Least Squares / Linear Regression

Sometimes we're given some data points of which we need to find a trend line or otherwise known as a regression line.
Let's say we're given a couple points, (2, 3) and (7, 4). The points really represent two equations (where we automaticall add a coefficient of 1):
$f(2) = \beta_0 + \beta_1 2 = 3$
$f(7) = \beta_0 + \beta_1 7 = 4$

$$\underset{A}{\begin{pmatrix} 1 & 2 \\ 1 & 7 \end{pmatrix}} \underset{\vec{x}}{\begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}} = \underset{\vec{b}}{\begin{pmatrix} 3 \\ 4 \end{pmatrix}}$$

If we take the matrix A, augment it with $\vec{b}$ and row-reduce the entire thing, you'll find we get:

$$\begin{pmatrix} 1 & 0 & \frac{13}{5} \\ 0 & 1 & \frac{1}{5} \end{pmatrix}$$

Now we know our beta parameters:
$\beta_0 = \frac{13}{5}$
$\beta_1 = \frac{1}{5}$

Ok, that's all cool and all, but sometimes we'll be given some data where after row-reducing, doesn't have a solution. The below is an example of a row-reduced matrix.

$$\begin{pmatrix} 1 & 0 & \frac{13}{5} \\ 0 & 1 & \frac{1}{5} \\ 0 & 0 & \frac{18}{5} \end{pmatrix}$$

You can see that the bottom row is inconsistent, $0 \neq \frac{18}{5}$. No line fits these points so we need to find a line that best fits these points, i.e. we need an approximate solution. This leads us to the Least Squares Solution, $A\hat{x} = \vec{b}^{\parallel}$.

The Long Way

Let's first approach this the long way.
Say you have the following points: (2,3),(7,4),(9,8).

Since $\vec{b}$ is not in the range(A), we need to project it onto the range(A). Therefore, $A\hat{x} = \vec{b}^{\parallel}$ has a solution even though $A\vec{x} = \vec{b}$ does not (which commonly happens when m > n). Why $\vec{b}^{\parallel}$? Because it's the closest vector in range(A) to $\vec{b}$ (The Best Approximation Theorem), which means whatever coefficients we get for $\vec{x}$ is really $\hat{x}$ because they're approximated.

The algo:

1. Find A. It's columns are x[1], x[2] below. b = {3,4,8}, the "observation" vector.
2. Gram-Schmidt-ify to get an orthogonal basis - v[1],v[2].
3. Find $\vec{b}^{\parallel}$, the orthogonal projection of $\vec{b}$ onto span of v[1], v[2].
4. Solve $A\hat{x}= \vec{b}^{\parallel}$.

$$\begin{pmatrix} 1 & 2 \\ 1 & 7 \\ 1 & 9 \end{pmatrix} * \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} = \begin{pmatrix} 3 \\ 4 \\ 8 \end{pmatrix}$$

```
Clear[b, x, proj, v, u]
b = {3, 4, 8};
x[1] = {1, 1, 1};
x[2] = {2, 7, 9};
p = 2;(*number of parameters*)
A = {x[1], x[2]}ᵀ;
Print["A=", A // MatrixForm]
Print["b=", b // MatrixForm]
```

$$\text{A=}\begin{pmatrix} 1 & 2 \\ 1 & 7 \\ 1 & 9 \end{pmatrix}$$

$$\text{b=}\begin{pmatrix} 3 \\ 4 \\ 8 \end{pmatrix}$$

Gram-Schmidt-ify to get an orthogonal basis.

```
For[i = 1, i ≤ p, i++,
 proj[i] = Sum[(x[i].v[j])/(v[j].v[j]) v[j], {j, 1, i-1}];
 v[i] = x[i] - proj[i];
 u[i] = v[i]/Sqrt[v[i].v[i]];
]
proj[1] = x[1] * 0;
T = Table[{x[i] // MatrixForm, proj[i] // Expand // MatrixForm,
    v[i] // Expand // MatrixForm, u[i] // Expand // MatrixForm}, {i, 1, p}];
T = Prepend[T, {"original vector",
    "orthogonal projection onto subspace spanned by prior vectors",
    "New Gram-Schmidt-ified vector", "Normalized"}];
```

