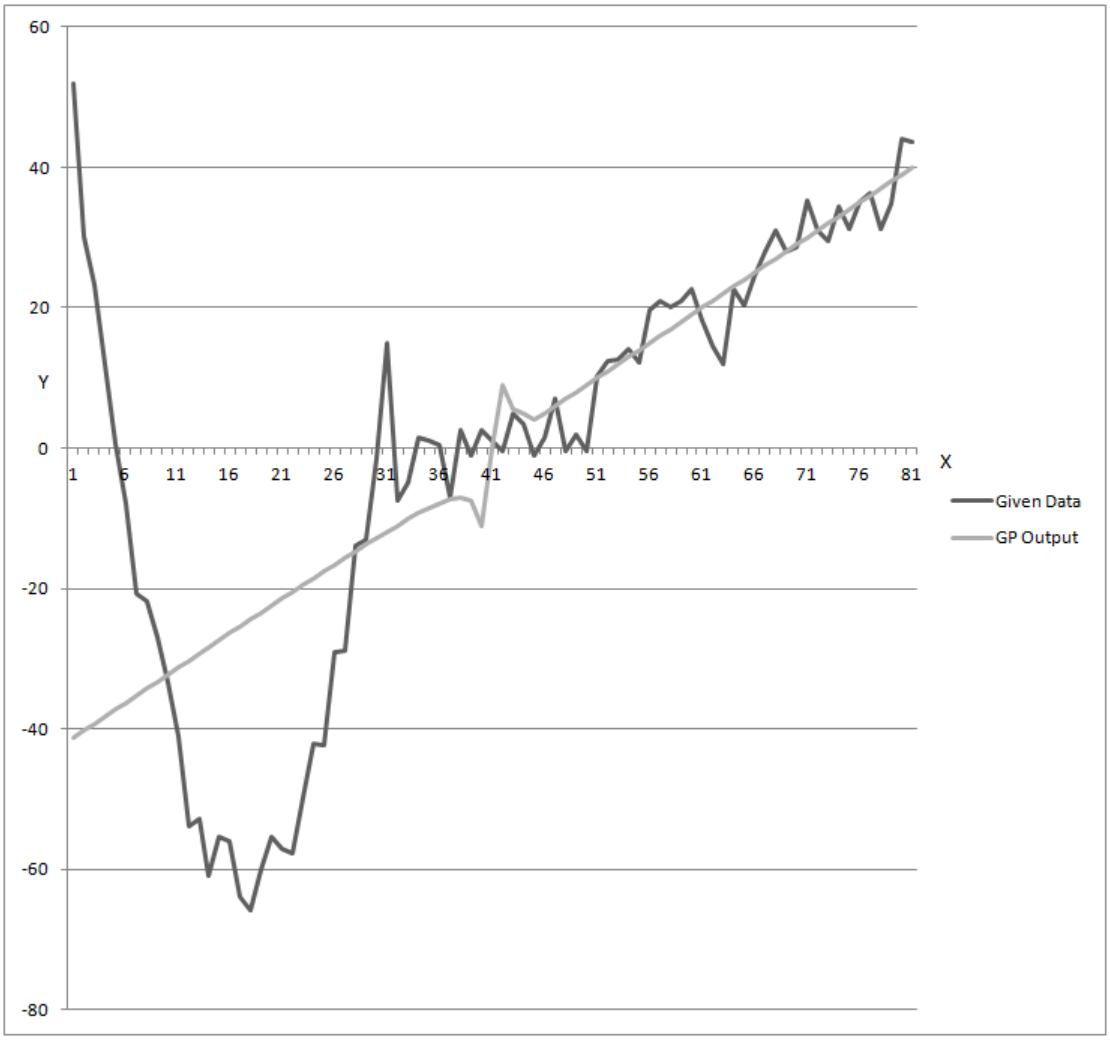
Algorithm	Generational
Population size	500
Selection method	Tournament, not selecting trees over a size of 200
Elitism	Copy the best individual twice
Crossover method	Random node switching between two trees.
Crossover rate	100% chance to crossover, 10% probability of getting a leaf
Mutation method	Create a new node of random type with a 10% probability
Operator/non-terminal set	$+ - * \div$ if then else
Terminal set	input of any size, constants from 0 to 10
Fitness function	The square root of the sum of the squared differences
Size control	During selection, no tree over the size of 200 will be picked

Fitnesses The following two graphs shows the best and average fitnesses over 60 generations. They are graphed separately because the average graph needs a logarithmic scale to show its shape. The graph shows that the average fitness is changing rapidly and has no pattern to whether it will get better or worse. The best fitness graph however shows that the best graph is always getting better. Just before generation 5 however there is a slight step in the wrong direction for that best fitness. This is because even though I carry over elites, if a smaller graph is found with almost the same solution it will be picked as the elite (Size Control).


