# CSI 436/536
# Introduction to Machine Learning

## Numerical optimization (1)

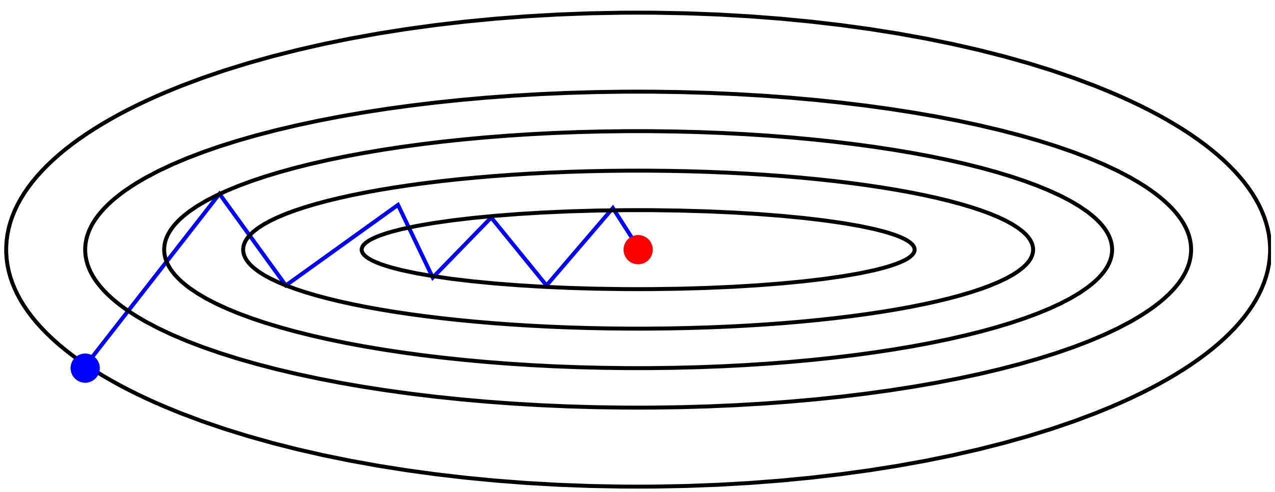
Professor Siwei Lyu
Computer Science
University at Albany, State University of New York

# Numerical optimization

- Direct methods: solution affords a closed-form and can be solved in one step
  - Example: LLSE, PCA, LDA, etc
- Iterative methods: solution is not in a closed-form equation, and needs to be obtained by iterative steps
  - Coordinate descent methods (Robust LLSE, LASSO)
  - Descent methods (every step reduce the objective)
    - Gradient descent (steepest descent) method
    - Newton's (varying metric) method
  - Non-descent methods
    - (Sub)gradient method
    - Stochastic (sub)gradient method

# descent algorithms

- an iterative algorithm, at each iteration
  - determine if **convergence** has been reached
  - determine a **direction** to go
  - determine a **step size** (how far to go along that direction)
  - together the direction and step size guarantees to **decrease** the objective function

