

Scientific Computing: An Introductory Survey

Chapter 3 – Linear Least Squares

Prof. Michael T. Heath

Department of Computer Science
University of Illinois at Urbana-Champaign

Copyright © 2002. Reproduction permitted
for noncommercial, educational use only.



Method of Least Squares

- Measurement errors are inevitable in observational and experimental sciences
- Errors can be smoothed out by averaging over many cases, i.e., taking more measurements than are strictly necessary to determine parameters of system
- Resulting system is *overdetermined*, so usually there is no exact solution
- In effect, higher dimensional data are projected into lower dimensional space to suppress irrelevant detail
- Such projection is most conveniently accomplished by method of *least squares*



Data Fitting

- Given m data points (t_i, y_i) , find n -vector \mathbf{x} of parameters that gives "best fit" to model function $f(t, \mathbf{x})$,

$$\min_{\mathbf{x}} \sum_{i=1}^m (y_i - f(t_i, \mathbf{x}))^2$$

- Problem is *linear* if function f is linear in components of \mathbf{x} ,

$$f(t, \mathbf{x}) = x_1\phi_1(t) + x_2\phi_2(t) + \dots + x_n\phi_n(t)$$

where functions ϕ_j depend only on t

- Problem can be written in matrix form as $\mathbf{Ax} \cong \mathbf{b}$, with $a_{ij} = \phi_j(t_i)$ and $b_i = y_i$



Example: Data Fitting

- Fitting quadratic polynomial to five data points gives linear least squares problem

$$\mathbf{Ax} = \begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ 1 & t_3 & t_3^2 \\ 1 & t_4 & t_4^2 \\ 1 & t_5 & t_5^2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \cong \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \mathbf{b}$$

- Matrix whose columns (or rows) are successive powers of independent variable is called *Vandermonde matrix*



Outline

- 1 Least Squares Data Fitting
- 2 Existence, Uniqueness, and Conditioning
- 3 Solving Linear Least Squares Problems



Linear Least Squares

- For linear problems, we obtain *overdetermined* linear system $\mathbf{Ax} = \mathbf{b}$, with $m \times n$ matrix \mathbf{A} , $m > n$
- System is better written $\mathbf{Ax} \cong \mathbf{b}$, since equality is usually not exactly satisfiable when $m > n$
- Least squares solution \mathbf{x} minimizes squared Euclidean norm of residual vector $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$,

$$\min_{\mathbf{x}} \|\mathbf{r}\|_2^2 = \min_{\mathbf{x}} \|\mathbf{b} - \mathbf{Ax}\|_2^2$$



Data Fitting

- Polynomial fitting

$$f(t, \mathbf{x}) = x_1 + x_2t + x_3t^2 + \dots + x_nt^{n-1}$$

is linear, since polynomial linear in coefficients, though nonlinear in independent variable t

- Fitting sum of exponentials

$$f(t, \mathbf{x}) = x_1e^{x_2t} + \dots + x_{n-1}e^{x_{n-1}t}$$

is example of nonlinear problem

- For now, we will consider only linear least squares problems



Example, continued

- For data

$$\begin{matrix} t & -1.0 & -0.5 & 0.0 & 0.5 & 1.0 \\ y & 1.0 & 0.5 & 0.0 & 0.5 & 2.0 \end{matrix}$$

overdetermined 5×3 linear system is

$$\mathbf{Ax} = \begin{bmatrix} 1 & -1.0 & 1.0 \\ 1 & -0.5 & 0.25 \\ 1 & 0.0 & 0.0 \\ 1 & 0.5 & 0.25 \\ 1 & 1.0 & 1.0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \cong \begin{bmatrix} 1.0 \\ 0.5 \\ 0.0 \\ 0.5 \\ 2.0 \end{bmatrix} = \mathbf{b}$$

