Elementary Mathematical Models, 2nd Edition

Dan Kalman, Sacha Forgoston, Albert Goetz

Technology Guide

Section 1: Introduction

This guide discusses calculator and computer technology options for numerical computation and graphics useful in the study of elementary mathematical models. It is organized into the following sections:

1. Introduction
2. Home Screen Iteration on Calculators
3. Tables and Graphs on Calculators
4. Tables and Graphs on Computers
5. Appendix A: Curve Fitting
6. Appendix B: Quadratic Growth Using Home Screen Iteration

The Elementary Mathematical Models (EMM) text emphasizes numerical and graphical approaches to understanding and using models. In particular, there are many examples and exercises featuring the systematic computation of terms of a sequence and/or graphs for sequences. There are also many examples and problems that involve developing models based on genuine data. Both graphical and numerical approaches are used to find models that match the given data closely, using informal trial-and-error curve fitting.

In our EMM classes, we consistently use technology for computation and to create graphs. Making it easy for students to quickly obtain accurate graphs and computational results encourages active exploration of the meaning and limitations of models. In the sections to follow we discuss calculator and computer resources that we have found particularly suited to our instructional goals.

Calculators. It is not our intention to survey all available calculators. We will provide detailed instructions for two different Texas Instruments products, a scientific calculator and a graphing calculator. This is partly a matter of convenience, and partly a reflection of the popularity of these calculators. However, we are not recommending TI products specifically; we make no comments on the relative merits of these calculators in comparison with calculators provided by other manufacturers.

In presenting this material, we have two goals. First, for those who wish to use TI calculators, this guide will explain how to perform various operations. Second, the guide should give readers a detailed picture of the capabilities of scientific and graphing calculators, and how they can be used in conjunction with our text.

Texas Instruments provides computer software emulators for its calculators. These can be used in a classroom setting with a computer projector, to demonstrate how to operate the calculator and interpret the results. We used the emulators to produce the screen images and key sequence diagrams that appear in this Guide. We acknowledge here that TI donated copies of the emulator software to our project for this purpose.

Smart phone applications are an alternative to hand held calculators. There are currently quite a variety of such applications, including emulators for most TI graphing calculators. A survey of such applications is beyond the scope of this work. However, we note that a key theme of EMM is the study of sequences defined by difference equations, and not all graphing calculators provide tools for this purpose. The calculator operations discussed here should provide some insights about features to look for if a phone application is under consideration.

Notation for Calculator Examples. Capabilities of calculators are illustrated below in several examples. These should enable the reader to duplicate each step on an actual calculator. For efficient description of required operations, we represent calculator keys as follows. Digits and arithmetic operations (such as + and ×), as well as decimal points and commas, are represented as normal text. We enclose key labels in brackets. So, for example, [enter] represents the *enter* key. There are variations in the labels that appear on different calculators. For this guide, we will not distinguish among minor differences in label formats. Thus [enter] represents a key labeled with “enter”, “ENTER”, or any other variant, and similarly for other key labels.

In some instances an operation is entered as a combination of keys, first pressing [2nd] or [alpha], and then another key. As an example, pressing first [2nd] and then [x^2] activates the square root function. This is indicated on the calculator keyboard by the appearance of the radical symbol √ above the [x^2] key. In our examples this combination would be represented as [2nd √]. Similarly, we show the command to retrieve the result of the calculator’s prior completed operation as [2nd ans]. To key this into the calculator, first press the [2nd] key, then press the key above which the label “ans” appears. That is the [(-)] key.

We also use a special convention to represent text that appears on the calculator screen, displaying it in **this font**.

Computers. In discussing the use of computers, we focus primarily on spreadsheet software. As is the case for calculators, we consider just one product, Microsoft Excel, and with similar caveats as above: this is a matter of convenience, and does not constitute a recommendation of Excel over competing products.

Webpages that perform mathematical operations offer one alternative to spreadsheet software. We will comment briefly on Wolfram Alpha.

Section 2: Home Screen Iteration on Calculators

In the Elementary Mathematical Models text, many of the models are defined in terms of a difference equation. This permits successive terms of a sequence to be computed by operating on the preceding terms. In most cases this process can be automated on a calculator using *Home Screen Iteration*. An example will show how this works.

> **Example 2.1.** A sequence begins with 12.67 and each term is found by adding 11.8 to the preceding term. We would like to compute the first five terms. We proceed as shown in the table below.

Step Number	Key In This	How Input Will Be Displayed	How Output Will Be Displayed
1	12.67 [enter]	**12.67**	**12.67**
2	+ 11.8 [enter]	**ans + 11.8**	**24.47**
3	[enter]	**ans + 11.8**	**36.27**
4	[enter]	**ans + 11.8**	**48.67**
5	[enter]	**ans + 11.8**	**59.87**

At step 1, the initial term is entered. Notice at step 2 that when the + key is pressed the calculator shows **ans +**. Here, **ans** refers to the result of the prior step, namely 12.67. This shows that in step 2 the prior answer will be added to 11.8. In each remaining step, when the enter key is pressed, the entire preceding operation is repeated. It is as if the user had rekeyed + 11.8 [enter] each time. Thus at each step the calculator adds 11.8 to the result of the preceding step.

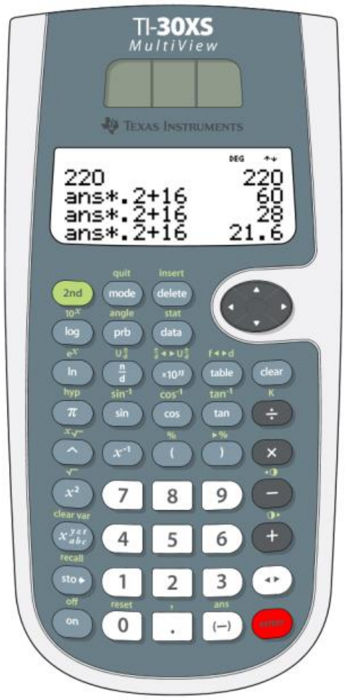
This process works on both scientific and graphing calculators. It can be applied whenever the next term to be found can be computed from the preceding term with a single calculator step. At right a similar computation is performed on a TI-30XS scientific calculator. This time the initial term is 220, and each new term is found by multiplying the preceding term by .2 and then adding 16. As before, the user does not actually key in the *ans* on each line; it is supplied by the calculator. The user keys in

220 [enter],

then

× .2 + 16 [enter].

That produces a result of 60. Then the user simply presses [enter] repeatedly.

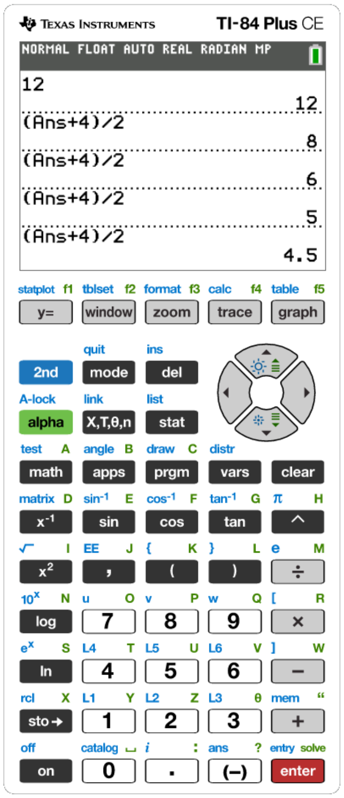


```
220                220
ans*.2+16           60
ans*.2+16           28
ans*.2+16         21.6
```

In some cases it is necessary to actually key in the *ans* from the preceding step, and a key on the calculator is available for that purpose. This is shown in the next example.

