6.006 Lecture 25: Solving Linear systems & Least-Squares Problems

- Reducing $Ax-b$ to a triagular form while preserving $\|Ax-b\|_2^2$
- Illustration of numerical issves
- More general problems: adding LS constraints, eigenvalues; the details in 6.337 / 18.335

- If A does not have this structure, we will transform it and r ~~th~~ so A is [upper triangular] ~~and~~ while preserving $\sum_i r_i^2$, so we end up with the optimal.

- Trick 1 (of 2): mix pairs of rows to introduce one zero at a time to matrix while preserving $\sum r_i^2$.

$$\begin{bmatrix} & & \\ & A & \\ & & \end{bmatrix}\begin{bmatrix} x \end{bmatrix} - \begin{bmatrix} b \end{bmatrix} \equiv \begin{bmatrix} - A_1 - \\ \vdots \\ - A_{m-1} - \\ - A_m - \end{bmatrix}\begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} - \begin{bmatrix} b_1 \\ \vdots \\ b_{m-1} \\ b_m \end{bmatrix}$$

$$\left[\begin{array}{c|c} - A_1 - & b_1 \\ \vdots & \vdots \\ - A_{m-1} - & b_{m-1} \\ - A_m - & b_m \end{array}\right] \rightarrow \left[\begin{array}{c|c} A_1 & b_1 \\ \vdots & \vdots \\ A_{m-2} & b_{m-2} \\ cA_{m-1} + sA_m & cb_{m-1} + sb_m \\ -sA_{m-1} + cA_m & -sb_{m-1} + cb_m \end{array}\right]$$

Let's find c, s such that position $(m, 1)$ becomes zero.

We have two constraints:

$$-s a_{m-1,1} + c a_{m,1} = 0$$

$$(c r_{m-1} + s r_m)^2 + (-s r_{m-1} + c r_m)^2 = r_{m-1}^2 + r_m^2$$

$$(c^2 + s^2) r_{m-1}^2 + (s^2 + c^2) r_m^2 = r_{m-1}^2 + r_m^2$$

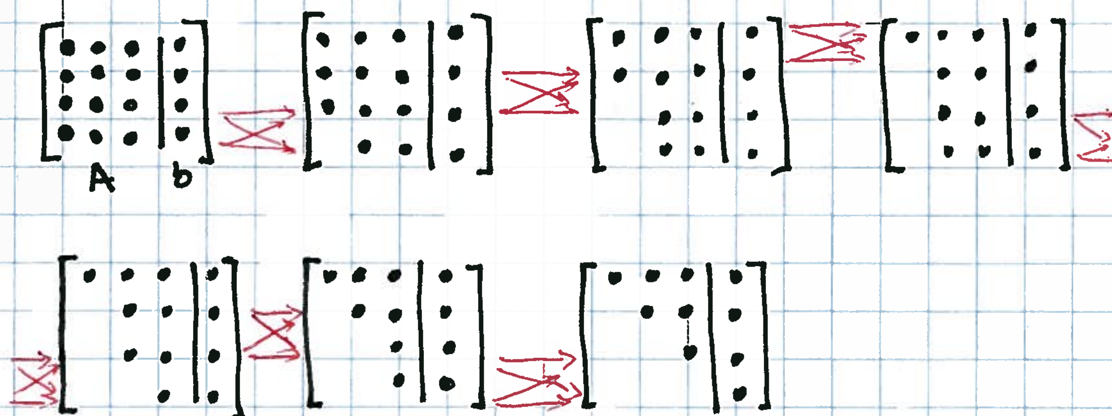
$$c^2 + s^2 = 1$$

and $c = \dfrac{s a_{m-1,1}}{a_{m,1}}$ or $s = \dfrac{c a_{m,1}}{a_{m-1,1}}$

This is called a Givens rotation

Solve the quadratic in c (or s) and we are done.

- Trick 2: Use trick 1 to create more and more zeros in A without destroying earlier ones (or while destroying just a few in more advanced versions).



