HHC 2026

Symbolic Regression with Claude AI

Presented by
Namir Shammas

Disclaimer!

- I used Claude AI to generate the majority of the code for this presentation.
- I developed reasonably detailed instructions to guide Claude AI to generate the different versions of the code. I tested the code and in the case of error, asked Claude AI to fix the errors.
- This approach sped up the time to generate the code by an order of magnitude.

Basics

- We typically use OLS where the regression model is given and we calculate the intercept and regression coefficients based on a given set of X and y data. The general model is:
  Y = a0 + a1*f1(X) + a2*f2(X) + … + an*fn(X)
- Symbolic Regression (SR) allows you to supply the observed/calculated data and yields the best model that correlates y and X. SR was a term introduced in1992 by John Koza.
- SR is not magic. Give it random noise data and you get a formula that is nonsense!

Basics (cont.)

- Simplest case of SR we have all seen is fitting linear, logarithmic, exponential, and power models, then selecting the best regression model. That software has appeared for the HP-41C and HP-67.
- As the number of variables and the hyper-space curvature increase, SR will struggle with the solution.

General Types of SR Algorithms

- Stochastic symbolic regression algorithms.
- Classical Genetic-Programming based symbolic regression algorithms.
- Gene Expressiom Programming based symbolic regression algorithms.
- Neural-network and Machine Learning enhanced Genetic-Programming based symbolic regression algorithms.

Symbolic Regression the Big Picture

HHC 2026

- Has three frameworks:
- Algorithm for the selection of independent variables in each factor and in each term of the regression model.. **This is the Who**.
- Algorithm for the selection of operators and transformations of independent variables . **This is the What**
- Algorithms for calculating the regression model statistics (intercept, regression coefficients, R-square, adjusted R-Square, confidence intervals table, regression ANOVA table, and other statistics. **This How**.

Selection of Variables

- Simple algorithm: each variable is by itself in a separate term. Model for M variables has M terms.
- Extreme selection. Each term has M variables/factors. Model involves M squared variables / factor.
- Stochastic selection. Each term selects at least one of the M variable at random, forming the first factor in the term. The other variables are selected at random, and **may or may not be included** as a new factor, based of comparing a uniform random number with a fixed or diminishing probability of selection.

Example of Selection of Variables

- We have a matrix with 4 columns (i.e. variables x1 to x4).
- Select at random variable x3 first.
- Select variable x4, but the random value generated is greater than the critical selection probability. Thus, variable x4 does not enter in the next factor
- Select variable x1, and the random value generated is less than the critical selection probability. Thus, variable x1 enters in the next factor

Example of Selection of Variables

- Select variable x2, but the random value generated is greater than the critical selection probability. Thus variable x2 does not enter in the next factor
- The model includes a term with variables x3 and x1.
- We repeat te above process for the other terms.

- NOTE: The number of terms is not tied up to the number of variables. So, for example, we can have a regression model with one variable and four terms! We can also have a model with a single term and four independent variables.

Example of Selection of Variables

- SR models can be used in time series analysis when plotting y and normalized time values.. The tested predictions are good! SR models differ significantly from legacy ARMA time series models.

Building the Math Expressions for the Regression Model

HHC 2026

- Specify the number of terms in the regression models. These terms are added (and having negative regression coefficients injects subtraction).
- Each term has a number of factors that are multiplied.
- To implement divisions, use reciprocal functions, such as 1/x, 1/sin(x).
- Each factor selects an independent variable at random and also applies a transformation function, selected at random.
- The transformation functions can include constants that are selected at random (within a given range).

Building the Math Expressions for the Regression Model

HHC 2026

- One of the main issues is whether or not the regression models include transformations for the dependent variable y.
- Using variable y without transformation significantly avoids issues of inverse transformations that avoid generating complex values or erroneous results.
- Using transformed y values (with error checking and complex values detection) can greatly reduces the number of competing regression models.

Evaluating the Regression Models

HHC 2026

- Brute force evaluation goes through every possible regression model and guarantees to get the very best regression model. As the number of variables, terms, and transformatioms increase, computation time becomes a nightmare!
- Use random search for regression models. Enough iterations suprisingly yield good ***surrogate*** approximations—what I call the ***Irish Twins***.
- Random model search can benefit from parallel programming!

