STAT 309: MATHEMATICAL COMPUTATIONS I
FALL 2023
LECTURE 11

1. 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 $C^{\mathsf{T}}\mathbf{x} = \mathbf{d}$, 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

$$\begin{array}{ll} \text{minimize} & \|\mathbf{b} - A\mathbf{x}\|_2^2 \\ \text{subject to} & C^{\mathsf{T}}\mathbf{x} = \mathbf{d} \end{array} \tag{1.1}$$

- again we will describe three methods, mathematically equivalent but with different numerical properties
- this problem is usually solved using *Lagrange multipliers*, define

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

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

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

  and so

$$A^{\mathsf{T}}A\mathbf{x} + C\boldsymbol{\lambda} = A^{\mathsf{T}}\mathbf{b} \tag{1.2}$$

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

$$C^{\mathsf{T}}\mathbf{x} = \mathbf{d} \tag{1.3}$$

- in optimization parlance, (1.2) and (1.3) are collectively called the *KKT conditions*
- method 1: together (1.2) and (1.3) give the system

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

Date: November 6, 2023, version 1.0. Comments, bug reports: `lekheng@uchicago.edu`.

- solving this linear system gives us a solution to (1.1) — it gives us both $\mathbf{x}$ (primal variables) and $\boldsymbol{\lambda}$ (dual variables) and is a trivial case of the *primal-dual interior point method* in optimization
- 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
- also it involves $A^{\mathsf{T}}A$ and as in the case of normal equation $\kappa_2(A^{\mathsf{T}}A) = \kappa_2(A)^2$
- if n is small, this is fine but generally we want a method that avoids actually forming $A^{\mathsf{T}}A$
- one way is to emulate what we did when we discussed norm-constrained least squares and use SVD but ideally we want to also avoid SVD since it is much more expensive than QR
- which brings us to the next method — even though it appears to involve forming $M^{\mathsf{T}}M$ for various matrices M, it actually doesn't
- method 2: if A has full column rank, then from $A^{\mathsf{T}}A\mathbf{x} = A^{\mathsf{T}}\mathbf{b} - C\boldsymbol{\lambda}$, we see that we can first compute $\mathbf{x} = \widehat{\mathbf{x}} - (A^{\mathsf{T}}A)^{-1}C\boldsymbol{\lambda}$ where $\widehat{\mathbf{x}}$ is the solution to the unconstrained least squares problem
$$\widehat{\mathbf{x}} \in \operatorname{argmin}\|A\mathbf{x} - \mathbf{b}\|_2$$
- then from the equation $C^{\mathsf{T}}\mathbf{x} = \mathbf{d}$ we obtain the $p \times p$ linear system
$$C^{\mathsf{T}}(A^{\mathsf{T}}A)^{-1}C\boldsymbol{\lambda} = C^{\mathsf{T}}\widehat{\mathbf{x}} - \mathbf{d} \tag{1.4}$$
which we can then solve for $\boldsymbol{\lambda}$
- this works because $A^{\mathsf{T}}A\widehat{\mathbf{x}} = A^{\mathsf{T}}\mathbf{b}$ and therefore
$$A^{\mathsf{T}}A\mathbf{x} = A^{\mathsf{T}}\mathbf{b} - C\boldsymbol{\lambda}$$
- the actual algorithm uses two QR factorization and does not actually require solving a system involving $M^{\mathsf{T}}M$ for any M
  - compute full-rank QR factorization of A
  $$A = Q\begin{bmatrix} R \\ 0 \end{bmatrix}$$
  with nonsingular $R \in \mathbb{R}^{n\times n}$
  - solve the unconstrained least squares problem $\min\|A\mathbf{x} - \mathbf{b}\|_2$ for $\widehat{\mathbf{x}}$
  - form $W = R^{-\mathsf{T}}C$ with p back substitutions
  $$R^{\mathsf{T}}\mathbf{w}_i = \mathbf{c}_i, \quad i = 1, \dots, p$$
  - compute QR factorization of W
  $$W = Q_1R_1$$
  - set
  $$\boldsymbol{\eta} = C^{\mathsf{T}}\widehat{\mathbf{x}} - \mathbf{d}$$
  - solve $R_1^{\mathsf{T}}R_1\boldsymbol{\lambda} = \boldsymbol{\eta}$ for $\boldsymbol{\lambda}$ with two back substitutions
  $$\begin{cases} R_1^{\mathsf{T}}\boldsymbol{\mu} = \boldsymbol{\eta}, \\ R_1\boldsymbol{\lambda} = \boldsymbol{\mu} \end{cases}$$
  - set $\mathbf{x} = \widehat{\mathbf{x}} - (R^{\mathsf{T}}R)^{-1}C\boldsymbol{\lambda}$ where term $\boldsymbol{\zeta} = (R^{\mathsf{T}}R)^{-1}C\boldsymbol{\lambda}$ is computed again with two back substitutions
  $$\begin{cases} R^{\mathsf{T}}\boldsymbol{\xi} = C\boldsymbol{\lambda}, \\ R\boldsymbol{\zeta} = \boldsymbol{\xi} \end{cases}$$