Example 2.2. A sequence begins with 12 and each term is found by adding 4 to the preceding term and then dividing the result by 2. We can compute the successive terms as shown in this table:

Step Number	Key In This	How Input Will Be Displayed	How Output Will Be Displayed
1	12 [enter]	**12**	**12**
2	([2nd ans]+4)÷2 [enter]	**(Ans + 4)/2**	**8**
3	[enter]	**(Ans + 4)/2**	**6**
4	[enter]	**(Ans + 4)/2**	**5**
5	[enter]	**(Ans + 4)/2**	**4.5**



```
NORMAL FLOAT AUTO REAL RADIAN MP
12
                    12
(Ans+4)/2
                     8
(Ans+4)/2
                     6
(Ans+4)/2
                     5
(Ans+4)/2
                   4.5
```

In the second step we need a left parenthesis to precede the answer from the prior step. That is why we must key in that prior answer explicitly, using the *ans* function. This is shown in table as [2nd ans], which indicates pressing first the [2nd] key and then the [(-)] key as mentioned earlier. The image shows the calculator after performing all the steps in the table.

In some cases, the preceding answer enters into a computation more than once. This is illustrated in the next example.

Example 2.3. A sequence begins with one, and each term is found by multiplying the preceding term by a linear function of that same preceding term. Specifically, if we denote a particular term by p, the next term is computed as $p \times (2.4 - 0.8p)$. To compute terms of this sequence using home screen iteration, we have to include *ans* for each instance of the preceding term p. Starting from an initial term of 1, the table for this computation would be:

Step Number	Key In This	How Input Will Be Displayed	How Output Will Be Displayed
1	1 [enter]	**1**	**1**
2	×(2.4-.8×[2nd ans]) [enter]	**Ans*(2.4-.8*Ans)**	**1.6**
3	[enter]	**Ans*(2.4-.8*Ans)**	**1.792**
4	[enter]	**Ans*(2.4-.8*Ans)**	**1.7317888**

Here, in step 2, *Ans* enters the computation twice. It is included automatically by the calculator with the first key press of ×. Then after the second × the user includes *Ans* again by explicitly keying in the [2nd ans] key combination.

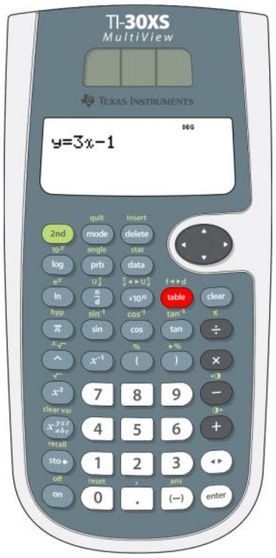
Home screen iteration can be used for all of the sequences in the text, except for quadratic growth sequences. For these, each term depends not only on the preceding term, but also on the term's position in the sequence. See Appendix B for a discussion of how home screen iteration can be adapted to quadratic growth sequences on TI graphing calculators.

Section 3: Tables and Graphs on Calculators

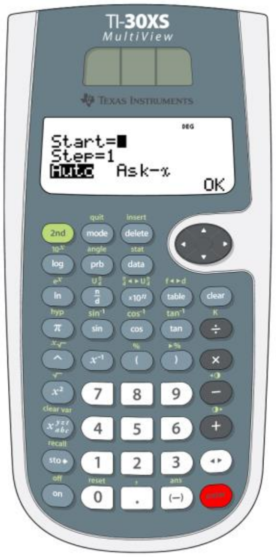
Home screen iteration allows rapid calculation of terms of a sequence, without any direct use of variables and equations. This has the advantage of being easy to use and understand. However, there are some limitations. For one thing, only the terms of the sequence that appear on the screen are accessible: it is not possible to scroll up to view terms that have rolled off the top of the display. For another, there is no explicit indication of the position of each term within the sequence. That is, the display shows the numerical value of a term, but does not indicate whether it is the ninth term or the ninety-ninth. In the text, the expression *position number* is used in this context. The ninth term is said to have a position number of 9; the ninety-ninth has position number 99.

An alternative to home screen iteration is to create a table showing both the terms of a sequence and the corresponding position numbers. We describe two approaches to this task below.

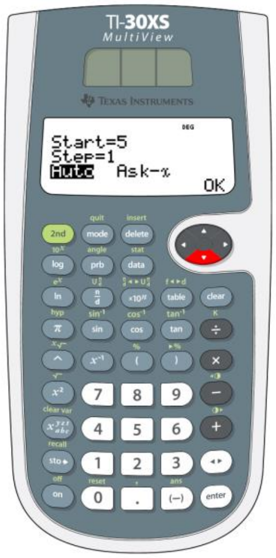
Tables for Functions. A *functional equation* expresses the terms of a sequence as a function of the position number. For example, $a_n = 3n - 1$ is a functional equation. We can also express it in the standard function notation as $a(n) = 3n - 1$. Both scientific and graphing calculators generate tables for such a function.



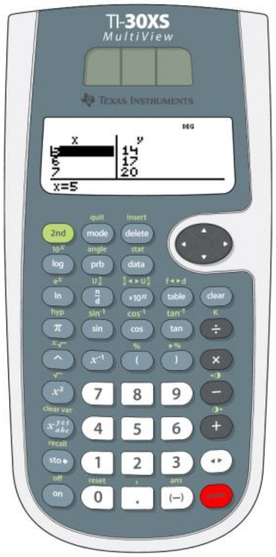
```
y=3x-1
```



```
Start=█
Step=1
Auto  Ask-x
             OK
```



```
Start=5
Step=1
Auto  Ask-x
             OK
```



```
  x   |  y
5     | 14
6     | 17
7     | 20
x=5
```

The figures above show how a table is obtained on a TI-30 calculator. When the [table] key is pressed, the first image appears. (Note that the [table] key is highlighted in red.) The equation for y is edited by the user to show the desired function, for this example, $y = 3x - 1$. Pressing the [enter] key then shows the second image. The user enters the starting value of the position number (we will use 5 for this example), and the step size. For consecutive terms of a sequence, the step size will be 1. In the third image all of the necessary information has been entered. Positioning the cursor on **OK** and pressing the [enter] key then produces the final image. It shows the values of *x* (which are position numbers) and *y* (the terms of the sequence). This table is scrollable: using the up and down arrow keys reveals additional lines above and below the current display.

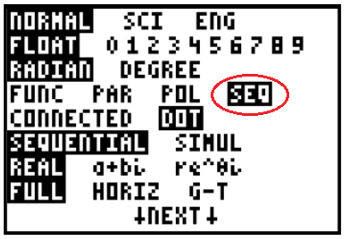
Graphing calculators operate similarly to produce a table for a functional equation. The equation is entered after pressing the [y =] key, and the initial value and step size defined after pressing the [2nd tblset] key. Pressing [2nd table] then displays a scrollable table. In addition, a graph can be displayed using the [graph] key. Settings for the graph can be defined using the [window] key.

Tables and Graphs for Difference Equations. Sequences can also be defined using a *difference equation*. This specifies how each term of a sequence can be computed using one or more preceding terms, and possibly the position number. Graphing calculators permit entry of a difference equation to define a sequence, and then can produce tables and graphs. Here we discuss these operations for the TI-84 calculator.

