

Best Rational Heteronomial Model Selection Part I

by
Namir Clement Shammas

Contents

Introduction	1
What are Heteronomials Good For?	4
The Case of One Independent Variable	5
Models for Rational Heteronomials of Order Two and Up	17
Handling <i>Flexible</i> Rational Heteronomial Models	17
Case 1	31
Case 2	36
Case 3	42
Case 4	47
Closing Remarks for Part I	55
About the ZIP File for this study	55

Introduction

Polynomials are popular constructs in math, calculus and regression analysis. Other popular constructs in regression analysis belong to multiple linear(ized) models. A simple example of a multiple regression model with two independent variables is:

$$Y = a + b_1 * X_1 + b_2 * X_2 \quad (1)$$

I will call the above model a linear *heteronomial*. It is linear because all the variables are in linear form. A common example of a nonlinear heteronomial is:

$$\ln(Y) = a + b_1 * \ln(X_1) + b_2 * \ln(X_2) \quad (2)$$

The above model is a multivariate power model.

The general form of a heteronomial is:

$$Y^{yp} = a + b_1 * X_1^{px1} + b_2 * X_2^{px2} + \dots + X_n^{pxn} \quad (3)$$


The above heteronomial shows that each variable in the model can have a power other than 1. The powers can be negative, zero (special code for using the function $\ln(x)$), and positive. The non-zero powers can be integers or floating-point numbers. Using floating point negative powers requires the data to be positive.

I can generalize equation (3) further by using the following form:

$$f(Y, yp) = a + b_1 * f(X_1, px1) + b_2 * f(X_2, px2) + \dots + b_n * f(X_n, pxn) \quad (4)$$

$$\begin{aligned} \text{Where } f(x, p) &= x^p \text{ when } p \neq 0 \\ &= \ln(x) \text{ when } p = 0 \end{aligned} \quad (5)$$

This three-part study will focus on heteronomials of various kinds. In this first part we examine the above kind of power transformations. Using other functions like trigonometric and hyperbolic functions requires a more elaborate power-coding scheme to map special powers to special functions. The power management scheme uses distinct initial powers, power increments, and final powers for each variable.

 Please note that the total number of models tested is the product of the total number of transformations for each variable. The total number of models can easily increase to very high values with the increase in the number of transformations for each variable.

Padé polynomials take the ratio of two polynomials. The general form of a Padé polynomial is:

$$Y = (a + b_1 * X + b_2 * X^2 + \dots + b_m * X^m) / (1 + c_1 * X + c_2 * X^2 + \dots + c_n * X^n) \quad (6)$$

Notice that the intercept of the denominator polynomial is always 1. The Padé polynomials have orders of m and n which may or may not be equal. Moreover, n may be greater than m, or vice versa.

Now let me introduce you to **rational heteronomials**. They are to heteronomials what Padé polynomials are to regular polynomials. The general form of rational heteronomials (with no cross-product terms) is:

$$Y^{py} = (a + b_1 * X_1^{px11} + b_2 * X_2^{px12} + \dots + b_n * X_n^{px1n}) / (1 + c_1 * X_1^{px21} + c_2 * X_2^{px22} + \dots + c_m * X_n^{px2m}) \quad (7)$$

Each variable in equation (7) has its own power. In fact, equation (7) shows the general form of *symmetrical rational heteronomials* that have the same independent variables appearing in the numerator and denominator, albeit being raised to different powers. The minimal condition for rational heteronomials is that **at least one independent variable must appear in both numerator and denominator**.

The linearized multiple regression model based on equation (7) is as follows.

$$Y^{py} = a + b_1 * X_1^{px11} + b_2 * X_2^{px12} + \dots + b_n * X_n^{px1n} - c_1 * Y^{py} * X_1^{px21} - c_2 * Y^{py} * X_2^{px22} - \dots - c_n * Y^{py} * X_n^{px2n} \quad (7b)$$

Notice that the terms in the denominator of equation (7) are now multiplied by Y^{py} and these terms are also **subtracted** from the terms in the numerators of equation (7).

A few general examples of non-symmetrical rational heteronomials appear next:

$$Y^{py} = (a + b_1 * X_1^{px11} + b_2 * X_2^{px12} + \dots + b_n * X_n^{px1n}) / (1 + c_1 * X_1^{px21} + c_2 * X_2^{px22} + \dots + c_m * X_m^{px2m}) \quad (7b)$$

$$Y^{py} = (a + b_1 * X_1^{px11} + b_2 * X_2^{px12} + \dots + b_n * X_n^{px1n}) / (1 + c_1 * X_1^{px21} + c_2 * X_3^{px23} + \dots + c_m * X_m^{px2m}) \quad (7c)$$

$$Y^{py} = (a + b_1 * X_1^{px11} + b_2 * X_2^{px12} + \dots + b_n * X_n^{px1n}) / (1 + c_1 * X_1^{px21} + c_2 * X_5^{px25} + \dots + c_m * X_m^{px2m}) \quad (7d)$$

$$Y^{py} = (a + b_1 * X_1^{px11} + b_2 * X_5^{px15} + \dots + b_n * X_n^{px1n}) / (1 + c_1 * X_1^{px21} + c_2 * X_3^{px23} + \dots + c_m * X_m^{px2m}) \quad (7e)$$

Selecting the best combination of variables in the numerator and denominator can be very tedious and more complicated than determining the best orders n and m for Padé polynomials. For example, given two independent variables, one must search through the best model in the following sets of general models:

$$Y^{py} = (a + b_1 * X_1^{px11} + b_2 * X_2^{px22}) / (1 + c_1 * X_1^{px21} + c_2 * X_2^{px22}) \quad (8a)$$

$$Y^{py} = (a + b_1 * X_1^{px11}) / (1 + c_1 * X_1^{px21} + c_2 * X_2^{px22}) \quad (8b)$$

$$Y^{py} = (a + b_1 * X_1^{px11} + b_2 * X_2^{px22}) / (1 + c_1 * X_1^{px21}) \quad (8c)$$

$$Y^{py} = (a + b_1 * X_2^{px12}) / (1 + c_1 * X_1^{px21} + c_2 * X_2^{px22}) \quad (8d)$$

$$Y^{py} = (a + b_1 * X_1^{px11} + b_2 * X_2^{px22}) / (1 + c_1 * X_2^{px22}) \quad (8e)$$

So, two independent variables will require you to find the best fit in five sets of rational heteronomials! The case of three independent variables will require you to find the best fit in 13 sets of rational heteronomials. And in the case of 4 independent variables, one must find the best fit in 29 sets of rational heteronomials. It's easy to see how the number of sets keep climbing as the number of independent variables increases! The search will require several functions that tests various kinds of models. For example, in the above case of two independent variables, we need three functions to determine the best of 5 sets of models.

This study looks at the selection of the best rational heteronomials models using MATLAB and Excel data files. To initially simplify things, I will choose the orders of these rational heteronomials so that the numerator and denominator parts use terms with the same sets of dependent variables. So, this study will look at rational heteronomials of orders (1, 1), (2, 2) and (3, 3). The function is written in MATLAB code.

What are Heteronomials Good For?

Aside from commonly used heteronomials in equations (1) and (2), what are other heteronomials good for? They help us search for **good new empirical relationships** between variables—ones that theoretical analysis would not easily deduce or derive.

Readers may ask about using optimization to zoom in on the best powers of heteronomials. While optimization is a good alternative, it does not easily lend itself to working with the $\ln(x)$ transformations. In addition, the best powers calculated using optimization would lack simpler power patterns that I use in this entire study. And finally, using optimization may not give you the exact same results every time—it is not deterministic, while the approach in this study is!

The Case of One Independent Variable

Let's start with the simple case of the regression model shown next:

$$Y^{py} = (a + b_1 * X_1^{px11}) / (1 + c_1 * X_1^{px21}) \quad (9)$$

The Excel sheet bestrathet2.xlsx is the one I use to handle modeling with equation (9). The workbook has the following sheets:

- The **Data** sheet has the following columns (see Figure 1):
 - Column A contains the values for the dependent variable Y. The column has the header Y in the first row. The header's text does not affect the calculations, so you can customize it.
 - Column B contains the values for the independent variable X. The column has the header X in the first row. In the case of multiple independent variables, the values for the second independent variable appear in column 3, the values for the third independent variable in appear column 4, and so on. The headers' texts do not affect the calculations, so you can customize it.
- The **Transf** sheet contains the information for the transformation ranges. It has the following columns (see Figure 2):
 - Column A has the header Y in row 1. Rows 2, 3, and 4 contain the values for the initial power, power increment, and final power used with variable Y.
 - Column B has the header X in row 1. Rows 2, 3, and 4 contain the values for the initial power, power increment, and final power used with variable X in the numerator part of the rational heteronomial.
 - Column C has the header YX in row 1. Rows 2, 3, and 4 contain the values for the initial power, power increment, and final power used with variable X in the denominator part of the rational heteronomial.
 - Column D has the header **Min Adj R-sqr** in row 1. Row 2 contains the minimum value for the adjusted R-square statistic used to qualify a model to be listed in the sheet **List**. This filtering helps reduce the number of results in sheet **List** by eliminating uninteresting models. If you wish to view all the results, set the minimum value to zero.
- Sheet **List** displays the list of models with qualifying adjusted R-square value (as appearing in cell D2 of the **Transf** sheet). Figure 3 shows a partial view of the sorted columns. The sheet has the following columns:
 - Column A displays the values for the adjusted R-square statistic.

- Column B displays the values for powers of variable Y.
- Column C displays the values for powers of variable X in the numerator part of the rational heteronomial.
- Column D displays the values for powers of variable X in the denominator part of the rational heteronomial.
- Column E displays the values for the intercepts.
- Column F displays the values for the slopes of variable X in the numerator part of the rational heteronomial.
- Column G displays the values for the slopes of variable X in the denominator part of the rational heteronomial.
- The **Project** sheet lists the variables in the **Data** sheet along with two additional columns. The first one has the header YCalc and contains the projected values for variable Y. The second column shows the percent error in the projected values of variable Y.
- The sheets **List2** and **Project2** appear if the best power for variable Y is not 1. The function redoes the calculations with untransformed values of Y.

Y	X
6	1
5.31554498	2
6.67107501	3
8.80025547	4
11.4967288	5
14.686079	6
18.330498	7
22.4066601	8
26.8983294	9
31.7932762	10

Figure 1. The **Data** sheet.

Y	X	YX	Min Adj R-sqr
-3	-3	-3	0.999

1	1	1
3	3	3

*Figure 2. The **Transf** sheet.*

Adj R-sqr	Ypwr	X1Pwr	X2Pwr	Intrecept	X1	X2
1	-1	0	2	0.2	0.2	0.2
1	1	2	0	5	1	1
0.999945494	3	3	0	67.09466808	5.008571411	-0.36636581
0.999849485	0	-2	-1	6.780787386	12.85148667	9.957905934
0.99984136	2	3	1	24.92086096	0.733977315	-0.024999591
0.999812835	2	3	2	24.75620119	0.83464345	-0.001510799
0.999802677	3	2	0	-38.55628074	29.32739712	-0.396782721
0.999796924	2	3	0	24.68662465	0.542149084	-0.190856911
0.999778971	3	3	1	-18.68240413	9.189479445	-0.071819608
0.999775966	2	3	3	23.91202119	0.873026785	-0.000114353

Figure 3. The *List* sheet (partial view).