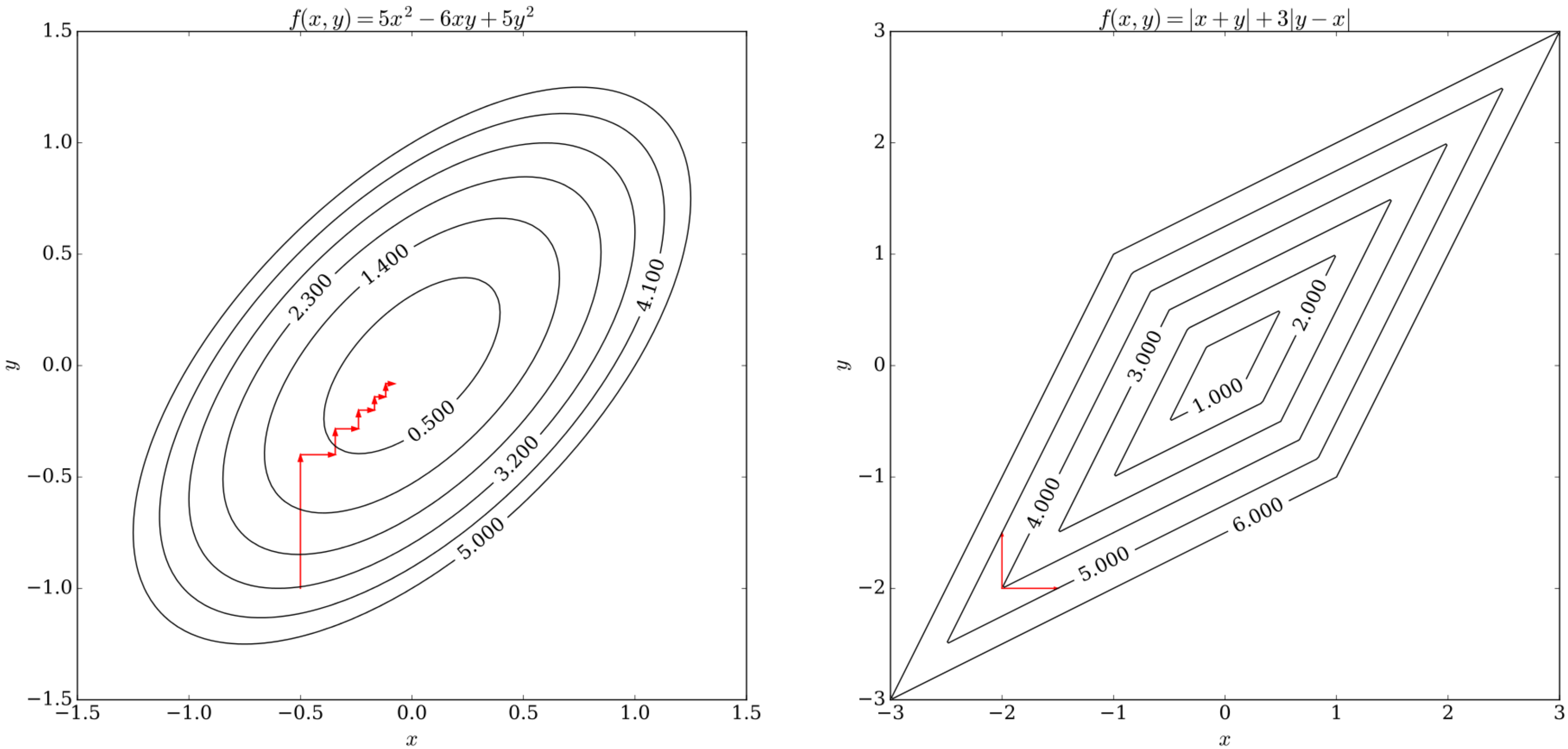


# minimizing by descending

- follow any descending direction with a step size
- general algorithm (local minimum)
  - initialize $t = 0$, $x_0$
  - while not converge
    - find a descending direction $\delta x$, such that $f(x) \geq f(x+\delta x)$
    - decide step size $\eta_t$
    - update $x_{t+1} = x_t + \eta_t \delta x$
    - $t = t+1$
  - end

# coordinate descent

- simple: each time solve 1D problem, no step size
- slow: convergence is $O(1/t^{1/d})$
- has trouble for non-smooth function



# General descent direction

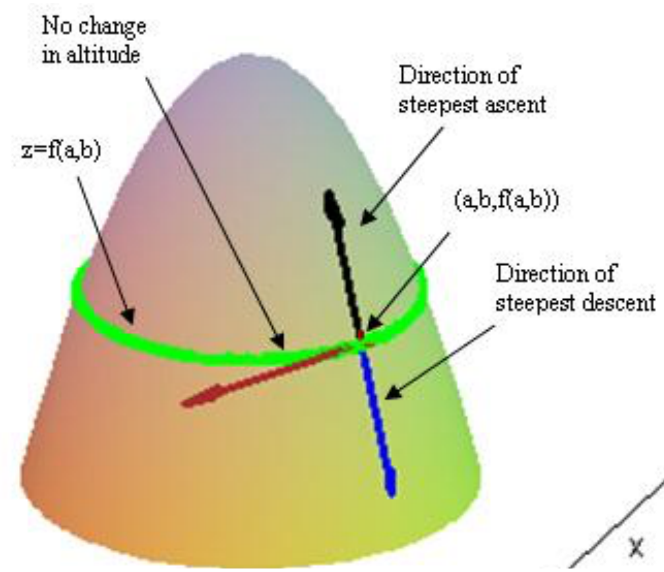
- Assume function f differential
  - expand function with first order Taylor series

    $$f(\mathbf{x}+\mathbf{d}) \doteq f(\mathbf{x}) + \mathbf{d}^T \nabla f(\mathbf{x})$$

  - we need $f(\mathbf{x}+\mathbf{d}) \leq f(\mathbf{x})$, so minimize $\mathbf{d}^T \nabla f(\mathbf{x})$
  - use Cauchy-Schwartz inequality we have

    $$-\|\mathbf{d}\|\|\nabla f(\mathbf{x})\| \leq \mathbf{d}^T \nabla f(\mathbf{x})$$

    minimum (equality holds) for $\mathbf{d} = - \nabla f(\mathbf{x})$
  - The negative gradient direction is the steepest descent direction
  - note that any direction with $\mathbf{d}^T \nabla f(\mathbf{x}) \leq 0$ is a descent direction



# determining step sizes

- precise line search
- backtrack search
- fixed step size
    - need objective function to have Lipschitz continuous gradients and step size smaller than 1/L to guarantee convergence
- variable step size (not a descent method)
  - may not guarantee decent of objective function
  - can still converge if choose carefully
    - we will discuss this case in (sub)gradient method