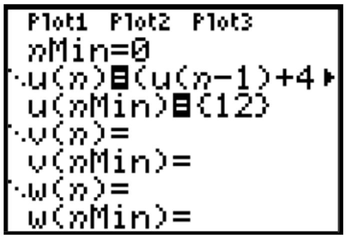
In Example 2.2, a sequence begins with 12 and each term is found by adding 4 to the preceding term and then dividing the result by 2. Using the same notation as in the text, the difference equation for this sequence would be expressed as either $a_{n+1} = (a_n + 4)/2$ or as $a_n = (a_{n-1} + 4)/2$. On the TI-84 calculator, only the variables *u*, *v*, and *w* can be used to represent terms in a difference equation. Also, parenthesis notation is used in place of subscript notation, and the left side of the difference equation is $a(n)$, not $a(n+1)$. Different calculators use different conventions. For example, the TI-82 calculator uses subscript notation.

The next example shows how a table and graph can be obtained on the TI-84 calculator for the same sequence as in Example 2.2.

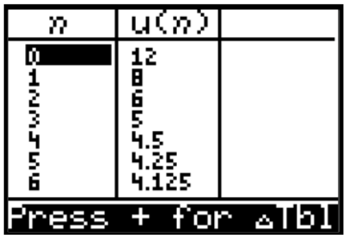
Example 3.1. A sequence begins with 12 and each term is found by adding 4 to the preceding term and then dividing the result by 2. We formulate the difference equation as $u_n = (u_{n-1} + 4)/2$. To enter this in the calculator, press the [mode] key and specify sequence mode, as illustrated at left below.



```
NORMAL  SCI  ENG
FLOAT  0 1 2 3 4 5 6 7 8 9
RADIAN  DEGREE
FUNC  PAR  POL  SEQ
CONNECTED  DOT
SEQUENTIAL  SIMUL
REAL  a+bi  re^θi
FULL  HORIZ  G-T
      ↓NEXT↓
```



```
Plot1  Plot2  Plot3
nMin=0
·u(n)=(u(n-1)+4▸
 u(nMin)={12}
·v(n)=
 v(nMin)=
·w(n)=
 w(nMin)=
```

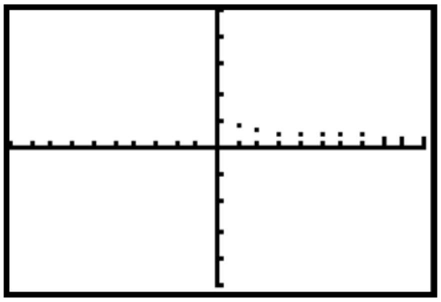


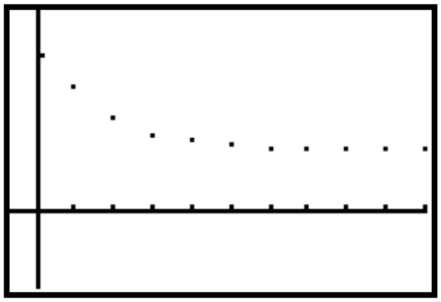
```
 n    u(n)
 0    12
 1    8
 2    6
 3    5
 4    4.5
 5    4.25
 6    4.125
Press + for ΔTbl
```

Next, press the [Y =] key producing a screen similar to the one in the center above. We assign the values *n*Min = 0 and u(*n*Min) = 12. This identifies 0 as the initial position number and 12 as the initial term of the sequence. We also complete the difference equation for u_n by filling in the right side of the equation for **u(*n*)**. The variable **u** is entered as [2nd u] (using the 7 key) and **n** is entered by pressing [x,T,θ,n]. The result is shown in the center image above. Note that the full difference equation does not appear because it is too long for the display. The complete equation would look like this: **u(*n*) = (u(*n*-1)+4)/2.**

Finally, we can obtain a table or graph for this sequence. Activate the table function by entering [2nd table]. The resulting screen appears at right above. The table shows both the position number *n* and the value *u*(*n*) for each term, and is scrollable.

Similarly, a graph for the sequence is produced by pressing the [graph] key. The image below at left shows the graph for this example with the default window settings. The graph at right was obtained using the [zoom] key. Formatting options for the graph can be set by pressing the [window] key. Formatting options for the table are set by pressing [2nd tblset]. Tables and graphs for functional equations are produced similarly, operating the calculator in *function* (**FUNC**) mode.



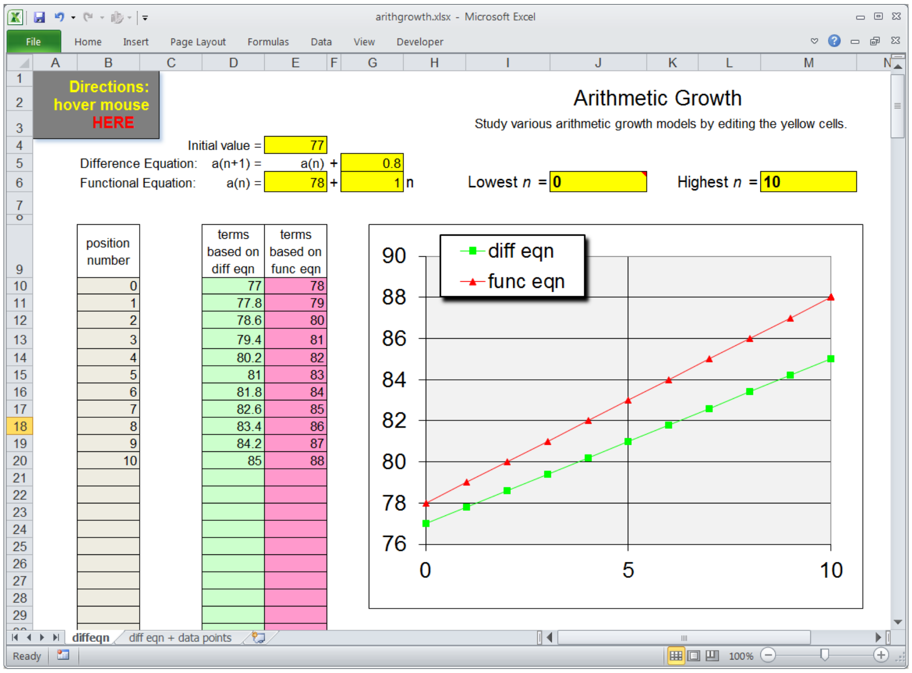


Section 4: Tables and Graphs on Computers

Tables and Graphs with Excel. A spreadsheet program displays an array of cells into which various types of data can be entered. The value in one cell may be defined in relation to values in other cells. This provides a convenient way to compute terms of a sequence. Like home screen iteration, spreadsheets allow the user to specify easily how one sequence term depends on preceding terms. But because the results are displayed in a table, the spreadsheet shares some of the advantages of the tables produced by a calculator.

There are a variety of spreadsheet programs for computers, as well as on calculators and smart phones. We will not show here how to create a spreadsheet file that computes terms of a sequence. Rather, we focus on a collection of files we have prepared using the Excel program. These are designed to facilitate the creation of tables and graphs for sequences, requiring the user only to enter numerical values for a few constants. For those without access to Excel, there are many alternative programs. While we have not surveyed them, we have had some limited experience with *LibreOffice Calc*, a free program that seems to open our xls files correctly and preserve all or most of the functionality. This program is available from https://www.libreoffice.org/ .

To illustrate, the image below shows a screen shot from one of the files we have created.



Arithmetic Growth

Study various arithmetic growth models by editing the yellow cells.

