Lecture 13: Least squares via QR-factorization

Amos Ron

University of Wisconsin - Madison

March 01, 2021

Outline

1. More on least squares
   - QR-factoring a rectangular matrix
   - Orthogonal transformation to least squares
2. Application: Least squares approximation
   - The approximation problem
   - An example

Outline

1. More on least squares
   - QR-factoring a rectangular matrix
   - Orthogonal transformation to least squares
2. Application: Least squares approximation
   - The approximation problem
   - An example

Blank page

Back to QR-factorization

Reviewing the factorization step

In the jth-step of the Householder algorithm for QR-factorization

- We create a Householder H_j, based on $x := A_{j-1}e_j$.
- We define $A_j := H_jA_{j-1}$.
- The first $(j-1)$st columns of A_{j-1} are preserved in A_j.
- A_je_j becomes a “good column".
- When computing A_j we need to update all the columns $k = j+1, \ldots, m$:

$$A_je_k := H_j(A_{j-1}e_k), \quad k = j+1, \ldots.$$

How to adapt if $A_{m\times n}$, $m > n$?

Back to QR-factorization

How to adapt if $A_{m\times n}$, $m>n$?

Adapting QR-factorization to rectangular A

There are only n steps, not $m-1$ steps.
In the jth-step of the Householder algorithm for QR-factorization

- We create a Householder H_j, based on $x := A_{j-1}e_j$.
- We define $A_j := H_j A_{j-1}$.
- The first $(j-1)$st columns of A_{j-1} are preserved in A_j.
- $A_j e_j$ becomes a “good column".
- When computing A_j we need to update all the columns $k=j+1,\ldots,n$:

$$A_j e_k := H_j(A_{j-1}e_k), \quad k=j+1,\ldots.$$

Back to QR-factorization

How to adapt if $A_{m\times n}$, $m > n$?

Adapting QR-factorization to rectangular A

There are only n steps, not $m-1$ steps.
In the jth-step of the Householder algorithm for QR-factorization

- We create a Householder H_j, based on $x := A_{j-1}e_j$.
- We define $A_j := H_j A_{j-1}$.
- The first $(j-1)$st columns of A_{j-1} are preserved in A_j.
- $A_j e_j$ becomes a “good column".
- When computing A_j we need to update all the columns $k = j+1, \ldots, n$:

$$A_j e_k := H_j(A_{j-1}e_k), \quad k = j+1, \ldots .$$

Short demo

Orthogonal transformation to least squares

We are given a least squares problem $Ax = b$, and multiply both sides by an orthgonal $Q_{m\times m}$:

$$QAx = Qb.$$

- If x^* is a solution of the new system then

$$||Ax^* - b||_2 = ||Q(Ax^* - b)||_2 = ||QAx^* - Qb||_2 \leq$$

$$||QAx - Qb||_2 = ||Q(Ax - b)||_2 = ||Ax - b||_2.$$

- So, x^* is also the least squares solution of the original problem!

Orthogonal transformation to least squares

Solving least squares via QR-factorization

- Step I: Factor $A = QR$, $Q_{m\times m}$, $R_{m\times n}$.
- Solve the least squares $Rx = Q'b$.
- We only still need to know how to solve least square with an upper triangular matrix.

Orthogonal transformation to least squares

Solving least squares via QR-factorization

- Step I: Factor $A = QR$, $Q_{m\times m}$, $R_{m\times n}$.
- Solve the least squares $Rx = Q'b$.
- We only still need to know how to solve least square with an upper triangular matrix.

So, we need to know how to solve

$$Rx = b,$$

with $R_{m\times n}$, $m > n$, upper triangular: $R(i,j) = 0, \quad i > j$. How to do that?

Orthogonal transformation to least squares

So, we need to know how to solve

$$Rx = b,$$

with $R_{m\times n}$, $m > n$, upper triangular: $R(i,j) = 0, \quad i > j$.
Discard all the equations $i = n+1, \ldots, m$. Solve the resulting square system.

