Gradient Descent

Dr. Xiaowei Huang

https://cgi.csc.liv.ac.uk/~xiaowei/

Up to now,

- Three machine learning algorithms:
  - decision tree learning
  - k-nn
  - linear regression

only optimization objectives are discussed, but how to solve?

Today’s Topics

- Derivative
- Gradient
- Directional Derivative
- Method of Gradient Descent
- Example: Gradient Descent on Linear Regression
- Linear Regression: Analytical Solution

Problem Statement: Gradient-Based Optimization

- Most ML algorithms involve optimization
- Minimize/maximize a function *f* (***x***) by altering ***x***
  - Usually stated as a minimization of e.g., the loss etc
  - Maximization accomplished by minimizing *–f(**x**)*
- *f* (***x***) referred to as objective function or criterion
  - In minimization also referred to as loss function cost, or error
  - Example:
    - linear least squares $f(x) = \frac{1}{2}||Ax - b||^2$
    - Linear regression $\hat{L}(f_w) = \frac{1}{m}\sum_{i=1}^{m}(w^T x^{(i)} - y^{(i)})^2$
  - Denote optimum value by ***x****=argmin *f* (***x***)

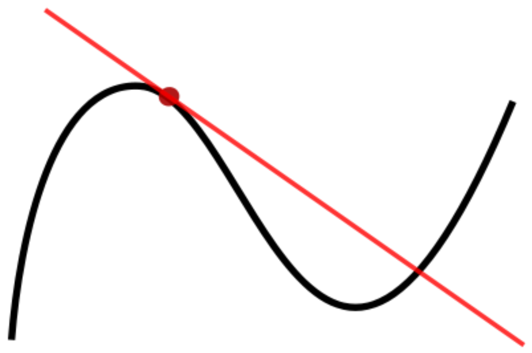
Derivative

Derivative of a function

- Suppose we have function $y=f(x)$, x, y real numbers
  - Derivative of function denoted: $f'(x)$ or as dy/dx
    - Derivative $f'(x)$ gives the slope of $f(x)$ at point x
    - It specifies how to scale a small change in input to obtain a corresponding change in the output:

$$f(x+\varepsilon) \approx f(x) + \varepsilon f'(x)$$

    - It tells how you make a small change in input to make a small improvement in y



Recall what's the derivative for the following functions:
$f(x) = x^2$
$f(x) = e^x$
…

Calculus in Optimization

- Suppose we have function $y = f(x)$, where *x, y are* real numbers
- Sign function:

$$sign(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ 1 & \text{if } x > 0 \end{cases}$$

- We know that

$$f(x - \epsilon sign(f'(x))) < f(x)$$

  for small ε.

- Therefore, we can reduce $f(x)$ by moving *x* in small steps with opposite sign of derivative

This technique is called *gradient descent* (Cauchy 1847)

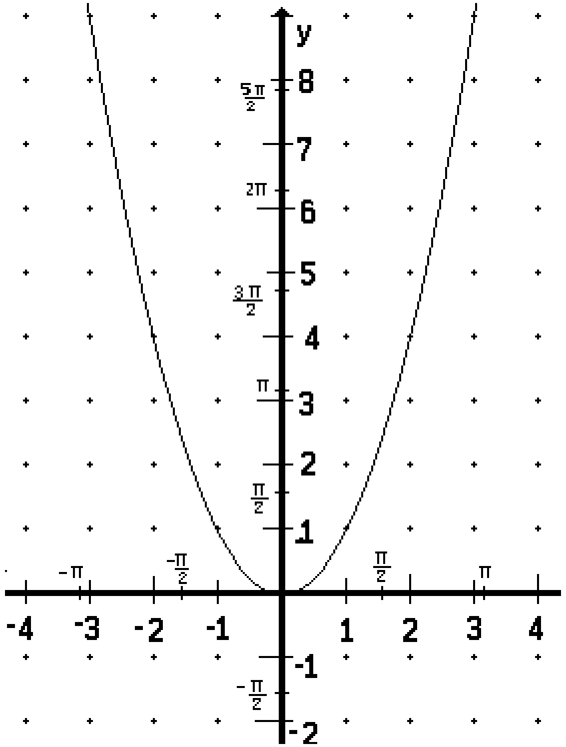
Why opposite?

