More review of mathematical tools:

- Linear regression
- Matrix formulation
- Multiple regression
- Matrix methods

- A first step in most biological studies is data acquisition and processing
- Consider data $\mathcal{D} = \{(x_i, y_i), i = 1, 2, \ldots, m\}$
- Hypothesis: data arises from some underlying function f but may be affected by experimental measurement errors

$$y_i = f(x_i) + e_i, i = 1, 2, \ldots, m$$

- Hypothesis: The function f is a first-degree polynomial $f(t) = c_0 + c_1 t$
- Recover c_0, c_1 to minimize the sum of squared observation errors e_i

$$E = \sum_{i=1}^{m} e_i^2$$

- Obtain a least-squares problem

$$\min_{a_0, a_1} E$$

- One way to obtain a_0, a_1 is through calculus stationarity conditions

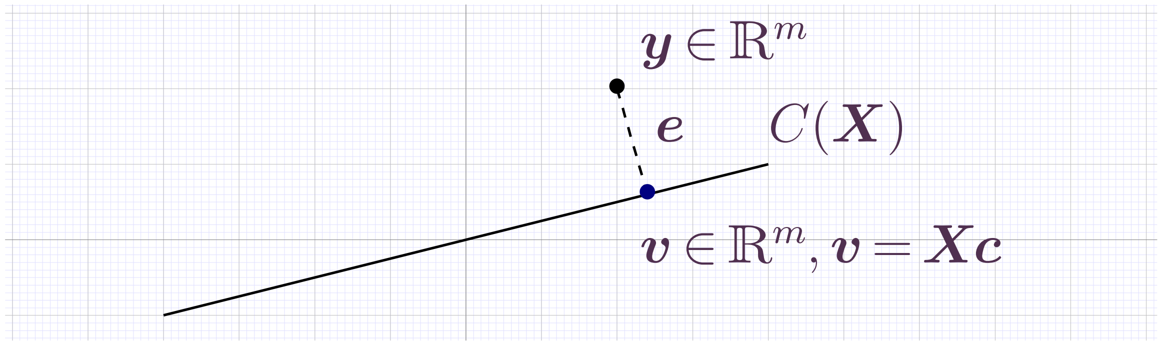
$$\frac{\partial E}{\partial c_0} = 0, \frac{\partial E}{\partial c_1} = 0.$$

$$\frac{\partial E}{\partial c_0} = \frac{\partial}{\partial c_0} \sum_{i=1}^{m} (y_i - c_0 - c_1 x_i)^2 = -2 \sum_{i=1}^{m} (y_i - c_0 - c_1 x_i) = 0$$

$$\frac{\partial E}{\partial c_1} = \frac{\partial}{\partial c_1} \sum_{i=1}^{m} (y_i - c_0 - c_1 x_i)^2 = -2 \sum_{i=1}^{m} (y_i - c_0 - c_1 x_i) x_i = 0$$

- These the *normal equations* but are not a recommended approach

- The best approximant is found when the error vector $\boldsymbol{e}$ is orthogonal to $C(\boldsymbol{A})$



$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}, \boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}^0 & \boldsymbol{x}^1 \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_m \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}, \boldsymbol{e} = \boldsymbol{v} - \boldsymbol{y}$$

$$\boldsymbol{e} \perp C(\boldsymbol{X}) \Rightarrow \boldsymbol{X}^T\boldsymbol{e} = 0 \Rightarrow \boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{c}) = 0 \Rightarrow (\boldsymbol{X}^T\boldsymbol{X})\,\boldsymbol{c} = \boldsymbol{X}^T\boldsymbol{y}$$

- The system $(\boldsymbol{X}^T\boldsymbol{X})\,\boldsymbol{c} = \boldsymbol{X}^T\boldsymbol{y}$ is *the normal system* of the LSQ problem
- Note that $\boldsymbol{X}^T\boldsymbol{X} \in \mathbb{R}^{n \times n}$, $n = 2$