Algorithm: Solving least square via QR-factorization

- QR-factor A.
- Remove from Q all columns $j = n+1, \ldots, m$:

$$Q_1 := Q(:, 1{:}n).$$

- Solve the square $n \times n$ upper triangular system

$$Q_1' A x = Q_1' b.$$

Theoretical explanation of what we did

Set

$$W = \mathrm{range}(\mathrm{A}).$$

Assume that the columns $w_1, \ldots, w_n$ of A are a basis for W.
We need $Ax^* - b \perp W$, i.e.,

$$(w_i, Ax^* - b) = 0, \quad i = 1, \ldots, n,$$

which is equivalent to the condition

$$A'(Ax^* - b) = 0.$$

Theoretical explanation of what we did

Set

$$W = \mathrm{range}(\mathrm{A}).$$

Assume that the columns $w_1, \ldots, w_n$ of A are a basis for W.
We need $Ax^* - b \perp W$, i.e.,

$$(w_i, Ax^* - b) = 0, \quad i = 1, \ldots, n,$$

which is equivalent to the condition

$$A'(Ax^* - b) = 0.$$

The 'only' thing the QR-factorization does is computing a new basis for W

$$q_1, \ldots, q_n, \quad q_i := Q(:, i).$$

So, we need

$$(q_i, Ax^* - b) = 0, \quad i = 1, \ldots, n,$$

Theoretical explanation of what we did

Set

$$W = \mathrm{range}(\mathrm{A}).$$

Assume that the columns $w_1, \ldots, w_n$ of A are a basis for W. We need $Ax^* - b \perp W$, i.e.,

$$(w_i, Ax^* - b) = 0, \quad i = 1, \ldots, n,$$

which is equivalent to the condition

$$A'(Ax^* - b) = 0.$$

The punch line: the condition number of the new equation

- $\mathrm{cond}_2(A'A) = \mathrm{cond}_2(A)^2$.
- $\mathrm{cond}_2(Q_1'A) = \mathrm{cond}_2(A)$.

Outline

1. More on least squares
   - QR-factoring a rectangular matrix
   - Orthogonal transformation to least squares
2. Application: Least squares approximation
   - The approximation problem
   - An example

The approximation problem

- f is some function defined on some interval $[a,b]$.
- We have input
$$Y = (Y(1), \ldots, Y(m))$$
on f, where
$$Y(i) \approx f(X(i)),$$
for some
$$X := (X(1), \ldots, X(m)) \subset [a,b].$$
- We assume that either m is large, or the $Y(i)$'s only approximate the $f(X(i))$'s.

The approximation problem

- f is some function defined on some interval $[a, b]$.
- We have input

$$Y = (Y(1), \ldots, Y(m))$$

on f, where

$$Y(i) \approx f(X(i)),$$

for some

$$X := (X(1), \ldots, X(m)) \subset [a, b].$$

- We assume that either m is large, or the $Y(i)$'s only approximate the $f(X(i))$'s.

The bias space G

We select a linear space G of functions defined on $[a, b]$ of small dimension n. Typically

$$n << m.$$

The approximation problem

The bias space G

We select a linear space G of functions defined on $[a, b]$ of small dimension n. Typically

$$n << m.$$

For example:

$$G = \Pi_{n-1} := \{ \text{ all polynomials of degree } < n\}.$$

Then we look for $g \in G$ that approximates the data we have on f.

We need to find $g \in G$ such that $f - g$ is “as small as possible": We can only measure

$$Y - g(X),$$

The approximation problem

We need to find $g \in G$ such that $f - g$ is “as small as possible":
We can only measure

$$Y - g(X),$$

with $g(X) := [g(X(1)), \ldots, g(X(m))]'$.

Least square approximation

Find $g^* \in G$ such that

$$||Y - g^*(X)||_2 \leq ||Y - g(X)||_2, \quad \forall g \in G.$$

The approximation problem

Least square approximation

Find $g^* \in G$ such that

$$||Y - g^*(X)||_2 \leq ||Y - g(X)||_2, \quad \forall g \in G.$$

Solution

- $A_{m \times n}$,
$$A(i,j) = g_j(X(i)).$$
- Solve least squares
$$Ac = Y.$$
- Then $g^* = \sum_{j=1}^{n} c(j) g_j$.

Approximation by linear polynomials

Choose $G = \Pi_1$. Then:

$$g_1(t) = 1, \; g_2(t) = t.$$

Then:

- $A(i,1) = 1$, $A(i,2) = X(i)$.
- $$A'A = \begin{pmatrix} m & \sum_{i=1}^m X(i) \\ \sum_{i=1}^m X(i) & \sum_{i=1}^m X(i)^2 \end{pmatrix}$$
- $$A'Y = \begin{pmatrix} \sum_{i=1}^m Y(i) \\ \sum_{i=1}^m X(i)Y(i) \end{pmatrix}$$