04/03/18. Newton, BFGS, NLCG.
Ref: Nocedal, Wright

$\min f(x)$:

$$f(x+p) = f(x) + p^T \nabla f(x) + \frac{1}{2} p^T \nabla^2 f(x) p + \cdots$$

for $\|p\|=1$, min when $p = \frac{\nabla f(x)}{\|\nabla f(x)\|}$

for $\nabla^2 f(x) \succ 0$ (Hessian) min when

$$0 = \nabla f(x) + \nabla^2 f(x) p$$
$$\Rightarrow p = -(\nabla^2 f(x))^{-1} \nabla f(x)$$

- Newton's method:

$$x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1} \nabla f(x_k)$$

Rmk: if $\nabla^2 f(x) \not\succ 0$, can still solve

$$\min_p \; p^T \nabla f(x) + \frac{1}{2} p^T \nabla^2 f(x) p$$
$$\text{s.t. } \|p\| \le R$$

→ trust-region method.

Can do line search $x_{k+1} = x_k - \alpha_k (\nabla^2 f(x_k))^{-1} \nabla f(x_k)$

to guarantee descent when $\nabla^2 f(x_k) \succ 0$.

Quadratic (local) convergence:

$$\|x_{k+1} - x^*\| \leq C \|x_k - x^*\|^2$$

→ doubles number of correct digits at every iteration

Difficulty: $\nabla^2 f(x_k)$ is a large matrix (to store & invert).

- Quasi-Newton methods:

approx. $\nabla^2 f(x_k)$ by B_k
or $(\nabla^2 f(x_k))^{-1}$ by H_k.

Idea: info about $\nabla^2 f(x)$ contained in changes of $\nabla f(x)$:

$$\underbrace{\nabla f(x+p)}_{\nabla f_{k+1}} = \underbrace{\nabla f(x)}_{\nabla f_k} + \underbrace{\nabla^2 f(x)}_{\nabla^2 f_k} \underbrace{p}_{x_{k+1} - x_k} + o(\|p\|)$$

$$\nabla^2 f_k \underbrace{(x_{k+1} - x_k)}_{s_k} \simeq \underbrace{\nabla f_{k+1} - \nabla f_k}_{y_k}$$

Choose B_{k+1}:
- $B_{k+1} s_k = y_k$ (secant eq.) (1)
- B_{k+1} low-rank modif. of B_k (2)
- $B_{k+1}^T = B_{k+1}$ (3)
- $B_{k+1} > 0$ when $s_k^T y_k > 0$ (4)

necessary, true when f strongly convex
true in a line search

Broyden, Fletcher, Goldfarb, Shanno (BFGS):

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$$

(rank-2 update)

Better yet, update $H_k = B_k^{-1}$ (from Woodbury formula)

$$H_{k+1} = (I - \rho_k s_k y_k^T) H_k (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T,$$

$$\rho_k = \frac{1}{y_k^T s_k}$$

Choose $H_0 = I$, for instance, or some other multiple of I.

then $x_{k+1} = x_k - \alpha_k H_k \nabla f_k$

Limited-memory BFGS: (LBFGS)

assemble H_k by keeping $\{s_i, y_i\}$ in memory for $i \le k$, discard $i \le k-m$

Use $k-m < i \le k$ for $H_k \nabla f_k$

Under smoothness & convexity assumptions, superlinear convergence: $\|x_{k+1} - x^*\| / \|x_k - x^*\| \to 0$

- Nonlinear conjugate gradients:

Linear CG designed to min $\phi(x)$,

$\phi(x) = \frac{1}{2} x^T A x - b^T x \qquad A > 0$

$\nabla \phi(x) = \boxed{Ax - b =: r}$ (residual)

* Would be simple if A were diagonal (coordinate-wise minimization)

$$x_{k+1} = x_k + \alpha_k p_k \quad \text{with} \quad \alpha_k = -\frac{r_k^T p_k}{p_k^T A p_k} = \operatorname{argmin}_\alpha \phi(x_k + \alpha p_k)$$

Directions p_k are conjugate: $p_i^T A p_j = 0$ $\quad \langle p_i, p_j \rangle_A$

Why?

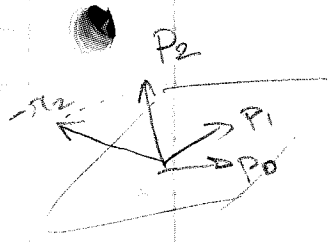
Change var: $S = [p_0\ p_1 \cdots p_{n-1}]$

$\hat{x} = S^{-1} x$ (coefficients in new basis such that $x = S\hat{x}$)

then $\phi(S\hat{x}) = \frac{1}{2} \hat{x}^T S^T A S \hat{x} - (S^T b)^T \hat{x}$

(i,j) element $= p_i^T A p_j \Rightarrow$ diagonal

$\rightarrow$ makes the minimization simple



Specific choice of conjugate directions: Gram-Schmidt A-orthogonalization (in the sense of $\langle \cdot, \cdot \rangle_A$) of $-r_k$:

$$p_k = -r_k + \beta_k p_{k-1} + \underbrace{\text{l.c. of } p_{k-2}, \dots, p_0}_{=0 \text{ remarkably}}$$

$$0 = p_{k-1}^T A p_k \Rightarrow \beta_k = \frac{p_{k-1}^T A r_k}{p_{k-1}^T A p_{k-1}}$$

p_i, $i = 0, \dots k$ form a basis of $\operatorname{Span}\{b, Ab, A^2 b, \dots, A^k b\}$ = Krylov subspace.

Minimization of $\phi(x)$ is exact in growing Krylov subspace, does not need to be revisited.

Also, $\min_{P_k} \| x_0 + P_k(A) r_0 - x^* \|_A$

$\rightarrow$ optimal in Krylov subspace, in the sense of the A norm

CG algorithm. Have r_k, p_k ($r_0 = Ax_0 - b$, $p_0 = -r_0$)

	NLCG
(1) $\alpha_k = -\frac{r_k^T p_k}{p_k^T A p_k}$	✓ $\operatorname{argmin}_\alpha \phi(x_k + \alpha p_k)$, of inexact line search
(2) $x_{k+1} = x_k + \alpha_k p_k$	✓
(3) $r_{k+1} = A x_{k+1} - b$	✓ $\nabla\phi(x_k)$
(4) $\beta_{k+1} = \frac{r_{k+1}^T A p_k}{p_k^T A p_k}$	× involves A, not ϕ.
(5) $p_{k+1} = -r_{k+1} + \beta_{k+1} p_k$	✓

To get nonlinear CG, find stand-alone formula for β_{k+1}. Make use of all the others ((1),(2),(3),(5)), alongside $\begin{cases} r_k^T r_i = 0 & i < k \\ r_k^T p_i = 0 & i < k \end{cases}$

$$\Rightarrow \beta_{k+1} = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$$ (\$, Nocedal-Wright p. 111)

$$= \frac{\nabla\phi(x_{k+1})^T \nabla\phi(x_{k+1})}{\nabla\phi(x_k)^T \nabla\phi(x_k)}$$ (new 4)

→ Fletcher-Reeves method.

Good because does not require storing a matrix, or history of gradients like BFGS. Linear convergence rate.

Polak-Ribière: $\beta_{k+1} = \frac{\nabla\phi_{k+1}^T (\nabla\phi_{k+1} - \nabla\phi_k)}{\nabla\phi_k^T \nabla\phi_k}$

also reduces to CG in the linear case