The Conjugate Gradient Algorithm

CS 292F

May 12, 2021

Lecture 12

Look for a solution to $Ax=b$ in

$$K_t(b) = \text{span}\{b, Ab, \dots, A^{t-1}b\}$$

If $q(z) = 1 - z\,p(z)$ is a polynomial with $q(0)=1$ and $q(\lambda_i)=0$ for every eigenvalues λ_i of A, then:

$$p(A)\,b = x.$$

q : $n+1$ constraints $\Rightarrow$ degree n.

$\Rightarrow$ p degree $n-1$ $\Rightarrow$ $x \in K_n(b)$

Actually $x \in K_t(b)$ where

t = # of <u>distinct</u> evals of A.

Theorem: The diameter of a connected graph is at most the number of different eigenvalues of its Laplacian.

Proof: If $b = \mathbb{1}_a = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$,

then Lb is nonzero only at a & its neighbors.

$L^k b$ is nonzero within k steps of a.

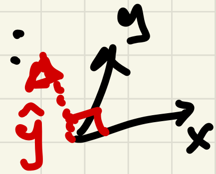
Element $b(a)$ affects only $L^k b$ in positions within dist. k.

If $Lx = b$, every elt of x depends on every elt of b (if connected)

so if $x \in K_t(b)$ then

$t \geq$ diameter of graph.

Given vectors x, y, we can <u>orthogonalize</u> y against x:



by $$\hat{y} = y - x\frac{x^Ty}{x^Tx}.$$

Note: $$x^T\hat{y} = \cancel{x^Ty} - \cancel{x^Tx}\,\frac{\cancel{x^Ty}}{\cancel{x^Tx}} = 0$$

HERE WE USE A DIFFERENT ORTHOGONALIZATION RELATED TO THE <u>A-norm</u> of vectors.

DEF If A is symm. pos. def., the A norm of vector x is

$$\|x\|_A = \sqrt{x^T A x} \quad \left(\text{recall } \|x\| = \sqrt{x^T x}\right)$$

Notice $\|x\|_A = \|A^{1/2} x\|$

where $A^{1/2} = \sum_{i=1}^{n} \lambda_i^{1/2} w_i w_i^T$

DEF x, y are A-orthogonal if $x^T A y = 0$

A-orthogonalization: given x, y

let $\hat{y} = y - x \dfrac{x^T A y}{x^T A x}$.

Then $x^T A \hat{y} = x^T A y - x^T A x \dfrac{x^T A y}{x^T A x} = 0$

CG will build an A-orthogonal basis for the Krylov subspace.

CG builds an A-ortho basis for $K_t(b)$,
<u>and</u> finds an "<u>optimal</u>" $x_t \in K_t(b)$.

Look at error in x_t. $e_t = x_t - x$ ← $Ax=b$

← the real answer

$$\|x_t - x\|_A^2 = (x_t - x)^T A (x_t - x)$$

$$= x_t^T A x_t - 2 x_t^T A x + x^T A x.$$

$$= x_t^T A x_t - 2 x_t^T b + x^T b$$

THEREFORE WE MINIMIZE:

$$\frac{1}{2} x_t^T A x_t - x_t^T b$$

... at each step t (in the space $K_{t+1}(b)$).

(FOLLOWING SPIELMAN § 35.3)

Let $p_0, \dots, p_t$ be an A-orthogonal basis of $K_{t+1}(b)$.

Let $$x_t = \sum_{i=0}^{t} c_i p_i$$ unknown coefficients

want to find the c_i that minimize $\frac{1}{2} x_t^T A x_t - x_t^T b$

$$\frac{1}{2} x_t^T A x_t - b^T x_t = \frac{1}{2}\left(\sum_{i=0}^{t} c_i p_i\right)^T A \left(\sum_{i=0}^{t} c_i p_i\right) - b^T \sum_{i=0}^{t} c_i p_i$$

$$= \frac{1}{2}\sum_{i=0}^{t} c_i^2 p_i^T A p_i + \frac{1}{2}\sum_{i \neq j} c_i c_j \underbrace{p_i^T A p_j}_{=0} - \sum_{i=0}^{t} c_i b^T p_i$$

$$= \sum_{i=0}^{t}\left(\frac{1}{2} c_i^2 p_i^T A p_i - c_i b^T p_i\right)$$

The terms in this sum don't depend on each other, so we can minimize the sum by minimizing each term in c_i separately.

Minimize (over c_i) $\frac{1}{2} c_i^2 p_i^T A p_i - c_i b^T p_i$.

min where $\frac{d}{dc_i} = 0$.

$$\frac{d}{dc_i}(*) = c_i p_i^T A p_i - b^T p_i$$

$$= 0 \quad \text{at} \quad c_i = \frac{b^T p_i}{p_i^T A p_i} \quad (*) .$$

ALGORITHM: Take $p_0 = b$

For $t = 0, 1, 2, \ldots$:

- compute Ap_t and A-orthogonalize it against the earlier p_i to get p_{t+1}
- compute c_{t+1} by $(*)_t$
- compute $x_{t+1} = \sum_{i=1} c_i p_i$

A-orthogonalize Ap_t against $p_0 \cdots p_t$.

$$p_{t+1} = (Ap_t) - \sum_{i=0}^{t} p_i \frac{(Ap_t)^T A (Ap_i)}{p_i^T A p_i} \qquad \frac{y^T A x}{x^T A x}$$

CLAIM: FOR $i < t-1$ then

$$(Ap_t)^T A p_i = 0$$

(THAT IS, Ap_t is already A-ortho to p_i)

$(Ap_t)^T A p_i = p_t^T A (A p_i)$, and

$$Ap_i \in \text{span}(p_0, p_1, \cdots, p_{i+1})$$

so, Ap_i is a linear combination of $p_0 \cdots p_{i+1}$, and for $i+1 < t$ these are already all A-ortho to p_t.

Thus

$$p_{t+1} = Ap_t - p_t \frac{(Ap_t)^T A p_t}{p_t^T A p_t} - p_{t-1} \frac{(Ap_t)^T A p_{t-1}}{p_{t-1}^T A p_{t-1}}$$

Then get c_{t+1} from ⊛ and x_{t+1}

Then

$$c_{t+1} = \frac{b^T p_t}{p_t^T A p_t}$$

and

$$x_{t+1} = x_t + c_{t+1} p_{t+1}.$$

This is an algorithm!

The rest is "programming".

Per iteration, this is:

$\Theta(1)$ matrix-vector products (1)

$\Theta(1)$ vector dot products etc. (a few)

So t iters costs

$\Theta(t)$ matvec + vector ops.