$$\texttt{T = Prepend}\left[\texttt{T, }\left\{\texttt{"}x_i\texttt{", "}x_i^{\parallel}=\sum_{j=1}^{i-1}\left(\frac{x_i\cdot v_j}{v_j\cdot v_j}\right)v_j\texttt{", "}v_i=x_i-v_i^{\parallel}\texttt{", "}u_i=\frac{v_i}{\|v_i\|}\texttt{"}\right\}\right];$$

```
Print[T // TableForm]
```

x_i	$x_i^{\parallel}=\sum_{j=1}^{i-1}\left(\frac{x_i\cdot v_j}{v_j\cdot v_j}\right)v_j$	$v_i=x_i-v_i^{\parallel}$
original vector	orthogonal projection onto subspace spanned by prior vectors	New Gram
$\begin{pmatrix}1\\1\\1\end{pmatrix}$	$\begin{pmatrix}0\\0\\0\end{pmatrix}$	$\begin{pmatrix}1\\1\\1\end{pmatrix}$
$\begin{pmatrix}2\\7\\9\end{pmatrix}$	$\begin{pmatrix}6\\6\\6\end{pmatrix}$	$\begin{pmatrix}-4\\1\\3\end{pmatrix}$

Find $\vec{b}^{\parallel}$.

```
For[i = 1, i ≤ p, i++,
 c[i] = (b.v[i])/(v[i].v[i]);
 Print["c[", i, "]=", c[i]]
]
bp = Sum[c[i] * v[i], {i, 1, p}];
Print["b∥=", bp // MatrixForm]
```

c[1]=5

$\texttt{c[2]=}\frac{8}{13}$

$b^{\parallel}=\begin{pmatrix}\frac{33}{13}\\\frac{73}{13}\\\frac{89}{13}\end{pmatrix}$

Solve $A\hat{x}=\vec{b}^{\parallel}$.

```
B = Append[Aᵀ, bp]ᵀ;
Print["B=[A|b]=", B // MatrixForm];
R = RowReduce[B];
Print["R=rref[B]=", R // MatrixForm];
```

B=[A|b]= $\begin{pmatrix} 1 & 2 & \frac{33}{13} \\ 1 & 7 & \frac{73}{13} \\ 1 & 9 & \frac{89}{13} \end{pmatrix}$

R=rref[B]= $\begin{pmatrix} 1 & 0 & \frac{17}{13} \\ 0 & 1 & \frac{8}{13} \\ 0 & 0 & 0 \end{pmatrix}$

The Short Way

Now we can use a faster way to solve $A\hat{x} = \vec{b}^{\parallel}$, using Thm 6.58.13 (the Normal Equation). Remember that the solutions to $A\hat{x} = \vec{b}^{\parallel}$ are also the solutions to $A^T A\hat{x} = A^T \vec{b}$.

```
Aᵀ.A // MatrixForm
Inverse[Aᵀ.A] // MatrixForm
Inverse[Aᵀ.A].Aᵀ // MatrixForm
β = Inverse[Aᵀ.A].Aᵀ.b;
Print["Least squares solution = ", β // MatrixForm]
e = b - A.β;
Print["Residual vector = ", e // MatrixForm]
Print["Residual = ", Norm[e] // N]
```

$\begin{pmatrix} 3 & 18 \\ 18 & 134 \end{pmatrix}$

$\begin{pmatrix} \frac{67}{39} & -\frac{3}{13} \\ -\frac{3}{13} & \frac{1}{26} \end{pmatrix}$

$\begin{pmatrix} \frac{49}{39} & \frac{4}{39} & -\frac{14}{39} \\ -\frac{2}{13} & \frac{1}{26} & \frac{3}{26} \end{pmatrix}$

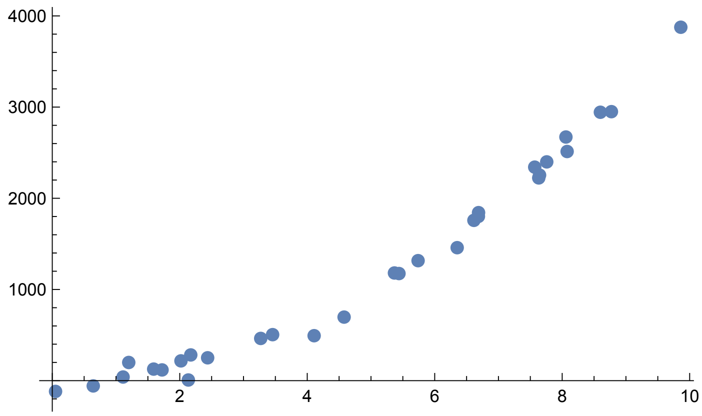
Least squares solution = $\begin{pmatrix} \frac{17}{13} \\ \frac{8}{13} \end{pmatrix}$