Here is the MATLAB function bestrathet2() for the calculations that determine the best rational heteronomial model in equation (9).

```
function [mdl1, mdl2, bestPwrs1, bestPwrs2] = bestrathet2(XlFile,outFile)

warning('off');
format long

dt = datetime('now','TimeZone','local','Format','y-MMM-d HH_mm_ss');
dtstr = string(dt);
if nargin < 2, outFile = strcat("bestathet2_",dtstr,".txt"); end
if nargin < 1, XlFile = strcat(pwd,'\bestathet2.xlsx'); end

diary(outFile)

fprintf("Function bestrathet2\n\n");
fprintf("%s\n", dtstr);
fprintf("Please wait ....\n\n");

fprintf("Excel file used is %s\n", XlFile);
fprintf("Diary file used is %s\n", outFile);
% Open Excel as a COM Automation server
Excel = actxserver('Excel.Application');
% Open Excel workbook
Workbook = Excel.Workbooks.Open(XlFile);
% Clear the content of the sheet
Workbook.Worksheets.Item('List').Range('A2:Z100000').ClearContents;
Workbook.Worksheets.Item('List2').Range('A2:Z100000').ClearContents;
Workbook.Worksheets.Item('Project').Range('A2:Z100000').ClearContents;
Workbook.Worksheets.Item('Project2').Range('A2:Z100000').ClearContents;
% Now save/close/quit
Workbook.Save;
Excel.Workbook.Close;
invoke(Excel, 'Quit');

hdrList = {'Adj R-square','YPwr','X1Pwr','YX1Pwr','Incercept','X1','YX1'};
writecell(hdrList,XlFile,'Sheet','List');
writecell(hdrList,XlFile,'Sheet','List2');
hdrProject = {'Y','X1','YCalc','%Err'};
writecell(hdrProject,XlFile,'Sheet','Project');
```



```

writecell(hdrProject,XlFile,'Sheet','Project2');

xyData = readmatrix(XlFile,'Sheet','Data');
[maxrows,~] = size(xyData);
ydata = xyData(:,1);
xdata = xyData(:,2);
xyTransf = readmatrix(XlFile,'Sheet','Transf');
yPwrs = xyTransf(1:3,1);
x1Pwrs = xyTransf(1:3,2);
x2Pwrs = xyTransf(1:3,3);
minAdjR2 = xyTransf(1,4);
yBuffer = zeros(maxrows,1);
x1Buffer = zeros(maxrows,1);
x2Buffer = zeros(maxrows,1);
yCols = 0;
for yPwr =yPwrs(1):yPwrs(2):yPwrs(3)
    yCols = yCols + 1;
    yBuffer(:,yCols)= fx(ydata,yPwr);
end

x1Cols = 0;
for x1Pwr =x1Pwrs(1):x1Pwrs(2):x1Pwrs(3)
    x1Cols = x1Cols + 1;
    x1Buffer(:,x1Cols)= fx(xdata,x1Pwr);
end

x2Cols = 0;
for x2Pwr =x2Pwrs(1):x2Pwrs(2):x2Pwrs(3)
    x2Cols = x2Cols + 1;
    x2Buffer(:,x2Cols)= fx(xdata,x2Pwr);
end

bestAdjR2 = 0;
gRow = 0;
xyList = zeros(1,7);

yCols = 0;
for yPwr =yPwrs(1):yPwrs(2):yPwrs(3)
    yCols = yCols + 1;
    y = yBuffer(:,yCols);

    x1Cols = 0 ;
    for x1Pwr =x1Pwrs(1):x1Pwrs(2):x1Pwrs(3)
        x1Cols = x1Cols + 1;
        x1 = x1Buffer(:,x1Cols);

        x2Cols = 0 ;
        for x2Pwr =x2Pwrs(1):x2Pwrs(2):x2Pwrs(3)
            x2Cols = x2Cols + 1;
            x2 = x2Buffer(:,x2Cols);

            x2 = -1 .* y .* x2;
            X = [x1 x2];

            mdl = fitlm(X,y);

            if mdl.Rsquared.Adjusted >= bestAdjR2

```

```

        bestAdjR2 = mdl.Rsquared.Adjusted;
        bestYpwr = yPwr;
        bestX1pwr = x1Pwr;
        bestX2pwr = x2Pwr;
        bestmdl = mdl;
    end

    if mdl.Rsquared.Adjusted >= minAdjR2
        gRow = gRow + 1;
        xyList(gRow,1) = mdl.Rsquared.Adjusted;
        xyList(gRow,2) = yPwr;
        xyList(gRow,3) = x1Pwr;
        xyList(gRow,4) = x2Pwr;
        xyList(gRow,5)= mdl.Coefficients{1,1};
        xyList(gRow,6)= mdl.Coefficients{2,1};
        xyList(gRow,7)= mdl.Coefficients{3,1};
    end
end
end
end
mdl = bestmdl;
bestPwrs1 = [bestYpwr bestX1pwr bestX2pwr];
bestPwrs2 = [bestYpwr bestX1pwr bestX2pwr]; % default assignement
[numRes,~] = size(xyList);
xyList = sortrows(xyList,1,"descend");
writematrix(xyList, XlFile, 'Sheet', 'List',
'Range',strcat('A2:G',num2str(numRes+1)));

% calculate projecttions
a = mdl.Coefficients{1,1};
b = mdl.Coefficients{2,1};
c = mdl.Coefficients{3,1};

ycalc = (a + b.*fx(xdata,bestX1pwr))./(1 + c.*fx(xdata,bestX2pwr));
if bestYpwr > 0
    ycalc = ycalc.^(1/bestYpwr);
elseif bestYpwr < 0
    ycalc = ycalc.^abs(bestYpwr);
else
    ycalc = exp(ycalc);
end
percErr = 100.*(ycalc-ydata)./ydata;
ProjMat = [ydata xdata ycalc percErr];
writematrix(ProjMat, XlFile, 'Sheet', 'Project', 'Range', strcat("A2:Z",
num2str(1+length(ydata))));

% Print the summary of the best regression results
fprintf("Summary:\n");
fprintf("Adj R-sqr = %14.12f", bestAdjR2);
mdl
mdl1 = mdl;
mdl2 = mdl; % default assignement
fprintf("Best Y power = %f\n", bestYpwr);
fprintf("Best numerator X power = %f\n", bestX1pwr);
fprintf("Best denominator X power = %f\n", bestX2pwr);

if bestYpwr ~= 1

```

```

fprintf("=====\n");
fprintf("\n\n----- Phase 2 ----- \n\n");
fprintf("=====\n");

bestAdjR2 = 0;
gRow = 0;
xyList = zeros(1,7);

y = ydata;

x1Cols = 0 ;
for x1Pwr =x1Pwrs(1):x1Pwrs(2):x1Pwrs(3)
    x1Cols = x1Cols + 1;
    x1 = x1Buffer(:,x1Cols);

    x2Cols = 0 ;
    for x2Pwr =x2Pwrs(1):x2Pwrs(2):x2Pwrs(3)
        x2Cols = x2Cols + 1;
        x2 = x2Buffer(:,x2Cols);

        x2 = -1 .* y .* x2;
        X = [x1 x2];

        mdl = fitlm(X,y);

        if mdl.Rsquared.Adjusted >= bestAdjR2
            bestAdjR2 = mdl.Rsquared.Adjusted;
            bestYpwr = 1;
            bestX1pwr = x1Pwr;
            bestX2pwr = x2Pwr;
            bestmdl = mdl;
        end

        if mdl.Rsquared.Adjusted >= minAdjR2
            gRow = gRow + 1;
            xyList(gRow,1) = mdl.Rsquared.Adjusted;
            xyList(gRow,2) = 1;
            xyList(gRow,3) = x1Pwr;
            xyList(gRow,4) = x2Pwr;
            xyList(gRow,5) = mdl.Coefficients{1,1};
            xyList(gRow,6) = mdl.Coefficients{2,1};
            xyList(gRow,7) = mdl.Coefficients{3,1};
        end
    end
end
mdl = bestmdl;
bestPwrs2 = [bestYpwr bestX1pwr bestX2pwr];
[numRes,~] = size(xyList);
xyList = sortrows(xyList,1,"descend");
writematrix(xyList, XlFile, 'Sheet', 'List2',
'Range',strcat('A2:G',num2str(numRes+1)));

% calculate projecttions
a = mdl.Coefficients{1,1};
b = mdl.Coefficients{2,1};
c = mdl.Coefficients{3,1};

```

```

    ycalc = (a + b.*fx(xdata,bestX1pwr))./(1 + c.*fx(xdata,bestX2pwr));
    percErr = 100.*(ycalc-ydata)./ydata;
    ProjMat = [ydata xdata ycalc percErr];
    writematrix(ProjMat, XlFile, 'Sheet', 'Project2', 'Range', strcat("A2:Z",
num2str(1+length(ydata))));

    % Print the summary of the best regression results
    fprintf("Summary:\n");
    fprintf("Adj R-sqr = %14.12f", bestAdjR2);

    mdl
    mdl2 = mdl;
    fprintf("Best Y power = %f\n", bestYpwr);
    fprintf("Best numerator X power = %f\n", bestX1pwr);
    fprintf("Best denominator X power = %f\n", bestX2pwr);
end

format short
diary off
end

function y = fx(x,pwr)
    if pwr > 0
        y = x.^pwr;
    elseif pwr < 0
        y = 1./x.^abs(pwr);
    else
        y = log(x);
    end
end
end

```

The above function has two parameters:

- The parameter XlFile specifies the Excel file that supplies the data and the information for the transformations. The file also receives output from the function.
- The parameter outFile is the name of the diary text file. It contains the echo of the console output.

The function has the following output parameters:

- The parameter mdl1 is the regression results object.
- The parameter mdl2 is the regression results object when the best power for the dependent variable is NOT 1. Otherwise, mdl2 is a copy of mdl1.
- The bestPwrs1 is the array of powers for all the variables.
- The bestPwrs2 is the array of powers for all the variables when the best power for the dependent variable is NOT 1. Otherwise, bestPwrs2 is a copy of bestPwrs1.

The function performs the following general tasks:

- Clear sheets **List**, **List2**, **Project**, and **Project2**.
- Read the data matrix is sheet **Data** and store it in variable xyData. This task uses the MATLAB function readmatrix().
- Determine the number of data points (i.e. rows) in the variable xyData.
- Store the data for variable Y in variable ydata.
- Store the data for variable X in variable xdata. In the case of multiple independent variables, this step stores their data in matrix xdata.
- Obtain and store the ranges of transformations for the variables X and Y in variables xyTransf, using the MATLAB function readmatrix(). Then copy the columns of matrix xyTransf into separate variables. The variable x1Pwrs stores the powers for the numerator part. The variable x2Pwrs stores the powers for the denominator part. Also obtain the minimal adjusted R-square value.
- Initialize the buffer matrices yBuffer, x1Buffer, and x2Buffer. The function calculates once the various transformations for the different variables and stores the in these buffers. The function recalls specific columns in each buffer as needed later in the calculations of the regression models.
- Initialize the best adjusted R-square value, the results matrix xyList, and the number of rows in that matrix.
- Use three for loops to calculate the data in the three matrix buffers used.
- Start the main process using three nested loops to access the transformed values of Y and X from the buffers.
- Create the regression matrix X and the vector column y needed to perform the multiple linearized regression using the MATLAB function fitlm(). As the number of independent variables in the numerator and denominator increases, the number of columns in matrix X increases accordingly.
- Determine if the new model object has a better adjusted R-square value than the one stored in variable bestAdjR2. If this condition is true, the function stores the adjusted R-square value, the powers for X and Y, and the model object.
- Determine if the new model object has an adjusted R-square value that is greater or equal to the one stored in variable minAdjR2. If this condition is true, the function lists the current model results in the next available row of matrix xyList.
- Sort the rows of matrix xyList using the values of the adjusted R-square statistic that are stored in column 1 of the matrix.

