Computationnal Intelligence

For this project I begin to work on the lisp one but very quickly I was blocked.

I begin by the way to do python project wich give me idea to work again on the lisp project.

For the lisp project , the only problem is read csv file. That's why the input data are artificialy made.

The aim of the lisp project is to find the function x*x + x + 1.

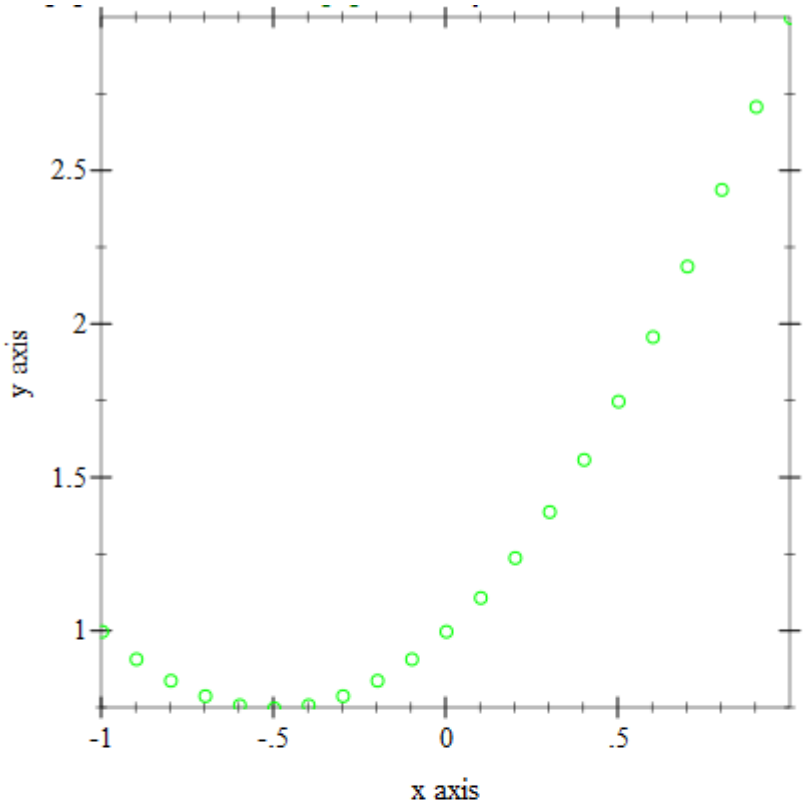
In order to do this I create a binary tree structure ((*tree left right node leaf)* with left and right the left and right tree, node the operation and leaf a former integer but not useful anymore) thanks to struct keyword. Thanks to pattern matching the recursive function was easy to make.

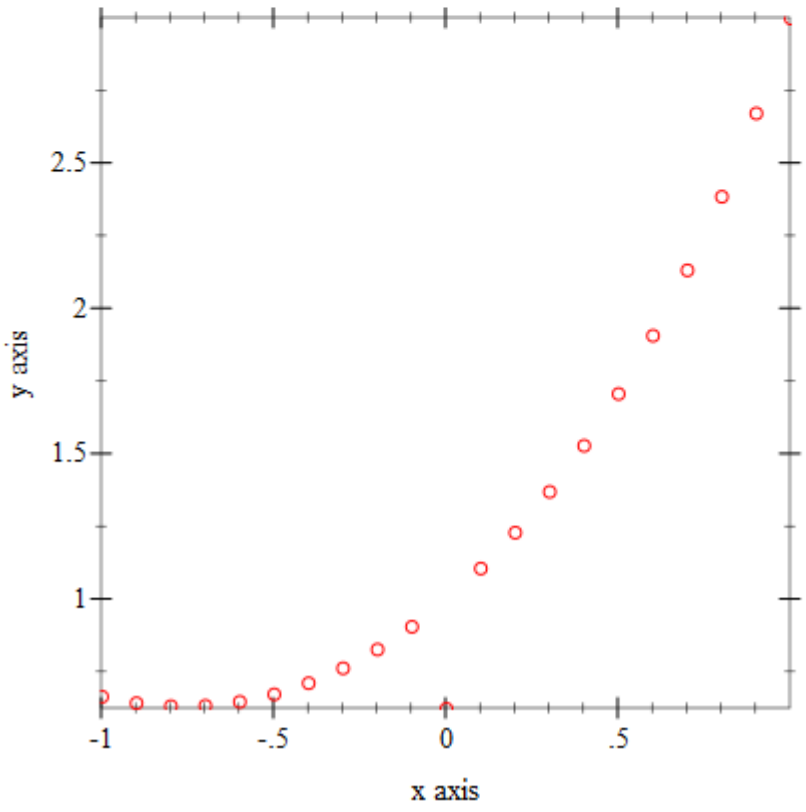
With several execution of the program I manage to have function very close to the target one.

