IE 607 Heuristic Optimization

Miscellaneous Methods - II

Genetic Programming (GP)

- J. F. Koza (1993), *Genetic Programming: On the Programming of Computers by Means of Natural Selection and Genetics*, MIT Press.
- Main Idea: use a GA to search over logical and arithmetic strings to find the best fit to a data set

GP Procedure

1. Define an objective function and a data set, or means of evaluation. Example: find the 3rd order polynomial that yields the smallest squared error over this set of (x_1, x_2, …, x_n, y)
2. Define a function set such as +, *, sin, cos, exp and define a terminal set (the independent variables and an ability to make constants).

GP Procedure (cont.)

3. Form an initial population of tree structures, considering bounds or expectations on depth
4. Evaluate the population, considering tree depth
5. Choose two members, preferring those with better objective functions, for crossover

GP Procedure (cont.)

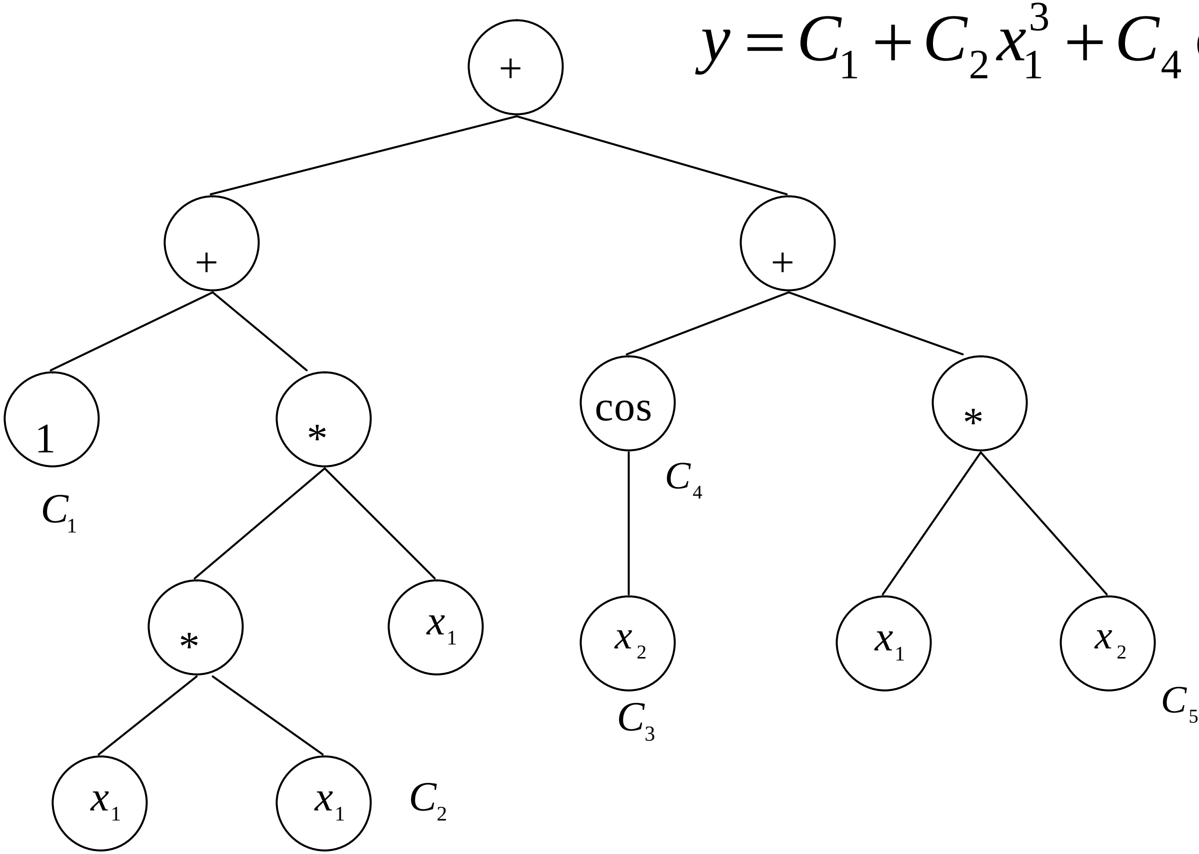
6. Perform crossover to create a child, then mutate the child.
7. Replace parents with children, or delete the worst of combined parents and children
8. Loop to step 5 and continue until stopping criteria are met