- Populate sheet **List** with the results from matrix xyList.
- Calculate the projected values of Y and their percent errors. The function copies the input data into sheet **Project**. In addition, it writes the projected values of Y and their percent errors in that sheet.
- Display on the console the summary of the best regression model object along with the best powers.
- Test if the best power for variable Y is not 1. If this condition is true, the code redoes the above calculations while using the variable Y without any power transformations. The results of this second run appear on the console and are also stored in sheets **List2** and **Project2**. You can compare sheets **List** and **List2**, and sheets **Project** and **Project2** to see which best model prevails.

Here is the test program test_bestrathet2.m to test function bestrathet2() using the data in figures 1 and 2.

```
clc
close all
clear all;

dt = datetime('now','TimeZone','local','Format','y-MMM-d HH_mm_ss');
dtstr = string(dt);
outFile = strcat("besttrathet2_",dtstr,".txt");
XlFile = strcat(pwd,'\besttrathet2.xlsx');

% mdl1 and mdl2 are saved for future calculatrions done by the user
[mdl1, mdl2, bestPwrs1, bestPwrs2] = bestrathet2(XlFile,outFile);
```

The program is basically very short. It sets up the values for parameters XlFile and outFile before calling function bestrathet2().

Before you run any MATLAB test program, open the Excel file bestrathet2.xlsx (or any other Excel file mentioned in that test program). If needed, enter or edit the data in sheets **Data** and **Transf**. Once you are done you **must close the Excel file** before you run the MATLAB test program. Examine the results in the MATLAB console as well as those in sheets **List** and **Project** (and maybe also sheets **List2** and **Project2** if they are not mostly clear) of the Excel workbook. Figure (3) shows the contents of that sheet and the list of the best model. Figure (4) shows the input data and the predicted values of Y and their percentage errors.

Adj R-sqr	Ypwr	X1Pwr	X2Pwr	Intrecept	X1	X2
1	-1	0	2	0.2	0.2	0.2
1	1	2	0	5	1	1
0.999945494	3	3	0	67.09466808	5.008571411	-0.36636581
0.999849485	0	-2	-1	6.780787386	12.85148667	9.957905934
0.99984136	2	3	1	24.92086096	0.733977315	-0.024999591
0.999812835	2	3	2	24.75620119	0.83464345	-0.001510799
0.999802677	3	2	0	-38.55628074	29.32739712	-0.396782721
0.999796924	2	3	0	24.68662465	0.542149084	-0.190856911
0.999778971	3	3	1	-18.68240413	9.189479445	-0.071819608
0.999775966	2	3	3	23.91202119	0.873026785	-0.000114353

Figure 3. The **List** sheet (partial view).

Y	X1	YCalc	%Err
6	1	6	-1.4803E-14
5.31554498	2	5.31554498	-3.34182E-14
6.67107501	3	6.67107501	-1.33139E-14
8.80025547	4	8.80025547	-2.01853E-14
11.4967288	5	11.4967288	-1.5451E-14
14.686079	6	14.686079	-2.4191E-14
18.330498	7	18.330498	0
22.4066601	8	22.4066601	0
26.8983294	9	26.8983294	0
31.7932762	#	31.7932762	-1.11744E-14

Figure 4. The **Project** sheet.

Here is the console output which is also copied to a diary file:

```
Function bestrathet2
```

```
2024-Dec-3 08_05_27
Please wait ....
```

```
1. Excel file used is C:\MATLAB\bestRatHet\bestRatHet2.xlsx
Diary file used is bestrathet2_2024-Dec-3 08_05_27.txt
Summary:
Adj R-sqr = 1.00000000000000
mdl =
```

Linear regression model:

$y \sim 1 + x1 + x2$

Estimated Coefficients:

	Estimate	SE	tStat	pValue
(Intercept)	5	0	Inf	0
x1	1	0	Inf	0
x2	1	0	Inf	0

Number of observations: 10, Error degrees of freedom: 7

R-squared: 1, Adjusted R-Squared: 1

F-statistic vs. constant model: 1.07e+31, p-value = 1.98e-107

Best Y power = 1.000000

Best numerator X power = 2.000000

Best denominator X power = 0.000000

The console output shows the regression table including the regression coefficients, their standard errors, the student-t values, and the p-Values. The latter values can shed valuable insight as to the eligibility of a variable/term to be part of the model. The best model is:

$$Y = (5 + X^2) / (1 + \ln(X))$$

The constant 1 in the denominator part is NOT calculated. It is an implicit part of the definition of the rational heteronomial. This fact is true for ALL rational heteronomials. Sheets **List** and **Project** confirms the above conclusion.

The MATLAB code contains the function `fx()` that transforms the value of a variable using a power value. The current implementation of function `fx()` handle negative, zero, and positive powers. Notice that the power zero causes function `fx()` to return the natural logarithm value of `x`.



Should you desire to use other functions for transformations, then you need to code function `fx()` to detect the supplied power values and use them to evaluate a special function. For example, if you are using the range of -3 to 3 in increments of 1 for regular powers, you can use the powers of 4 and 5 to evaluate, for example, the sine and cosine functions. The function `fx()` should use separate if statements to detect the value of 4 and 5 and return `sin(x)` and `cos(x)`, respectively. The code for function `fx()` would look like:

```
function y = fx(x,pwr):
```



```

if pwr == 4
    y = sin(x)
    return;
if pwr == 5
    y = cos(x);
    return;
if pwr > 0:
    y = x.^pwr;
elif pwr < 0:
    y = 1./x.^abs(pwr);
else:
    y = log(x);
end

```

As shown above, function `fx()` should detect the special coded powers first.

Models for Rational Heteronomials of Order Two and Up

The study includes versions of `bestrathet2.m` that work with rational heteronomials of orders 2 and 3. The code for these functions is very similar to that of `bestrathet2.py`. Of course, since these functions have two or more independent variables, the code has expanded parts to handle the additional variables. For example, the MATLAB code uses four or more nested loops to prepare the data for regression calculations. Likewise, the Excel workbooks have additional columns in the sheets mentioned earlier to accommodate the additional variables. In addition, the functions have additional *buffer matrices* to store transformed data.

Files `bestrathet4.m` and `test_bestrathet4.m` handle the case of two independent variables appearing on both the numerator and denominator of the rational heteronomials. Likewise, files `bestrathet6.m` and `test_bestrathet6.m` handle the case of three independent variables.

Handling *Flexible* Rational Heteronomial Models

In the above section, I presented the simple case for a rational heteronomial model, the number of terms in the numerator and denominator are equal and use the same independent variables. In this section, I present a much more flexible version of rational heteronomial model selection, where the number of terms in the numerator and denominator need not be equal! The only restriction is that function `bestrathet10Flex1()` handles a total of 9 variables and 10 terms in the

numerator and denominator—with a minimum of one term in the numerator OR in the denominator. This means that you can model an equation with at most 9 independent variables:

- A minimum of two independent variables. The model can have one independent variable in the numerator and two independent variables in the denominator, or work with the reverse configuration.
- A maximum of nine independent variables with one duplicate independent variable appearing in both the numerator and denominator parts.
- A total of n terms with at most $n-1$ independent variables (not exceeding 9 variables) appearing in both the numerator and denominator parts.

Instead of hard coding the selection of the variables, for the numerator and denominator, in the MATLAB code itself, I decided to put the data for the selection in the Excel workbook. I use the second row in the sheet **Transf** to select the variables and their placements in numerator and denominator. The values in that row select variables to appear in the numerator part using **positive indices**. Likewise, values in that row select variables to appear in the denominator part using **negative indices**. Rows 3, 4, and 5 contain the initial powers, power increments, and final powers used with each variable. Sheet **Transf** has the following columns:

- Column A has the header Y in row 1. **Row 2 has the total number of terms in the numerator and the denominator. That number has nothing to do with the dependent variable.** Rows 3, 4, and 5 contain the values for the initial power, power increment, and final power used with variable Y.
- Column B has the header X1 in row 1. Row 2 contains a positive index for the variable used as X1. Rows 3, 4, and 5 contain the values for the initial power, power increment, and final power used with variable X1 in the numerator part of the rational heteronomial.
- Column C has the header X2 (or YX1) in row 1 to indicate if it is a term in the numerator or denominator part. Row 2 contains the index for the variable used in this term. If the index is positive, the variable is in the numerator part. If the index is negative, the variable is in the denominator part. Rows 3, 4, and 5 contain the values for the initial power, power increment, and final power used with variable X in the numerator or denominator part of the rational heteronomial.
- Columns D and beyond are like column C in that row 2 contains a positive or negative index to a variable used in that term. Rows 3, 4, and 5 contain

the values for the initial power, power increment, and final power used with variable used in the numerator or denominator part.

- The column before the last one has the header **YX** (followed by an integer) in row 1. Row 2 contains a *negative* index for the variable used as in the last term of the denominator part. Rows 3, 4, and 5 contain the values for the initial power, power increment, and final power used with variable X in the denominator part of the rational heteronomial.
- The last column has the header **Min Adj R-sqr** in row 1. Row 2 contains the minimum value for the adjusted R-square statistic used to qualify a model to be listed in the sheet **List**. This filtering helps reduce the number of results in sheet **List** by eliminating uninteresting models.

The Excel sheet **must have** the following sheets:

- The **Data** sheet with the input data.
- The **Transf** sheet with the variable selection indices, power ranges for the variables, and the minimum value for the adjusted R-square statistic.
- The **Project** and **Project2** sheets.
- The **List** and **List2** sheets.

Let's look at the function in file bestrathet10Flex.m and how it works with the following three different rational heteronomial model category:

1. Case 1: File bestrathet10Flex1.xlsx has data for four independent variables X1 through X4. The numerator has transformations of variable X1. The denominator has transformations of variables X1 through X4.
2. Case 2: File bestrathet10Flex2.xlsx has data for 4 independent variables X1 through X4. The numerator has transformations of variables X1 through X4. The denominator has transformations of variables X1 through X4.
3. Case 3: File bestrathet10Flex3.xlsx has data for 4 independent variables X1 through X4. The numerator has transformations of variables X1 through X4. The denominator has transformations of variables X1 and X2.
4. Case 4: Similar to case 1, with Y values squared. This case will yield two sets of results—the first is for the best power of Y. The second is for the best powers using linear values for Y. You can then compare the results.

