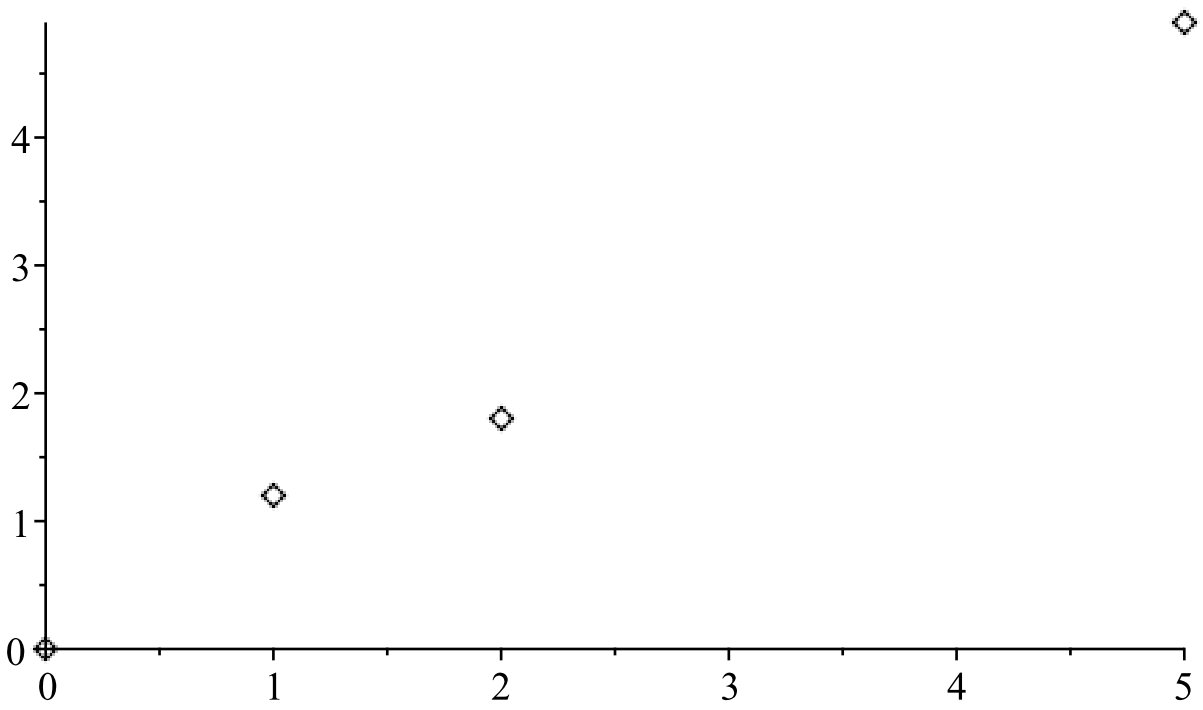
9/17/2019

```
> points := [[0, 0], [1, 1.2], [2, 1.8], [5, 4.9]]
```

