Linear Algebra Project (Week #3)

Least-Squares Approximation

Suppose we have obtained experimentally the following data:

$\boldsymbol{x}$:	x_1	x_2	x_3	...	x_n
$\boldsymbol{y}$:	y_1	y_2	y_3	...	y_n

That is (x_1, y_1), (x_2, y_2), (x_3, y_3),…,(x_n, y_n).

We want a mathematical model function, $y = f(x)$, between x and y that best fits all the points above, with minimal error. This can help to approximate future values of y given an expected value of x.

Depending on how the data is distributed and the degree of precision we need, we can choose three different types of model function:

- Linear polynomial $y = a + bx$
- Quadratic polynomial $y = a + bx + cx^2$
- Cubic polynomial $y = a + bx + cx^2 + dx^3$

Let's look at the technique creating a linear model function.

1. Convert the data into a system of linear equations, with unknown coefficients a and b:

$$\begin{cases} y_1 = a + bx_1 \\ y_2 = a + bx_2 \\ y_3 = a + bx_3 \\ \quad\vdots \\ y_n = a + bx_n \end{cases}$$

2. Translate the system to matrix form:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ 1 & \vdots \\ 1 & x_n \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix}$$

3. Create some new labels:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} = \vec{u} \qquad \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ 1 & \vdots \\ 1 & x_n \end{bmatrix} = M \qquad \begin{bmatrix} a \\ b \end{bmatrix} = \vec{v}$$

4. Substitute these new labels into the system's matrix form to create a simplified equation:

$$\vec{u} = M \cdot \vec{v}$$

5. Solve for the unknown coefficients ($\vec{v}$): $\vec{v} = (M^{\mathrm{t}} \cdot M)^{-1} \cdot M^{\mathrm{t}} \cdot \vec{u}$

> Note that we cannot directly solve the equation by inverting matrix M ($\vec{v} = M^{-1} \cdot \vec{u}$), because M is not a square matrix. So we must use a more complicated method, called a "least-squares approximation," to get as close to a solution as possible:
>
> 1. Multiply both sides of our equation by the transpose of M: $M^{\mathrm{t}} \cdot \vec{u} = M^{\mathrm{t}} \cdot M \cdot \vec{v}$.
> 2. Since $M^{\mathrm{t}} \cdot M$ produces a square matrix, check to see if it's invertible: Does $\det(M^{\mathrm{t}} \cdot M) \neq 0$?
> 3. If $M^{\mathrm{t}} \cdot M$ is invertible, multiply both sides of the equation by the inverse: $(M^{\mathrm{t}} \cdot M)^{-1} \cdot M^{\mathrm{t}} \cdot \vec{u} = (M^{\mathrm{t}} \cdot M)^{-1} \cdot M^{\mathrm{t}} \cdot M \cdot \vec{v}$
> 4. Because $(M^{\mathrm{t}} \cdot M)^{-1} \cdot M^{\mathrm{t}} \cdot M = I$, we can reduce the equation to: $\vec{v} = (M^{\mathrm{t}} \cdot M)^{-1} \cdot M^{\mathrm{t}} \cdot \vec{u}$

Example: Given the data in the following table, find the most-precise linear model equation:

$\boldsymbol{x}$:	2	3	4	5
$\boldsymbol{y}$:	3	5	3	6

Systems of equations:

$$\begin{cases} 3 = a + 2b \\ 5 = a + 3b \\ 3 = a + 4b \\ 6 = a + 5b \end{cases}$$

Matrix form:

$$\begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 1 & 4 \\ 1 & 5 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 3 \\ 5 \\ 3 \\ 6 \end{bmatrix}$$

Simplified form:

$$M \cdot \vec{v} = \vec{u}$$

Solution:

$$\vec{v} = (M^{\mathrm{t}} \cdot M)^{-1} \cdot M^{\mathrm{t}} \cdot \vec{u}$$

$$\vec{v} = \left\{ \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 3 & 4 & 5 \end{bmatrix} \cdot \begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 1 & 4 \\ 1 & 5 \end{bmatrix} \right\}^{-1} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 3 & 4 & 5 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 5 \\ 3 \\ 6 \end{bmatrix}$$

$$\vec{v} = \begin{bmatrix} 1.8 \\ 0.7 \end{bmatrix}$$

Linear model: $\boxed{y = 1.8 + 0.7x}$

> Note: If our data does not appear to have a linear distribution or if we need greater precision, we can increase the number of terms in our model equation, moving from a linear equation to a quadratic equation to a cubic equation and beyond. This will change the size of our $\vec{u}$ and M:
>
> $$\vec{u} = M \cdot \vec{v} \leftrightarrow [n \times 1] = [n \times o] \cdot [o \times 1]$$
>
> Where n is the number of data points, o is the degree of the model equation, and $n \geq o$.

Python Procedure

We'll illustrate the Python code needed for this approach using the data set we just used for our example above to create the quadratic model **and** make a prediction for $x = 6$.

1. Import Python's `sympy` library:

```
import sympy as sy
```

2. Create our matrix M (and printing it to make sure it's been entered correctly):

```
M = sy.Matrix([[1,2,4],
               [1,3,9],
               [1,4,16],
               [1,5,25]])
sy.pprint(M)
```

3. Create our vector $\vec{u}$:

```
u = sy.Matrix([3,5,3,6])
```

4. Calculate the solutions for our approximate model:

```
v = (M.T * M).inv() * M.T * u
sy.pprint(v)
```

5. Create a model equation, with which to make predictions:

```
def f2(x):
    return v[0] + v[1] * x + v[2] * x ** 2
```

6. Calculate the predicted value for $x = 6$:

```
print("Prediction at x = 6")
print(f2(6))
```

Your output should look like this:

```
⎡1  2  4 ⎤
⎢        ⎥
⎢1  3  9 ⎥
⎢        ⎥
⎢1  4  16⎥
⎢        ⎥
⎣1  5  25⎦
⎡ 91 ⎤
⎢ ── ⎥
⎢ 20 ⎥
⎢    ⎥
⎢-21 ⎥
⎢────⎥
⎢ 20 ⎥
⎢    ⎥
⎣1/4 ⎦
Prediction at x = 6
29/4
```

Creating a Cubic Approximation

Using Python and the least-squares approximation method outlined above, find a cubic model equation ($y = a + bx + cx^2 + dx^3$) for the following data ***and*** predict the y-value for $x = 8$:

$\boldsymbol{x}$**:**	1	2	3	4	5	6	7
$\boldsymbol{y}$**:**	1	3	5	3	6	4	9

Word Problem with Least-Squares Approximation

An experiment measured the kinematic viscosity of oil (measured in centistokes) at different temperatures (measured in degrees Celsius). The observed data is:

$\boldsymbol{T}$**:**	20	40	60	80	100	120	140
$\boldsymbol{v}$**:**	135	145	155	163	172	180	185

1. Using Python and the least-squares method explained above, calculate the most-precise cubic model of this relationship.
2. Use your cubic model to predict the oil's viscosity at a temperature of 150° C.

END