Classroom Tips and Techniques: Least-Squares Fits

Robert J. Lopez
Emeritus Professor of Mathematics and Maple Fellow
Maplesoft

Introduction

The least-squares fitting of functions to data can be done in Maple with eleven different commands from four different packages. The *CurveFitting* and *LinearAlgebra* packages each have a **LeastSquares** command, the former limited to the fitting of univariate linear models; the latter, applicable to univariate or multivariate linear models. The *Optimization* package has the **LSSolve** and **NLPSolve** commands, the former specifically designed to solve least-squares problems; the latter, capable of minimizing a nonlinear sum-of-squares.

These seven command from the *Statistics* package can return some measure of *regression analysis* (see Table 2): **Fit**, **LinearFit**, **PolynomialFit**, **ExponentialFit**, **LogarithmicFit**, **PowerFit**, and **NonlinearFit**. The **Fit** command passes problems to either **LinearFit** or **NonlinearFit**, as appropriate. The **NonlinearFit** command invokes *Optimization*'s **LSSolve** command, while the remaining commands (implementing a linearization where necessary) make use of *LinearAlgebra*'s **LeastSquares** command.

This month's article will explore each of these eleven tools, examine the spectrum of problems to which they apply, and give examples of their use.

Tools

Table 1 summarizes the eleven least-squares commands available in Maple.

Package	Command	Comments
CurveFitting	**LeastSquares**	• Fit a univariate linear model to data • Exact solutions supported • Fitting curve can have free parameters
LinearAlgebra	**LeastSquares**	• Fit a univariate or multivariate linear model to data • Obtain general or minimum-norm least-squares solution • Input can be set of linear equations

		• Exact solutions supported • Fitting curve can have free parameters
Optimization	**LSSolve**	• Obtain local minimum of $\frac{1}{2}\sum_{k=1}^{n}\left(g_k(\mathbf{u})-y_k\right)^2$ • Input: list of residuals $g_k(\mathbf{u})-y_k$ • Supports equality and/or inequality constraints, and bounds on variables • Both supported methods use differentiation • Numeric solutions only
	NLPSolve	• Obtain local minimum of $SS=\sum_{k=1}^{n}\left(f(\mathbf{x}_k)-y_k\right)^2$ • Input: SS • Supports equality and/or inequality constraints, and bounds on variables • Methods include nonlinear simplex (Nelder-Mead) for unconstrained multivariate objective functions • Numeric solutions only
Statistics	**Fit**	• Passes least-squares fit of a linear model to **LinearFit**, and of a nonlinear model to **NonlinearFit** • Accepts model only as an expression
	LinearFit	• Passes least-squares fit of a linear model to (numerical) *LinearAlgebra* • Model input as list or vector of component expressions or functions
	PolynomialFit	• Passes least-squares fit of a polynomial to (numerical) *LinearAlgebra* • Input: polynomial degree, data, and independent variable
	ExponentialFit	• Linearizes the fitting function $y=a\,e^{b\,x}$ to $\ln(y)=\ln(a)+b\,x$, and passes problem to (numerical) *LinearAlgebra* • Input: Data and independent variable
	LogarithmicFit	• Treats the fitting function $y=a+b\ln(x)$ as linear in $\ln(x)$ and passes problem to (numerical) *LinearAlgebra*

		• Input: Data and independent variable
	PowerFit	• Linearizes the fitting function $y = a x^b$ to $\ln(y) = \ln(a) + b\ln(x)$, and passes problem to (numerical) *LinearAlgebra* • Input: Data and independent variable
	NonlinearFit	• Passes the least-squares fit of a nonlinear model to the **LSSolve** command in *Optimization*, obtaining a local best-fit • Input: Model as expression or function, data, independent variable

Table 1 Maple commands for least-squares fitting

The **LeastSquares** command in the *CurveFitting* package fits a univariate linear model to data. The input data can be a list of points, or separate lists (or vectors) of values for the independent and dependent variables. The data points can be weighted, and the particular linear model can be provided as an expression linear in the model parameters. Computations are done in exact/symbolic form, so the fitting curve can contain free parameters that are not solve for. Both the Context Menu for a list of lists, and the Curve Fitting Assistant provide an interactive interface to this command.

The **LeastSquares** command in the *LinearAlgebra* package provides a number of additional functionalities for linear models: the model can be multivariate; the first argument can be a set of equations; for rank-deficient models both the general and minimum-norm solutions are available; and the user has control over the name of free parameters in a general solution. Like the *CurveFitting* version, this command can also work in exact/symbolic mode, so inputs and outputs can contain symbolic terms.

When applied to floating-point data, the **LeastSquares** command in *LinearAlgebra* will implement calculations based either on a QR decomposition, or a singular-values decomposition. Since the QR decomposition does not readily determine the rank of the decomposed matrix, least-squares fits based on this approach can fail for rank-deficient matrices. Because the default settings for automatically switching to the more robust singular-values approach can be thwarted by a given matrix, the safest policy appears to be setting the *method* option to SVD in all cases.

The **LSSolve** command in the *Optimization* package provides a local solution to both linear and nonlinear least-squares problems. The objective function (a sum of squares of deviations) shown in Table 1 is minimized, possibly subject to constraints (equality, inequality, bounds on variables). The input to the command could be a list $[\mathbf{y}, A]$ corresponding to the linear least-squares problem $A\mathbf{u} = \mathbf{y}$. Alternatively, the input to the command could be a list of residuals (deviations) of the form $g_k(\mathbf{u}) - y_k$. If the model is given by the function $y = F(\mathbf{x}; \mathbf{u})$, where **u** is a vector of parameters, then $g_k(\mathbf{u}) = F(\mathbf{x}_k; \mathbf{u})$. If the least-squares solution of the (inconsistent) equations $F_k(\mathbf{x}) = y_k, k = 1,\ldots, n$, is required, then $g_k(\mathbf{x}) = F_k(\mathbf{x})$, enabling **LSSolve** to accept what is essentially a list of equations.

The **NLPSolve** command in the *Optimization* package provides a local extreme for a multivariate function, whether linear or nonlinear. If this function is the sum-of-squares of deviations, then finding its minimum is equivalent to solving a least-squares problem. This command is included in Table 1 because it can invoke the Nelder-Mead method (nonlinear simplex in Maple), which is the only option in Maple that does not use differentiation to find local extrema of unconstrained multivariate functions. (Additional derivative-free options are available in Dr. Moiseev's *DirectSearch* package, the details of which were discussed here.)

All seven regression commands in the *Statistics* package work in floating-point arithmetic only. Output for each command can be one or a list of the items in Table 2, or a module containing all the relevant regression-analysis details shown in the table. The first eight items are available for the **NonlinearFit** command; all 16 are available for the other six regression commands. The help page for these options can be obtained by clicking here, or by executing the command **?Statistics,Regression,Solution**.

degreesoffreedom leastsquaresfunction parametervalues parametervector residuals residualmeansquare residualstandarddeviation residualsumofsquares
AtkinsonTstatistic condidenceintervals CookDstatistic externallystandardizedresiduals internallystandardizedresiduals leverages standarderrors variancecovariancematrix
Table 2 Regression analysis elements

Table 1 indicates that the **Fit** command is an interface to the **LinearFit** and **NonlinearFit** commands. The **LinearFit** and **PolynomialFit** commands invoke the numeric version of the **LeastSquares** command in *LinearAlgebra*, as do the **ExponentialFit**, **LogarithmicFit**, and **PowerFit** commands (after linearization). The **NonlinearFit** command invokes the **LSSolve** command in *Optimization*.

Universe of Discourse

Tables 3 and 4 summarize the universe of discourse for the least-squares options in Maple, Table 3 dealing with univariate models; and Table 4, multivariate models. The characteristics of this universe can be taken as linear/nonlinear, overdetermined/underdetermined/exactly determined, consistent/inconsistent, univariate/multivariate, provided the notion of "determined" is given a precise meaning. At first glance, a system can have more equations than unknown parameters, but if the equations are redundant, there may actually be fewer distinct equations in the system than unknowns. Such a system would be underdetermined, but by a simple count of equations, might be called overdetermined. We opt for the former meaning, namely, that the terms *underdetermined*, *overdetermined*, and *exactly determined* be applied only after all redundancies have been eliminated.

According to Table 3, whether a univariate model is linear or nonlinear, if there are more distinct equations than unknowns (i.e., if the system is truly overdetermined), then the system is necessarily inconsistent, and a least-squares solution is appropriate. So too for the underdetermined, inconsistent system - a least-squares solution is appropriate, and will contain free parameters. The underdetermined, consistent system will also have a general solution containing free parameters, but here, the least-squares technique need not be invoked.

Underdetermined or exactly determined consistent systems are essentially interpolation problems; inconsistent systems are the ones requiring least-squares techniques. Underdetermined linear models will have a parameter-dependent general solution, from which can be extracted a unique solution of minimum norm (L_2).

Univariate Models			
	Overdetermined	Underdetermined & Consistent	Underdetermined & Inconsistent
Linear	**LeastSquares** (*CurveFitting*) **LeastSquares** (*LinearAlgebra*) **LSSolve** (*Optimization*) **Fit** (*Statistics*) **LinearFit** (*Statistics*)	**solve** **LinearSolve** (*LinearAlgebra*) **LeastSquares** (*CurveFitting*) **LeastSquares** (*LinearAlgebra*) **LSSolve**[1] (*Optimization*)	**LeastSquares** (*CurveFitting*) **LeastSquares** (*LinearAlgebra*)
Nonlinear	**LSSolve** (*Optimization*) **NonlinearFit** (*Statistics*) **NLPSolve**[3] (Statistics)	**solve**[2] **LSSolve**[1] (*Optimization*)	**solve**[2] (normal equations) **LSSolve**[1] (*Optimization*)

Table 3 Problems and tools for fitting univariate models to data

(1) Local extrema, at best. (2) Can fail for intractable algebra. (3) Minimize sum-of-squares of deviations.

Table 4 categorizes multivariate linear models, those given as $A\mathbf{u} = \mathbf{v}$, according to the number of rows (*r*) and columns (*c*) in the matrix *A*. However, this classification is affected by the rank of A. Systems that are truly underdetermined have a parameter-dependent general solution, which, if projected onto the row space of *A*, becomes the minimum-norm solution. Full-rank matrices *A* with at least as many rows as columns have a trivial null space, and hence any associated linear system has a unique solution, even if it is in the least-squares sense.