Function Set (Node Primitives)

Levels	Primitives
First Group	$x_1, x_2, \ldots, x_n, 1$
Second Group (unary)	sin, cos, exp, ln
Third Group (binary)	$+, *, \div$

Example of Tree Structure

$$y = C_1 + C_2 x_1^3 + C_4 \cos(C_3 x_2) + C_5 x_1 x_2$$

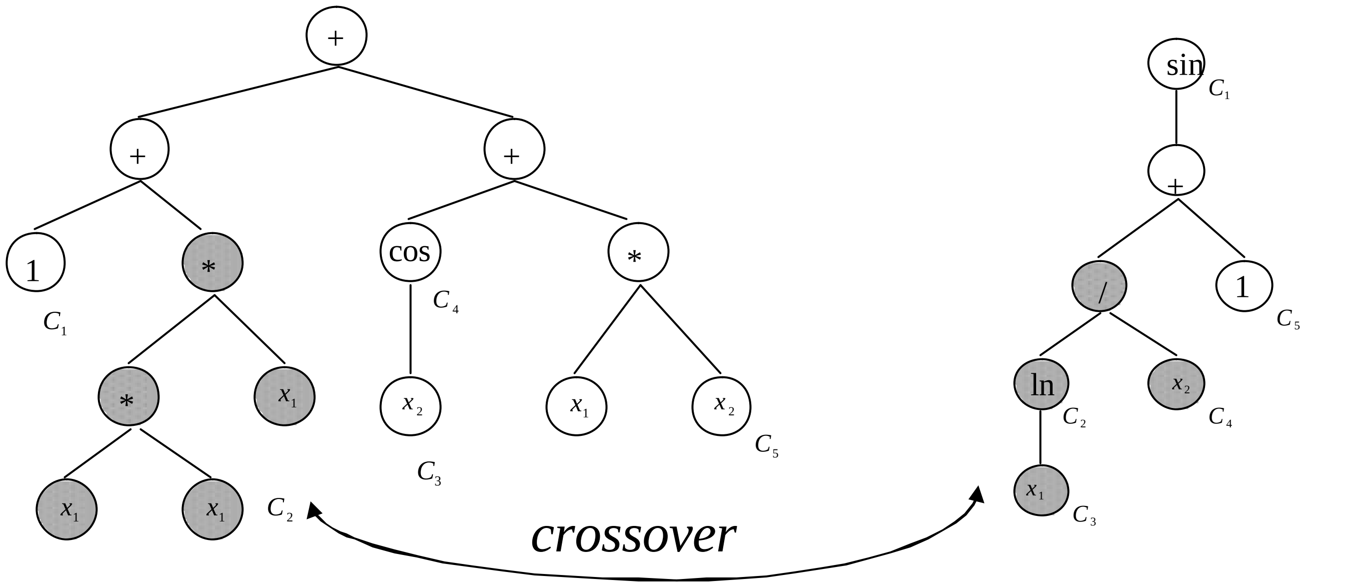


Crossover of Tree Structure

Before: **Parent 1** **Parent 2**

$$y = C_1 + C_2 x_1^3 + C_4 \cos(C_3 x_2) + C_5 x_1 x_2$$

$$y = C_1 \sin\left(\frac{C_2 \ln(C_3 x_1)}{C_4 x_2} + C_5\right)$$



After: **Offspring 1** **Offspring 2**

$$y = C_1 + \frac{C_2 \ln(C_3 x_1)}{C_4 x_2} + C_5 \cos(C_6 x_2) + C_7 x_1 x_2$$

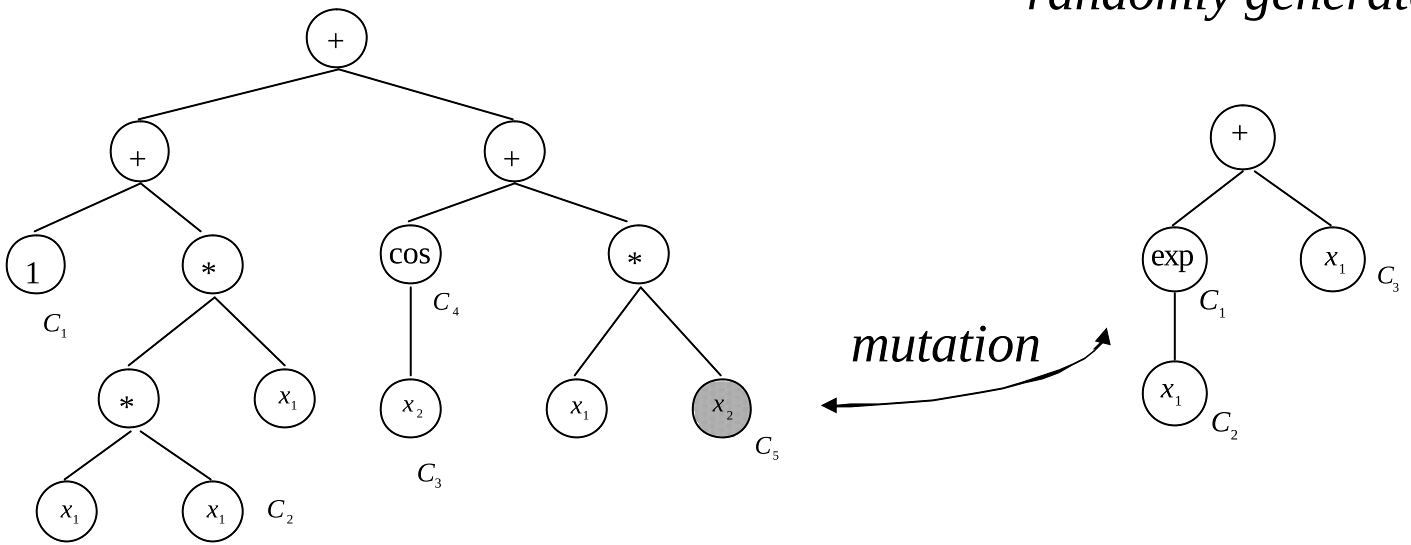