The code for function bestrathet10Flex() is a major superset version of bestrathet2.m, so I will discuss the differences after showing you the function's listing. The code uses a double-iteration loop (what I called a *run*) to reduce the number of lines.

```

function [mdl1, mdl2, bestPwrs1, bestPwrs2] =
bestrathet10Flex(XlFile,outFile)

warning('off')

dt = datetime('now','TimeZone','local','Format','y-MMM-d HH_mm_ss');
dtstr = string(dt);
if nargin < 2, outFile = strcat("bestrathet10Flex1_",dtstr,".txt"); end
if nargin < 1, XlFile = strcat(pwd,'\bestrathet10Flex1.xlsx'); end
diary(outFile)
% Flexible rational heteronomials
fprintf("Function bestrathet10Flex\n\n");
fprintf("%s\n", dtstr);
fprintf("Please wait ....\n\n");

fprintf("Excel file used is %s\n", XlFile);
fprintf("Diary file is %s\n", outFile);

% Open Excel as a COM Automation server
Excel = actxserver('Excel.Application');
% Open Excel workbook
Workbook = Excel.Workbooks.Open(XlFile);
% Clear the content of the sheets
Workbook.Worksheets.Item('List').Range('A2:Z100000').ClearContents;
Workbook.Worksheets.Item('List2').Range('A2:Z100000').ClearContents;
Workbook.Worksheets.Item('Project').Range('A2:Z100000').ClearContents;
Workbook.Worksheets.Item('Project2').Range('A2:Z100000').ClearContents;
% Now save/close/quit
Workbook.Save;
Excel.Workbook.Close;
invoke(Excel, 'Quit');

xyData = readmatrix(XlFile,'Sheet','Data');
[maxrows,cols] = size(xyData);
ydata = xyData(:,1);
xData = xyData(:,2:cols);
xyTransf = readmatrix(XlFile,'Sheet','Transf');
nTerms = xyTransf(1,1);
xidx = xyTransf(1,2:nTerms+1);

hdrList = {'Adj R-square','YPwr'};
c = 3;
for i=1:nTerms
    j = abs(xidx(i));
    if xidx(i) > 0
        s = strcat("XPwr", num2str(j));
    else
        s = strcat("YXPwr", num2str(j));
    end
    hdrList(c) = cellstr(s);
    c = c + 1;
end

hdrList(c) = {'Intercept'};
c = c + 1;

```

```

for i=1:nTerms
    j = abs(xidx(i));
    if xidx(i) > 0
        s = strcat("XPwr", num2str(j));
    else
        s = strcat("YXPwr", num2str(j));
    end
    hdrList(c) = cellstr(s);
    c = c + 1;
end

writecell(hdrList,XlFile,'Sheet','List');
writecell(hdrList,XlFile,'Sheet','List2');

idxArr = [];
for i=1:nTerms
    j = abs(xidx(i));
    % search in idxArr
    if length(idxArr) > 0
        bFound = false;
        for k=1:length(idxArr)
            if j == idxArr(k)
                bFound = true;
                break;
            end
        end
        if ~bFound, idxArr = [idxArr j]; end
    else
        idxArr = [idxArr j];
    end
end

idxArr = sort(idxArr);

hdrProject = {'Y'};
c = 1;
for i=1:length(idxArr)
    j = idxArr(i);
    s = strcat("X", num2str(i));
    c = c + 1;
    hdrProject(c) = cellstr(s);
end
c = c + 1;
hdrProject(c) = {'YCalc'};
c = c + 1;
hdrProject(c) = {'%Err'};
writecell(hdrProject,XlFile,'Sheet','Project');
writecell(hdrProject,XlFile,'Sheet','Project2');

flag = zeros(nTerms,1);
for i=1:nTerms
    if xidx(i) < 0, flag(i) = 1; end
    xidx(i) = abs(xidx(i));
end

yPwrs = xyTransf(2:4,1);
x1Pwrs = xyTransf(2:4,2);

```

```

x2Pwrs = xyTransf(2:4,3);
x3Pwrs = xyTransf(2:4,4);
if nTerms >= 4
    x4Pwrs = xyTransf(2:4,5);
else
    x4Pwrs = [1 1 1];
end
if nTerms >= 5
    x5Pwrs = xyTransf(2:4,6);
else
    x5Pwrs = [1 1 1];
end
if nTerms >= 6
    x6Pwrs = xyTransf(2:4,7);
else
    x6Pwrs = [1 1 1];
end
if nTerms >= 7
    x7Pwrs = xyTransf(2:4,8);
else
    x7Pwrs = [1 1 1];
end
if nTerms >= 8
    x8Pwrs = xyTransf(2:4,9);
else
    x8Pwrs = [1 1 1];
end
if nTerms >= 9
    x9Pwrs = xyTransf(2:4,10);
else
    x9Pwrs = [1 1 1];
end
if nTerms >= 10
    x10Pwrs = xyTransf(2:4,11);
else
    x10Pwrs = [1 1 1];
end
minAdjR2 = xyTransf(1,nTerms+2);
yBuffer = zeros(maxrows,1);
x1Buffer = zeros(maxrows,1);
x2Buffer = zeros(maxrows,1);
x3Buffer = zeros(maxrows,1);
x4Buffer = zeros(maxrows,1);
x5Buffer = zeros(maxrows,1);
x6Buffer = zeros(maxrows,1);
x7Buffer = zeros(maxrows,1);
x8Buffer = zeros(maxrows,1);
x9Buffer = zeros(maxrows,1);
x10Buffer = zeros(maxrows,1);

yCols = 0;
for yPwr =yPwrs(1):yPwrs(2):yPwrs(3)
    yCols = yCols + 1;
    yBuffer(:,yCols) = fx(ydata,yPwr);
end

x1Cols = 0;

```

```

j = xidx(1);
for x1Pwr =x1Pwrs(1):x1Pwrs(2):x1Pwrs(3)
    x1Cols = x1Cols + 1;
    x1Buffer(:,x1Cols) = fx(xData(:,j),x1Pwr);
end

x2Cols = 0;
j = xidx(2);
for x2Pwr =x2Pwrs(1):x2Pwrs(2):x2Pwrs(3)
    x2Cols = x2Cols + 1;
    x2Buffer(:,x2Cols) = fx(xData(:,j),x2Pwr);
end

x3Cols = 0;
j = xidx(3);
for x3Pwr =x3Pwrs(1):x3Pwrs(2):x3Pwrs(3)
    x3Cols = x3Cols + 1;
    x3Buffer(:,x3Cols) = fx(xData(:,j),x3Pwr);
end

if nTerms >= 4
    x4Cols = 0;
    j = xidx(4);
    for x4Pwr =x4Pwrs(1):x4Pwrs(2):x4Pwrs(3)
        x4Cols = x4Cols + 1;
        x4Buffer(:,x4Cols) = fx(xData(:,j),x4Pwr);
    end
end

if nTerms >= 5
    x5Cols = 0;
    j = xidx(5);
    for x5Pwr =x5Pwrs(1):x5Pwrs(2):x5Pwrs(3)
        x5Cols = x5Cols + 1;
        x5Buffer(:,x5Cols) = fx(xData(:,j),x5Pwr);
    end
end

if nTerms >= 6
    x6Cols = 0;
    j = xidx(6);
    for x6Pwr =x6Pwrs(1):x6Pwrs(2):x6Pwrs(3)
        x6Cols = x6Cols + 1;
        x6Buffer(:,x6Cols) = fx(xData(:,j),x6Pwr);
    end
end

if nTerms >= 7
    x7Cols = 0;
    j = xidx(7);
    for x7Pwr =x7Pwrs(1):x7Pwrs(2):x7Pwrs(3)
        x7Cols = x7Cols + 1;
        x7Buffer(:,x7Cols) = fx(xData(:,j),x7Pwr);
    end
end

if nTerms >= 8

```

```

x8Cols = 0;
j = xidx(8);
for x8Pwr = x8Pwrs(1):x8Pwrs(2):x8Pwrs(3)
    x8Cols = x8Cols + 1;
    x8Buffer(:,x8Cols) = fx(xData(:,j),x8Pwr);
end
end

if nTerms >= 9
    x9Cols = 0;
    j = xidx(9);
    for x9Pwr = x9Pwrs(1):x9Pwrs(2):x9Pwrs(3)
        x9Cols = x9Cols + 1;
        x9Buffer(:,x9Cols) = fx(xData(:,j),x9Pwr);
    end
end

if nTerms >= 10
    x10Cols = 0;
    j = xidx(10);
    for x10Pwr = x10Pwrs(1):x10Pwrs(2):x10Pwrs(3)
        x10Cols = x10Cols + 1;
        x10Buffer(:,x10Cols) = fx(xData(:,j),x10Pwr);
    end
end

bestxpwr = zeros(10,1);

for iRun=1:2

    bestAdjR2 = 0;
    gRow = 0;
    xyList = zeros(1,1+2*(1+nTerms));
    yCols = 0;
    for yPwr = yPwrs(1):yPwrs(2):yPwrs(3)
        yCols = yCols + 1;
        if iRun==1
            y = yBuffer(:,yCols);
        else
            y = ydata;
        end

        x1Cols = 0 ;
        for x1Pwr = x1Pwrs(1):x1Pwrs(2):x1Pwrs(3)
            x1Cols = x1Cols + 1;
            x1 = x1Buffer(:,x1Cols);

            x2Cols = 0 ;
            for x2Pwr = x2Pwrs(1):x2Pwrs(2):x2Pwrs(3)
                x2Cols = x2Cols + 1;
                x2 = x2Buffer(:,x2Cols);

                if flag(2)==1, x2 = -1 .* y .* x2; end

                x3Cols = 0 ;
                for x3Pwr = x3Pwrs(1):x3Pwrs(2):x3Pwrs(3)
                    x3Cols = x3Cols + 1;

```



```

x3 = x3Buffer(:,x3Cols);

if flag(3)==1, x3 = -1 .* y .* x3; end

x4Cols = 0 ;
for x4Pwr = x4Pwrs(1):x4Pwrs(2):x4Pwrs(3)
    if nTerms >= 4
        x4Cols = x4Cols + 1;
        x4 = x4Buffer(:,x4Cols);

        if flag(4)==1, x4 = -1 .* y .* x4; end
    else
        x4 = [];
    end

x5Cols = 0;
for x5Pwr = x5Pwrs(1):x5Pwrs(2):x5Pwrs(3)
    if nTerms >= 5
        x5Cols = x5Cols + 1;
        x5 = x5Buffer(:,x5Cols);

        if flag(5)==1, x5 = -1 .* y .* x5; end
    else
        x5 = [];
    end

x6Cols = 0;
for x6Pwr = x6Pwrs(1):x6Pwrs(2):x6Pwrs(3)
    if nTerms >= 6
        x6Cols = x6Cols + 1;
        x6 = x6Buffer(:,x6Cols);

        if flag(6)==1, x6 = -1 .* y .* x6; end
    else
        x6 = [];
    end

x7Cols = 0;
for x7Pwr = x7Pwrs(1):x7Pwrs(2):x7Pwrs(3)
    if nTerms >= 7
        x7Cols = x7Cols + 1;
        x7 = x7Buffer(:,x7Cols);

        if flag(7)==1, x7 = -1 .* y .* x7; end
    else
        x7 = [];
    end

x8Cols = 0 ;
for x8Pwr = x8Pwrs(1):x8Pwrs(2):x8Pwrs(3)
    if nTerms >= 8
        x8Cols = x8Cols + 1;
        x8 = x8Buffer(:,x8Cols);

        if flag(8)==1, x8 = -1 .* y .* x8; end
    else
        x8 = [];
    end