- this works because
$$A^\mathsf{T} A = \begin{bmatrix} R^\mathsf{T} & 0 \end{bmatrix} Q^\mathsf{T} Q \begin{bmatrix} R \\ 0 \end{bmatrix} = R^\mathsf{T} R$$
and
$$R_1^\mathsf{T} R_1 = (Q_1^\mathsf{T} W)^\mathsf{T} (Q_1^\mathsf{T} W) = W^\mathsf{T} Q_1 Q_1^\mathsf{T} W = C^\mathsf{T} R^{-1} R^{-\mathsf{T}} C = C^\mathsf{T} (R^\mathsf{T} R)^{-1} C = C^\mathsf{T} (A^\mathsf{T} A)^{-1} C$$
- method 2, like method 1, has more unknowns than the unconstrained least squares problem, which is a downside because the constraints should have the effect of eliminating unknowns, not adding them
- the next method overcomes this by solving (1.1) without introducing any Lagrange multipliers
- method 3: suppose $p \le n$, then computing the QR factorization of C yields
$$C = Q_2 \begin{bmatrix} R_2 \\ 0 \end{bmatrix}$$
where R_2 is a $p \times p$ upper triangular matrix
- then the constraint $C^\mathsf{T} \mathbf{x} = \mathbf{d}$ takes the form
$$R_2^\mathsf{T} \mathbf{u} = \mathbf{d}, \quad Q_2^\mathsf{T} \mathbf{x} = \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}$$
where $\mathbf{v}$ is to be determined later
- then
$$\begin{aligned} \|\mathbf{b} - A\mathbf{x}\|_2 &= \|\mathbf{b} - AQ_2 Q_2^\mathsf{T} \mathbf{x}\|_2 \\ &= \left\| \mathbf{b} - \widetilde{A} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} \right\|_2, \qquad \widetilde{A} = AQ_2 \\ &= \left\| \mathbf{b} - \begin{bmatrix} \widetilde{A}_1 & \widetilde{A}_2 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} \right\|_2 \\ &= \|\mathbf{b} - \widetilde{A}_1 \mathbf{u} - \widetilde{A}_2 \mathbf{v}\|_2 \end{aligned}$$
- thus we can obtain $\mathbf{x}$ by the following algorithm:
  - compute the QR factorization of C
  - compute $\widetilde{A} = AQ_2$
  - solve $R_2^\mathsf{T} \mathbf{u} = \mathbf{d}$ to obtain a solution $\mathbf{u}_*$
  - solve the new least squares problem
$$\mathbf{v}_* = \operatorname{argmin} \|(\mathbf{b} - \widetilde{A}_1 \mathbf{u}_*) - \widetilde{A}_2 \mathbf{v}\|_2$$
  - compute
$$\mathbf{x} = Q_2 \begin{bmatrix} \mathbf{u}_* \\ \mathbf{v}_* \end{bmatrix}$$
- method 3 has the advantage that there are fewer unknowns in each system that needs to be solved, and also that $\kappa_2(\widetilde{A}_2) \le \kappa_2(\widetilde{A}) = \kappa_2(A)$
- the drawback is that sparsity or other structure in A can be destroyed when we form $\widetilde{A}$

