Best Linearized Regression Program With the HP-35s

By
Namir C Shammas

Overview

This section introduces you to the best fit program and looks at program rationale and features.

Quick Introduction

The best linearized regression program implements automatic best linear regression and offers the following features:

1. Storing data points once and reusing them for different regression scenarios. The program stores the (x, y) data points as 2-D vectors, a new feature of the HP-35s. This storage scheme reduces the number of registers used by half.
2. The ability to delete, edit, view, and swap data points. Thus the regression program incorporates basic operations to manage the data points.
3. The ability to select separate sets of transformation for the x and for the y observations.

HP's Legacy with Automatic Best Fit Programs

The HP-35s, like many HP calculators, has built-in linear regression. This built-in feature makes it very easy to calculate the slope and intercept for the best straight line that passes though a set of (x, y) data points. A linear regression is the simplest case of regression analysis since it assumes a linear relationship between the independent variable x and the dependent variable y. Not all variables in science, finance, and other disciplines relate to each other in a linear manner. However, by applying mathematical transformations on either or both x and y variables it is possible to linearize the fit. HP has implemented on other calculator models automatic best fit by using the logarithm function to transform data, Along with linear values, HP has offers fitting the models shown in the next table.

Model Name	Equation
Linear	y = a + b x
Logarithmic	y = a + b ln(x)
Exponential	ln(y) = a + b x
Power	ln(y) = a + b ln(x)

By applying the combination of two transformation (linear and logarithmic), you get a set of four models. The program offers up to ten transformations for each of the variables x and y. These transformations give you up to a hundred different curves to test with your data.

Design Features

The design of an automatic best fit program involves several choices:

1. The set of mathematical transformations

2. Whether to hard code the set of mathematical transformations into the program, or offer the user a choice. This aspect is crucial because the presence of zeros and negative data eliminates the use of mathematical transformations like square roots, logarithms, and reciprocals.
3. Whether or not to allow the dynamic shifting and scaling of values during the regression phase.
4. The number of best curve fits to display.

Transformations Table

Looking at design choices number 1 and 2, in the last subsection, I present the following table that contains the set of mathematical transformations:

Transformation Index	Transformation Function f(x)
0	ln(x)
1	x
2	x^2
3	x^3
4	1/x
5	√x
6	1/x^2
7	1/√x
8	Custom 1: current default is $\sqrt[3]{x}$
9	Custom 2: current default is $1/\sqrt[3]{x}$

The above table shows a set of transformation and their associated indices. The table shows an expanded set of mathematical transformations, which the program can apply to determine the best curve fit. Using the indices, the program user can select which set of transformations to apply to each variable. The program implements the set of transformations as an integer made up of the single digits in the above table. I will call this number the *transformation group value*. To mimic HP's classical automatic best fit we use the transformation group value of 10 with each variable. Here the number 10 is made up of the digit 1 and 0 which represent the indices for linear and logarithm in the above table, respectively. The program initializes the transformation group for x and y using the value 4210 for each variable. This value is made up of the digits 4, 2, 1, and 0 which represent the reciprocal, square, linear, and logarithmic transformations, respectively. You can enter these digits in any order, but you must observe the following little rule. The digit 0 must appear to the right of any other digit. So the transformation group values of 4210, 2104, 4120, 4021, 4012 all perform the same set of transformations, although in different order. The order of applying a transformation is from least significant digit to most significant digit. Notice that the number 0421 will not perform the logarithmic transformation, because the digit 0 in this case is not significant. Why not? The program, working with copies of the transformation group values, performs the following tasks to manage the transformations:

1. Extracts the least significant digit and uses that digit to determine the next mathematical transformation
2. Reduces the transformation group values by 10 and keeps the integer part
3. Repeats steps 1 and 2 until the transformation group value is zero. This is why the digit 0 must be significant. Putting it at the leftmost place will cause the program to deduce that there are no more transformation indices to process.

The program design allows you to use separate transformation group values for the x and y variables. For example you can use the transformation group value of 3210 for x and just the value of 10 for y. This flexibility allows you to avoid certain transformations for either variable but not the other.

