Lecture 6

CS 510

The Householder matrix and application in QR method

Householder Matrix

Let $w \in \mathbb{R}^n$ be a nonzero vector of norm $= 1$,

$$w^T w = 1$$

Define

$$H = I - 2ww^T \quad , \quad ww^T = \begin{bmatrix} \\ \\ \end{bmatrix} \begin{bmatrix} & & \end{bmatrix}$$

$$= \begin{bmatrix} n \times n \end{bmatrix}$$

$$H^T = I - 2ww^T = H$$

Also note

$$HH = (I - 2ww^T)(I - 2ww^T)$$

$$= I - 4ww^T + 4(ww^T)(ww^T)$$

But $(ww^T)ww^T = w(\underset{\overset{\shortparallel}{1}}{w^Tw})w^T = ww^T$

So $H \cdot H = H^2 = I$

Suppose x is a given non-zero vector. Is there a w s.t.

$$H = I - 2ww^T$$

satisfies

$$Hx = \pm \|x\|_2 e_1 \ ?$$

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Answer: $x = (x_1, \ldots, x_n)$ given

Let $v = x + \text{sign}(x_1) \|x\| e_1$,

where $\text{sign}(x_1) = \begin{cases} 1, & \text{if } x_1 \geq 0 \\ -1, & \text{if } x < 0 \end{cases}$

$$w = \frac{v}{\|v\|}$$

Then if

$$H = I - 2ww^T,$$

$$Hx = \text{sign}(x_1)\|x\| e_1$$

We can in general find w so that

$$H = I - 2ww^T$$

satisfies

$$HX = \mathrm{sign}(X_k)\,\|X\|\, e_k$$

$$e_k = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \end{pmatrix} \leftarrow \text{1 in } k\text{-th position}$$

Ex.

$$x = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \quad \|x\| = \sqrt{2}$$

$$v = x - \sqrt{2}\, e_1 = \begin{bmatrix} -1-\sqrt{2} \\ 1 \end{bmatrix}$$

$$\|v\|^2 = 1 + 2 + 2\sqrt{2} + 1 = (4 + 2\sqrt{2})$$

$$w = \frac{\begin{bmatrix} -1-\sqrt{2} \\ 1 \end{bmatrix}}{(4+2\sqrt{2})^{1/2}}$$

so

$$H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \frac{2}{(4+2\sqrt{2})} \begin{bmatrix} (-1-\sqrt{2})^2 & -1-\sqrt{2} \\ -1-\sqrt{2} & 1 \end{bmatrix}$$

$$Hx = \begin{bmatrix} -1 \\ 1 \end{bmatrix} - \frac{2}{4+2\sqrt{2}} \begin{bmatrix} (-1-\sqrt{2})(1+\sqrt{2}+1) \\ 1+\sqrt{2}+1 \end{bmatrix}$$

$$= \begin{bmatrix} x \\ 0 \end{bmatrix}$$

✓

QR Method

Suppose A is $n \times n$ real matrix

we can use Householder matrices

to get

$$A = QR$$

where $Q^{-1} = Q^T$, (Thus $Q^T Q = I$)

and R is upper triangular

$$A = [c_1, c_2, \dots, c_n], \quad c_i = i\text{-th column of } A$$

We can find H_1 so that

$$H_1 A = \begin{bmatrix} a'_{11} & a'_{12} & \cdots & a'_{1n} \\ 0 & a'_{21} & \cdots & a'_{2n} \\ \vdots & \vdots & & \vdots \\ 0 & a'_{n1} & \cdots & a'_{nn} \end{bmatrix}$$

Now we find H_2 so that

$$H_2 H_1 A =$$

$$\begin{bmatrix} a^1_{11} & a^1_{12} & \cdots & a^1_{1n} \\ 0 & a^2_{22} & \cdots & a^2_{2n} \\ \vdots & 0 & & \\ \vdots & \vdots & \cdots & \\ 0 & 0 & & \end{bmatrix}$$

In order to find such H_2 imagine

$$H_1 A = \left[\begin{array}{c|ccc} a_{11}^1 & a_{12}^1 & \cdots & a_{1n}^1 \\ \hline 0 & & & \\ \vdots & & A_2 & \\ 0 & & & \end{array}\right]$$

A_2 is $(n-1) \times (n-1)$ matrix.

Now we pick

$$H_2 = \left[\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline 0 & & & \\ \vdots & & \hat{H}_2 & \\ 0 & & & \end{array}\right]$$

In other words $H_2 H_1 A$ does not touch the first column of $H_1 A$.

