STAT 309: MATHEMATICAL COMPUTATIONS I
FALL 2013
LECTURE 10

- we will discuss a general principle for solving linear systems and least squares problems via matrix factorization

1. BACKSOLVE

- backsolve refers to a simple, intuitive way of solving linear systems of the form $R\mathbf{x} = \mathbf{y}$ or $L\mathbf{x} = \mathbf{y}$ where R is upper-triangular and L is lower-triangular
- take $R\mathbf{x} = \mathbf{y}$ for illustration

$$\begin{bmatrix} r_{11} & \cdots & r_{1n} \\ & \ddots & \vdots \\ & & r_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

- start at the bottom and work out way up

$$\begin{aligned} y_n &= r_{nn}x_n \\ y_{n-1} &= r_{n-1,n}x_n + r_{n-1,n-1}x_{n-1} \\ &\vdots \\ y_1 &= r_{11}x_1 + r_{12}x_2 + \cdots + r_{1n}x_n \end{aligned}$$

- we get

$$\begin{aligned} x_n &= \frac{y_n}{r_{nn}} \\ x_{n-1} &= \frac{y_{n-1} - r_{n-1,n}(y_n/r_n)}{r_{n-1,n-1}} \\ &\vdots \end{aligned}$$

- this requires that $r_{kk} \neq 0$ for all $k = 1, \ldots, n$, which is guaranteed if R is nonsingular
- for example we could use QR factorization
- given $A \in \mathbb{C}^{n\times n}$ nonsingular and $\mathbf{b} \in \mathbb{C}^n$
- step 1: find QR factorization $A = QR$
- step 2: form $\mathbf{y} = Q^*\mathbf{b}$
- step 3: backsolve $R\mathbf{x} = \mathbf{y}$ to get $\mathbf{x}$

2. GENERAL PRINCIPLE FOR FACTORING MATRICES

- it is easy to solve $A\mathbf{x} = \mathbf{b}$ if
  - A is unitary or orthogonal (includes permutation matrices)
  - A is upper- or lower-triangular (includes diagonal matrices)
  - $A\mathbf{x} = \mathbf{b}$ with such A can be solved with $O(n^2)$ flops
  - if A represents a special orthogonal matrix like the discrete Fourier or wavelet transforms, then $A\mathbf{x} = \mathbf{b}$ can in fact be solved in $O(n \log n)$ flops using algorithms like fast Fourier or fast wavelet transforms

Date: October 27, 2013, version 1.0. Comments, bug reports: lekheng@galton.uchicago.edu.

- if A is not one of these forms, we factorize A into a product of matrices of these forms
- this includes all the basic matrix factorizations LU, QR, SVD, EVD
- actually to the above list, we could also add
  - A is bidiagonal/tridiagonal (or banded, i.e., $a_{ij} = 0$ if $|i - j| > b$ for some *bandwidth* $b \ll n$)
  - A is Toeplitz or Hankel, i.e., $a_{ij} = a_{i-j}$ or $a_{ij} = a_{i+j}$ — constant on the diagonals or the opposite diagonals
  - A is semiseparable
  - $A\mathbf{x} = \mathbf{b}$ with bidiagonal or tridiagonal A can be solved in $O(n)$ flops
  - $A\mathbf{x} = \mathbf{b}$ with Toeplitz or Hankel A can be solved in $O(n^2 \log n)$ flops
  - these are often called structured matrices
- in this course we will just restrict ourselves to unitary and triangular factors
- but we will discuss a general principle for solving linear systems and least squares problems based on rank-retaining factorizations that works with any structured matrices