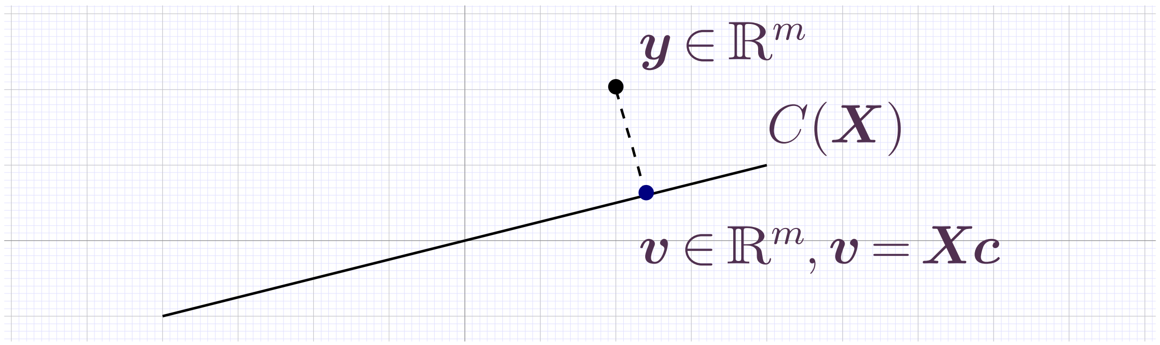
- Notice that $y_i \cong c_0 \cdot 1 + c_1 \cdot x_i$ is then expressed as

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} \cong \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_m \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \end{bmatrix} \Rightarrow \boldsymbol{y} \cong \boldsymbol{X}\boldsymbol{c}$$

- Least squares problem is then

$$\min_{\boldsymbol{c} \in \mathbb{R}^2} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{c}\|_2$$

- This is solved by projection: $\boldsymbol{Q}\boldsymbol{R} = \boldsymbol{X}$, $\boldsymbol{P}_{C(\boldsymbol{X})} = \boldsymbol{Q}\boldsymbol{Q}^T$, $\boldsymbol{R}\boldsymbol{c} = \boldsymbol{Q}^T\boldsymbol{y}$



- Consider noisy data containing many measurements $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- Assume data without noise would conform to a linear polynomial $z_i = c_0 + c_1 x_i$

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

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

```
In[38]:= m=100; h=1.0/(m-1); x=Table[i h,{i,0,m-1}];
         c0=2; c1=3; z=c0+c1*x; R=0.0;
         r=Table[Random[Real,{-R,R}],m];
         y=z+r;
In[42]:= data = Transpose[{x,y}]; Fit[data,{1,t},t]
In[43]:=
```

- Consider noisy data containing many measurements $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- Assume data without noise would conform to a linear polynomial $z_i = c_0 + c_1 x_i$

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

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

```
In[38]:= m=100; h=1.0/(m-1); x=Table[i h,{i,0,m-1}];
         c0=2; c1=3; z=c0+c1*x; R=0.0;
         r=Table[Random[Real,{-R,R}],m];
         y=z+r;
```

```
In[42]:= data = Transpose[{x,y}]; Fit[data,{1,t},t]
```

```
In[43]:=
```

- Consider noisy data containing many measurements $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- Assume data without noise would conform to a linear polynomial $z_i = c_0 + c_1 x_i$

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

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

```
In[38]:= m=100; h=1.0/(m-1); x=Table[i h,{i,0,m-1}];
         c0=2; c1=3; z=c0+c1*x; R=0.0;
         r=Table[Random[Real,{-R,R}],m];
         y=z+r;
```

```
In[42]:= data = Transpose[{x,y}]; Fit[data,{1,t},t]
```

$3.\,t + 2.$

```
In[43]:=
```

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

$$y_i = z_i + r_i = c_0 + c_1 x_i + c_2 x_i^2 + c_3 x_i^3 + r_i, i = 1, \ldots, m$$

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

```
In[43]:= m=100; h=1.0/(m-1); x=Table[i h,{i,0,m-1}];
         c0=2; c1=3; c2=-1; c3=1; z=c0+c1*x+c2*x^2+c3*x^3; R=0.1;
         r=Table[Random[Real,{-R,R}],m];
         y=z+r;
In[46]:= data = Transpose[{x,y}]; Fit[data,{1,t,t^2,t^3},t]
In[47]:=
```

