9. QR factorization and Algorithm

why factorizing a matrix?

- Solve $Ax=b$ efficiently
- Retrieve matrix parameters/properties
  λ, $\|\cdot\|$, det, inertia, etc
- So far: $A=LU$

the QR factorization

- $A=QR$, where $Q^{-1}=Q^{*}$ $(Q^{-1}=Q^{T})$ and R is upper triangular with positive (non-neg.) numbers over its diagonal (rect A: Q is square, R is trapezoidal).
- Useful to calculate $\lambda(A)$ reliably via the QR algorithm;
- Solve $Ax=b$ and LS reliably, via a numerically stable algorithm;
- Used for reliable estimation algs: QR-RLS AF, Kalman, etc

9.1. Orthonormal Bases from LI bases

In the generic case, generate a orthonormal set of vecs from an existing LI set of vecs.

$$\underset{\text{full rank}}{A_{M\times N}} \longrightarrow \underset{\text{full rank}}{Q_{M\times N}}$$

$$[a_1\, a_2 \cdots a_N] \rightarrow [q_1\, q_2 \cdots q_N]$$

$M > N$ $\quad Q^*Q = I_N$ (cols $\perp$)

tall

$M < N$ $\quad QQ^* = I_M$ (rows $\perp$)

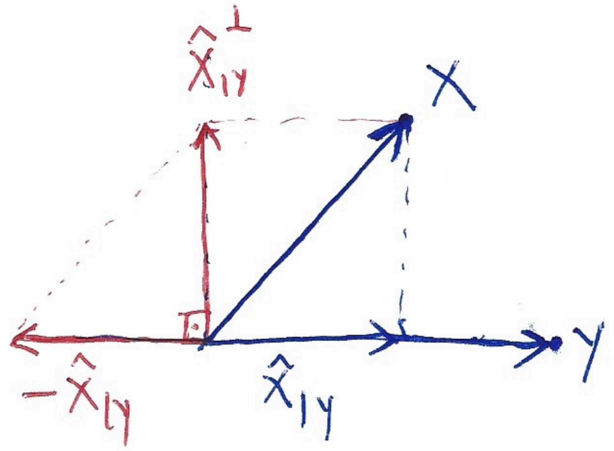
wide / fat

$M = N$ $\quad Q^*Q = QQ^* = I$

square Q is unitary

$Q^{-1} = Q^*$ $\quad (Q^{-1} = Q^T$ real case$)$

How to proceed? Recall a projection (ortho) of vec x onto vec y



$$\hat{x}_{|y} \triangleq \langle x, y\rangle \|y\|^{-2} y$$

$$x - \hat{x}_{|y} = \hat{x}^{\perp}_{|y} \quad \text{or}$$

$$\boxed{x = \hat{x}_{|y} + \hat{x}^{\perp}_{|y}}$$ orthog. decomp.

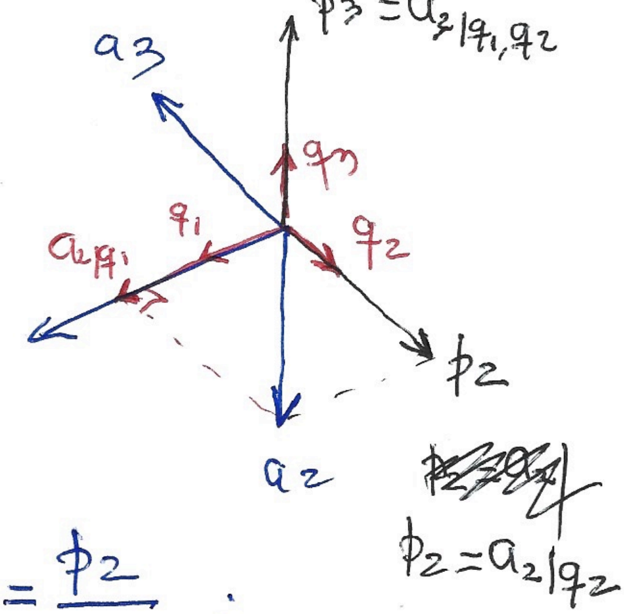
$\|\hat{x}_{|y}\| = \|x\| \cos\theta$ and points along y.

this procedure can be extended sequentially to an arbitrary set of LI vecs.

the Gram-Schmidt procedure

Sequential orthonormalization via $\perp$ projections.

