Evolutionary Symbolic Regression of the Deviation of Land-Surface Temperature by Year

Mike Dean

Objective

To identify a pattern in global temperature if one exists.

Data

Obtained from Berkley Earth, dedicated to studying climate change and reduce greenhouse gas emissions.

The values represent the deviation from the average land-temperature recorded between Jan 1951 and Dec 1980: **8.70°C +/- 0.06°C.**

Deviations were measured for all years between 1753 and 2013.

Methods

Using Clojush/PushGP evolution was performed on the numerical value of the year (i.e. 1851 would be 1851.0).

Push was given random sequences of years to train a function to produce the annual deviation which was then tested on another random sequence of years.

Code

```
(defn train [n i]
  (def train-data (rand-nth (partition n (shuffle data1))))
  (def test-data (rand-nth (partition i (shuffle (remove (set train-data) data1)))))
  (def argmap-train
   {..
  :ultra-probability 1
  :ultra-alternation-rate 0.01
  :ultra-alignment-deviation 1
  :ultra-mutation-rate 0.05
  :return-simplified-on-failure true
```

The function above takes two inputs which randomize the dataset and creates two partitions of the data, the first is used to develop a program while the second set is what the program is tested on to determine the error.

```
(ns Regression.operators
 (:use [clojush.pushstate]
       [clojush.util])
  )

(defn ps
  "Protected square root; returns 0 if the radicand is less
than zero."
  [radicand]
  (if (> 0 radicand)
    radicand
    (java.lang.Math/sqrt radicand)))

(defn plog
  [num]
  (if (>= 0 num)
    0
    (java.lang.Math/log num)))

(define-registered
  float_sqrt
  (fn [state]
    (if (not (empty? (:float state)))
      (push-item (keep-number-reasonable
                   (ps (stack-ref :float 0 state)))
                 :float
                 (pop-item :float state))
      state)))
```

Various mathematical operations allowed on the input values.

```
(define-registered
  float_log
  (fn [state]
    (if (not (empty? (:float state)))
      (push-item (keep-number-reasonable
                   (plog (stack-ref :float 0 state)))
                 :float
                 (pop-item :float state))
      state)))

(define-registered
  float_ex
  (fn [state]
    (if (not (empty? (:float state)))
      (push-item (keep-number-reasonable
                   (Math/pow Math/E (stack-ref :float 0
state)))
                 :float
                 (pop-item :float state))
      state)))

(define-registered
  float_cbrt
  (fn [state]
    (if (not(empty? (:float state)))
      (push-item (keep-number-reasonable
                   (Math/cbrt (stack-ref :float 0
state)))
                 :float
                 (pop-item :float state))
      state)))
```

```
:error-function (fn [program]
                     (doall
                       (for [[input output] train-data]
                         (let [state (run-push program
                                               (push-item input :auxiliary
                                                          (push-item input :float
                                                                     (make-push-state))))
                               top-float (top-item :float state)]
                           (if (number? top-float)
                             (abs (- top-float output))
                             1000.0)))))

   :atom-generators (list (fn [] (lrand 15))
                          'in
                          'float_div
                          'float_mult
                          'float_add
                          'float_sub
                          'float_div
                          'float_log
                          'float_ex
                          'float_tan
                          'float_sin
                          'float_cos
                          'float_sqrt
                          'float_cbrt


  )})
```