Selecting Operators & Functions

- Operators can be +,-, *, /, and ^. Factors used in stochastic symbolic regression use mainly *. The + operator adds terms to form a math expression.
- Simple, common functions with single parameters: sqrt(x), 1/x, x^2, log(x), ln(x), exp(x), and 10^x.
- Trigonometric and hyperbolic functions.
- More advanced functions like J(0,x), J(1,x), LagrangePoly(3,x), LegendrePoly(4,x), HermitePoly(4,x), and many many more

Selecting Operators & Functions

- You can build a long list of all possible simple and nested transformation functions.. You have a good idea about what possible models you can get.
- You can provide the program with a list of basic functions and ask it to create expressions with nested functions. May require a higher number of iterations. The resulting models are more stochastic.

Handling Outliers

- To handle noise and outliers in the data, replace the OLS calculations with robust regression algorithms such as Ridge Regression, Lasso regression, ElasticNet regression. Huber regression, and other variants.
- Robust regression methods are better equipped to handle outliers and yield less biased regression coefficients.

Selecting Operators & Functions

- Single nested functions like sin(sin(x)), sin(cos(x), sin(sqrt(x), and many many more.
- Double nested functions like sin(cos(sin(x))), sin(cos((log(x)), cos(sin(sqrt(x)), and many many more.
- The arguments for x , in the above functions, can be multiplied by a random value and shifted by a random value—param(2)+param(1)*x. So we can have for example sin(param(2)+param(1)*x), cos(param(2)+param(1)*x), tanh(param(2)+param(1)*x), sin(cos(param(2)+param(1)*x)), and many many more.

Selecting Operators & Functions

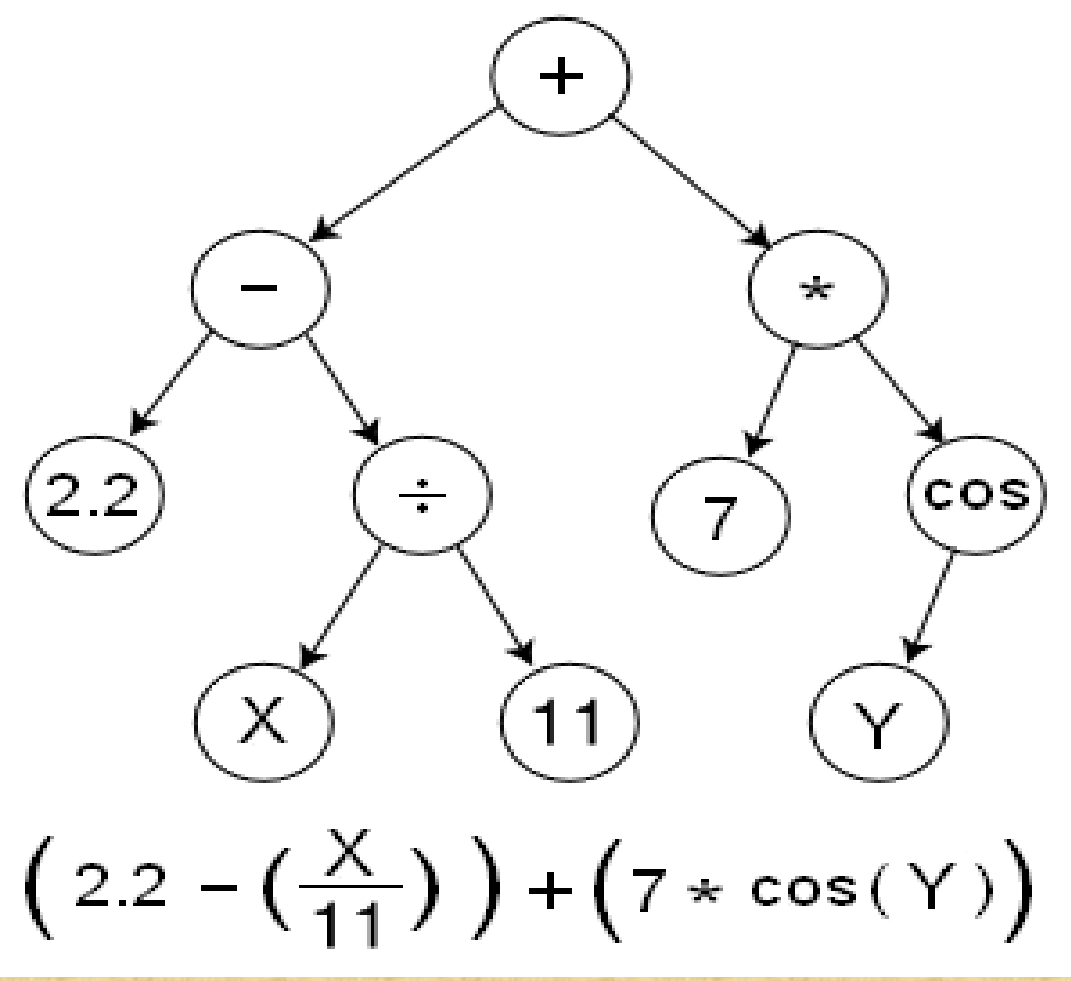
- The function list can be implemented as a long list from which elements are randomly selected.
- The function list can be divided into subgroups. The code will first choose a subgroup at random and then a function within that group at random.