3. RANK-RETAINING FACTORIZATIONS

- let $A \in \mathbb{C}^{m\times n}$ with $\operatorname{rank}(A) = r$, a *rank-retaining factorization* is a factorization of A into
$$A = GH$$
where $G \in \mathbb{C}^{m\times r}$ and $H \in \mathbb{C}^{r\times n}$ and
$$\operatorname{rank}(G) = \operatorname{rank}(H) = r$$
  - example: reduced SVD $A = U\Sigma V^*$, $U \in \mathbb{C}^{m\times r}$, $\Sigma \in \mathbb{C}^{r\times r}$, $V \in \mathbb{C}^{n\times r}$ where we could pick $G = U\Sigma$ and $H = V^*$ or $G = U$ and $H = \Sigma V^*$
  - example: reduced QR $A\Pi = QR$, $Q \in \mathbb{C}^{m\times r}$, $R \in \mathbb{C}^{r\times n}$, where we could pick $G = Q$ and $H = R\Pi^\top$
  - example: reduced LU $\Pi_1 A \Pi_2 = LU$, $L \in \mathbb{C}^{m\times r}$, $U \in \mathbb{C}^{r\times n}$, where we could pick $G = \Pi_1^\top L$ and $H = U\Pi_2^\top$
- easy facts: if $A = GH$ is rank-retaining, then
  (1) $G^*G \in \mathbb{C}^{r\times r}$ is nonsingular
  (2) $HH^* \in \mathbb{C}^{r\times r}$ is nonsingular
  (3) $\operatorname{im}(A) = \operatorname{im}(G)$
  (4) $\ker(A^*) = \ker(G^*)$
  (5) $\ker(A) = \ker(H)$
  (6) $\operatorname{im}(A^*) = \operatorname{im}(H^*)$
- prove these as exercises

4. GENERAL PRINCIPLE FOR LINEAR SYSTEMS AND LEAST SQUARES

- given $A \in \mathbb{C}^{m\times n}$ and $\mathbf{b} \in \mathbb{C}^m$, two of the most common problems are
  - if $A\mathbf{x} = \mathbf{b}$ is consistent and A is full column rank, we want the unique solution
  - if $A\mathbf{x} = \mathbf{b}$ is inconsistent and A is full column rank, we want the unique least squares solution
- the trouble is that when A is rank deficient, i.e., not full rank, then the solution is not unique and so we want the minimum length solution instead
  - if $A\mathbf{x} = \mathbf{b}$ is consistent and A is rank deficient, we want the minimum length solution
$$\min\{\|\mathbf{x}\|_2 : A\mathbf{x} = \mathbf{b}\} \tag{4.1}$$
  - if $A\mathbf{x} = \mathbf{b}$ is inconsistent and A is rank deficient, we want the minimum length least squares solution
$$\min\{\|\mathbf{x}\|_2 : \mathbf{x} \in \operatorname{argmin}\|\mathbf{b} - A\mathbf{x}\|_2\} \tag{4.2}$$

- if we can solve the min length versions then we can solve the full column rank versions, so let's focus on the min length version