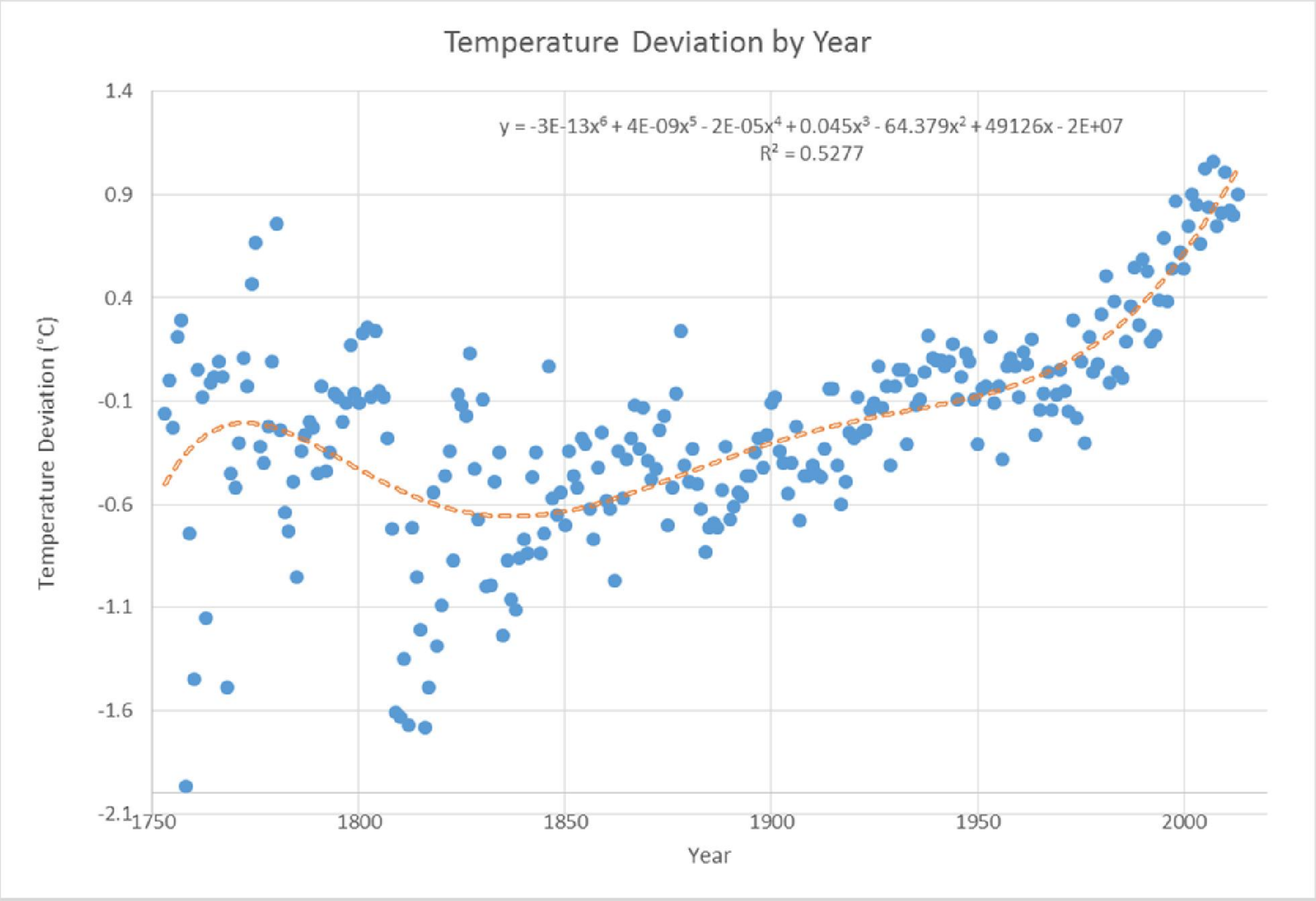
Definition of the error function which is the difference of the output of the evolved program and the y-value of the data.