Classic Genetic-Programming Based Symbolic Regression

HHC 2026

- Genetic Programming (GP) is a version of Genetic Algorithms that creates mathematical expressions based on a set of operators/functions and values (obtained from variables and random constants).
- GP builds a forest of expression trees whose roots and non-terminal nodes are operators or functions, and terminal nodes are references to variables or randomly selected constants.

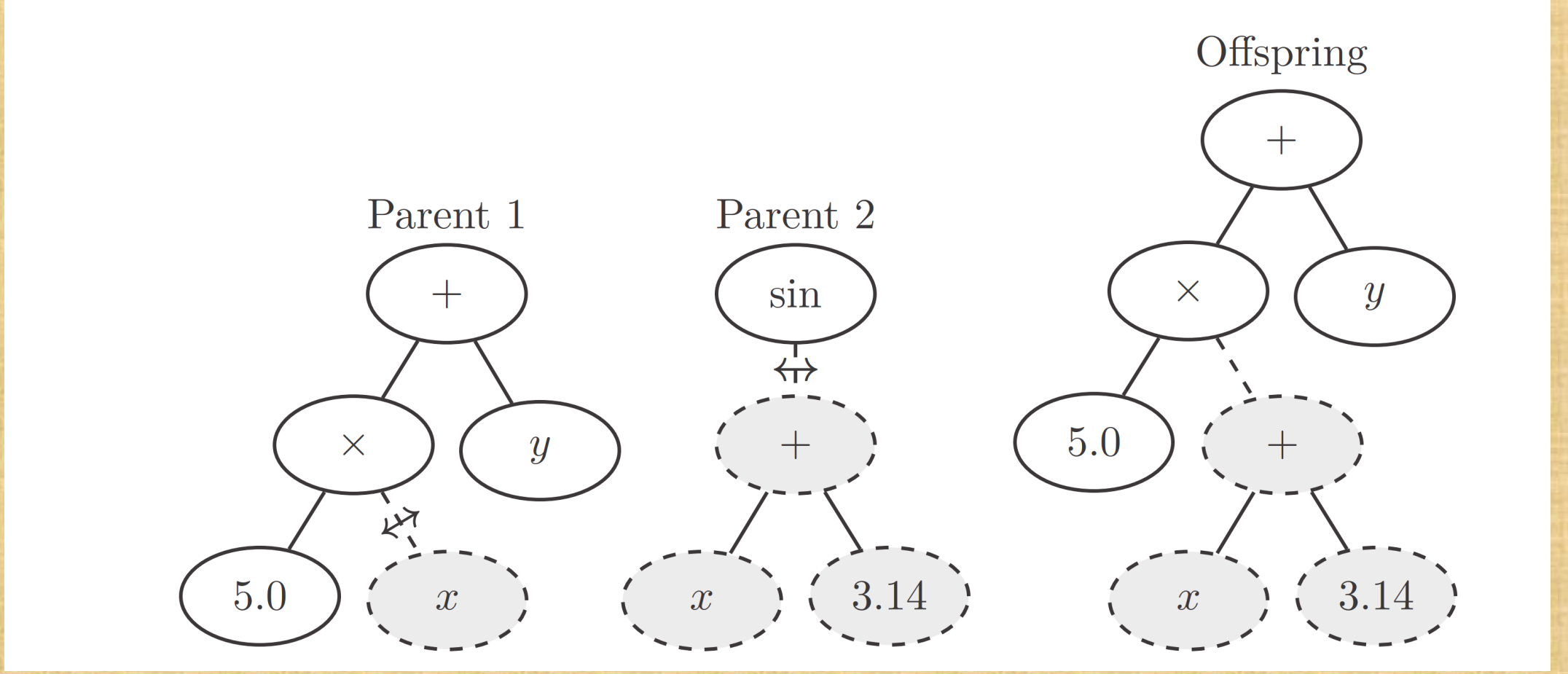
Sample Expression Tree



Genetic-Programming Trees

- GP trees are built by specifying the number of trees and the maximum tree depth.
- The GP algorithm build a forest of expression trees, selects random pairs to create children tree expressions.
- If the children tree expressions yield better solutions, they replace their parents.
- The new/old parents are ***possibly*** mutated to get better individuals. If these individuals are better than the parents, they replace the parents.