2. COMPUTING THE QR FACTORIZATION

- there are two common ways to compute the QR decomposition:
  - using *Householder matrices*, developed by Alston S. Householder
  - using *Givens rotations*, also known as *Jacobi rotations*, used by Wallace Givens and originally invented by Jacobi for use with in solving the symmetric eigenvalue problem in 1846

  - the Gram–Schmidt or modfied Gram–Schmidt orthogonalization discussed in previous lecture works in principle but has numerical stability issues and are not usually used
- roughly speaking, Gram–Schmidt applies a sequence of triangular matrices to orthogonalize A (i.e., transform A into an orthogonal matrix Q),
$$AR_1^{-1}R_2^{-1}\cdots R_{n-1}^{-1} = Q$$
whereas Householder and Givens QR apply a sequence of orthogonal matrices to triangularize A (i.e., transform A into an upper triangular matrix R),
$$Q_{n-1}^{\mathsf{T}}\cdots Q_2^{\mathsf{T}}Q_1^{\mathsf{T}}A = R$$
- orthogonal transformations are highly desirable in algorithms as they preserve lengths and therefore do not blow up the errors present at every stage of the computation

3. Why orthogonal/unitary

- unitary and orthogonal matrices are awesome because they preserve length
- it also preserves the length of your errors and so your errors don't get magnified during your computations
- more precisely, if we multiply a vector $\mathbf{a} \in \mathbb{C}^n$ or a matrix $A \in \mathbb{C}^{n\times k}$ by another matrix $X \in \mathrm{GL}(n)$ we usually magnify whatever error there is in $\mathbf{a}$ or A by $\kappa_2(X)$, the condition number of X
- more precisely, unitary and orthogonal matrices are awesome because they are perfectly conditioned, i.e., $\kappa_2(U) = 1$ for all $U \in U(n)$ (but converse is not true)
- for instance if we carry out the same backward error analysis in lecture **8** for the eigenvalue decomposition $A = Q\Lambda Q^*$ of a normal matrix $A \in \mathbb{C}^{n\times n}$, we have
$$A + \Delta A = Q(\Lambda + \Delta\Lambda)Q^*$$
thus $\Delta A = Q(\Delta\Lambda)Q^*$ and thus
$$\|\Delta A\| = \|\Delta\Lambda\|$$
by the unitarity of Q
- perturbation of any size in A causes perturbation of the same size in Λ, the condition number of eigenvalues of normal matrices is always 1
- same thing for singular value decomposition $A = U\Sigma V^*$, we have
$$A + \Delta A = U(\Sigma + \Delta\Sigma)V^*$$
thus $\Delta A = U(\Delta\Sigma)V^*$ and thus
$$\|\Delta A\| = \|\Delta\Sigma\|$$
by the unitarity of U and V
- perturbation of any size in A causes perturbation of the same size in Σ
- this is why we don't ever hear of "condition number of singular values" as it is always 1

4. Orthogonalization using Householder reflections

- it is natural to ask whether we can introduce more zeros with each orthogonal rotation and to that end, we examine *Householder reflections*
- consider a matrix of the form $H = I - \tau\mathbf{u}\mathbf{u}^{\mathsf{T}}$, where $\mathbf{u} \neq \mathbf{0}$ and τ is a nonzero constant
- a H that has this form is called a *symmetric rank*-1 *change* of I
- can we choose τ so that H is also orthogonal?

- from the desired relation $H^\top H = I$ we obtain

$$\begin{aligned} H^\top H &= (I - \tau \mathbf{u}\mathbf{u}^\top)^\top (I - \tau \mathbf{u}\mathbf{u}^\top) \\ &= I - 2\tau \mathbf{u}\mathbf{u}^\top + \tau^2 \mathbf{u}\mathbf{u}^\top \mathbf{u}\mathbf{u}^\top \\ &= I - 2\tau \mathbf{u}\mathbf{u}^\top + \tau^2 (\mathbf{u}^\top \mathbf{u}) \mathbf{u}\mathbf{u}^\top \\ &= I - (\tau^2 \mathbf{u}^\top \mathbf{u} - 2\tau) \mathbf{u}\mathbf{u}^\top \\ &= I + \tau(\tau \mathbf{u}^\top \mathbf{u} - 2) \mathbf{u}\mathbf{u}^\top \end{aligned}$$