Example: three LI vecs a_1, a_2 and a_3.



$$p_1 = a_1 ; \quad q_1 = \frac{p_1}{\|p_1\|} = \frac{p_1}{\|a_1\|} ;$$

$$p_2 = a_2 - a_{2|q_1}$$

$$p_2 = a_2 - \underbrace{\langle a_2, q_1 \rangle \overbrace{\|q_1\|^{-2}}^{1} q_1}_{proj(a_2;\, q_1)} ; \quad q_2 = \frac{p_2}{\|p_2\|} ;$$

because $q_1 \perp q_2$

$$p_3 = a_3 - a_{3|q_1,q_2} = a_3 - (a_{3|q_1} + a_{3|q_2})$$

$$= a_3 - \underbrace{(\langle a_3, q_1 \rangle q_1 + \langle a_3, q_2 \rangle q_2)}_{proj(a_3;\, q_1, q_2)} ; \quad q_3 = \frac{p_3}{\|p_3\|}$$

$$\underset{\text{LI}}{[a_1\ a_2\ a_3]} \xrightarrow{G.S.} \underset{\text{Orthonormal}}{[q_1\ q_2\ q_3]}$$

for k LI vecs

$$p_k = a_k - proj(a_k; q_1, q_2, \dots, q_{k-1})$$

$$= a_k - \sum_{l=1}^{k-1} \langle a_k, q_l \rangle q_l ; \quad q_k = \frac{p_k}{\|p_k\|}$$

9.2. the QR theorem

A full col/row rank

<u>thm</u>: Let $A \in \mathbb{F}^{M\times N}$. then there exist a unitary matrix $Q \in \mathbb{F}^{M\times M}$ and an upper-Δ matrix $R \in \mathbb{F}^{M\times N}$ with $r_{ii} = [R]_{ii} \in \mathbb{R}^+$, so that

<u>Full QR</u>: $A = QR$, $R = \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix}$, $Q^{-1} = Q^*$

$$Q = [Q_1\ Q_2].$$

<u>Short QR</u>:

$$A = Q_1 \bar{R}$$

(we do not need Q_2 to form A)

there are cases in which we may need the full unitary matrix $Q = [Q_1\ Q_2]$. How to find Q_2? Extend Q_1 by completing with an extra set of orthonormal vecs to Q_1

$$[Q_1 \,\vdots\, Q_2]$$

→ Generate random vecs $\{a_k\}$ then proceed with GS using Q_1, $N-1 > k > M$

→ See LAUB thm 10.3 to find Q_2 via Householder matrices

Proof for QR thm: manipulate GS into matrix form. Matrix A is reconstructed sequentially by "inverting" the GS method. Let's consider the 3×3 case to illustrate.

$$q_1 = \frac{p_1}{\|p_1\|} = \frac{a_1}{\|a_1\|} \Rightarrow \boxed{a_1 = \|a_1\| q_1}$$

$$q_2 = \frac{p_2}{\|p_2\|} = \frac{a_2 - \langle a_2, q_1\rangle}{\|a_2 - \langle a_2, q_1\rangle\|} \Rightarrow a_2 = \|a_2 - \langle a_2, q_1\rangle q_1\| q_2 + \langle a_2, q_1\rangle q_1$$

or $$\boxed{a_2 = \langle a_2, q_1\rangle q_1 + \|a_2 - \langle a_2, q_1\rangle q_1\| q_2}$$

$$q_3 = \frac{p_3}{\|q_3\|} = \frac{a_3 - \langle a_3, q_1\rangle q_1 - \langle a_3, q_2\rangle q_2}{\| \cdot \|} \Rightarrow$$