Sample Child Tree



Genetic-Programming Trees

- The algorithm fits the models of the updated parents with the X and y data to obtain the correlation statistics (such as the adjusted R-square value).
- The process is repeated for a specific number of iterations, time span, or both.
- The GP algorithm yields the math expression that has the best correlation statistics (such as the adjusted R-square value or the root mean error squared).
- **ACHTUNG**! GP-based symbolic regression can yield **huge expressions**! Do not lend themselves for hand calculations!

GP, NN, and ML

- GP has benefited from Machine Learning and Neural Networks (along with parallel processing).
  - The transformer pre-training pipeline symbolic regression algorithm.
  - The program-skeleton representation symbolic regression algorithm.
  - The Kolmogorov-Arnold Networks symbolic regression algorithm.

GP, Neural Networks, and ML

- The Deep Symbolic Regression (DSR) approach symbolic regression algorithm
- The Neuro-symbolic and DL-augmented GP symbolic regression algorithm.

Example

Given y = 5 + 2*x1 + 4*x2/sqrt(x3) with random values for x1, x2, and x3, obtain the best models with 2, 3, and 4 terms. Each model has up to 3 factors. Test models are error-free to check model fitness.

After obtaining the best model for each term, we supply a new (and smaller) set of random test values for x1, x2, and x3. We use the best models to calculate y^ and compare it with the calculate y for the test data.

Test Set 1

```
============================================================
Set 1  (3 vars):  y = 5 + 2*x1 + 4*x2/sqrt(x3)
============================================================
MaxTerms = 2, P (max factors) = 3, varSelectConst = 0.75, MaxIters = 200000

Best model (found at iteration 79819 of 200000 valid candidates):
  y = 13.5691 + (-0.0411653)*((0.77337) + (-10.631)*x1 + (-3.3672)*x1^2) + (0.0600321)*((-1.8173) + (9.8974)/sqrt(x3) +
(0.69981)/x3)*((-12.41) + (-13.855)*sqrt(x2) + (13.291)*x2)

Model statistics (training, N = 500):
   Terms         : 2
   Max factors    : 3
   R-square      : 0.969668
   Adjusted R-sq  : 0.969546
   RMSE          : 1.37556
   Elapsed time   : 36.6 s

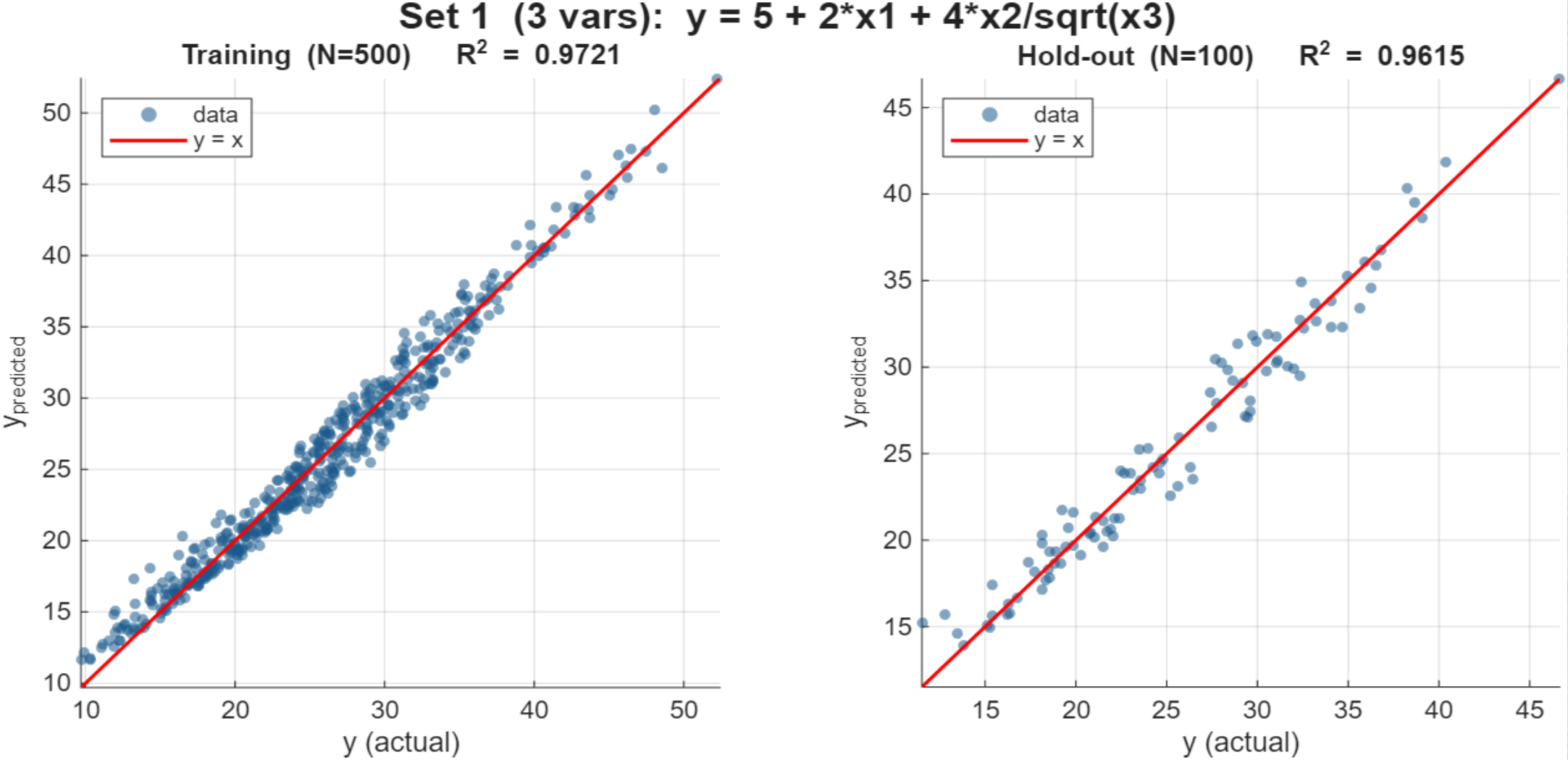
Model statistics (hold-out, N = 100):
   R-square      : 0.963455
   Adjusted R-sq  : 0.962701
   RMSE          : 1.45885
```