5. min length linear systems via rank-retaining factorization

- we start from the consistent case: $\mathbf{b} \in \mathrm{im}(A)$ and so $\mathbf{b} = A\mathbf{x}$ for some $\mathbf{x} \in \mathbb{C}^n$
  - recall the Fredholm alternative that we proved in the homework:
$$\mathbb{C}^n = \mathrm{im}(A^*) \oplus \ker(A)$$
  - $\mathbf{x} \in \mathbb{C}^n$ can be written uniquely as
$$\mathbf{x} = \mathbf{x}_0 + \mathbf{x}_1, \quad \mathbf{x}_0 \in \ker(A),\ \mathbf{x}_1 \in \mathrm{im}(A^*),\ \mathbf{x}_0^*\mathbf{x}_1 = 0$$
  - since
$$\mathbf{b} = A\mathbf{x} = A\mathbf{x}_0 + A\mathbf{x}_1 = A\mathbf{x}_1$$
  $\mathbf{x}_1$ is also a solution to the linear system
  - by Pythagoras theorem
$$\|\mathbf{x}\|_2^2 = \|\mathbf{x}_0\|_2^2 + \|\mathbf{x}_1\|_2^2 \geq \|\mathbf{x}_1\|_2^2$$
  - so for a minimum length solution we set $\mathbf{x}_0 = \mathbf{0}$, i.e., the minimum length solution is given by $\mathbf{x} = \mathbf{x}_1$
- now we will see how to find $\mathbf{x}_1$ using a rank-retaining factorization
$$A = GH \tag{5.1}$$
  - since $\mathbf{x}_1 \in \mathrm{im}(A^*)$, so for some $\mathbf{v} \in \mathbb{C}^m$,
$$\mathbf{x}_1 = A^*\mathbf{v} \tag{5.2}$$
  - by easy fact (iii), $\mathbf{b} \in \mathrm{im}(A) = \mathrm{im}(G)$ and so for some $\mathbf{s} \in \mathbb{C}^r$,
$$\mathbf{b} = G\mathbf{s} \tag{5.3}$$
  - so upon substituting (5.1), (5.2), (5.3), $A\mathbf{x}_1 = \mathbf{b}$ becomes
$$GHH^*G^*\mathbf{v} = G\mathbf{s}$$
  - now multiply by G^* to get
$$(G^*G)HH^*G^*\mathbf{v} = (G^*G)\mathbf{s}$$
  - by easy fact (i), G^*G is nonsingular and so
$$HH^*G^*\mathbf{v} = \mathbf{s}$$
  - by easy fact (ii), HH^* is nonsingular and so
$$G^*\mathbf{v} = (HH^*)^{-1}\mathbf{s}$$
- this gives an algorithm for solving the minimum length linear system (4.1)
  - step 1: compute rank retaining factorization $A = GH$
  - step 2: solve $G\mathbf{s} = \mathbf{b}$ for $\mathbf{s} \in \mathbb{C}^r$
  - step 3: solve $HH^*\mathbf{z} = \mathbf{s}$ for $\mathbf{z} \in \mathbb{C}^r$
  - step 4: compute $\mathbf{x}_1 = H^*\mathbf{z}$
- this works because
$$A\mathbf{x}_1 = GH\mathbf{x}_1 = GHH^*\mathbf{z} = G(HH^*)(HH^*)^{-1}\mathbf{s} = G\mathbf{s} = \mathbf{b}$$
- note that the system in steps 2 and 3 involve a full-rank G and a nonsingular HH^* — both have unique solutions
- example: if $A\Pi = QR$ is the reduced QR, then with $G = Q$ and $H = R\Pi^\top$

- step 2: $Q\mathbf{s} = \mathbf{b}$ is easy to obtain via

$$Q^*Q\mathbf{s} = Q^*\mathbf{b}$$

and so $\mathbf{s} = Q^*\mathbf{b}$

- step 3: $R\Pi^\top \Pi R^*\mathbf{z} = \mathbf{s}$ is also easy to obtain via two backsolves

$$\begin{cases} R\mathbf{y} = \mathbf{s} \\ R^*\mathbf{z} = \mathbf{y} \end{cases}$$

- example: if $A = U\Sigma V^*$ is the reduced SVD, then with $G = U$ and $H = \Sigma V^*$
  - step 2: $U\mathbf{s} = \mathbf{b}$ is easy to obtain via

$$U^*U\mathbf{s} = U^*\mathbf{b}$$

and so $\mathbf{s} = U^*\mathbf{b}$

  - step 3: $\Sigma V^* V \Sigma \mathbf{z} = \mathbf{s}$ is just

$$\Sigma^2\mathbf{z} = \mathbf{s}$$

or

$$\begin{bmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_r^2 \end{bmatrix} \begin{bmatrix} z_1 \\ \vdots \\ z_r \end{bmatrix} = \begin{bmatrix} s_1 \\ \vdots \\ s_r \end{bmatrix}$$

and so for $k = 1, \ldots, r$,

$$z_k = s_k/\sigma_k^2$$

6. min length least squares via rank-retaining factorization

- we now consider the inconsistent case: $\mathbf{b} \notin \mathrm{im}(A)$
  - this time we use the other part of the Fredholm alternative:

$$\mathbb{C}^m = \ker(A^*) \oplus \mathrm{im}(A)$$

  - any $\mathbf{b} \in \mathbb{C}^m$ can be written uniquely as

$$\mathbf{b} = \mathbf{b}_0 + \mathbf{b}_1, \quad \mathbf{b}_0 \in \ker(A^*),\ \mathbf{b}_1 \in \mathrm{im}(A),\ \mathbf{b}_0^*\mathbf{b}_1 = 0$$