Initial value = 77

Difference Equation: a(n+1) = a(n) + 0.8

Functional Equation: a(n) = 78 + 1 n

Lowest *n* = 0

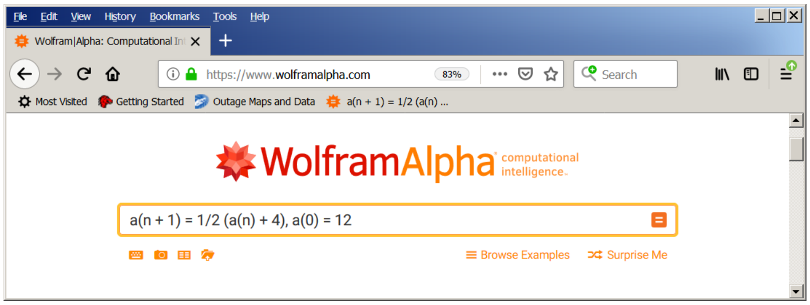
Highest *n* = 10

position number	terms based on diff eqn	terms based on func eqn
0	77	78
1	77.8	79
2	78.6	80
3	79.4	81
4	80.2	82
5	81	83
6	81.8	84
7	82.6	85
8	83.4	86
9	84.2	87
10	85	88

This spreadsheet is specialized to exploring arithmetic growth sequences, where each term is computed by adding a constant to the preceding term. The user enters numerical values in the yellow cells to define a difference equation, a functional equation, and a range of values for *n*, representing position numbers. The spreadsheet automatically populates a table of values and displays a graph.

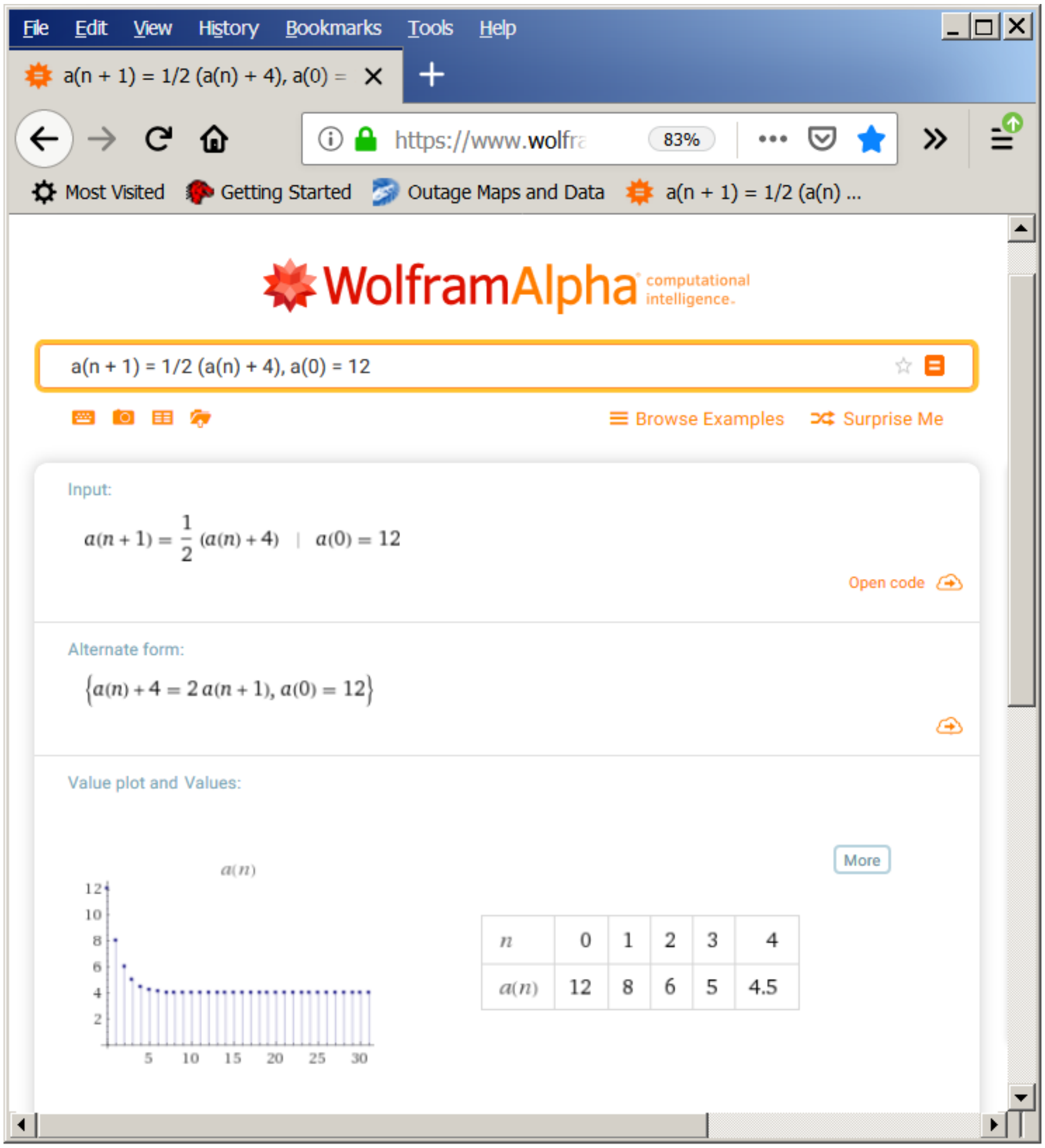
This is one of a collection of spreadsheets available at the emm2e.info website. There is also a User's Guide that provides a description and instructions for each spreadsheet. The spreadsheets provide an easy means for obtaining tables and graphs for all of the model types considered in the text. They also support an informal trial and error approach to fitting a model to a set of data. See the User’s Guide for more information about this resource.

Wolfram Alpha. This free website provides a natural-language interface to a broad range of mathematical tools. The figure below shows one way to enter the definition of the same sequence we considered in Examples 2.2 and 3.1.



a(n + 1) = 1/2 (a(n) + 4), a(0) = 12

After typing the expression in the orange box, hit the enter key. The website responds by interpreting the input and displaying related information. The image below shows part of the output, including a table and graph. Other information has been edited out of the image for the sake of clarity. Notice the box that says *More* above and to the right of the table of values. Clicking it will cause the webpage to display additional entries of the table.



a(n + 1) = 1/2 (a(n) + 4), a(0) = 12

Input:

$$a(n+1) = \frac{1}{2}(a(n)+4) \quad | \quad a(0) = 12$$

Alternate form:

$$\{a(n)+4 = 2\,a(n+1),\ a(0) = 12\}$$

Value plot and Values:

n	0	1	2	3	4
$a(n)$	12	8	6	5	4.5

Wolfram Alpha provides a simple resource for finding terms of a sequence in table and graphical form. The user has less control of how the graph is formatted and how many terms of the sequence are displayed than is the case for calculators and spreadsheets. But for many situations, the provided table and graph are sufficient to understand the properties of a sequence.

Section 5: Appendix A
Curve Fitting

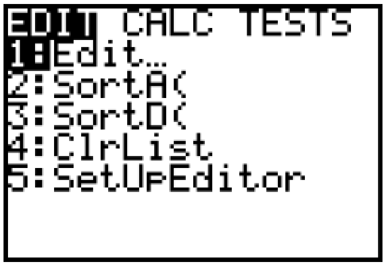
In the Elementary Mathematical Models text many examples and exercises involve fitting a model of a specified type to a set of data. This can be done in an informal trial-and-error process using spreadsheets in the collection described in Section 4. The text mentions the existence of more theoretical methods for curve fitting, but does not delve into the subject. Here we briefly mention a few options for obtaining optimized fits for those students and instructors who wish to pursue this approach. Curve fitting is sometimes referred to as *regression*.