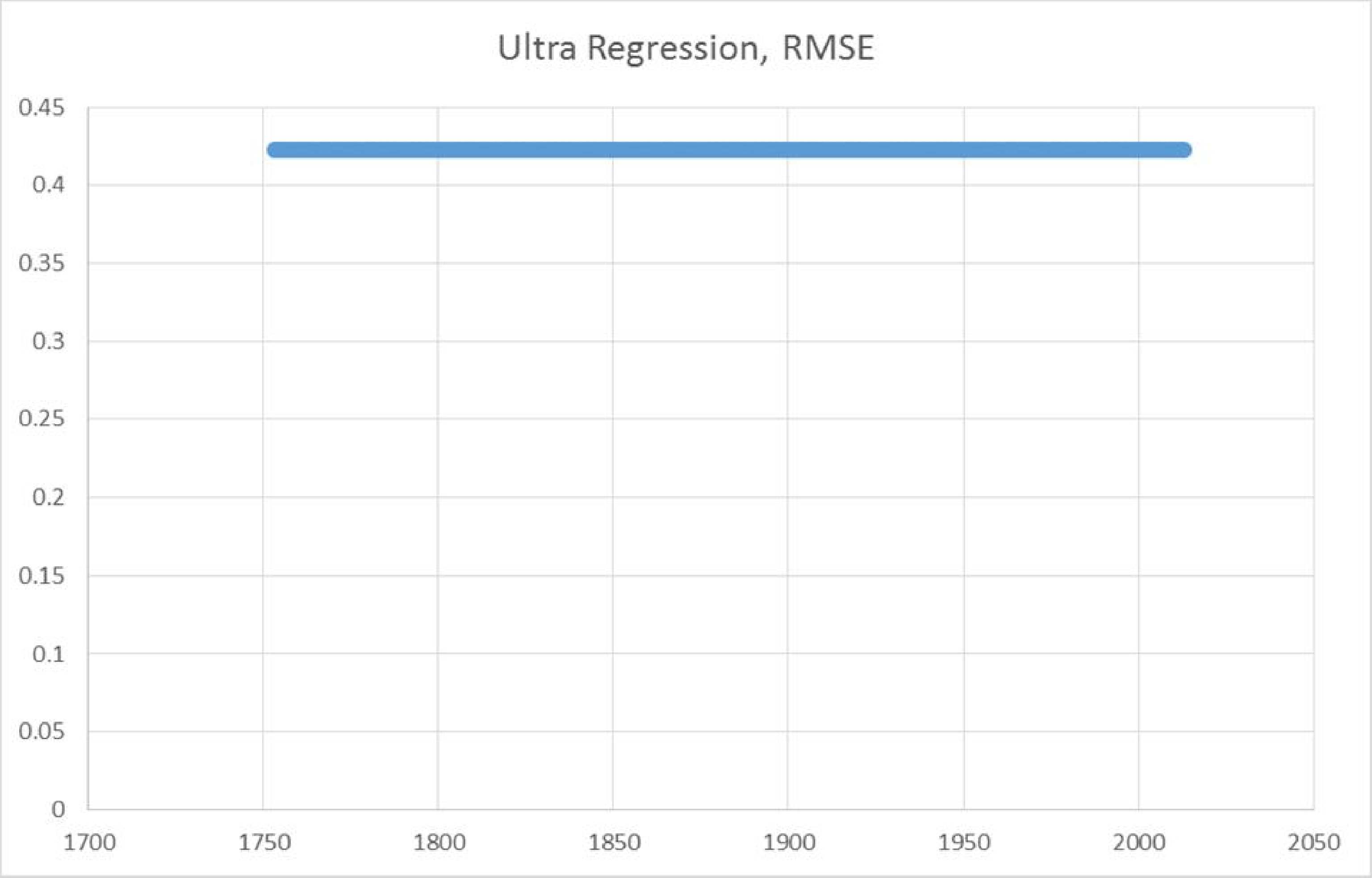
```
(def best-program (:program (pushgp argmap-train)))


(defn test-program [x]
  (:float (run-push (replace {'in x} best-program) (make-push-state)))]
   )
 )



(defn test-all-error [] (/ (reduce + (map abs (vec (map - (map last data1)
(flatten (map test-program (map first data1)))))))
                           (count (map abs (vec (map - (map last data1) (flatten
(map test-program (map first data1))))))))
)
```

