

Recall: Problems with basic Newton

- (i) Needs derivatives: may be impossible or expensive to compute
- (ii) Needs solution of a linear system each iteration. Demanding for large problems
- (iii) Converges only when starting close enough
- (iv) The Hessian may be singular or indefinite. Can cause problems to solve the linear system, or can cause Newton direction to be non descent

Problem (iv) “cured” by Modified Cholesky, guaranteeing descent

0

1 / 7

Globalization—line search (problem (iii))

- ▶ A general descent method: $x_{k+1} = x_k + \alpha_k p_k$, where p_k is a descent direction
- ▶ p_k descent direction: $f(x_k + \alpha p_k) < f(x_k)$ for $\alpha > 0$ small enough
- ▶ Too large α may cause $f(x_k + \alpha p_k) > f(x_k)$!
- ▶ The **line search** subproblem: $\min_{\alpha > 0} f(x_k + \alpha p_k)$:
- ▶ Not enough to demand $f(x_k + \alpha p_k) < f(x_k)$ (to guarantee convergence). Convergence theory needs a stronger condition, such as the **Armijo condition**:

$$f(x_k + \alpha p_k) \leq f(x_k) + \alpha \mu p_k^T \nabla f(x_k)$$

Very small $\mu = 10^{-4}$ OK.

0

2 / 7

Damped Newton

Line search with Newton’s method is called **Damped Newton**

Backtracking line search:

- ▶ Start with $\alpha = 1$
- ▶ Check the Armijo condition
- ▶ If needed, reduce α until the Armijo condition is satisfied.
- ▶ $\alpha < 1$ only needed when far from the solution. When close enough, Newton will converge without step size reduction.

0

3 / 7

Damped Newton with modified Cholesky

Algorithm

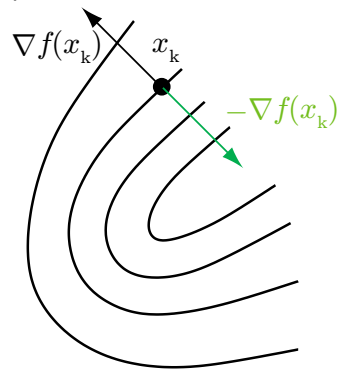
1. Choose initial guess x_0 , tolerance ϵ , and parameter μ
2. For $k = 0, 1, \dots$
 - 2.1 If $\|\nabla f(x_k)\| \leq \epsilon$ **stop**
 - 2.2 Compute modified factorization LL^T of $\nabla^2 f(x_k)$
($LL^T = E + \nabla^2 f(x_k)$ for some diagonal $E \geq 0$)
 - 2.3 Solve $LL^T s_k = -\nabla f(x_k)$
 - 2.4 $\alpha_k \leftarrow 1$
 - 2.5 While $f(x_k + \alpha_k s_k) > f(x_k) + \mu \alpha_k s_k^T \nabla f(x_k)$,
 $\alpha \leftarrow \alpha/2$
 - 2.6 Set $x_{k+1} \leftarrow x_k + \alpha s_k$

0

4 / 7

The steepest-descent method

Computing the Newton step with the crude approximation $\nabla^2 f(x_k) \approx I$ yields $s_k = -(\nabla^2 f(x_k))^{-1} \nabla f(x_k) \approx -\nabla f(x_k)$



The negative gradient:

- ▶ Is perpendicular to the level curves
- ▶ Points in the steepest downhill direction

The steepest-descent direction is the best direction *locally*

0

5 / 7

The steepest-descent algorithm:

- ▶ Easy to implement
- ▶ Does not need second derivatives: useable for large problems
- ▶ Only linear convergence rate, even with exact line search
- ▶ Terribly inefficient for *ill-conditioned* problem: large condition numbers of the Hessian
- ▶ Sensitive to **scaling** of the variables (Newton's method is **not** sensitive to scaling)
- ▶ Usually much better to use **the conjugate gradient or a quasi-Newton algorithm**

0

6 / 7

Quasi-Newton methods

Exact relation:

$$f(x) = f(x_k) + (x - x_k)^T \nabla f(x_k) + \frac{1}{2}(x - x_k)^T \nabla^2 f(\xi)(y - x),$$

where $\xi = \alpha x_k + (1 - \alpha)x$ for some $\alpha \in [0, 1]$.

Newton:

$$\phi_k^N(x) = f(x_k) + (x - x_k)^T \nabla f(x_k) + \frac{1}{2}(x - x_k)^T \nabla^2 f(x_k)(y - x)$$

Steepest-descent:

$$\phi_k^{SD}(x) = f(x_k) + (x - x_k)^T \nabla f(x_k) + \frac{1}{2}(x - x_k)^T I(y - x)$$

Quasi-Newton:

$$\phi_k^{QN}(x) = f(x_k) + (x - x_k)^T \nabla f(x_k) + \frac{1}{2}(x - x_k)^T B_k(y - x)$$

0

7 / 7