Multivariate Linear Models			
A	Rank of A	$A\mathbf{u} = \mathbf{v}$	Appropriate Commands
r	Full rank	Consistent (necessarily)	**LinearSolve** and **LeastSquares** (*LinearAlgebra*)
	Deficient	Consistent	**LinearSolve** and **LeastSquares** (*LinearAlgebra*)
		Inconsistent	**LeastSquares** (*LinearAlgebra*)
$r = c$	Full rank	Consistent (necessarily)	**LinearSolve** (*LinearAlgebra*)
	Deficient	Consistent	**LinearSolve** and **LeastSquares** (*LinearAlgebra*)
		Inconsistent	**LeastSquares** (*LinearAlgebra*)
r	Full rank	Consistent	**LinearSolve** and **LeastSquares** (*LinearAlgebra*) **LSSolve** (*Optimization*)
		Inconsistent	**LinearSolve** and **LeastSquares** (*LinearAlgebra*) **LSSolve** (*Optimization*)
	Deficient	Consistent	**LinearSolve** and **LeastSquares** (*LinearAlgebra*)
		Inconsistent	**LeastSquares** (*LinearAlgebra*)
Table 4 Problems and tools for fitting multivariate linear models to data			

For either of Tables 3 or 4, the bifurcation induced by the exact/floating-point distinction arises only for invocations of the **LeastSquares** command in *LinearAlgebra*, and is dealt with only in the context of specific examples. Problems that turn out to be consistent and exactly determined are actually interpolation problems, and not least-squares problems.

The overdetermined nonlinear multivariate model can be solved with the **NonlinearFit** command in *Statistics*, and the **LSSolve** command in *Optimization*. Of course, the sum-of-squares of deviations can be directly minimized by, for example, the **NLPSolve** command in *Optimization*. Underdetermined nonlinear multivariate models pose a special challenge. None of the tools in *LinearAlgebra* apply, and the numeric tools in *Optimization* and *Statistics* provide only local solutions, so these tools will not return a general solution. If the algebra is tractable, it might be possible for the **solve** command to yield the general solution: for a consistent system, apply it directly to the equations; for an inconsistent system, to the normal equations.

Examples

This section contains some 21 examples illustrating the use of the Maple commands in Table 1. The organization of the examples is based on Table 3 and 4, and the remarks on nonlinear multivariate systems following Table 4.

Linear Univariate Models

Overdetermined Case

Example 1.

Fit $f(x) = \sum_{k=1}^{5} b_k \sin(k x)$ to ten data points along $y = x^2, 0 \le x \le 3$.

Solution

• Define f, the fitting function. • Context Menu: Assign Function	$f(x) = \sum_{k=1}^{5} b_k \sin(k x) \xrightarrow{\text{assign as function}} f$
Define a list of *x*-values (as floats). • Form a list of corresponding *y*-values: $y_k = x_k^2$	$X := [seq(3.\, k/10, k = 1..10)]:$ $Y := map(x \to x^2, X):$

Apply the **LeastSquares** command from the *CurveFitting* package.

$CurveFitting\text{:-}LeastSquares(X, Y, x, curve = f(x))$

$$3.76165223338178 \sin(x) - 3.19173112257587 \sin(2.x) + 2.07663466326689 \sin(3.x) - 1.67456333719396 \sin(4.x) + 1.36994912296017 \sin(5.x)$$

Apply the **LeastSquares** command from the *LinearAlgebra* package.
The arguments are a set of equations of the form $f(x_k) = y_k$, and a set of parameters (the b_k).

$LinearAlgebra{:-}LeastSquares\left(\{Equate(f{\sim}(X), Y)[\,]\}, \left\{seq\left(b_k, k = 1..5\right)\right\}\right)$

$$\{b_1 = 3.76165223338178, b_2 = -3.19173112257587, b_3 = 2.07663466326690, b_4 = -1.67456333719396, b_5 = 1.36994912296018\}$$

Notice that the output is not the fitting function, but a set of equations defining the parameters.

To the problem in the form $A\mathbf{u} = \mathbf{v}$, apply the **LeastSquares** command from *LinearAlgebra*.

$A, \mathbf{v} := LinearAlgebra{:-}GenerateMatrix(Equate(f{\sim}(X), Y), [seq(b[k], k = 1..5)])$:
$LinearAlgebra{:-}LeastSquares(A, \mathbf{v})$

$$\begin{bmatrix} 3.76165223338178 \\ -3.19173112257587 \\ 2.07663466326689 \\ -1.67456333719396 \\ 1.36994912296017 \end{bmatrix}$$

The output is now a vector of values for the parameters.

Apply the **LSSolve** command from the *Optimization* package. The input is the list $[\mathbf{v}, A]$.

$Optimization{:-}LSSolve([\mathbf{v}, A])$

$$\left[15.4472790754279341, \begin{bmatrix} 3.76165223338178 \\ -3.19173112257587 \\ 2.07663466326690 \\ -1.67456333719395 \\ 1.36994912296018 \end{bmatrix} \right]$$

The output is a list, the first member of which is half the sum of the squares of the deviations;

and the second of which is a vector of values for the parameters.

Apply the **LinearFit** command from the *Statistics* package.
The arguments are a list of basis functions for the linear model, the data, and the independent variable for the model.

$infolevel[Statistics] := 5 :$
$Statistics\text{:-}LinearFit([seq(\sin(k\,x), k = 1 ..5)], X, Y, x);$
$infolevel[Statistics] := 0 :$

```
In LinearFit (algebraic form)
SVD tolerance set to .10e-11
confidence level set to .95
final value of residual sum of squares: 30.8945581508559
```

$$3.76165223338178\sin(x) - 3.19173112257587\sin(2\,x) + 2.07663466326689\sin(3\,x) - 1.67456333719396\sin(4\,x) + 1.36994912296018\sin(5\,x)$$

By setting **infolevel** to 5, additional information about the calculation is printed. The fitting function is returned.

Apply the **Fit** command from *Statistics*; the first argument must now be the model function.

$infolevel[Statistics] := 5 :$
$Statistics\text{:-}Fit(f(x), X, Y, x);$
$infolevel[Statistics] := 0 :$

```
In Fit
In LinearFit (algebraic form)
SVD tolerance set to .10e-11
confidence level set to .95
final value of residual sum of squares: 30.8945581508559
```

$$3.76165223338178\sin(x) - 3.19173112257587\sin(2\,x) + 2.07663466326689\sin(3\,x) - 1.67456333719396\sin(4\,x) + 1.36994912296018\sin(5\,x)$$

Notice how **Fit** passed the problem off to **LinearFit**.

Apply the **Fit** command from *Statistics* so as to return *m*, a module whose exports are the entries of Table 2.

$m := Statistics\text{:-}Fit(f(x), X, Y, x, output = solutionmodule) :$

Access the exports of module m singly.

$m\text{:-}Results(residualsumofsquares) = 30.8945581508559$

Not recommended: Return all 16 exports in a (poorly formatted) list: $m\text{:-}Results(\,)$

The following device line-breaks the exports, making them easier to read.

$print\sim(m{:-}Results(\,))$

$$"residualmeansquare" = 6.17891163017118$$

$$"residualsumofsquares" = 30.8945581508559$$

$$"residualstandarddeviation" = 2.48574166601664$$

$$"degreesoffreedom" = 5$$

$$"parametervalues" = [b_1 = 3.76165223338178, b_2 = -3.19173112257587, b_3 = 2.07663466326689, b_4 = -1.67456333719396, b_5 = 1.36994912296018]$$

$$"parametervector" = \begin{bmatrix} 3.76165223338178 \\ -3.19173112257587 \\ 2.07663466326689 \\ -1.67456333719396 \\ 1.36994912296018 \end{bmatrix}$$

$$"leastsquaresfunction" = 3.76165223338178 \sin(x) - 3.19173112257587 \sin(2x) + 2.07663466326689 \sin(3x) - 1.67456333719396 \sin(4x) + 1.36994912296018 \sin(5x)$$

$$"standarderrors" = [1.08633928624710, 1.08640823324193, 1.08653361917527, 1.08673480293253, 1.08704868015567]$$

$$"confidenceintervals" = \begin{bmatrix} 0.969153050778061 .. 6.55415141598549 \\ -5.98440753749473 .. -0.399054707657017 \\ -0.716364063585975 .. 4.86963339011976 \\ -4.46807921875607 .. 1.11895254436815 \\ -1.42437359850927 .. 4.16427184442962 \end{bmatrix}$$

$$"residuals" = [-0.651900003011075, 0.126282436577216, 0.682282516720525, -0.276512477020958, -0.774750390140227, 0.533532883759732, 0.982373145633400, -1.16534900214506, -1.54200136432816, 4.93213034031380]$$

$$"leverages" = [0.550782059814024, 0.498868482572188, 0.553198268490694, 0.494886911249347, 0.559423632069402, 0.485338818017396, 0.575017276356971, 0.457820688982370, 0.637257282710881, 0.187406579736728]$$

"variancecovariancematrix" = [[1.18013304484385, -0.0000972356594965247, 0.000149804711438968, -0.000207447757466391, 0.000272460314586118], [-0.0000972356594965247, 1.18028284925586, -0.000304682378108956, 0.000422262245780442, -0.000555209310981035], [0.000149804711438968, -0.000304682378108956, 1.18055530559811, -0.000652439442233279, 0.000859492794283295], [-0.000207447757466391, 0.000422262245780442, -0.000652439442233279, 1.18099253190481, -0.00119864674122283], [0.000272460314586118, -0.000555209310981035, 0.000859492794283295, -0.00119864674122283, 1.18167483302819]]

$$"internallystandardizedresiduals" = \begin{bmatrix} -0.391287924113307 \\ 0.0717647377427275 \\ 0.410630143631415 \\ -0.156518050338247 \\ -0.469564452923559 \\ 0.299188222315640 \\ 0.606226409874577 \\ -0.636690758151175 \\ -1.02998147810306 \\ 2.20111078796061 \end{bmatrix}$$

$$"externallystandardizedresiduals" = \begin{bmatrix} -0.355463236131498 \\ 0.0642214164835666 \\ 0.373632885928074 \\ -0.140338221430155 \\ -0.429569616429641 \\ 0.270030134268448 \\ 0.563323182526688 \\ -0.594064378291893 \\ -1.03790786809869 \\ 11.1776416023537 \end{bmatrix}$$

$$"CookDstatistic" = \begin{bmatrix} 0.0375444355399597 \\ 0.00102538403057817 \\ 0.0417539545617528 \\ 0.00480038643762231 \\ 0.0559938114788459 \\ 0.0168827269821086 \\ 0.0994510374197327 \\ 0.0684604202547037 \\ 0.372739081781486 \\ 0.223473141178579 \end{bmatrix}$$

$$"AtkinsonTstatistic" = \begin{bmatrix} -0.393600858716140 \\ 0.0640762452574915 \\ 0.415745997372561 \\ -0.138910361312608 \\ -0.484053496059439 \\ 0.262224967280043 \\ 0.655258152585124 \\ -0.545895794170522 \\ -1.37567820556700 \\ 5.36791891496095 \end{bmatrix}$$

[ ]

Underdetermined Case

Consistent

Example 2.