Example

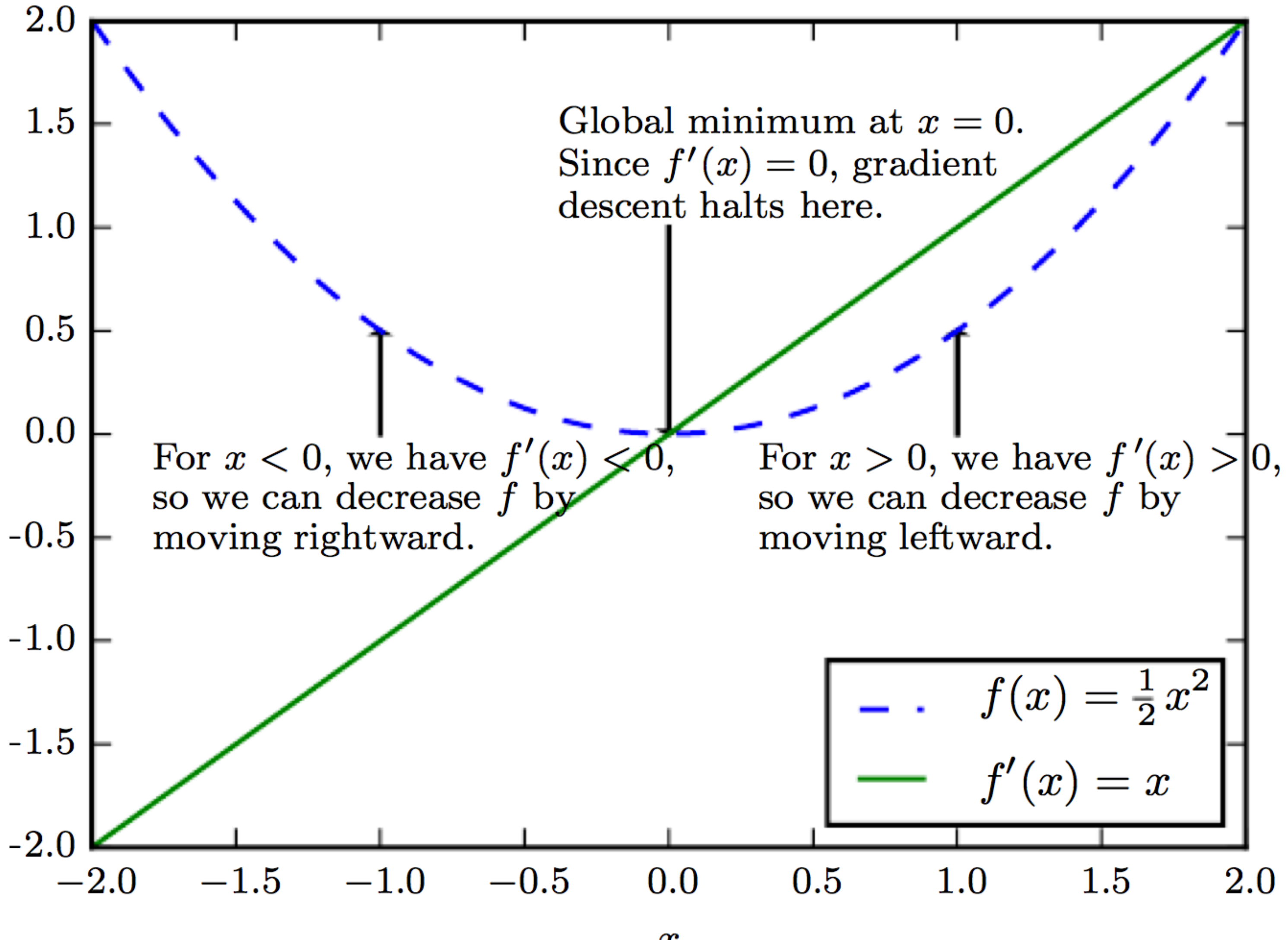
- Function $f(x) = x^2$ $\varepsilon = 0.1$
- $f'(x) = 2x$

- For $x = -2$, $f'(-2) = -4$, $\text{sign}(f'(-2))=-1$
- $f(-2- \varepsilon*(-1)) = f(-1.9) < f(-2)$

- For $x = 2$, $f'(2) = 4$, $\text{sign}(f'(2)) = 1$
- $f(2- \varepsilon*1) = f(1.9) < f(2)$