$$points := [[0, 0], [1, 1.2], [2, 1.8], [5, 4.9]] \quad (1)$$

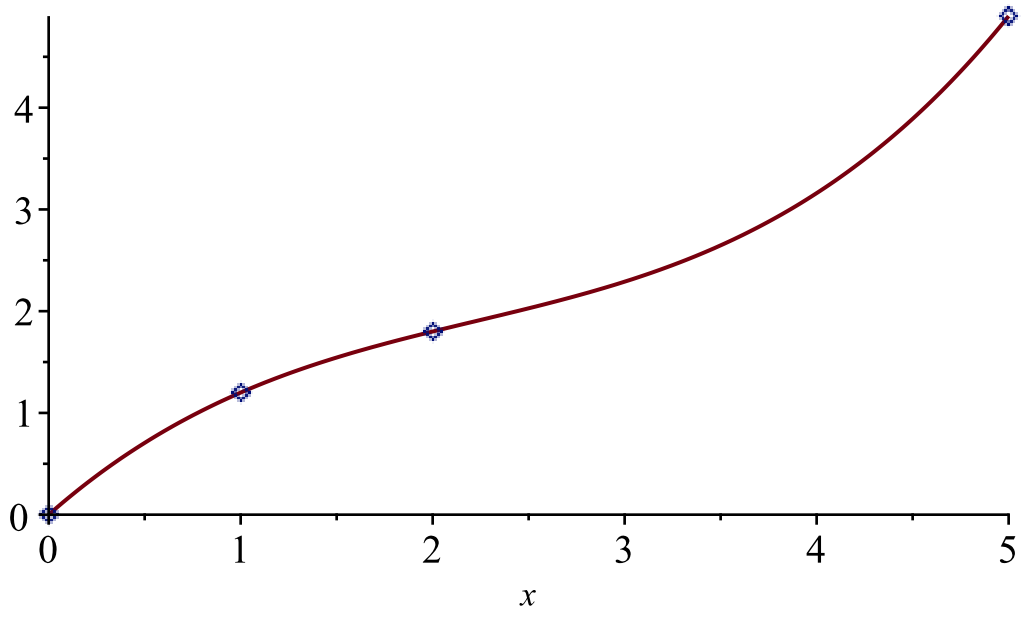
```
> with(plots):
> pointplot(points, symbolsize = 18)
```




```
> with(CurveFitting):
> p := PolynomialInterpolation(points, x);
```

$$p := 0.08166666667\,x^3 - 0.5450000000\,x^2 + 1.663333333\,x \quad (2)$$

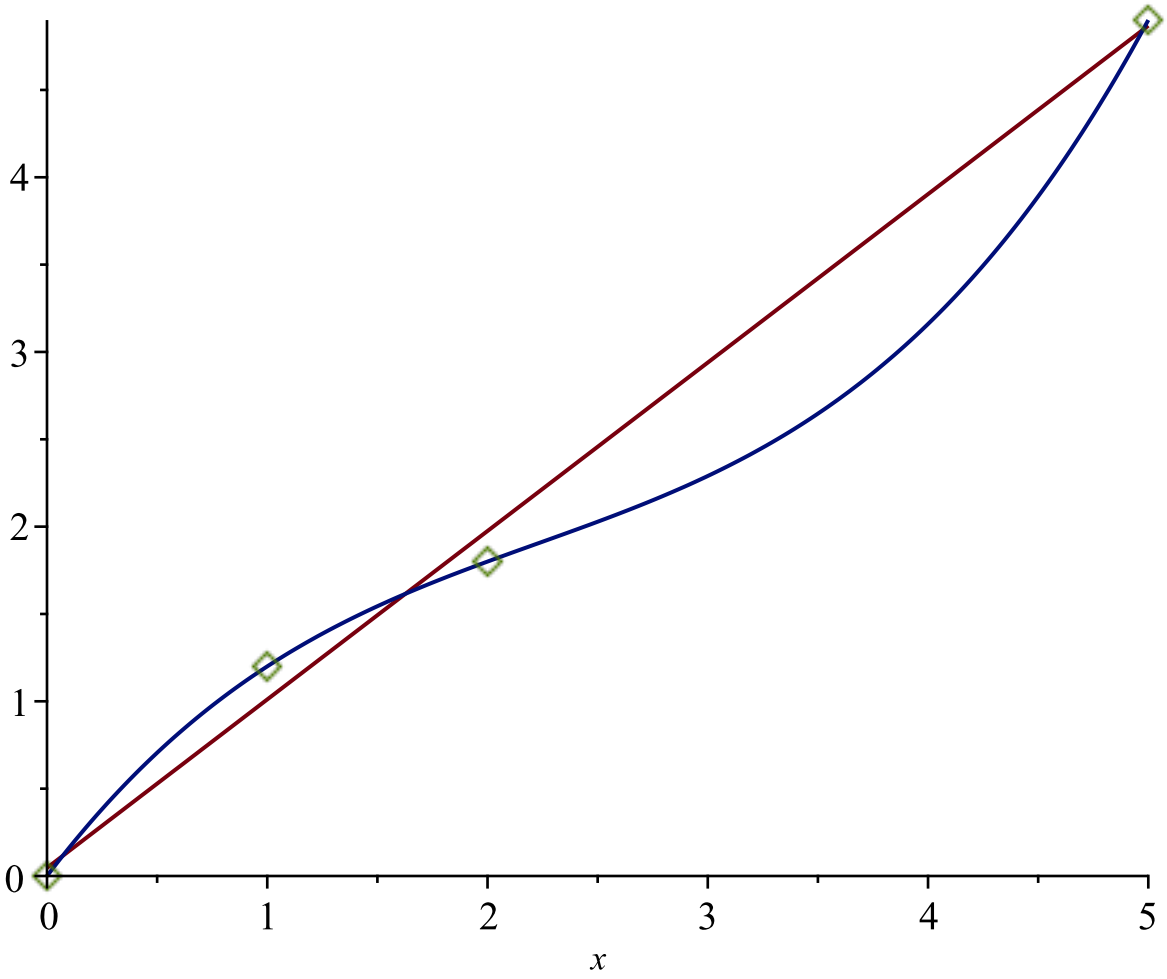
```
> plot([p, points], x = 0..5, style = [line, point], symbolsize = 18)
```