- This algorithm is called the Givens QR factorization (we factored $A = Q_1 Q_2 \cdots Q_K R = QR$, Q orthonormal, R upper triangular)

- Complexity: $O(n)$ operations per rotation

  $O(nm)$ rotations

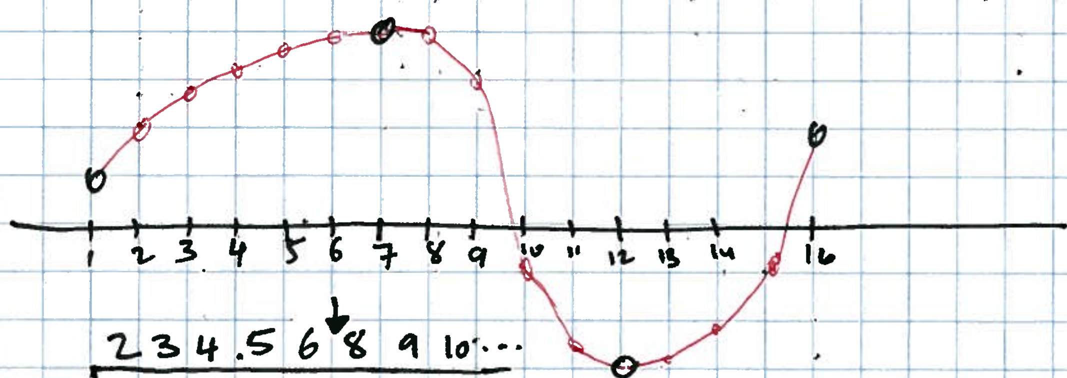
  $O(mn^2)$ operations total.

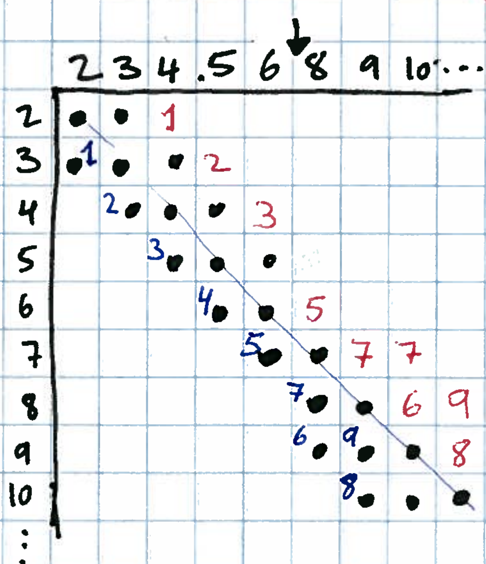
- Numerical issues:
  - Avoid divisions by 0 (& by small numbers) when computing c, s. This is easy.
  - Zero in a_{ii}: set x_i arbitrarily (e.g. $x_i = 0$) because $r_i = b_i$ anyway. Then go on.

    Small nonzeros a_{ii} are hard to deal with; advanced

Solving Sparse Least Squares Problems

In the smooth-function reconstruction problem, A has plenty of zeros to begin with; we can exploit that.





Evolution of the nonzero structure of the matrix

order of elimination

order of filling in

For this problem, if we only constrain the values of f(x) at a constant # points, total cost is $O(N)$.

There are clever elimination-ordering algs for other sparsity patterns, for general sparse matrices, and for other factorizations.

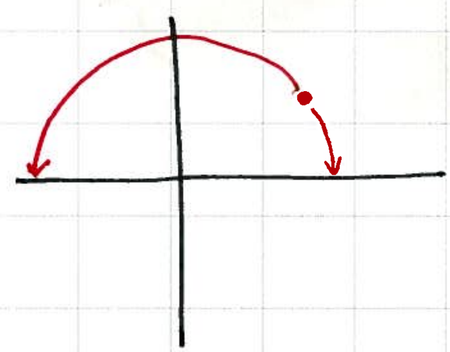
Additional numerical issues in Givens Rotations

- Solving a quadratic $ax^2+bx+c=0$

$$x_{1,2} = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$$

make $-b$ and $\pm\sqrt{b^2-4ac}$ have same sign; avoid subtracting large #'s to recover a small difference

- The choice boils down to which of 2 rotations to use:



- In Python:

```
x=1.0
y=1e-17
```

10^{-17}

```
x+y-x  => 0.0
x-x+y  => 1e-17
```

} floating point arithmetic is not commutative; other axioms that hold.