- it follows that if $\tau = 2/\mathbf{u}^\top \mathbf{u}$, then $H^\top H = I$ for any nonzero $\mathbf{u}$
- without loss of generality, we can stipulate that $\mathbf{u}^\top \mathbf{u} = 1$, and therefore H takes the form $H = I - 2\mathbf{v}\mathbf{v}^\top$, where $\mathbf{v}^\top \mathbf{v} = 1$
- why is the matrix H called a reflection?
- this is because for any nonzero vector $\mathbf{x}$, $H\mathbf{x}$ is the reflection of $\mathbf{x}$ across the hyperplane that is normal to $\mathbf{v}$
- for example, consider the 2×2 case and set $\mathbf{v} = \begin{bmatrix} 1 & 0 \end{bmatrix}^\top$ and $\mathbf{x} = \begin{bmatrix} 1 & 2 \end{bmatrix}^\top$, then

$$H = I - 2\mathbf{v}\mathbf{v}^\top = I - 2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - 2 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

  and therefore

$$H\mathbf{x} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$$

- now, let $\mathbf{x}$ be any vector, we wish to construct H so that $H\mathbf{x} = \alpha \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^\top = \alpha \mathbf{e}_1$ for some α
- from the relations

$$\|H\mathbf{x}\|_2 = \|\mathbf{x}\|_2, \qquad \|\alpha \mathbf{e}_1\|_2 = |\alpha| \|\mathbf{e}_1\|_2 = |\alpha|$$

  we obtain $\alpha = \pm \|\mathbf{x}\|_2$
- to determine H, we observe that

$$\mathbf{x} = H^{-1}(\alpha \mathbf{e}_1) = \alpha H \mathbf{e}_1 = \alpha (I - 2\mathbf{v}\mathbf{v}^\top)\mathbf{e}_1 = \alpha(\mathbf{e}_1 - 2\mathbf{v}\mathbf{v}^\top \mathbf{e}_1) = \alpha(\mathbf{e}_1 - 2\mathbf{v} v_1)$$

  which yields the system of equations

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} = \alpha \begin{bmatrix} 1 - 2v_1^2 \\ -2v_1 v_2 \\ \vdots \\ -2v_1 v_m \end{bmatrix}$$

- from the first equation $x_1 = \alpha(1 - 2v_1^2)$ we obtain

$$v_1 = \pm \sqrt{\frac{1}{2}\left(1 - \frac{x_1}{\alpha}\right)}$$

- for $i = 2, \ldots, m$, we have

$$v_i = -\frac{x_i}{2\alpha v_1}$$

- it is best to choose α to have the opposite sign of x_1 to avoid cancellation in v_1
- it is conventional to choose the $+$ sign for α if $x_1 = 0$
- note that the matrix H is never formed explicitly: for any vector $\mathbf{b}$, the product $H\mathbf{b}$ can be computed as follows

$$H\mathbf{b} = (I - 2\mathbf{v}\mathbf{v}^\top)\mathbf{b} = \mathbf{b} - 2(\mathbf{v}^\top \mathbf{b})\mathbf{v} \tag{4.1}$$

- this process requires only $2n$ operations

- it is easy to see that we can represent H simply by storing only $\mathbf{v}$, which we will call the Householder vector
- we showed how a Householder reflection of the form $H = I - 2\mathbf{u}\mathbf{u}^\mathsf{T}$ could be constructed so that given a vector $\mathbf{x}$, $H\mathbf{x} = \alpha\mathbf{e}_1$
- now, suppose that that $\mathbf{x} = \mathbf{a}_1$ is the first column of a matrix $A \in \mathbb{R}^{m\times n}$ with full column rank $n \le m$, then we construct a Householder reflection $H_1 = I - 2\mathbf{u}_1\mathbf{u}_1^\mathsf{T}$ such that $H_1\mathbf{x} = \alpha\mathbf{e}_1$, and we have