Gradient Descent Illustrated



For x<0, f(x) decreases with x and *f'(x)<0*

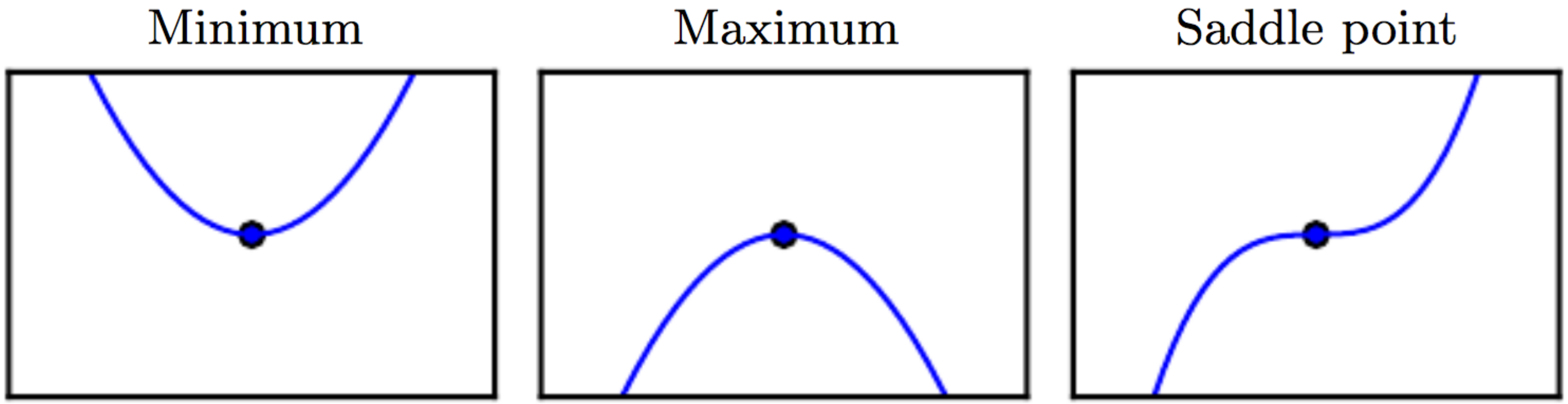
For x>0, *f(x)* increases with x and *f'*(x)*>0*

Use *f'(x)* to follow function downhill

Reduce *f(x)* by going in direction opposite sign of derivative *f'*(x)