Average
Fitness
1.00E+38
1.00E+36
1.00E+34
1.00E+32
1.00E+30
1.00E+28
1.00E+26
1.00E+24
1.00E+22
1.00E+20
1.00E+18
1.00E+16
1.00E+14
1.00E+12
1.00E+10
1.00E+08
1.00E+06
1.00E+04
1.00E+02
1.00E+00
Generation
1
6
11
16
21
26
31
36
41
46
51
56
Average


Best
Fitness
250
200
150
100
50
0
Generation
1
6
11
16
21
26
31
36
41
46
51
56
Best

Graph The following graph shows the data we were given and the data that my GP was able to produce. As you can see, the solution that I was able to produce through the GP was nothing close to what the actual data was producing. However compared to some of the worse fitness values, I am thoroughly impressed with the outcome considering how bad some of those graphs must look in comparison.



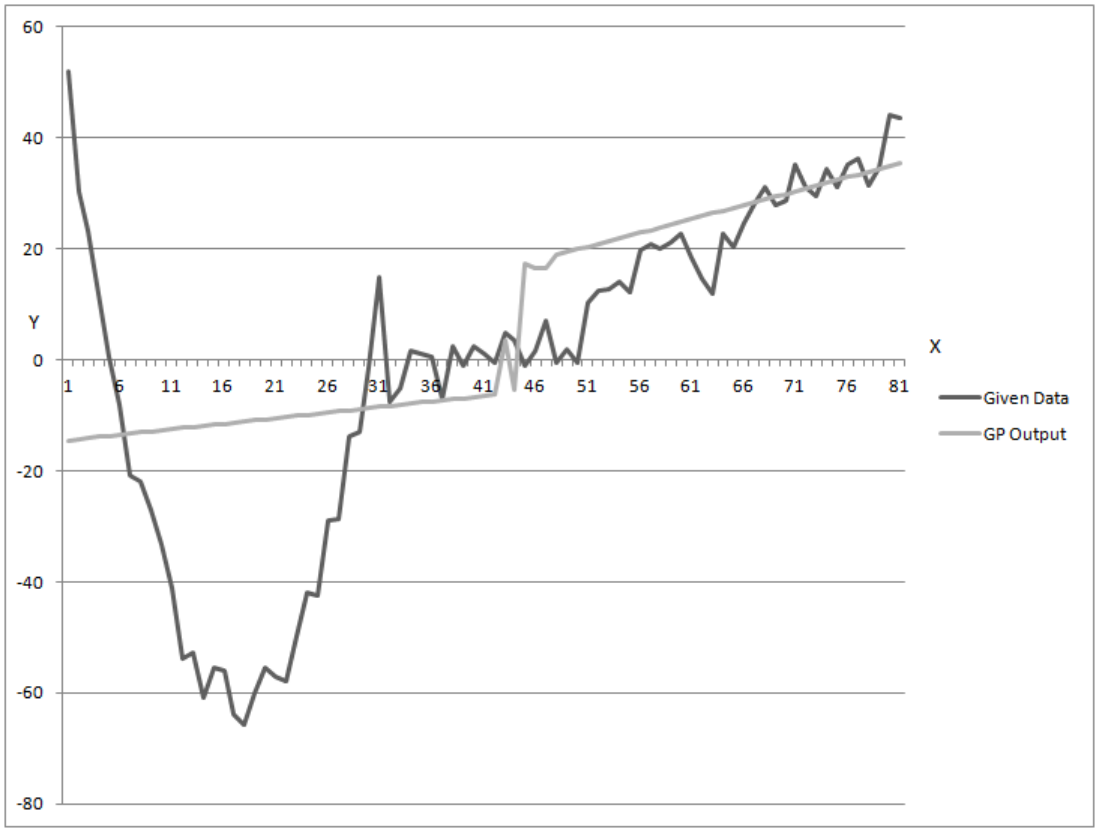
2 Discussion

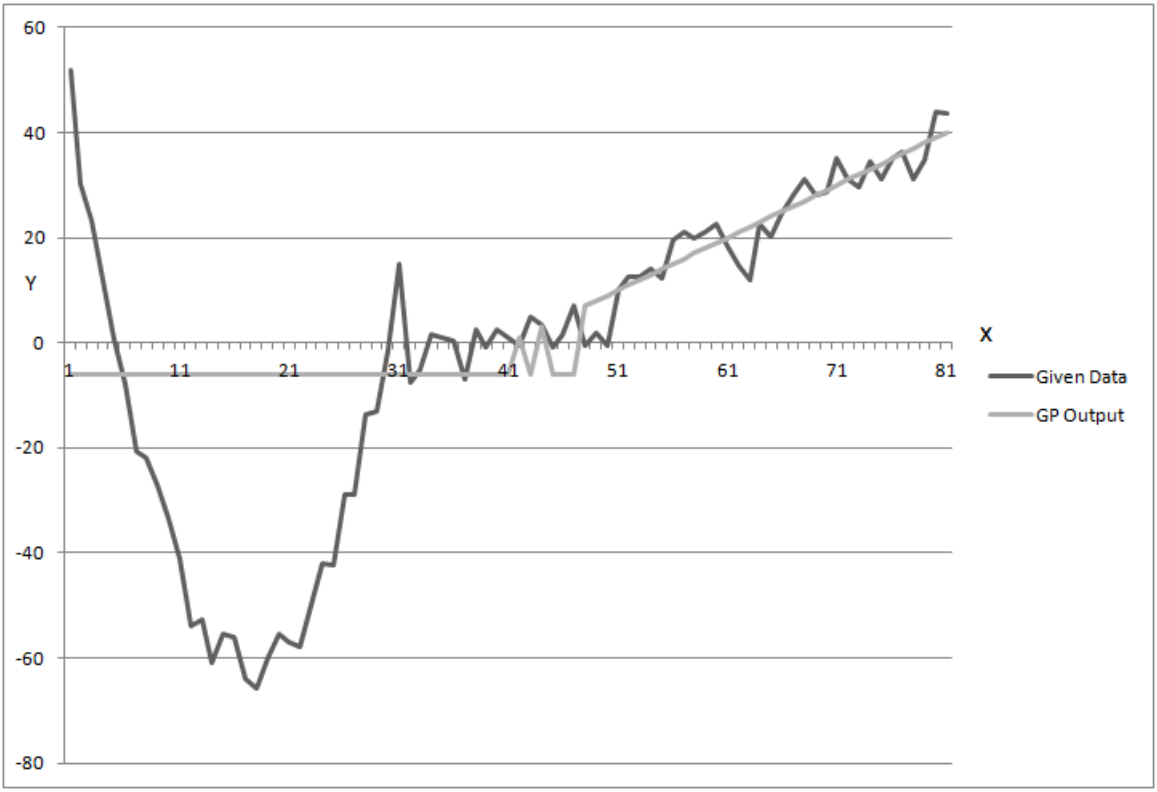
Results While I am happy with the GP being able to come up with a solution that is not just a straight line, some things arguably could have gone better and improved my GP substantially. The first thing is that even with deletion and protection on the tree, after roughly 60 generations the GP will fail because it has run out of memory. This is aggravating because with more time evolving it is possible that the GP would get a much better solution.

Other than memory issues most everything else worked smoothly. Crossover was interesting as I had some debugging prints that would show how much of a tree was being swapped and interestingly enough it kept fairly small but when a big change to a tree happened, large jumps in fitness were almost always observed. These jumps were almost always for the worse but at least they showed that they were doing something.

Graphs After multiple runs through the GP, it seems like my average fitness graph is a little abnormal in the fact that it stays somewhat stable. This is not the case for most of my other averages. After this section is a little bonus of some other graphs I got because I thought it was interesting that they all seem to be roughly the same graph with small variations at certain points which amazes me. If only I could get my program to evolve just a bit longer, I might be able to get to a point where a crossover or mutation takes two ok graphs and puts them together to make a great graph. Another observation is that more operators are definitely needed to get an almost perfect solution here. There is just no way to get some of the movements that the noise is producing without something that isn't a standard mathematical operator.

3 Appendix A


60
40
20
Y
0
-20
-40
-60
-80
1
6
11
16
21
26
31
36
41
46
51
56
61
66
71
76
81
X
Given Data
GP Output


60
40
20
Y
0
-20
-40
-60
-80
1
11
21
31
41
51
61
71
81
X
Given Data
GP Output