- Solution, which we will see later how to compute, is

$$\mathbf{x} = [0.086 \quad 0.40 \quad 1.4]^T$$

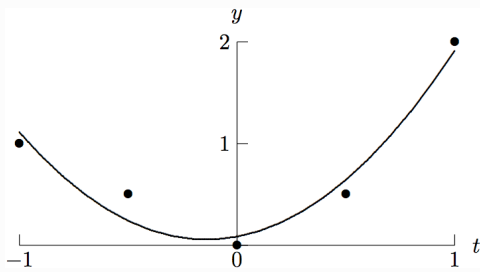
so approximating polynomial is

$$p(t) = 0.086 + 0.4t + 1.4t^2$$



Example, continued

- Resulting curve and original data points are shown in graph



< interactive example >



Normal Equations

- To minimize squared Euclidean norm of residual vector

$$\begin{aligned} \|r\|_2^2 &= r^T r = (b - Ax)^T (b - Ax) \\ &= b^T b - 2x^T A^T b + x^T A^T A x \end{aligned}$$

take derivative with respect to x and set it to 0,

$$2A^T A x - 2A^T b = 0$$

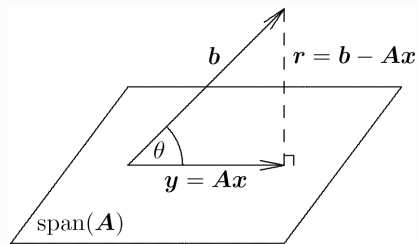
which reduces to $n \times n$ linear system of *normal equations*

$$A^T A x = A^T b$$



Orthogonality, continued

- Geometric relationships among b , r , and $\text{span}(A)$ are shown in diagram



Pseudoinverse and Condition Number

- Nonsquare $m \times n$ matrix A has no inverse in usual sense
- If $\text{rank}(A) = n$, *pseudoinverse* is defined by

$$A^+ = (A^T A)^{-1} A^T$$

and condition number by

$$\text{cond}(A) = \|A\|_2 \cdot \|A^+\|_2$$

- By convention, $\text{cond}(A) = \infty$ if $\text{rank}(A) < n$
- Just as condition number of square matrix measures closeness to singularity, condition number of rectangular matrix measures closeness to rank deficiency
- Least squares solution of $Ax \cong b$ is given by $x = A^+ b$



Existence and Uniqueness

- Linear least squares problem $Ax \cong b$ *always* has solution
- Solution is *unique* if, and only if, columns of A are *linearly independent*, i.e., $\text{rank}(A) = n$, where A is $m \times n$
- If $\text{rank}(A) < n$, then A is *rank-deficient*, and solution of linear least squares problem is not unique
- For now, we assume A has full column rank n



Orthogonality

- Vectors v_1 and v_2 are *orthogonal* if their inner product is zero, $v_1^T v_2 = 0$
- Space spanned by columns of $m \times n$ matrix A , $\text{span}(A) = \{Ax : x \in \mathbb{R}^n\}$, is of dimension at most n
- If $m > n$, b generally does not lie in $\text{span}(A)$, so there is no exact solution to $Ax = b$
- Vector $y = Ax$ in $\text{span}(A)$ closest to b in 2-norm occurs when residual $r = b - Ax$ is *orthogonal* to $\text{span}(A)$,

$$0 = A^T r = A^T (b - Ax)$$

again giving system of *normal equations*

$$A^T A x = A^T b$$



Orthogonal Projectors

- Matrix P is *orthogonal projector* if it is *idempotent* ($P^2 = P$) and *symmetric* ($P^T = P$)
- Orthogonal projector onto orthogonal complement $\text{span}(P)^\perp$ is given by $P_\perp = I - P$
- For any vector v ,

$$v = (P + (I - P)) v = Pv + P_\perp v$$

- For least squares problem $Ax \cong b$, if $\text{rank}(A) = n$, then

$$P = A(A^T A)^{-1} A^T$$

is orthogonal projector onto $\text{span}(A)$, and

$$b = Pb + P_\perp b = Ax + (b - Ax) = y + r$$



Sensitivity and Conditioning

- Sensitivity of least squares solution to $Ax \cong b$ depends on b as well as A
- Define angle θ between b and $y = Ax$ by

$$\cos(\theta) = \frac{\|y\|_2}{\|b\|_2} = \frac{\|Ax\|_2}{\|b\|_2}$$

- Bound on perturbation Δx in solution x due to perturbation Δb in b is given by

$$\frac{\|\Delta x\|_2}{\|x\|_2} \leq \text{cond}(A) \frac{1}{\cos(\theta)} \frac{\|\Delta b\|_2}{\|b\|_2}$$



Sensitivity and Conditioning, contnued

- Similarly, for perturbation E in matrix A ,

$$\frac{\|\Delta x\|_2}{\|x\|_2} \lesssim ([\text{cond}(A)]^2 \tan(\theta) + \text{cond}(A)) \frac{\|E\|_2}{\|A\|_2}$$

- Condition number of least squares solution is about $\text{cond}(A)$ if residual is small, but can be squared or arbitrarily worse for large residual