# precise line search

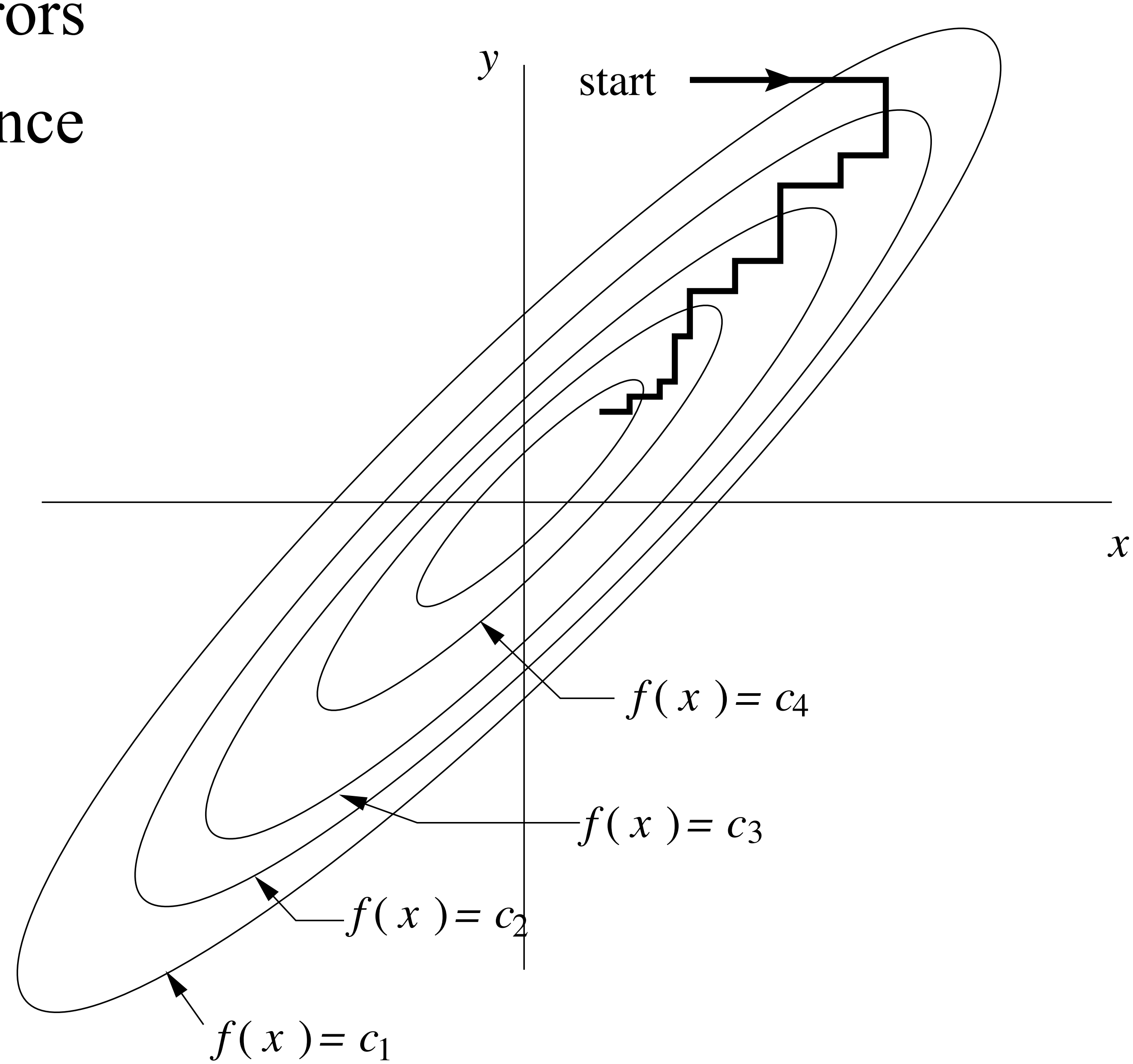
- precise line search (Cauchy principle)

$$\min_{\eta} f(\mathbf{x}_t + \eta\mathbf{d}_t)$$

  - 1D function of η, so can be computed exactly
  - derivation $\mathbf{d}_t^T\nabla f(\mathbf{x}_t + \alpha_t\mathbf{d}_t) = \mathbf{d}_t^T\nabla f(\mathbf{x}_{t+1}) = 0$
    - gradient at the next point orthogonal to descent direction
    - zigzag trajectory for gradient descent method

# convergence

- mathematical convergence $\nabla f(x_t) = 0$, difficult to check due to numerical errors
- numerical convergence
  - $0 < \|\nabla f(x_t)\| < \varepsilon$
  - $f(x) - f(x+\delta x) \geq \varepsilon$



# a different view of gradient descent

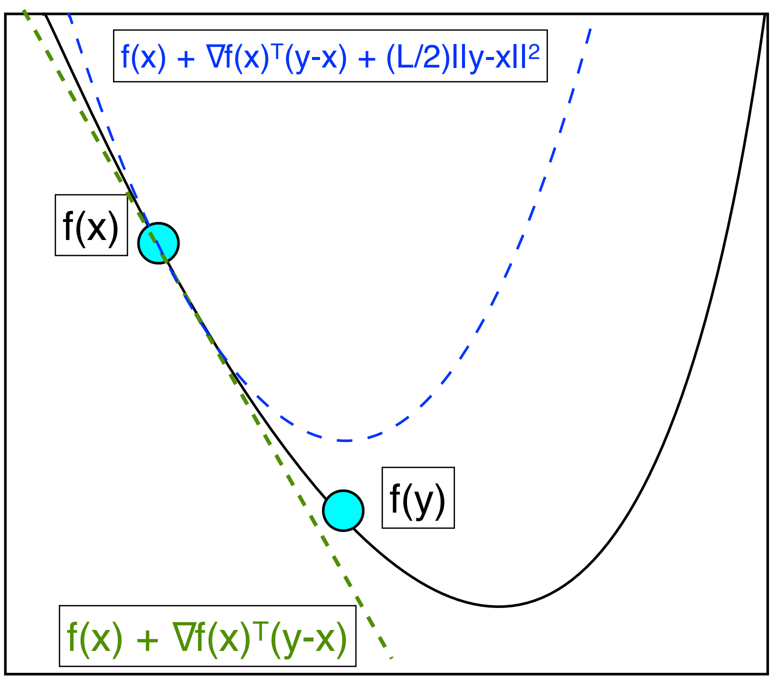
- gradient descent can be viewed as minimizing the quadratic approximation of objective f