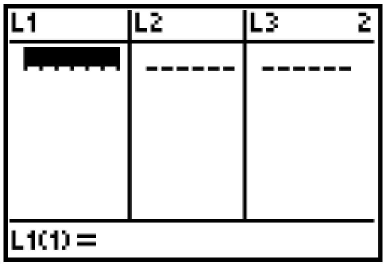
Calculators. Some graphing calculators have built in functions to obtain optimized fits of curves to data. In the example below a quadratic function is fit to a set of points.

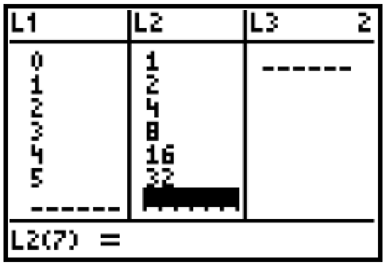
Example 5.1. We wish to fit a quadratic growth model to the sequence 1, 2, 4, 8, 16, 32. The terms of the sequence and the position numbers appear in the table at right, where we have assumed that the first term, 1, has position number 0.

position number	term
0	1
1	2
2	4
3	8
4	16
5	32

On the TI-84, each column of the table will be entered as a list. Press the [STAT] key to obtain the first screen shown below. Then key in 1 to obtain the edit window, as depicted in the second screen below with no defined entries in any of the lists. If list entries have been defined previously, the window will look similar to the third screen below.





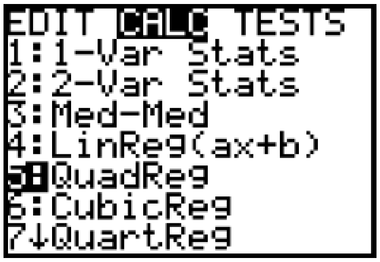


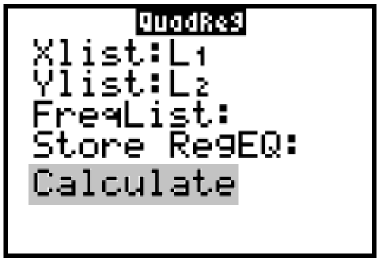
L1	L2	L3
0	1	------
1	2	
2	4	
3	8	
4	16	
5	32	

L2(7) =

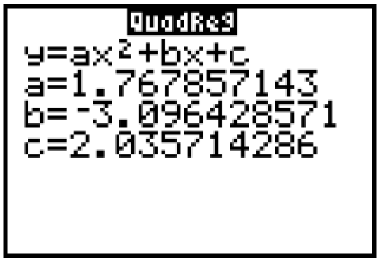
List values are entered as follows. First, use the ▶ and ◀ keys to move the insertion point into any desired column. If there are no defined entries in the column, the insertion point automatically goes to first position. Enter a numerical value and press either [enter] or ▼ to continue with the next entry. Entries that have already been defined can be changed by moving the insertion point with ▲ or ▼ to the desired location, and then entering a new value. To append new values to a previously defined list move the insertion point to the line of dashes. The third image above shows the calculator screen after entering the position numbers for our example in list 1 and the sequence terms in list 2.

Once the data values have been entered, press the [STAT] key again. This time use ▶ to highlight CALC as shown in the first screen below. Item 5 on the displayed menu activates the quadratic regression application. (Press 5 or else use cursor keys to highlight the 5 and then press the enter key.) That leads to the second screen below.





```
Xlist:L1
Ylist:L2
FreqList:
Store RegEQ:
Calculate
```



```
y=ax²+bx+c
a=1.767857143
b=-3.096428571
c=2.035714286
```

Observe that the Xlist and Ylist are identified as L_1 and L_2.[1] No input has been defined on the next two lines representing optional parameters. Using the cursor keys, highlight the calculate command and press the enter key. This reveals the third screen above. It gives the form of the quadratic equation, and the optimal values of the coefficients a, b, and c.

The TI-84 provides similar curve fitting applications for quite a few types of equations. The ones that are applicable in EMM are shown in the table below.

Description	Equation
linear	$y = mx + b$
quadratic	$y = ax^2 + bx + c$
exponential	$y = ab^x$
logistic	$y = \dfrac{c}{1 + ae^{-bx}}$

In the equations, a, b, c, and m are coefficients whose values are determined by the calculator. As far as we can tell, the calculator cannot fit an equation of the form $y = ab^x + c$. This type of equation is necessary for mixed models in chapter 5 of the text.

For linear and quadratic functions, the calculator implements a linear least-squares analysis. Here the coefficients enter into the equation linearly and there are exact theoretical formulas for the optimal coefficients. For the exponential equation the calculator computes a linear least-squares fit after transforming the equation using logarithms. Therefore, the coefficients obtained minimize the least-squares error[2] for the transformed data, but not for the original data. For the logistic equation a nonlinear optimization algorithm is used. This is iterative in nature, and attempts to minimize the least-squares error using the original (untransformed) data. Such algorithms generally find coefficients that are only approximately optimal.

[1] These inputs can be edited if data have been entered in different columns of the list definition screen. Notice that L_1 through L_6 appear as labels above the number keys 1 through 6. To access list L_3, for example, key in [2nd L_3].

[2] The least-squares error is computed as the square root of the sum of the squared differences between original *y* values and those produced by the model equation.

Excel. There are two different curve fitting procedures in Excel, corresponding to linear and nonlinear least-squares optimization. These are included in two application packages that are available in Excel, but are not always loaded when the software is installed. To add them to your installation, open the file tab at the top of the Excel window and select Options from the menu along the left side of the screen. On the window that pops up, select Add-Ins on the menu at left. Select the Analysis Toolpak for linear least-squares and Solver Add-in for the nonlinear least-squares applications.

Linear Least-Squares. Linear least-squares finds the best coefficients in an equation that is a linear combination of known columns or functions. As an example, we revisit the quadratic curve fitting problem discussed in Example 5.1.

Example 5.2. We wish to fit a quadratic growth model to the sequence shown in this table:

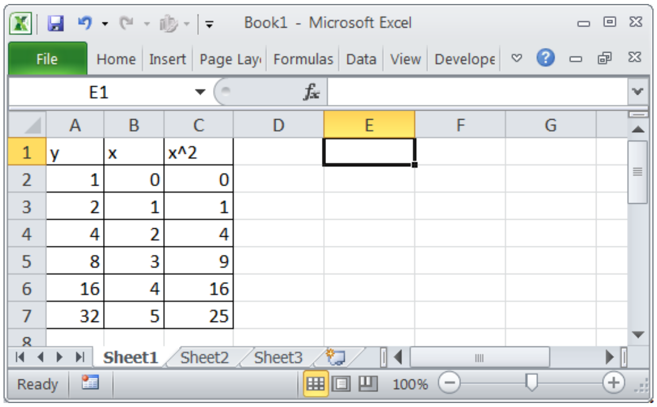
position number	0	1	2	3	4	5
term	1	2	4	8	16	32

We can consider the first row to be *x* values and the second to be *y* values. Then our goal is to find an equation of the form $y = a + bx + cx^2$ that approximates the *y* values as closely as possible. Expressed in terms of vectors or lists we want the equation

$$\begin{bmatrix} 1 \\ 2 \\ 4 \\ 8 \\ 16 \\ 32 \end{bmatrix} = a \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} + b \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix} + c \begin{bmatrix} 0^2 \\ 1^2 \\ 2^2 \\ 3^2 \\ 4^2 \\ 5^2 \end{bmatrix}$$

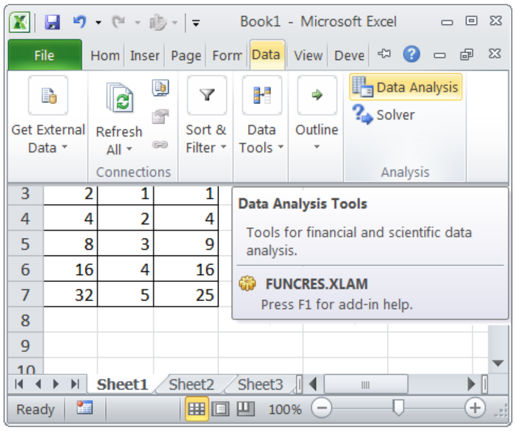
to hold as closely as possible. The column on the left side of the equation contains the *y* values. The columns on the right correspond to the constant function 1, the values of *x*, and the values of x^2. Finding the coefficients *a*, *b*, and *c*, is referred to as multiple regression, because we combine multiple known columns to approximate the *y* column.

