6 Linear Least Squares Problems

Instead of solving $Ax = b$ we will solve

$$\min_x \|b - Ax\|_2.$$

when A has full column rank.

6 Linear Least Squares Problems

Instances of least squares arise in machine learning, computer vision, and computer graphics applications. It often arises in applications where data fitting is required: We search for an approximating function $v(t, x)$ depending on a continuous variable t that fits data pairs (t_i, b_i).

6 Linear Least Squares Problems

In section 6.1 we formulate an $n \times n$ system of linear equations, the *normal equations*, that yield the solution for the stated minimization problem. Solving the normal equations is a fast and straightforward approach. However, the resulting algorithm is not as stable as it can be in general. Orthogonal transformations provide a more stable way to proceed, and this is described in section 6.2. Algorithms to carry out QR decomposition are described in section 6.3.

6.1 Normal Equations

Geometry of dot product:

- $\vec{a} \cdot \vec{b} = a^T b = \|a\| \|b\| \cos\theta$.
- Find the length and angle between $a = (1, 4, 0, 2)$ and $b = (2, -2, 1, 3)$.
- a is perpendicular to b if $a^T b = 0$.
- least squares solution: residual orthogonal to column space, $0 = a^T \hat{r} = a^T (b - a\hat{x})$.

6.1 Normal Equations

Geometry of a projection as matrix multiplication (as a linear transformation).

- $Pb = a\frac{a^T b}{a^T a}$ projection of b onto line spanned by a.
- Find the projection of $b = (-2, -2, -2), b = (10, -5, 5)$, and $b = (1, 2, 3)$ onto the line spanned by $a = (1, 1, 1)$.
- Find the projection matrix of the projection onto the line spanned by $a = (1, 1, 1)$.
- What multiple of $a = (1, 1, 1)$ is closest to the point $(2, 4, 4)$?

6.1 Normal Equations

Use the Pythagorean theorem to verify the following are equivalent when a and b are vectors in $\mathbb{R}^n$ and x is a scalar.

- $\hat{x} = \min_x \|b - ax\|_2$.
- residual $r = b - A\hat{x}$ is perpendicular to a.
- The normal equation, $a^T ax = a^T b$, is satisfied.
- $ax = a\frac{a^T b}{a^T a} = Pb$.

6.1 Normal Equations

Show that the following are equivalent when A is an $m \times n$ matrix with $m > n$ and independent columns, and b is an m dimensional vector.

- $\min_x \|b - Ax\|_2$.
- residual $r = b - A\hat{x}$ is perpendicular to columns of A, $A^T r = 0$.
- The normal equation, $A^T A x = A^T b$, is satisfied.
- $Ax = A(A^T A)^{-1} A^T b$.
- If you read section 6.1 in our textbook you will see a cool way to prove this using multivariable calculus.

6.1 Normal Equations

Equations to know:

- Normal equations: $A^T A\hat{x} = A^T b$.
- Least Squares solution: $\hat{x} = (A^T A)^{-1} A^T b$.
- Projection: $Pb = A(A^T A)^{-1} A^T b$.
- Solve the normal equations to find the projection of $b = (4, 5, 6)$ onto the plane spanned by $a_1(1, 1, 0)$ and $a_2(2, 3, 0)$.

6.1 Normal Equations

Geometry of matrix multiplication as a linear transformation.

- Project the vector $b = (1, 2, 2)$ onto the line through $a = (1, 1, 1)$. Check that $\hat{r} = b - Pb$ is perpendicular to a.
- Find the best least squares solution $\hat{x}$ to $3x = 10, 4x = 5$. How is the residual minimized? Check that the residual $\hat{r} = b - A\hat{x}$ is perpendicular to the column of $A = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$.
- Solve $Ax = b$ by least squares when $A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}, b = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$.

  Verify that the residual $\hat{r} = b - A\hat{x}$ is perpendicular to the columns of A.

6.1 Algorithm: Least squares via Normal equations

- Form $B = A^T A$ and $y = A^T b$.
- Compute Cholesky Factorization: $B = GG^T$.
- Solve lower triangular system $Gz = y$ for z.
- Solve lower triangular system $G^T x = z$ for x.

6.1 Algorithm: Least squares via Normal equations

Example: $Ax = b$ when $A = \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ 1 & 1 \end{bmatrix}, b = \begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$.