This way in n steps we can compute

$$H_n H_{n-1} \cdots H_1 A = R$$

R upper triangular.

Let $Q^T = H_n H_{n-1} \cdots H_1$

Then

$$Q = H_1^T \cdots H_n^T = H_1 \cdots H_n$$

Also

$$QQ^T = H_1 \cdots \underbrace{H_n \times H_n}_{I} \cdots \times H_1$$

$$= \cdots I$$

so

$$A = QR$$

We can show the # of operations is $O(n^3)$ (like LU factorization)

Applications of QR factorization

1. Solving $Ax = b$

If we compute $QR = A$

Then we solve

$$QRx = b$$

Let

$$y = Rx$$

Then we get

$$Qy = b$$

$$\Rightarrow y = Q^T b$$

. Then we solve for x

in

$$RX = y$$

It is somewhat like LU factorization but more stable. However, it takes twice as many operations

2. Application in Computing all eigenvalues of an $n \times n$ matrix A.

It works as follows

Compute QR factorization of A.

Let $A_1 = A$. Compute $Q_1 R_1$ factorization of A_1.

Let $A_2 = R_1 Q_1$

Since $A_1 = Q_1 R_1 \Rightarrow Q_1^T A_1 = R_1$

So $A_2 = Q_1^T A_1 Q_1$

Next compute QR factorization of A_2

$$A_2 = Q_2 R_2$$

Let $A_3 = R_2 Q_2$

It can be shown that under some conditions

A_m converges to an upper triangular matrix whose diagonal entries are eigenvalues.

Convergence could be slow.

QR factorization is also valid if A is a matrix with complex entries.

In such case we define

$$H = I - 2ww^*$$

where w^* is conjugate transpose

In this general case

$$H^* = H$$

3. Finding all roots of a polynomial

suppose we want to compute roots of

$$p(z) = a_n z^n + a_{n-1} z^{n-1} + \dots + a_1 z + a_0$$

Consider $n \times n$ matrix

$$A = \begin{bmatrix} -\frac{a_{n-1}}{a_n}, & -\frac{a_{n-2}}{a_n}, & \cdots & -\frac{a_1}{a_n} & -\frac{a_0}{a_n} \\ 1 & 0 & \cdots & & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & & & & \\ 0 & \cdots & & 1 & 0 \end{bmatrix}$$

Let

$$v = \begin{bmatrix} \lambda^{n-1} \\ \lambda^{n-2} \\ \vdots \\ \lambda^{1} \\ \lambda^{0} \end{bmatrix}, \text{ where } \lambda \text{ is a root of } P(z)$$

We claim $Av = \lambda v$

$$A v = \begin{bmatrix} \dfrac{-a_{n-1}\lambda^{n-1} - \cdots - a_1\lambda - a_0}{a_n} \\ \lambda^{n-1} \\ \lambda^{n-2} \\ \\ \\ \lambda \end{bmatrix}$$

But $$\lambda v = \begin{bmatrix} \lambda^n \\ \lambda^{n-1} \\ \vdots \\ \lambda \end{bmatrix},$$ & these are the same.

We can thus apply QR method to find all roots of $p(z)$.

However this is not necessarily the preferred method.

Power Method to compute dominant eigenvalue of a matrix.

Dominant means eigenvalue with largest modulus.

Power Method:

Pick $v_0 \in \mathbb{R}^n$

Inductively define

$$\mu_k = v_k^T A v_k$$

$$v_{k+1} = Av_k / \|Av_k\|.$$

Suppose the eigenvalues of A are

$\lambda_1, \cdots, \lambda_n$

& suppose

$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$

That is to say the

We claim:

μ_k converges to λ_1.

pf:

Since v_k is obtained from v_0 by repeated multiplication by A, k times

$$v_1 = \frac{Av_0}{\|Av_0\|} = A\frac{v_0}{\|Av_0\|}$$

$$= A\frac{v_0}{\|Av_0\|} = A(\alpha_1$$

suppose $v_k = \alpha_1 u_1 \lambda^k +$

$$v_k = \alpha_1 \lambda_1^k u_1 + \alpha_2 \lambda_2^k u_2 + \cdots + \alpha_n \lambda_n^k u_n$$

for some constants

$\alpha_1, \ldots, \alpha_n$

Then

$$A v_k = \alpha_1 \lambda_1^{k+1} u_1 + \alpha_2 \lambda_2^{k+1} + \cdots + \alpha_n \lambda_n^{k+1} u_n$$