Residual vector = $\begin{pmatrix} \frac{6}{13} \\ -\frac{21}{13} \\ \frac{15}{13} \end{pmatrix}$

Residual = 2.0381

Now a new example with randomly generated data.

```
(*generate random data*)
n = 30;
M = 10;
a = RandomReal[{0, M}, n];
g = 2 - 5 t + 40 t^2;
b = g /. {t → a};
e = RandomVariate[NormalDistribution[0, 100], n];
b = b + e;
ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]]

(*Compute Correlation coefficient*)
x = a - Mean[a];
y = b - Mean[b];
r = x.y / (Norm[x] * Norm[y]);
Print["Correlation coefficient is r = ", N[r]]

(*Establish model*)
Clear[t];
f = {1, t};

(*Construct A*)
A = Table[f /. t → a[[i]], {i, 1, Length[a]}];
Print["Model function is f = ", f]
Print["A = ", A // MatrixForm];

(*Compute and plot least sqaures fit*)
β = Inverse[Aᵀ.A].Aᵀ.b;
Print["Least squares solution = ", β // MatrixForm]
e = b - A.β;
Print["Residual = ", Norm[e] // N]
h = f.β;
Print["Best fit model function = ", h]
Show[ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]],
 Plot[h, {t, Min[a] - 1, Max[a] + 1}]]

(*Establish model*)
Clear[t];
f = {1, t, t^2};

(*Construct A*)
A = Table[f /. t → a[[i]], {i, 1, Length[a]}];
Print["Model function is f = ", f]
Print["A = ", A // MatrixForm];

(*Compute and plot least sqaures fit*)
β = Inverse[Aᵀ.A].Aᵀ.b;
Print["Least squares solution = ", β // MatrixForm]
e = b - A.β;
Print["Residual = ", Norm[e] // N]
h = f.β;
Print["Best fit model function = ", h]
Show[ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]],
 Plot[h, {t, Min[a] - 1, Max[a] + 1}]]
```



Correlation coefficient is r = 0.970527

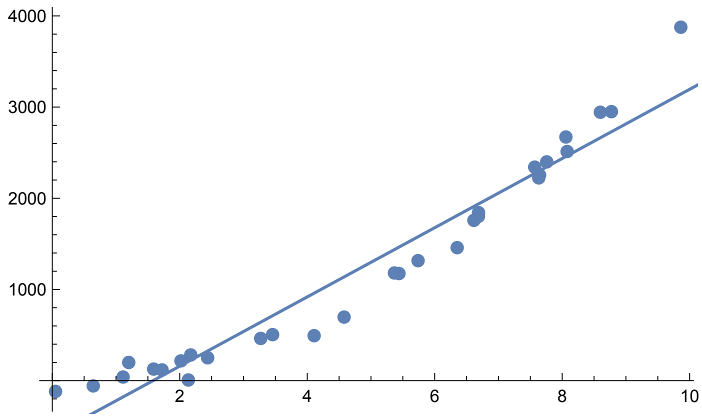
Model function is f = {1, t}

$$A = \begin{pmatrix} 1 & 1.1996 \\ 1 & 7.7576 \\ 1 & 2.13487 \\ 1 & 8.05847 \\ 1 & 7.63219 \\ 1 & 0.0513538 \\ 1 & 5.36849 \\ 1 & 8.77227 \\ 1 & 2.01908 \\ 1 & 0.64289 \\ 1 & 6.35248 \\ 1 & 7.569 \\ 1 & 2.43752 \\ 1 & 4.10831 \\ 1 & 6.615 \\ 1 & 1.59084 \\ 1 & 2.17268 \\ 1 & 1.11084 \\ 1 & 6.68779 \\ 1 & 5.73913 \\ 1 & 8.07761 \\ 1 & 3.27042 \\ 1 & 3.45693 \\ 1 & 7.64491 \\ 1 & 4.57847 \\ 1 & 8.599 \\ 1 & 1.72204 \\ 1 & 9.86106 \\ 1 & 6.68554 \\ 1 & 5.43943 \end{pmatrix}$$