Fit $f(x) = a x^2 + b x + c$ to the data points $(1, 5)$ and $(2, 3)$.

Solution

• Initialize Maple.	$restart;\ with(LinearAlgebra):$
• Define the fitting function: • Context Menu: Assign Function	$f(x) = a x^2 + b x + c \xrightarrow{\text{assign as function}} f$

This is an interpolation problem requiring the solution of two consistent equations in three unknowns.
It is not necessarily a least-squares problem.

• Write and solve two (consistent) equations in three unknowns.	$S := solve(\{f(1)=5, f(2)=3\})$ $$\left\{a = -\frac{7}{2} + \frac{1}{2}c, b = \frac{17}{2} - \frac{3}{2}c, c = c\right\}$$
• Obtain the general solution, a one-parameter family of interpolating functions.	$eval(f(x), S)$ $$\left(-\frac{7}{2} + \frac{1}{2}c\right)x^2 + \left(\frac{17}{2} - \frac{3}{2}c\right)x + c$$

Use the **LeastSquares** command from the *CurveFitting* package:

$CurveFitting\text{:-}LeastSquares(\langle 1, 2\rangle, \langle 5, 3\rangle, x, curve = f(x))$

$$7 + 2\_t_3 + (-2 - 3\_t_3)x + \_t_3 x^2$$

Although the calculation is passed off to the **LeastSquares** command in *LinearAlgebra*, which has provision for controlling the name of the free parameter, this control is lacking in the *CurveFitting* package.

Use the **LeastSquares** command from *LinearAlgebra*. The arguments here will be a set of equations and a set of parameters. Note the control over the free parameter.

$q := LinearAlgebra\text{:-}LeastSquares(\{f(1)=5, f(2)=3\}, \{a, b, c\}, free = s);$
$eval(f(x), q)$

$$\{a = s_1, b = -3s_1 - 2, c = 2s_1 + 7\}$$

$$s_1 x^2 + (-3s_1 - 2)x + 2s_1 + 7$$

The **LeastSquares** command in *LinearAlgebra* returns a set of equations defining the parameters, which then have to be transferred to the model to obtain the fitting function. The appearance of the free parameter is best explained via the matrix formulation of the problem.

Cast the problem in the form $A\mathbf{u} = \mathbf{v}$.

• Convert equations to matrix/vector form.	$A, \mathbf{v} := GenerateMatrix([f(1)=5, f(2)=3], [a, b, c])$

	$\begin{bmatrix} 1 & 1 & 1 \\ 4 & 2 & 1 \end{bmatrix}, \begin{bmatrix} 5 \\ 3 \end{bmatrix}$
• Obtain the general solution.	$\mathbf{V} := \mathit{LeastSquares}(A, \mathbf{v}, \mathit{free} = s)$ $\begin{bmatrix} s_1 \\ -3\,s_1 - 2 \\ 2\,s_1 + 7 \end{bmatrix}$
• Obtain the minimum-norm solution.	$\mathit{LeastSquares}(A, \mathbf{v}, \mathit{optimize} = \mathit{true})$ $\begin{bmatrix} -\frac{10}{7} \\ \frac{16}{7} \\ \frac{29}{7} \end{bmatrix}$
• Project **V** onto the row space of *A*.	$N := A^{\%T}:$ $P := N.(N^{\%T}.N)^{-1}.N^{\%T}:$ $P.\mathbf{V}$ $\begin{bmatrix} -\frac{10}{7} \\ \frac{16}{7} \\ \frac{29}{7} \end{bmatrix}$

The numeric solvers of the *Optimization* package, being local, will not necessarily find the minimum-norm solution, and might return any member of the general solution. The numeric solvers of the *Statistics* package reject underdetermined problems.

Inconsistent

Example 3.

Fit $f(x) = a\,x^2 + b\,x + c$ to the data point $(1, 5)$ and $(1, 3)$.

Solution

• Initialize Maple.	*restart*

• Define the fitting function: • Context Menu: Assign Function	$f(x) = a\,x^2 + b\,x + c \xrightarrow{\text{assign as function}} f$

Use the **LeastSquares** command from
the *CurveFitting* package:

Use the **LeastSquares** command from the *CurveFitting* package:

$CurveFitting:\text{-}LeastSquares(\langle 1, 1\rangle, \langle 5, 3\rangle, x, curve = f(x))$

$$4 - \_t_2 - \_t_3 + \_t_2\,x + \_t_3\,x^2$$

Use the **LeastSquares** command from *LinearAlgebra*. The arguments here will be a set of equations and a set of parameters.

$q := LinearAlgebra:\text{-}LeastSquares(\{f(1) = 5, f(1) = 3\}, \{a, b, c\}, free = s);$
$eval(f(x), q)$

$$\{a = 4 - s_2 - s_3, b = s_2, c = s_3\}$$

$$(4 - s_2 - s_3)\,x^2 + s_2\,x + s_3$$

The **LeastSquares** command in *LinearAlgebra* returns a set of equations defining the parameters, which then have to be transferred to the model to obtain the fitting function. The appearance of the free parameter is best explained via the matrix formulation of the problem.

Cast the problem in the form $A\mathbf{u} = \mathbf{v}$.

$A, \mathbf{v} := LinearAlgebra:\text{-}GenerateMatrix([f(1) = 5, f(1) = 3], [a, b, c])$

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 5 \\ 3 \end{bmatrix}$$

Obtain the general solution

$ParamVect := LinearAlgebra:\text{-}LeastSquares(A, \mathbf{v}, free = s);$
$eval(f(x), Equate(\langle a, b, c\rangle, ParamVect))$

$$\begin{bmatrix} 4 - s_2 - s_3 \\ s_2 \\ s_3 \end{bmatrix}$$

$$(4 - s_2 - s_3)\,x^2 + s_2\,x + s_3$$

Obtain the minimum-norm solution

$LinearAlgebra{:-}LeastSquares(A, \mathbf{v}, optimize = true)$

$$\begin{bmatrix} \frac{4}{3} \\ \frac{4}{3} \\ \frac{4}{3} \end{bmatrix}$$

The null space of A is spanned by the vectors $\{\langle -1, 1, 0\rangle, \langle -1, 0, 1\rangle\}$.

Nonlinear Univariate Models

Overdetermined Case

Logarithmic Model

Example 4.

Fit $y = a + b\ln(x)$ to the 45 data points shown in Figure 1.

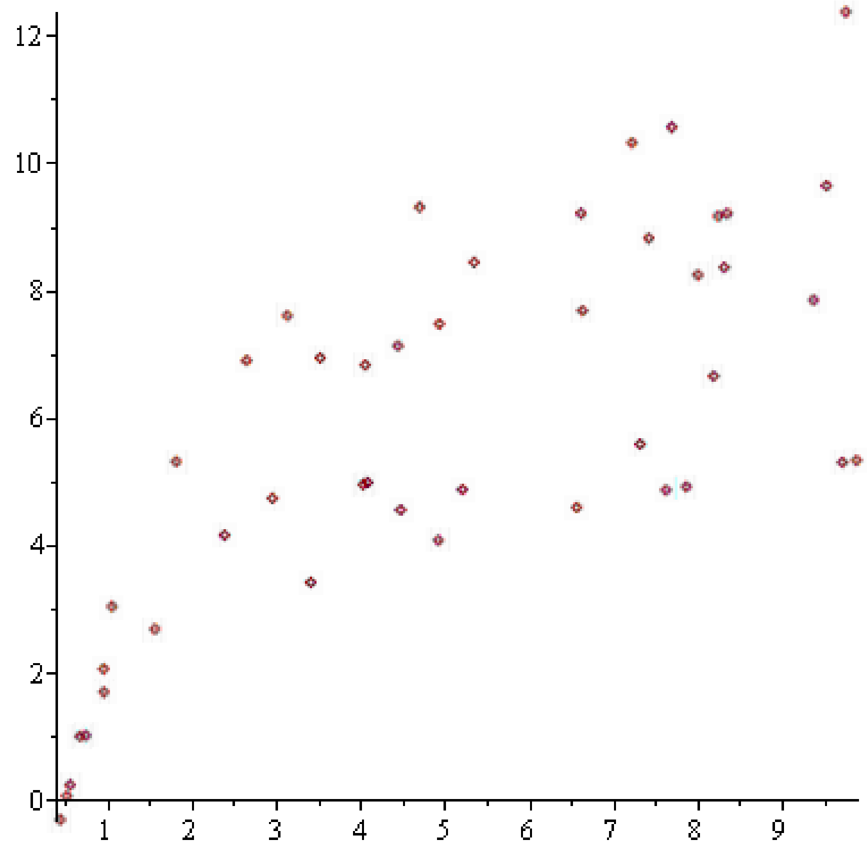

Figure 1 Data points to be fitted with $y = a + b\ln(x)$. (Data hidden behind table: lists X and Y of values of the independent and dependent variables, respectively.)