Example: Normal Equations Method

- For polynomial data-fitting example given previously, normal equations method gives

$$A^T A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ -1.0 & -0.5 & 0.0 & 0.5 & 1.0 \\ 1.0 & 0.25 & 0.0 & 0.25 & 1.0 \end{bmatrix} \begin{bmatrix} 1 & -1.0 & 1.0 \\ 1 & -0.5 & 0.25 \\ 1 & 0.0 & 0.0 \\ 1 & 0.5 & 0.25 \\ 1 & 1.0 & 1.0 \end{bmatrix}$$

$$= \begin{bmatrix} 5.0 & 0.0 & 2.5 \\ 0.0 & 2.5 & 0.0 \\ 2.5 & 0.0 & 2.125 \end{bmatrix},$$

$$A^T b = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ -1.0 & -0.5 & 0.0 & 0.5 & 1.0 \\ 1.0 & 0.25 & 0.0 & 0.25 & 1.0 \end{bmatrix} \begin{bmatrix} 1.0 \\ 0.5 \\ 0.0 \\ 0.5 \\ 2.0 \end{bmatrix} = \begin{bmatrix} 4.0 \\ 1.0 \\ 3.25 \end{bmatrix}$$



Shortcomings of Normal Equations

- Information can be lost in forming $A^T A$ and $A^T b$
- For example, take

$$A = \begin{bmatrix} 1 & 1 \\ \epsilon & 0 \\ 0 & \epsilon \end{bmatrix}$$

where ϵ is positive number smaller than $\sqrt{\epsilon_{\text{mach}}}$

- Then in floating-point arithmetic

$$A^T A = \begin{bmatrix} 1 + \epsilon^2 & 1 \\ 1 & 1 + \epsilon^2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

which is singular

- Sensitivity of solution is also worsened, since

$$\text{cond}(A^T A) = [\text{cond}(A)]^2$$



Augmented System Method, continued

- Introducing scaling parameter α gives system

$$\begin{bmatrix} \alpha I & A \\ A^T & O \end{bmatrix} \begin{bmatrix} r/\alpha \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}$$

which allows control over relative weights of two subsystems in choosing pivots

- Reasonable rule of thumb is to take

$$\alpha = \max_{i,j} |a_{ij}| / 1000$$

- Augmented system is sometimes useful, but is far from ideal in work and storage required



Normal Equations Method

- If $m \times n$ matrix A has rank n , then symmetric $n \times n$ matrix $A^T A$ is positive definite, so its Cholesky factorization

$$A^T A = LL^T$$

can be used to obtain solution x to system of normal equations

$$A^T A x = A^T b$$

which has same solution as linear least squares problem $Ax \cong b$

- Normal equations method involves transformations

rectangular \rightarrow square \rightarrow triangular



Example, continued

- Cholesky factorization of symmetric positive definite matrix $A^T A$ gives

$$A^T A = \begin{bmatrix} 5.0 & 0.0 & 2.5 \\ 0.0 & 2.5 & 0.0 \\ 2.5 & 0.0 & 2.125 \end{bmatrix} = \begin{bmatrix} 2.236 & 0 & 0 \\ 0 & 1.581 & 0 \\ 1.118 & 0 & 0.935 \end{bmatrix} \begin{bmatrix} 2.236 & 0 & 1.118 \\ 0 & 1.581 & 0 \\ 0 & 0 & 0.935 \end{bmatrix} = LL^T$$

- Solving lower triangular system $Lz = A^T b$ by forward-substitution gives $z = [1.789 \ 0.632 \ 1.336]^T$
- Solving upper triangular system $L^T x = z$ by back-substitution gives $x = [0.086 \ 0.400 \ 1.429]^T$



Augmented System Method

- Definition of residual together with orthogonality requirement give $(m+n) \times (m+n)$ augmented system

$$\begin{bmatrix} I & A \\ A^T & O \end{bmatrix} \begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}$$

- Augmented system is not positive definite, is larger than original system, and requires storing two copies of A
- But it allows greater freedom in choosing pivots in computing LDL^T or LU factorization



Orthogonal Transformations

- We seek alternative method that avoids numerical difficulties of normal equations
- We need numerically robust transformation that produces easier problem without changing solution
- What kind of transformation leaves least squares solution unchanged?
- Square matrix Q is *orthogonal* if $Q^T Q = I$
- Multiplication of vector by orthogonal matrix preserves Euclidean norm

$$\|Qv\|_2^2 = (Qv)^T Qv = v^T Q^T Qv = v^T v = \|v\|_2^2$$

- Thus, multiplying both sides of least squares problem by orthogonal matrix does not change its solution