Excel automatically includes the constant column on the right side of the equation. To prepare for the multiple regression function, we enter the non-constant columns in the spreadsheet as shown in Screen A below. The column labels *y*, *x*, and x^2 are included for convenience. These are not interpreted as inputs by the regression function. Next, select the Data tab at the top of the Excel window, and from the menu that appears, select Data Analysis. This is shown in Screen B, and causes the menu in Screen C to appear. Select Regression and click OK.



	A	B	C
1	y	x	x^2
2	1	0	0
3	2	1	1
4	4	2	4
5	8	3	9
6	16	4	16
7	32	5	25

Screen A



3	2	1	1
4	4	2	4
5	8	3	9
6	16	4	16
7	32	5	25

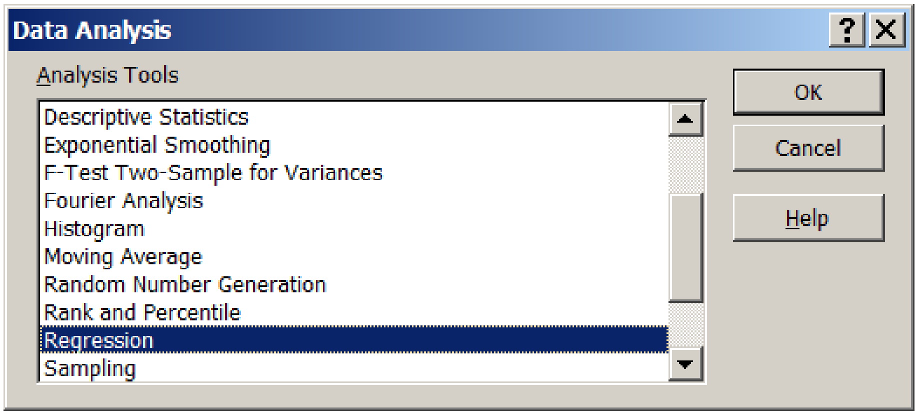
Data Analysis Tools

Tools for financial and scientific data analysis.

FUNCRES.XLAM

Press F1 for add-in help.

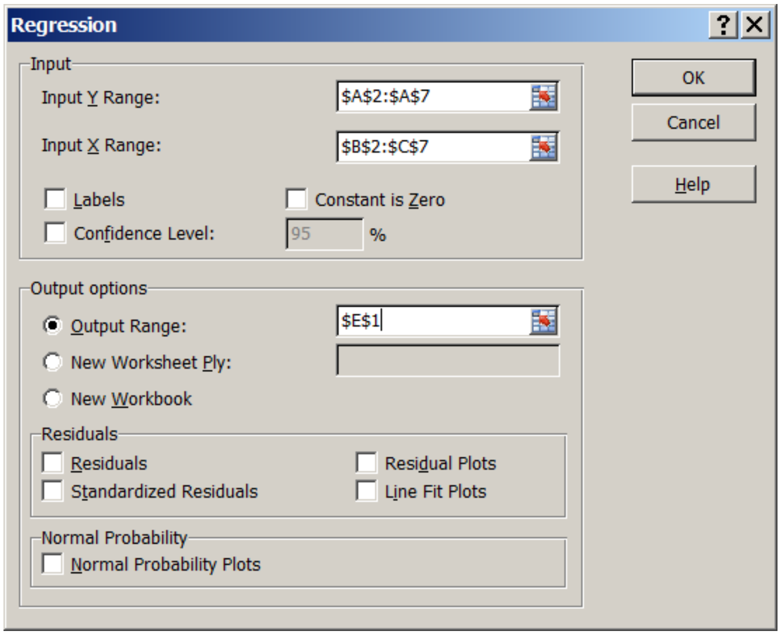
Screen B



Screen C

Now a window appears where we specify the inputs for the regression app. We enter the range of cells (A2 through A7) of y values as the Input Y Range. Likewise, we enter the range of both x and x^2 columns (B2 through C7) as the cells in the Input X Range. We also select the Output Range option and enter there the upper left-hand cell for the cells that will be output by the regression app. Clicking OK activates the app.

The result is shown in Screen E on the next page.



Input Y Range: $A$2:$A$7

Input X Range: $B$2:$C$7

□ Labels □ Constant is Zero

□ Confidence Level: 95 %

● Output Range: $E$1

○ New Worksheet Ply:

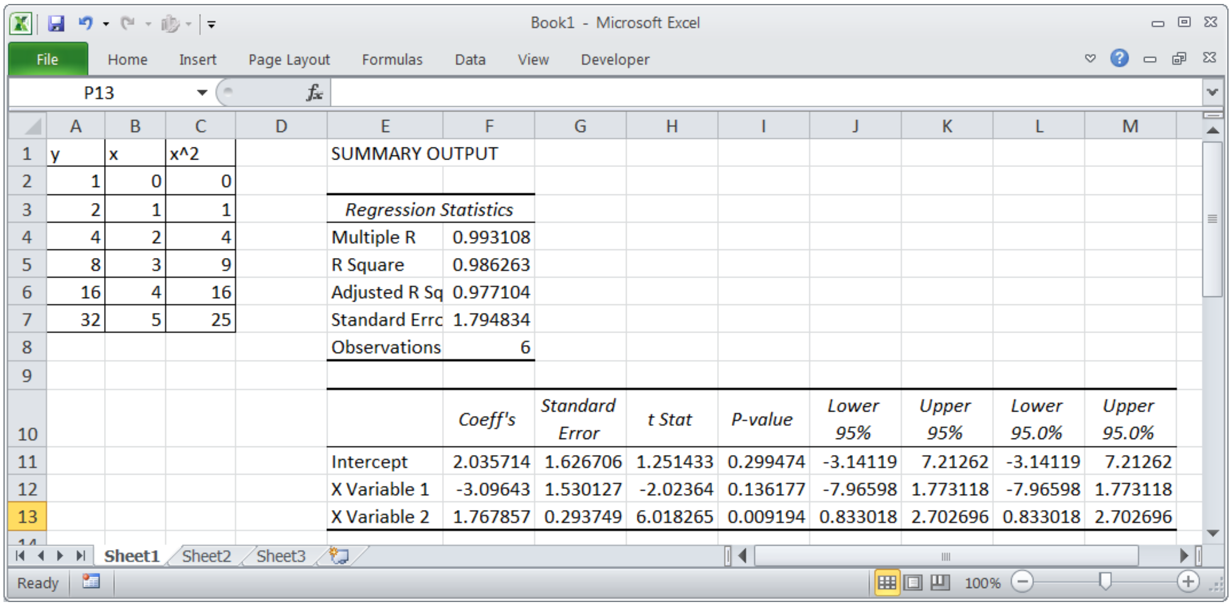
○ New Workbook

□ Residuals □ Residual Plots

□ Standardized Residuals □ Line Fit Plots

□ Normal Probability Plots

Screen D



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	y	x	x^2		SUMMARY OUTPUT								
2	1	0	0										
3	2	1	1		*Regression Statistics*								
4	4	2	4		Multiple R	0.993108							
5	8	3	9		R Square	0.986263							
6	16	4	16		Adjusted R Sq	0.977104							
7	32	5	25		Standard Errc	1.794834							
8					Observations	6							
9													
10						*Coeff's*	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
11					Intercept	2.035714	1.626706	1.251433	0.299474	-3.14119	7.21262	-3.14119	7.21262
12					X Variable 1	-3.09643	1.530127	-2.02364	0.136177	-7.96598	1.773118	-7.96598	1.773118
13					X Variable 2	1.767857	0.293749	6.018265	0.009194	0.833018	2.702696	0.833018	2.702696

