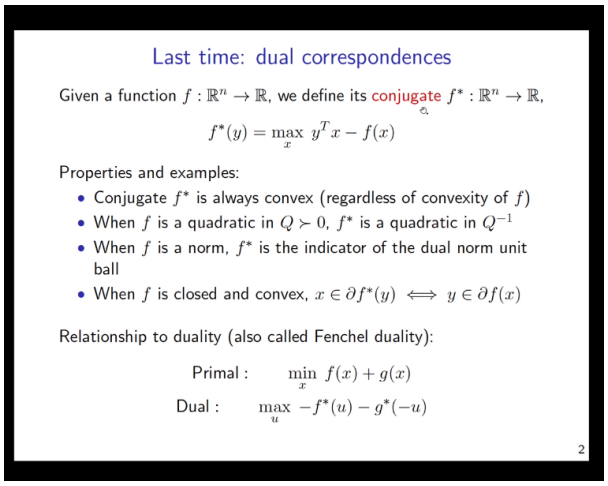



Key properties

1. Conjugate $f^*$is always convex in y
2. When f is closed and convex, $x \in \partial f^*(y) <\Longrightarrow y \in \partial f(x)$
    - if x is in the subdifferential of the conjugate for a given value, then the value is in the subdiff of the primal
3. $min_x\ f(x) + g(x) <\Longrightarrow max_u\ -f^*(u) - g*(-u)$

**Newton's method**

- this seems unrelated to duality, but the relationship will reveal itself!

Given smooth, unconstrained convex optimization

$min_x\ f(x)$

Assuming f is convex, twice differentiable, and dom(f) = $R^n$.

**Gradient descent:** $x^{(k)} = x^{(k-1)} - t_k . \nabla f\left(x^{(k-1)}\right)$

- move in the direction of negative gradient

**Newton's method:** $x^{(k)} = x^{(k-1)} - \nabla^2 f\left(x^{(k-1)}\right)^{-1} . \nabla f\left(x^{(k-1)}\right)$

- this is a different direction entirely! What is the motivation for this?
- the inverse hessian is itself a matrix, so this is a linear operation on the gardient

**A. Motivation**

Gradient descent can be interpreted as minimizing the quadratic approximation

$$f(y) \approx f(x) + \nabla f(x)^T (y-x) + \frac{1}{2t}||y-x||_2^2$$

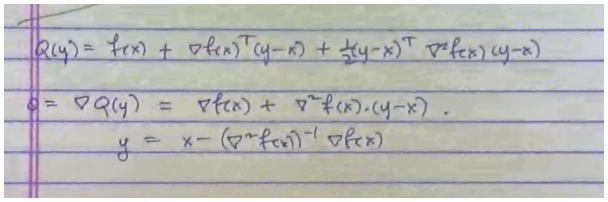
$$= f(x) + \nabla f(x)^T (y-x) + (y-x)^T \frac{1}{2t}(y-x)$$

where $\frac{1}{2t}$ does not say anything about the geometry of the real problem.

In contrast, Newton's method minimizes a better quadratic approximation

$$f(y) \approx f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(x)(y-x)$$

Minimizing over y yields the update step.

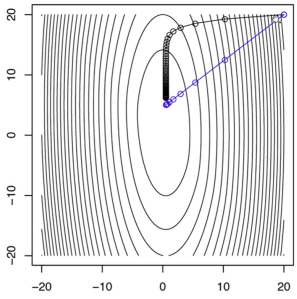



For a quadratic criterion function, Newton's method will yield the solution in one step - since the gradient would be linear in the optimization variable - so you would get the exact solution (?)

- because the quadratic approximation IS the real criterion function too!
-

Consider minimizing $f(x) = (10x_1^2 + x_2^2)/2 + 5\log(1 + e^{-x_1-x_2})$ (this must be a nonquadratic ... why?)