$$\boxed{a_3 = \langle a_3, q_1\rangle q_1 + \langle a_3, q_2\rangle q_2 + \|a_3 - \langle a_3, q_1\rangle q_1 - \langle a_3, q_2\rangle q_2\|}$$

$$[a_1\ a_2\ a_3] = [q_1\ q_2\ q_3][r_1\ r_2\ r_3] = QR$$

$$R = \begin{bmatrix} \|a_1\| & \langle a_2, q_1\rangle & \langle a_3, q_1\rangle \\ 0 & \|a_2 - \langle a_2, q_1\rangle\| & \langle a_3, q_2\rangle \\ 0 & 0 & \|a_3 - \langle a_3, q_1\rangle q_1 - \langle a_3, q_2\rangle q_2\| \end{bmatrix}$$

9.3. QR Implementation

GS motivates the construction of QR decomp., but, in its original form, is numerically unreliable. there are some stable implementations:

1) Householder reflections: via elementary unitary reflector matrices H_k.

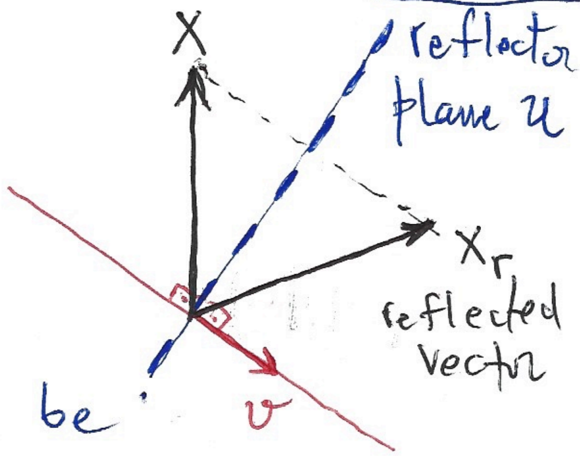
$$\underbrace{H_{N-1}\cdots H_2 H_1}_{Q^*} A = R$$

Each matrix H_k annihilates an entire col below the $a_{kk} = [A]_{kk}$ pivot.

Consider vector $z^T = [\otimes \; \otimes \; \otimes \cdots \otimes]$

$\otimes$ = any number

$$H\begin{bmatrix}\otimes\\ \otimes\\ \vdots\\ \otimes\end{bmatrix} \longrightarrow \begin{bmatrix}\otimes\\ 0\\ 0\\ \vdots\\ 0\end{bmatrix}$$

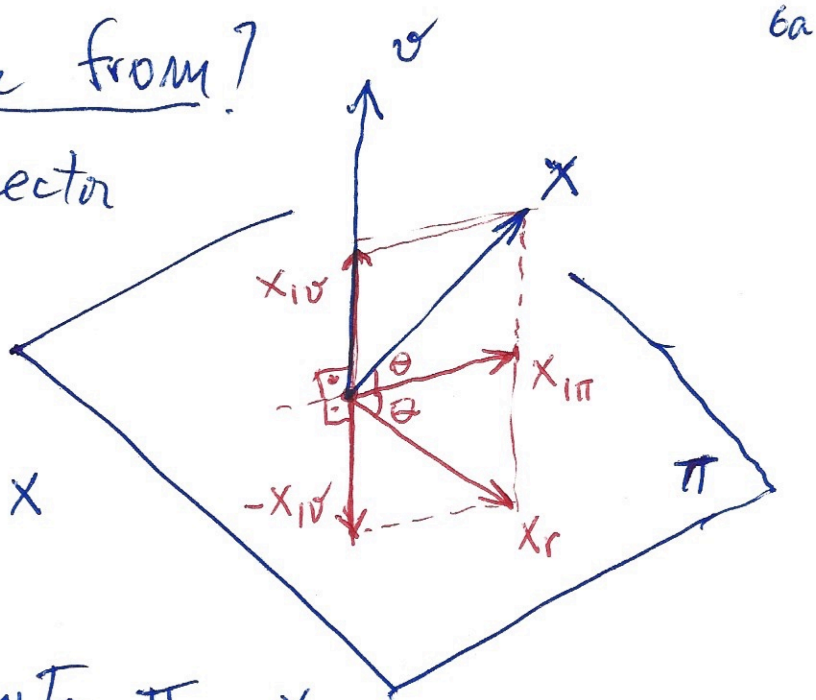


the trick is to impose X_r to be aligned with a canonical basis vector, then build H so that $Hx = c \cdot e_1$.

v: vector normal to plane u

the general form: $\boxed{H = I - \frac{2vv^*}{v^*v}}$ or $\boxed{H = I - 2vv^*}$ for $\|v\| = 1$

Where does H come from?



v is the orthogonal vector to the reflection hyperplane π.