Screen E

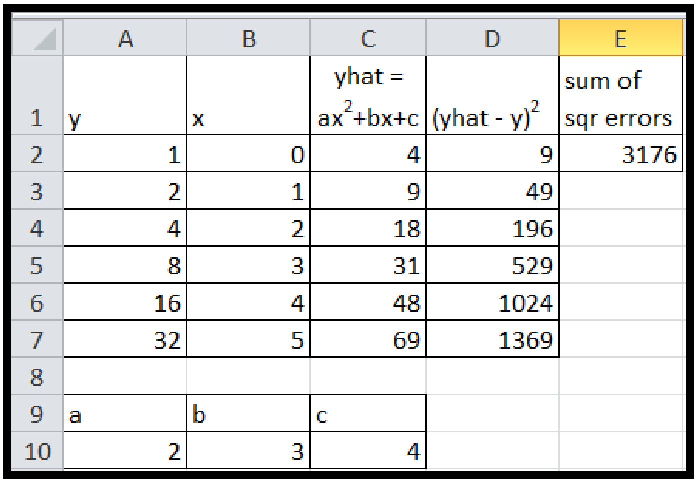
The coefficients in the regression equation are shown at the bottom of the image under the column head *Coeff's*. The intercept is given as 2.035714. That is the constant term of the equation. The row for X Variable 1 corresponds to the *x* column. Its coefficient is –3.09643. Similarly, X Variable 2 corresponds to the x^2 column, and has coefficient 1.767857. Thus, the regression results give us the equation

$$y = 1.767857x^2 - 3.09643x + 2.035714.$$

These coefficients are the same as those computed by the calculator quadratic regression in Example 5.1. They have been rounded off by Excel to fit the column width of the output cells. If Excel is formatted to show nine decimal places, the results here agree with those from the calculator exactly.

Nonlinear Least-Squares. The nonlinear least-squares app can be used to optimize a collection of parameters that enter into the calculation of a cell on a spreadsheet. First the sheet is arranged to calculate the target cell's value for one set of parameter values. Then, when the nonlinear least-squares app is launched, the user identifies which cells are parameters, what the target cell is, and what the goal of the optimization is. For fitting a function to data, we select as the target cell value the computed value of the sum of squared deviations between the original *y* data and the model data, and our goal is to minimize that value. The optimization app is iterative in nature, and may find a local, rather than a global, minimum. We illustrate the process with the next example.

Example 5.3. We will again find a quadratic model for the data considered in Example 5.1. This time, however, we will use the nonlinear optimizer, and will compute the model values in a single column.

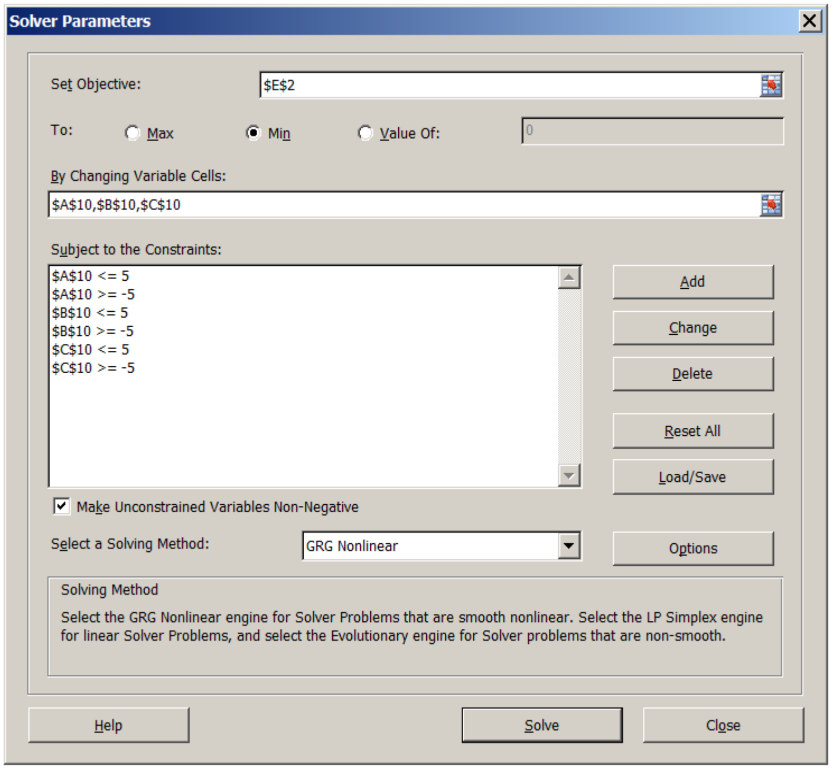


	A	B	C	D	E
1	y	x	yhat = ax^2+bx+c	$(\text{yhat - y})^2$	sum of sqr errors
2	1	0	4	9	3176
3	2	1	9	49	
4	4	2	18	196	
5	8	3	31	529	
6	16	4	48	1024	
7	32	5	69	1369	
8					
9	a	b	c		
10	2	3	4		

In the figure at right we have entered the *y* and *x* values in columns A and B, and values for coefficients *a*, *b*, and *c* below. The entries in column C are computed using the formula $ax^2 + bx + c$, where the coefficients are replaced with references to the cells A10, B10, and C10. The formula that appears in cell C2 is

=$A$10*B2^2+$B$10*B2+$C$10.

The dollar signs are included so that the references to the coefficient cells are absolute locations, and are not changed when the formula is copied to other cells. Thus we can copy cell C2 to cells C3 through C7 to populate those cells with correct values. Each value in column D is computed as the square of the difference between corresponding values of *y* and *y*-hat. Cell E2 shows the sum of the entries in column D. This is the sum of squared errors from our model, and is the value we wish to minimize. The initial coefficient values of 2, 3, and 4 were chosen arbitrarily. However, we will restrict the range over which each coefficient is varied to the interval from -5 to 5. In practice, the nonlinear regression app will perform better if we can limit the coefficient values in this way.