We compare gradient descent (black) to Newton's method (blue), where both take steps of roughly same length

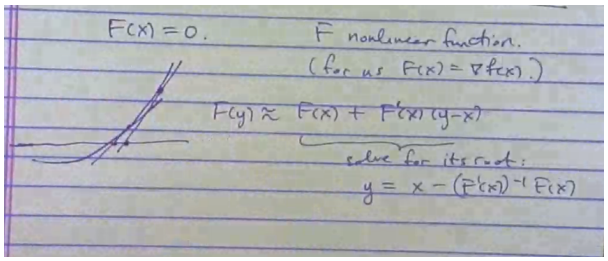



- Newton's method is intuitively better because it uses the curvature inherent to your problem, and not the assumption we make with GD i.e. that the curvature is some constant $\frac{1}{2t}$

Today

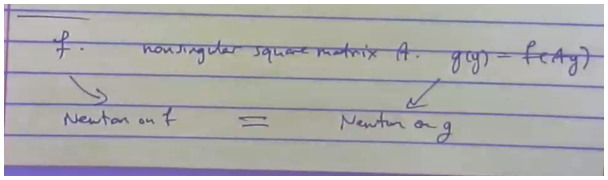
1. Interpretation
2. Backtracking line searing
3. Convergence
4. Equality constrained Newton
5. Quasi Newton

## 1. Interpretation



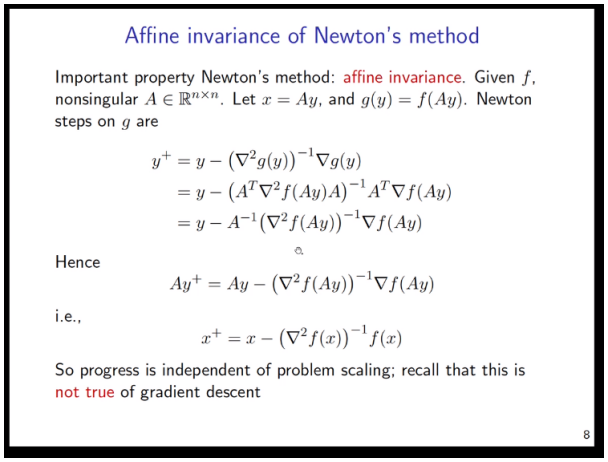

- Finding the roots of polynomial equations

**Distinguishing properties of Newton's method**
vs Gradient descent: Affine invariance




- doesn't just need to a linear transformation: can also be an affine transformation
- i.e. Newton update steps on f(y) = Newton update steps on g(y) = f(Ay + b)

Note: when doing matrix derivatives e.g. $\partial(Ax)$: take everything before the variable of interest and transpose it!



Affine invariance of Newton's method

Important property Newton's method: affine invariance. Given $f$, nonsingular $A \in \mathbb{R}^{n\times n}$. Let $x = Ay$, and $g(y) = f(Ay)$. Newton steps on $g$ are

$$\begin{aligned} y^+ &= y - (\nabla^2 g(y))^{-1} \nabla g(y) \\ &= y - (A^T \nabla^2 f(Ay) A)^{-1} A^T \nabla f(Ay) \\ &= y - A^{-1} (\nabla^2 f(Ay))^{-1} \nabla f(Ay) \end{aligned}$$

Hence

$$Ay^+ = Ay - (\nabla^2 f(Ay))^{-1} \nabla f(Ay)$$

i.e.,

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

So progress is independent of problem scaling; recall that this is not true of gradient descent

8

- if you scaled your problem, Newton's method would be just as good
- This is not true for gradient descent! Where problem conditioning matters a lot of it to be useful

– else you would just improve the performance by scaling your function by some amount