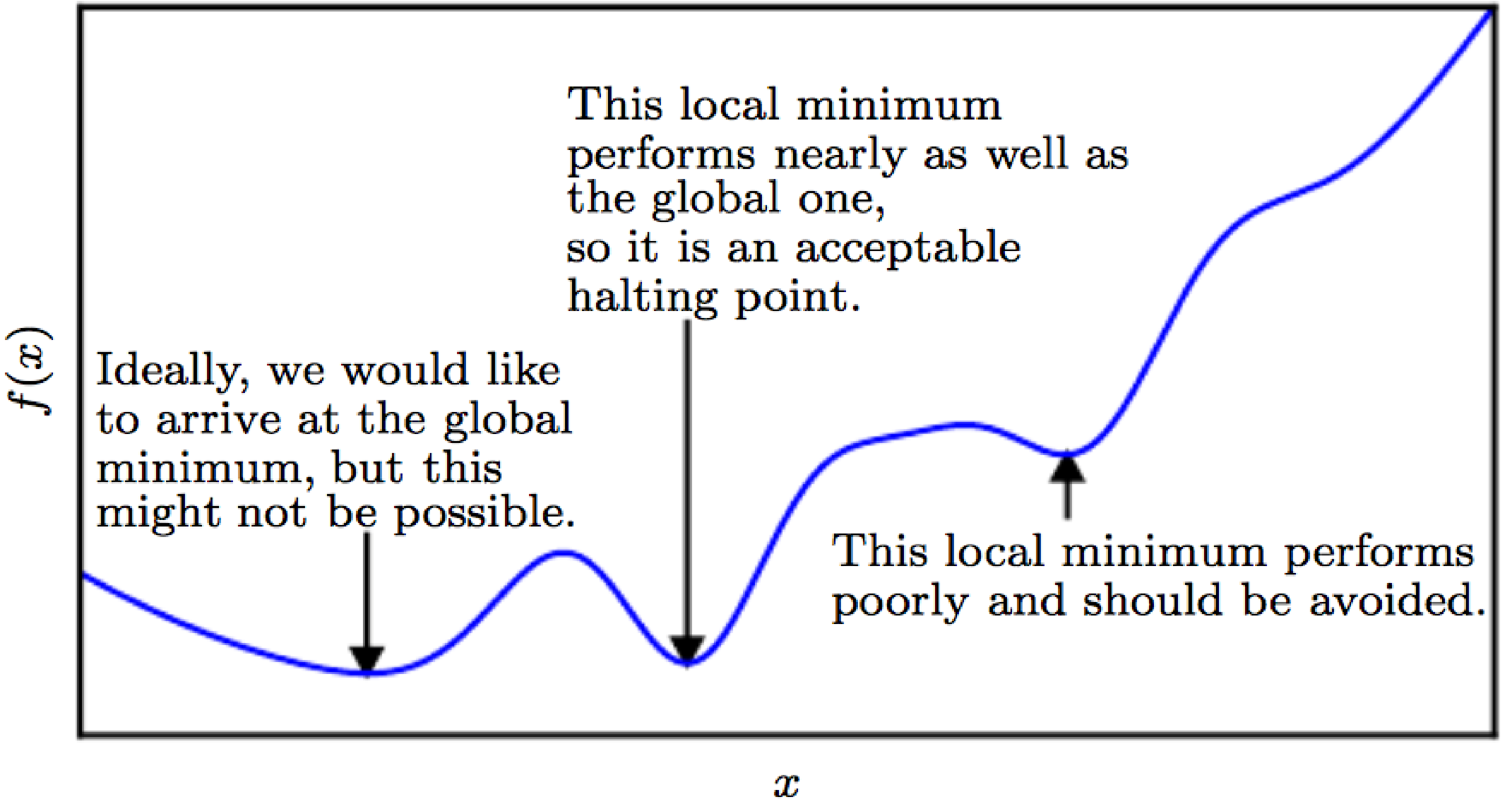
Stationary points, Local Optima

- When $f'(x) = 0$ derivative provides no information about direction of move
- Points where $f'(x) = 0$ are known as *stationary* or critical points
  - Local minimum/maximum: a point where *f(x)* lower/ higher than all its neighbors
  - Saddle Points: neither maxima nor minima



Presence of multiple minima

- Optimization algorithms may fail to find global minimum
- Generally accept such solutions



Gradient

Minimizing with multiple dimensional inputs

- We often minimize functions with multiple-dimensional inputs

$$f : \mathrm{R}^n \rightarrow \mathrm{R}$$

- For minimization to make sense there must still be only one (scalar) output

Functions with multiple inputs

- Partial derivatives

$$\frac{\partial}{\partial x_i} f(x)$$

measures how *f* changes as only variable x_i increases at point ***x***

- Gradient generalizes notion of derivative where derivative is wrt a vector
- Gradient is vector containing all of the partial derivatives denoted

$$\nabla_x f(x) = (\frac{\partial}{\partial x_1} f(x), \ldots, \frac{\partial}{\partial x_n} f(x))$$

Example

- $y = 5x_1^5 + 4x_2 + x_3^2 + 2$
- so what is the exact gradient on instance (1,2,3)

- the gradient is $(25x_1^4, 4, 2x_3)$
- On the instance (1,2,3), it is (25,4,6)

Functions with multiple inputs

- Gradient is vector containing all of the partial derivatives denoted

$$\nabla_x f(x) = (\frac{\partial}{\partial x_1} f(x), ..., \frac{\partial}{\partial x_n} f(x))$$

  - Element *i* of the gradient is the partial derivative of *f* wrt x_i
- Critical points are where every element of the gradient is equal to zero

$$\nabla_x f(x) = 0 \equiv \begin{cases} \frac{\partial}{\partial x_1} f(x) = 0 \\ \ldots \\ \frac{\partial}{\partial x_n} f(x) = 0 \end{cases}$$

Example

- $y = 5x_1^5 + 4x_2 + x_3^2 + 2$
- so what are the critical points?

- the gradient is $(25x_1^4, 4, 2x_3)$
- We let $25x_1^4 = 0$ and $2x_3 = 0$, so all instances whose x_1 and x_3 are 0. but 4 /= 0. So there is no critical point.

Directional Derivative

Directional Derivative

- Directional derivative in direction u (a unit vector) is the slope of function f in direction u
- This evaluates to

$$u^T \nabla_x f(x)$$

- Example: let $u^T = (u_x, u_y, u_z)$ be a unit vector in Cartesian coordinates, so

$$||u||_2 = \sqrt{u_x^2 + u_y^2 + u_z^2} = 1$$

then

$$u^T \nabla_x f(x) = \frac{\partial f}{\partial x} u_x + \frac{\partial f}{\partial y} u_y + \frac{\partial f}{\partial z} u_z$$

Directional Derivative

- To minimize *f* find direction in which *f* decreases the fastest

$$\min_{u, u^T u = 1} u^T \nabla_x f(x) = \min_{u, u^T u = 1} ||u||_2 \cdot ||\nabla_x f(x)||_2 \cdot \cos\theta$$