(results functions are in commentary at the end of the program)

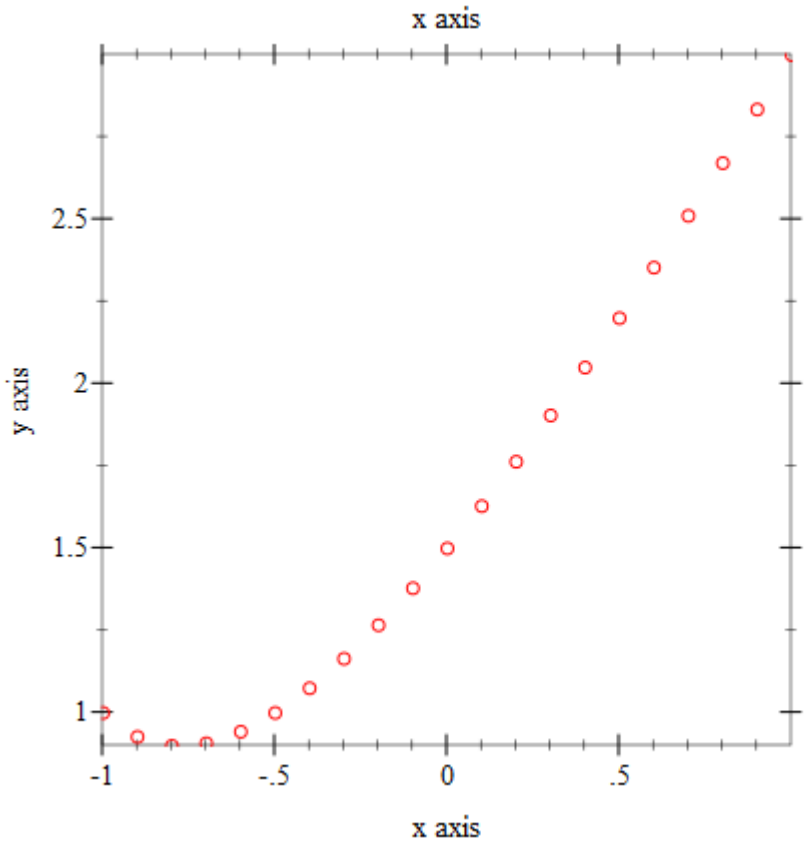
Since I didn't manage to plot curve I read by hand and simplify by hand the tree (we just need to know that in order to avoid division error I create safe_div operation that return "a" when "a/b" return error (if "b" equal 0))

Here a some of the result function that I obtain: with a population of 300 and 100 hundred generation. The algorithm is much more faster than python one. In green this is the aim curve and in red the best one find after the genetic algorithm.

