Quasi Newton methods:

we have a model function

$$m_k = f_k + \nabla f_k p + \frac{1}{2} p' B_k p$$

B_k is symmetric p.d

B_k could be Hessian, but that isn't attractive (not pd ? derivatives hard? ~~etc~~

instead, we will update the model

minimizer is: $p_k = -B_k^{-1} \nabla f_k$

next point is:

$$x_{k+1} = x_k + \alpha_k p_k$$

where α_k satisfies Wolfe conds.

Idea: update B_k

$$m_{k+1} = f_{k+1} + \nabla f_{k+1} p + \frac{1}{2} p' B_{k+1} p$$

→ require gradient of m_{k+1} match gradient of f_{k+1} at k, $k+1$

$k+1$ already ok

k:

$$m_{k+1}(-\alpha_k p_k) = \nabla f_{k+1} - \alpha_k B_{k+1} p_k = \nabla$$

or

$$\alpha_k B_{k+1} p_k = \nabla f_{k+1} - \nabla f_k$$

write $s_k = \alpha_k p_k$, $y_k = \nabla f_{k+1} - \nabla f_k$

Then $B_{k+1} s_k = y_k$

Notice for this to be true $s_k^T y_k > 0$

(But wolfe conds will ensure this; which follows from B_{k+1} p.d.)

1) NO UNIQUE B_{k+1}
2) could require $\|B_{k+1} - B_k\|$ small in ? norm

. careful choice of norm gives

$$B_{k+1} = \left(I - \gamma_k y_k s_k^T\right) B_k \left(I - \gamma_k s_k y_k^T\right) + \underline{y_k y_k^T} \cdot \gamma_k$$

where $\gamma_k = \dfrac{1}{s_k^T y_k}$

This update is DFP.

Notice, we care about $H_k = B_k^{-1}$

$$H_{k+1} = H_k - \frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k} + \frac{s_k s_k^T}{y_k^T s_k}.$$

This is effective, but can do better.

BFGS:

instead of constraining B_k, constrain H_k $(= B_k^{-1})$

we had $B_k s_k = y_k$

Now, instead, require

$$H_{k+1} y_k = s_k$$

which gives

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

where $\gamma_k = \frac{1}{y_k^T s_k}$

This is best, gives superlinear convergence

What is H_0?

options:
- the Hessian inverse
- Identity
- Scaled Identity
- 1/diag of Hessian.

Big attraction — Limited memory versions

BFGS:

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

(α_k: from line search)

$$H_{k+1} = V_k^T H_k V_k + \gamma_k s_k s_k^T$$

$$\gamma_k = \frac{1}{y_k^T s_k} \qquad V_k = \left(I - \gamma_k y_k s_k^T\right)$$

Notice we could apply recursively:

$$H_{k+1} = V_k^T \left[\left(V_{k-1}^T H_{k-1} V_{k-1} + \gamma_{k-1} s_{k-1} s_{k-1}^T \right] V_k + \gamma_k s_k s_k^T$$

etc.

Notice also:

- multiply by $S_k S_k^T$ is vector mult.
  (ie $S_k S_k^T x = S_k (S_k^T x)$)
- V_k is also easy
  $(I - \gamma_k y_k S_k^T) x = x - y_k \cdot (\gamma_k (S_k^T x))$
- recording state is just a question of keeping y_k, S_k, γ_k

IDEA:

- store last m (y_k, S_k, γ_k)
- throwing away oldest
- have an initial H_{k-m} for each step
  (identity; scaled identity; diag)
- This is L-BFGS

METHOD OF CHOICE FOR BIG UNCONSTRAINED PROB