**Newton decrement**

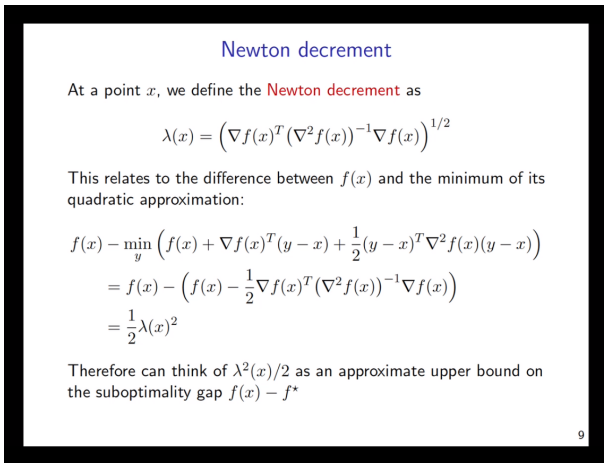
At a point x, the Newton decrement is $\lambda(x) = \left(\nabla f(x)^T \left(\nabla^2 f(x)\right)^{-1} \nabla f(x)\right)^{1/2}$

Interpretation 1: This is analogous to the duality gap, to the extent that you think that the quadratic approximation is helpful: the difference between $f(x)$ and the min of the quadratic approximation to f(x)

$$f(x) - min_y\left(f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(x)(y-x)\right)$$

$$= \frac{1}{2}\lambda(x)^2$$

This serves as an approximate upper bound on the suboptimality gap $f(x) - f_{opt}$




Interpretation 2:

The norm induced by a matrix $||x||_A = x^T Ax$

Seeing the Newton step as an update step,

$x^{(k)} = x^{(k-1)} - \nabla^2 f\left(x^{(k-1)}\right)^{-1} . \nabla f\left(x^{(k-1)}\right) \implies x^+ = x - t.v$

$where\ v = \nabla^2 f(x)^{-1} . \nabla f(x)$

Then, the length of the Newton update step in the norm defined by the hessian = square of the Newton decrement: $||v||_{\nabla^2 f(x)} = \lambda(x)^2$

- so in a sense, the Newton decrements tells you something about the size of each

update

Another interpretation of Newton decrement: if Newton direction is $v = -(\nabla^2 f(x))^{-1}\nabla f(x)$, then

$$\lambda(x) = \left(v^T \nabla^2 f(x) v\right)^{1/2} = \|v\|_{\nabla^2 f(x)}$$

i.e., $\lambda(x)$ is the length of the Newton step in the norm defined by the Hessian $\nabla^2 f(x)$

Note that the Newton decrement, like the Newton steps, are affine invariant; i.e., if we defined $g(y) = f(Ay)$ for nonsingular $A$, then $\lambda_g(y)$ would match $\lambda_f(x)$ at $x = Ay$

## 2. Backtracking line search

### Backtracking line search

So far what we've seen is called pure Newton's method. This need not converge. In practice, we use damped Newton's method (i.e., Newton's method), which repeats

$$x^+ = x - t\left(\nabla^2 f(x)\right)^{-1}\nabla f(x)$$

Note that the pure method uses $t = 1$

Step sizes here typically are chosen by backtracking search, with parameters $0 < \alpha \leq 1/2$, $0 < \beta < 1$. At each iteration, we start with $t = 1$ and while

$$f(x + tv) > f(x) + \alpha t \nabla f(x)^T v,$$

we shrink $t = \beta t$, else we perform the Newton update. Note that here $v = -(\nabla^2 f(x))^{-1}\nabla f(x)$, so $\nabla f(x)^T v = -\lambda^2(x)$