  - since $\mathbf{b}_1 - A\mathbf{x} \in \mathrm{im}(A)$, it must also be orthogonal to $\mathbf{b}_0$ and by Pythagoras

$$\|\mathbf{b} - A\mathbf{x}\|_2^2 = \|\mathbf{b}_0 + \mathbf{b}_1 - A\mathbf{x}\|_2^2 = \|\mathbf{b}_0\|_2^2 + \|\mathbf{b}_1 - A\mathbf{x}\|_2^2 \geq \|\mathbf{b}_0\|_2^2$$

  - so for a least squares solution, we must have

$$\|\mathbf{b}_1 - A\mathbf{x}\|_2^2 = 0$$

i.e.,

$$A\mathbf{x} = \mathbf{b}_1 \tag{6.1}$$

  - this is always consistent since $\mathbf{b}_1 \in \mathrm{im}(A)$
  - now we apply the result in the previous section to the consistent system (6.1) to obtain a minimum length solution
- suppose we have a rank-retaining factorization $A = GH$
  - by easy fact (iv), $\ker(A^*) = \ker(G^*)$ and so

$$G^*\mathbf{b} = G^*(\mathbf{b}_0 + \mathbf{b}_1) = G^*\mathbf{b}_0 + G^*\mathbf{b}_1 = G^*\mathbf{b}_1 \tag{6.2}$$

  - now mutiply (6.1) by G^* and use (6.2) to get

$$G^*A\mathbf{x} = G^*\mathbf{b}$$

note that we don't need to know $\mathbf{b}_1$ if we have a rank-retaining factorization

– following the previous section, we can write down an algorithm to get the minimum length solution to a least squares problem (4.2) as

$$\mathbf{x}_1 = H^*(HH^*)^{-1}(G^*G)^{-1}G^*\mathbf{b}$$

– note that all we need to know is the rank-retaining factorization of A and $\mathbf{b}$

- a consequence is that given a rank-retaining factorization $A = GH$, the Moore-Penrose pseudoinverse of A is given by

$$A^\dagger = H^*(HH^*)^{-1}(G^*G)^{-1}G^* \tag{6.3}$$

- as an exercise (6.3) and write down the algorithm for solving (4.2) with a rank-retaining factorization
- example: if $A = U\Sigma V^*$ is the reduced SVD, then $A^\dagger = V\Sigma^{-1}U^*$ since (6.3) with $G = U$ and $H = \Sigma V^*$ yields

$$A^\dagger = V\Sigma(\Sigma V^*V\Sigma)^{-1}(U^*U)^{-1}U^* = V\Sigma\Sigma^{-2}U^* = V\Sigma^{-1}U^*$$

- example: if $A\Pi = QR$ is the reduced QR, then $A^\dagger = \Pi R^*(RR^*)^{-1}Q^*$ since (6.3) with $G = Q$ and $H = R\Pi^\top$ yields

$$A^\dagger = \Pi R^*(R\Pi^\top\Pi R^*)^{-1}(Q^*Q)^{-1}Q^* = \Pi R^*(RR^*)^{-1}Q^*$$

7. Magnitude of determinant

- we continue our discussion of applications of QR
- let's start with an easy one:

$$|\det(A)| = |\det(QR)| = |\det(Q)||\det(R)| = |\det(R)| = \prod_{k=1}^{n}|r_{kk}|$$

- we used two facts: determinant of unitary matrix must have absolute value 1, determinant of triangular (upper or lower) matrix is just product of diagonal elements

8. Orthonormal bases for fundamental subspaces

- from the previous lecture notes, the full QR factorization of A can be written

$$A = [Q_1, Q_2]\begin{bmatrix} R_1 \\ 0 \end{bmatrix}$$

- the columns of Q_1 is an orthonormal basis for $\operatorname{im}(A)$ (follows from Gram-Schmidt) and the columns of Q_2 is an orthonormal basis for $\ker(A^*)$
- if we need orthonormal bases for $\operatorname{im}(A^*)$ and $\ker(A)$, we find the full QR factorization of A^*
- this is a cheaper way than SVD to obtain orthonormal bases for the fundamental subsapces