We want to reflect x onto X_r across π.

1) First project X onto π: $X_{|\pi}$

$$X_{|\pi} = X - X_{|v} \qquad (1)$$

recall that the projection of x onto v is

$$X_{|v} = X - \langle x, v\rangle \|v\|^{-2} v.$$

2) then, from the picture above

$$X_r = X_{|\pi} + (-X_{|v}) = X_{|\pi} - X_{|v} \qquad (2)$$

(1) into (2): $X_r = (X - X_{|v}) - X_{|v} = X - 2X_{|v}$

or $\boxed{X_r = X - 2\langle x, v\rangle \|v\|^{-2} v}$

3) Considering the usual inner product for real numbers (for simplicity)

$$X_r = x - \frac{2x^T v}{v^T v} v = x - \frac{2\overbrace{(v^T x)}^{\text{scalar}} v}{v^T v} = x - 2\frac{v(v^T x)}{v^T v}$$

$$X_r = \left(I - \frac{2vv^T}{v^T v}\right)X = Hx, \quad \text{where} \quad \boxed{H \triangleq I - \frac{2vv^T}{v^T v}} \text{ or } \boxed{H \triangleq I - \frac{2vv^*}{v^* v}}$$

H is orthogonal/unitary and symmetric. (Hermitian)

Testing for orthog/unitarity:

$$H^T H = \left(I - \frac{2vv^T}{v^Tv}\right)\left(I - \frac{2vv^T}{v^Tv}\right) = H^2 \quad (H^T = H)$$

$$= I - \frac{2vv^T}{v^Tv} - \frac{2vv^T}{v^Tv} + 4\frac{(v\cancel{v^T v}v^T)}{(v^T\cancel{v})^2}$$

$$= I - \frac{4vv^T}{v^Tv} + \frac{4vv^T}{v^Tv} = I \quad \therefore H^T H = H^2 = I$$

or $\boxed{H^{-1} = H^T}$.

<u>Finding vector $v \perp \Pi$</u>

We assume v will be used in a triangularization process of some matrix A. (Say $x = a_1$ (first col of A)) In the $\triangledown$-process, we want to reflect x onto one of the canonical vectors, say $e_1^T = [1\ 0 \cdots 0]$. ~~In other words $x = \alpha e_1$~~ thus $v \in \text{span}(x, e_1)$. For instance, $\boxed{v = x + \alpha e_1}$ (3). In other words

$$x_r = x - 2\frac{v^Tx}{v^Tv}x = x - 2\frac{v^Tx}{v^Tv}(x + \alpha e_1)$$

$$= x - 2\frac{v^Tx}{v^Tv}x - \overbrace{\frac{2\alpha v^Tx}{v^Tv}}^{\beta} e_1 = \left(1 - 2\frac{v^Tx}{v^Tv}\right)x - \beta e_1$$

$x_r = \left(\frac{\alpha^2 - \|x\|_2^2}{\|v\|^2}\right)x - \beta e_1$. Then, for x_r aligned with e_1 we must have $\alpha^2 = \|x\|_2^2$, or $\boxed{\alpha = \pm \|x\|_2}$

Then $$\boxed{v = x \pm \|x\|_2 e_1}. \qquad (4)$$

the sign in ~~Eq.~~ Eq. 4 may be selected to guarantee a non-negative "diagonal" for R, as in the QR factorization; or to improve numerical accuracy in finite precision operations.

<u>Examples</u>: say $x^T = [1\ 1\ 1\ 1]$.