Triangular Least Squares Problems

- As with square linear systems, suitable target in simplifying least squares problems is triangular form
- Upper triangular overdetermined ($m > n$) least squares problem has form

$$\begin{bmatrix} R \\ O \end{bmatrix} x \cong \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

where R is $n \times n$ upper triangular and b is partitioned similarly

- Residual is

$$\|r\|_2^2 = \|b_1 - Rx\|_2^2 + \|b_2\|_2^2$$



QR Factorization

- Given $m \times n$ matrix A , with $m > n$, we seek $m \times m$ orthogonal matrix Q such that

$$A = Q \begin{bmatrix} R \\ O \end{bmatrix}$$

where R is $n \times n$ and upper triangular

- Linear least squares problem $Ax \cong b$ is then transformed into triangular least squares problem

$$Q^T Ax = \begin{bmatrix} R \\ O \end{bmatrix} x \cong \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = Q^T b$$

which has same solution, since

$$\|r\|_2^2 = \|b - Ax\|_2^2 = \|b - Q \begin{bmatrix} R \\ O \end{bmatrix} x\|_2^2 = \|Q^T b - \begin{bmatrix} R \\ O \end{bmatrix} x\|_2^2$$



Computing QR Factorization

- To compute QR factorization of $m \times n$ matrix A , with $m > n$, we annihilate subdiagonal entries of successive columns of A , eventually reaching upper triangular form
- Similar to LU factorization by Gaussian elimination, but use orthogonal transformations instead of elementary elimination matrices
- Possible methods include
 - Householder transformations
 - Givens rotations
 - Gram-Schmidt orthogonalization



Example: Householder Transformation

- If $a = \begin{bmatrix} 2 & 1 & 2 \end{bmatrix}^T$, then we take

$$v = a - \alpha e_1 = \begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix} - \alpha \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix} - \begin{bmatrix} \alpha \\ 0 \\ 0 \end{bmatrix}$$

where $\alpha = \pm \|a\|_2 = \pm 3$

- Since a_1 is positive, we choose negative sign for α to avoid cancellation, so $v = \begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix} - \begin{bmatrix} -3 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 5 \\ 1 \\ 2 \end{bmatrix}$

- To confirm that transformation works,

$$Ha = a - 2 \frac{v^T a}{v^T v} v = \begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix} - 2 \frac{15}{30} \begin{bmatrix} 5 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} -3 \\ 0 \\ 0 \end{bmatrix}$$

< interactive example >



Triangular Least Squares Problems, continued

- We have no control over second term, $\|b_2\|_2^2$, but first term becomes zero if x satisfies $n \times n$ triangular system

$$Rx = b_1$$

which can be solved by back-substitution

- Resulting x is least squares solution, and minimum sum of squares is

$$\|r\|_2^2 = \|b_2\|_2^2$$

- So our strategy is to transform general least squares problem to triangular form using orthogonal transformation so that least squares solution is preserved



Orthogonal Bases

- If we partition $m \times m$ orthogonal matrix $Q = [Q_1 \ Q_2]$, where Q_1 is $m \times n$, then

$$A = Q \begin{bmatrix} R \\ O \end{bmatrix} = [Q_1 \ Q_2] \begin{bmatrix} R \\ O \end{bmatrix} = Q_1 R$$

is called *reduced* QR factorization of A

- Columns of Q_1 are orthonormal basis for $\text{span}(A)$, and columns of Q_2 are orthonormal basis for $\text{span}(A)^\perp$
- $Q_1 Q_1^T$ is orthogonal projector onto $\text{span}(A)$
- Solution to least squares problem $Ax \cong b$ is given by solution to square system

$$Q_1^T Ax = Rx = c_1 = Q_1^T b$$



Householder Transformations

- Householder transformation* has form

$$H = I - 2 \frac{vv^T}{v^T v}$$

for nonzero vector v

- H is orthogonal and symmetric: $H = H^T = H^{-1}$
- Given vector a , we want to choose v so that