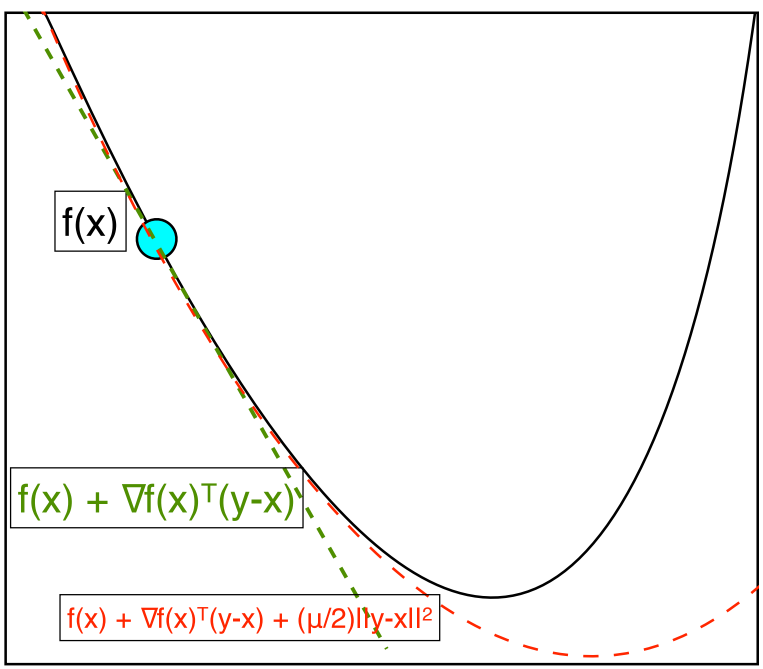
$$x^+ = \arg\min_y \left\{ f(x) + \nabla f(x)^T (y - x) + \frac{1}{2\alpha} \|y - x\|^2 \right\}$$

- we see the case when f has Lipschitz gradient and $\alpha <= 1/L$
  - the function being optimized is a majorization of function f, $F(x,y)$
  - $F(x,x) = f(x)$, $f(x) \leq F(x,y)$
  - gradient descent is a majorization minimization
    $f(x_{t+1}) \leq F(x_{t+1},x_t) \leq F(x_t,x_t) = f(x_t)$
  - very important view to develop proximal algorithms

# fixed step size

- we discuss when function has Lipschitz continuous gradient and is strongly convex



# functions with Lipschitz gradient

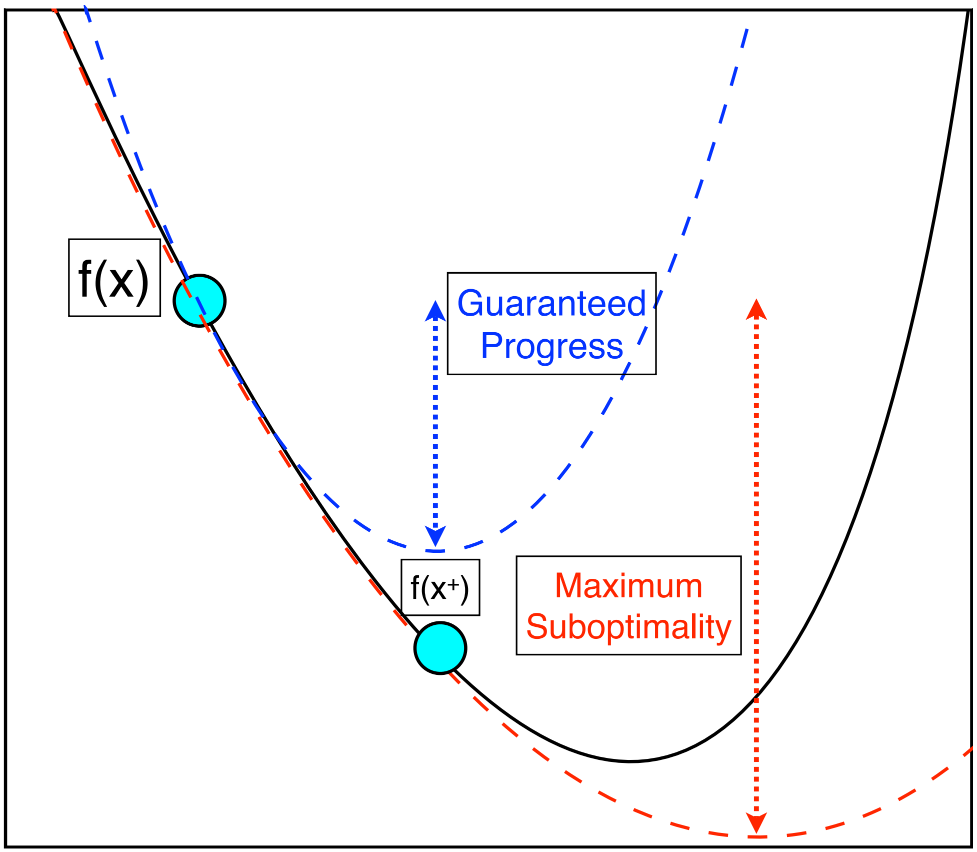
- a function has Lipschitz gradient if exist L > 0
  $$\|\nabla f(z)-\nabla f(x)\| \leq L\|z-x\|,$$
- descent lemma
  $f(y) \leq f(x) + \nabla f(x)^T(y-x) + L/2\ \|y-x\|^2$
  - global quadratic lower bound on function value
- minimize the right hand side w.r.t. y
  - $x^+ = x - (1/L)\nabla f(x)$
  - $f(x^+) \leq f(x) - (1/2L)\|\nabla f(x)\|^2$
  - i.e., $x^+$ will decrease the objective function

# strongly convex function

- a function is strongly convex if $\nabla^2 f(z) \succcurlyeq \mu I$, so
  $f(y) \geq f(x) + \nabla f(x)(y-x) + \mu/2||y-x||^2$
  - global quadratic upper bound on function value
- minimize both sides w.r.t. y
  - $f^* = f(x^*) \geq f(x) - (1/2\mu)||\nabla f(x)||^2$
  - this is an upper-bound how far we are from the optimal solution

# results

- combining both results, we have
  - $f(x^+) \leq f(x) - (1/2L)||\nabla f(x)||^2$
  - $f(x^*) \geq f(x) - (1/2\mu)||\nabla f(x)||^2$