> $l := LeastSquares(points, x)$

$$l := 0.0464285714285716 + 0.964285714285714\,x \quad \textbf{(3)}$$

> $plot([l, p, points], x = 0..5, style = [line\$2, point], symbolsize = 18)$



> $points := [[0, 0], [1, 1.2], [2, 1.8], [5, 4.9]]$

$$points := [[0, 0], [1, 1.2], [2, 1.8], [5, 4.9]] \quad \textbf{(4)}$$

To do this by hand, we set up the function to minimize -- the sum of the squared error, take partials, solve.
Assume my line is y=m*x+b
error at x0 is eps0 = m*x0 + b - y0
add em up, take derivatives, blah blah...

> $err := p \rightarrow m \cdot p[1] + b - p[2]$

$$err := p \mapsto m\,p_1 + b - p_2 \quad \textbf{(5)}$$

> $err(points[1])$

$$b \quad \textbf{(6)}$$

> $err(points[2])$

$$b + m - 1.2 \quad \textbf{(7)}$$

Write a function for total error

> $E := pts \rightarrow add(err(pts[i])^2, i = 1..4)$

Warning, `i` is implicitly declared local to procedure `E`

$$E := pts \mapsto add\left(err\left(pts_i\right)^2, i = 1..4\right) \tag{8}$$

> $E(points)$

$$b^2 + (b + m - 1.2)^2 + (b + 2\,m - 1.8)^2 + (b + 5\,m - 4.9)^2 \tag{9}$$

Want to let function decide how many points

> $nops(points)$

$$4 \tag{10}$$

> $E := pts \rightarrow \frac{add\left(\ err(pts[i])^2, i = 1 \ldots nops(pts)\ \right)}{nops(pts)}$

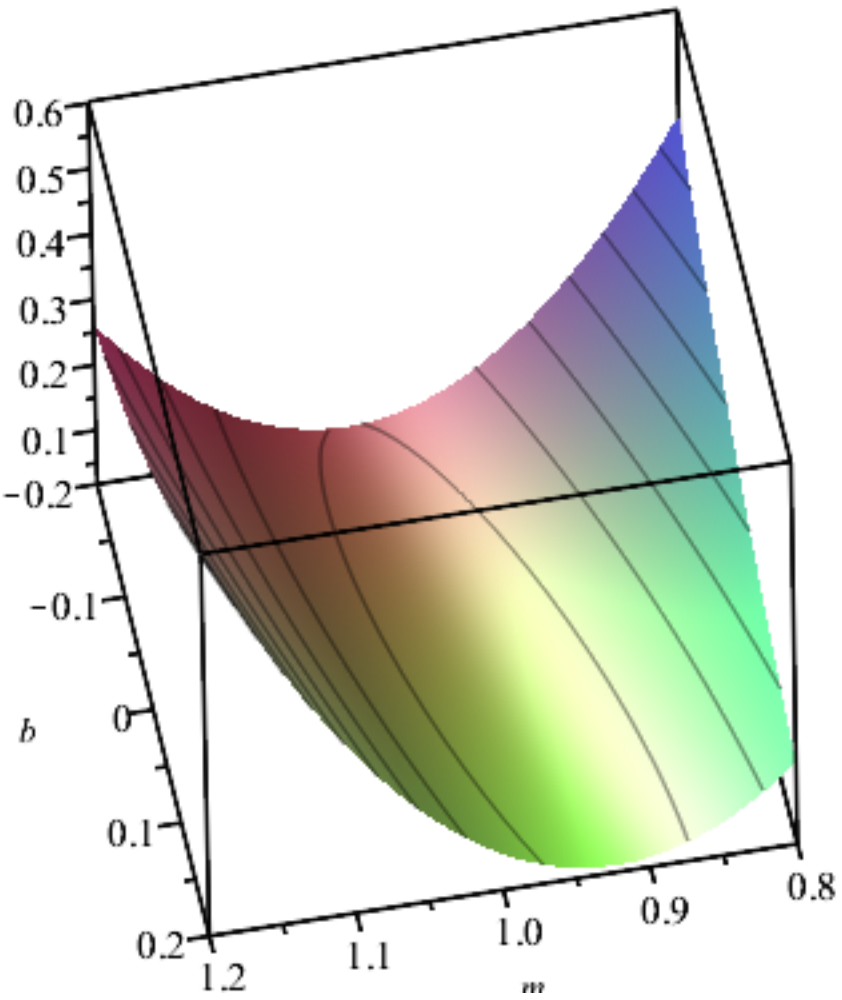
Warning, `i` is implicitly declared local to procedure `E`