Least squares solution = $\begin{pmatrix} -599.841 \\ 379.512 \end{pmatrix}$

Residual = 1465.83

Best fit model function = -599.841 + 379.512 t



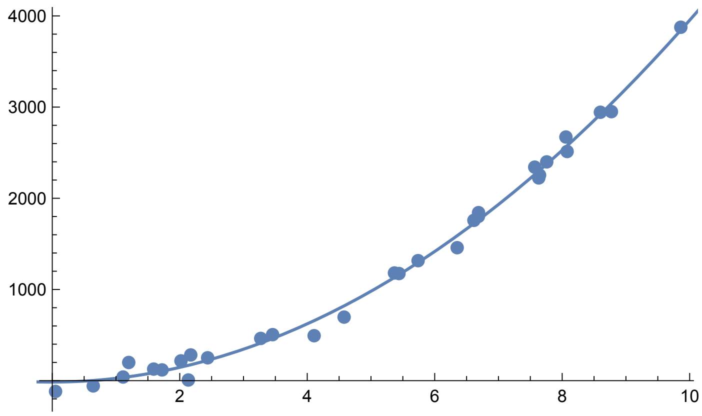
Model function is f = {1, t, t^2}

$$A = \begin{pmatrix} 1 & 1.1996 & 1.43904 \\ 1 & 7.7576 & 60.1803 \\ 1 & 2.13487 & 4.55769 \\ 1 & 8.05847 & 64.9389 \\ 1 & 7.63219 & 58.2503 \\ 1 & 0.0513538 & 0.00263722 \\ 1 & 5.36849 & 28.8207 \\ 1 & 8.77227 & 76.9528 \\ 1 & 2.01908 & 4.07669 \\ 1 & 0.64289 & 0.413308 \\ 1 & 6.35248 & 40.354 \\ 1 & 7.569 & 57.2897 \\ 1 & 2.43752 & 5.94152 \\ 1 & 4.10831 & 16.8782 \\ 1 & 6.615 & 43.7582 \\ 1 & 1.59084 & 2.53076 \\ 1 & 2.17268 & 4.72054 \\ 1 & 1.11084 & 1.23396 \\ 1 & 6.68779 & 44.7266 \\ 1 & 5.73913 & 32.9376 \\ 1 & 8.07761 & 65.2478 \\ 1 & 3.27042 & 10.6956 \\ 1 & 3.45693 & 11.9504 \\ 1 & 7.64491 & 58.4446 \\ 1 & 4.57847 & 20.9624 \\ 1 & 8.599 & 73.9428 \\ 1 & 1.72204 & 2.96543 \\ 1 & 9.86106 & 97.2405 \\ 1 & 6.68554 & 44.6965 \\ 1 & 5.43943 & 29.5874 \end{pmatrix}$$

Least squares solution = $\begin{pmatrix} -15.4032 \\ 1.37295 \\ 39.5426 \end{pmatrix}$

