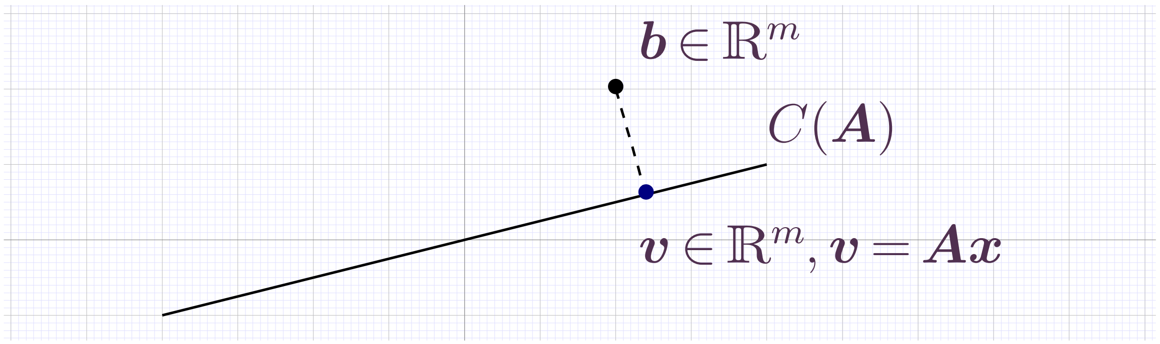
- New concepts:
  - Polynomial interpolation
  - LSQ polynomial approximant

- Consider approximating $\boldsymbol{b} \in \mathbb{R}^m$ by linear combination of n vectors, $\boldsymbol{A} \in \mathbb{R}^{m \times n}$
- Make approximation error $\boldsymbol{e} = \boldsymbol{b} - \boldsymbol{v} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$ as small as possible

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2$$

Error is measured in the 2-norm $\Rightarrow$ the *least squares problem* (LSQ)



- Solution is the projection of $\boldsymbol{b}$ onto $C(\boldsymbol{A})$

$$\boldsymbol{Q}\boldsymbol{R} = \boldsymbol{A}, \boldsymbol{P}_{C(\boldsymbol{A})} = \boldsymbol{Q}\boldsymbol{Q}^T, \boldsymbol{v} = (\boldsymbol{Q}\boldsymbol{Q}^T)\boldsymbol{b}$$

- The vector $\boldsymbol{x}$ is found by back-substitution from

$$\boldsymbol{v} = (\boldsymbol{Q}\boldsymbol{Q}^T)\boldsymbol{b} = (\boldsymbol{Q}\boldsymbol{R})\boldsymbol{x} \Rightarrow \boldsymbol{R}\boldsymbol{x} = \boldsymbol{Q}^T\boldsymbol{b}.$$

- LSQ has myriad applications throughout science
- Consider data $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- The *polynomial interpolant* of degree $m-1$ passes through the data points

$$p_{m-1}(x) = c_0 + c_1 x + \cdots + c_{m-1} x^{m-1} = \begin{bmatrix} 1 & x & \ldots & x^{m-1} \end{bmatrix} \boldsymbol{c}$$

- Impose conditions $p_{m-1}(x_i) = y_i, i = 1, \ldots, m$ to obtain linear system

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^{m-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{m-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \ldots & x_m^{m-1} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{m-1} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix} \Rightarrow \boldsymbol{A}\boldsymbol{c} = \boldsymbol{y}, \boldsymbol{A} \in \mathbb{R}^{m \times m}$$

```
>> m=4; x=transpose(1:m); y=1 - 2*x + 3*x.^2 - 4*x.^3;
```

```
>> A=[x.^0 x.^1 x.^2 x.^3]; [Q,R]=qr(A); c = R \ (Q'*y);␣c'
```

- LSQ has myriad applications throughout science
- Consider data $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- The *polynomial interpolant* of degree $m-1$ passes through the data points

$$p_{m-1}(x) = c_0 + c_1 x + \cdots + c_{m-1} x^{m-1} = \left[\begin{array}{cccc} 1 & x & \ldots & x^{m-1} \end{array}\right] \boldsymbol{c}$$

- Impose conditions $p_{m-1}(x_i) = y_i, i = 1, \ldots, m$ to obtain linear system

$$\left[\begin{array}{ccccc} 1 & x_1 & x_1^2 & \ldots & x_1^{m-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{m-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \ldots & x_m^{m-1} \end{array}\right] \left[\begin{array}{c} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{m-1} \end{array}\right] = \left[\begin{array}{c} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{array}\right] \Rightarrow \boldsymbol{A}\boldsymbol{c} = \boldsymbol{y}, \boldsymbol{A} \in \mathbb{R}^{m \times m}$$

```
>> m=4; x=transpose(1:m); y=1 - 2*x + 3*x.^2 - 4*x.^3;
```

```
>> A=[x.^0 x.^1 x.^2 x.^3]; [Q,R]=qr(A); c = R \ (Q'*y); c'
```