So

$$v_k^T v_k = \alpha_1^2 \lambda_1^{2k} + \alpha_2^2 \lambda_2^{2k} + \dots \alpha_n^2 \lambda_n^{2k}$$

$$= 1$$

$$v_k^T A v_k = \alpha_1^2 \lambda_1^{2k+1} + \alpha_2^2 \lambda_2^{2k+1} + \dots$$

So $$\frac{v_k^T A v_k}{v_k^T v_k} = \frac{\lambda_1^{2k+1}}{\lambda_1^{2k}} \left[\frac{\alpha_1^2 + \alpha_2^2 \left(\frac{\lambda_2}{\lambda_1}\right)^{2k+1} \dots}{\alpha_1^2 + \alpha_2 \left(\frac{\lambda_2}{\lambda}\right)^{2k} + \dots} \right]$$

Now

$$\left[\cdot \;\; \cdot \cdot \cdot \right] \longrightarrow 1$$

$$\frac{\lambda_1^{2k+1}}{\lambda_1^{2k}} = \lambda_1$$

So $\mu_k \longrightarrow \lambda_1$

Triangle Algorithm

Given a set of point

$S = \{v_1, \dots, v_n\} \subseteq \mathbb{R}^m$

& $p \in \mathbb{R}^m$

we want to know if $p \in \text{Conv}(S)$

i.e. if

$$p = \alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n$$

where $\alpha_1 + \alpha_2 + \dots + \alpha_n = 1$, $\alpha_i \geq 0$

If we let

$$A = \begin{bmatrix} v_1 & \cdots & v_n \end{bmatrix}, \quad m \times n$$

$$\alpha = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}$$

then we want to see

if

$$A\alpha = P, \quad e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

$$e^T \alpha = 1$$

$$\alpha \geq 0$$

Given $p' \in \text{Conv}(S)$, say

$$p' = \alpha_1 v_1 + \dots + \alpha_n v_n, \quad \sum \alpha_i = 1, \; \alpha_i \geq 0$$

we say $v_j \in S$ is a pivot

if

$$\|p' - v_j\| \geq \|p - v_j\|$$

Equivalently if

$$(p - p')^T v_j \geq \frac{1}{2}\left(\|p\|^2 - \|p'\|^2\right).$$

We say p' is a witness if no pivot exists, i.e.

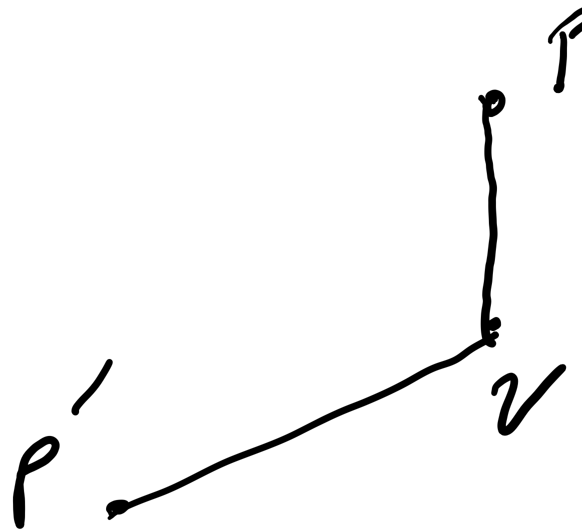
$$(p-p')^T v_i < \frac{1}{2}\left(\|p\| - \|p'\|^2\right)$$

for $i=1,\dots,n$

If v is a pivot then we can get closer to p.

we project p on $p'v$.

We could also have this situation:

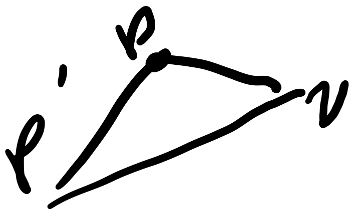


In this case we simply take $p'' = v$.

P'' can be computed as:

Let $$\alpha_x = \frac{(P-P')^T (V_j - P')}{\|V_j - P'\|^2}$$

If we have the first situations



then

$$P'' = (1-\alpha_x)P' + \alpha_x V = \sum_{i=1}^{n} \alpha_i' V_i$$

$$\alpha_j' = (1-\alpha_x)\alpha_j + \alpha_x, \quad \underset{i \neq j}{\alpha_i'} = (1-\alpha_x)\alpha_i,$$

So given $p' = \sum \alpha_i v_i$ we can compute a representation of p'' in terms of v_i's

$$p'' = \sum \alpha_i' v_i$$

Triangle Algorithm

Input S, p, ϵ

step 0. Let $p' = v = \text{argmin}\{\|p - v_i\|\}$

i.e. the closest v_i to p.