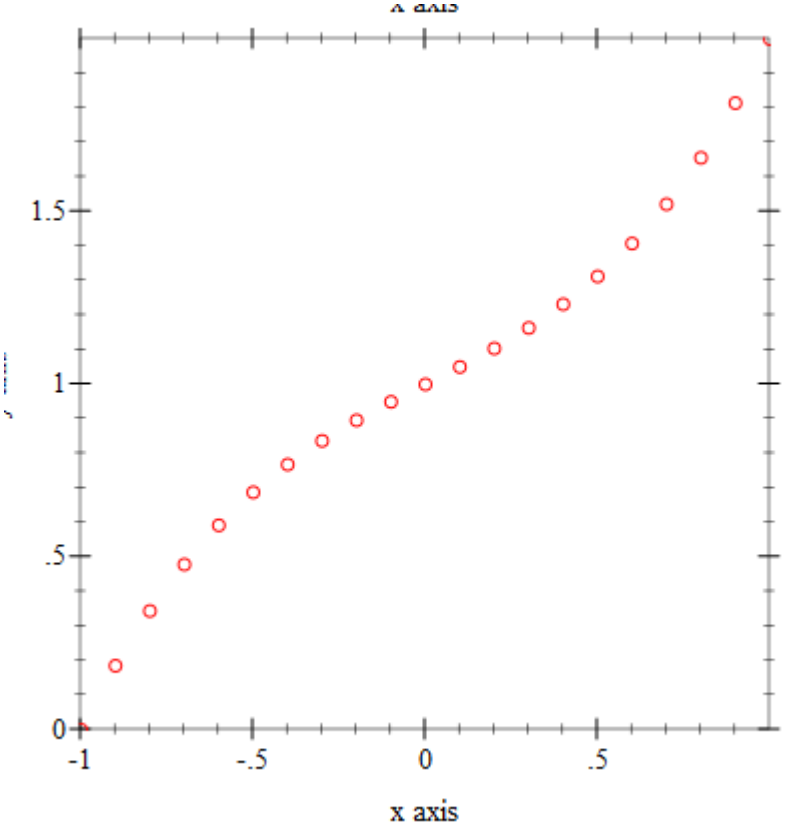

2.5
2
1.5
1
y axis
-1
-.5
0
.5
x axis


2.5
2
1.5
1
y axis
-1
-.5
0
.5
x axis

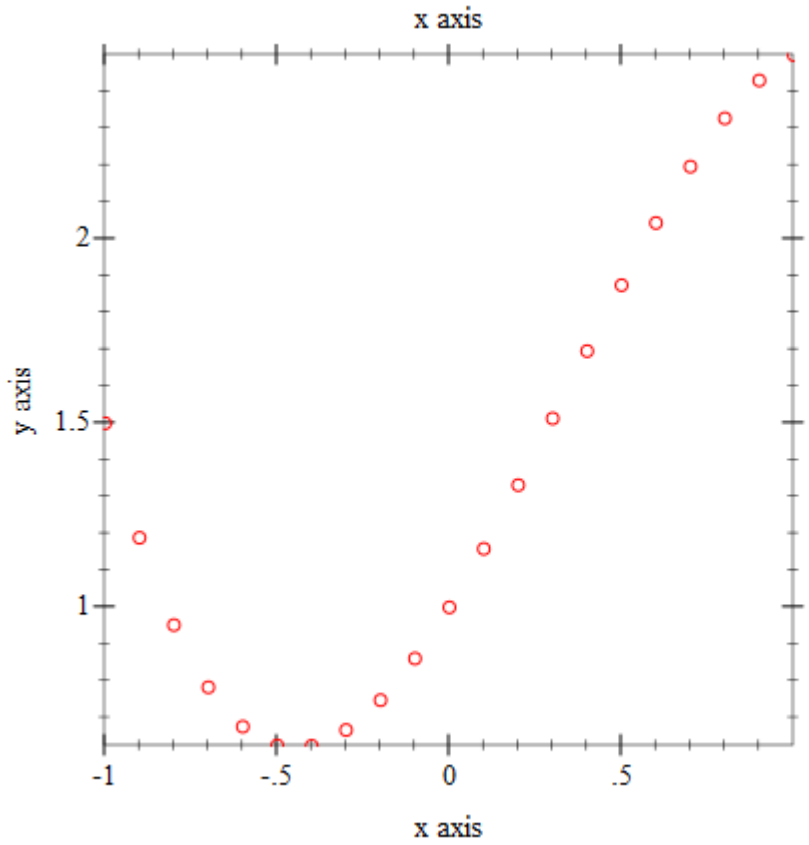
>

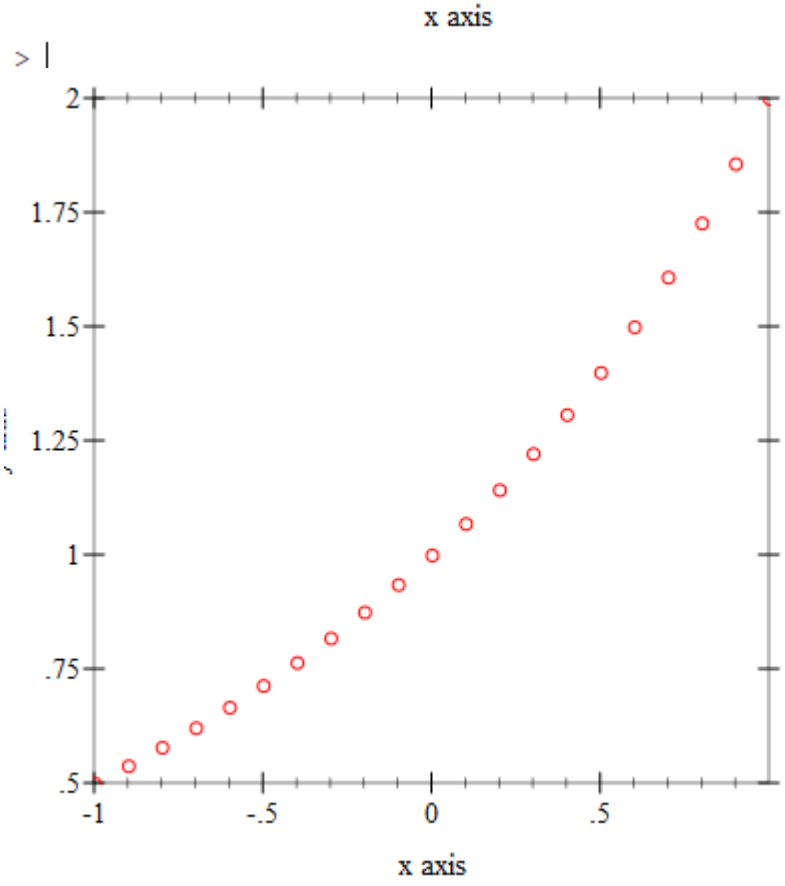