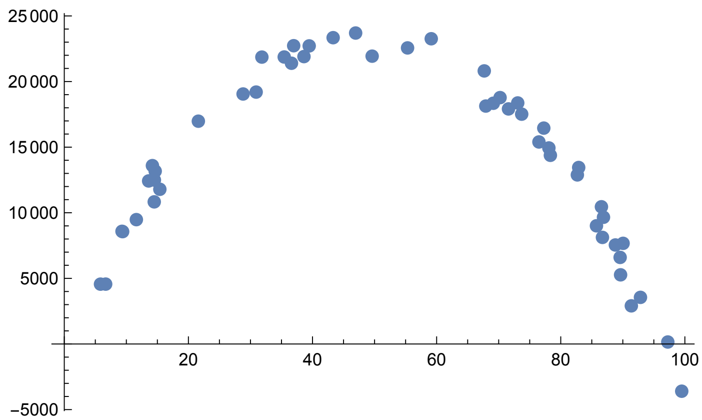
Residual = 447.123

Best fit model function = $-15.4032 + 1.37295\, t + 39.5426\, t^2$


4000
3000
2000
1000
2
4
6
8
10

More Examples

```
(*generate random data*)
n = 50;
M = 100;
a = RandomReal[{0, M}, n];
g = -9.8 t^2 + 950 t;
b = g /. {t → a};
e = RandomVariate[NormalDistribution[0, 1000], n];
b = b + e;
ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]]

(*Compute Correlation coefficient*)
x = a - Mean[a];
y = b - Mean[b];
r = x.y / (Norm[x] * Norm[y]);
Print["Correlation coefficient is r = ", N[r]]

(*Establish model*)
Clear[t];
f = {1, t};

(*Construct A*)
A = Table[f /. t → a[[i]], {i, 1, Length[a]}];
Print["Model function is f = ", f]
(*Print["A = ",A//MatrixForm];*)

(*Compute and plot least sqaures fit*)
β = Inverse[Aᵀ.A].Aᵀ.b;
Print["Least squares solution = ", β // MatrixForm]
e = b - A.β;
Print["Residual = ", Norm[e] // N]
h = f.β;
Print["Best fit model function = ", h]
Show[ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]],
 Plot[h, {t, Min[a] - 1, Max[a] + 1}]]

(*Establish model*)
Clear[t];
f = {1, t, t^2};

(*Construct A*)
A = Table[f /. t → a[[i]], {i, 1, Length[a]}];
Print["Model function is f = ", f]
(*Print["A = ",A//MatrixForm];*)

(*Compute and plot least sqaures fit*)
β = Inverse[Aᵀ.A].Aᵀ.b;
Print["Least squares solution = ", β // MatrixForm]
e = b - A.β;
Print["Residual = ", Norm[e] // N]
h = f.β;
Print["Best fit model function = ", h]
Show[ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]],
 Plot[h, {t, Min[a] - 1, Max[a] + 1}]]
```



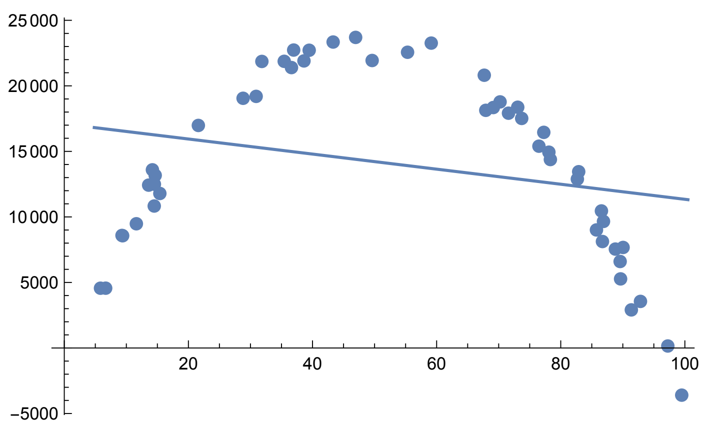
Correlation coefficient is r = -0.254976

Model function is f = {1, t}

Least squares solution = $\begin{pmatrix} 17094.3 \\ -57.5037 \end{pmatrix}$

Residual = 46 728.3

Best fit model function = 17 094.3 - 57.5037 t

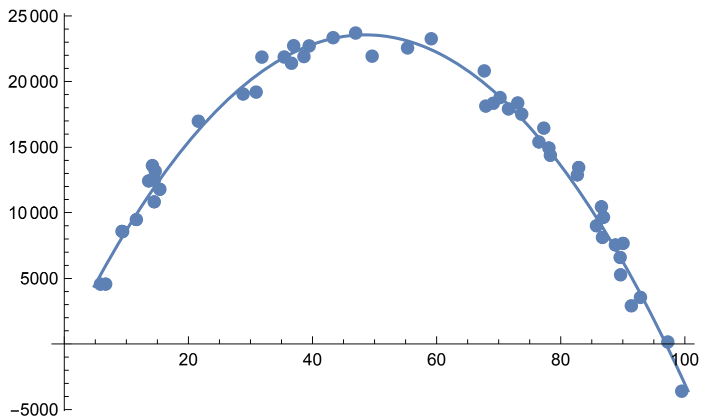


Model function is f = $\{1, t, t^2\}$

Least squares solution = $\begin{pmatrix} -78.6681 \\ 974.294 \\ -10.0404 \end{pmatrix}$