9. Full rank least squares problem

- while the general method considered in sections 6 works for matrices of any rank, there are better alternatives to solve least squares problem when the coefficient matrix A has full column rank
- here we seek to minimize $\|A\mathbf{x} - \mathbf{b}\|_2$ where $A \in \mathbb{C}^{m\times n}$ has $\operatorname{rank}(A) = n \leq m$ and $\mathbf{b} \in \mathbb{C}^m$
- such problems *always* have unique solution $\mathbf{x}^*$ (why?)
- so there is no question of finding a min length solution — since there's only one solution in this case, we don't get to choose
- we consider three methods:
  (1) QR factorization
  (2) normal equations
  (3) augmented system

- mathematically they all give the same solution (i.e., in exact arithmetic) but they have different numerical properties
- so one has to know all three since each is good/bad under different circumstances

10. full rank least squares via QR

- the first approach is to take advantage of the fact that the 2-norm is invariant under orthogonal transformations, and seek an orthogonal matrix Q such that the transformed problem
$$\min \|A\mathbf{x} - \mathbf{b}\|_2 = \min \|Q^*(A\mathbf{x} - \mathbf{b})\|_2$$
is "easy" to solve
- we could use the QR factorization of A
$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix} = Q_1 R$$
- then $Q_1^* A = R$ and
$$\begin{aligned} \min \|A\mathbf{x} - \mathbf{b}\|_2 &= \min \|Q^*(A\mathbf{x} - \mathbf{b})\|_2 \\ &= \| \min \|(Q^*A)\mathbf{x} - Q^*\mathbf{b}\|_2 \\ &= \min \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} \mathbf{x} - Q^*\mathbf{b} \right\|_2 \end{aligned}$$
- if we partition
$$Q^*\mathbf{b} = \begin{bmatrix} \mathbf{c} \\ \mathbf{d} \end{bmatrix}$$
then
$$\min \|A\mathbf{x} - \mathbf{b}\|_2^2 = \min \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{c} \\ \mathbf{d} \end{bmatrix} \right\|_2^2 = \min \|R\mathbf{x} - \mathbf{c}\|_2^2 + \|\mathbf{d}\|_2^2$$
- therefore the minimum is achieved by the vector $\mathbf{x}$ such that $R\mathbf{x} = \mathbf{c}$ and therefore
$$\min_{\mathbf{x} \in \mathbb{C}^n} \|A\mathbf{x} - \mathbf{b}\|_2 = \|\mathbf{d}\|_2 =: \rho_{\text{LS}}$$

11. full rank least squares via normal equations

- the second approach is to define
$$\varphi(\mathbf{x}) = \frac{1}{2} \|A\mathbf{x} - \mathbf{b}\|_2^2$$
which is a differentiable function of $\mathbf{x}$
- we can minimize $\varphi(\mathbf{x})$ by noting that $\nabla\varphi(\mathbf{x}) = A^*(A\mathbf{x} - \mathbf{b})$ which means that $\nabla\varphi(\mathbf{x}) = \mathbf{0}$ if and only if
$$A^*A\mathbf{x} = A^*\mathbf{b} \tag{11.1}$$
- this system of equations is called the *normal equations*, and were used by Gauss to solve least squares problems
- we saw at least two other ways to derive (11.1) in the homeworks
- while I said that it's generally a bad idea to solve the normal equations to get the least squares solution, this is not always the case
- for example, if $n \ll m$ then A^*A is $n \times n$, which is a much smaller system to solve than solving $\min \|A\mathbf{x} - \mathbf{b}\|_2^2$ via finding QR of A, and if $\kappa(A^*A)$ is not too large, we can indeed solve (11.1)
- for A of full column rank, the matrix A^*A is positive definite and one should apply Cholesky factorization (to be discussed later) to the matrix A^*A in order to solve (11.1)
- which is the better method?