Solution

• Define *f*, the logarithmic model function.	$f := x \rightarrow a + b\ln(x)$:
• Form SS , the sum of squares of deviations:	$SS := \sum_{k=1}^{45} \left(f(X_k) - Y_k\right)^2$:
Apply the **LogarithmicFit** command from *Statistics*	
$Statistics$:-$LogarithmicFit(X, Y, x, output = [leastsquaresfunction, residualsumofsquares])$ $[2.18326708059312 + 2.73411741365235\ln(x), 138.348025811279]$	
Apply the **NonlinearFit** command from *Statistics*	

$Statistics{:-}NonlinearFit(f(x), X, Y, x, output = [leastsquaresfunction, residualsumofsquares])$ $[2.18326708944400 + 2.73411741330571 \ln(x), 138.3480258]$
Minimize SS via the *Optimization* package
$Optimization{:-}Minimize(SS)$ $[138.34802581127, [a = 2.18326708059312, b = 2.73411741365235]]$
Form and solve the normal equations, then evaluate the resulting SS
$Params := fsolve(\{diff(SS, a), diff(SS, b)\}, \{a, b\});$ $eval(SS, Params)$ $\{a = 2.183267101, b = 2.734117401\}$ 138.3480257

The solution obtained by the linearization in **LogarithmicFit** closely matches the solutions obtained by methods that do not linearize.

Power Model

Example 5.

Fit $y = a\,x^b$ to the 45 data points shown in Figure 2.

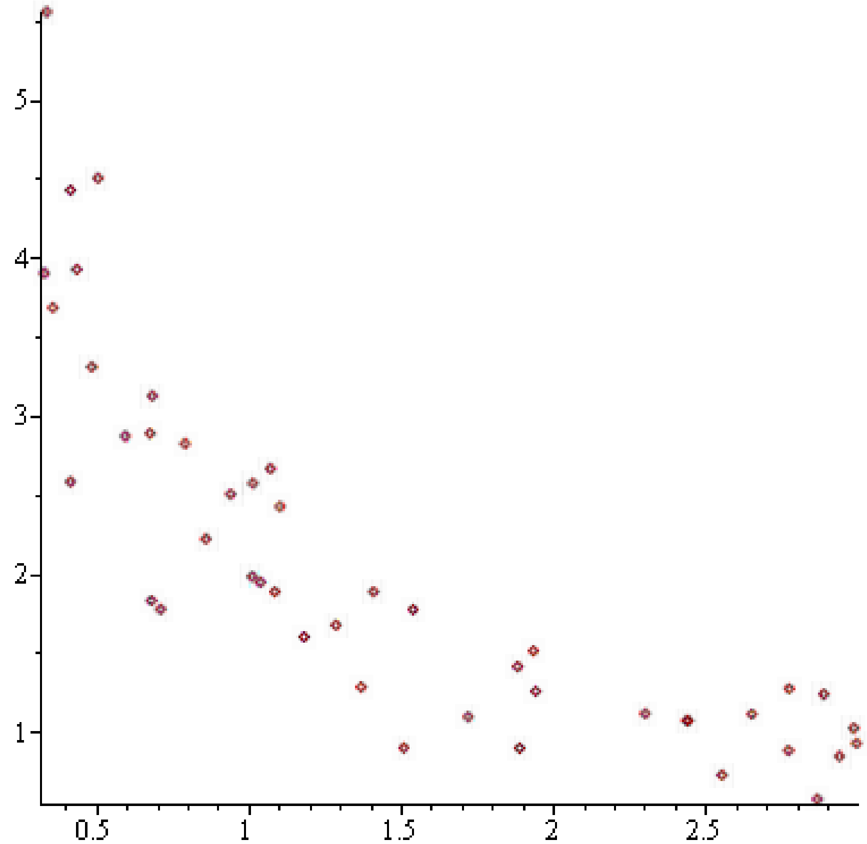

Figure 2 Data points to be fitted with $y = a\,x^b$. (Data hidden behind table: lists X and Y of values of the independent and dependent variables, respectively.)

Solution

• Context Menu: Assign Function	$g(x) = a\,x^b \xrightarrow{\text{assign as function}} g$
• Form SS, the sum of squares of deviations:	$SS := \sum_{k=1}^{45} \left(g(X_k) - Y_k\right)^2 :$
Apply the **PowerFit** command from *Statistics*	
$S := Statistics\text{:-}PowerFit(X, Y, x, output = [leastsquaresfunction, residualsumofsquares, parametervalues])$ $$\left[\frac{2.01022643676392}{x^{0.734986123143025}}, 2.13617170057370, \begin{bmatrix} 0.698247370832303 \\ -0.734986123143025 \end{bmatrix}\right]$$	

Evaluate SS for the linearized fit
$eval\left(SS, \left[a = e^{S[3][1]}, b = S[3][2]\right]\right) = 9.92674436684541$
Apply the **NonlinearFit** command from *Statistics*
$Statistics\text{:-}NonlinearFit(g(x), X, Y, x, output = [leastsquaresfunction, residualsumofsquares])$ $\left[\frac{2.06386565178199}{x^{0.711979687108144}}, 9.824989264\right]$
Minimize SS via the *Optimization* package
$Optimization\text{:-}Minimize(SS)$ $[9.82498926439639142, [a = 2.06386575402487, b = -0.711979619494700]]$
Form and solve the normal equations, then evaluate the resulting SS
$Params := fsolve(\{diff(SS, a), diff(SS, b)\}, \{a, b\});$ $eval(SS, Params)$ $\{a = 2.063865652, b = -0.7119796891\}$ 9.824989260

The parameters computed by the linearization in **PowerFit** differ slightly from those computed by the other methods which don't linearize. The sum of squares of residuals returned by **PowerFit** is for the *linearized* model, not the nonlinear model; when corrected for the linearization, it is slightly larger than the value for the nonlinear fits.

Exponential Model

Example 6.

Fit $y = a\,e^{b\,x}$ to the 45 data points shown in Figure 3.

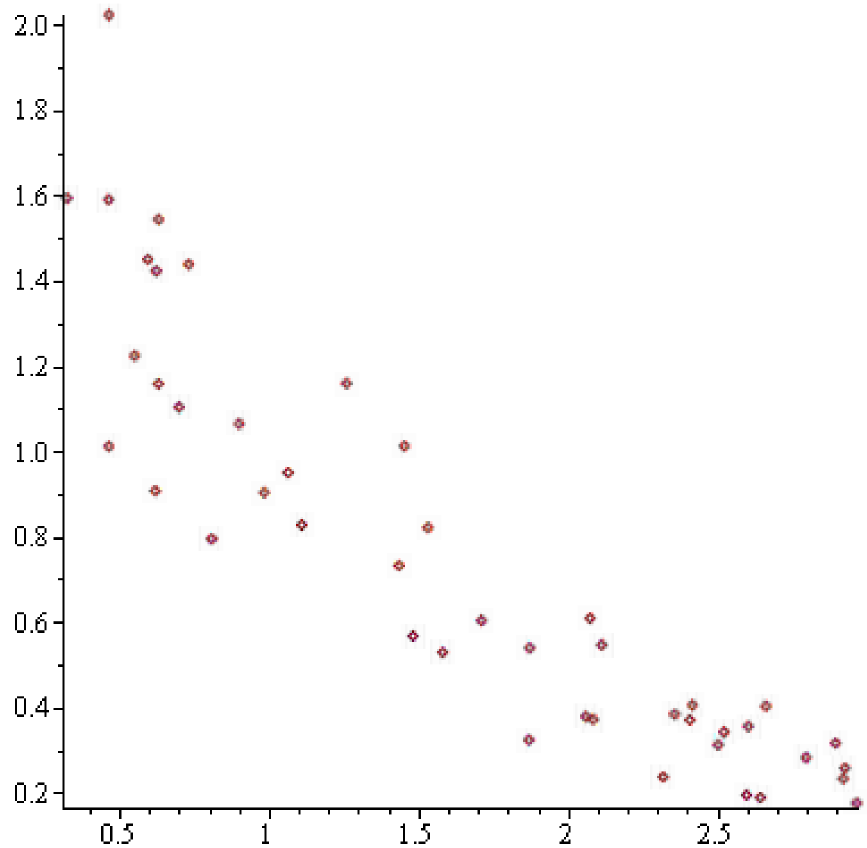

Figure 3 Data points to be fitted with $y = a\,e^{b\,x}$. (Data hidden behind table: lists *X* and *Y* of values of the independent and dependent variables, respectively.)

Solution

• Context Menu: Assign Function	$h(x) = a\,\mathrm{e}^{b\,x} \xrightarrow{\text{assign as function}} h$
• Form SS , the sum of squares of deviations:	$SS := \sum_{k=1}^{45} \left(h(X_k) - Y_k\right)^2 :$
Apply the **ExponentialFit** command from *Statistics*	
$S := Statistics{:-}ExponentialFit(X, Y, x, output = [leastsquaresfunction, residualsumofsquares, parametervalues])$	

$$\left[2.02149034952439\,\mathrm{e}^{-0.736048918381843\,x},\ 2.37018488848112,\ \left[\begin{array}{c}0.703835036169110\\-0.736048918381843\end{array}\right]\right]$$

Evaluate SS for the linearized fit

$eval\left(SS, \left[a = \mathrm{e}^{S[3][1]}, b = S[3][2]\right]\right)$ = 1.46492358156090

Apply the **NonlinearFit** command from *Statistics*

$Statistics{:-}NonlinearFit(h(x), X, Y, x, output = [leastsquaresfunction, residualsumofsquares])$

$$\left[2.03997327656541\,\mathrm{e}^{-0.723787355983985\,x},\ 1.450953524\right]$$

Minimize SS via the *Optimization* package

$Optimization{:-}Minimize(SS)$

$$[35.1003272108939157, [a = -52.5680106376265, b = -138.822756507913]]$$

Form and solve the normal equations, then evaluate the resulting SS

$Params := fsolve(\{diff(SS, a), diff(SS, b)\}, \{a, b\}, a = 1\,..3, b = -1\,..1);$
$eval(SS, Params)$

$$\{a = 2.039973278, b = -0.7237873536\}$$

$$1.450953523$$

The parameters computed by the linearization in **ExponentialFit** differ slightly from those computed by the other methods which don't linearize. The sum of squares of residuals returned by **ExponentialFit** is for the *linearized* model, not the nonlinear model; when corrected for the linearization, it is slightly larger than the value for the nonlinear fits.

Michaelis-Menten Model

Example 7.

Fit $v(s) = \dfrac{a\,s}{b+s}$, the Michaelis-Menten model, to the 46 data points shown in Figure 4.