Test Set 1

y = 5 + 2*x1 + 4*x2/sqrt(x3)

Best model (found at iteration 79819 of 200000 valid candidates):
y = 13.5691 + (-0.0411653)*((0.77337) + (-10.631)*x1 + (-3.3672)*x1^2) + (0.0600321)*((-1.8173) + (9.8974)/sqrt(x3) + (0.69981)/x3)*((-12.41) + (-13.855)*sqrt(x2) + (13.291)*x2)

Test Set 1 Graphs



Test Set 2

==========

Set 1 (3 vars): y = 5 + 2*x1 + 4*x2/sqrt(x3)

==========

MaxTerms = 3, P (max factors) = 3, varSelectConst = 0.75, MaxIters = 200000

Best model (found at iteration 87689 of 200000 valid candidates):

y = 8.80191 + (0.155895)*sin((-12.271)*cos(sin((10.802 -10.746*x2))))*cos((-3.4134)*sinh((-2.1752 +14.865*x3))) + (-0.712383)*(-6.8109 -8.27*x2)*1/cuberoot((-4.3209 +7.5761*x3)) + (0.00856244)*(-5.6125 +4.9767*x1)^2

Model statistics (training, N = 500):

Terms : 3

Max factors : 3

R-square : 0.969280

Adjusted R-sq : 0.969095

RMSE : 1.4299

Elapsed time : 53.0 s

Model statistics (hold-out, N = 100):