Solver Parameters

Set Objective: $E$2

To: ○ Max ● Min ○ Value Of: 0

By Changing Variable Cells: $A$10,$B$10,$C$10

Subject to the Constraints:

$A$10 <= 5
$A$10 >= -5
$B$10 <= 5
$B$10 >= -5
$C$10 <= 5
$C$10 >= -5

☑ Make Unconstrained Variables Non-Negative

Select a Solving Method: GRG Nonlinear

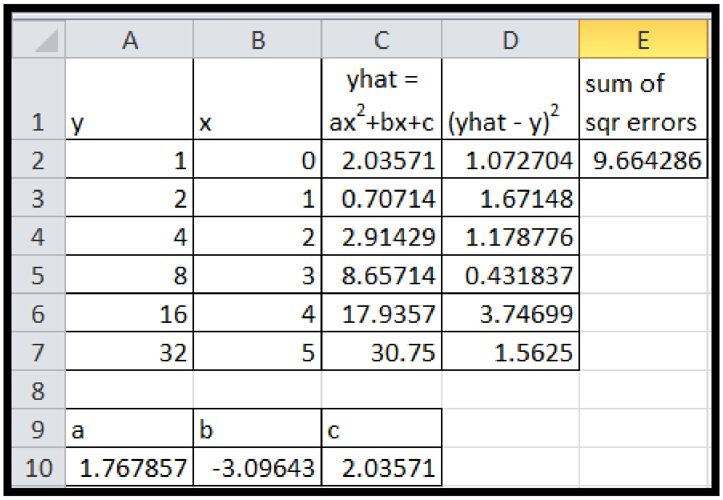
Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

The regression app is activated using the same procedure as shown in Screen B of Example 5.2, except that Solver is selected instead of Data Analysis. That creates a window where we specify the inputs to the regression app. See the image at right.

We have entered E2 as the target cell (*ie*. the value to be minimized), selected the Min option, and entered A10, B10, and C10 as the cells containing the parameters to be determined. Constraints have been added for each parameter, restricting its values to the interval from –5 to 5. The app is executed by clicking on Solve.

That writes the optimized values of the parameters into cells A10, B10, and C10, and the optimal value of the target is then shown in E2. At the same time, a window appears, asking whether we wish to accept the results, or restore the spreadsheet to its state prior to running the app. For the purpose of this example, we accepted the results of the app. The results appear in the image below.



	A	B	C	D	E
1	y	x	yhat = ax^2+bx+c	$(\text{yhat - y})^2$	sum of sqr errors
2	1	0	2.03571	1.072704	9.664286
3	2	1	0.70714	1.67148	
4	4	2	2.91429	1.178776	
5	8	3	8.65714	0.431837	
6	16	4	17.9357	3.74699	
7	32	5	30.75	1.5625	
8					
9	a	b	c		
10	1.767857	-3.09643	2.03571		

The results of the optimization are the same as those found earlier.

This approach to fitting a function to a set of data is more widely applicable than the linear least-squares method. It can be used with exponential and logistic functions, as well as for the shifted exponential function $y = ab^x + c$ that is needed for Chapter 5 of the text. And in all of these contexts it minimizes the sum of squared errors for the original, untransformed, data. At the same time, it works just as well with other error measures. For example, if the target cell is computed as the sum of the absolute errors, then the optimization will minimize that value.

On the other hand, in working with this method we have found that the results it produces depend on the initial values of the parameters entered. In particular, running the app twice sometimes produces better results than running it once. That is, if the "optimal" parameter values from the first run are used as the initial parameter values for the second run, a lower minimum is sometimes obtained, with modified "optimal" parameters. We have always found the optimized results to be better than what we have obtained with informal trial and error. But for serious optimization tasks, it would be prudent to consider the possible implications of results that are not necessarily best possible.

Section 6: Appendix B

Quadratic Growth Using Home Screen Iteration

Home screen iteration cannot be used as in Section 2 to compute terms of a quadratic growth sequence because each term depends on both its preceding term and its position in the sequence.

The method can be adapted for use with TI graphing calculators, as we discuss here. This topic is included for completeness, and for most students will probably not be useful. The discussion will make free use of the terminology and concepts of difference equations and sequences.

A quadratic growth sequence is one specified by a difference equation of the form

$$a_{n+1} = a_n + d + en,$$

where *d* and *e* are constants and the index variable *n* represents position number. Notice that the right-hand side of the difference equation depends on both a_n and *n*. Both of these values have to be updated to compute each successive term of the sequence. By combining both update steps into a single calculator command, it is possible to compute terms of the sequence using home screen iteration. This works on the TI-84 calculator as illustrated in the following example. Note that the colon character is used to separate multiple calculations on a single command line. Something similar is probably possible using other graphing calculators.

For the example, we use the difference equation

$$a_{n+1} = a_n + 7 + 2n,$$

with an initial term of $a_0 = 10$. Successive values of n and a_n will be stored in the calculator as variables N and A. We begin the iterative process by assigning N = 0 and A = 10. Then, in a single line, we compute the new value of A, increment N , and then retrieve the value of A. The process is shown in the table below. Note that there are three commands included in the third step, separated by colons. The final command retrieves the value of variable A, so that is what is displayed after the three commands are executed.

Step	Command Entered	Result
1	0 [STO▶] N [enter]	0
2	10 [STO▶] A [enter]	10
3	A + 7 + 2N [STO▶] A : N+1 [STO▶] N : A [enter]	17
4	[enter]	26
5	[enter]	37

The N, A, and colon are keyed in using the [ALPHA] key. For example, A is keyed in as [ALPHA][MATH]. This is not shown in the table for the sake of brevity.

Notice that this approach uses explicit assignment of values to variables, rather than the automatically assigned value accessed by [2nd ans]. For steps 4, 5, and beyond, each time the enter key is pressed, the computations from step 3 are repeated. Thus the displayed results show successive terms of our sequence.

On the TI-84 it is also possible to compute terms of a quadratic growth sequence using list operations. In this approach we incorporate the values of the index and the corresponding term of a sequence into a list of the form $L = \{n \;\; a_n\}$. As we compute successive values of the list, the first entry is always *n* and the second entry is always a_n . These are accessed on the calculator as L(1) and L(2).

For the sequence in the example above, we initially create the list L = {0   10}. Then we operate on this to create an updated list for the next term of the sequence by computing

{L(1) + 1   L(2) + 7 + 2L(1)}.

Here L(1) takes the place of N and L(2) takes the place of A. Thus, the command above computes an updated list in which N has been incremented by one, and a new A value has been computed according to the difference equation.

The list after each calculation can be accessed using the [2nd ans] key. Also, when entering a list from the keypad the terms must be separated with commas. The actual computation keyed into the calculator is therefore

{[2nd ans](1) + 1 ,   [2nd ans](2) + 7 + 2[2nd ans](1)} [enter].

This single update step can be incorporated into a home screen iteration process as shown in the table below. Note that the result displayed each time the enter key is pressed will be a two element list.

Step	Command Entered	Result
1	{0,10} [enter]	{0 10}
2	{[2nd ans](1) + 1, [2nd ans](2) + 7 + 2[2nd ans](1)} [enter]	{1 17}
3	[enter]	{2 26}
4	[enter]	{3 37}

On TI calculators lists are added component wise. This can be used to simplify the update command in step 2 above. Once we have entered the initial list, the following command can be keyed in:

+ { 1, 7 + 2 [2nd ans] (1) } [enter] .

This uses both list addition and an implied reference to the previous answer. When the initial + is keyed in, the calculator inserts [2nd ans] , just as if the user had keyed in

[2nd ans] + { 1, 7 + 2 [2nd ans] (1) } [enter] .

This mimics our original use of home screen iteration, but we are operating on lists rather than on individual numbers.

With the exception of quadratic growth, all of the growth sequences studied in the text can be easily generated using simple home screen iteration. In contrast, for quadratic growth sequences, both approaches described in this section have greater conceptual and syntactic complexity. For many students this may defeat the purpose of home screen iteration, namely, to provide a simple and memorable procedure for generating terms of a recursively defined sequence. On the other hand, for students who quickly master and remember calculator procedures, there may be some value in extending home screen iteration to quadratic growth sequences.