# results

- combining both results, we have
  - $f(x^+) \leq f(x) - (1/2L)||\nabla f(x)||^2$
  - $f(x^*) \geq f(x) - (1/2\mu)||\nabla f(x)||^2$
  - we have

    $$2\mu(f(x) - f(x^*)) \leq ||\nabla f(x)||^2 \leq 2L(f(x) - f(x^+))$$

    then

    $$f(x^+) - f^* \leq c(f(x) - f^*), \text{ where } c = \mu/L < 1$$

- using this result in gradient descent, and set

  $$x_{t+1} = x_t - (1/L)\nabla f(x_t)$$

  we have

  $$f(x_t) - f^* \leq c^t(f(x_0) - f^*)$$

  i.e., error reduces by a constant factor c

# results

- GD descent for a function with Lipschitz gradient and strongly convex converges exponential with factor c = μ/ L
    - c is known as the condition number
    - it is bounded by the ratio of minimum and maximum of the Hessian matrix
- the convergence speed of GD is strongly affected by the condition number

# backtrack search

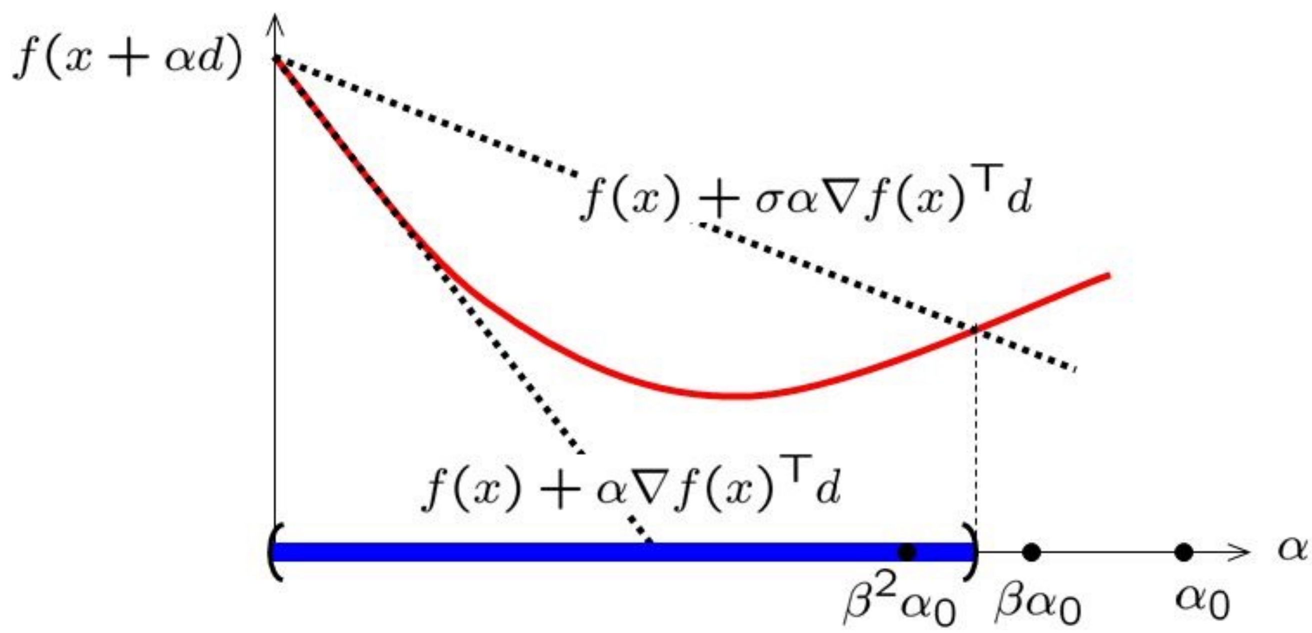
- using definition of convex function we have

$$f(x - \alpha\nabla f(x)) \geq f(x) - \alpha||\nabla f(x)||^2$$

  so the decrement of objective function is limited by

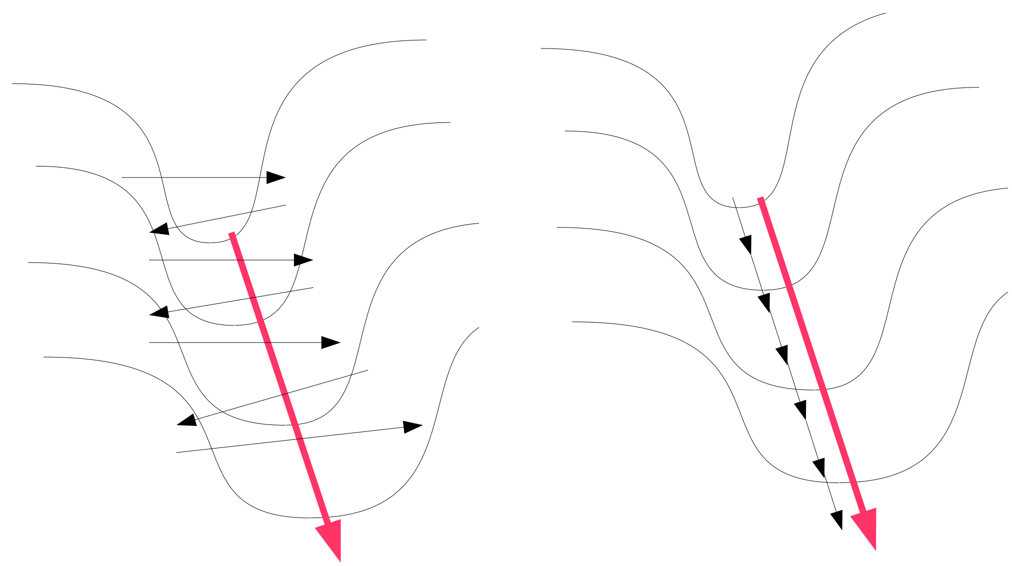
$$\varepsilon = \alpha||\nabla f(x)||^2 \geq f(x) - f(x - \alpha\nabla f(x))$$