- $B = A^T A = \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix}$ and $y = A^T b = \begin{bmatrix} 6 \\ 4 \end{bmatrix}$.
- $B = LU$.
- $B = LU = LD\tilde{U}$.
- $B = GG^T$ when $G = LD^{\frac{1}{2}} = \begin{bmatrix} \sqrt{3} & 0 \\ \frac{1}{\sqrt{3}} & \sqrt{\frac{8}{3}} \end{bmatrix}$.

6.1 Algorithm: Least squares via Normal equations

Example: Find the least squares solution to $Ax = b$ when
$$A = \begin{bmatrix} 1 & 0 & 1 \\ 2 & 3 & 5 \\ 5 & 3 & -2 \\ 3 & 5 & 4 \\ -1 & 6 & 3 \end{bmatrix}, b = \begin{bmatrix} 4 \\ -2 \\ 5 \\ -2 \\ 1 \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$ using Python in two ways: 1. Using Cholesky factorization and forward and back substitution, 2. Using numpy's built-in *lstsq* algorithm.

SOLUTION using Cholesky

```
import numpy as np
from numpy import linalg as LA
# set up problem
A = np.array([[1,0,1],
             [2,3,5],
              [5,3,-2],
              [3,5,4],
              [-1,6,3]], float)
b = np.array([4,-2,5,-2,1])
# Cholesky factorize and forward and back sub.
B = A.T@A
G = LA.cholesky(B)
c = LA.solve(G,A.T@b)
x = LA.solve(G.T,c)
print(x)
# check that residual is perp. to columns
r = b - A@x
print(A.T@r)
```

SOLUTION using lstsq algorithm

```
import numpy as np
# set up
A = np.array([[1,0,1],
             [2,3,5],
              [5,3,-2],
              [3,5,4],
              [-1,6,3]], float)
b = np.array([4,-2,5,-2,1])
# solve using lstsq
x = np.linalg.lstsq(A,b,rcond=None)[0]
# check that residual is perp. to columns
r = b - A@x
print(A.T@r)
```

6.1 Application: Data fitting

- Example: Find the linear polynomial $y = mx + b$ that best fits the data points $(0, 27), (1, 0), (2, 0)$, and $(3, 0)$ using least squares.
- Example: Find the quadratic polynomial $y = ax^2 + bx + c$ that best fits the data points $(0, 27), (1, 0), (2, 0)$, and $(3, 0)$ using least squares.

6.2 Orthogonal Transformations and QR

- The main drawback of the normal equations for solving least squares problems is accuracy in the presence of large condition numbers.
- Information may be lost when forming $A^T A$ when $\kappa(A)$ is large, since $\kappa(A^T A) \approx \kappa(A)^2$.

6.2 Orthogonal Transformations and QR

A matrix Q is orthogonal if its columns are orthonormal or if $Q^TQ = I$.

- Verify that $Q = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & 0 & \frac{-2}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \end{bmatrix}$ is orthogonal.
- Verify that $Q = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & 0 \\ \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{2}} \end{bmatrix}$ is orthogonal. Find the least squares solution to $Q\hat{x} = b$ when $b = (1, 0, -3)$.
- Verify that $A = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$ is not orthogonal.

6.2 Orthogonal Transformations and QR

Here are the standard methods for solving the linear least squares problem.

- Normal equations: fast, simple, intuitive, but less robust in ill-conditioned situations.
- QR decomposition: this is the "standard" way used in general-purpose software. It is often more computationally expensive than the normal equations approach but is more robust.
- SVD method: used mostly when A is rand deficient or nearly rank deficient (in which case the QR approach may not be sufficiently robust). The SVD approach is very robust but is significantly more expensive in general.

6.2 Orthogonal Transformations and QR

- Instead of solving the normal equations $A^TA = A^Tb$, first factor $A = QR$, when the columns of Q are orthnormal and R is upper triangular.
- After QR factorizing $Ax = QRx = b$ can be solved by a matrix multiplication $(Q^TQ)Rx = Q^Tb$ followed by back substitution on $Rx = Q^Tb$.n
- In the text two methods are discussed to factor $A = QR$: the Gram-Schmidt algorithm and the Householder reflectors method.