Here is another generation with 500 population and 200 generations



still with 500 and 200 but not very good

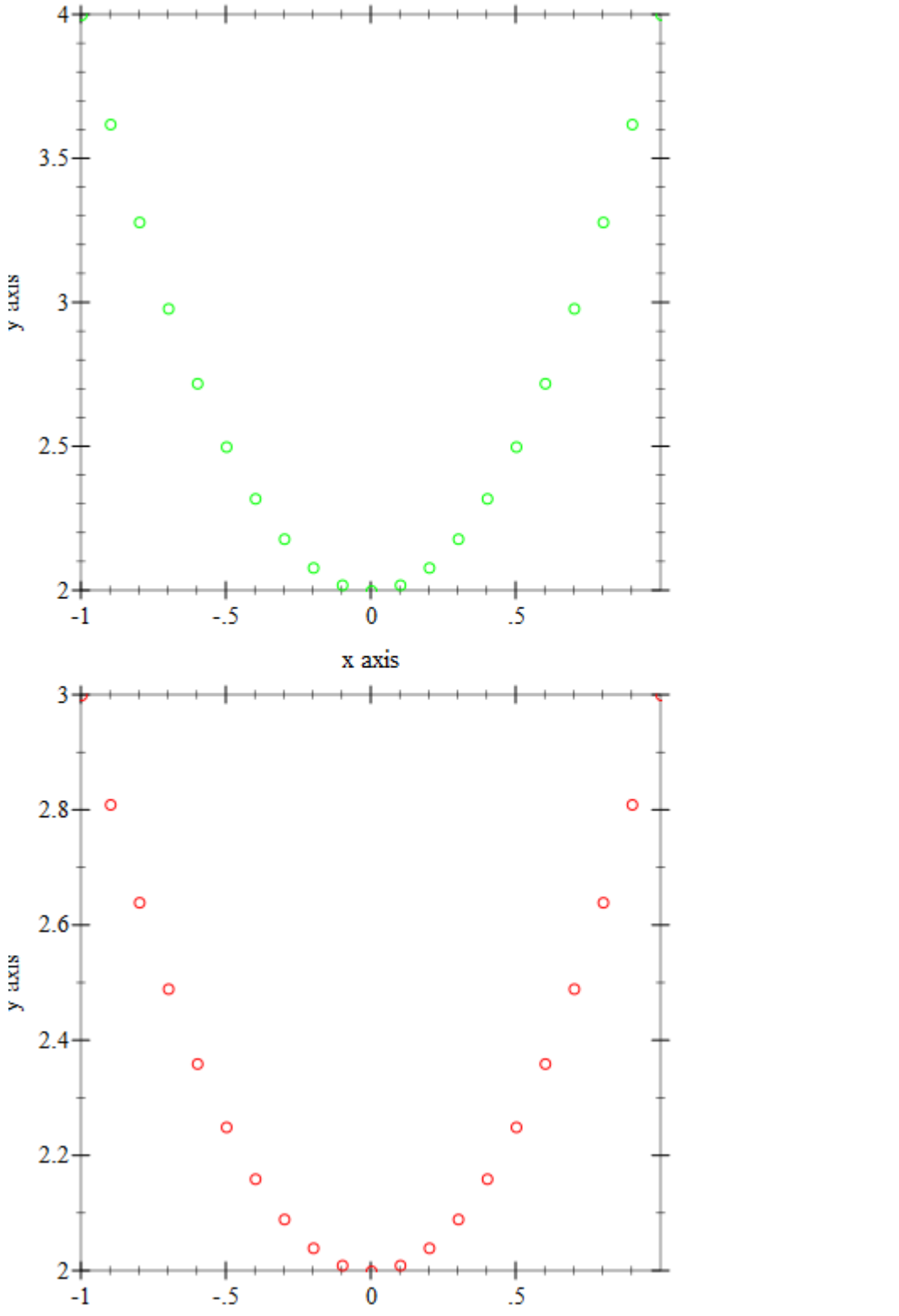




Here it is with 500 of pop and 300 generations