- we pick constants $0 < \beta < 1$, $0 < \sigma < 1$, if current $\alpha$ does not lead to decrement of $\sigma\varepsilon$, then $\alpha = \alpha\beta$
- this is known as Amijo's rule for backtracking search



# problem with gradient descent

- Gradient descent is too local, taking steps optimal locally
  - if we know the overall landscape of the objective function, i.e., the curvature, then convergence can be a lot faster
- not affine invariant: depending on variable representation
- linear convergence time with regards to number of iteration

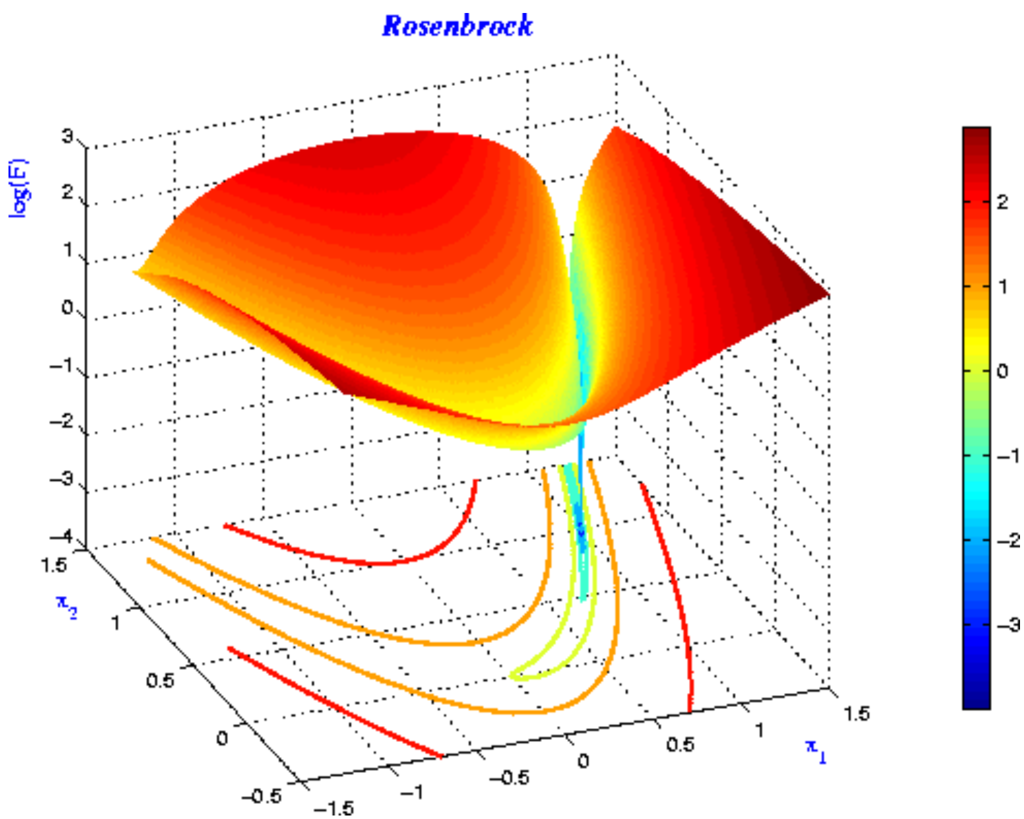


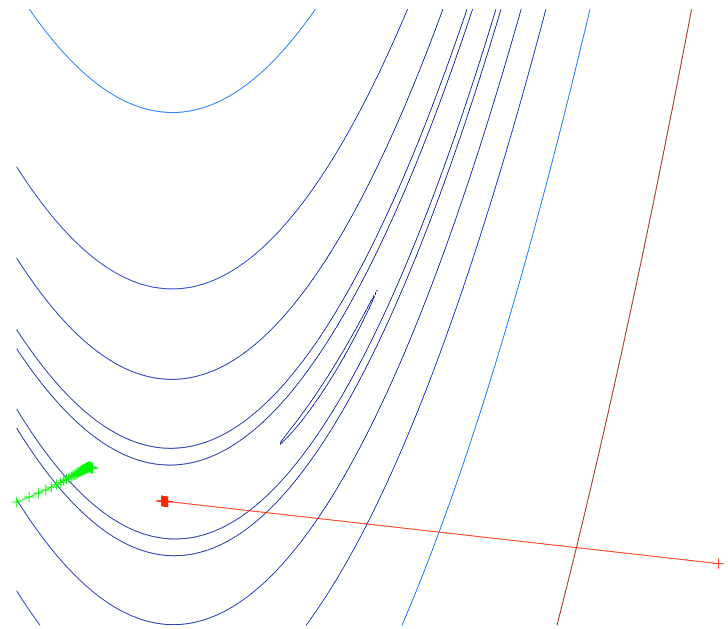
# problem with gradient descent

- Rosenbrock function (the banana function)