$$E := pts \mapsto \frac{add\left(err\left(pts_i\right)^2, i = 1..nops(pts)\right)}{nops(pts)} \tag{11}$$

> $E(points)$

$$\frac{b^2}{4} + \frac{(b + m - 1.2)^2}{4} + \frac{(b + 2\,m - 1.8)^2}{4} + \frac{(b + 5\,m - 4.9)^2}{4} \tag{12}$$

> $plot3d(E(points), m = .8..1.2, b = -.2..0.2, style = patchcontour)$



> $md := diff(E(points), m)$

(13)

$$md := 4\,b + 15\,m - 14.65000000 \quad (13)$$

```
> mb := diff(E(points), b)
```

$$mb := 2\,b + 4\,m - 3.950000000 \quad (14)$$

```
> solve({md = 0, mb = 0})
```

$$\{b = 0.04642857143,\ m = 0.9642857143\} \quad (15)$$

```
> l := LeastSquares(points, x)
```

$$l := 0.0464285714285716 + 0.964285714285714\,x \quad (16)$$

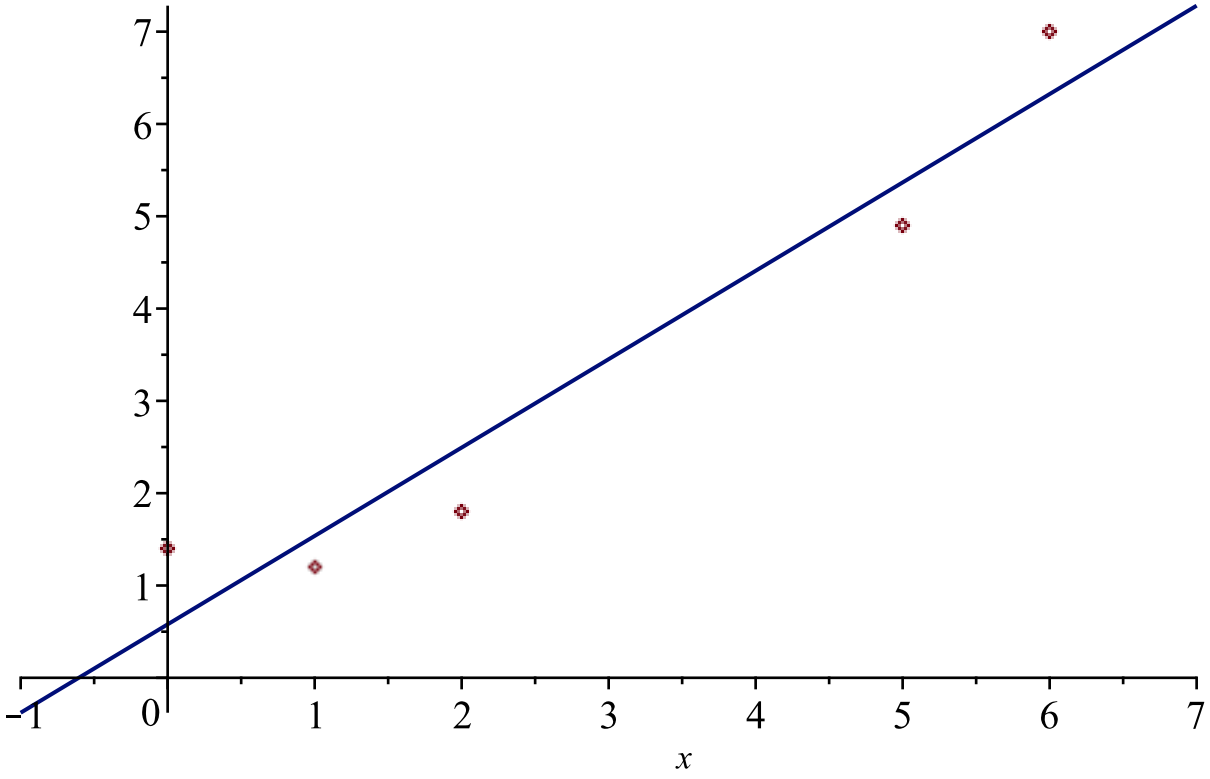
```
> [[0, 0], [1, 1.2], [2, 1.8], [5, 4.9], [6, 7]]
```