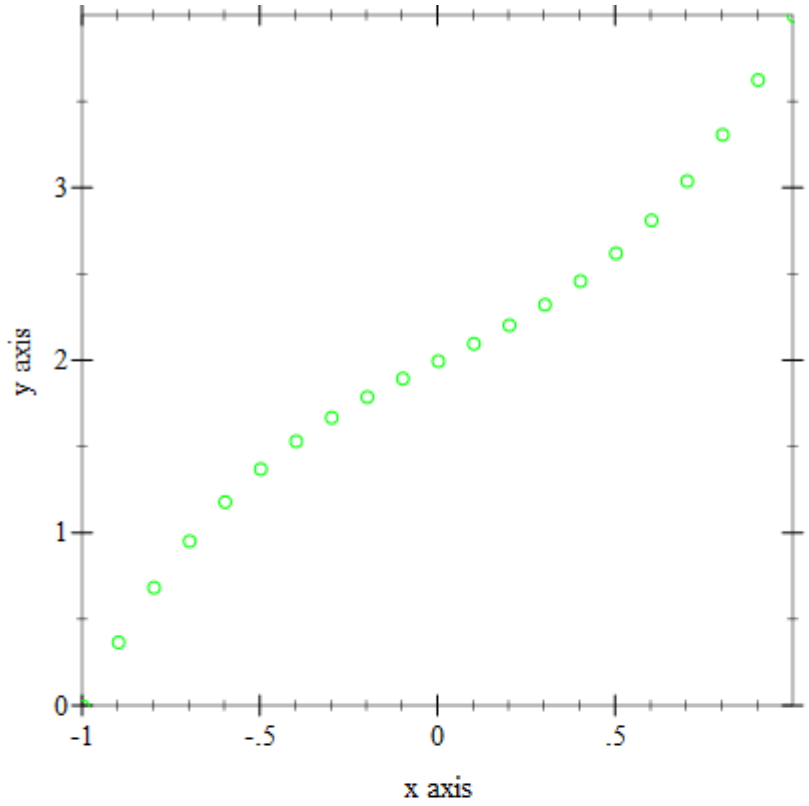
This is quite obvious but the more the population and generation are high the best the result look like

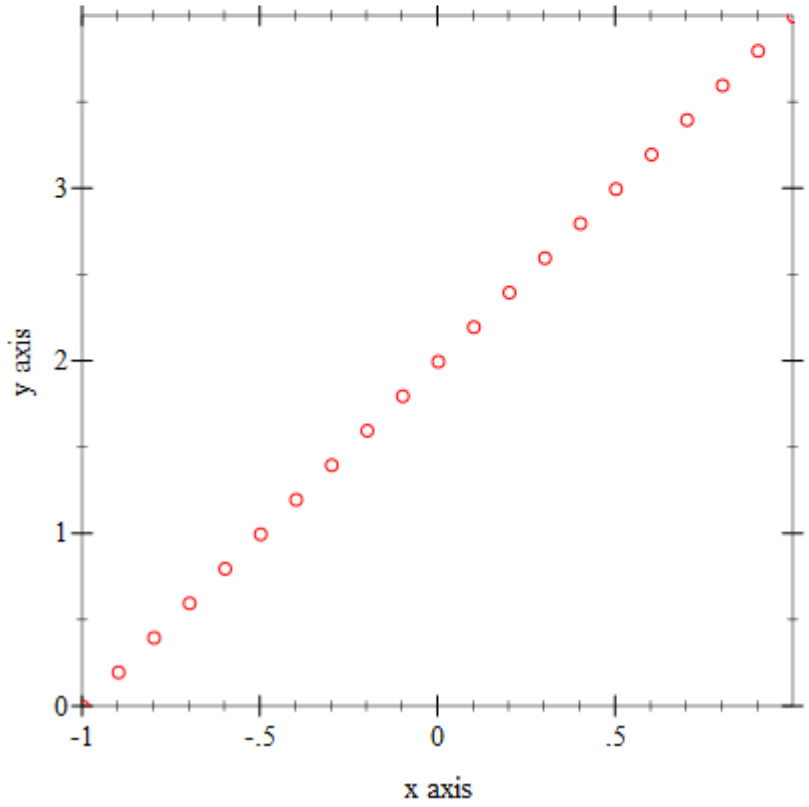
But it seems that is we had a big population but not enough generation the result won't be good enough and that's fully understandable since the low number of generation won't allow a good sort of bad prog.