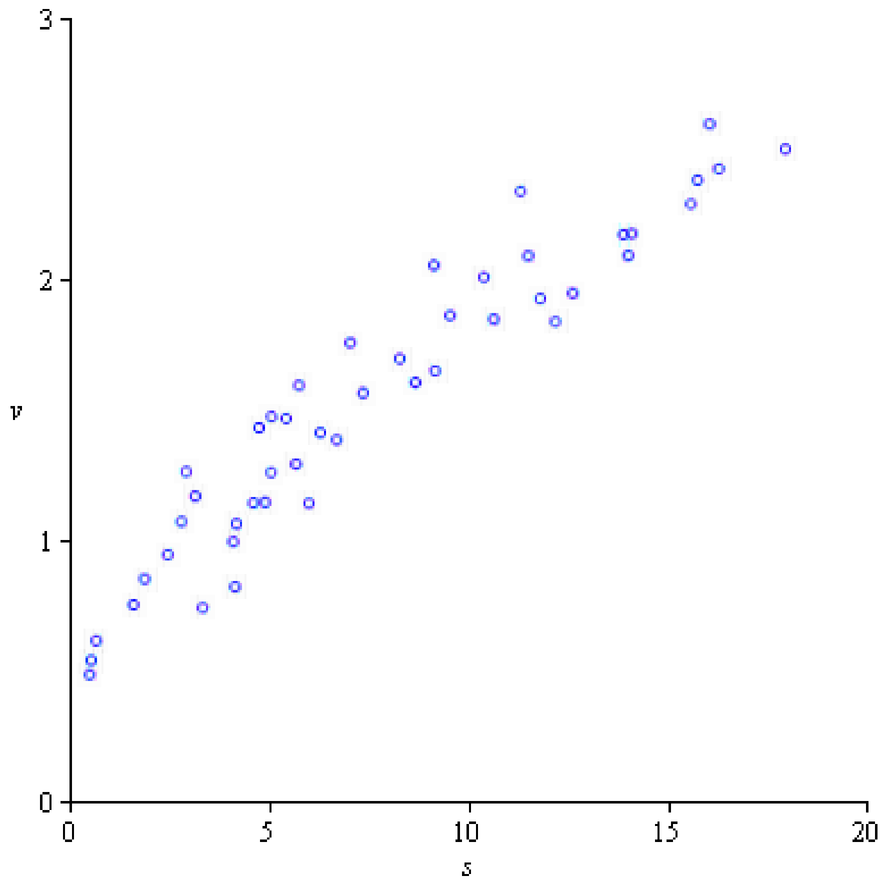

Figure 4 Data points to be fitted with $v(s) = \frac{a\,s}{b+s}$. (Data hidden behind table: lists S and V of values of the independent and dependent variables, respectively.)	

This example is a summary of one that appears in the ebook, *Advanced Engineering Mathematics with Maple*, an example that also appears in the Reporter article *Nonlinear Fit, Optimization, and the DirectSearch Package*. The fit is obtained with two different linearizations, and nonlinearly, the outcome being that the linearized solutions provide decidedly poor fits.

Solution

• Define $v(s)$: • Context Menu: Assign Function	$v(s) = \frac{a\,s}{b+s} \xrightarrow{\text{assign as function}} v$
Define SS , sum of squares of deviations.	$SS := \sum_{k=1}^{46} \left(v(S_k) - V_k\right)^2 :$

Linearization 1	Linearization 2
The linearization $\frac{1}{v} = \frac{1}{a} + \frac{b}{a}\,\frac{1}{s}$ requires the computation of the reciprocals $z_k = \frac{1}{s_k}$ and $w_k = \frac{1}{v_k}$.	The linearization $v = a - b\frac{v}{s}$ requires a list of ratios $\lambda_k = \frac{v_k}{s_k}$ with new independent variable $\lambda = \frac{v}{s}$.
$Z := map(x \rightarrow 1/x, S)$: $W := map(x \rightarrow 1/x, V)$:	$\Lambda := zip((x, y) \rightarrow y/x, S, V)$:
Obtain the least-squares regression line $w = A + B\,z$	
Here, $a = 1/A$ and $b = B/A$.	Here, $a = A$ and $b = -B$.
$L_1 := w = CurveFitting\text{:-}LeastSquares(Z, W, z)$ $w = 0.545885861428528 + 0.806123612341098\,z$	$L_2 := w = CurveFitting\text{:-}LeastSquares(\Lambda, V, \lambda)$ $w = 2.07808353050660 - 1.92146514300325\,\lambda$
$a_1 := 1/coeff(rhs(L_1), z, 0)$; $b_1 := a_1\,coeff(rhs(L_1), z, 1)$ 1.83188477786016 1.47672557452130	$a_2 := coeff(rhs(L_2), \lambda, 0)$; $b_2 := -coeff(rhs(L_2), \lambda, 1)$ 2.07808353050660 1.92146514300325
$v_1 := eval(v(s), [a = a_1, b = b_1])$ $\frac{1.83188477786016\,s}{1.47672557452130 + s}$	$v_2 := eval(v(s), [a = a_2, b = b_2])$ $\frac{2.07808353050660\,s}{1.92146514300325 + s}$

Nonlinear Fit
$Sol := Statistics\text{:-}NonlinearFit(v(s), S, V, s, output = [leastsquaresfunction, residualsumofsquares])$ $\left[\frac{3.47770837575172\,s}{8.22462695881459 + s}, 1.622578611\right]$
$Optimization\text{:-}LSSolve([seq(v(S_k) - V_k, k = 1..46)])$

$[0.811289305695100, [a = 3.47770839084770, b = 8.22462704378866]]$	
$fsolve\left(\left\{\frac{\partial}{\partial a} SS = 0, \frac{\partial}{\partial b} SS = 0\right\}, \{a, b\}, a = 0..10, b = 1..10\right)$ $\{a = 3.477708271, b = 8.224627001\}$	
• Sum of Squares for L_1 :	$eval(SS, [a = a_1, b = b_1])$ = 6.43191745197000
• Sum of Squares for L_2 :	$eval(SS, [a = a_2, b = b_2])$ = 4.52689912329230
• Sum of Squares for nonlinear fitting function:	$Sol[2]$ = 1.622578611

The **NonlinearFit** command from *Statistics* can return the fitting function, but the *Statistics* package's **LSSolve** command, whose input is a list of deviations (called residuals) returns the parameter values. In this example, it is even possible to form the normal equation and solve them numerically.

Figure 5 compares the graphs of the three fitting functions. Both from the graph and from the values of SS , it should be clear that linearizations do not necessarily provide the best fits to data.

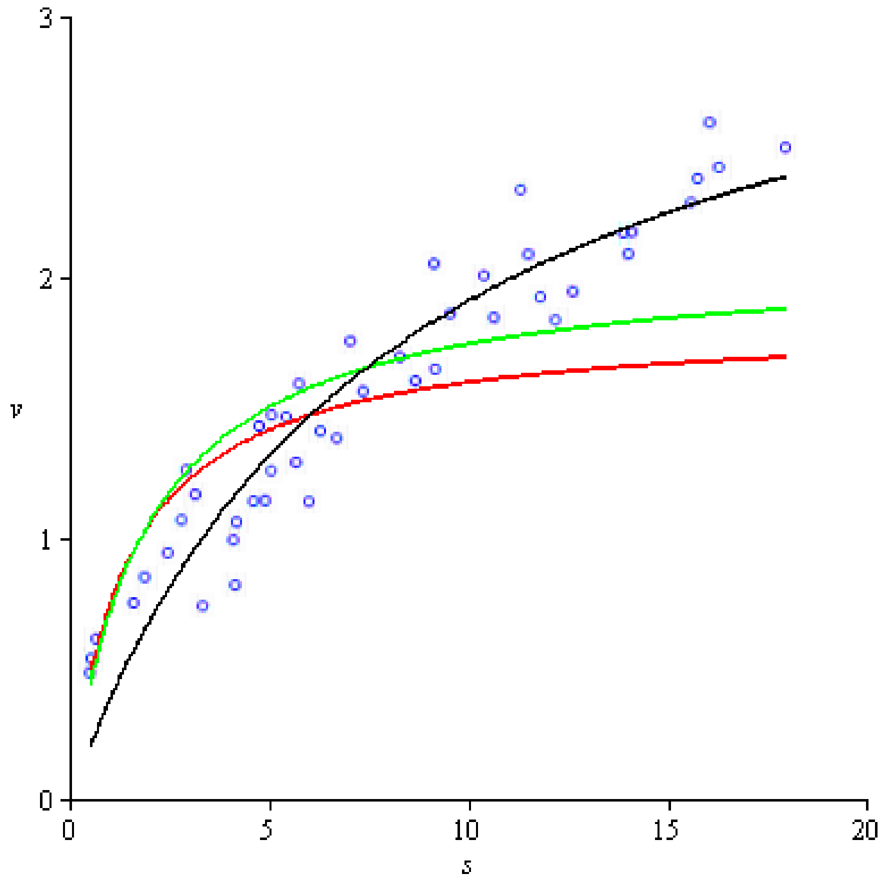

Figure 5 Nonlinear fit (black), L_1 (red), L_2 (green) superimposed on Figure 4

Underdetermined Case

Consistent

Example 8.

Fit $f(x) = e^{ax}(b + \ln(c + x))$ to the two points $(1, 3)$ and $(3, 1)$.

This problem is essentially an interpolation, with the expected result being a one-parameter family of curves all going through the two given points. If f were a linear function, such an outcome, and the means to achieve it, would be clear. For this particular f, it is possible to find this one-parameter family of solutions, but in general, it might not be possible to implement the requisite manipulations.

Exact Solution

• Restart Maple.	*restart*

• Control-drag the equation $f(x) = \ldots$ • Context Menu: Assign Function	$f(x) = e^{a\,x}\,(b + \ln(c + x)) \xrightarrow{\text{assign as function}} f$
Solve for *b* and *c* as functions of *a*. • Obtain the one-parameter family of solutions.	$q := solve(\{f(1) = 3, f(3, 1)\}, \{b, c\}) :$
$simplify(eval(f(x), q))$ $$-e^{a\,x}\left(\ln(2) + \ln\left(\frac{1}{-1 + e^{-3\,e^{-a}}}\right) - 3\,e^{-a} - \ln\left(\frac{3 - e^{-3\,e^{-a}} - x + x\,e^{-3\,e^{-a}}}{-1 + e^{-3\,e^{-a}}}\right)\right)$$	

The **LSSolve** command in the *Optimization* package requires at least as many residuals as parameters; otherwise, an error results. Hence, it really does not apply here.