```

```

end

x9Cols = 0;
for x9Pwr = x9Pwrs(1):x9Pwrs(2):x9Pwrs(3)
    if nTerms >= 9
        x9Cols = x9Cols + 1;
        x9 = x9Buffer(:,x9Cols);

        if flag(9)==1, x9 = -1 .* y .* x9; end
    else
        x9 = [];
    end

    x10Cols = 0;
    for x10Pwr = x10Pwrs(1):x10Pwrs(2):x10Pwrs(3)
        if nTerms >= 10
            x10Cols = x10Cols + 1;
            x10 = x10Buffer(:,x10Cols);

            if flag(10)==1, x10 = -1 .* y .* x10; end
        else
            x10 = [];
        end

        X = [x1 x2 x3 x4 x5 x6 x7 x8 x9 x10];

        mdl = fitlm(X,y);

        if mdl.Rsquared.Adjusted >= bestAdjR2
            bestAdjR2 = mdl.Rsquared.Adjusted;
            bestYpwr = yPwr;
            bestxpwr(1) = x1Pwr;
            bestxpwr(2) = x2Pwr;
            bestxpwr(3) = x3Pwr;
            if nTerms >=4, bestxpwr(4) = x4Pwr; end
            if nTerms >=5, bestxpwr(5) = x5Pwr; end
            if nTerms >=6, bestxpwr(6) = x6Pwr; end
            if nTerms >=7, bestxpwr(7) = x7Pwr; end
            if nTerms >=8, bestxpwr(8) = x8Pwr; end
            if nTerms >=9, bestxpwr(9) = x9Pwr; end
            if nTerms >=10, bestxpwr(10) = x10Pwr; end
            bestmdl = mdl;
        end

        if mdl.Rsquared.Adjusted >= minAdjR2
            gRow = gRow + 1;
            xyList(gRow,1) = mdl.Rsquared.Adjusted;
            xyList(gRow,2) = yPwr;
            xyList(gRow,3) = x1Pwr;
            xyList(gRow,4) = x2Pwr;
            xyList(gRow,5) = x3Pwr;
            gCol = 6;
            if nTerms >= 4
                xyList(gRow,gCol) = x4Pwr;
                gCol = gCol + 1;
            end
            if nTerms >= 5

```

```

        xyList(gRow,gCol) = x5Pwr;
        gCol = gCol + 1;
    end
    if nTerms >= 6
        xyList(gRow,gCol) = x6Pwr;
        gCol = gCol + 1;
    end
    if nTerms >= 7
        xyList(gRow,gCol) = x7Pwr;
        gCol = gCol + 1;
    end
    if nTerms >= 8
        xyList(gRow,gCol) = x8Pwr;
        gCol = gCol + 1;
    end
    if nTerms >= 9
        xyList(gRow,gCol) = x9Pwr;
        gCol = gCol + 1;
    end
    if nTerms >= 10
        xyList(gRow,gCol) = x10Pwr;
        gCol = gCol + 1;
    end
    xyList(gRow,gCol)= mdl.Coefficients{1,1};
    gCol = gCol + 1;
    xyList(gRow,gCol)= mdl.Coefficients{2,1};
    gCol = gCol + 1;
    xyList(gRow,gCol)= mdl.Coefficients{3,1};
    gCol = gCol + 1;
    xyList(gRow,gCol)= mdl.Coefficients{4,1};
    gCol = gCol + 1;
    gCol2 = 5;
    if nTerms >= 4
        xyList(gRow,gCol)= mdl.Coefficients{gCol2,1};
        gCol = gCol + 1;
        gCol2 = gCol2 + 1;
    end
    if nTerms >= 5
        xyList(gRow,gCol)= mdl.Coefficients{gCol2,1};
        gCol = gCol + 1;
        gCol2 = gCol2 + 1;
    end
    if nTerms >= 6
        xyList(gRow,gCol)= mdl.Coefficients{gCol2,1};
        gCol = gCol + 1;
        gCol2 = gCol2 + 1;
    end
    if nTerms >= 7
        xyList(gRow,gCol)= mdl.Coefficients{gCol2,1};
        gCol = gCol + 1;
        gCol2 = gCol2 + 1;
    end
    if nTerms >= 8
        xyList(gRow,gCol)= mdl.Coefficients{gCol2,1};
        gCol = gCol + 1;
        gCol2 = gCol2 + 1;
    end
end

```

```

        if nTerms >= 9
            xyList(gRow,gCol)= mdl.Coefficients{gCol2,1};
            gCol = gCol + 1;
            gCol2 = gCol2 + 1;
        end
        if nTerms >= 10
            xyList(gRow,gCol)= mdl.Coefficients{gCol2,1};
        end
    end
end
end
end
end
end
end
end
end
end
end
end

[numRes,~] = size(xyList);
mdl = bestmdl;
if iRun==1
    mdl1 = mdl;
    mdl2 = mdl;
else
    mdl2 = mdl;
end

bestPwrs = [bestYpwr];
coeffs = [mdl.Coefficients{1,1}];
for i=1:nTerms
    bestPwrs = [bestPwrs bestxpwr(i)];
    coeffs = [coeffs mdl.Coefficients{i+1,1}];
end
if iRun==1
    bestPwrs1 = bestPwrs;
    bestPwrs2 = bestPwrs;
else
    bestPwrs2 = bestPwrs;
end
xyList = sortrows(xyList,1,"descend");
if iRun==1
    writematrix(xyList, XlFile, 'Sheet', 'List',
'Range',strcat('A2:Z',num2str(numRes+1)));
elseif iRun==2 && bestYpwr == 1
    writematrix(xyList, XlFile, 'Sheet', 'List2',
'Range',strcat('A2:Z',num2str(numRes+1)));
end
% calculate projected values
ycalc = ydata;
for i=1:length(ydata)
    sum1 = coeffs(1);
    sum2 = 1;
    for j=1:nTerms

```

```

% z = mdl.Coefficients{j+1,1}*fx(xData(i,xidx(j)),xyList(1,j+2));
z = coeffs(j+1)*fx(xData(i,xidx(j)),bestPwrs(1+j));
if flag(j) == 0
    sum1 = sum1 + z;
else
    sum2 = sum2 + z;
end
end

ycalc(i) = sum1 / sum2;
if iRun==1
    if bestYpwr > 0
        ycalc(i) = ycalc(i)^(1/bestYpwr);
    elseif bestYpwr < 0
        ycalc(i) = ycalc(i)^abs(bestYpwr);
    else
        ycalc(i) = exp(ycalc(i));
    end
end
end
percErr = 100.*(ycalc-ydata)./ydata;
ProjMat = [ydata xData ycalc percErr];
if iRun==1
    writematrix(ProjMat, XlFile, 'Sheet', 'Project', 'Range', strcat("A2:Z",
num2str(1+length(ydata))));
elseif iRun==2 && bestYpwr == 1
    writematrix(ProjMat, XlFile, 'Sheet', 'Project2', 'Range', strcat("A2:Z",
num2str(1+length(ydata))));
end
% Print the summary of the best regression results
if iRun==1, mdl1=mdl; end
fprintf("Summary:\n");
fprintf("Adj R-sqr = %14.12f", mdl.Rsquared.Adjusted);
mdl

fprintf("Best Y power = %f\n", bestYpwr);
fprintf("Best numerator X%d power = %f\n", xidx(1), bestxpwr(1));

fprintf("Best %s X%d power = %f\n", classify(flag(2)), xidx(2),
bestxpwr(2));
fprintf("Best %s X%d power = %f\n", classify(flag(3)), xidx(3),
bestxpwr(3));
for i=4:10
    if nTerms >= i
        fprintf("Best %s X%d power = %f\n", classify(flag(i)), xidx(i),
bestxpwr(i));
    end
end

if iRun==1 && bestYpwr ~= 1
    fprintf("=====\n");
    fprintf("\n\n----- Phase 2 ----- \n\n");
    fprintf("----- Set yPwr = 1 ----- \n");
    fprintf("=====\n");
    yPwrs = [1 1 1];
end
end % iRun loop

```

```

diary off
end

function y = fx(x,pwr)
    if pwr > 0
        y = x.^pwr;
    elseif pwr < 0
        y = 1./x.^abs(pwr);
    else
        y = log(x);
    end
end

function s=classify(f)
    if f==0
        s = "numerator";
    else
        s = "denominator";
    end
end

```

Notice the following tasks and statements in the above code:

- The function uses **many** if statements to manage the placements of different terms in the numerator or the denominator part. In addition, the function uses if statements to determine if certain independent variables are used and whether to show or hide certain results related to these variables.
- Read the number of terms from cell A2 in sheet **Transf**.
- Initialize the array *flag* that tells the function if a term belongs to the numerator part (when an element of array flag is zero) or belongs to the denominator part (when an element of array flag is 1).
- Initialize the array *xidx* that maps the regression variables to the independent variables as they appear in the numerator or denominator part.
- Assign values to the elements of array *xidx*. Notice that the first three elements are assigned 1, 2, and 3. Likewise, the last three elements are assigned 1, 2, and 3. These assignments mean that we have a rational heteronomial with equal number of terms in the numerator and denominator. You can change the values to say 1, and 2 for the numerator terms, and 1 through 3 (assuming there are four columns for the independent variables in sheet **Data**) for the denominator part.
- The function uses more arrays to store the power values and more matrix buffers.
- The rest of the function is an expanded version of *besttrathet2.m*. Notice that the nested loops use the array *flag* to determine if an array of X values

belongs to the denominator part. If so, it multiplies the array by minus the vector y .

- The regression matrix now has several transformed variables that depends on the number of terms.

Case 1

Let's consider case 1. Figures 5 and 6 show the data in sheets **Data** and **Transf**. Open file bestrathet10Flex1.xlsx and inspect these sheets to make sure the values in the sheets match those in figures 5 and 6. When you are done, close the Excel file.

Y	X1	X2	x3	X4
0.4853954	1	55	71	22
0.694997	2	67	55	21
1.0577411	3	41	67	25
1.5394146	4	65	41	29
2.2191647	5	45	45	27
3.1461697	6	25	35	32
4.0682398	7	34	29	31
5.0112996	8	19	70	33
6.1595398	9	29	48	34
6.9848029	10	71	50	35
8.8348573	11	25	55	38
10.551569	12	29	35	41
12.253242	13	22	45	42
13.475513	14	27	65	45
15.510707	15	21	67	48
20.103322	16	10	20	50
22.133614	17	11	21	55
24.118982	18	12	22	65
25.860208	19	15	27	67
29.241815	20	16	17	70
31.315614	21	21	18	71
35.673864	22	10	19	79
37.214091	23	31	11	80
40.377993	24	32	10	85
42.868659	25	34	12	87
45.08876	26	38	15	90
52.353042	27	11	16	95
52.388459	28	16	31	92
55.967954	29	15	32	97
62.891138	30	10	33	66

Figure 5. The **Data** sheet in file *bestrathet8Flex1.xlsx*.

Y	X1	YX1	YX2	YX3	YX4	Min AdjR2
5	1	-1	-2	-3	-4	0.999
0	0	0	0	0	0	
1	1	1	1	1	1	
2	2	2	2	2	2	

Figure 6. The **Transf** sheet in file *bestrathet10Flex1.xlsx*.