11

- same interpreation/intuition as backtracking for gradient descent

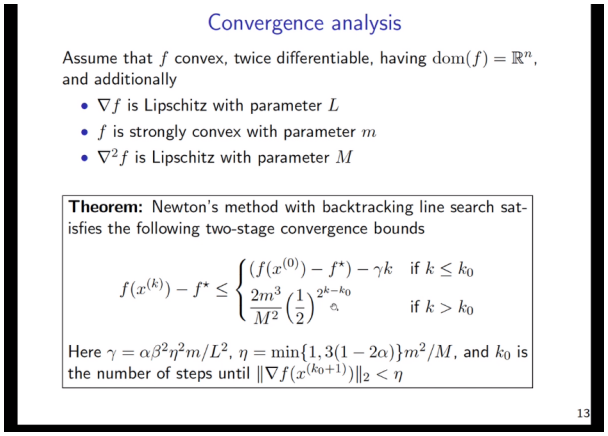
## 3. Convergence analysis

- extremely fast! 'Quadratic convergence' - $\frac{1}{2}^{2^k}$ !!
  - for quadratic problems, the optimum is found in one step, as discussed earlier

For a Hessian to be Lipschitz M means

$||\nabla^2 f(x) - \nabla^2 f(y)||_{op} \leqslant M\,||x - y||_2$

- operator norm = largest eigenvalue of the matrix
  - remember that the Hessian is a matrix!




- local quadratic convergence, since it only kicks in after a nuber of steps $k_0$
  - $k_0$ is the nummber of steps nbefore the the L2 norm of the Jacobian is less than $\eta$, i.e. it is is smaller than some value
- THe first phase is considered a 'damped phase', and the second is the 'pure' phase
- In the damped phase, we decrease the criterion by $\gamma k$ after every step k
  - backtracking picks some t $\neq$ 1
- In the pure phase, backtracking picks t = 1
  - and you never get out of this phase! so there are two distinct phases

In more detail, convergence analysis reveals $\gamma > 0$, $0 < \eta \leq m^2/M$ such that convergence follows two stages

- Damped phase: $\|\nabla f(x^{(k)})\|_2 \geq \eta$, and

$$f(x^{(k+1)}) - f(x^{(k)}) \leq -\gamma$$

- Pure phase: $\|\nabla f(x^{(k)})\|_2 < \eta$, backtracking selects $t = 1$, and

$$\frac{M}{2m^2}\|\nabla f(x^{(k+1)})\|_2 \leq \left(\frac{M}{2m^2}\|\nabla f(x^{(k)})\|_2\right)^2$$

Note that once we enter pure phase, we won't leave, because

$$\frac{2m^2}{M}\left(\frac{M}{2m^2}\eta\right)^2 \leq \eta$$

when $\eta \leq m^2/M$

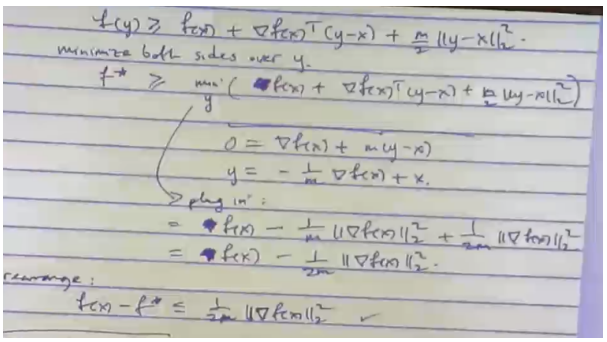
Note the decrease in the pure phase - this is where 'quadratic convergence' comes from.

Note: if a function is m-strongly convex, that means

$$f(y) \geqslant f(x) + \nabla f(x)(y-x) + \frac{m}{2}||y-x||_2^2$$

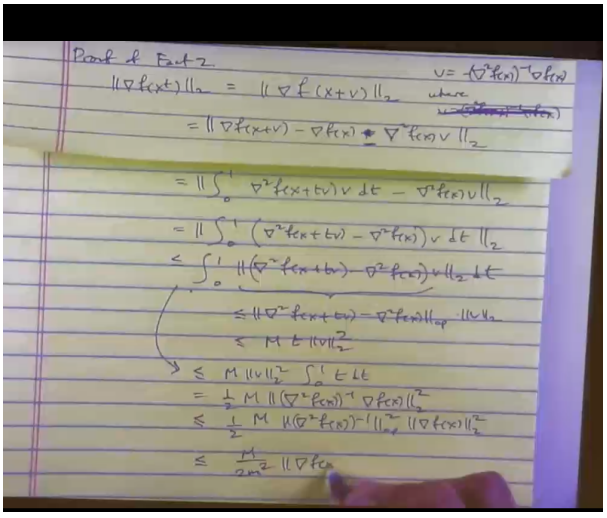
Fact 1: this also means that $f(x) - f_{opt} \leqslant \frac{1}{2m}||\nabla f(x)||_2^2$

