Recall: Goal is to minimize: $\Phi(x) = \frac{1}{2}x^T A x - b^T x$

$\uparrow$ sym. PD

- Initialize $x^{(0)}$; $r_0 = Ax^{(0)} - b = \nabla\Phi(x^{(0)})$

  $p_0 = -r_0$

- $\alpha_t = \dfrac{r_t^T r_t}{p_t^T A p_t}$

- $x^{(t+1)} = x^{(t)} + \alpha_t p_t$

- $r_{t+1} = r_t + \alpha_t A p_t$

- $\beta_{t+1} = \dfrac{r_{t+1}^T r_{t+1}}{r_t^T r_t}$

- $p_{t+1} = -r_{t+1} + \beta_{t+1} p_t$

CG (version 1)

Can we improve CG (in the quadratic setting $\phi(x) = \frac{1}{2} x^T A x - b^T x$?

PD, sym.

Idea : Preconditioning

First, some background :

* Recall that we seek x^* : $Ax^* = b$

sym. PD, $n \times n$

* It will turn out that the performance of CG depends on the eigenvalues of A.

How? Define $\|z\|_A = \sqrt{z^T A z}$

$z \in \mathbb{R}^n$

e.g. If $A = I$, then $\|z\|_A = \|z\|$
$= \sqrt{z^T z}$

$\boxed{\begin{array}{l}\text{Theorem: If } A \text{ has eigenvalues} \\ 0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n, \text{ then} \\ \|x^{(t+1)} - x^*\|_A^2 \leq \left(\dfrac{\lambda_{n-t} - \lambda_1}{\lambda_{n-t} + \lambda_1}\right)^2 \|x^{(0)} - x^*\|_A^2\end{array}}$

This implies, choosing $t+1 = n \Leftrightarrow t = n-1$

$$\|x^{(n)} - x^*\|_A^2 \leq 0 . \|x^{(0)} - x^*\|_A^2 = 0$$

That is, convergence within n-steps!

Additionally, if for example

$\lambda_{n-1} = \lambda_n$, then convergence to x^*

takes only $n-1$ steps!

It is also true that

$$\|x^{(t)} - x^*\|_A \le 2\left(\frac{\sqrt{\underbrace{\kappa(A)}_{\text{cond'n number.}}} - 1}{\sqrt{\kappa(A)} + 1}\right)^t \|x^{(0)} - x^*\|_A$$

· Compare with GD

Now, back to preconditioning!

Idea is to convert the problem to an equivalent one with "better" eigenvalues $\Rightarrow$ better convergence

How? $\quad \Phi(x) = \frac{1}{2}x^T A x - b^T x$

Define $\begin{cases} \hat{x} = Cx \\ \hat{\Phi}(\hat{x}) = \frac{1}{2}\hat{x}^T \overbrace{(C^{-T} A C^{-1})}^{\hat{A}} \hat{x} - \underbrace{b^T C^{-1}}_{\hat{b}^T} \hat{x} \end{cases}$

notation: $C^{-T} = (C^{-1})^T$.

So now, we can solve for $\hat{x}$, then when done, solve for x

$(\hat{x} = Cx)$.

In particular, convergence now depends on eigenvalues of $C^{-T}AC^{-1}$

$$\left(\hat{\Phi}(\hat{x}) = \frac{1}{2}\hat{x}^T \hat{A} \hat{x} - \hat{b}^T \hat{x} \right)$$

$\hookrightarrow C^{-T}AC^{-1}$

Idea: Choose C so that

$$\kappa(C^{-T}AC^{-1}) \ll \kappa(A)$$

This leads to an modification of CG which we call

preconditioned CG

Given $M = C^T C$

Initialize $x^{(0)}$, $r_0 = Ax^{(0)} - b = \nabla\Phi(x_0)$

- Solve $My_0 = r_0$ to get y_0
- set $p_0 = -y_0$

While $r_t \neq 0$ or too big

Preconditioned CG

- $\alpha_t = \dfrac{r_t^T y_t}{p_t^T A p_t}$
- $x^{(t+1)} = x^{(t)} + \alpha_t p_t$
- $r_{t+1} = r_t + \alpha_t A p_t$
- Solve $My_{t+1} = r_{t+1}$ for y_{t+1}
- $\beta_{t+1} = \dfrac{r_{t+1}^T y_{t+1}}{r_t^T y_t}$
- $p_{t+1} = -y_{t+1} + \beta_{t+1} p_t$

Remarks: • If $M = I$, then this is just CG.