The code for the MATLAB test program test_bestrathet10Flex1.m is:

```
clc
close all
clear all;

dt = datetime('now','TimeZone','local','Format','y-MMM-d HH_mm_ss');
dtstr = string(dt);
outFile = strcat("bestrathet10Flex1_",dtstr,".txt");
XlFile = strcat(pwd,'\bestrathet10Flex1.xlsx');

% mdl1 and mdl2 are saved for future calculatrions done by the user
[mdl1, mdl2] = bestrathet10Flex(XlFile,outFile);
```

The console output of the above test program is:

```
Function bestrathet10Flex

2024-Dec-3 08_39_24
Please wait ....

Excel file used is C:\MATLAB\bestRatHet\bestrathet10Flex1.xlsx
Diary file is bestrathet10Flex1_2024-Dec-3 08_39_24.txt
Summary:
Adj R-sqr = 1.00000000000000
mdl =

Linear regression model:
    y ~ 1 + x1 + x2 + x3 + x4 + x5

Estimated Coefficients:

              Estimate      SE      tStat      pValue
              _____      ___      _____      _____
(Intercept)          5         0         Inf         0
x1                   1         0         Inf         0
x2                   1         0         Inf         0
x3                   1         0         Inf         0
x4                   1         0         Inf         0
```

```

x5          1          0          Inf          0

Number of observations: 30, Error degrees of freedom: 24
R-squared: 1, Adjusted R-Squared: 1
F-statistic vs. constant model: 2.32e+29, p-value = 0
Best Y power = 1.000000
Best numerator X1 power = 2.000000
Best denominator X1 power = 0.000000
Best denominator X2 power = 0.000000
Best denominator X3 power = 0.000000
Best denominator X4 power = 0.000000

```

The above output indicates that all the model's terms are significant. The last few lines, of the above output, indicate the mapping of the independent variables in the numerator and denominator parts. This mapping **is not included** with the results in the sheet **List**. The best model is:

$$Y = (5 + X_1^2) / (1 + \ln(X_1) + \ln(X_2) + \ln(X_3) + \ln(X_4))$$

Figure 7 shows the **List** sheet. The leading results confirm that the above rational heteronomial is indeed the best.

Adj R-square	YPwr	XPwr1	YXPwr1	YXPwr2	YXPwr3	YXPwr4	Intercept	XPwr1	YXPwr1	YXPwr2	YXPwr3	YXPwr4
1	1	2	0	0	0	0	5	1	1	1	1	1
0.999970342	2	2	0	2	2	2	-9.118022298	0.416858571	-0.278470285	1.48206E-05	2.83794E-05	1.55541E-06
0.999968882	2	2	0	2	2	1	-9.264998707	0.420639121	-0.281050695	1.48425E-05	2.87428E-05	0.000244187
0.999967188	2	2	0	1	2	2	-9.585855739	0.42454114	-0.278994396	0.000682134	2.74559E-05	1.32292E-06
0.999965811	2	2	0	1	2	1	-9.705122257	0.427756701	-0.281113127	0.000684068	2.7737E-05	0.00020463
0.999965667	2	2	0	2	1	2	-9.710291379	0.417518031	-0.283635301	1.73806E-05	0.001468628	1.45891E-06
0.999964036	2	2	0	2	1	1	-9.846365019	0.421018977	-0.286030059	1.74295E-05	0.001484212	0.000225958
0.999963539	2	2	0	2	2	0	-9.761505228	0.443430606	-0.295921087	1.56376E-05	3.00264E-05	0.016927624
0.999962197	2	2	0	1	1	2	-10.25527013	0.426776773	-0.284034604	0.000799685	0.001412659	1.18666E-06
0.999961335	2	2	0	0	2	2	-10.15281303	0.439440222	-0.284451275	0.013518349	2.62991E-05	1.012E-06
0.999961118	2	2	0	1	2	0	-10.08273697	0.445710196	-0.292206106	0.000720519	2.85828E-05	0.013041282

Figure 7. A partial view of the **List** sheet in file *besttrathet10Flex1.xlsx*.

Figure 8 also confirms that the above model is best, by first of the extremely small percentage error values.

Y	X1	X2	X3	X4	Ycalc	%Err
0.485395442	1	55	71	22	0.485395442	-6.97613E-13
0.694997041	2	67	55	21	0.694997041	-4.47286E-13
1.057741121	3	41	67	25	1.057741121	-2.51908E-13
1.539414563	4	65	41	29	1.539414563	-1.87512E-13
2.219164746	5	45	45	27	2.219164746	-1.20069E-13
3.146169679	6	25	35	32	3.146169679	-8.46914E-14
4.068239839	7	34	29	31	4.068239839	-8.7328E-14
5.011299637	8	19	70	33	5.011299637	-7.08941E-14
6.159539842	9	29	48	34	6.159539842	-2.88391E-14
6.984802914	10	71	50	35	6.984802914	-1.27159E-14
8.834857306	11	25	55	38	8.834857306	-2.01062E-14
10.55156903	12	29	35	41	10.55156903	-1.6835E-14
12.25324162	13	22	45	42	12.25324162	-2.89941E-14
13.47551322	14	27	65	45	13.47551322	-2.63642E-14
15.5107073	15	21	67	48	15.5107073	0
20.10332169	16	10	20	50	20.10332169	-1.76723E-14
22.13361375	17	11	21	55	22.13361375	1.60512E-14
24.11898245	18	12	22	65	24.11898245	-1.47299E-14
25.86020754	19	15	27	67	25.86020754	0
29.2418148	20	16	17	70	29.2418148	-1.21494E-14
31.31561422	21	21	18	71	31.31561422	-1.13449E-14
35.67386373	22	10	19	79	35.67386373	0
37.21409091	23	31	11	80	37.21409091	0
40.37799339	24	32	10	85	40.37799339	-3.51946E-14
42.86865873	25	34	12	87	42.86865873	-1.65749E-14
45.0887596	26	38	15	90	45.0887596	-1.57588E-14
52.35304187	27	11	16	95	52.35304187	-2.71443E-14
52.38845885	28	16	31	92	52.38845885	0
55.96795384	29	15	32	97	55.96795384	-3.80866E-14
62.8911378	30	10	33	66	62.8911378	-2.2596E-14

*Figure 8. The **Project** sheet in file bestrathet10Flex1.xlsx.*

Case 2

This case deals with a rational heteronomial with four independent variables in both the numerator and denominator. Figures 9 and 10 show the sheets **Data** and **Transf** in file bestrathet10Flex2.xlsx. Open that Excel file and inspect these sheets to make sure the values in the sheets match those in figures 9 and 10. When you are done, close the Excel file.

Y	X1	X2	x3	X4
12.458483	1	55	71	22
11.737728	2	67	55	21
11.106282	3	41	67	25
11.435651	4	65	41	29
10.873907	5	45	45	27
10.205867	6	25	35	32
11.149991	7	34	29	31
13.871858	8	19	70	33
14.109644	9	29	48	34
17.362224	10	71	50	35
17.108771	11	25	55	38
17.987238	12	29	35	41
19.929123	13	22	45	42
22.660316	14	27	65	45
24.682256	15	21	67	48
26.265259	16	10	20	50
28.683357	17	11	21	55
31.37667	18	12	22	65
33.561745	19	15	27	67
36.678622	20	16	17	70
39.039196	21	21	18	71
43.552754	22	10	19	79
45.716187	23	31	11	80
49.204164	24	32	10	85
51.918709	25	34	12	87
54.556737	26	38	15	90
61.054774	27	11	16	95
61.617858	28	16	31	92
65.494414	29	15	32	97
70.465872	30	10	33	66

Figure 9. The **Data** sheet in file *bestrathet10Flex2.xlsx*.

Y	X1	X2	X3	x4	YX1	YX2	YX3	YX4	Min AdjR2
8	1	2	3	4	-1	-2	-3	-4	0.999
0	0	0	0	0	0	0	0	0	
1	1	1	1	1	1	1	1	1	
2	2	2	2	2	2	2	2	2	

Figure 10. The **Transf** sheet in file *bestrathet10Flex2.xlsx*.

The code for the MATLAB test program test_bestrathet10Flex2.m is:

```
clc
close all
clear all;

dt = datetime('now','TimeZone','local','Format','y-MMM-d HH_mm_ss');
dtstr = string(dt);
outFile = strcat("bestrathet10Flex2_",dtstr,".txt");
XlFile = strcat(pwd,'\bestrathet10Flex2.xlsx');

% mdl1 and mdl2 are saved for future calculatrions done by the user
[mdl1, mdl2] = bestrathet10Flex(XlFile,outFile);
```

The console output is:

```
Function bestrathet10Flex

2024-Dec-3 10_22_55
Please wait ....

Excel file used is
I:\DropBox\Dropbox\MATLAB\bestRatHet\bestrathet10Flex2.xlsx
Diary file is bestrathet10Flex2_2024-Dec-3 10_22_55.txt
Summary:
Adj R-sqr = 1.00000000000000
mdl =

Linear regression model:
y ~ 1 + x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8

Estimated Coefficients:
```

	Estimate	SE	tStat	pValue
(Intercept)	5	0	Inf	0
x1	1	0	Inf	0
x2	1	0	Inf	0
x3	1	0	Inf	0
x4	1	0	Inf	0
x5	1	0	Inf	0

x6	1	0	Inf	0
x7	1	0	Inf	0
x8	1	0	Inf	0

Number of observations: 30, Error degrees of freedom: 21
 R-squared: 1, Adjusted R-Squared: 1
 F-statistic vs. constant model: 1.41e+29, p-value = 6.92e-300
 Best Y power = 1.000000
 Best numerator X1 power = 2.000000
 Best numerator X2 power = 1.000000
 Best numerator X3 power = 1.000000
 Best numerator X4 power = 1.000000
 Best denominator X1 power = 0.000000
 Best denominator X2 power = 0.000000
 Best denominator X3 power = 0.000000
 Best denominator X4 power = 0.000000

The console output indicate that the best model is:

$$Y = (5 + X_1^2 + X_2 + X_3 + X_4) / (1 + \ln(X_1) + \ln(X_2) + \ln(X_3) + \ln(X_4))$$

Figures 11a and 11b show the **List** sheet. The leading results confirm that the above rational heteronomial is indeed the best.

Adj R-square	YPwr	XPwr1	XPwr2	XPwr3	XPwr4	YXPwr1	YXPwr2	YXPwr3	YXPwr4	Intercept
1	1	2	1	1	1	0	0	0	0	5
0.999981646	1	0	2	2	2	1	2	0	0	1.36267599
0.999981361	1	0	1	2	2	1	1	0	0	1.46617463
0.999981338	1	2	0	0	2	1	1	2	0	0.88078357
0.999981248	1	2	1	0	2	1	1	2	0	0.35418588
0.999981127	1	2	1	2	2	1	1	0	0	1.11867336
0.999981005	1	2	2	0	2	1	2	2	0	0.46377241
0.999980995	1	1	2	2	2	1	2	0	0	1.24757487
0.999980901	1	2	0	1	2	1	2	2	0	1.33006884

Figure 11a. The **List** sheet (left partial view) in file *besttrathet10Flex2.xlsx*.