$$y = C_1 \sin(C_2 x_1^3 + C_3)$$

Mutation of Tree Structure

Before: **Parent 1**

$$y = C_1 + C_2 x_1^3 + C_4 \cos(C_3 x_2) + C_5 x_1 x_2$$

randomly generated tree



After: **Mutant**

$$y = C_1 + C_2 x_1^3 + C_3 \cos(C_4 x_2) + C_5 x_1 \exp(C_6 x_1) + C_7 x_1^2$$

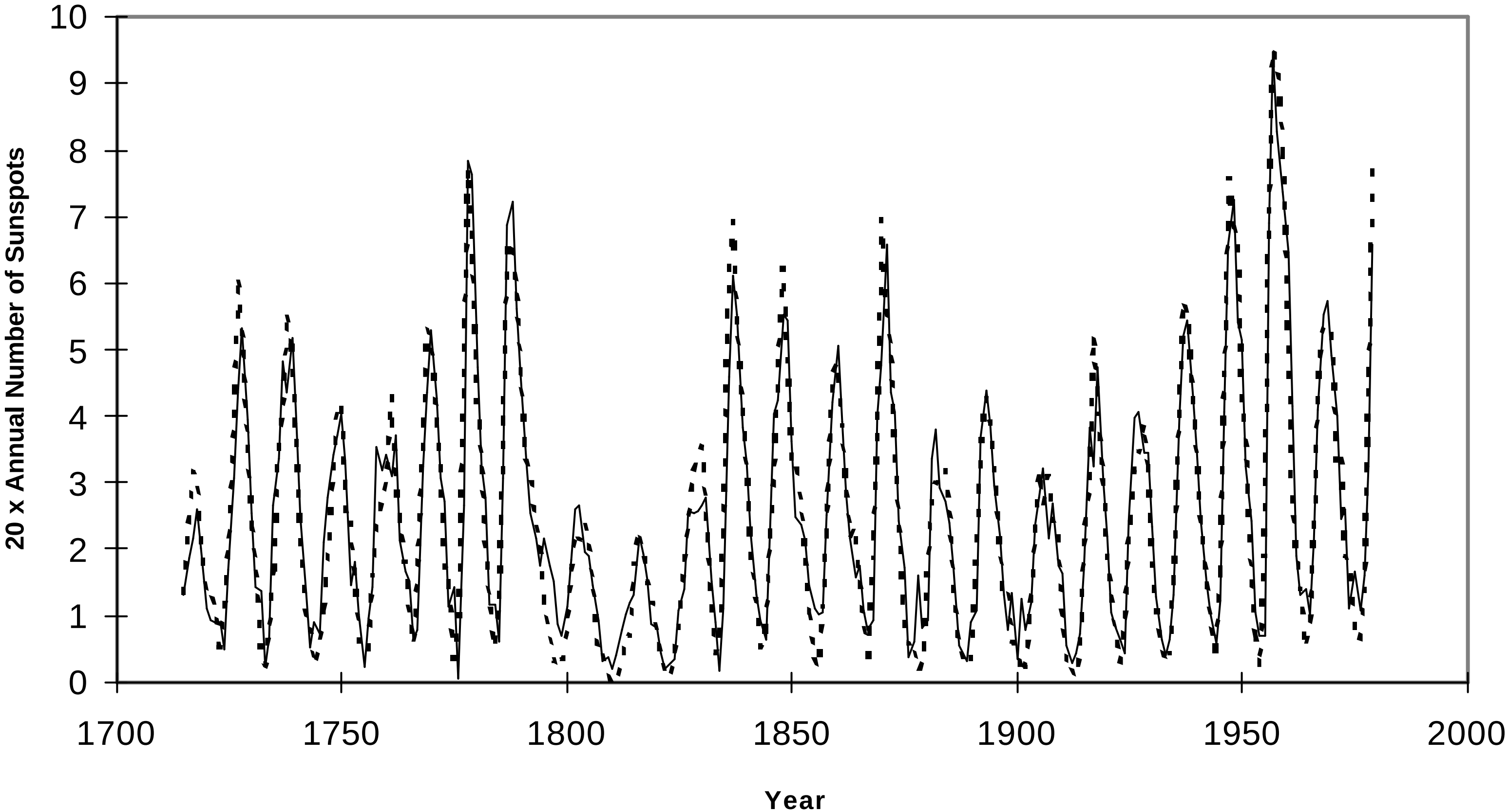
Test Problem 3, Sunspot Data

- Sunspot data from 1700 to 1995
- Highly cyclic with peak and bottom values approximately in every 11.1 years
- Cycle is not symmetric. The number of counts reaches to maximum value faster than it drops to a minimum
- Training range: 1700-1979
- Validation range: 1980-1995