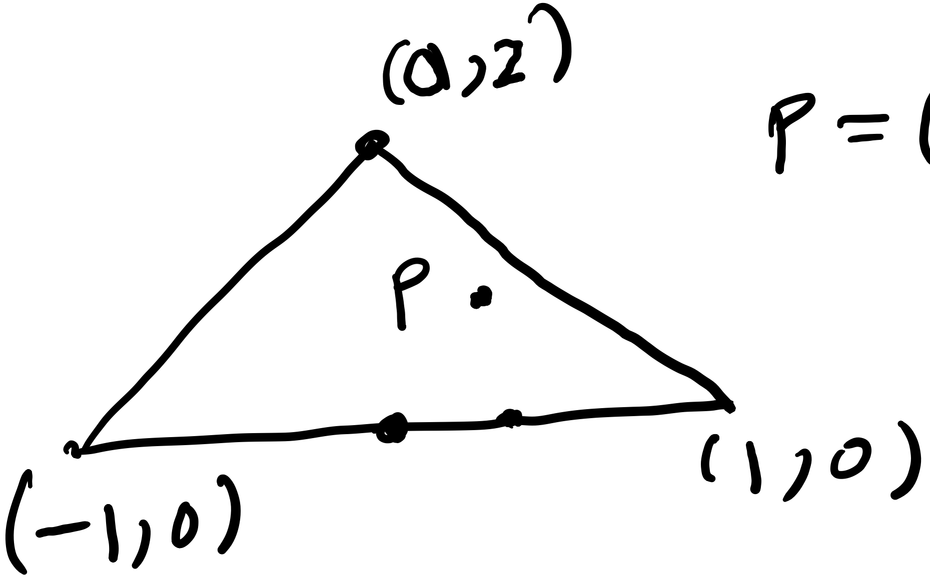
step 1. If $\|p - p'\| \leq \epsilon \|p - v\|$, output p' as an approximate solution, stop.

Step 2. Replace v with a pivot v_j, compute P''.

Replace P' with P'' and Goto step 1.

Try one or two iterations of the algorithm both geometrically and algebraically.

Ex.



$P = (\frac{1}{2}, 1)$

Solving $Ax = b$

Assume A is $n \times n$, invertible.

Thm. Suppose $x_* = A^{-1}b \geq 0$

There exist $\alpha > 0$, $x \geq 0$

such that

$$Ax = \alpha b$$

$$\Sigma x_i + \alpha = 1 \quad , \quad x_i \geq 0, \alpha \geq 0$$

In other words

$$0 = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \text{ is in } \mathrm{Conv}(A, -b).$$

We can solve $Ax = b$
with the assumption that $x_* = A^{-1}b \geq 0$
via the triangle algorithm:

In each iteration we have a pair (x, α) such that

$$p' = Ax - \alpha b \ , \quad \sum x_i + \alpha = 1$$

$$x_i \ , \alpha \geq 0$$

we check if

$$\left\| A \frac{x}{\alpha} - b \right\| \leq \epsilon \max\{\|a_1\|, \dots, \|a_n\|, \|b\|\}.$$

a_i = i-th column of A

If so we stop.

otherwise, we iterate.

What if $A^{-1}b \ngeq 0$?

If $Ax = b$

then $Ax + w = b + w$

Suppose we choose $w = tAe$

where $e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$, t a scalar.

Then $A(x + te) = b + tAe$

$\exists\, t_*$ such that $x + t_* e \geq 0$

and t_x is the smallest such t such that $x + te \geq 0$.

If we knew such t_x, we add $t_x Ae$ to both sides and do as before.

Since we don't know t_x we try to pick a t & incrementally change it if necessary.

Incremental Triangle Alg.

Assume for a given $t_0 \geq 0$ we have tried to use triangle algorithm to find x_0 s.t.

$$\|A(x_0 - t_0 e) - b\| \leq \varepsilon \max\{\|a_1\|, \dots, \|a_n\|, \|b\|\}$$

where a_i = i-th Column of A

If this is possible we are done.

otherwise, the simplest strategy is to increase t_0 to t_1 & repeat.

There are better ideas. (Later)