$$A^{(2)} = H_1 A = \begin{bmatrix} r_{11} & & & \\ 0 & & & \\ \vdots & \mathbf{a}_2^{(2)} & \cdots & \mathbf{a}_n^{(2)} \\ 0 & & & \end{bmatrix}$$

where we denote the constant α by r_{11}, as it is the $(1,1)$ element of the updated matrix $A^{(2)}$
- we can next construct H_2 such that

$$H_2\mathbf{a}_2^{(2)} = \begin{bmatrix} a_{12}^{(2)} \\ r_{22} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad u_{12} = 0, \quad H_2 = \begin{bmatrix} 1 & 0 & \\ 0 & & \\ \vdots & h_{ij} & \\ 0 & & \end{bmatrix}$$

- note that the first column of $A^{(2)}$ is unchanged by H_2
- continuing this process, we obtain

$$H_{m-1}\cdots H_1 A = A^{(m)} = \begin{bmatrix} R \\ 0 \end{bmatrix}$$

where $R \in \mathbb{R}^{n\times n}$ is an upper triangular matrix and where

$$H_k = \begin{bmatrix} I_{k-1} & 0 \\ 0 & I_{m-k+1} - 2v_k v_k^\mathsf{T} \end{bmatrix} \in \mathbb{R}^{m\times m}$$

for $k = 1, \dots, m-1$
- we have thus factored $A = Q\left[\begin{smallmatrix} R \\ 0 \end{smallmatrix}\right]$, where $Q = H_1 H_2 \cdots H_{m-1} \in \mathbb{R}^{m\times m}$ is an orthogonal matrix
- when implementing the Householder QR algorithm, not only are the Householder matrices H_i not stored, the orthogonal factor Q is never formed explicitly
- instead we just keep the sequence of Householder vectors $v_1 \in \mathbb{R}^m, v_2 \in \mathbb{R}^{m-1}, \dots, v_{m-1} \in \mathbb{R}^1$, note that this is a sequence of vectors of decreasing dimensions
- whenever we need to use Q in the form of matrix-vector product $\mathbf{x} \mapsto Q\mathbf{x}$, $\mathbf{y} \mapsto Q^\mathsf{T}\mathbf{y}$ or matrix-matrix product $X \mapsto QX$, $Y \mapsto Q^\mathsf{T}Y$, we just rely on $\mathbf{v}_1, \dots, \mathbf{v}_m$ and (4.1) (you'd be asked to do this in Homework **3**)
- if we implement our Householder QR algorithm carefully, we may simply overwrite the entries the existing entries in A with the entries of $\mathbf{v}_1, \dots, \mathbf{v}_m$ and R as the algorithm proceeds

- for example if $A \in \mathbb{R}^{6\times 5}$, i.e., $m = 6$ and $n = 5$, then at the end of the Householder QR algorithm, the entries of A would become

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} \\ v_2^{(1)} & r_{22} & r_{23} & r_{24} & r_{25} \\ v_3^{(1)} & v_3^{(2)} & r_{33} & r_{34} & r_{35} \\ v_4^{(1)} & v_4^{(2)} & v_4^{(3)} & r_{44} & r_{45} \\ v_5^{(1)} & v_5^{(2)} & v_5^{(3)} & v_5^{(4)} & r_{55} \\ v_6^{(1)} & v_6^{(2)} & v_6^{(3)} & v_6^{(4)} & v_6^{(5)} \end{bmatrix} \tag{4.2}$$

  where $\mathbf{v}_k = \left[v_k^{(k)}, v_{k+1}^{(k)}, \dots, v_m^{(k)}\right]^\mathsf{T} \in \mathbb{R}^{m-k+1}$
- note that we have dropped the first entry of $\mathbf{v}_k$ in (4.2) since it can be recovered from

$$v_k^{(k)} = \sqrt{1 - \left(v_{k+1}^{(k)}\right)^2 - \cdots - \left(v_m^{(k)}\right)^2}$$

  by virtue of the fact that $\|\mathbf{v}_k\|_2 = 1$
- if you do not want to be bothered with recovering the $v_k^{(k)}$, you can just create an additional row to accommodate all entries of $\mathbf{v}_1, \dots, \mathbf{v}_m$