6.2 Orthogonal Transformations and QR

```
# ALGORITHM: Example 6.5 p. 154
import numpy as np
A = np.array([[1, 0],
              [1, 1],
              [1, 2]],float)
b = np.array([[0.1],
              [0.9],
              [2.0]],float)
Q, R = np.linalg.qr(A) # use numpy's qr program
x = np.linalg.solve(R, Q.T @ b) # use numpy's solve progran
print(x)
```

6.3 Householder and Gram-Schmidt

A particular robust QR factorization for the least squares problem is through Householder reflectors. First, however, we describe the Gram-Schmidt algorithm to factor $A = QR$ when Q is orthogonal and R is upper triangular. The Gram-Schmidt QR algorithm conceptually important and easier to grasp than Householder. Householder reflectors though more difficult to understand are used more frequently in professional software.

6.3 Householder and Gram-Schmidt

- Gram-Schmidt Algorithm to factor a matrix A with full column rank.
- Take one column v_i of A at a time starting with the leftmost first.
- Find $v_i^{\perp} = v_i - v_i^{\|}$ by subtracting the parallel part to the span of all previous (to the left) columns.
- Then normalize $u_i = \frac{v_i^{\perp}}{\|v_i\|}$.

6.3 Householder and Gram-Schmidt

The Gram-Schmidt formula, $v_i^{\perp} = v_i - v_i^{\|}$, is easy to remember: subtract from the old ith column v_i all the projections from the earlier (to the left) columns $v_1, v_2, \ldots,$ and v_{k-1}, one at a time.

- $v_i^{\perp} = v_i - (v_1^T v_i) v_1$
- $v_i^{\perp} = v_i - (v_2^T v_i) v_2, \ldots$
- $v_i^{\perp} = v_i - (v_{k-1}^T v_i) v_{k-1}$

6.3 Householder and Gram-Schmidt

In Python, using assignment, we can do all this in a loop (please try to write a loop to do Gram-Schmidt). HINT: do one subtraction each loop.

- $v_i = v_i - (v_1^T v_i)v_1$
- $v_i = v_i - (v_2^T v_i)v_2, \ldots$
- $v_i = v_i - (v_{k-1}^T v_i)v_{k-1}$

6.3 Householder and Gram-Schmidt

Here is how QR factorization works for a general 3×3 matrix:

$$A = \begin{bmatrix} \vec{a_1} & \vec{a_2} & \vec{a_3} \end{bmatrix} = \begin{bmatrix} \vec{q_1} & \vec{q_2} & \vec{q_3} \end{bmatrix} \begin{bmatrix} \vec{q_1}^T \vec{a_1} & \vec{q_1}^T \vec{a_2} & \vec{q_1}^T \vec{a_3} \\ 0 & \vec{q_2}^T \vec{a_2} & \vec{q_2}^T \vec{a_3} \\ 0 & 0 & \vec{q_3}^T \vec{a_3} \end{bmatrix} = QR.$$

6.3 Householder and Gram-Schmidt

Here is an example of how QR factorization works in a 3×3 matrix:

$$A = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & \sqrt{2} \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} \sqrt{2} & \frac{1}{\sqrt{2}} & \sqrt{2} \\ & \frac{1}{\sqrt{2}} & \sqrt{2} \\ & & 1 \end{bmatrix} = QR.$$

6.3 Example: Gram-Schmidt Algorithm

Find the QR factorization of $A = \begin{bmatrix} 1 & 1 \\ 1 & 9 \\ 1 & 9 \\ 1 & 1 \end{bmatrix}$ using Gram-Schmidt.

Use the QR factorization of A to solve $Ax = b$ when $b = \begin{bmatrix} 6 \\ 0 \\ 0 \\ 0 \end{bmatrix}$.

6.3 Example: Gram-Schmidt Algorithm

Find the QR factorization of $A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 9 & 2 \\ 1 & 9 & 2 \\ 1 & 1 & 2 \end{bmatrix}$ using Gram-Schmidt.

Use the QR factorization of A to solve $Ax = b$ when $b = \begin{bmatrix} 6 \\ 0 \\ 0 \\ 0 \end{bmatrix}$.

6.3 Householder Reflectors

- Let v and w be vectors with $\|v\| = \|w\|$ and let $a = v - w$. Then $H = I - 2P = I - 2\frac{aa^T}{a^Ta}$ is symmetric orthogonal and $Hw = v$.
- Find H when $v = [5, 0]$ and $w = [3, 4]$.
- $H = \begin{bmatrix} 0.6 & 0.8 \\ 0.8 & -0.6 \end{bmatrix}$
- Use this Householder reflector to find the QR factorization of $A = \begin{bmatrix} 3 & 1 \\ 4 & 3 \end{bmatrix}$.