The program handles transformations number 8 and 9 in a bit different manner. The program design uses the labels X and Y as a special placeholder where you can code in your particular transformations and apply them to the variables x and/or y. The label T is the routine that performs the mathematical transformations and calls the code in labels X and Y when the transformation index is 8 or 9, respectively. By setting two separate labels, the novice and average users should find it easier to edit code in isolated and dedicated labels. A good programmer can certainly edit the code in label T itself. However, she or he should observe how the code flows and how it goes through the sequence of comparing the transformation index with different integers.

Shifting and Scaling

Regarding the subject of dynamic shifting and scaling of variables, I decided that such operations may work very well with specific transformations while at the same time might backfire with other transformations. The suitability of scaling and shifting greatly depends on the nature of the data entering the regression. The labels X and Y should be suitable venues that allow you to implement custom scaling and shifting and incorporate them with specific mathematical transformation. For example, you can code label X to get $\ln(x_0 - x)$ where x_0 is a fixed value, either hard coded or recall from a memory register. Another example is to code label Y to apply the transformation $\sqrt{(x + x_0)}$ which can deal with all negative values of x, if x_0 is large enough. This approach allows you to implement scaling and shifting for specific transformation and avoid applying them across the board with all of the transformation. You can certainly apply scaling and shifting to you raw data points separately *before* you key them in the program, if you deem such mathematical operations as appropriate for all the transformations. One example is shifting your data points to eliminate negative values and zeros. Such an operation allows you to apply any or all of the mathematical transformations supported by the automatic best fit program.

And the Winner is?

What about the results of the best curve fit? Simply presenting the very best curve fit for a large combination of curves, which can reach a hundred, may seem too limited. On the other extreme, displaying the results for all the curves, sorted using the values of the coefficient of determination, r^2, is perhaps too demanding for the resources of the HP-35s. The current program design displays the top three best fits.

Instructions

The following table shows the steps involved in using the various parts of the linear regression program. The commands show the label truncated to the first letter when step 001 is implied. Thus XEQ A, for example, is the same as typing XEQ A001 or XEQ A [Enter]. This convention is used throughout this document.

When it comes to the output, the program pauses to display an output tag for each output, before showing the numerical values of that output. The output usually displays results in pairs to take advantage of the double display lines of the calculator. Using output tags labels the output and makes the program more user-friendly. In addition, when a task mentions viewing two output values, call them A and B, then

A appears in the Y stack register and B appears in the X stack register. Likewise, when a task mentions the output of three values, call them A, B, and C, then A is placed in the Z stack register, B appears in the Y stack register, and C appears in the X stack register. Since the calculator displays the Y and X stack registers, the value of A is not immediately visible. You will need to use the R↓ key to view that value.

Task #	Task/Subtask	Input	Command	Output X	Output Y
1	Initialize the transformation group values. **YOU MUST PERFORM THIS TASK AT LEAST ONCE.**		XEQ K		
2	Initialize the program.		XEQ I	0	
3	Add a new data point.		XEQ A	X?	
	Enter a value for x.	x	R/S	Y?	
	Enter a value for y.	y	R/S	Number of data points	
4	Delete a data point.		XEQ D	I?	
	Enter the index of the data point to delete. If you enter an invalid index the program displays the message INVALID DATA.	Index of data point to delete	R/S	Number of data points	
5	Edit a data point.		XEQ E	X?	
	Enter a new value for x.	x	R/S	Y?	
	Enter a new value for y.	y	R/S	I?	
	Enter the index of the data point to edit. If you enter an invalid index the program displays the message INVALID DATA.	Index	R/S		
6	View a data point.		XEQ V	I?	
	Enter the index of the data point to view. If you enter an invalid index the program displays the message INVALID DATA.	Index	R/S	Data point	Index
	To view the next data point. You can repeat this subtask to view a succession of data points.		R/S	Data point	Index
	When you have		R/S	Number of	