- the criterion difference depends on the gradient at point x!

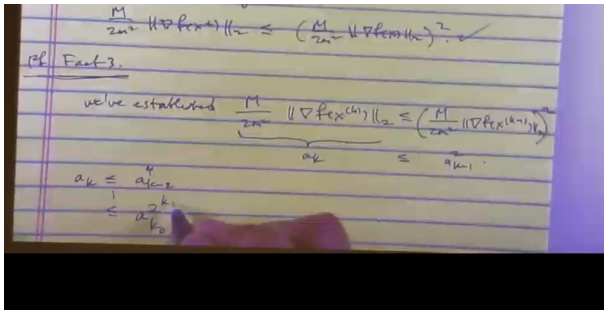


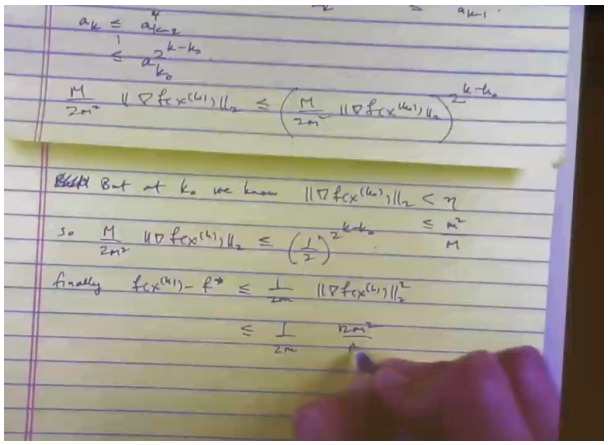

Fact 2: $\frac{M}{2m^2}||\nabla f(x^+)||_2 \leqslant \left(\frac{M}{2m^2}||\nabla f(x)||_2\right)^2$

- The L2 norm of the gradient of the updated step is less than the squared L2 norm of the gradient at the current step, modulo a scaling factor

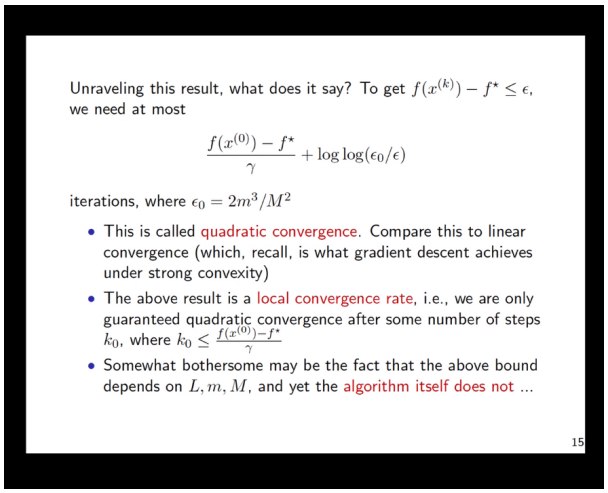



Fact 3: $f\left(x^{(k)}\right) - f_{opt} \leqslant \frac{2m^3}{M^2}\left(\frac{1}{2}\right)^{2^{k-k_0}}$