• Extra computational effort: need to solve $My_t = r_t$ at each step.

In practice, to reduce the cost of solving $My = r$ we design M so that $My = r$ can be solved quickly while simultaneously having favorable eigenvalue properties.

For example, we can pick

$M = \tilde{L}\tilde{L}^T$ where $\tilde{L}$ is a sparse approximation to L which is the Cholesky factor associated with A.

That is, write $A = LL^T$ (A is PD)

$L \rightsquigarrow \tilde{L}$ (sparser, more zeros in $\tilde{L}$)

$$\Rightarrow C^{-T} A C^{-1} \approx -\tilde{L}^{-1} A \tilde{L}^{-1}$$

$$\approx I$$

↑ low cond'n number

Won't go into more details on preconditioning, more details in numerical linear algebra class)

Nonlinear CG (for solving general optimization problems)

$\min F(x)$

(1) Recall that in CG we select d_t to minimize

$$\Phi(x_t + a p_t) \longleftarrow \text{quadratic}$$

and we had a closed form expression for d_t

- Now, we don't have this closed form expression anymore since $\Phi(x)$ is now replaced by a general function $F(x)$

$\Longrightarrow$ Line search!

(2) Previously we had

$$r_t = \nabla \Phi(x^{(t)}) = A x^{(t)} - b$$

Instead we now use

$$r_t = \nabla F(x^{(t)})$$

With these 2 stages, we get the Fletcher-Reeves version of CG

Algorithm (FR):

Initialize $x^{(0)}$; set $F_0 = F(x^{(0)})$

$$\nabla F_0 = \nabla F(x^{(0)})$$

$$P_0 = -\nabla F_0$$

While $\nabla F_t \neq 0$, or ∇F_t too big

- Find α_t using line-search
- set $x^{(t+1)} = x^{(t)} + \alpha_t P_t$
- Evaluate $\nabla F_{t+1} = \nabla F(x^{(t+1)})$
- $\beta_{t+1}^{FR} = \dfrac{\nabla F_{t+1}^T \nabla F_{t+1}}{\nabla F_t^T \nabla F_t}$
- $P_{t+1} = -\nabla F_{t+1} + \beta_{t+1}^{FR} P_t \quad (*)$

Remarks: Only needs gradient evals & inner prods

$\Rightarrow$ similar cost to GD

But, how do we pick α_t?

The issue is that $(*) \Rightarrow$

$$(**) \quad \nabla F_t^T P_t = -\|\nabla F_t\|^2 + \beta_{t+1}^{FR} \nabla F_t^T P_{t-1}$$

so for P_t to be a descent direction

we need $\nabla F_t^T P_t$ to be negative

Good news: If α_t minimizes

$$F(x^{(t)} + a P_{t-1})$$

then $\nabla F_t^T P_{t-1} = 0$ (why? "Chain rule")

so $(**) \Rightarrow \nabla F_t^T P_t < 0$

Bad news : $F(x^{(t)} + a p_{t-1})$ may be difficult to minimize perfectly so α_t, coming from a line search may not guarantee that

$$\nabla F_t^T p_t < 0$$

Solution : Make sure Wolfe cond's are satisfied when solving for α_t :

$$(a)\ F(x^{(t)} + \alpha_t p_t) \le F(x^{(t)}) + c_1 \alpha_t \nabla F_t^T p_t$$

$$\rightarrow (b)\ \left| \nabla F(x^{(t)} + \alpha_t p_t)^T p_t \right| \le -c_2 \underbrace{\nabla F_t^T p_t}_{-ve}$$

Here $0 < c_1 < c_2 < 1/2$

Can be shown that (b) $\Rightarrow$ (**) is neg.

There are variants of the FR-CG algorithm where β is chosen differently

Example: Polak-Ribière:

$$\beta_{t+1}^{PR} = \frac{\nabla F_{t+1}^T (\nabla F_{t+1}^T - \nabla F_t^T)}{\|\nabla F_t\|^2}$$

(does not guarantee that P_t is a descent direction, however

$\beta_{t+1}^{+} = \max\{\beta_{t+1}^{PR}, 0\}$ fixes this (with wolfe cond's).