Task #	Task/Subtask	Input	Command	Output X	Output Y
	finished viewing the last data point and press R/S the calculator displays the message INVALID DATA.			data points + 1	
7	View all data points (Note the program displays the results using the PSE statement.) If you spot an error in the data you entered earlier, use the command XEQ E to overwrite the erroneous data point with the correct one.		XEQ W	Data point	Index
8	Swap data points.		XEQ S	I?	
	Enter the index of the first data point. If you enter an invalid index the program displays the message INVALID DATA.	Index of the first data point	R/S	J?	
	Enter the index of the second data point. If you enter an invalid index the program displays the message INVALID DATA.	Index of the second data point	R/S		
9	To view or assign new transformation group values.		XEQ J	X? <Tx group value>	
	Optionally enter a new value for transforming x	Optional new value	R/S	Y? <Ty group value>	
	Optionally enter a new value for transforming y	Optional new value	R/S		
10	Calculate the best regression fits.[1]		XEQ C		
	View the transformation indices for the best regression			[r^2, Tx, Ty]	

Task #	Task/Subtask	Input	Command	Output X	Output Y
	fit. Tx is the transformation index for x. Ty is the transformation index for y.[2]				
	View best regression fit.		R/S	[slope, intercept]	[r^2, Tx, Ty]
	View the transformation indices for the 2nd best regression fit.[2]			[r^2, Tx, Ty]	
	View the 2nd best regression fit.		R/S	[slope, intercept]	[r^2, Tx, Ty]
	View the transformation indices for the 2nd best regression fit.[2]			[r^2, Tx, Ty]	
	View the 3rd best regression fit.		R/S	[slope, intercept]	[r^2, Tx, Ty]
11	To re-examine the best fit results.		XEQ F		
	View the transformation indices for the best regression fit. [2]			[r^2, Tx, Ty]	
	View best regression fit.		R/S	[slope, intercept]	[r^2, Tx, Ty]
	View the transformation indices for the 2nd best regression fit. [2]			[r^2, Tx, Ty]	
	View the 2nd best regression fit.		R/S	[slope, intercept]	[r^2, Tx, Ty]
	View the transformation indices for the 3rd best regression fit. [2]			[r^2, Tx, Ty]	
	View the 3rd best regression fit.		R/S	[slope, intercept]	[r^2, Tx, Ty]

[1] While executing label C, the program switches to the FIX 0 display mode and briefly pauses to show the transformation indices. The program performs this kind of pause twice. The first pause occurs during the model iterations, so you can follow the progress of the program. The second pause occurs when the program displays the final results.

[2] The display is in the FIX 0 mode which will round the value of r^2 to the nearest integer (1 or 0). The second and third digits in the 3-D vector are the relevant part of the output since they represent the transformation indices. Using FIX 0 reduces the distraction from all the superfluous zeros and makes the integer values of the transformation indices stand out and easy to read.

Comments on Runtime Errors

The program validates the indices that you enter to manage data points to prevent the corruption of the data points. Aside from this feature, you are responsible to selecting transformations that that will avoid runtime errors, if you have negative values and zeros. The best fit program is a convenient tool that cycles through different regression curves and find the best three. The application is not a magical data mining tool. The best way to use the program is to know your data!

Program Resources

Labels	Purpose	Memory Registers	Purpose
A	Add a new data point	A	
D	Delete an existing data point	B	
E	Edit a data point	C	Data counter
V	View a data point	D	Ty Group
W	View all data points	E	Tx Group Initial
I	Initialize data points	F	Tx Group
J	View/Set transformation groups	G	Tx
K	Initialize/Reset transformation groups	H	Ty
S	Swap two data points	I	Index
C	Calculate best regression fit	J	Index
F	View best three curve fits	K	[r^2,Tx,Ty] for best fit
Z	Set/reset display format	L	[m, b] for best
X	Custom Transform #1	M	[r^2,Tx,Ty] for 2nd best fit
Y	Custom Transform #2	N	[m, b] for 2nd best fit
		O	[r^2,Tx,Ty] for 3rd best fit
Labels for internal use		P	[m, b] for 3rd best fit
H	Validate indices and set flag 1 if index if invalid	Q	
G	Prompt for X and Y	R	r^2
T	Transform data	S	
P	Parse the value of the transformation index group	T	
B	Find the best three curve fits	U	
		V	
Flag 1	Index invalid when set	W	
		X	Used
		Y	Used
		Z	
		0	[Tx Group, Ty Group]