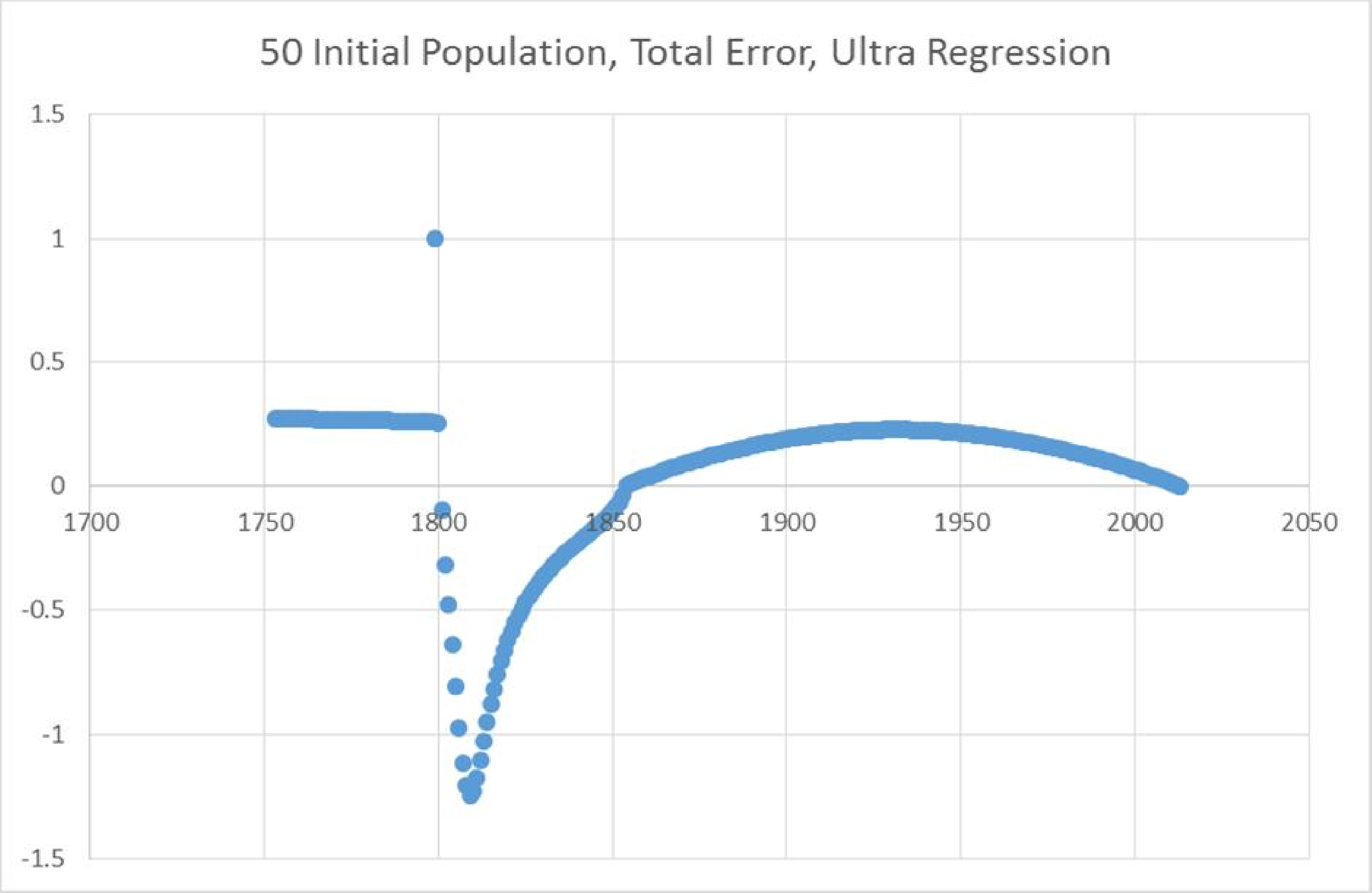
These functions take the best program and assess the error on the test cases, i.e. the cases which did not train the program.