Functions Identified

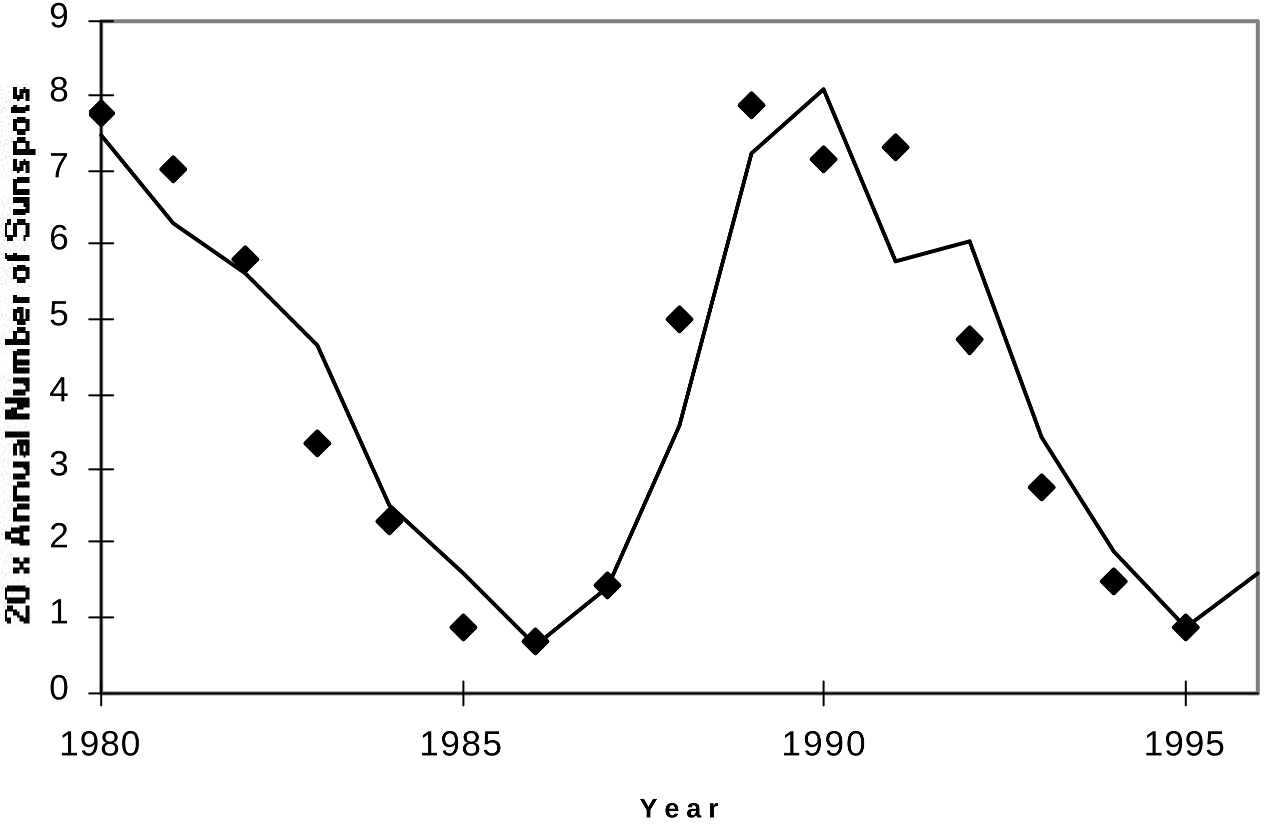
Model	Equation	SSE
A	$1.1965(t-1) - 0.4585(t-2) + 0.2471(t-9)$	61964
B	$0.8337(t-1) - 1.1989\exp(-0.3512(t-9))$ $+ 15.7476\exp(-2.7260\exp(-0.3263(t-1)) - 0.6271(t-2))$	45533
C	$1.2410\exp(-0.6099\cos(1.4097(t-4) + 0.4282(t-1))$ $- 0.8446(t-2))(t-1) + 0.8064(t-1) - 0.1316(t-4) + 0.1148(t-9)$	40341
D	$1.6258\exp(-0.5564\cos(-1.4893(t-4)$ $+ 0.6979\exp(-3.3442\cos(0.2561(t-4))$ $+ 2.8807(t-2)) - 3.1756(t-2)) - 0.7485(t-2)) - 0.9362(t-2)(t-1)$ $+ 0.8253(t-1) - 0.1413(t-4) + 0.1046(t-9)$	38715

Model D



10
9
8
7
6
5
4
3
2
1
0
20 x Annual Number of Sunspots
1700
1750
1800
1850
1900
1950
2000
Year

Extrapolation of Model D



Variable Neighborhood Search (VNS) and its Variations

- N. Mladenovic and P. Hansen (1997), "Variable neighborhood search," *Computers & Operations Research*, 24(11), 1097-1100.
- <u>Main Idea</u>: systematic change of neighborhood within the search

Variable Neighborhood Descent (VND)

Procedure (Max Problem)

1. Select the set of neighborhood structures N_k, k = 1, …, k_{max}
2. Generate an initial solution x
3. Set k = 1
4. Find the best neighbor x′ of x in N_k