$$[[0, 0], [1, 1.2], [2, 1.8], [5, 4.9], [6, 7]] \quad (17)$$

```
> newl := LeastSquares(newguy, x);
```

$$newl := 0.579104477611941 + 0.957462686567164\,x \quad (18)$$

```
> plot([newguy, newl], x = -1..7, style = [point, line])
```



Let's fit a quadratic y=a*x^2 + b*x + c to my newguy data.

```
> E := pts -> add(err(pts[i])^2, i = 1 .. nops(pts)) / nops(pts)
```

Warning, `i` is implicitly declared local to procedure `E`

$$E := pts \mapsto \frac{add\left(err\left(pts_i\right)^2, i = 1..nops(pts)\right)}{nops(pts)} \quad (19)$$

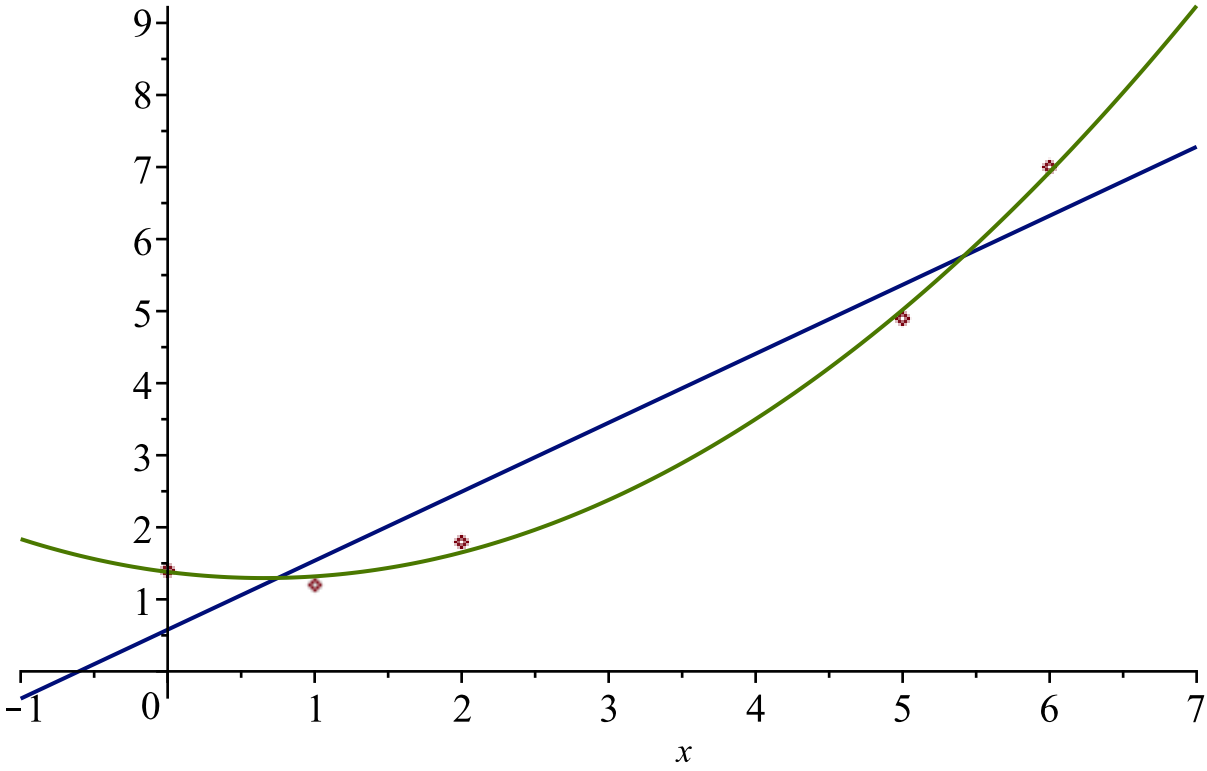
```
> err := p -> a*p[1]^2 + b*p[1] + c - p[2]:
> thing := E(newguy):
> ABC := solve({diff(thing, a) = 0, diff(thing, b) = 0, diff(thing, c) = 0})
```