XPwr1	XPwr2	XPwr3	XPwr4	YXPwr1	YXPwr2	YXPwr3	YXPwr4
1	1	1	1	1	1	1	1
-0.0834913	-0.000146	-5.147E-05	-0.0016145	0.00846905	-8.353E-06	-0.0055303	-0.3148389
-0.0718233	-0.0084707	-4.828E-05	-0.0016117	0.00832095	-0.0004282	-0.0051339	-0.3129746
0.00313559	-0.1313647	0.24147591	-0.001597	0.0085131	-0.0002855	2.9557E-06	-0.3088225
0.00279625	-0.0036604	0.27735587	-0.0016236	0.00873841	-0.0002509	3.6445E-06	-0.3125984
-0.0066143	-0.0162945	-9.71E-05	-0.0016823	0.00794986	-0.0007412	-0.0096606	-0.3278056
0.00304711	-5.988E-05	0.24391507	-0.0016124	0.00870331	-4.538E-06	3.1166E-06	-0.3116915
-0.0230824	-0.0001598	-5.617E-05	-0.001634	0.00876113	-8.952E-06	-0.006093	-0.3185346
0.0055861	-0.0419877	0.00931197	-0.0015604	0.00851979	-2.535E-06	4.2617E-06	-0.3014259

*Figure 11b. The **List** sheet (right partial view) in file bestrathet10Flex2.xlsx.*

Figure 12 also confirms that the above model is best, by first of the extremely small percentage error values.

Y	X1	X2	X3	X4	YCalc	%Err
12.458483	1	55	71	22	12.458483	0
11.7377278	2	67	55	21	11.7377278	0
11.1062818	3	41	67	25	11.1062818	0
11.435651	4	65	41	29	11.435651	1.55335E-14
10.8739073	5	45	45	27	10.8739073	-3.26719E-14
10.2058675	6	25	35	32	10.2058675	1.74053E-14
11.1499907	7	34	29	31	11.1499907	-1.59315E-14
13.8718584	8	19	70	33	13.8718584	1.28055E-14
14.1096436	9	29	48	34	14.1096436	0
17.3622244	10	71	50	35	17.3622244	0
17.1087713	11	25	55	38	17.1087713	2.07655E-14
17.9872385	12	29	35	41	17.9872385	-1.97513E-14
19.9291229	13	22	45	42	19.9291229	-1.78267E-14
22.6603158	14	27	65	45	22.6603158	0
24.682256	15	21	67	48	24.682256	-1.43938E-14
26.2652594	16	10	20	50	26.2652594	-2.70526E-14
28.6833566	17	11	21	55	28.6833566	0
31.3766702	18	12	22	65	31.3766702	2.26456E-14
33.5617448	19	15	27	67	33.5617448	0
36.678622	20	16	17	70	36.678622	-3.87442E-14
39.0391962	21	21	18	71	39.0391962	-1.82008E-14
43.5527539	22	10	19	79	43.5527539	-3.26291E-14
45.7161866	23	31	11	80	45.7161866	1.55425E-14
49.2041641	24	32	10	85	49.2041641	0
51.9187089	25	34	12	87	51.9187089	0
54.556737	26	38	15	90	54.556737	-1.30239E-14
61.0547736	27	11	16	95	61.0547736	-1.16378E-14
61.6178578	28	16	31	92	61.6178578	-1.15314E-14
65.4944141	29	15	32	97	65.4944141	0
70.4658715	30	10	33	66	70.4658715	0

*Figure 12. The **Project** sheet in file bestrathet10Flex2.xlsx.*

Case 3

This case deals with a rational heteronomial with four independent variables in both the numerator and denominator. Figures 13 and 14 show the sheets **Data** and **Transf** in file bestrathet10Flex3.xlsx. Open that Excel and inspect these sheets to make sure the values in the sheets match those in figures 13 and 14. When you are done, close the Excel file.

Y	X1	X2	x3	X4
26.800693	1	55	71	2.2
22.567585	2	67	55	2.1
21.420518	3	41	67	2.5
19.799772	4	65	41	2.9
19.123766	5	45	45	2.7
17.335938	6	25	35	3.2
18.556084	7	34	29	3.1
26.77676	8	19	70	3.3
25.348386	9	29	48	3.4
30.336016	10	71	50	3.5
31.707308	11	25	55	3.8
31.683244	12	29	35	4.1
36.838988	13	22	45	4.2
42.898996	14	27	65	4.5
47.804003	15	21	67	4.8
48.722886	16	10	20	5
53.200806	17	11	21	5.5
57.958253	18	12	22	6.5
62.337569	19	15	27	6.7
65.747473	20	16	17	7
69.416968	21	21	18	7.1
82.253775	22	10	19	7.9
77.151917	23	31	11	8
82.616087	24	32	10	8.5
88.402725	25	34	12	8.7
94.102059	26	38	15	9
115.10768	27	11	16	9.5
118.96194	28	16	31	9.2
127.58387	29	15	32	9.7
142.39722	30	10	33	6.6

Figure 13. The **Data** sheet in file *bestrathet10Flex3.xlsx*.

Y	X1	X2	X3	X4	YX1	YX2	Min AdjR2
6	1	2	3	4	-1	-2	0.999
0	0	0	0	0	0	0	
1	1	1	1	1	1	1	
2	2	2	2	2	2	2	

Figure 14. The **Transf** sheet in file *bestrathet10Flex3.xlsx*.

The code for the MATLAB test program test_bestrathet10Flex3m is:

```
clc
close all
clear all;

dt = datetime('now','TimeZone','local','Format','y-MMM-d HH_mm_ss');
dtstr = string(dt);
outFile = strcat("bestrathet10Flex3_",dtstr,".txt");
XlFile = strcat(pwd,'\bestrathet10Flex3.xlsx');

% mdl1 and mdl2 are saved for future calculatrions done by the user
[mdl1, mdl2] = bestrathet10Flex(XlFile,outFile);
```

The console output is:

```
Function bestrathet10Flex

2024-Dec-3 10_38_53
Please wait ....

Excel file used is
I:\DropBox\Dropbox\MATLAB\bestRatHet\bestrathet10Flex3.xlsx
Diary file is bestrathet10Flex3_2024-Dec-3 10_38_53.txt
Summary:
Adj R-sqr = 1.00000000000000
mdl =

Linear regression model:
y ~ 1 + x1 + x2 + x3 + x4 + x5 + x6

Estimated Coefficients:

```

	Estimate	SE	tStat	pValue
(Intercept)	5	1.8986e-06	2.6335e+06	1.5994e-133
x1	1	7.4947e-08	1.3343e+07	9.8981e-150
x2	1	7.7081e-08	1.2973e+07	1.8878e-149
x3	1	7.2e-08	1.3889e+07	3.9347e-150
x4	1	3.6861e-07	2.7129e+06	8.0753e-134
x5	1	9.4515e-08	1.058e+07	2.0544e-147

```

x6          1          7.7525e-08          1.2899e+07          2.1546e-149

Number of observations: 30, Error degrees of freedom: 23
Root Mean Squared Error: 1.12e-06
R-squared: 1, Adjusted R-Squared: 1
F-statistic vs. constant model: 3.79e+29, p-value = 0
Best Y power = 1.000000
Best numerator X1 power = 2.000000
Best numerator X2 power = 1.000000
Best numerator X3 power = 1.000000
Best numerator X4 power = 1.000000
Best denominator X1 power = 0.000000
Best denominator X2 power = 0.000000

```

The console output indicate that the best model is:

$$Y = (5 + X_1^2 + X_2 + X_3 + X_4) / (1 + \ln(X_1) + \ln(X_2))$$

Figure 15 shows the **List** sheet. The leading results confirm that the above rational heteronomial is indeed the best.

Adj R-square	YPwr	XPwr1	XPwr2	XPwr3	XPwr4	YXPwr1	YXPwr2	Intercept	XPwr1	XPwr2	XPwr3	XPwr4	YXPwr1	YXPwr2
1	1	2	1	1	1	0	0	5	1	1	1	1	1	1
0.999945826	1	2	1	1	2	0	0	8.590019948	0.979826287	0.959551825	0.969189135	0.061677015	0.976700679	0.973538355
0.999904858	1	2	1	1	0	0	0	2.736322378	1.033432708	1.037787086	1.027331582	4.289648798	1.042610837	1.027761836
0.999800087	2	0	2	2	0	0	0	-459.7198306	-326.1461174	0.021746717	0.042530144	1092.863439	-0.300879305	0.023964431
0.999791752	2	0	1	2	0	0	0	-466.5857975	-331.4936189	1.461832499	0.041543507	1096.120417	-0.30159378	0.024831225
0.999784263	2	0	0	2	0	0	0	-257.3635725	-336.0949004	-13.09473756	0.036926918	1011.684437	-0.296587348	0.017100696
0.99978247	2	0	2	2	0	0	1	-405.7275228	-301.0691013	0.022731964	0.038323815	1003.066536	-0.287894923	0.001025895
0.999777932	2	0	2	1	0	0	0	-505.6746229	-343.400974	0.018010633	3.353551147	1118.322477	-0.299411088	0.022178829
0.999771678	2	0	1	2	0	0	1	-391.5667806	-306.6958668	1.31771567	0.036701283	995.196416	-0.288047961	0.001019668

Figure 15. The **List** sheet in file *besttrathet10Flex3.xlsx*.

Figure 16 also confirms that the above model is best, by first of the extremely small percentage error values.

Y	X1	X2	X3	X4	YCalc	%Err
26.8006931	1	55	71	2	26.8006931	-2.38609E-13
22.5675848	2	67	55	2	22.5675848	-7.87128E-14
21.4205181	3	41	67	3	21.4205181	-4.97567E-14
19.7997719	4	65	41	3	19.7997719	1.79432E-14
19.1237656	5	45	45	3	19.1237656	3.7155E-14
17.3359379	6	25	35	3	17.3359379	6.148E-14
18.5560843	7	34	29	3	18.5560843	9.57291E-14
26.7767595	8	19	70	3	26.7767595	-1.32679E-14
25.3483864	9	29	48	3	25.3483864	1.40155E-14
30.3360161	10	71	50	4	30.3360161	2.34224E-14
31.7073081	11	25	55	4	31.7073081	-1.12047E-14
31.6832435	12	29	35	4	31.6832435	2.24265E-14
36.8389876	13	22	45	4	36.8389876	-1.92878E-14
42.8989962	14	27	65	5	42.8989962	-1.65632E-14
47.8040026	15	21	67	5	47.8040026	-4.4591E-14
48.7228858	16	10	20	5	48.7228858	-4.375E-14
53.2008059	17	11	21	6	53.2008059	-2.67117E-14
57.9582532	18	12	22	7	57.9582532	0
62.3375685	19	15	27	7	62.3375685	-1.13983E-14
65.7474727	20	16	17	7	65.7474727	-2.16143E-14
69.4169678	21	21	18	7	69.4169678	-2.04717E-14
82.2537747	22	10	19	8	82.2537747	-3.45537E-14
77.1519167	23	31	11	8	77.1519167	-3.68386E-14
82.6160873	24	32	10	9	82.6160873	-1.72011E-14
88.4027251	25	34	12	9	88.4027251	-1.60751E-14
94.1020591	26	38	15	9	94.1020591	-4.53046E-14
115.107683	27	11	16	10	115.107683	-1.23457E-14
118.961942	28	16	31	9	118.961942	-1.19457E-14
127.583866	29	15	32	10	127.583866	0
142.397222	30	10	33	7	142.397222	-3.99189E-14

*Figure 16. The **Project** sheet in file bestrathet10Flex3.xlsx.*

Case 4

This case deals with a rational heteronomial with four independent variables in both the numerator and denominator. Figures 17 and 18 show the sheets **Data** and **Transf** in file bestrathet10Flex4.xlsx. Open that Excel and inspect these sheets to make sure the values in the sheets match those in figures 17 and 18. When you are done, close the Excel file.