Labels	Purpose	Memory Registers	Purpose
		1	Data point 1
		2	Data point 2
		…	
		#Count	Data point #Count

Customizing the Program

The program lends itself to several aspects of customization. The next sections discuss these labels.

Labels X and Y

Labels X and Y hold the code for the two custom mathematical transformations. The code in label T invokes labels X and Y. By using two dedicated labels, the user can easily edit the code in these labels to customize the transformations. Each label should be coded such that it gets a value from stack register X and returned the transformed value in that same register. You can use the memory registers that are not used by the program to store intermediate results.

The default coding for labels X and Y is:

```
X001    LBL X
X002    3
X003    1/x
X004    y^x
X005    RTN
Y001    LBL Y
Y002    3
Y003    1/x
Y004    y^x
Y005    1/x
Y006    RTN
```

The above labels support the transformations that return the cube root and its reciprocal.

Label Z

Label Z contains a single command—one that sets the "default" display format. Currently, label Z sets the display format to FIX 5 (in line Z002 of the listing). You can edit this command to offer your preferred display format. The program uses label Z to switch between FIX 0, needed to efficiently display the transformation indices, and the "default" display format. Here is the current implementation of label Z:

```
Z001    LBL Z
Z002    FIX 5
Z003    RTN
```

Label K

Label K sets the default transformation group numbers. This value is currently set to 4210 (in line K004 of the listing). You can easily edit this value to set different initial default transformation group numbers. You can also invoke XEQ J to dynamically change the transformation group numbers. These numbers will remain in effect until you clear the machine, clear the memory, or execute either command XEQ K or XEQ J. Here is the current implementation of label K:

```
K001    LBL K
K002    0
K003    STO I
K004    4210
K005    [REGX,REGX]
K006    STO (I)
K007    RTN
```

Program Listing

Line	Command	Comment
Z001	LBL Z	Set/restore display mode
Z002	FIX 5	
Z003	RTN	
X001	LBL X	Label to contain the custom transformation #1
X002	3	
X003	1/x	
X004	y^x	
X005	RTN	
Y001	LBL Y	Label to contain the custom transformation #2
Y002	3	
Y003	1/x	
Y004	y^x	
Y005	1/x	
Y006	RTN	
I001	LBL I	Initialize data points
I002	0	
I003	STO C	
I004	RTN	
A001	LBL A	Add a new data point
A002	XEQ G	Get the data point
A003	1	Increment data point counter
A004	RCL+ C	
A005	STO C	
A006	STO I	
A007	R↓	
A008	STO (I)	Store the new data point
A009	R↑	
A010	RTN	
E001	LBL E	Edit a data point
E002	INPUT I	Get the index of the data point to overwrite
E003	XEQ H	Validate the index
E004	XEQ G	Get the new data point
E005	STO (I)	
E006	RTN	
D001	LBL D	Delete an existing data point
D002	INPUT I	Get the index of the data point to delete
D003	XEQ H	Validate the index
D004	RCL C	
D005	x=y?	Is the data point to delete the last one?
D006	GTO D020	Skip loop to overwrite data point