  - where θ is angle between u and the gradient
  - Substitute $||u||_2 = 1$ and ignore factors that not depend on u this simplifies to

$$\min_u \cos\theta$$

  - This is minimized when u points in direction opposite to gradient
- In other words, the *gradient points directly uphill, and the negative gradient points directly downhill*

Method of Gradient Descent

Method of Gradient Descent

- The gradient points directly uphill, and the negative gradient points directly downhill
- Thus we can decrease f by moving in the direction of the negative gradient
  - This is known as the method of steepest descent or gradient descent
- Steepest descent proposes a new point

$$x' = x - \epsilon \nabla_x f(x)$$

  - where ϵ is the learning rate, a positive scalar. Set to a small constant.

Choosing ϵ: Line Search

- We can choose ϵ in several different ways
- Popular approach: set ϵ to a small constant
- Another approach is called *line search*:
  - Evaluate

$$f(x - \epsilon \nabla_x f(x))$$

for several values of ϵ and choose the one that results in smallest objective function value

Example: Gradient Descent on Linear Regression

Example: Gradient Descent on Linear Regression

- Linear regression: $\hat{L}(f_w) = \frac{1}{m}\sum_{i=1}^{m}(w^T x^{(i)} - y^{(i)})^2 = \frac{1}{m}||Xw - y||_2^2$
- The gradient is

$$
\begin{aligned}
& \nabla_w \hat{L}(f_w) \\
= \; & \nabla_w \tfrac{1}{m}||Xw - y||_2^2 \\
= \; & \nabla_w [(Xw - y)^T (Xw - y)] \\
= \; & \nabla_w [w^T X^T X w - 2 w^T X^T y + y^T y] \\
= \; & 2X^T X w - 2X^T y
\end{aligned}
$$

Example: Gradient Descent on Linear Regression

- Linear regression: $\hat{L}(f_w) = \frac{1}{m}\sum_{i=1}^{m}(w^T x^{(i)} - y^{(i)})^2 = \frac{1}{m}||Xw - y||_2^2$
- The gradient is $\nabla_w \hat{L}(f_w) = 2X^T X w - 2X^T y$
- Gradient Descent algorithm is
  - Set step size ϵ, tolerance δ to small, positive numbers.
  - *While* $||X^T X w - X^T y||_2 > \delta$ do

$$x \longleftarrow x - \epsilon(X^T X w - X^T y)$$

Linear Regression: Analytical solution

Convergence of Steepest Descent

- Steepest descent converges when every element of the gradient is zero
  - In practice, very close to zero
- We may be able to avoid iterative algorithm and jump to the critical point by solving the following equation for x