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

- Substituting into formula for H , we can take

$$v = a - \alpha e_1$$

and $\alpha = \pm \|a\|_2$, with sign chosen to avoid cancellation



Householder QR Factorization

- To compute QR factorization of A , use Householder transformations to annihilate subdiagonal entries of each successive column
- Each Householder transformation is applied to entire matrix, but does not affect prior columns, so zeros are preserved
- In applying Householder transformation H to arbitrary vector u ,

$$Hu = \left(I - 2 \frac{vv^T}{v^T v} \right) u = u - \left(2 \frac{v^T u}{v^T v} \right) v$$

which is much cheaper than general matrix-vector multiplication and requires only vector v , not full matrix H



Householder QR Factorization, continued

- Process just described produces factorization

$$H_n \cdots H_1 A = \begin{bmatrix} R \\ O \end{bmatrix}$$

where R is $n \times n$ and upper triangular

- If $Q = H_1 \cdots H_n$, then $A = Q \begin{bmatrix} R \\ O \end{bmatrix}$
- To preserve solution of linear least squares problem, right-hand side b is transformed by same sequence of Householder transformations
- Then solve triangular least squares problem $\begin{bmatrix} R \\ O \end{bmatrix} x \approx Q^T b$

Michael T. Heath

Scientific Computing

33 / 61

Example: Householder QR Factorization

- For polynomial data-fitting example given previously, with

$$A = \begin{bmatrix} 1 & -1.0 & 1.0 \\ 1 & -0.5 & 0.25 \\ 1 & 0.0 & 0.0 \\ 1 & 0.5 & 0.25 \\ 1 & 1.0 & 1.0 \end{bmatrix}, \quad b = \begin{bmatrix} 1.0 \\ 0.5 \\ 0.0 \\ 0.5 \\ 2.0 \end{bmatrix}$$

- Householder vector v_1 for annihilating subdiagonal entries of first column of A is

$$v_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} -2.236 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 3.236 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Michael T. Heath

Scientific Computing

35 / 61

Example, continued

- Applying resulting Householder transformation H_2 yields

$$H_2 H_1 A = \begin{bmatrix} -2.236 & 0 & -1.118 \\ 0 & 1.581 & 0 \\ 0 & 0 & -0.725 \\ 0 & 0 & -0.589 \\ 0 & 0 & 0.047 \end{bmatrix}, \quad H_2 H_1 b = \begin{bmatrix} -1.789 \\ 0.632 \\ -1.035 \\ -0.816 \\ 0.404 \end{bmatrix}$$

- Householder vector v_3 for annihilating subdiagonal entries of third column of $H_2 H_1 A$ is

$$v_3 = \begin{bmatrix} 0 \\ 0 \\ -0.725 \\ -0.589 \\ 0.047 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0.935 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -1.660 \\ -0.589 \\ 0.047 \end{bmatrix}$$

Michael T. Heath

Scientific Computing

37 / 61

Givens Rotations

- Givens rotations** introduce zeros one at a time
- Given vector $[a_1 \ a_2]^T$, choose scalars c and s so that

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \alpha \\ 0 \end{bmatrix}$$

with $c^2 + s^2 = 1$, or equivalently, $\alpha = \sqrt{a_1^2 + a_2^2}$

- Previous equation can be rewritten

$$\begin{bmatrix} a_1 & a_2 \\ a_2 & -a_1 \end{bmatrix} \begin{bmatrix} c \\ s \end{bmatrix} = \begin{bmatrix} \alpha \\ 0 \end{bmatrix}$$

- Gaussian elimination yields triangular system

$$\begin{bmatrix} a_1 & a_2 \\ 0 & -a_1 - a_2^2/a_1 \end{bmatrix} \begin{bmatrix} c \\ s \end{bmatrix} = \begin{bmatrix} \alpha \\ -\alpha a_2/a_1 \end{bmatrix}$$

Michael T. Heath

Scientific Computing

39 / 61

Householder QR Factorization, continued

- For solving linear least squares problem, product Q of Householder transformations need not be formed explicitly
- R can be stored in upper triangle of array initially containing A
- Householder vectors v can be stored in (now zero) lower triangular portion of A (almost)
- Householder transformations most easily applied in this form anyway

Michael T. Heath

Scientific Computing

34 / 61

Example, continued

- Applying resulting Householder transformation H_1 yields transformed matrix and right-hand side

$$H_1 A = \begin{bmatrix} -2.236 & 0 & -1.118 \\ 0 & -0.191 & -0.405 \\ 0 & 0.309 & -0.655 \\ 0 & 0.809 & -0.405 \\ 0 & 1.309 & 0.345 \end{bmatrix}, \quad H_1 b = \begin{bmatrix} -1.789 \\ -0.362 \\ -0.862 \\ -0.362 \\ 1.138 \end{bmatrix}$$