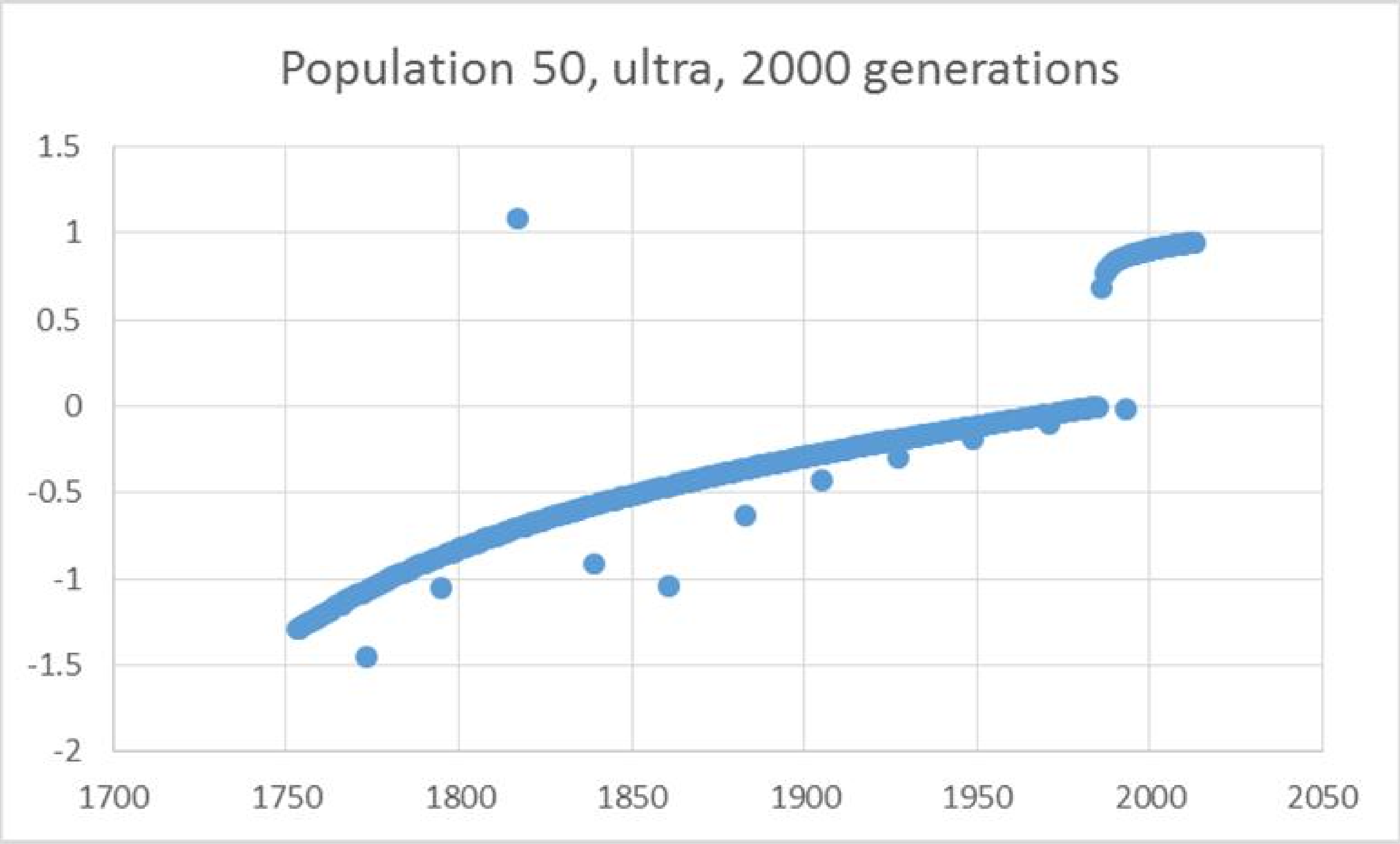
Results

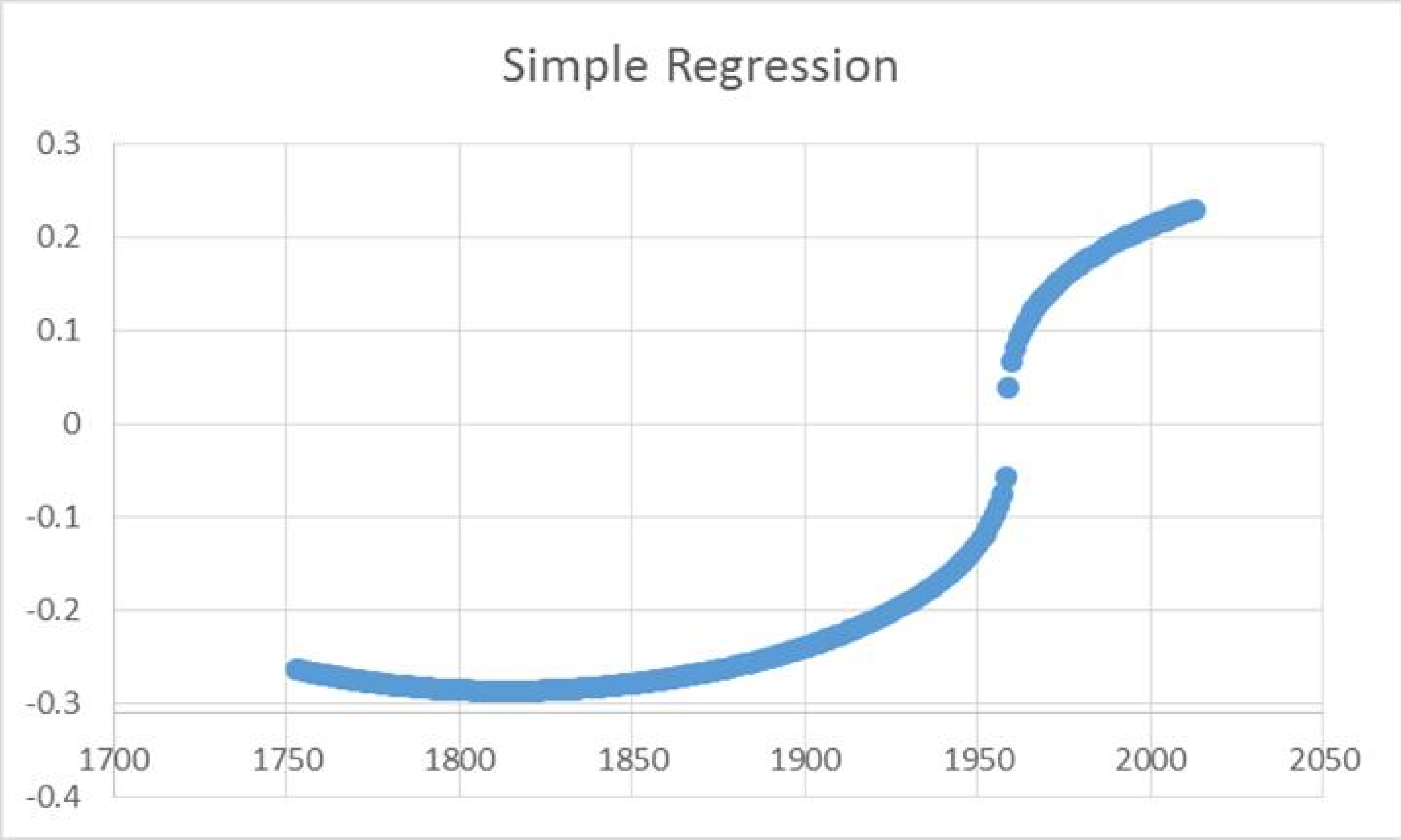
```
(-7.2810737922891064 -16.592578541392385 float_sin float_div float_div
  float_tan float_tan float_cos float_sqrt float_ex float_sin in float_sqrt
  float_sin float_sin float_sin float_sin float_sqrt float_sin float_mult float_tan
  float_tan float_tan 1.905420410743711 float_cbrt float_div float_tan
  float_sin -5.537283407645102 float_tan float_log float_tan float_tan
  float_tan float_tan float_add float_tan)
```