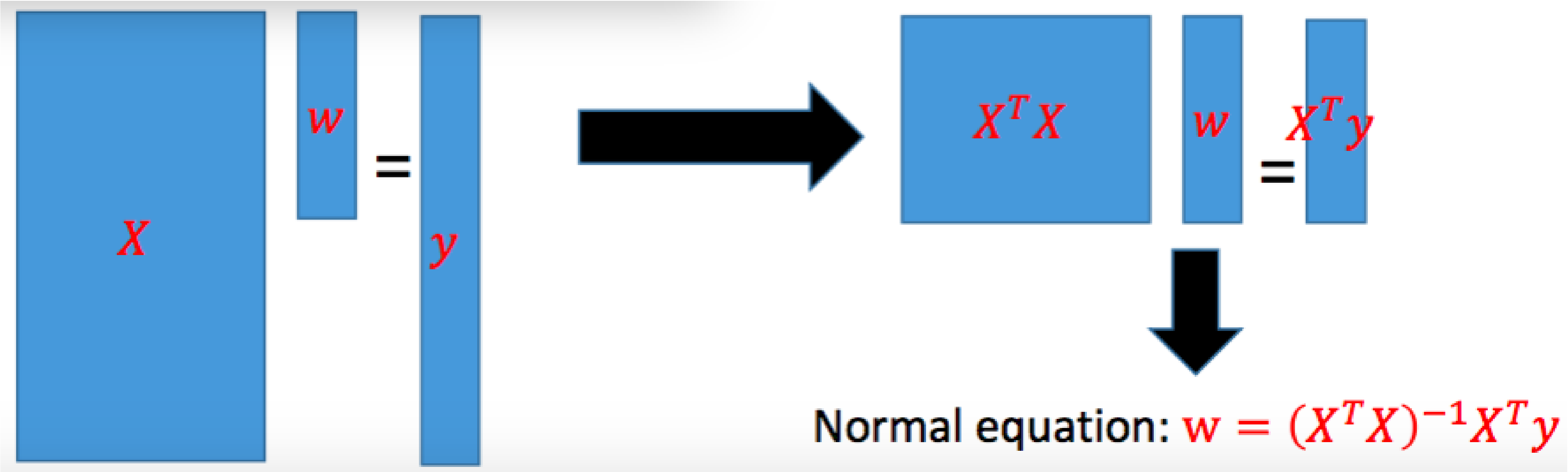
$$\nabla_x f(x) = 0$$

Linear Regression: Analytical solution

- Linear regression: $\hat{L}(f_w) = \frac{1}{m}\sum_{i=1}^{m}(w^T x^{(i)} - y^{(i)})^2 = \frac{1}{m}||Xw - y||_2^2$
- The gradient is $\nabla_w \hat{L}(f_w) = 2X^T X w - 2X^T y$
- Let $\nabla_w \hat{L}(f_w) = 2X^T X w - 2X^T y = 0$
- Then, we have $w = (X^T X)^{-1} X^T y$

Linear Regression: Analytical solution

- Algebraic view of the minimizer
- If X is invertible, just solve $Xw = y$ and get $w = X^{-1}y$
- But typically X is a tall matrix



Generalization to discrete spaces

Generalization to discrete spaces

- Gradient descent is limited to continuous spaces
- Concept of repeatedly making the best small move can be generalized to discrete spaces
- Ascending an objective function of discrete parameters is called *hill climbing*

Exercises

- Given a function $f(x)= e^x/(1+e^x)$, how many critical points?
- Given a function $f(x_1,x_2)= 9x_1^2+3x_2+4$, how many critical points?
- Please write a program to do the following: given any differentiable function (such as the above two), an ε, and a starting x and a target x', determine whether it is possible to reach x' from x. If possible, how many steps? You can adjust ε to see the change of the answer.

Extended Materials

Beyond Gradient: Jacobian and Hessian matrices

- Sometimes we need to find all derivatives of a function whose input and output are both vectors
- If we have function *f:* R_m -> R_n
  - Then the matrix of partial derivatives is known as the Jacobian matrix *J* defined as

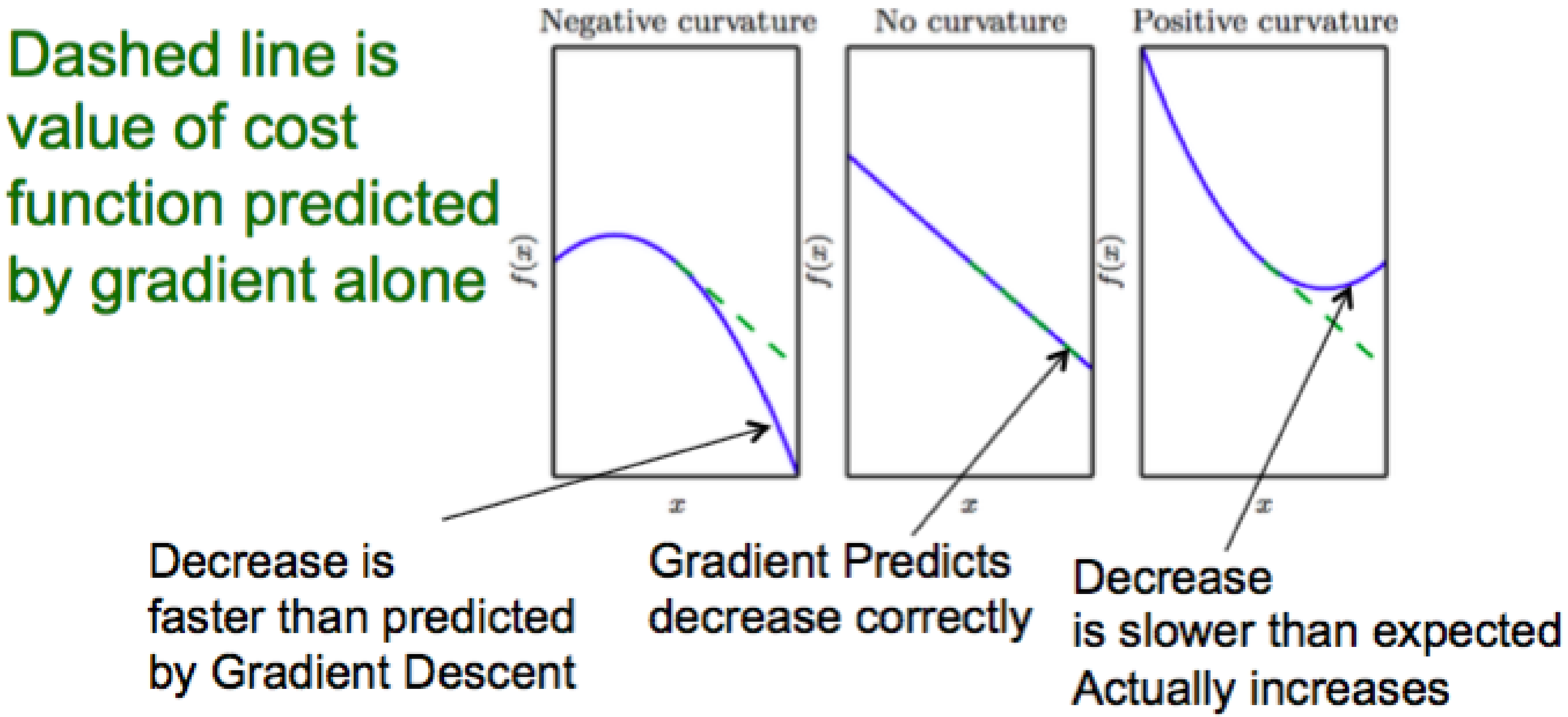
$$\boxed{J_{i,j} = \frac{\partial}{\partial x_j} f(x)_i}$$