$v_- \triangleq x - \|x\|_2 e_1 = [-1\ 1\ 1\ 1]^T$, $H_- \triangleq I - \dfrac{2 v_- v_-^T}{\|v_-\|^2}$.

$H_- x = [2\ 0\ 0\ 0]^T = x_r$.

$v_+ \triangleq x + \|x\|_2 e_1 = [3\ 1\ 1\ 1]^T$, $H_+ \triangleq I - \dfrac{2 v_+ v_+^T}{\|v_+\|^2}$.

$H_+ x = [-2\ 0\ 0\ 0]^T = x_r$.

Also: say $x^T = [-1\ 1\ 1\ 1]$.

$v_- = [-3\ 1\ 1\ 1]^T$, $H_- x = [2\ 0\ 0\ 0]^T = x_r$.

$v_+ = [1\ 1\ 1\ 1]$, $H_+ x = [-2\ 0\ 0\ 0]^T = x_r$

That is, Householder transformations are unitary (orthogonal) can be used for obtaining the QR decomposition but we must be careful to guarantee $[R]_{ii} \geq 0$

Triangularizing $A_{3\times3}$ via Householder

$$A = [a_1 \; a_2 \; a_3] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$P_k \triangleq I - \frac{2 v_k v_k^T}{v_k^T v_k}.$$

$e_1^T = [1 \; 0 \; 0]$

$$H_1 = P_1 = I - \frac{2 v_1 v_1^T}{v_1^T v_1}, \qquad v_1 = a_1 \pm \|a_1\|_2 e_1$$

$$H_1 A = \begin{bmatrix} \pm\|a_1\|_2 & a_{12}^{(1)} & a_{13}^{(1)} \\ 0 & \boxed{a_{22}^{(1)}} & \boxed{a_{23}^{(1)}} \\ 0 & \underbrace{\boxed{a_{32}^{(1)}}}_{a_2^{(1)}} & \underbrace{\boxed{a_{33}^{(1)}}}_{a_3^{(1)}} \end{bmatrix} \triangleq A^{(1)} = \left[\begin{array}{c|cc} \pm\|a_1\|_2 & a_{12}^{(1)} & a_{13}^{(1)} \\ \hline 0 & a_2^{(1)} & a_3^{(1)} \\ 0 & & \end{array}\right]$$

$$H_2 = \left[\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & & \\ 0 & \multicolumn{2}{c}{P_2} \end{array}\right], \qquad P_2 = I_{3\times3} - \frac{2 v_2 v_2^T}{v_2^T v_2}, \quad v_2 \text{ is } 2\times1$$

$$v_2 = a_2^{(1)} \pm \|a_2^{(1)}\|_2 \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

$$H_2 A^{(1)} = \begin{bmatrix} \pm\|a_1\|_2 & a_{12}^{(1)} & a_{13}^{(1)} \\ 0 & \pm\|a_2^{(1)}\|_2 & a_{23}^{(2)} \\ 0 & 0 & a_{33}^{(2)} \end{bmatrix} \triangleq A^{(2)} \equiv R \text{ upper triangular.}$$

Note that it is not a problem that $a_{33}^{(2)}$ is not directly related to the norm of col 3 of $A^{(2)}$.

2) Givens Rotations: triangularize A via a sequence of unitary rotations, anihilating one element at a time. Good for sparse matrices (does not destroy sparsity; Householder might do it). Also good for parallel implementations.

$$G(i,k) = \begin{array}{c} \\ \\ (G(i,k,\theta)) \; i \\ \\ k \\ \\ \\ \end{array} \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & & \ddots & & & & & \vdots \\ 0 & \cdots & & c & \cdots & s & \cdots & 0 \\ \vdots & & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & & -s & \cdots & c & \cdots & \vdots \\ \vdots & & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \end{array}$$

$$\qquad\qquad\quad i \qquad\quad k$$

For $X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_i \\ \vdots \\ x_k \\ \vdots \\ x_N \end{bmatrix}$

$c \triangleq \cos\theta$, $s \triangleq \sin\theta$

$$G X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \pm(x_i^2 + x_k^2)^{1/2} \\ \vdots \\ 0 \\ \vdots \\ x_N \end{bmatrix} \begin{array}{c} \\ \\ \\ i \\ \\ k \\ \\ \\ \end{array}$$

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} x_i \\ x_k \end{bmatrix} = \begin{bmatrix} c x_i + s x_k \\ -s x_i + c x_k \end{bmatrix}$$

such that $c^2 + s^2 = 1$ (unitary).