$$f(x,y) = (1-x)^2 + 100(y-x^2)^2.$$

GD



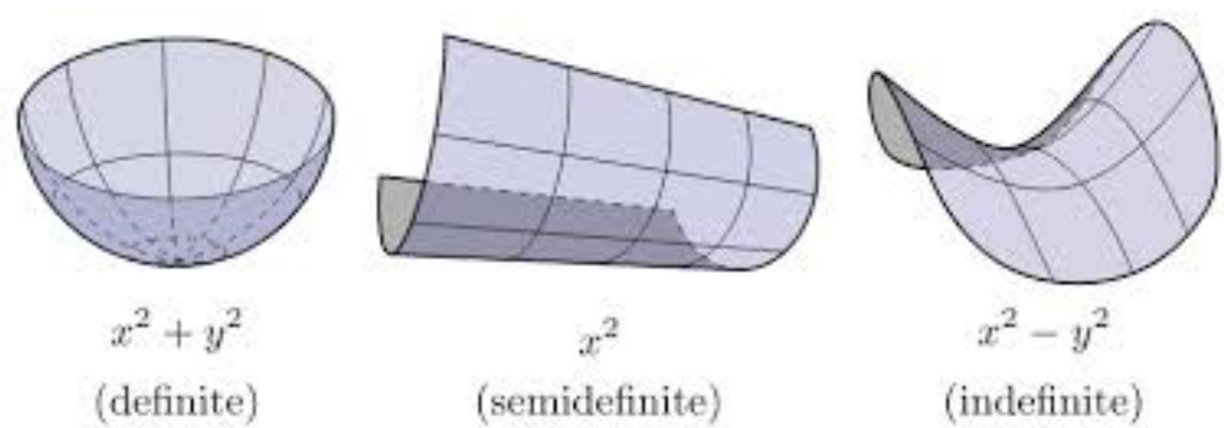



# Hessian matrix

- symbolically, Hessian is outer product of gradient operator

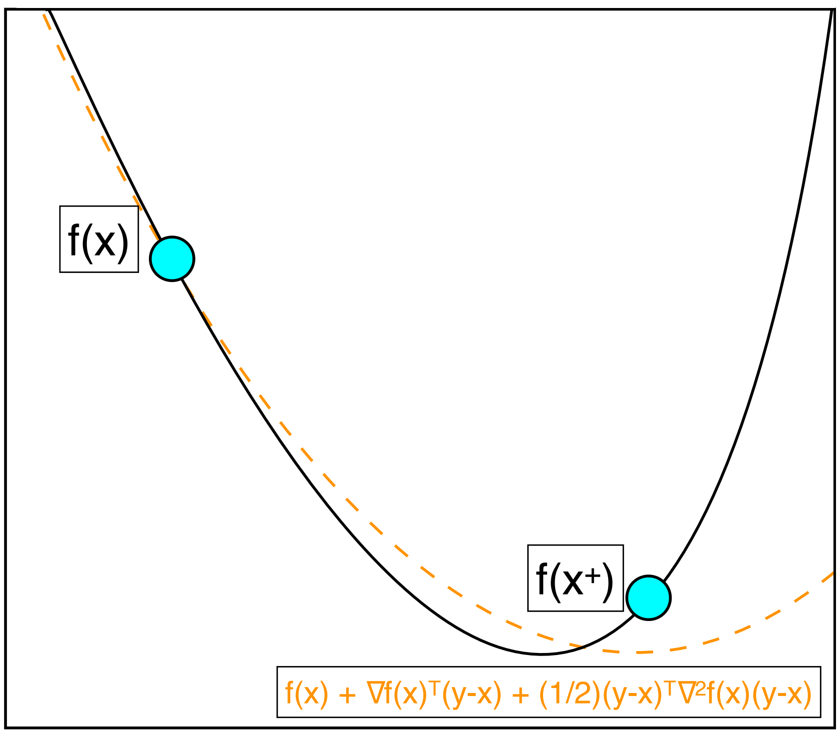
$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix} \quad \text{and} \quad \nabla^2 f = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}$$