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

$$y_i = z_i + r_i = c_0 + c_1 x_i + c_2 x_i^2 + c_3 x_i^3 + r_i, i = 1, \ldots, m$$

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

```
In[43]:= m=100; h=1.0/(m-1); x=Table[i h,{i,0,m-1}];
         c0=2; c1=3; c2=-1; c3=1; z=c0+c1*x+c2*x^2+c3*x^3; R=0.1;
         r=Table[Random[Real,{-R,R}],m];
         y=z+r;
```

```
In[46]:= data = Transpose[{x,y}]; Fit[data,{1,t,t^2,t^3},t]
```

```
In[47]:=
```

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

$$y_i = z_i + r_i = c_0 + c_1 x_i + c_2 x_i^2 + c_3 x_i^3 + r_i, i = 1, \ldots, m$$

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

```
In[43]:= m=100; h=1.0/(m-1); x=Table[i h,{i,0,m-1}];
         c0=2; c1=3; c2=-1; c3=1; z=c0+c1*x+c2*x^2+c3*x^3; R=0.1;
         r=Table[Random[Real,{-R,R}],m];
         y=z+r;
```

```
In[46]:= data = Transpose[{x,y}]; Fit[data,{1,t,t^2,t^3},t]
```

$$0.773748\, t^3 - 0.539778\, t^2 + 2.75652\, t + 2.02553$$

```
In[47]:=
```

- Consider noisy data containing many measurements $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- Assume data without noise would conform to a linear polynomial $z_i = c_0 + c_1 x_i$

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

```
In[67]:= m=100; h=1.0/m; x=Table[i h,{i,1,m}];
         c0=2; c1=3; z=c0+c1*x; L=0.2;
         r=Table[Random[Real,{-L,L}],m];
         y=z+r;
In[71]:= X=Transpose[{x^0,x^1}]; {Q,R}=QRDecomposition[X];
In[72]:= c = LinearSolve[R,Q.y]
In[73]:=
```

- Consider noisy data containing many measurements $\mathcal{D}=\{(x_i, y_i), i=1, \ldots, m\}$
- Assume data without noise would conform to a linear polynomial $z_i = c_0 + c_1 x_i$

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

```
In[67]:= m=100; h=1.0/m; x=Table[i h,{i,1,m}];
         c0=2; c1=3; z=c0+c1*x; L=0.2;
         r=Table[Random[Real,{-L,L}],m];
         y=z+r;
```

```
In[71]:= X=Transpose[{x^0,x^1}]; {Q,R}=QRDecomposition[X];
In[72]:= c = LinearSolve[R,Q.y]
In[73]:=
```

- Consider noisy data containing many measurements $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- Assume data without noise would conform to a linear polynomial $z_i = c_0 + c_1 x_i$

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

```
In[67]:= m=100; h=1.0/m; x=Table[i h,{i,1,m}];
         c0=2; c1=3; z=c0+c1*x; L=0.2;
         r=Table[Random[Real,{-L,L}],m];
         y=z+r;
```

```
In[71]:= X=Transpose[{x^0,x^1}]; {Q,R}=QRDecomposition[X];
```

```
In[72]:= c = LinearSolve[R,Q.y]
```

```
In[73]:=
```

- Consider noisy data containing many measurements $\mathcal{D} = \{(x_i, y_i), i = 1, \ldots, m\}$
- Assume data without noise would conform to a linear polynomial $z_i = c_0 + c_1 x_i$

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

```
In[67]:= m=100; h=1.0/m; x=Table[i h,{i,1,m}];
         c0=2; c1=3; z=c0+c1*x; L=0.2;
         r=Table[Random[Real,{-L,L}],m];
         y=z+r;
```

```
In[71]:= X=Transpose[{x^0,x^1}]; {Q,R}=QRDecomposition[X];
```

```
In[72]:= c = LinearSolve[R,Q.y]
```

$\{2.00376, 3.01624\}$

```
In[73]:=
```