If $c = \dfrac{x_i}{(x_i^2 + x_k^2)^{1/2}}$

and $s = \dfrac{-x_k}{(x_i^2 + x_k^2)^{1/2}}$

then $-s x_i + c x_k = 0$.

Forming the product GX we have:

$$GX = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ c x_i + s x_k \\ \vdots \\ -s x_i + c x_k \\ \vdots \\ x_N \end{bmatrix} \begin{matrix} \\ \\ \\ y_i \\ \\ y_k \\ \\ \\ \end{matrix} \triangleq Y.$$

We must have $\|Y\|_2 = \|X\|_2$, since G is unitary / orthogonal.

$y_i = c x_i + s x_k$.

$y_k = -s x_i + c x_k$.

Moved i, k terms to the end

$$\|X\|_2^2 = \sum_{\ell=1}^{N} x_\ell^2 = \underbrace{x_1^2 + x_2^2 + \cdots + x_N^2}\, + x_i^2 + x_k^2 \qquad y_\ell = x_\ell,\ \ell \neq i,k.$$

$$- \quad \|Y\|_2^2 = \sum_{\ell=1}^{N} y_\ell^2 = y_1^2 + y_2^2 + \cdots + y_N^2 + y_i^2 + y_k^2$$

$$0 = x_i^2 + x_k^2 - y_i^2 - y_k^2$$

$$y_i^2 + y_k^2 = x_i^2 + x_k^2 \iff \|X\|_2^2 = \|Y\|_2^2$$

$$c^2 x_i^2 + \cancel{2cs\,x_i x_k} + s^2 x_k^2 + s^2 x_i^2 - \cancel{2sc\,x_i x_k} + c^2 x_k^2 = x_i^2 + x_k^2$$

$(c^2 + s^2) x_i^2 + (s^2 + c^2) x_k^2 = x_i^2 + x_k^2$. Selecting $c^2 + s^2 = s^2 + c^2 = 1$ assures that $\|X\|_2^2 = \|Y\|_2^2$, or $\|X\|_2 = \|Y\|_2$.

$$\boxed{c^2 + s^2 = 1}$$

if we choose $c = \dfrac{x_i}{(x_i^2 + x_k^2)^{1/2}}$, $s = \dfrac{-x_k}{(x_i^2 + x_k^2)^{1/2}}$,

then $Y = GX$ becomes

$$
\begin{array}{cc}
GX = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \pm (x_i^2 + x_k^2)^{1/2} \\ \vdots \\ 0 \\ \vdots \\ x_N \end{bmatrix} &
\begin{array}{c} \\ \\ \\ i \\ \\ k \\ \\ \\ \end{array}
\end{array}
$$

Givens matrix is a rank-2 modification of the identity matrix; ~~skew symmetric~~

It is unitary/orthogonal. We do not need to introduce the notion of an angle Θ, but it is useful to indicate the "net rotation". Rotation may be counter clockwise, as it was defined here, or clockwise if we swap the signs for s and $-s$ in the G matrix.

9.4. The QR algorithm

Find $\lambda(A)$ in a stable/robust manner.
The QR alg is a recursive similarity transformations method.

Note that, in $A=QR$, $\mathrm{diag}(R) = r_{ii} \in \mathbb{R}^+$ and $r_{ii} = \lambda(R)$. ~~Also~~, But $\lambda(R) \neq \lambda(A)$ in the general case. Also $\lambda(B) = \lambda(A)$, if $B = PAP^{-1}$. We want $\lambda(A)$. Also $\lambda(C) = \lambda(A)$ if $C = P^{-1}AP$.

