# Time-Varying Systems and Computations

## Unit 4.1

Klaus Diepold

WS 2024

# QR Decomposition

- For a square, full rank matrix $$T = QR$$
- Orthonormal matrix $$Q^T Q = Q Q^T = 1$$
- Upper triangular matrix

$$R = \begin{bmatrix} * & * & * & \dots & * \\ & * & * & \dots & * \\ & & \ddots & & \vdots \\ & & & \ddots & \vdots \\ & & & & * \end{bmatrix}$$

# QR Decomposition

- For “tall“ martrices with full column rank $T \in \mathcal{R}^{m \times n} \quad m \geq n$

$$T = \underbrace{\left[\begin{array}{c|c} Q_1 & Q_2 \end{array}\right]}_{Q} \cdot \underbrace{\left[\begin{array}{c} R \\ \hline 0 \end{array}\right]}_{\tilde{R}} = Q_1 \cdot R,$$

Compute orthogonal Basis for column space

$$\mathrm{im}(T) = \mathrm{im}(Q_1)$$
$$\mathrm{im}(T)^{\perp} = \mathrm{im}(Q_2)$$

$$R = \begin{bmatrix} * & * & * & \dots & * \\ & * & * & \dots & * \\ & & \ddots & & \vdots \\ & & & \ddots & \vdots \\ & & & & * \end{bmatrix} \qquad R \in \mathcal{R}^{n \times n}$$

# Solving Least Squares with QR

- Linear system of equations

$$Tu = y \qquad\qquad T \in \mathcal{R}^{m \times n}$$

- Different types of problems

$$\begin{cases} m > n & \text{overdetermined} \\ n = m & \text{square} \\ m < n & \text{underdetermined.} \end{cases}$$

Focus right now (overdetermined, square)

- QR decomposition

$$\left[\ Q_1 \mid Q_2\ \right] \cdot \left[\frac{R}{0}\right] u = b$$

# Solving Least Squares with QR

- QR Decomposition

$$\left[\begin{array}{c|c} Q_1 & Q_2 \end{array}\right] \cdot \left[\begin{array}{c} R \\ \hline 0 \end{array}\right] u = b$$

Matlab call

$[Q_1, R] = qr(T, 0)$

- Pre-multiply both sides with $Q^T$ produces

$$\left[\begin{array}{c} Ru \\ \hline 0 \end{array}\right] = \left[\begin{array}{c} Q_1^T \\ \hline Q_2^T \end{array}\right] b := \left[\begin{array}{c} \beta_1 \\ \hline \beta_2 \end{array}\right]$$

- Solve for vector $u$ and determine error $e$

$$u = R^{-1}\beta_1 \qquad e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} \beta_1 - Ru \\ \beta_2 \end{bmatrix} \left( = \begin{bmatrix} 0 \\ \beta_2 \end{bmatrix} \right)$$

# Solving Least Squares with Normal Equation

- Linear system of equations $Tu = y \qquad T \in \mathcal{R}^{m\times n}$
- Build Normal Equation $T^T T u = T^T y$
- Compute Cholesky Factor $T^T T = R^T R$

  Same $R$ as in QRD

  $$T^T T = (R^T \underbrace{Q^T)(Q}_{1} R) = R^T R$$
- Plug in Cholesky Factor $R$ $\quad R^T \underbrace{(Ru)}_{x} = \underbrace{T^T y}_{b}$
- Solve for $u$ using two triangular systems $\quad R^T x = b \qquad x = Ru$

# Squaring the condition number

- Condition number of original problem ➡ $\text{cond}\ (T) := \frac{\sigma_1}{\sigma_n}$ ☺

$$T = U\Sigma V^T = U \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix} V^T$$

- Condition number of normal equation ➡ $\text{cond}(T^T T) = \frac{\sigma_1^2}{\sigma_n^2}$ ☹

$$T^T T = V\Sigma \underbrace{U^T U}_{1} \Sigma V^T = \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_n^2 \end{bmatrix}$$

$$= V\Sigma^2 V^T = V \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_n^2 \end{bmatrix} V^T$$

# Algorithms for Computing QR

- (Modified) Gram-Schmidt Orthogonalzation

  Numerically not reliable, never used in practical applications
- Householder QR

  Standard algorithm for computing QR, de-facto standard in all numerical packages
- Givens (Jacobi) QR

  Algorithm of choice for dedicated hardware implementation and parallel computing
- Generalized Rotations

  Competition for Householder for sitations where rotations are requird

  Mostly unknown

# QR Decomposition

- QR provides  alternative method to solve linear least squares problems
- QR does not square the condition number → more robust
- R from QR is identical to R from Cholesky factorization

  (check with Matlab)
- QR decomposition approch can be adjusted to handle underdetermined and rank deficient problems
- QR allows for straight forward updating mechanisms