5. If f(x′) > f(x), x = x′ & k = 1

   Otherwise, k = k+1
6. Loop to step 4, until k = k_{max}
7. Loop to step 3, until no improvement is obtained

Reduced Variable Neighborhood Search (RVNS)

Procedure (Max Problem)

1. Select the set of neighborhood structures N_k, k = 1, …, k_{max}
2. Generate an initial solution x
3. Set k = 1
4. Generate a solution x′ at random from N_k of x
5. If f(x′) > f(x), x = x′ & k = 1

   Otherwise, k = k+1
6. Loop to step 4, until k = k_{max}
7. Loop to step 3, until the stopping criteria are met

Stuffs about RVNS

- For very large instances in which local search is costly
- K_{max} is usually equal to 2
- Maximum number of iterations between two improvements is usually the stopping criterion

Basic Variable Neighborhood Search (VNS)

Procedure (Max Problem)

1. Select the set of neighborhood structures N_k, k = 1, …, k_{max}
2. Generate an initial solution x
3. Set k = 1
4. Generate a solution x′ at random from N_k of x
5. Apply local search to x′ & find the best neighbor x″ of x′
6. If f(x″) > f(x), x = x″ & k = 1

   Otherwise, k = k+1
7. Loop to step 4, until k = k_{max}
8. Loop to step 3, until the stopping criteria are met