**Convergence analysis**



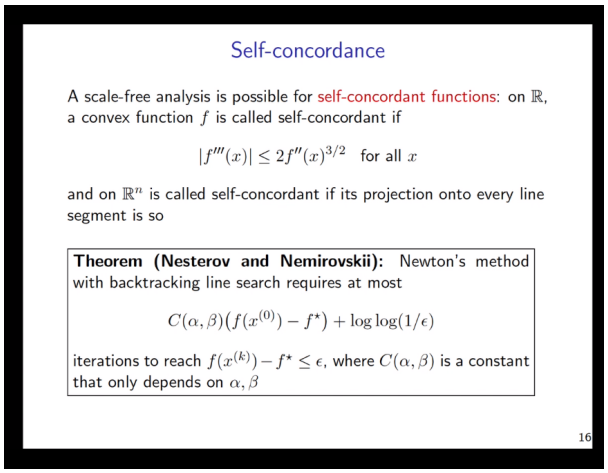

Quadratic convergence because $k_0 = \log\log(\epsilon_0/\epsilon)$

- just need to make this many steps in the quadratic convergence phase
- this is a local rate - this is how fast it converges once you are close
- the global rate is just linear!
- but in practice, you typically enter the quadratic convergence very quickly, especially with a warm start
  - this is what interior point methods do - you never leave this phase

There is a contradiction here - if Newton's method is truly scale invariant, then why does these bounds depend on the Lipschitz/strong convexity properties of the objective function?

**Self concordance**

the canonical self concordant function is $-log(x)$: gives equality for the bottom definition $|f'''(x)| \leqslant 2f''(x)^{3/2}$



Self-concordance

A scale-free analysis is possible for self-concordant functions: on $\mathbb{R}$, a convex function $f$ is called self-concordant if

$$|f'''(x)| \leq 2f''(x)^{3/2} \quad \text{for all } x$$

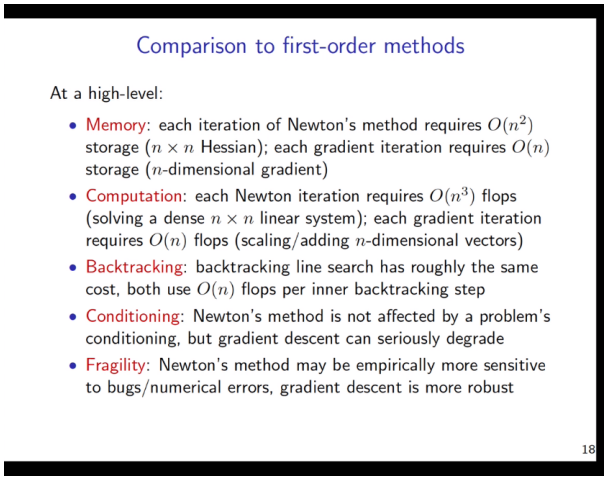
and on $\mathbb{R}^n$ is called self-concordant if its projection onto every line segment is so

**Theorem (Nesterov and Nemirovskii):** Newton's method with backtracking line search requires at most

$$C(\alpha, \beta)\big(f(x^{(0)}) - f^\star\big) + \log\log(1/\epsilon)$$

iterations to reach $f(x^{(k)}) - f^\star \leq \epsilon$, where $C(\alpha, \beta)$ is a constant that only depends on $\alpha, \beta$

16



Comparison to first-order methods

At a high-level:

- Memory: each iteration of Newton's method requires $O(n^2)$ storage ($n \times n$ Hessian); each gradient iteration requires $O(n)$ storage ($n$-dimensional gradient)
- Computation: each Newton iteration requires $O(n^3)$ flops (solving a dense $n \times n$ linear system); each gradient iteration requires $O(n)$ flops (scaling/adding $n$-dimensional vectors)
- Backtracking: backtracking line search has roughly the same cost, both use $O(n)$ flops per inner backtracking step
- Conditioning: Newton's method is not affected by a problem's conditioning, but gradient descent can seriously degrade
- Fragility: Newton's method may be empirically more sensitive to bugs/numerical errors, gradient descent is more robust

18