Second derivative

- Derivative of a derivative
- For a function f*:* R^n -> R the derivative wrt x_i of the derivative of f wrt x_j is denoted as $\frac{\partial^2}{\partial x_i \partial x_j} f$
- In a single dimension we can denote $\frac{\partial^2}{\partial x^2} f$ by $f''(\boldsymbol{x})$
- Tells us how the first derivative will change as we vary the input
- This is important as it tells us whether a gradient step will cause as much of an improvement as based on gradient alone

Second derivative measures curvature

- Derivative of a derivative
- Quadratic functions with different curvatures



Hessian

- Second derivative with many dimensions
- $H\,(\,f\,)\,(x)$ is defined as $$H(f)(x)_{i,j} = \frac{\partial^2}{\partial x_i \partial x_j} f(x)$$
- Hessian is the Jacobian of the gradient
- Hessian matrix is symmetric, i.e., $H_{i,j} = H_{j,i}$
  - anywhere that the second partial derivatives are continuous
  - So the Hessian matrix can be decomposed into a set of real eigenvalues and an orthogonal basis of eigenvectors
    - Eigenvalues of *H* are useful to determine learning rate as seen in next two slides

Role of eigenvalues of Hessian

- Second derivative in direction d is $d^T H d$
  - If d is an eigenvector, second derivative in that direction is given by its eigenvalue
  - For other directions, weighted average of eigenvalues (weights of 0 to 1, with eigenvectors with smallest angle with d receiving more value)
- Maximum eigenvalue determines maximum second derivative and minimum eigenvalue determines minimum second derivative

Learning rate from Hessian

- Taylor's series of *f*(***x***) around current point $\boldsymbol{x}^{(0)}$

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}^{(0)}) + (\boldsymbol{x} - \boldsymbol{x}^{(0)})^T \boldsymbol{g} + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}^{(0)})^T H(\boldsymbol{x} - \boldsymbol{x}^{(0)})$$

  - where ***g*** is the gradient and *H* is the Hessian at $x^{(0)}$
- If we use learning rate ε the new point ***x*** is given by $\boldsymbol{x}^{(0)}-\varepsilon\boldsymbol{g}$. Thus we get

$$f(\boldsymbol{x}^{(0)} - \varepsilon \boldsymbol{g}) \approx f(\boldsymbol{x}^{(0)}) - \varepsilon \boldsymbol{g}^T \boldsymbol{g} + \frac{1}{2}\varepsilon^2 \boldsymbol{g}^T H \boldsymbol{g}$$

  - There are three terms:
    - original value of *f*,
    - expected improvement due to slope, and
    - correction to be applied due to curvature
  - Solving for step size when correction is least gives $\varepsilon^* \approx \dfrac{\boldsymbol{g}^T \boldsymbol{g}}{\boldsymbol{g}^T H \boldsymbol{g}}$