Line	Command	Comment
D007	RCL I	
D008	1	
D009	+	
D010	STO J	
D011	RCL (J)	Start loop to overwrite deleted data point
D012	STO (I)	
D013	1	
D014	STO+ I	
D015	STO+ J	
D016	RCL I	
D017	RCL C	
D018	x<y?	Continue overwriting?
D019	GTO D011	Resume with next iteration
D020	1	Decrement data point counter
D021	STO- C	
D022	RCL C	
D023	RTN	
H001	LBL H	Validate an index
H002	CF 1	
H003	RCL C	
H004	x<y?	Index exceed data point counter?
H005	SF 1	
H006	R↓	
H007	0	
H008	x≥y?	Index is 0 or less?
H009	SF 1	
H010	R↓	
H011	FS? 1	Invalid index?
H012	GTO R001	Display error message
H013	CF 1	
H014	RTN	
S001	LBL S	Swap data points
S002	INPUT I	
S003	XEQ H	Validate index
S004	INPUT J	
S005	XEQ H	
S006	RCL (I)	Recall data points on the stack
S007	RCL (J)	
S008	STO (I)	Store data point in swapped indices
S009	x↔y	
S010	STO (J)	
S011	RTN	
V001	LBL V	View a data point
V002	INPUT I	Get the index of the data point to view
V003	XEQ H	Validate index

Line	Command	Comment
V004	RCL (I)	Recall data point
V005	STOP	
V006	1	
V007	RCL I	
V008	+	
V009	STO I	
V010	GTO V003	
W001	LBL W	View all data points
W002	1	
W003	RCL C	
W004	1E3	
W005	÷	
W006	+	
W007	STO I	Store loop index
W008	RCL I	Start loop to view data points
W009	RCL (I)	
W010	PSE	
W011	ISG I	
W012	GTO W008	Resume to next loop iteration
W013	RTN	
G001	LBL G	Get a data point
G002	INPUT X	Prompt user for x value
G003	INPUT Y	Prompt user for y value
G004	[X,Y]	Create data point
G005	RTN	
R001	LBL R	Display INVALID DATA message
R002	CF 1	
R003	ACOS(2)	
R004	RTN	
T001	LBL T	Transform data. Works for x and y data.
T002	x≠0?	
T003	GTO T007	
T004	x↔y	
T005	LN	ln(x) transformation
T006	RTN	
T007	1	
T008	x≠y?	Transformation index is not 1?
T009	GTO T012	
T010	REGZ	No transformation
T011	RTN	
T012	1	
T013	+	
T014	x≠y?	Transformation index is not 2?
T015	GTO T019	
T016	REGZ	

Line	Command	Comment
T017	x^2	x^2 transformation
T018	RTN	
T019	1	
T020	+	
T021	x≠y?	Transformation index is not 3?
T022	GTO T027	
T023	REGZ	
T024	3	
T025	y^x	x^3 transformation
T026	RTN	
T027	1	
T028	+	
T029	x≠y?	Transformation index is not 4?
T030	GTO T034	
T031	REGZ	
T032	1/x	1/x transformation
T033	RTN	
T034	1	
T035	+	
T036	x≠y?	Transformation index is not 5?
T037	GTO T041	
T038	REGZ	
T039	√x	√x transformation
T040	RTN	
T041	1	
T042	+	
T043	x≠y?	Transformation index is not 6?
T044	GTO T049	
T045	REGZ	
T046	x^2	
T047	1/x	1/ x^2 transformation
T048	RTN	
T049	1	
T050	+	
T051	x≠y?	Transformation index is not 7?
T052	GTO T057	
T053	REGZ	
T054	√x	
T055	1/x	1/√x transformation
T056	RTN	
T057	1	
T058	+	
T059	x≠y?	Transformation index is not 8?
T060	GTO T064	