Y	X1	X2
1.0946427	1	55
1.2353066	2	67
1.5520093	3	41
1.7891023	4	65
2.1623453	5	45
2.6117508	6	25
2.8884745	7	34
3.3844365	8	19
3.6194929	9	29
3.7254829	10	71
4.3637737	11	25
4.6631356	12	29
5.112911	13	22
5.383666	14	27
5.8361882	15	21
6.5545201	16	10
6.8689603	17	11
7.183703	18	12
7.4173446	19	15
7.7354759	20	16
7.9318331	21	21
8.7454218	22	10
8.3991931	23	31
8.718338	24	32
9.0188868	25	34
9.2870699	26	38
10.47162	27	11
10.538111	28	16
10.934812	29	15
11.618882	30	10

Figure 17. The **Data** sheet in file *besttrathet10Flex4.xlsx*.

Y	X1	YX1	YX2	Min AdjR2
3	1	-1	-2	0.99
0	0	0	0	
1	1	1	1	
3	3	3	3	

Figure 18. The **Transf** sheet in file *besttrathet10Flex4.xlsx*.

The code for the MATLAB test program test_besttrathet10Flex4.m is:

```
clc
close all
clear all;

dt = datetime('now','TimeZone','local','Format','y-MMM-d HH_mm_ss');
dtstr = string(dt);
outFile = strcat("besttrathet10Flex4_",dtstr,".txt");
XlFile = strcat(pwd,'\besttrathet10Flex4.xlsx');

% mdl1 and mdl2 are saved for future calculatrions done by the user
[mdl1, mdl2] = besttrathet10Flex(XlFile,outFile);
```

The console output is:

```
Function besttrathet10Flex

2024-Dec-4 02_42_39
Please wait ....

Excel file used is C:\MATLAB\bestRatHet\besttrathet10Flex4.xlsx
Diary file is besttrathet10Flex4_2024-Dec-4 02_42_39.txt
Summary:
Adj R-sqr = 1.00000000000000
mdl =

Linear regression model:
    y ~ 1 + x1 + x2 + x3

Estimated Coefficients:

```

	Estimate	SE	tStat	pValue
(Intercept)	5	0	Inf	0
x1	1	0	Inf	0
x2	1	0	Inf	0
x3	1	0	Inf	0

```

Number of observations: 30, Error degrees of freedom: 26
R-squared: 1, Adjusted R-Squared: 1
F-statistic vs. constant model: 9.76e+29, p-value = 0
Best Y power = 2.000000
Best numerator X1 power = 2.000000
Best denominator X1 power = 0.000000
Best denominator X2 power = 0.000000
=====

----- Phase 2 -----

----- Set yPwr = 1 -----
=====

Summary:
Adj R-sqr = 0.999828114833
mdl =

Linear regression model:
y ~ 1 + x1 + x2 + x3

Estimated Coefficients:

```

	Estimate	SE	tStat	pValue
(Intercept)	0.99197	0.02529	39.223	1.1525e-24
x1	-0.0040374	0.0001509	-26.756	1.8955e-20
x2	-0.35543	0.0038328	-92.735	2.6312e-34
x3	-0.0072624	0.0024906	-2.9159	0.0072107

```

Number of observations: 30, Error degrees of freedom: 26
Root Mean Squared Error: 0.0416
R-squared: 1, Adjusted R-Squared: 1
F-statistic vs. constant model: 5.62e+04, p-value = 1.16e-49
Best Y power = 1.000000
Best numerator X1 power = 2.000000
Best denominator X1 power = 0.000000
Best denominator X2 power = 0.000000

```

The console output indicate that the best model is:

$$Y^2 = (5 + X_1^2) / (1 + \ln(X_1) + \ln(X_2))$$

Setting the power of variable Y to 1 yields a model with an adjusted R-square of 0.999828114833 and the model:

$$Y = (0.99197 - 0.0040374 * X_1^2) / (1 - 0.35543 * \ln(X_1) - 0.0072624 * \ln(X_2))$$

Figure 19 shows the **List** sheet. The leading results confirm that the above rational heteronomial is indeed the best.

Adj R-square	YPwr	XPwr1	YXPwr1	YXPwr2	Intercept	XPwr1	YXPwr1	YXPwr2
1	2	2	0	0	5	1	1	1
0.999950701	3	3	0	0	-5.208198039	-0.155681655	-0.663242906	-0.619695623
0.999925549	2	2	1	0	2.960661268	0.310714196	0.012264705	0.316381573
0.999924809	3	2	0	0	-4.250995358	0.47635172	-0.263433076	0.070413331
0.999910732	3	2	0	1	-3.07147014	0.40817095	-0.23436114	0.002945834
0.999893581	2	3	0	0	1.524662124	-0.00191641	-0.376076447	-0.039679581
0.999836693	3	2	0	2	-1.330597668	0.371083783	-0.232532562	5.44488E-05
0.999828115	1	2	0	0	0.991965047	-0.004037406	-0.355434712	-0.007262445
0.999821232	2	3	0	1	1.647144537	-0.002079966	-0.407539342	-0.002017501

Figure 19. The List sheet in file bestrathet10Flex4.xlsx.

Figure 20 also confirms that the above model is best, noticing the extremely small percentage error values.

Y	X1	X2	YCalc	%Err
1.094642687	1	55	1.094642687	-8.72241E-13
1.23530662	2	67	1.23530662	-5.9317E-13
1.552009328	3	41	1.552009328	-3.43366E-13
1.789102266	4	65	1.789102266	-2.10986E-13
2.162345325	5	45	2.162345325	-1.23224E-13
2.61175082	6	25	2.61175082	-8.50175E-14
2.888474542	7	34	2.888474542	-4.61236E-14
3.384436504	8	19	3.384436504	-2.6243E-14
3.61949286	9	29	3.61949286	-2.45388E-14
3.725482929	10	71	3.725482929	-2.38406E-14
4.363773666	11	25	4.363773666	0
4.663135555	12	29	4.663135555	0
5.112911019	13	22	5.112911019	0
5.383666004	14	27	5.383666004	0
5.836188166	15	21	5.836188166	-1.52185E-14
6.554520101	16	10	6.554520101	-1.35506E-14
6.868960269	17	11	6.868960269	-1.29303E-14
7.183703044	18	12	7.183703044	0
7.417344566	19	15	7.417344566	-1.19743E-14
7.735475908	20	16	7.735475908	-1.14819E-14
7.931833087	21	21	7.931833087	-1.11976E-14
8.745421843	22	10	8.745421843	-2.03118E-14
8.399193079	23	31	8.399193079	0
8.718337963	24	32	8.718337963	0
9.018886827	25	34	9.018886827	0
9.287069859	26	38	9.287069859	-1.91272E-14
10.47161979	27	11	10.47161979	-1.69635E-14
10.53811127	28	16	10.53811127	-1.68565E-14
10.93481247	29	15	10.93481247	-1.6245E-14
11.61888178	30	10	11.61888178	-1.52885E-14

Figure 20. The **Project** sheet in file *bestrathet10Flex4.xlsx*.

Figure 21 shows the contents of sheet **List2**. You can see the effect of forcing the variable to be linear.

Adj R-square	YPwr	XPwr1	YXPwr1	YXPwr2	Intercept	XPwr1	YXPwr1	YXPwr2
0.999828115	1	2	0	0	0.991965047	-0.004037406	-0.355434712	-0.007262445
0.999803352	1	2	0	1	1.011106325	-0.004091589	-0.360824946	-0.000253297
0.999779588	1	2	0	2	1.026387754	-0.00405503	-0.360774329	-2.11289E-06
0.999771915	1	2	0	3	1.03972221	-0.004005644	-0.359609916	-1.28696E-09
0.999618225	1	3	0	2	1.122277589	-7.27322E-05	-0.313584871	4.89521E-06
0.99961332	1	3	0	1	1.121253157	-7.33725E-05	-0.314892586	0.000274067
0.999608161	1	3	0	3	1.117612819	-7.28155E-05	-0.313371289	6.92623E-08
0.999595409	1	3	0	0	1.121431001	-7.45188E-05	-0.317816665	0.004472127
0.999567989	1	1	0	0	0.825774187	0.406127554	-0.022679457	0.082702608

Figure 21. The **List2** sheet in file *bestrathet10Flex4.xlsx*.

Y	X1	X2	YCalc	%Err
1.094642687	1	55	1.01754118	-7.043531905
1.23530662	2	67	1.349498079	9.243976907
1.552009328	3	41	1.640435788	5.697546886
1.789102266	4	65	1.944382337	8.679217199
2.162345325	5	45	2.225881848	2.938315291
2.61175082	6	25	2.491742893	-4.594922562
2.888474542	7	34	2.808641668	-2.763842055
3.384436504	8	19	3.062793363	-9.503595083
3.61949286	9	29	3.417366564	-5.584381663
3.725482929	10	71	3.905254404	4.825454267
4.363773666	11	25	4.049284494	-7.206816773
4.663135555	12	29	4.447191616	-4.630874149
5.112911019	13	22	4.700153292	-8.072851747
5.383666004	14	27	5.272671406	-2.06169175
5.836188166	15	21	5.441396763	-6.764542056
6.554520101	16	10	18.9426809	189.0017973
6.868960269	17	11	7.154953293	4.163556236
7.183703044	18	12	6.9660635	-3.029628914
7.417344566	19	15	7.029872837	-5.223860451
7.735475908	20	16	7.336024599	-5.163887958
7.931833087	21	21	7.564598207	-4.629886637
8.745421843	22	10	8.338428192	-4.653790961
8.399193079	23	31	8.205172882	-2.309986152
8.718337963	24	32	8.617068944	-1.161563354
9.018886827	25	34	9.02369808	0.053346418
9.287069859	26	38	9.418500644	1.415201854
10.47161979	27	11	10.33149746	-1.338115123
10.53811127	28	16	10.62680346	0.841632691
10.93481247	29	15	11.10051457	1.515362997
11.61888178	30	10	11.7083143	0.769717125

*Figure 22. The **Project2** sheet in file bestrathet10Flex4.xlsx.*

The rise in percentage of errors in Figure 22 explains why the adjusted R-square dropped from 1 to 0.999828114833.

Closing Remarks for Part I

The functions `bestrathert2()`, and its augmented versions `bestrathert4()` and `bestrathert6()`, help introduce you to the concept of rational heteronomial curve fitting. The real winner is function `bestrathet10Flex()` due to its flexibility and ability to handle 9 independent variables. That function is the product of several code iterations. These functions depend on partnering Excel files to provide data for the regression, information for the number of terms, the orientation of the terms (numerator or denominator), and the power values.

My hope is that rational heteronomials can help you discover new empirical relations between variables.

About the ZIP File for this study

You will find the code for the various MATLAB files and Excel files in the ZIP file for this document. The ZIP file contains the PDF version of the document, and the files used in this three-part study.