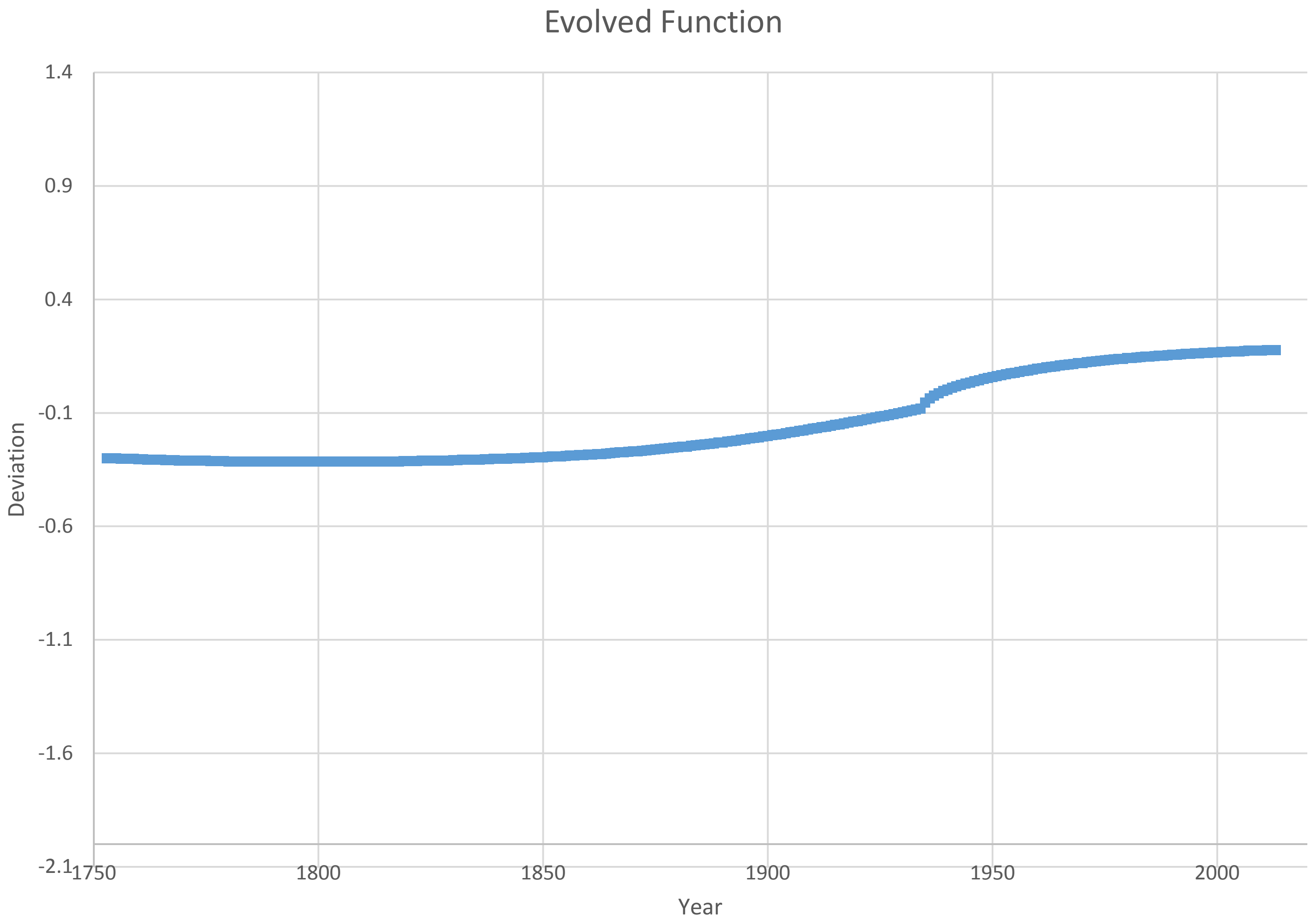

Temperature Deviation by Year
y = -3E-13x⁶ + 4E-09x⁵ - 2E-05x⁴ + 0.045x³ - 64.379x² + 49126x - 2E+07
R² = 0.5277
Temperature Deviation (°C)
1.4
0.9
0.4
-0.1
-0.6
-1.1
-1.6
-2.1
1750
1800
1850
1900
1950
2000
Year