Inconsistent

Example 9.

Fit $f(x) = e^{a\,x}\,(b + \ln(c + x))$ to the two points $(1, 3)$ and $(1, 5)$.

The equations $f(1) = 3$ and $f(1) = 5$ would necessarily be inconsistent, so this is not an interpolation, but a problem of fitting by least squares. A general solution consisting of a two-parameter family of curves is expected.

General Solution

• If necessary, define f.	$f := x \rightarrow e^{a\,x}\,(b + \ln(c + x)) :$
• Define *S* the sum of squares.	$S := \left((f(1) - 3)^2 + (f(1) - 5)^2\right) :$
• Obtain and solve the normal equations. • • The parameters *a* and *b* are free, and $c = c(a, b)$.	$q := solve\left(\left\{\frac{d}{da}\,S, \frac{d}{db}\,S, \frac{d}{dc}\,S\right\}\right)$ $\left\{a = a, b = b, c = e^{-\frac{e^{a}\,b - 4}{e^{a}}} - 1\right\}$
• Evaluate the sum of squares for this solution.	$expand(eval(S, q))$ assuming $real$ $= 2$

• Obtain the general solution to the underdetermined, inconsistent least-squares problem.	$eval(f(x), q) = e^{a\,x}\left(b + \ln\left(e^{-\frac{e^{a}\,b-4}{e^{a}}} - 1 + x\right)\right)$

Casual inspection shows that every member of the general solution passes through $(1, 4)$.

As in Example 8, the **LSSolve** command in the *Optimization* package requires at least as many residuals as parameters; otherwise, an error results. Hence, it really does not apply here.

Linear Multivariate Models

This section considers linear multivariate models that are cast in the form $A\mathbf{u} = \mathbf{v}$. As per Table 4, the examples are classified by the properties of the $r \times c$ matrix A, and the vector **v**.

• Initialize Maple.	$restart;\ with(LinearAlgebra):$

$r < c$

Full Rank

Define the full-rank matrix

$$Af := \begin{bmatrix} 9 & -5 & -72 & -85 \\ 45 & 47 & -79 & -19 \\ -10 & -54 & 75 & 57 \end{bmatrix}:$$

for which $Rank(Af) = 3$.

Consistent

Example 10.

Solve the least-squares problem $A\mathbf{u} = \mathbf{v}$, where $A = Af$, and $\mathbf{v} = \langle 1, 2, 3\rangle$.

Since there are fewer equations than variables, the system $A\mathbf{u} = \mathbf{v}$ cannot be inconsistent. Each row in A represents the left-hand side of a distinct equation, so no matter what appears on the right, the equations in the system must be consistent.

Solution

• Apply the **LeastSquares** command, designating *s* as the free variable base-name. • • The result is the general solution containing one free parameter. • • The solution is **u**, a vector of parameter values.	$\mathbf{Ugen} := LeastSquares(Af, \langle 1, 2, 3 \rangle, free = s)$ $\begin{bmatrix} \frac{23661}{76427} + \frac{197391}{76427} s_2 \\ s_2 \\ \frac{14011}{76427} + \frac{193352}{76427} s_2 \\ -\frac{10262}{76427} - \frac{147376}{76427} s_2 \end{bmatrix}$
• Obtain the minimum-norm least-squares solution. • • This is the projection of the general solution onto the row space of *Af*.	$LeastSquares(Af, \langle 1, 2, 3 \rangle, optimize)$ $\begin{bmatrix} \frac{613579001}{6927264966} \\ -\frac{1778379167}{20781794898} \\ -\frac{344638939}{10390897449} \\ \frac{319439654}{10390897449} \end{bmatrix}$
Obtain *P*, the matrix that projects onto the row space of *A*.	
• The columns of *N* are a basis for the row space.	$N := Matrix(map(Transpose, RowSpace(Af))) :$ $P := N.(N^{\%T}.N)^{-1}.N^{\%T} :$
• Project **Ugen** onto the row space. • • The result is the minimum-norm solution, a vector that lies in the row space of *Af*.	$P.\mathbf{Ugen} = \begin{bmatrix} \frac{613579001}{6927264966} \\ -\frac{1778379167}{20781794898} \\ -\frac{344638939}{10390897449} \\ \frac{319439654}{10390897449} \end{bmatrix}$

Rank-Deficient

Define the rank-deficient matrix

$$Ad := \begin{bmatrix} -18 & -9 & 63 & 72 \\ -11 & -8 & 37 & 33 \\ -3 & 1 & 12 & 23 \end{bmatrix}:$$

for which $Rank(Ad) = 2$.

Consistent

Example 11.

Solve the least-squares problem $A\mathbf{u} = \mathbf{v}$, where $A = Ad$, and $\mathbf{v} = \langle 9, 3, 4 \rangle$.

That the system is consistent can be seen from

$$ReducedRowEchelonForm(\langle Ad | \langle 9, 3, 4\rangle\rangle) = \begin{bmatrix} 1 & 0 & -\frac{19}{5} & -\frac{31}{5} & -1 \\ 0 & 1 & \frac{3}{5} & \frac{22}{5} & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

General solution:

$$LeastSquares(Ad, \langle 9, 3, 4\rangle, free = s) = \begin{bmatrix} -1 + \frac{19}{5} s_3 + \frac{31}{5} s_4 \\ 1 - \frac{3}{5} s_3 - \frac{22}{5} s_4 \\ s_3 \\ s_4 \end{bmatrix}$$

Minimum-norm solution:

$$LeastSquares(Ad, \langle 9, 3, 4\rangle, optimize) = \begin{bmatrix} \frac{221}{6065} \\ \frac{608}{6065} \\ -\frac{95}{1213} \\ \frac{261}{1213} \end{bmatrix}$$

Inconsistent

Example 12.

Solve the least-squares problem $A\mathbf{u} = \mathbf{v}$, where $A = Ad$, and $\mathbf{v} = \langle 1, 2, 3 \rangle$.

That the system is inconsistent can be seen from

$$ReducedRowEchelonForm(\langle Ad | \langle 1, 2, 3 \rangle \rangle) = \begin{bmatrix} 1 & 0 & -\frac{19}{5} & -\frac{31}{5} & 0 \\ 0 & 1 & \frac{3}{5} & \frac{22}{5} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

General solution:

$$LeastSquares(Ad, \langle 1, 2, 3 \rangle, free = s) = \begin{bmatrix} -\frac{344}{1055} + \frac{19}{5} s_3 + \frac{31}{5} s_4 \\ \frac{423}{1055} - \frac{3}{5} s_3 - \frac{22}{5} s_4 \\ s_3 \\ s_4 \end{bmatrix}$$

Minimum-norm solution:

$$LeastSquares(Ad, \langle 1, 2, 3 \rangle, optimize) = \begin{bmatrix} \frac{26881}{1279715} \\ \frac{63113}{1279715} \\ -\frac{12856}{255943} \\ \frac{22207}{255943} \end{bmatrix}$$

$r = c$

Full Rank

A full-rank matrix in a square system must necessarily be consistent, and therefore have a unique solution. There cannot be a least-squares problem in this case.

Rank-Deficient

Define the rank-deficient matrix

$$Bd := \begin{bmatrix} -6 & 42 & 30 & -24 \\ 22 & 6 & 2 & 32 \\ 30 & -50 & -38 & 64 \\ -14 & -2 & 0 & -21 \end{bmatrix}:$$

for which $Rank(Bd) \;=\; 2$.

Consistent

Example 13.

Solve the least-squares problem $A\mathbf{u} = \mathbf{v}$, where $A = Bd$, and $\mathbf{v} = \langle 3, 2, -2, -9/8 \rangle$.

That the system is consistent can be seen from

$$ReducedRowEchelonForm(\langle Bd|\langle 3, 2, -2, -9/8\rangle\rangle) \;=\; \begin{bmatrix} 1 & 0 & -\frac{1}{10} & \frac{31}{20} & \frac{11}{160} \\ 0 & 1 & \frac{7}{10} & -\frac{7}{20} & \frac{13}{160} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

General solution:

$$LeastSquares(Bd, \langle 3, 2, -2, -9/8\rangle, free = s) \;=\; \begin{bmatrix} \frac{3}{7} - \frac{31}{7} s_2 - 3 s_3 \\ s_2 \\ s_3 \\ \frac{20}{7} s_2 + 2 s_3 - \frac{13}{56} \end{bmatrix}$$

Minimum-norm solution:

$$LeastSquares(Bd, \langle 3, 2, -2, -9/8\rangle, optimize) = \begin{bmatrix} \frac{257}{8204} \\ \frac{73}{1172} \\ \frac{83}{2051} \\ \frac{439}{16408} \end{bmatrix} \xrightarrow{\text{at 10 digits}} \begin{bmatrix} 0.03132618235 \\ 0.06228668942 \\ 0.04046806436 \\ 0.02675524135 \end{bmatrix}$$

Because this system is essentially just an underdetermined one, a general solution is available with the **LinearSolve** command in the *LinearAlgebra* package.

$$LinearSolve(Bd, \langle 3, 2, -2, -9/8\rangle, free = s) = \begin{bmatrix} s_1 \\ -7 s_1 + \frac{9}{16} - \frac{21}{2} s_4 \\ 10 s_1 - \frac{11}{16} + \frac{31}{2} s_4 \\ s_4 \end{bmatrix}$$

Seek a least-squares solution numerically:

$$\mathbf{Un} := LeastSquares(evalf(Bd), evalf(\langle 3, 2, -2, -9/8\rangle)) = \begin{bmatrix} 0.940625000000000 \\ -0.115625000000000 \\ 0. \\ -0.562500000000000 \end{bmatrix}$$

That this is a member of the general solution can be seen by projecting it onto the row space of *Bd*.