Line	Command	Comment
T061	REGZ	
T062	XEQ X	Invoke first custom transformation
T063	RTN	
T064	REGZ	
T065	XEQ Y	Invoke second custom transformation
T066	RTN	
P001	LBL P	Parse the transformation index group
P002	10	
P003	÷	
P004	ENTER	
P005	IP	
P006	x↔y	
P007	FP	
P008	10	
P009	×	
P010	IP	Update the transformation index group in Y stack register
P011	RTN	Index for next transformation in X stack register
B001	LBL B	Find Best fit
B002	RCL O	
B003	[1,0,0]	
B004	×	Get 3rd best r^2
B005	RCL R	Get current r^2
B006	x<x?	Current r^2 is less than 3rd best r^2?
B007	RTN	Exit
B008	RCL M	
B009	[1,0,0]	
B010	×	
B011	RCL R	Get 2nd best r^2
B012	x>y?	Current r^2 is better than 2nd best r^2?
B013	GTO B019	
B014	[R,G,H]	Found new 3rd best r^2
B015	STO O	
B016	[m,b]	
B017	STO P	
B018	RTN	Exit
B019	RCL K	
B020	[1,0,0]	
B021	×	Get best r^2
B022	RCL R	
B023	x>y?	Is the current r^2 better than best r^2?
B024	GTO B032	
B025	[R,G,H]	Found new 2nd best r^2
B026	x↔ M	Copy old 2nd best to 3rd best
B027	x↔ O	

Line	Command	Comment
B028	[m,b]	
B029	x↔ N	
B030	x↔ P	
B031	RTN	Exit
B032	[R,G,H]	Found new best r^2
B033	x↔ K	Move old 2nd best to 3rd best fit
B034	x↔ M	
B035	x↔ O	
B036	[m,b]	Move old best to 2nd best fit
B037	x↔ L	
B038	x↔ N	
B039	x↔ P	Store new best fit
B040	RTN	Exit
K001	LBL K	Initialize/Reset transformation groups
K002	0	
K003	STO I	
K004	4210	
K005	[REGX,REGX]	
K006	STO (I)	
K007	RTN	
J001	LBL J	View/Select transformation groups
J002	0	
J003	STO I	
J004	RCL (I)	
J005	[1,0]	
J006	×	
J007	STO X	
J008	RCL (I)	
J009	[0,1]	
J010	×	
J011	STO Y	
J012	INPUT X	Prompt for transformation group for x
J013	INPUT Y	Prompt for transformation group for y
J014	[X,Y]	
J015	STO (I)	Update transformation groups
J016	RTN	
C001	LBL C	Perform the linearized regression and find the best fit
C002	3	
C003	RCL C	Not enough data?
C004	x<y?	
C005	GTO R001	
C006	[0,0]	Initialize best r^2 registers
C007	STO L	
C008	STO N	

Line	Command	Comment
C009	STO P	
C010	2	
C011	[-1,0,0]	
C012	STO K	Store -1 in best r^2
C013	×	
C014	STO M	Store -2 in best r^2
C015	LASTx	
C016	+	
C017	STO O	Store -3 in best r^2
C018	0	
C019	STO I	
C020	RCL (I)	Recall master value for transformation groups
C021	[1,0]	
C022	×	
C023	STO E	Store Tx group value
C024	RCL (I)	
C025	[0,1]	
C026	×	
C027	STO D	Store Ty group value
C028	RCL D	Outer loop to transform y -------------------
C029	XEQ P	Parse the Ty group value
C030	STO H	Store the next Ty value
C031	x↔y	
C032	STO D	Store the reduced value of the Ty group
C033	RCL E	
C034	STO F	Copy initial Tx group value
C035	RCL F	Intermediate loop to transform x ----------------
C036	XEQ P	Parse the Tx group value
C037	STO G	Store the next Tx value
C038	x↔y	
C039	STO F	Store the reduced value of the Tx group
C040	1	
C041	RCL C	
C042	1E3	
C043	÷	
C044	+	
C045	STO I	Initialize loop to process data points
C046	ClΣ	
C047	FIX 0	
C048	[G,H]	
C049	PSE	
C050	XEQ Z	Restore display format
C051	RCL (I)	Innermost loop ---------------------------------------
C052	[1,0]	