$$ABC := \{a = 0.1973522167,\ b = -0.2590517241,\ c = 1.380295567\} \quad (20)$$

> $Q := subs(ABC, a \cdot x^2 + b \cdot x + c)$

$$Q := 0.1973522167\, x^2 - 0.2590517241\, x + 1.380295567 \quad \textbf{(21)}$$

> $plot([newguy, newl, Q], x = -1..7, style = [point, line\$2])$



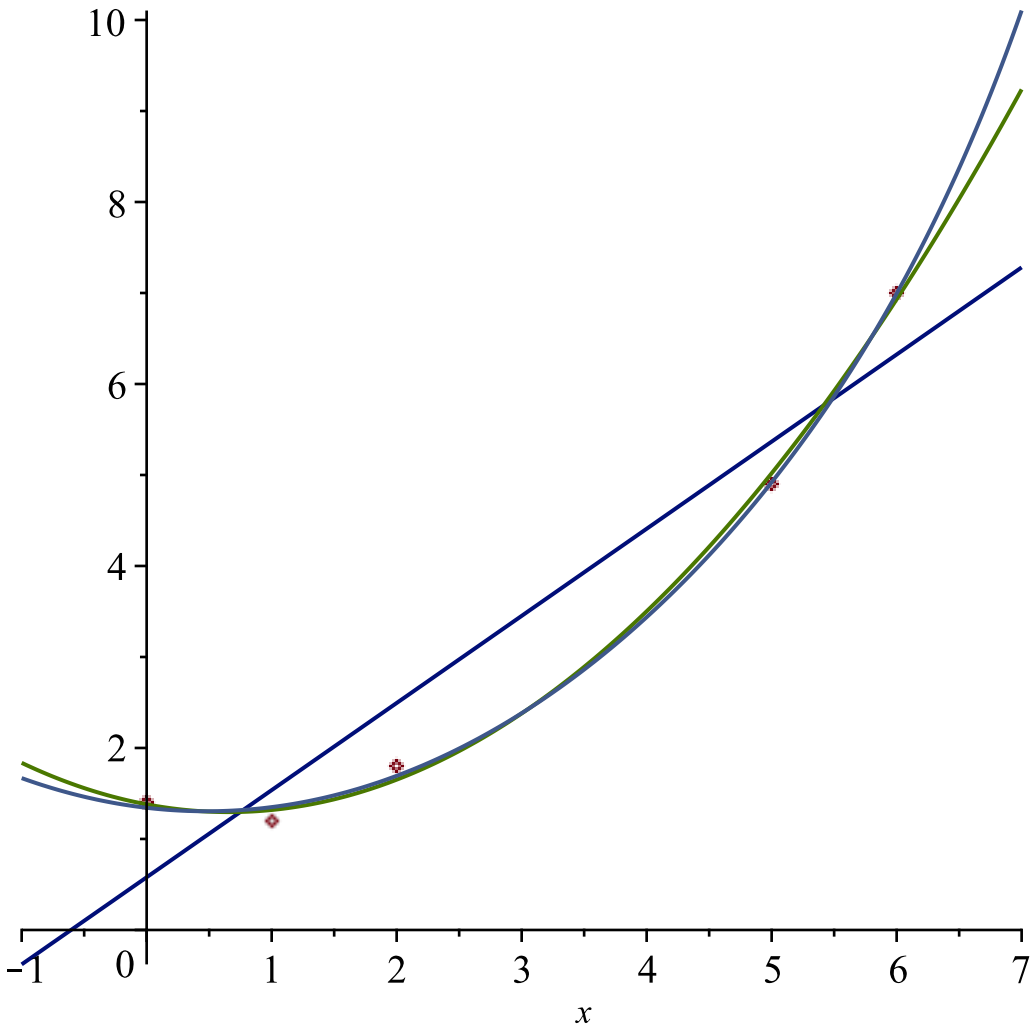
> $LeastSquares(newguy, x, curve = a \cdot x^2 + b \cdot x + c)$