- Householder vector v_2 for annihilating subdiagonal entries of second column of $H_1 A$ is

$$v_2 = \begin{bmatrix} 0 \\ -0.191 \\ 0.309 \\ 0.809 \\ 1.309 \end{bmatrix} - \begin{bmatrix} 0 \\ 1.581 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -1.772 \\ 0.309 \\ 0.809 \\ 1.309 \end{bmatrix}$$

Michael T. Heath

Scientific Computing

36 / 61

Example, continued

- Applying resulting Householder transformation H_3 yields

$$H_3 H_2 H_1 A = \begin{bmatrix} -2.236 & 0 & -1.118 \\ 0 & 1.581 & 0 \\ 0 & 0 & 0.935 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad H_3 H_2 H_1 b = \begin{bmatrix} -1.789 \\ 0.632 \\ 1.336 \\ 0.026 \\ 0.337 \end{bmatrix}$$

- Now solve upper triangular system $Rx = c_1$ by back-substitution to obtain $x = [0.086 \ 0.400 \ 1.429]^T$

< interactive example >

Michael T. Heath

Scientific Computing

38 / 61

Givens Rotations, continued

- Back-substitution then gives

$$s = \frac{\alpha a_2}{a_1^2 + a_2^2} \quad \text{and} \quad c = \frac{\alpha a_1}{a_1^2 + a_2^2}$$

- Finally, $c^2 + s^2 = 1$, or $\alpha = \sqrt{a_1^2 + a_2^2}$, implies

$$c = \frac{a_1}{\sqrt{a_1^2 + a_2^2}} \quad \text{and} \quad s = \frac{a_2}{\sqrt{a_1^2 + a_2^2}}$$

Michael T. Heath

Scientific Computing

40 / 61

Example: Givens Rotation

- Let $\mathbf{a} = [4 \ 3]^T$
 - To annihilate second entry we compute cosine and sine
- $$c = \frac{a_1}{\sqrt{a_1^2 + a_2^2}} = \frac{4}{5} = 0.8 \quad \text{and} \quad s = \frac{a_2}{\sqrt{a_1^2 + a_2^2}} = \frac{3}{5} = 0.6$$

- Rotation is then given by

$$\mathbf{G} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} = \begin{bmatrix} 0.8 & 0.6 \\ -0.6 & 0.8 \end{bmatrix}$$

- To confirm that rotation works,

$$\mathbf{G}\mathbf{a} = \begin{bmatrix} 0.8 & 0.6 \\ -0.6 & 0.8 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \end{bmatrix} = \begin{bmatrix} 5 \\ 0 \end{bmatrix}$$



Givens QR Factorization

- Straightforward implementation of Givens method requires about 50% more work than Householder method, and also requires more storage, since each rotation requires two numbers, c and s , to define it
- These disadvantages can be overcome, but requires more complicated implementation
- Givens can be advantageous for computing QR factorization when many entries of matrix are already zero, since those annihilations can then be skipped

< interactive example >



Gram-Schmidt Orthogonalization

- Process can be extended to any number of vectors $\mathbf{a}_1, \dots, \mathbf{a}_k$, orthogonalizing each successive vector against all preceding ones, giving *classical Gram-Schmidt* procedure

```

for k = 1 to n
    q_k = a_k
    for j = 1 to k - 1
        r_jk = q_j^T a_k
        q_k = q_k - r_jk q_j
    end
    r_kk = ||q_k||_2
    q_k = q_k / r_kk
end
    
```

- Resulting \mathbf{q}_k and r_{jk} form reduced QR factorization of \mathbf{A}



Modified Gram-Schmidt QR Factorization

- Modified Gram-Schmidt algorithm

```

for k = 1 to n
    r_kk = ||a_k||_2
    q_k = a_k / r_kk
    for j = k + 1 to n
        r_kj = q_k^T a_j
        a_j = a_j - r_kj q_k
    end
end
    
```

< interactive example >



Givens QR Factorization

- More generally, to annihilate selected component of vector in n dimensions, rotate target component with another component