R-square : 0.970273

Adjusted R-sq : 0.969344

RMSE : 1.40517 RMSE : 1.45885

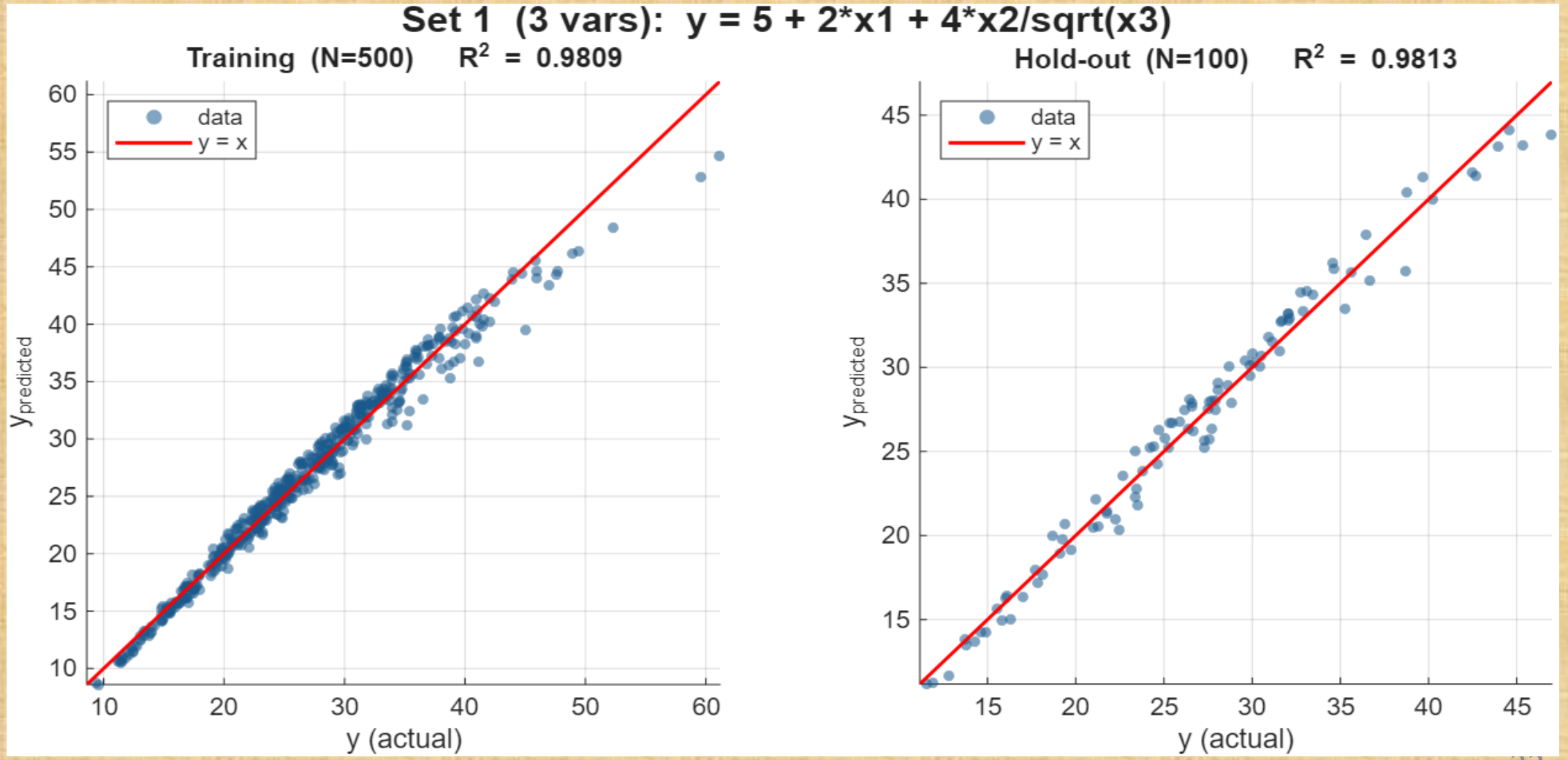
Test Set 2

y = 5 + 2*x1 + 4*x2/sqrt(x3)

Best model (found at iteration 87689 of 200000 valid candidates):

y = 8.80191 + (0.155895)*sin((-12.271)*cos(sin((10.802 -10.746*x2))))*cos((-3.4134)*sinh((-2.1752 +14.865*x3))) + (-0.712383)*(-6.8109 - 8.27*x2)*1/cuberoot((-4.3209 +7.5761*x3)) + (0.00856244)*(-5.6125 +4.9767*x1)^2

Test Set 2 Graphs



Test Set 3

```
============================================================
Set 1  (3 vars):  y = 5 + 2*x1 + 4*x2/sqrt(x3)
============================================================
MaxTerms = 4, P (max factors) = 3, varSelectConst = 0.75, MaxIters = 200000

Best model (found at iteration 142016 of 200000 valid candidates):
  y = 10.6277 + (-0.454105)*cos((0.37917)*exp((-13.167 +7.1876*x1))) + (-0.015489)*((8.7874) + (-2.6282)*x1 + (-10.977)*x1^2)
+ (0.0581169)*((-1.1312) + (8.5138)/sqrt(x3) + (-2.7012)/x3)*((-13.853) + (11.82)*sqrt(x2) + (10.936)*x2) + (0.00654475)*sin((-
12.642)*lagrangePoly(4,(-9.0198 +12.274*x3)))

Model statistics (training, N = 500):
   Terms         : 4
   Max factors    : 3
   R-square       : 0.979697
   Adjusted R-sq  : 0.979533
   RMSE           : 1.1209
   Elapsed time   : 70.6 s

Model statistics (hold-out, N = 100):
   R-square       : 0.979908
   Adjusted R-sq  : 0.979062
   RMSE           : 1.14513
```