$$1.38029556650246 - 0.259051724137931\, x + 0.197352216748768\, x^2 \quad \textbf{(22)}$$

> $QE := LeastSquares(newguy, x, curve = d \cdot \exp(x) + a \cdot x^2 + b \cdot x + c)$

$$QE := 1.340954938 - 0.1612964565\, x + 0.1658744719\, x^2 + 0.001608778212\, e^x \quad \textbf{(23)}$$

> $plot([newguy, newl, Q, QE], x = -1..7, style = [point, line\$6])$



curve needs to depend linearly on coeffs. So this fails:

```
> NOPE := LeastSquares(newguy, x, curve = a·sin(b·x) + c·cos(d·x))
Error, (in CurveFitting:-LeastSquares) curve to fit is not linear in the parameters
```

>

Let's try minimizing the mean-quartic error (ie, 4th power)

```
> E := pts → add( err(pts[i])^4, i = 1 .. nops(pts) ) / nops(pts)
Warning, `i` is implicitly declared local to procedure `E`
```

$$E := pts \mapsto \frac{add\left(err\left(pts_i\right)^4, i = 1 \,..nops(pts)\right)}{nops(pts)} \tag{24}$$

```
> err := p→m·p[1] + b − p[2];
```

$$err := p \mapsto m\,p_1 + b - p_2 \tag{25}$$

```
> thing := E(newguy);
```

$$thing := \frac{(b-1.4)^4}{5} + \frac{(b+m-1.2)^4}{5} + \frac{(b+2\,m-1.8)^4}{5} + \frac{(b+5\,m-4.9)^4}{5} + \frac{(b+6\,m-7)^4}{5} \tag{26}$$

```
> ABC := solve( {diff(thing, m) = 0, diff(thing, b) = 0})
```

$$ABC := \{b = 0.6733429812, m = 0.9466208219\}, \{b = 0.6439920990 + 0.2736493403\,I, m = 1.018533813 - 0.2277463280\,I\}, \{b = 1.059411553 + 1.297318403\,I, m = 0.8823968023 - 0.3097733817\,I\}, \{b = 1.148343207 + 1.550020341\,I, m = 1.014794135 - 0.4191710161\,I\}, \{b = 0.5774570383 + 1.128721934\,I, m = 0.9458888432 - 0.01682123138\,I\}, \{b = 0.5774570383 - 1.128721934\,I, m = 0.9458888432 + 0.01682123138\,I\}, \{b = 1.148343207 - 1.550020341\,I, m = 1.014794135 + 0.4191710161\,I\}, \{b = 1.059411553 - 1.297318403\,I, m = 0.8823968023 + 0.3097733817\,I\}, \{b = 0.6439920990 - 0.2736493403\,I, m = 1.018533813 + 0.2277463280\,I\} \quad (27)$$