- this is not a simple question to answer
- the normal equations produce an $\mathbf{x}^*$ whose relative error depends on $\kappa_2(A^\mathsf{T}A) = \kappa_2(A)^2$, whereas the QR factorization produces an $\mathbf{x}^*$ whose relative error depends on $\kappa_2(A) + \rho_{\text{LS}}\kappa_2(A)^2$
- so the QR factorization method in the previous section is appealing if ρ_{LS} is small, i.e., $\mathbf{b}$ is very close to $\text{im}(A)$, the span of the columns of A — which is more often than not the case (e.g. in linear regression) as the most common reason for wanting to solve $\min\|A\mathbf{x} - \mathbf{b}\|_2$ is when we expect $A\mathbf{x} \approx \mathbf{b}$ for some $\mathbf{x}$
- the normal equations involve much less arithmetic when $n \ll m$ and the $n \times n$ matrix A^*A requires less storage

12. full rank least squares via augmented system

- we can cast the normal equation in another form
- let $\mathbf{r} = \mathbf{b} - A\mathbf{x}$ be the residual
- now by the normal equations

$$A^*\mathbf{r} = A^*\mathbf{b} - A^*A\mathbf{x} = \mathbf{0}$$

- and so we obtain the system

$$\begin{aligned} \mathbf{r} + A\mathbf{x} &= \mathbf{b} \\ A^*\mathbf{r} &= \mathbf{0} \end{aligned}$$

- in matrix form, we get

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

- this is often a large system since the coefficient matrix has dimension $(m+n) \times (m+n)$, but it preserves the sparsity of A

13. least squares with linear constraints

- suppose that we wish to fit data as in the least squares problem, except that we are using different functions to fit the data on different subintervals
- a common example is the process of fitting data using cubic splines, with a different cubic polynomial approximating data on each subinterval
- typically it is desired that the functions assigned to each piece form a function that is continuous on the entire interval within which the data lies
- this requires that *constraints* be imposed on the functions themselves
- it is also not uncommon to require that the function assembled from these pieces also has a continuous first or even second derivative, resulting in additional constraints
- the result is a *least squares problem with linear constraints*, as the constraints are applied to coefficients of predetermined functions chosen as a basis for some function space, such as the space of polynomials of a given degree
- the general form of a least squares problem with linear constraints is as follows: we wish to find an $\mathbf{x} \in \mathbb{R}^n$ that minimizes $\|A\mathbf{x} - \mathbf{b}\|_2$, subject to the constraint $C^\top\mathbf{x} = \mathbf{d}$, where $A \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{n\times p}$, $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{d} \in \mathbb{R}^p$ are given

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{b} - A\mathbf{x}\|_2^2 \\ \text{subject to} \quad & C^\top\mathbf{x} = \mathbf{d} \end{aligned} \tag{13.1}$$

- again we will describe three methods, mathematically equivalent but with different numerical properties

- this problem is usually solved using *Lagrange multipliers*, define

$$L(\mathbf{x}, \boldsymbol{\lambda}) = \|\mathbf{b} - A\mathbf{x}\|_2^2 + 2\boldsymbol{\lambda}^\top C^\top \mathbf{x}$$

- in optimization parlance, the function L is called the *Lagrangian* and $\boldsymbol{\lambda}^\mathsf{T} = [\lambda_1, \ldots, \lambda_p]^\mathsf{T}$ is the vector of Lagrange multipliers
- setting derivative with respect to $\mathbf{x}$ to zero yields

$$\mathbf{0} = \nabla_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\lambda}) = 2(A^\top A\mathbf{x} - A^\top \mathbf{b} + C\boldsymbol{\lambda})$$

  and so

$$A^\top A\mathbf{x} + C\boldsymbol{\lambda} = A^\top \mathbf{b} \tag{13.2}$$

- note that $\mathbf{0} = \nabla_{\boldsymbol{\lambda}} L(\mathbf{x}, \boldsymbol{\lambda})$ just gives us back the constraint

$$C^\top \mathbf{x} = \mathbf{d} \tag{13.3}$$

- in optimization parlance, (13.2) and (13.3) are collectively called the *KKT conditions*
- method 1: together (13.2) and (13.3) give the system

$$\begin{bmatrix} A^\top A & C \\ C^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} A^\top \mathbf{b} \\ \mathbf{d} \end{bmatrix}$$