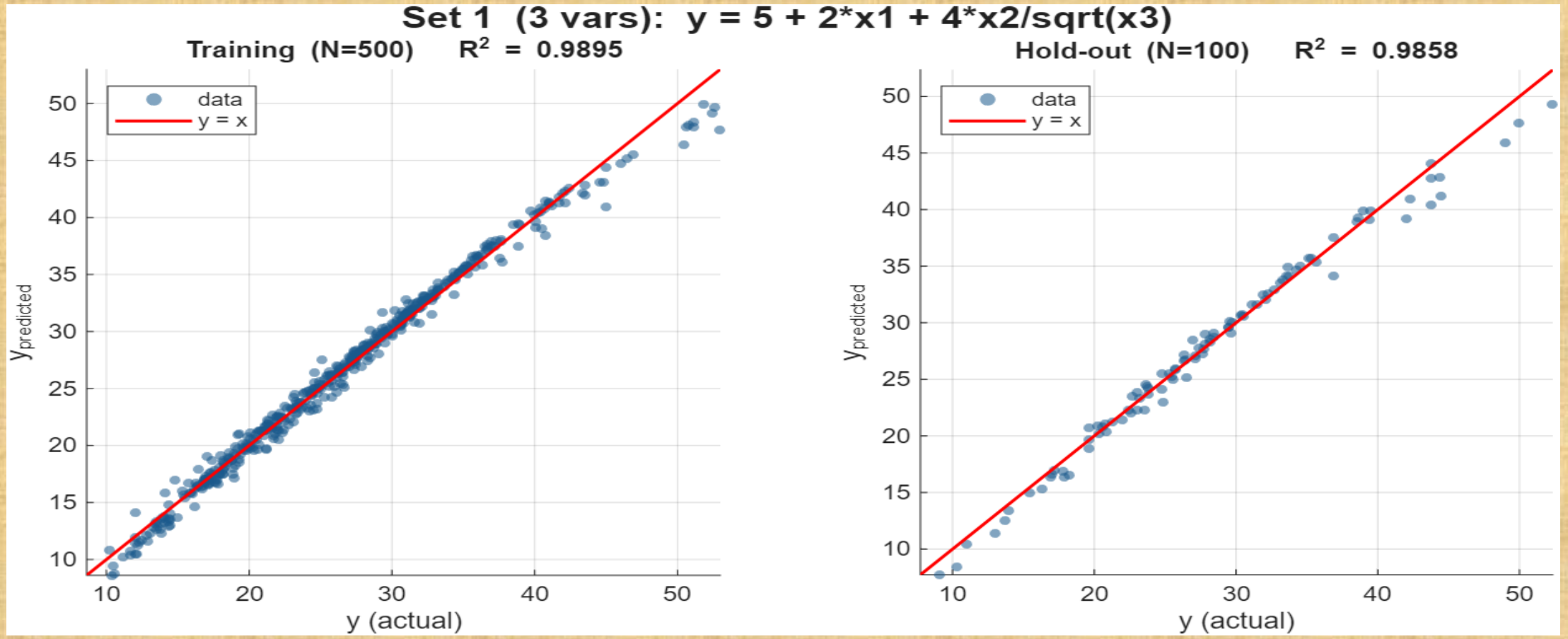
Test Set 3

y = 5 + 2*x1 + 4*x2/sqrt(x3)

Best model (found at iteration 87689 of 200000 valid candidates):

y = 10.6277 + (-0.454105)*cos((0.37917)*exp((-13.167 +7.1876*x1))) + (-0.015489)*((8.7874) + (-2.6282)*x1 + (-10.977)*x1^2) + (0.0581169)*((-1.1312) + (8.5138)/sqrt(x3) + (-2.7012)/x3)*((-13.853) + (11.82)*sqrt(x2) + (10.936)*x2) + (0.00654475)*sin((-12.642)*lagrangePoly(4,(-9.0198 +12.274*x3)))