1) $A_0 = A$; $A_0 \rightarrow Q_1 R_1$ (QR decomposition)

$$\lambda(Q_1^{-1} A_0 Q_1) = \lambda(Q_1^* A_0 Q_1) = \lambda(A_0) = \lambda(A).$$

$$Q_1^{-1} A_0 Q_1 = Q_1^* A_0 Q_1 = Q_1^*(Q_1 R_1) Q_1 = R_1 Q_1.$$

then: $\lambda(A) = \lambda(A_0) = \lambda(Q_1^* A_0 Q_1) = \lambda(R_1 Q_1)$.

2) $A_1 \triangleq R_1 Q_1$; $A_1 \rightarrow Q_2 R_2$ (QR decomp).

$$\lambda(A) = \lambda(A_0) = \lambda(R_1 Q_1) = \lambda(A_1) = \lambda(Q_2^{-1} A_1 Q_2)$$
$$= \lambda(Q_2^{-1}(Q_2 R_2) Q_2) = \lambda(R_2 Q_2).$$

Trick: generate next A_{k+1} by efficiently inverse multiplying $R_{k+1} Q_{k+1}$

3) $A_2 = R_2 Q_2$; $A_2 \rightarrow Q_3 R_3$ ...

Recursion
1) $A_k \rightarrow Q_{k+1} R_{k+1}$
2) $A_{k+1} = R_{k+1} Q_{k+1}$
3) Repeat

For k large enough, $A_k \longrightarrow \nabla_k$, whose diagonal tends to $\lambda(A)$.

9.5. QR Least-Squares

Consider a lin system $Ax = b$.

If $\exists A^{-1}$, than a possible solution is $x = A^{-1}b$, and $\boxed{\kappa(A) \triangleq \|A\| \|A^{-1}\|}$ provides an upper bound on how accurately this system can be solved.

Now, if $A_{M \times N}$ is rectangular but, say, full col rank, an approximate least-squares solution ~~is~~ follows from

$$A^* A \hat{x}_{LS} = A^* b \Rightarrow \boxed{\hat{x}_{LS} = (A^* A)^{-1} A^* b}$$

Now, let's check what happens to $\kappa(A^* A)$.

$$\kappa(A^* A) = \|A^* A\| \|(A^* A)^{-1}\| \le \|A^*\| \|A\| \|A^{-1} A^{*^{-1}}\|$$

$$\le \|A^*\| \|A\| \|A^{-1}\| \|A^{*^{-1}}\| = \|A\| \|A\| \|A^{-1}\| \|(A^{-1})^*\|$$

$$= \|A\| \|A^{-1}\| \|A\| \|A^{-1}\| = \kappa^2(A)$$

BACK

or $\kappa(A^* A) \le \kappa^2(A)$. this is bad news for numerical stability in finite precision!

A better approach is to explore QR decomposition: $A = QR = [Q_1 \ Q_2]\begin{bmatrix}\bar{R}\\0\end{bmatrix}$.
Or, even better, the short QR decomposition:

$$A = Q_1 \bar{R}$$, where $\bar{R}$ is square and non-singular for full col-rank A, and Q_1 has orthonormal columns.

$$A^* A \hat{x}_{LS} = A^* b$$

$$(Q_1\bar{R})^*(Q_1\bar{R})\hat{x}_{LS} = (Q_1\bar{R})^* b$$

$$\bar{R}^* Q_1^* Q_1 \bar{R} \hat{x}_{LS} = \bar{R}^* Q_1^* b$$

$$\bar{R}^* \bar{R} \hat{x}_{LS} = \bar{R}^* Q_1^* b \Rightarrow (\bar{R}^*)^{-1} \Rightarrow \boxed{\bar{R}\hat{x}_{LS} = Q_1^* b}$$

steps

1) Decompose $A = Q_1\bar{R}$ (or $Q_1^* A = \bar{R}$)
2) Form $d \triangleq Q_1^* b$
3) Find $\hat{x}_{LS}$ via back substitution in $\bar{R}\hat{x}_{LS} = d$.