• The columns of N are a basis for the row space.	$N := Matrix(map(Transpose, RowSpace(Bd)))$: $P := N.(N^{\%T}.N)^{-1}.N^{\%T}$:
• Project **Un** onto the row space. The result is the minimum-norm solution, a vector that lies in the row space of *Bd*.	$P.\mathbf{Un} = \begin{bmatrix} 0.447222343907994 \\ -0.306663521498669 \\ -0.483306687542500 \\ -0.194115879932644 \end{bmatrix}$

However, to obtain the minimum-norm solution numerically, specify that the calculation is to be based on the singular value decomposition, rather than on the default QR decomposition.

$$LeastSquares(evalf(Bd), evalf(\langle 3, 2, -2, -9/8 \rangle), method = SVD) = \begin{bmatrix} 0.0313261823500731 \\ 0.0622866894197952 \\ 0.0404680643588493 \\ 0.0267552413456850 \end{bmatrix}$$

Inconsistent

Example 14.

Solve the least-squares problem $A\mathbf{u} = \mathbf{v}$, where $A = Bd$, and $\mathbf{v} = \langle 1, 2, 3, 4 \rangle$.

That the system is inconsistent can be seen from

$$ReducedRowEchelonForm(\langle Bd | \langle 1, 2, 3, 4 \rangle \rangle) = \begin{bmatrix} 1 & 0 & -\frac{1}{10} & \frac{31}{20} & 0 \\ 0 & 1 & \frac{7}{10} & -\frac{7}{20} & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

General solution:

$$LeastSquares(Bd, \langle 1, 2, 3, 4 \rangle, free = s) = \begin{bmatrix} -\frac{404}{19341} - \frac{31}{7} s_2 - 3 s_3 \\ s_2 \\ s_3 \\ \frac{20}{7} s_2 + 2 s_3 + \frac{668}{19341} \end{bmatrix}$$

Minimum-norm solution:

$$LeastSquares(Bd, \langle 1, 2, 3, 4\rangle, optimize) = \begin{bmatrix} \frac{49996}{5666913} \\ -\frac{3356}{809559} \\ -\frac{7148}{1888971} \\ \frac{9524}{629657} \end{bmatrix} \xrightarrow{\text{at 10 digits}} \begin{bmatrix} 0.008822440013 \\ -0.004145466853 \\ -0.003784070798 \\ 0.01512569542 \end{bmatrix}$$

Seek a least-squares solution numerically:

$$LeastSquares(evalf(Bd), evalf(\langle 1, 2, 3, 4\rangle)) = \begin{bmatrix} 6.80441594719296\ 10^{15} \\ -6.76082606329234\ 10^{14} \\ -1.27011194448307\ 10^{15} \\ -4.47188847847824\ 10^{15} \end{bmatrix}$$

The default method, based on a QR decomposition, utterly fails because this decomposition does not have an efficient way to determine rank. For problems such as this (and Example 13), specify the method as the one based on the singular value decomposition.

$$LeastSquares(evalf(Bd), evalf(\langle 1, 2, 3, 4\rangle), method = SVD) = \begin{bmatrix} 0.00882244001275476 \\ -0.00414546685294091 \\ -0.00378407079833411 \\ 0.0151256954182992 \end{bmatrix}$$

Alternatively, use the **LinearFit** command from the *Statistics* package. Although this command is based on the **LeastSquares** from *LinearAlgebra*, there is an additional wrapper that attempts to deal with the issues raised by numeric calculations.

$Statistics{:-}LinearFit([x, y, z, w], evalf(Bd), evalf(\langle 1, 2, 3, 4\rangle), [x, y, z, w])$

```
Warning, model is not of full rank
```

$$0.00882244001275474 x - 0.00414546685294091 y - 0.00378407079833411 z + 0.0151256954182992 w$$

The deficiency in rank of the matrix Bd has been detected, and the calculation is based on the singular value decomposition. The control is via the ratio of the smallest to the largest singular values, which is the reciprocal of an estimated condition number for the input matrix. If this ratio is smaller than the default threshold 10^{-12}, the matrix is deemed to be ill-conditioned, and the least-squares calculation is based on the singular value decomposition. This default threshold is modified with the *svdtolerance* parameter.

$S := SingularValues(evalf(Bd))$:
$ReciprocalEstimatedConditionNumber = S[4]/S[1]$

$$ReciprocalEstimatedConditionNumber = 1.11657249978031\ 10^{-17}$$

$infolevel[Statistics] := 5$:
$Statistics\text{:-}LinearFit([x, y, z, w], evalf(Bd), evalf(\langle 1, 2, 3, 4\rangle), [x, y, z, w])$

```
In LinearFit (algebraic form)
SVD tolerance set to .10e-11
confidence level set to .95
Warning, model is not of full rank
rank =
final value of residual sum of squares: 26.3923271806008
```

$$0.00882244001275474\,x - 0.00414546685294091\,y - 0.00378407079833411\,z + 0.0151256954182992\,w$$

$Statistics\text{:-}LinearFit([x, y, z, w], evalf(Bd), evalf(\langle 1, 2, 3, 4\rangle), [x, y, z, w], svdtolerance = 1e{-}17)$;
$infolevel[Statistics] := 0$:

```
In LinearFit (algebraic form)
SVD tolerance set to .1e-16
confidence level set to .95
Warning, model is not of full rank
rank =
final value of residual sum of squares: 18.6547034958714
```

$$1.76749781558234\ 10^{14}\,x - 2.56320475960509\ 10^{13}\,y - 2.10788093061934\ 10^{13}\,z - 1.15392040315389\ 10^{14}\,w$$

The reciprocal of the estimated condition number is slightly larger than 10^{-17}, but that is well below the default threshold of 10^{-12}, so the first least-squares calculation is based on the singular value decomposition; in the second where the reciprocal of the estimated condition number is slightly larger than the threshold, the calculation is based on the default QR decomposition, and consequently fails.

$r > c$

Full Rank

Define the full-rank matrix

$$Cf := \begin{bmatrix} -74 & -60 & 35 \\ 13 & 51 & -54 \\ 32 & 20 & -17 \\ 48 & -46 & -25 \end{bmatrix}:$$

for which $Rank(Cf) = 3$.

Consistent

Example 15.

Solve the least-squares problem $A\mathbf{u} = \mathbf{v}$, where $A = Cf$, and $\mathbf{v} = \langle 1, 30, 1, 50091/13319 \rangle$.

That the system is consistent can be seen from

$$ReducedRowEchelonForm(\langle Cf | \langle 1, 30, 1, 50091/13319 \rangle \rangle) = \begin{bmatrix} 1 & 0 & 0 & -\frac{3966}{13319} \\ 0 & 1 & 0 & -\frac{454}{13319} \\ 0 & 0 & 1 & -\frac{8783}{13319} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Consequently, this is not a least-squares problem, but a properly determined system with a unique solution, obtainable for example, by **LinearSolve** in *LinearAlgebra*.

$$LinearSolve(Cf, \langle 1, 30, 1, 50091/13319 \rangle) = \begin{bmatrix} -\frac{3966}{13319} \\ -\frac{454}{13319} \\ -\frac{8783}{13319} \end{bmatrix}$$

Inconsistent

Inconsistent

Example 16.

Solve the least-squares problem $A\mathbf{u} = \mathbf{v}$, where $A = Cf$, and $\mathbf{v} = \langle 1, 2, 3, 4 \rangle$.

That the system is inconsistent can be seen from

$$\mathit{ReducedRowEchelonForm}(\langle Cf | \langle 1,2,3,4\rangle\rangle) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Because the matrix is full-rank, the null space is empty, and the least-squares solution is unique.

$$\mathbf{U} := \mathit{LeastSquares}(Cf, \langle 1,2,3,4\rangle) = \begin{bmatrix} -\frac{92649103}{54194075900} \\ -\frac{2590616723}{54194075900} \\ -\frac{1141568146}{13548518975} \end{bmatrix} \xrightarrow{\text{at 10 digits}} \begin{bmatrix} -0.001709579903 \\ -0.04780258137 \\ -0.08425778110 \end{bmatrix}$$

In general, the sum-of-squares of residuals is given by $\|A\mathbf{u} - \mathbf{v}\|_2^2$.

In general, the sum-of-squares of residuals is given by $\|A\mathbf{u} - \mathbf{v}\|_2^2$.

$$\| Cf.\mathbf{U} - \langle 1,2,3,4\rangle \|_2^2 = \frac{206391215809}{27097037950} \xrightarrow{\text{at 10 digits}} 7.616744538$$

Alternatively, the solution can also be found with the **LSSolve** command in the *Optimization* package.

$$S := \mathit{Optimization}{:-}\mathit{LSSolve}([\langle 1,2,3,4\rangle, Cf])$$

$$\left[3.80837226913578464, \begin{bmatrix} -0.00170957990262550 \\ -0.0478025813703376 \\ -0.0842577810981735 \end{bmatrix} \right]$$

The first member of the output list is half the sum-of-squares of the residuals; doubling this number gives $2\,S_1 = 7.616744538$.

Rank-Deficient

Define the rank-deficient matrix

$$Cd := \begin{bmatrix} -84 & -11 & 77 \\ -41 & 80 & 46 \\ 67 & 13 & -61 \\ 70 & 21 & -63 \end{bmatrix}:$$

for which $Rank(Cd) \;=\; 2$.

Consistent

Example 17.

Solve the least-squares problem $A\mathbf{u} = \mathbf{v}$, where $A = Cd$, and $\mathbf{v} = \langle 101, 101, -78, -77 \rangle$.

That the system is consistent can be seen from

$$ReducedRowEchelonForm(\langle Cd | \langle 101, 101, -78, -77 \rangle \rangle) \;=\; \begin{bmatrix} 1 & 0 & -\frac{66}{71} & -\frac{91}{71} \\ 0 & 1 & \frac{7}{71} & \frac{43}{71} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

General solution:

$$LeastSquares(Cd, \langle 101, 101, -78, -77 \rangle, free = s) \;=\; \begin{bmatrix} \frac{31}{7} - \frac{66}{7} s_2 \\ s_2 \\ -\frac{71}{7} s_2 + \frac{43}{7} \end{bmatrix}$$

Minimum-norm solution:

$$LeastSquares(Cd, \langle 101, 101, -78, -77 \rangle, optimize) \;=\; \begin{bmatrix} -\frac{3122}{4723} \\ \frac{5099}{9446} \\ \frac{6307}{9446} \end{bmatrix}$$

The general solution of an overdetermined but consistent system can also be found with the **LinearSolve** command from *LinearAlgebra*.

$$LinearSolve\left(Cd, \langle 101, 101, -78, -77\rangle, free = \sigma\right) = \begin{bmatrix} \sigma_1 \\ -\frac{7}{66}\sigma_1 + \frac{31}{66} \\ \frac{71}{66}\sigma_1 + \frac{91}{66} \end{bmatrix}$$

It is left to the reader to show that by appropriately redefining the free parameter in one general solution, the other will be obtained.