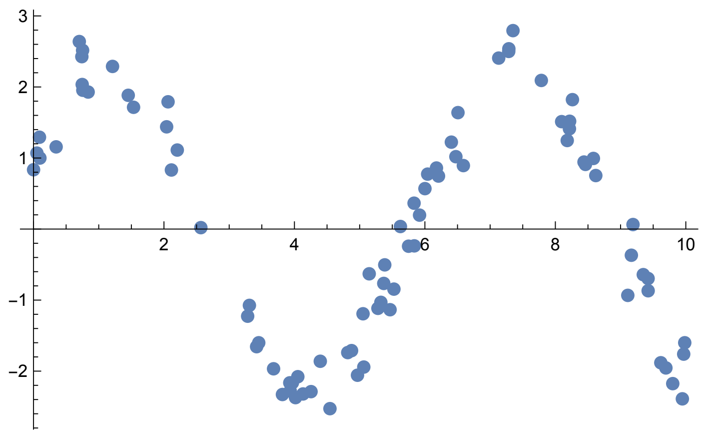
Residual = 6682.58

Best fit model function = $-78.6681 + 974.294\, t - 10.0404\, t^2$



Yet another example with randomly generated data.

```
(*generate random data*)
n = 85;
M = 10;
a = RandomReal[{0, M}, n];
g = Cos[t] + 2 Sin[t];
b = g /. {t → a};
e = RandomVariate[NormalDistribution[0, .3], n];
b = b + e;
ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]]

(*Compute Correlation coefficient*)
x = a - Mean[a];
y = b - Mean[b];
           x.y
r = ─────────────────── ;
    Norm[x] * Norm[y]
Print["Correlation coefficient is r = ", N[r]]

(*Establish model*)
Clear[t];
f = {1, t};

(*Construct A*)
A = Table[f /. t → a[[i]], {i, 1, Length[a]}];
Print["Model function is f = ", f]
(*Print["A = ",A//MatrixForm];*)

(*Compute and plot least sqaures fit*)
β = Inverse[Aᵀ.A].Aᵀ.b;
Print["Least squares solution = ", β // MatrixForm]
e = b - A.β;
Print["Residual = ", Norm[e] // N]
h = f.β;
Print["Best fit model function = ", h]
Show[ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]],
 Plot[h, {t, Min[a] - 1, Max[a] + 1}]]

(*Establish model*)
Clear[t];
f = {1, Sin[t], Cos[t]};

(*Construct A*)
A = Table[f /. t → a[[i]], {i, 1, Length[a]}];
Print["Model function is f = ", f]
(*Print["A = ",A//MatrixForm];*)

(*Compute and plot least sqaures fit*)
β = Inverse[Aᵀ.A].Aᵀ.b;
Print["Least squares solution = ", β // MatrixForm]
e = b - A.β;
Print["Residual = ", Norm[e] // N]
h = f.β;
Print["Best fit model function = ", h]
Show[ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]],
 Plot[h, {t, Min[a] - 1, Max[a] + 1}]]
```



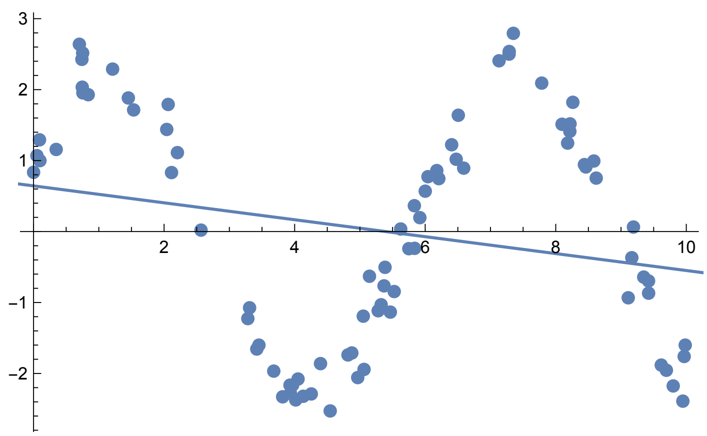
Correlation coefficient is r = -0.216133