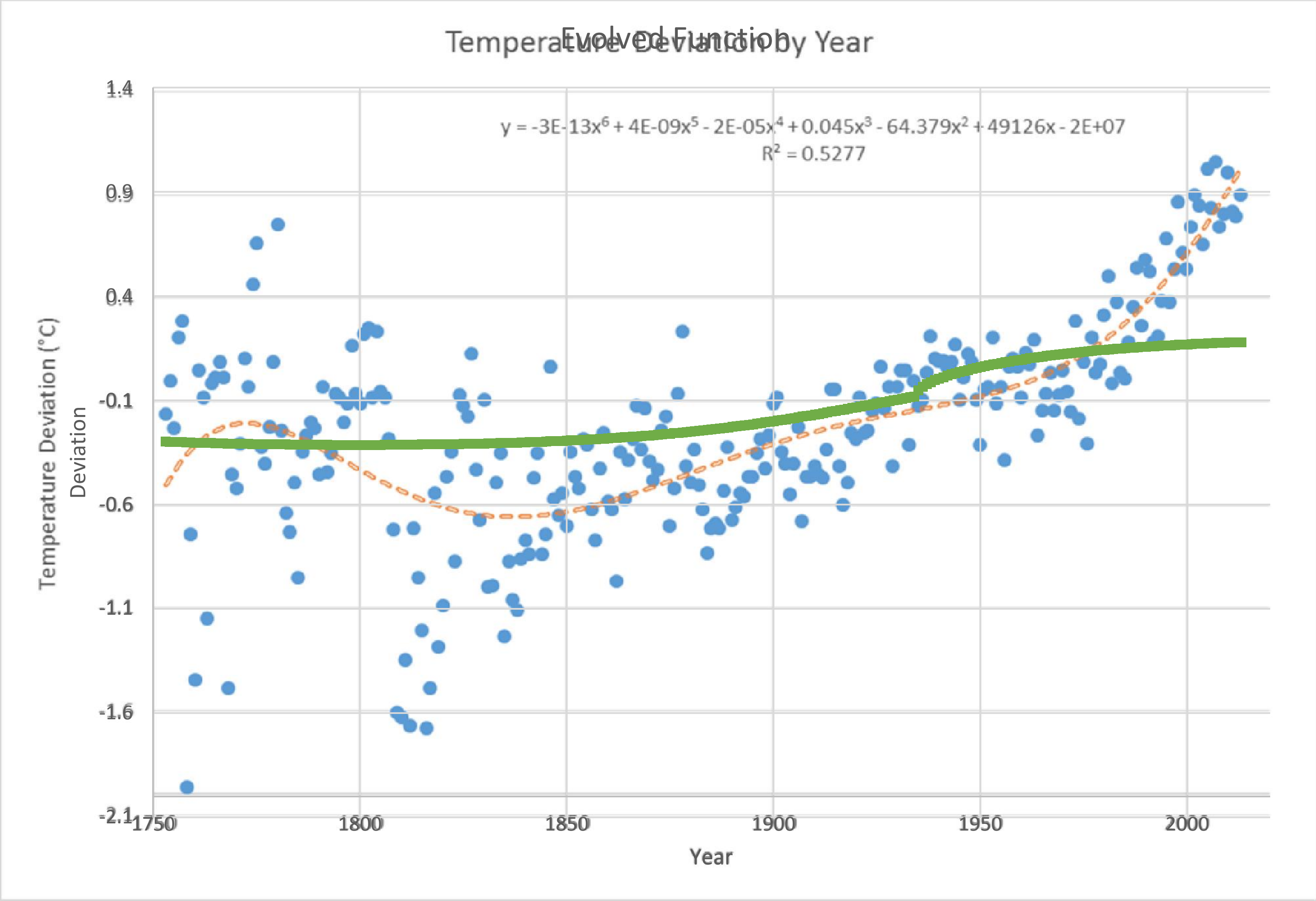

Ultra Regression, RMSE
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
1700
1750
1800
1850
1900
1950
2000
2050


50 Initial Population, Total Error, Ultra Regression
1.5
1
0.5
0
-0.5
-1
-1.5
1700
1750
1800
1850
1900
1950
2000
2050


Population 50, ultra, 2000 generations
1.5
1
0.5
0
-0.5
-1
-1.5
-2
1700
1750
1800
1850
1900
1950
2000
2050


Simple Regression
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
1700
1750
1800
1850
1900
1950
2000
2050


Evolved Function
1.4
0.9
0.4
-0.1
-0.6
-1.1
-1.6
-2.1
1750
1800
1850
1900
1950
2000
Year
Deviation


Evolved Function
Temperature Deviation by Year
y = -3E-13x⁶ + 4E-09x⁵ - 2E-05x⁴ + 0.045x³ - 64.379x² + 49126x - 2E+07
R² = 0.5277
1.4
0.9
0.4
-0.1
-0.6
-1.1
-1.6
-2.1
1750
1800
1850
1900
1950
2000
Year
Temperature Deviation (°C)
Deviation

Outcome

Is there a pattern?

2015: 8.88°C = 47.98°F