Inconsistent

Example 18.

Solve the least-squares problem $A\mathbf{u} = \mathbf{v}$, where $A = Cd$, and $\mathbf{v} = \langle 1, 2, 3, 4\rangle$.

That the system is inconsistent can be seen from

$$ReducedRowEchelonForm(\langle Cd|\langle 1, 2, 3, 4\rangle\rangle) = \begin{bmatrix} 1 & 0 & -\frac{66}{71} & 0 \\ 0 & 1 & \frac{7}{71} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

General solution:

$$LeastSquares(Cd, \langle 1, 2, 3, 4\rangle, free = s) = \begin{bmatrix} \frac{22579}{59829} - \frac{66}{7}s_2 \\ s_2 \\ -\frac{71}{7}s_2 + \frac{7723}{19943} \end{bmatrix}$$

Minimum-norm solution:

$$LeastSquares(Cd, \langle 1, 2, 3, 4\rangle, optimize) = \begin{bmatrix} \frac{454084}{40367481} \\ \frac{1045071}{26911654} \\ -\frac{178369}{26911654} \end{bmatrix} \xrightarrow{\text{at 10 digits}} \begin{bmatrix} 0.01124875738 \\ 0.03883339909 \\ -0.006627946391 \end{bmatrix}$$

Numeric linear algebra:

$$LeastSquares(evalf(Cd), evalf(\langle 1, 2, 3, 4\rangle)) = \begin{bmatrix} -1.45011918329997\ 10^{14} \\ 1.53800519440906\ 10^{13} \\ -1.55997669718633\ 10^{14} \end{bmatrix}$$

This calculation fails because the default QR-based method does not recognize that the rank-deficiency of the matrix. The more robust SVD-based method must be invoked.

$$LeastSquares(evalf(Cd), evalf(\langle 1, 2, 3, 4\rangle), method = SVD) = \begin{bmatrix} 0.0112487573846879 \\ 0.0388333990917095 \\ -0.00662794639080900 \end{bmatrix}$$

The **LSSolve** command from the *Optimization* package can find only a local solution, that is, one member of the general solution family.

$$S := Optimization\text{:-}LSSolve([evalf(\langle 1, 2, 3, 4\rangle), evalf(Cd)])$$

$$\left[7.06546156546156556, \begin{bmatrix} 0.0174099469874118 \\ 0.0381799395883903 \\ 0. \end{bmatrix}\right]$$

Project this solution onto the row space of Cd

• The columns of N are a basis for the row space; P projects onto the row space.	$N := Matrix(map(Transpose, RowSpace(Cd)))$: $P := N.(N^{\%T}.N)^{-1}.N^{\%T}$:
• The projection is the minimum-norm solution.	$P.S_2 = \begin{bmatrix} 0.0112487573846879 \\ 0.0388333990917095 \\ -0.00662794639080898 \end{bmatrix}$

In this example, the **LinearFit** command from the *Statistics* package finds the minimum-norm solution, but this outcome is dependent on the relative values of the default setting of the *svdtolerance* parameter, and the reciprocal of the approximate condition number computed for Cd .

$Statistics{:-}LinearFit([x, y, z], evalf(Cd), evalf(\langle 1, 2, 3, 4\rangle), [x, y, z])$

Warning, model is not of full rank

$$0.0112487573846879\,x + 0.0388333990917095\,y - 0.00662794639080898\,z$$

- The rank-deficiency of Cd had been detected, and the SVD-based method invoked. The minimum-norm solution is returned.
- The reciprocal of the approximate condition number of Cd :

$$SV := SingularValues(evalf(Cd)) = \begin{bmatrix} 183.938056091007 \\ 84.6096420123703 \\ 1.03565630293946\;10^{-14} \\ 0. \end{bmatrix}$$

$$ReciprocalConditionNumber = SV[4]/SV[1]$$

$$ReciprocalConditionNumber = 0.$$

- This value is smaller than 10^{-12} , the default svdtolerance parameter, so the more robust SVD-based method is invoked.

Nonlinear Multivariate Fit

Overdetermined Case

The first two columns of the matrix

$$M := \begin{bmatrix} 2.2467 & 5.2219 & 6.5622 \\ 2.0083 & 6.0656 & 6.3261 \\ 5.8386 & 1.1084 & 11.942 \\ 7.7071 & 5.9855 & 32.096 \\ 4.6193 & 4.6921 & 15.297 \end{bmatrix}:$$

are the abscissas and ordinates, respectively, of five data points $(x_k, y_k), k = 1,\ldots, 5$. The numbers in the third column are five corresponding observations $z_k = f(x_k, y_k)$.

Example 19.

Fit the function $f(x,y) = a\,x^b y^c$ to the data in *M*.

Since $Rank(M)$ = 3 , these data points generate a set of overdetermined nonlinear equations that are necessarily inconsistent. In contrast to the linear case, there is no functionality for obtaining a least-squares fit for nonlinear *equations*. The tools of *Statistics* and *Optimization* are the only ones that apply.

Solution

• Specify the nonlinear model: define the function f.	$f := (x,y) \rightarrow a\,x^b y^c$:
• Form SS , the sum of squares of residuals.	$SS := add\left(\left(f\left(M_{k,1}, M_{k,2}\right) - M_{k,3}\right)^2, k = 1..5\right)$:
Apply the **NonlinearFit** command from *Statistics*	
$Statistics\text{:-}NonlinearFit(f(x,y), M, [x,y], output = [leastsquaresfunction, residualsumofsquares])$ $[1.28883204859941\,x^{1.23745132612891}\,y^{0.383635147679040}, 0.09674158460]$	
Apply the **LSSolve** command from *Optimization*	
$S := Optimization\text{:-}LSSolve\left(\left[seq\left(f\left(M_{k,1}, M_{k,2}\right) - M_{k,3}, k = 1..5\right)\right]\right)$ $[0.0483707922990616, [a = 1.28883204859941, b = 1.23745132612891, c = 0.383635147679040]]$	
• Half the sum of squares is given by S_1 . • Double it to get the minimized SS .	$2 \cdot S_1$ = 0.09674158460
Apply the **Minimize** command from *Optimization*	
$Optimization\text{:-}Minimize(SS, iterationlimit = 2000)$ $[0.0967415845997482982, [a = 1.28883182832959, b = 1.23745142926965, c = 0.383635132900081]]$	

The results from all three approaches are fairly consistent.

Underdetermined Case

Consistent

The first two columns of the matrix

$$M := \begin{bmatrix} 2.2467 & 5.2219 & 6.5622 \\ 2.0083 & 6.0656 & 6.3261 \end{bmatrix}:$$

are the abscissas and ordinates, respectively, of two data points $(x_k, y_k), k = 1, 2$. The numbers in the third column are two corresponding observations $z_k = f(x_k, y_k)$.

Example 20.

Fit the function $f(x, y) = a\,x^b y^c$ to the data in *M*.

Since the data generate a set of two equations in three unknown parameters, this is an interpolation problem in which $Rank(M) = 2$ suggests there will be a general solution with one free parameter. In the nonlinear case, there is no theory by which a (unique) minimum-norm solution is extracted.

Solution

• Specify the nonlinear model by defining f.	$f := (x, y) \rightarrow a\,x^b y^c$:
• From the two given data points, form two equations in the three unknown parameters.	$q_1 := f(M_{1,1}, M_{1,2}) = M_{1,3}$; $q_2 := f(M_{2,1}, M_{2,2}) = M_{2,3}$ $a\,2.2467^b\,5.2219^c = 6.5622$ $a\,2.0083^b\,6.0656^c = 6.3261$
• Solve two equations for any two parameters in terms of the third. Here, a is the free parameter.	$S := solve(\{q_1, q_2\}, \{b, c\})$ $\{b = -5.878610145 \ln(0.1523879187\,a) + 5.390184700 \ln(0.1580752754\,a), c = -2.639756998 \ln(0.1580752754\,a) + 2.273944135 \ln(0.1523879187\,a)\}$

• The general solution is a fitting function dependent on one free parameter:
$'F'(x,y) = simplify(eval(f(x,y), S))$ assuming $a :: real, x > 0, y > 0$ $F(x,y)$ $= a\, x^{\frac{558197239}{500000000}}\, a^{-0.4884254450\ln(x) - 0.3658128630\ln(y)}\, y^{\frac{591487297}{1000000000}}$

Numeric solutions that seek to minimize a sum-of-squares of residuals return, at best, individual members of this family of solutions.

Inconsistent

Example 21.

Fit $f(x,y) = a\,x^b y^c$ to the two points $(2.2467, 5.2219, 6.5622)$ and $(2.2467, 5.2219, 6.3261)$.

The data determines two inconsistent equations in the three unknown parameters $\{a, b, c\}$. This is no longer an interpolation; it is a least-squares problem.

Solution

• Specify the nonlinear model by defining f.	$f := (x,y) \to a\,x^b y^c :$
• Define P and Q, the two data points.	$P := [2.2467, 5.2219, 6.5622] :$ $Q := [2.2467, 5.2219, 6.3261] :$
• Form SS, the sum of squares of residuals.	$SS := (f(P_1, P_2) - P_3)^2 + (f(Q_1, Q_2) - Q_3)^2 :$
• Form and solve the three normal equations.	
$q := solve\left(\left\{\frac{d}{da} SS = 0, \frac{d}{db} SS = 0, \frac{d}{dc} SS = 0\right\}\right)$ $\left\{a = a, b = b, c = 0.6050114354 \ln\left(\frac{6.444150000}{a}\right) - 0.4897340527\, b\right\}$	
• The general solution is a fitting function dependent on two free parameters:	
$'F'(x,y) = eval(f(x,y), q)$	

$$F(x, y) = a\, x^{b}\, y^{0.6050114354 \ln\left(\frac{6.444150000}{a}\right) - 0.4897340527\, b}$$

Numeric solutions that seek to minimize SS return, at best, individual members of this family of solutions.