Model function is f = {1, t}

Least squares solution = $\begin{pmatrix} 0.643567 \\ -0.119277 \end{pmatrix}$

Residual = 14.4434

Best fit model function = 0.643567 - 0.119277 t

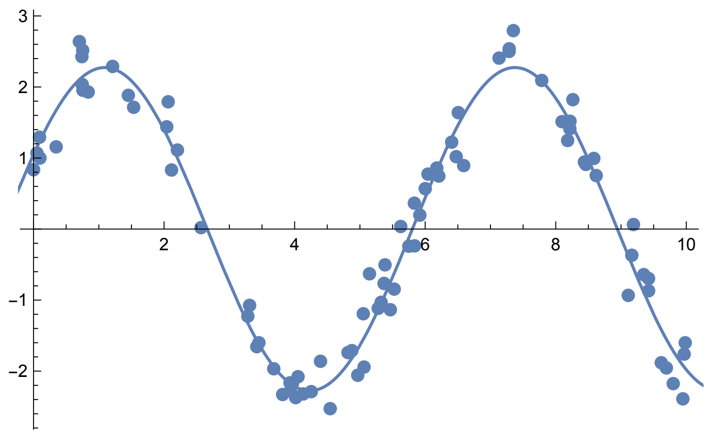


Model function is f = {1, Sin[t], Cos[t]}

Least squares solution = $\begin{pmatrix} -0.00117609 \\ 2.01716 \\ 1.05049 \end{pmatrix}$

Residual = 2.74365

Best fit model function = -0.00117609 + 1.05049 Cos[t] + 2.01716 Sin[t]



Example with non-random data - problem #10 in Section 6.6 (p.376).

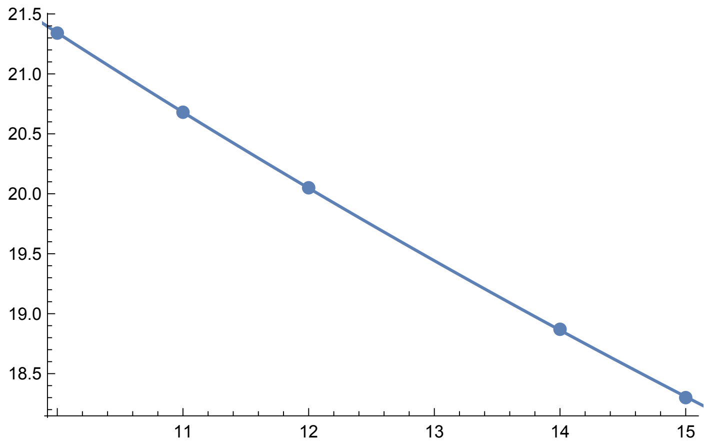
```
a = {10, 11, 12, 14, 15};
b = {21.34, 20.68, 20.05, 18.87, 18.30};

(*Establish model*)
Clear[t];
f = {e^(-0.02 t), e^(-0.07 t)};

(*Construct A*)
A = Table[f /. t → a[[i]], {i, 1, Length[a]}];
Print["Model function is f = ", f]
(*Print["A = ",A//MatrixForm];*)

(*Compute and plot least sqaures fit*)
β = Inverse[Aᵀ.A].Aᵀ.b;
Print["Least squares solution = ", β // MatrixForm]
e = b - A.β;
Print["Residual = ", Norm[e] // N]
h = f.β;
Print["Best fit model function = ", h]
Show[ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]],
 Plot[h, {t, Min[a] - 1, Max[a] + 1}]]
```

Model function is f = $\{e^{-0.02\,t}, e^{-0.07\,t}\}$

Least squares solution = $\begin{pmatrix} 19.9411 \\ 10.1015 \end{pmatrix}$

Residual = 0.0114745

Best fit model function = $10.1015\, e^{-0.07\,t} + 19.9411\, e^{-0.02\,t}$



Example with non-random data - problem #10 in Section 6.6 (p.376). We get a slightly different answer than the solution manual because their approach has round off error.

```
a = {44, 61, 81, 113, 131};
b = {91, 98, 103, 110, 112};

(*Establish model*)
Clear[t];
f = {1, Log[t]};

(*Construct A*)
A = Table[f /. t → a[[i]], {i, 1, Length[a]}];
Print["Model function is f = ", f]
(*Print["A = ",A//MatrixForm];*)

(*Compute and plot least sqaures fit*)
β = Inverse[Aᵀ.A].Aᵀ.b // N;
Print["Least squares solution = ", β // MatrixForm]
e = b - A.β;
Print["Residual = ", Norm[e] // N]
h = f.β;
Print["Best fit model function = ", h]
Show[ListPlot[Transpose[{a, b}], PlotStyle → PointSize[Large]],
 Plot[h, {t, Min[a] - 1, Max[a] + 1}]]
```

Model function is f = {1, Log[t]}

Least squares solution = $\begin{pmatrix} 17.9243 \\ 19.385 \end{pmatrix}$

Residual = 0.784091

Best fit model function = 17.9243 + 19.385 Log[t]