- intuition of Hessian matrix



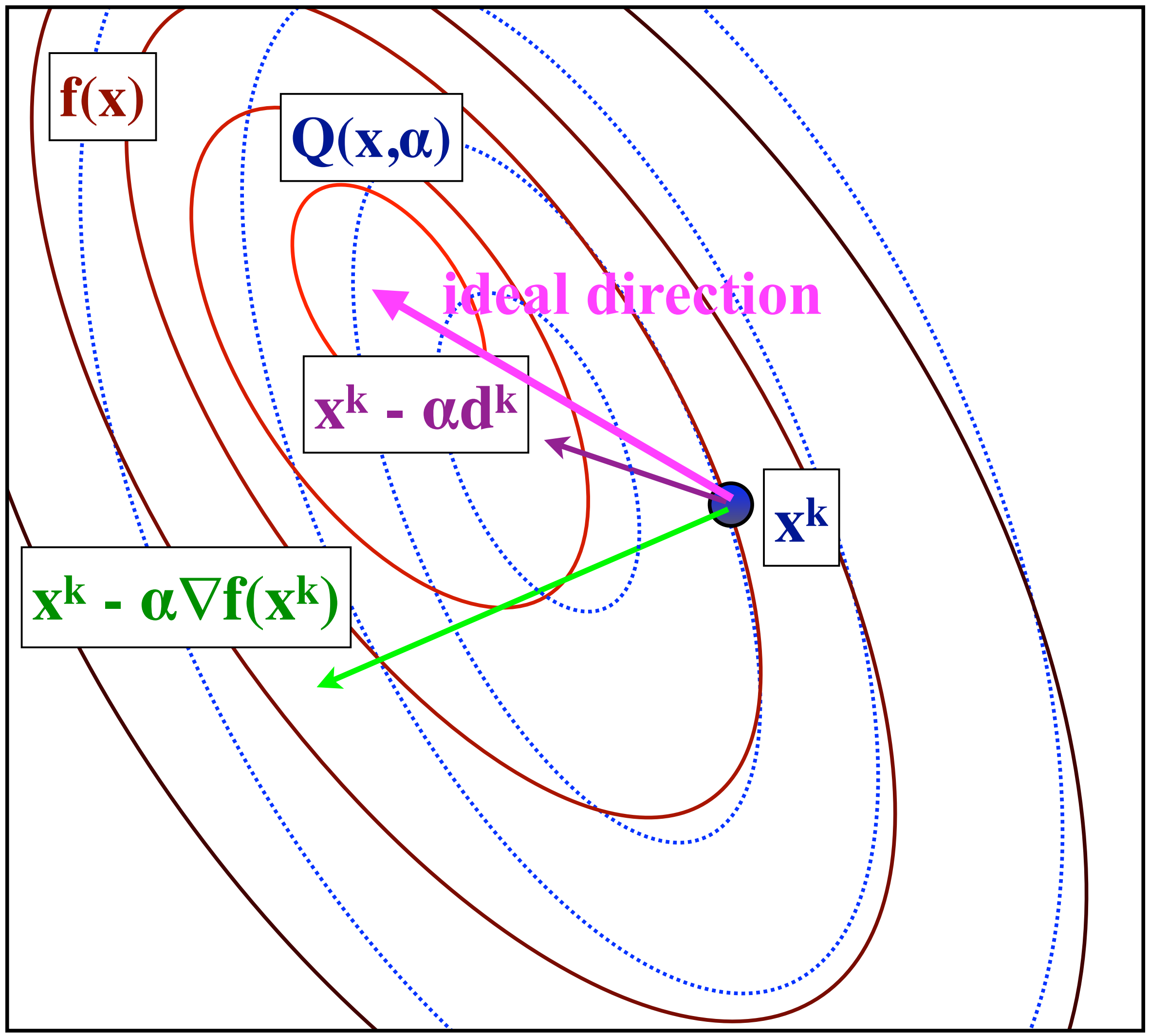
# Newton's method

- quadratic approximation of a function
  $f(\mathbf{x}+\mathbf{h}) \doteq f(\mathbf{x}) + \mathbf{h}^T\nabla f(\mathbf{x}) + ½\ \mathbf{h}^T\nabla^2 f(\mathbf{x})\mathbf{h}$
- best approximation is $\mathbf{h}=- \nabla^2 f(\mathbf{x})^{-1}\nabla f(\mathbf{x})$
- converge very fast, roughly $O(r^{2^t})$ for some r
- initialize: $t= 0$, $x_0$
- while not converge
  - $d_t = -\nabla^2 f(x_t)^{-1}\nabla f(x_t)$
  - decide step size $\alpha_t$
    - line search or Amijo
  - update $x_{t+1} = x_t + \alpha_t d_t$



# GD vs. Newton

- GD is too local, taking steps optimal locally
- Newton's method considers curvature, less locally



# GD vs. Newton

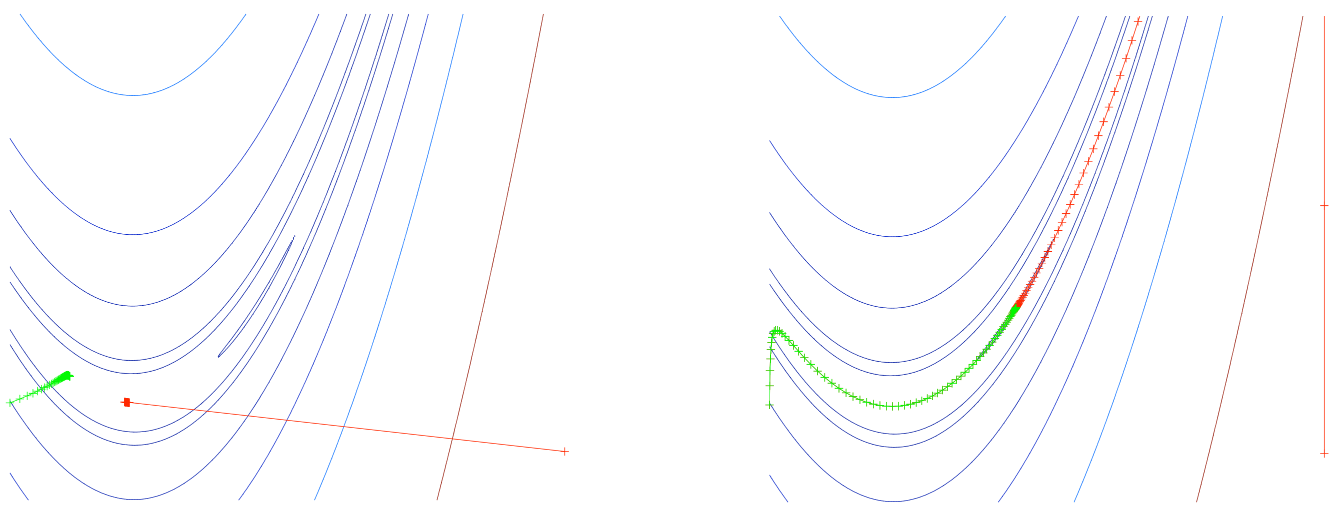
- convergence speed (in number of iterations)
  - linear vs. quadratic
- complexity of each iteration (in data dimension)
  - linear vs. quadratic (or cubic if matrix inversion)
    - forming Hessian matrix $O(d^2)$
    - inverting Hessian matrix $O(d^3)$
- for small scale problems, Newton always preferred
- for medium scale problems, some smart tricks can help to improve Newton's method (like quasi-newton)
- for large scale problems, GD (particularly stochastic GD) is the only choice

# Comparing GD & Newton's method

- Rosenbrock function (the banana function)

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2.$$

  - GD can go fast down to the valley but stuck
  - Newton's method makes continuous progress



# problems with Newton's method

- inversion of Hessian may be hard to compute
  - no explicit matrix inversion
  - solve quadratic optimization problem
    - conjugate gradient descent
- Hessian may be too large to form
  - quasi-Newton BFGS, L-BFGS,
  - Gauss-Newton
- H may not be p.d., i.e., objective not convex
  - Levenberg-Marquadt