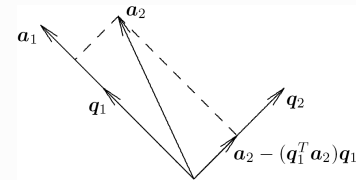
$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & c & 0 & s & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -s & 0 & c & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} a_1 \\ \alpha \\ a_3 \\ 0 \\ a_5 \end{bmatrix}$$

- By systematically annihilating successive entries, we can reduce matrix to upper triangular form using sequence of Givens rotations
- Each rotation is orthogonal, so their product is orthogonal, producing QR factorization



Gram-Schmidt Orthogonalization

- Given vectors \mathbf{a}_1 and \mathbf{a}_2 , we seek orthonormal vectors \mathbf{q}_1 and \mathbf{q}_2 having same span
- This can be accomplished by subtracting from second vector its projection onto first vector and normalizing both resulting vectors, as shown in diagram



< interactive example >



Modified Gram-Schmidt

- Classical Gram-Schmidt procedure often suffers loss of orthogonality in finite-precision
- Also, separate storage is required for \mathbf{A} , \mathbf{Q} , and \mathbf{R} , since original \mathbf{a}_k are needed in inner loop, so \mathbf{q}_k cannot overwrite columns of \mathbf{A}
- Both deficiencies are improved by *modified Gram-Schmidt* procedure, with each vector orthogonalized in turn against all *subsequent* vectors, so \mathbf{q}_k can overwrite \mathbf{a}_k



Rank Deficiency

- If $\text{rank}(\mathbf{A}) < n$, then QR factorization still exists, but yields singular upper triangular factor \mathbf{R} , and multiple vectors \mathbf{x} give minimum residual norm
- Common practice selects minimum residual solution \mathbf{x} having smallest norm
- Can be computed by QR factorization with column pivoting or by singular value decomposition (SVD)
- Rank of matrix is often not clear cut in practice, so relative tolerance is used to determine rank



Example: Near Rank Deficiency

- Consider 3×2 matrix

$$A = \begin{bmatrix} 0.641 & 0.242 \\ 0.321 & 0.121 \\ 0.962 & 0.363 \end{bmatrix}$$

- Computing QR factorization,

$$R = \begin{bmatrix} 1.1997 & 0.4527 \\ 0 & 0.0002 \end{bmatrix}$$

- R is extremely close to singular (exactly singular to 3-digit accuracy of problem statement)
- If R is used to solve linear least squares problem, result is highly sensitive to perturbations in right-hand side
- For practical purposes, $\text{rank}(A) = 1$ rather than 2, because columns are nearly linearly dependent



QR with Column Pivoting, continued

- Basic solution** to least squares problem $Ax \cong b$ can now be computed by solving triangular system $Rz = c_1$, where c_1 contains first k components of $Q^T b$, and then taking

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

- Minimum-norm solution** can be computed, if desired, at expense of additional processing to annihilate S
- $\text{rank}(A)$ is usually unknown, so rank is determined by monitoring norms of remaining unreduced columns and terminating factorization when maximum value falls below chosen tolerance

< interactive example >



Example: SVD

- SVD of $A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$ is given by $U\Sigma V^T =$

$$\begin{bmatrix} .141 & .825 & -.420 & -.351 \\ .344 & .426 & .298 & .782 \\ .547 & .0278 & .664 & -.509 \\ .750 & -.371 & -.542 & .0790 \end{bmatrix} \begin{bmatrix} 25.5 & 0 & 0 \\ 0 & 1.29 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} .504 & .574 & .644 \\ -.761 & -.057 & .646 \\ .408 & -.816 & .408 \end{bmatrix}$$

< interactive example >



Pseudoinverse

- Define pseudoinverse of scalar σ to be $1/\sigma$ if $\sigma \neq 0$, zero otherwise
- Define pseudoinverse of (possibly rectangular) diagonal matrix by transposing and taking scalar pseudoinverse of each entry
- Then **pseudoinverse** of general real $m \times n$ matrix A is given by

$$A^+ = V\Sigma^+U^T$$

- Pseudoinverse always exists whether or not matrix is square or has full rank
- If A is square and nonsingular, then $A^+ = A^{-1}$
- In all cases, minimum-norm solution to $Ax \cong b$ is given by $x = A^+ b$



QR with Column Pivoting

- Instead of processing columns in natural order, select for reduction at each stage column of remaining unreduced submatrix having maximum Euclidean norm
- If $\text{rank}(A) = k < n$, then after k steps, norms of remaining unreduced columns will be zero (or "negligible" in finite-precision arithmetic) below row k
- Yields orthogonal factorization of form

$$Q^T A P = \begin{bmatrix} R & S \\ O & O \end{bmatrix}$$

where R is $k \times k$, upper triangular, and nonsingular, and permutation matrix P performs column interchanges