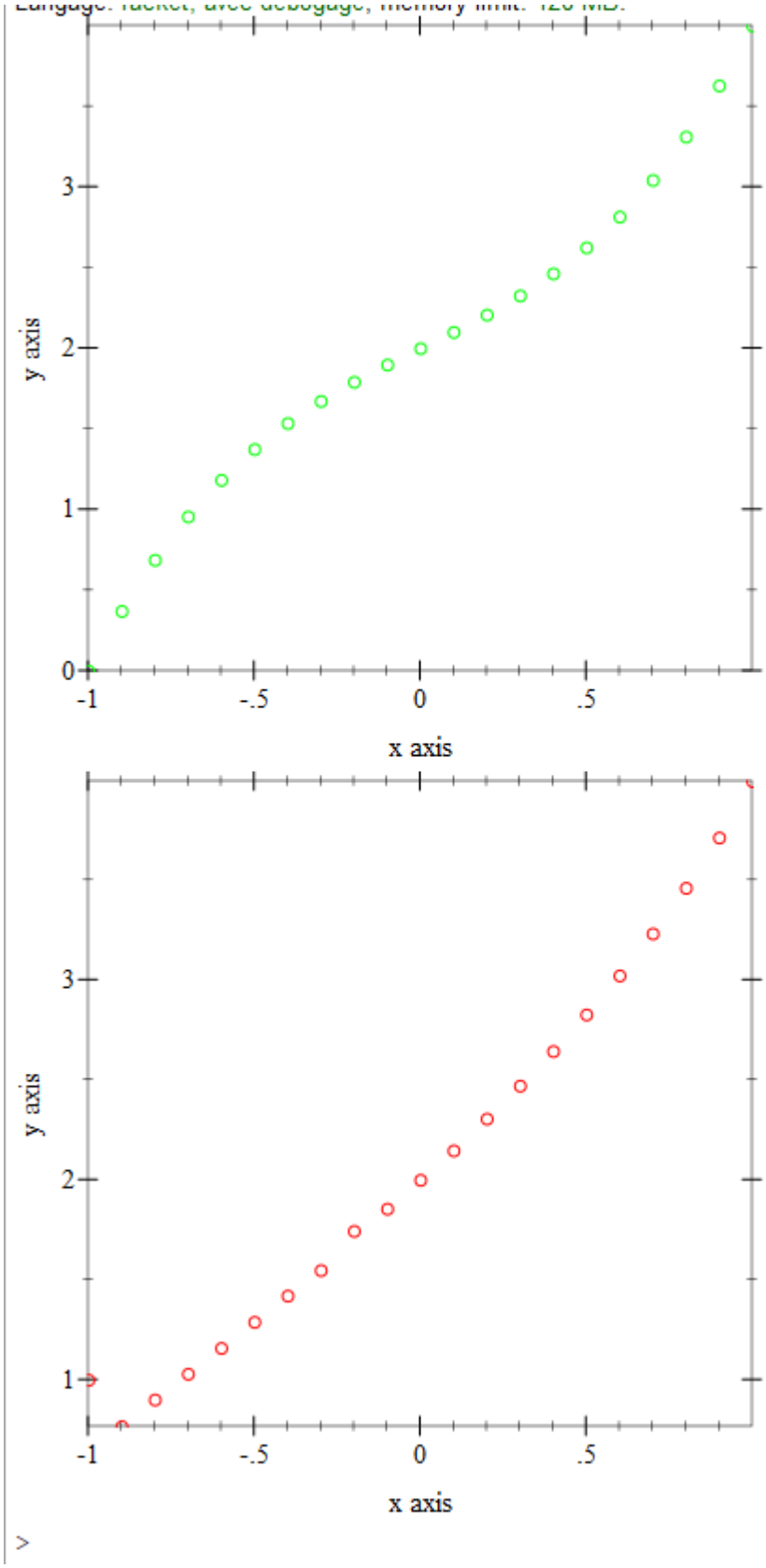


*Another function (2*x*x + 2) with a population of 500 and 2000 generations*

*A last function with the same population and generation but x*x*x + 2 function*


y axis
3
2
1
0
-1
-.5
0
.5
x axis


y axis
3
2
1
0
-1
-.5
0
.5
x axis



This is with 10000 pop and 5000 generations