Line	Command	Comment
C053	×	
C054	RCL G	Get current Tx value
C055	XEQ T	Transform x
C056	STO X	
C057	RCL (I)	
C058	[0,1]	
C059	×	
C060	RCL H	Get current Ty value
C061	XEQ T	Transform y
C062	STO Y	
C063	RCL X	
C064	Σ+	
C065	ISG I	
C066	GTO C051	End of innermost loop - - - - - - - - - - - - - - - -
C067	r	
C068	x^2	
C069	STO R	Store r^2. You can insert a R/S or PSE command before the next line to stop or pause the program in order to examine the value of r^2.
C070	XEQ B	Rank r^2 and regression results among the best
C071	RCL F	Next Tx?
C072	x≠0?	
C073	GTO C035	End of intermediate loop - - - - - - - - - - - - - - - -
C074	RCL D	
C075	x≠0?	
C076	GTO C028	End of outer loop -
F001	LBL F	View best three fits
F002	SF 10	
F003	BEST FIT	This is a "tag" created using EQN button
F004	PSE	Display output tag
F005	FIX 0	
F006	RCL K	Show transformation indices
F007	PSE	
F008	XEQ Z	Restore display format
F009	RCL L	Best fit
F010	STOP	
F011	2 ND BEST	This is a "tag" created using EQN button
F012	PSE	Display output tag
F013	FIX 0	
F014	RCL M	Show transformation indices
F015	PSE	
F016	XEQ Z	Restore display format
F017	RCL N	Best 2nd fit
F018	STOP	
F019	3RD BEST	This is a "tag" created using EQN button

Line	Command	Comment
F020	PSE	Display output tag
F021	CF 10	
F022	FIX 0	
F023	RCL O	Show transformation indices
F024	PSE	
F025	XEQ Z	Restore display format
F026	RCL P	Best 3rd fit
F027	RTN	

Example

Let's look at an example that exercises various routines in the regression program. The next Table contains (x, y) data points that we want to fit with the best model.

x	y
1	1
2	4
3	9
4	16
5	25
6	36
7	49
8	64
9	81
10	100

The above data represents points for $y=x^2$. This is an example where two models will provide the best fit—the quadratic fit (y vs. x^2) and the power fit (ln(y) vs. ln(x)).

Perform the following tasks:

1. Initialize the transformation groups numbers
2. Use the transformation group values of 210 for both variables to determine the best fit.
3. Reset the data point counters
4. Enter the data points shown in the above table.
5. Calculate and display the best fits.

The following table shows the tasks using the FIX 5 display mode as set by label Z.

Task #	Task/Subtask	Input	Command	Output
1	Initialize the transformation group numbers.		XEQ K	
2	Set the transformation groups.		XEQ J	X?
	Enter the new transformation group for variable x.	210	R/S	Y?
	Enter the new transformation group for variable y.	210	R/S	
3	Initialize the program.		XEQ I	0
4	Add the first data point.		XEQ A	X?
	Enter a value for x.	1	R/S	Y?
	Enter a value for y.	1	R/S	1
5	Add the second data point.		XEQ A	X?
	Enter a value for x.	2	R/S	Y?
	Enter a value for y.	4	R/S	2
6	Repeat step 3 for all other data point.			10.00000
7	View all data points (Note the program displays the results using the PSE statement). If you spot an error use the command XEQ E to overwrite the erroneous		XEQ W	10.01000 [10.00000,100.00000]

Task #	Task/Subtask	Input	Command	Output
	data point with the correct one.			
8	Calculate the regression coefficients.		XEQ C	[1., 2., 1.]
				[1.00000,2.00000] [1. 00000,0.00000]
	Best fit is y= 0 + 1 * x^2 with r^2 = 1.		R/S	[1., 0., 0.]
				[1..00000,0.00000] [2.00000,9.371E-11]
	Second best curve fit is: ln(y) = 9.371E-11 + 2 ln(x) with r^2 = 1.		R/S	[1., 1., 1.]
				[0.94876,1.00000] [11.00000,-22.00000]
	Third best curve fit is: y = -22 + 11 x with r^2 = 0.94876.			

The above example provides the following best curve fits:

- The best curve fit is y = 0 + x^2 with r^2 = 1
- The second best fit is ln(y) = 9.371E-11 + 2 ln(x) with r^2 = 1. This is considered an alternate best fit. This case is rather unique.
- The third best fit is y = -22 + 11 x with r^2 =0.94876.