Singular Value Decomposition

- Singular value decomposition (SVD) of $m \times n$ matrix A has form

$$A = U\Sigma V^T$$

where U is $m \times m$ orthogonal matrix, V is $n \times n$ orthogonal matrix, and Σ is $m \times n$ diagonal matrix, with

$$\sigma_{ij} = \begin{cases} 0 & \text{for } i \neq j \\ \sigma_i \geq 0 & \text{for } i = j \end{cases}$$

- Diagonal entries σ_i , called **singular values** of A , are usually ordered so that $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n$
- Columns u_i of U and v_i of V are called left and right **singular vectors**



Applications of SVD

- Minimum norm solution** to $Ax \cong b$ is given by

$$x = \sum_{\sigma_i \neq 0} \frac{u_i^T b}{\sigma_i} v_i$$

For ill-conditioned or rank deficient problems, "small" singular values can be omitted from summation to stabilize solution

- Euclidean matrix norm:** $\|A\|_2 = \sigma_{\max}$
- Euclidean condition number of matrix:** $\text{cond}(A) = \frac{\sigma_{\max}}{\sigma_{\min}}$
- Rank of matrix:** number of nonzero singular values



Orthogonal Bases

- SVD of matrix, $A = U\Sigma V^T$, provides orthogonal bases for subspaces relevant to A
- Columns of U corresponding to nonzero singular values form orthonormal basis for $\text{span}(A)$
- Remaining columns of U form orthonormal basis for orthogonal complement $\text{span}(A)^\perp$
- Columns of V corresponding to zero singular values form orthonormal basis for null space of A
- Remaining columns of V form orthonormal basis for orthogonal complement of null space of A



Lower-Rank Matrix Approximation

- Another way to write SVD is

$$A = U\Sigma V^T = \sigma_1 E_1 + \sigma_2 E_2 + \dots + \sigma_n E_n$$

with $E_i = u_i v_i^T$

- E_i has rank 1 and can be stored using only $m + n$ storage locations
- Product $E_i x$ can be computed using only $m + n$ multiplications
- Condensed approximation to A is obtained by omitting from summation terms corresponding to small singular values
- Approximation using k largest singular values is closest matrix of rank k to A
- Approximation is useful in image processing, data compression, information retrieval, cryptography, etc.

[< interactive example >](#)

Comparison of Methods

- Forming normal equations matrix $A^T A$ requires about $n^2 m / 2$ multiplications, and solving resulting symmetric linear system requires about $n^3 / 6$ multiplications
- Solving least squares problem using Householder QR factorization requires about $mn^2 - n^3 / 3$ multiplications
- If $m \approx n$, both methods require about same amount of work
- If $m \gg n$, Householder QR requires about twice as much work as normal equations
- Cost of SVD is proportional to $mn^2 + n^3$, with proportionality constant ranging from 4 to 10, depending on algorithm used

Comparison of Methods, continued

- Householder is more accurate and more broadly applicable than normal equations
- These advantages may not be worth additional cost, however, when problem is sufficiently well conditioned that normal equations provide sufficient accuracy
- For rank-deficient or nearly rank-deficient problems, Householder with column pivoting can produce useful solution when normal equations method fails outright
- SVD is even more robust and reliable than Householder, but substantially more expensive

Total Least Squares

- Ordinary least squares is applicable when right-hand side b is subject to random error but matrix A is known accurately
- When all data, including A , are subject to error, then total least squares is more appropriate
- Total least squares minimizes orthogonal distances, rather than vertical distances, between model and data
- Total least squares solution can be computed from SVD of $[A, b]$

Comparison of Methods, continued

- Normal equations method produces solution whose relative error is proportional to $[\text{cond}(A)]^2$
- Required Cholesky factorization can be expected to break down if $\text{cond}(A) \approx 1/\sqrt{\epsilon_{\text{mach}}}$ or worse
- Householder method produces solution whose relative error is proportional to

$$\text{cond}(A) + \|r\|_2 [\text{cond}(A)]^2$$

which is best possible, since this is inherent sensitivity of solution to least squares problem

- Householder method can be expected to break down (in back-substitution phase) only if $\text{cond}(A) \approx 1/\epsilon_{\text{mach}}$ or worse