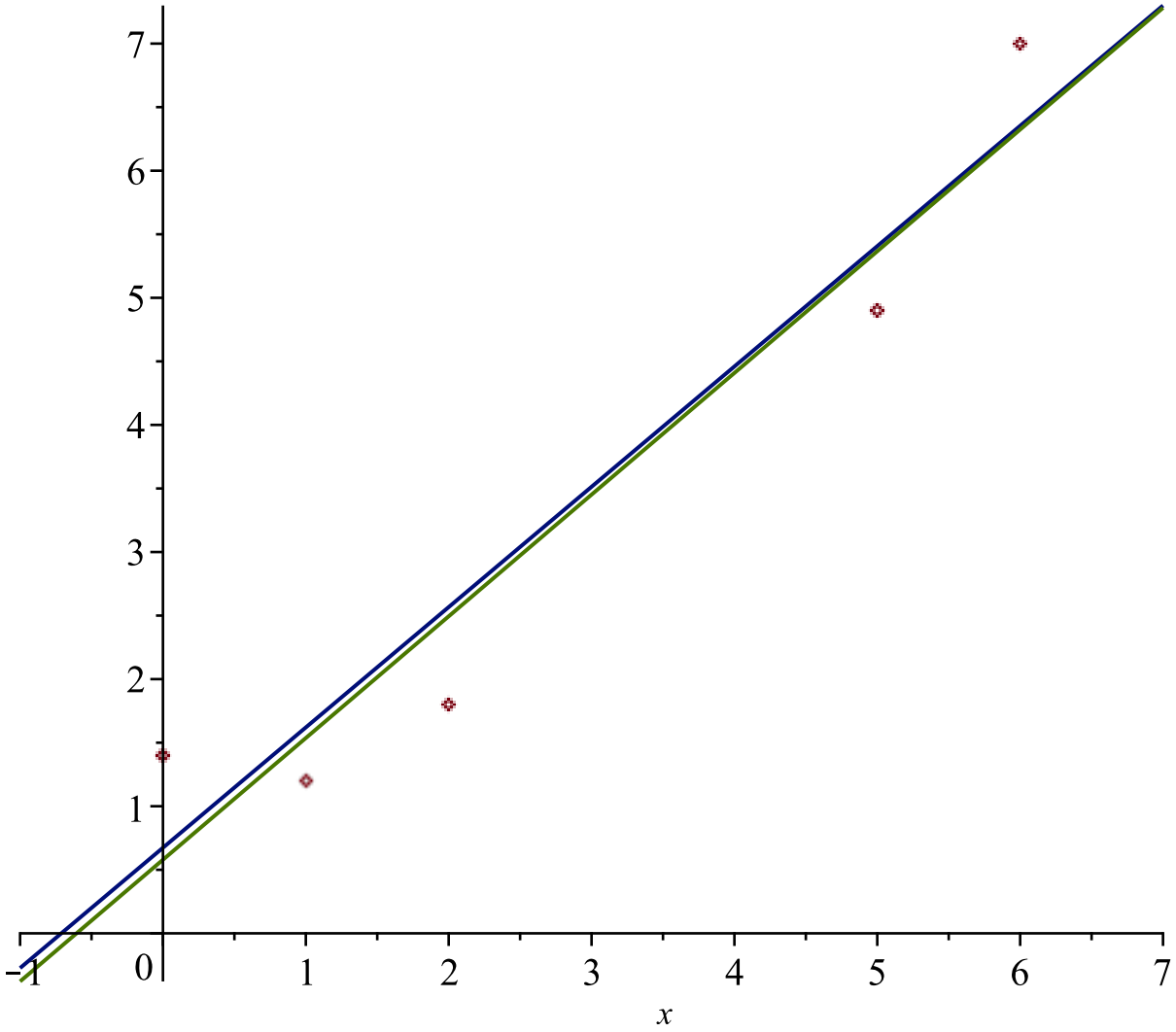
```
> qline := subs(ABC[1], m·x + b)
```

$$qline := 0.9466208219\,x + 0.6733429812 \quad (28)$$

```
> newl
```

$$0.579104477611941 + 0.957462686567164\,x \quad (29)$$

```
> plot([newguy, qline, newl], x = -1..7, style = [point, line$4])
```




```
>
```