- solving this linear system gives us the solution to (13.1)
- this method preserves the sparsity of C but involves a coefficient matrix of size $(n+p) \times (n+p)$, larger than the next two methods
- method 2: from $A^\top A\mathbf{x} = A^\top \mathbf{b} - C\boldsymbol{\lambda}$, we see that we can first compute $\mathbf{x} = \hat{\mathbf{x}} - (A^\top A)^{-1} C\boldsymbol{\lambda}$ where $\hat{\mathbf{x}}$ is the solution to the unconstrained least squares problem

$$\hat{\mathbf{x}} \in \operatorname{argmin} \|A\mathbf{x} - \mathbf{b}\|_2$$

- then from the equation $C^\top \mathbf{x} = \mathbf{d}$ we obtain the equation

$$C^\top (A^\top A)^{-1} C\boldsymbol{\lambda} = C^\top \hat{\mathbf{x}} - \mathbf{d} \tag{13.4}$$

  which we can now solve for $\boldsymbol{\lambda}$
- note that (13.4) is a $p \times p$ linear system
- the algorithm proceeds as follows:
  - solve the unconstrained least squares problem $\min \|A\mathbf{x} - \mathbf{b}\|_2$ for $\hat{\mathbf{x}}$
  - compute $A = QR$
  - form $W = (R^\top)^{-1} C$
  - compute $W = PU$, the QR factorization of W
  - solve $U^\top U\boldsymbol{\lambda} = \boldsymbol{\eta} = C^\top \hat{\mathbf{x}} - \mathbf{d}$ for $\boldsymbol{\lambda}$ and note that

$$\begin{aligned} U^\top U &= (P^\top W)^\top (P^\top W) \\ &= W^\top P P^\top W \\ &= C^\top R^{-1} (R^\top)^{-1} C \\ &= C^\top (R^\top R)^{-1} C \\ &= C^\top (R^\top Q^\top QR)^{-1} C \\ &= C^\top (A^\top A)^{-1} C \end{aligned}$$

  - set $\mathbf{x} = \hat{\mathbf{x}} - (A^\top A)^{-1} C\boldsymbol{\lambda}$
- method 2 is not the most practical since it has more unknowns than the unconstrained least squares problem, which is odd because the constraints should have the effect of eliminating unknowns, not adding them

- method 3: suppose we compute the QR factorization of C to obtain

$$Q^\top C = \begin{bmatrix} R \\ 0 \end{bmatrix}$$

  where R is a $p \times p$ upper triangular matrix
- then the constraint $C^\top \mathbf{x} = \mathbf{d}$ takes the form

$$R^\top \mathbf{u} = \mathbf{d}, \quad Q^\top \mathbf{x} = \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}$$

- then

$$\begin{aligned} \|\mathbf{b} - A\mathbf{x}\|_2 &= \|\mathbf{b} - AQQ^\top \mathbf{x}\|_2 \\ &= \left\| \mathbf{b} - \tilde{A} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} \right\|_2, \qquad \tilde{A} = AQ \\ &= \left\| \mathbf{b} - \begin{bmatrix} \tilde{A}_1 & \tilde{A}_2 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} \right\|_2 \\ &= \|\mathbf{b} - \tilde{A}_1 \mathbf{u} - \tilde{A}_2 \mathbf{v}\|_2 \end{aligned}$$

- thus we can obtain $\mathbf{x}$ by the following algorithm:
  - compute the QR factorization of C
  - compute $\tilde{A} = AQ$
  - solve $R^\top \mathbf{u} = \mathbf{d}$
  - solve the new least squares problem of minimizing $\|(\mathbf{b} - \tilde{A}_1 \mathbf{u}) - \tilde{A}_2 \mathbf{v}\|_2$
  - compute

$$\mathbf{x} = Q \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}$$

- method 3 has the advantage that there are fewer unknowns in each system that needs to be solved, and also that $\kappa_2(\tilde{A}_2) \leq \kappa_2(A)$
- the drawback is that sparsity can be destroyed