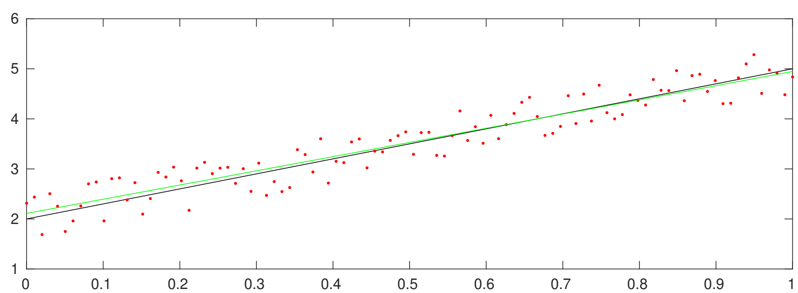
$\left(\begin{array}{cccc} 1.0000 & -2.0000 & 3.0000 & -4.0000 \end{array}\right)$

- Consider noisy data containing many measurements $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- Assume data without noise would conform to some polynomial law

$$y_i = z_i + r_i = c_0 + c_1 x_i + r_i, i = 1, \ldots, m$$

with r_i a random number uniformly distributed within $(-R, R)$.

```
>> m=100; x=linspace(0,1,m)';
>> c0=2; c1=3; z=c0+c1*x;  y=(z+rand(m,1)-0.5);
>> A=ones(m,2); A(:,2)=x(:); [Q,R]=qr(A); c = R \ (Q'*y);␣c'
>> w=A*c; plot(x,z,'k',x,y,'.r',x,w,'g');
```



- Consider noisy data containing many measurements $\mathcal{D}=\{(x_i,y_i),i=1,\ldots,m\}$
- Assume data without noise would conform to some polynomial law

$$y_i = z_i + r_i = c_0 + c_1 x_i + r_i, i = 1, \ldots, m$$

with r_i a random number uniformly distributed within $(-R, R)$.

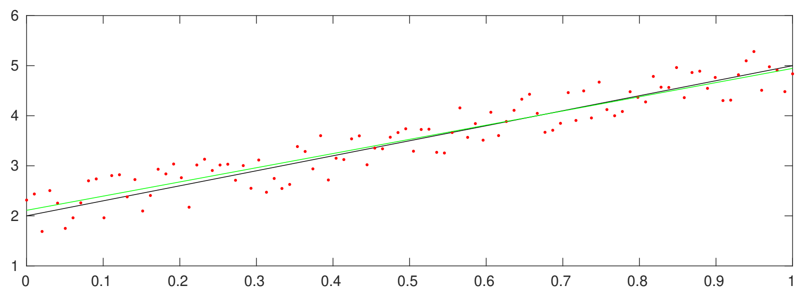
```
>> m=100; x=linspace(0,1,m)';
```

```
>> c0=2; c1=3; z=c0+c1*x;  y=(z+rand(m,1)-0.5);
```

```
>> A=ones(m,2); A(:,2)=x(:); [Q,R]=qr(A); c = R \ (Q'*y);␣c'
```

(1.9506 3.0310)

```
>> w=A*c; plot(x,z,'k',x,y,'.r',x,w,'g');
```

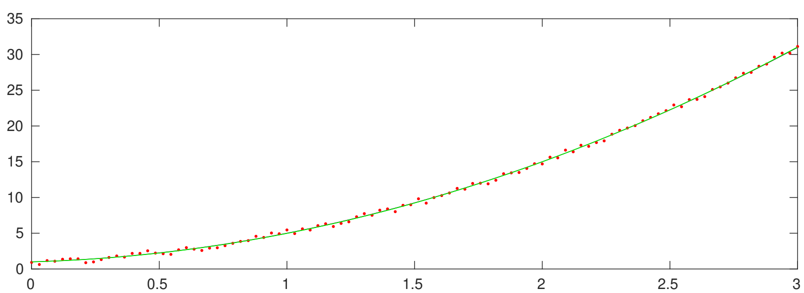


- Consider noisy data containing many measurements $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- Assume data without noise would conform to some polynomial law

$$y_i = z_i + r_i = c_0 + c_1 x_i + r_i, i = 1, \ldots, m$$

with r_i a random number uniformly distributed within $(-R, R)$.

```
>> m=100; x=linspace(0,3,m)';
>> c0=1; c1=1; c2=3; z=c0+c1*x+c2*x.^2;  y=(z+rand(m,1)-0.5);
>> A=ones(m,3); A(:,2)=x(:); A(:,3)=x(:).^2; [Q,R]=qr(A); c=R\(Q'*y);␣c'
>> w=A*c; plot(x,z,'k',x,y,'.r',x,w,'g');
```



- Consider noisy data containing many measurements $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- Assume data without noise would conform to some polynomial law

$$y_i = z_i + r_i = c_0 + c_1 x_i + r_i, i = 1, \ldots, m$$

with r_i a random number uniformly distributed within $(-R, R)$.

```
>> m=100; x=linspace(0,3,m)';
```

```
>> c0=1; c1=1; c2=3; z=c0+c1*x+c2*x.^2;  y=(z+rand(m,1)-0.5);
```

```
>> A=ones(m,3); A(:,2)=x(:); A(:,3)=x(:).^2; [Q,R]=qr(A); c=R\(Q'*y);␣c'
```

(1.0047 0.9641 3.0143)

```
>> w=A*c; plot(x,z,'k',x,y,'.r',x,w,'g');
```