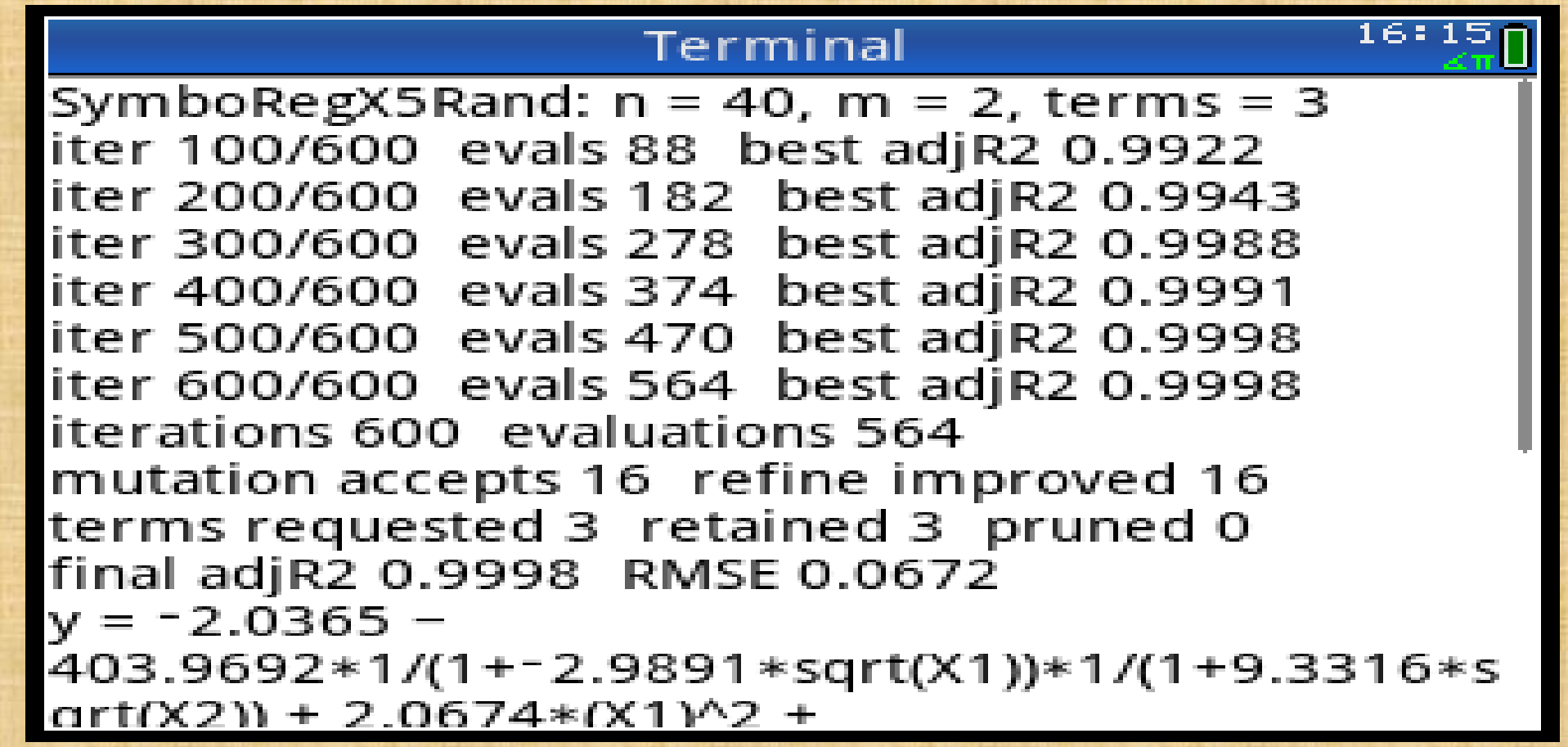
Test Set 3 Graphs



HP Prime Program

Conference files include TWO HP Prime programs, with documentation, that performs Symbolic regression!

HP Prime Program



HP Prime Program

Terminal 16:17

```
final adjR2 0.9998  RMSE 0.0672
y = -2.0365 -
403.9692*1/(1+-2.9891*sqrt(X1))*1/(1+9.3316*s
qrt(X2)) + 2.0674*(X1)^2 +
23679.2646*1/(1+-34.2087*X1)*1/(1+47.6299*s
qrt(X2))
R2 0.9998  adjR2 0.9998  F 68106.5
AIC -98.7374  BIC -91.9819  RMSE 0.0672
Intercept: -2.0365  se 0.0619  p 0
1/(1+-2.9891*sqrt(X1))*1/(1+9.3316*sqrt(X2)):
-403.9692  se 7.8762  p 0
(X1)^2: 2.0674  se 0.0072  p 0
1/(1+-34.2087*X1)*1/(1+47.6299*sqrt(X2)):
23679.2646  se 545.8072  p 0
```

Notes

I used Claude AI a lot in generating most of the code for the various symbolic regression functions in MATLAB, Python, and Excel VBA.

For documentation purpose, I asked Claude AI to generate high-level, medium-level, and low-level pseudo-code for the SR code.

Conclusion

- Symbolic regression is a very powerful tool.
- Modeling requires several runs with different operational parameters.
- Stochastic SR does well and allows you to control the make up of the set of tested regression models and get the best regression model.
- GP-based SR is more evolutionary and can yield bigger regression models

Conclusion (Cont)

- AI-based SR yield more advanced regression models and and can also generate big regression models

Thank You!