Second Derivative Test: Critical Points

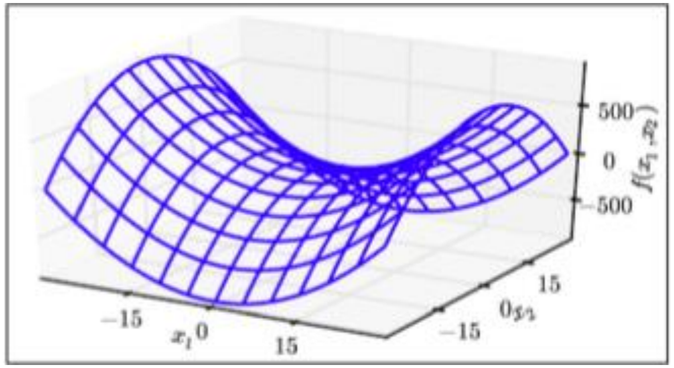
- On a critical point $f'(x)=0$
- When $f''(x)>0$ the first derivative $f'(x)$ increases as we move to the right and decreases as we move left
- We conclude that x is a local minimum
- For local maximum, $f'(x)=0$ and $f''(x)<0$
- When $f''(x)=0$ test is inconclusive: x may be a saddle point or part of a flat region

Multidimensional Second derivative test

- In multiple dimensions, we need to examine second derivatives of all dimensions
- Eigendecomposition generalizes the test
- Test eigenvalues of Hessian to determine whether critical point is a local maximum, local minimum or saddle point
- When H is positive definite (all eigenvalues are positive) the point is a local minimum
- Similarly negative definite implies a maximum

Saddle point

- Contains both positive and negative curvature
- Function is $f(\boldsymbol{x})=x_1^2-x_2^2$



  - Along axis x_1, function curves upwards: this axis is an eigenvector of H and has a positive value
  - Along x_2, function corves downwards; its direction is an eigenvector of H with negative eigenvalue
  - At a saddle point eigen values are both positive and negative

Inconclusive Second Derivative Test

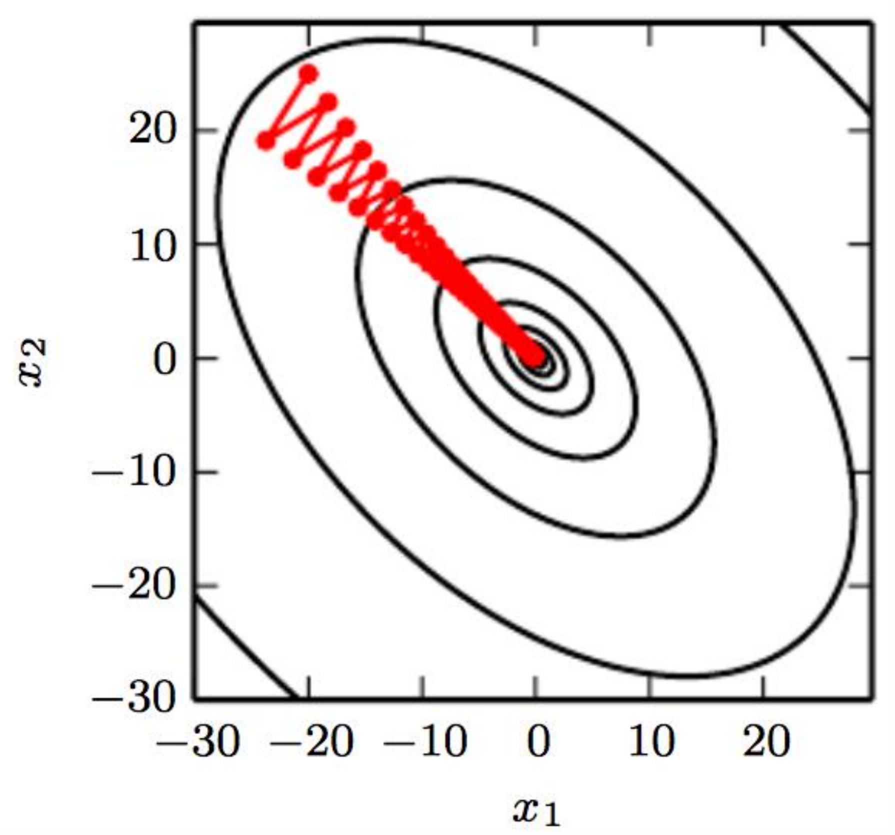
- Multidimensional second derivative test can be inconclusive just like univariate case
- Test is inconclusive when all non-zero eigen values have same sign but at least one value is zero
    - since univariate second derivative test is inconclusive in cross-section corresponding to zero eigenvalue

Poor Condition Number

- There are different second derivatives in each direction at a single point
- Condition number of H e.g., $\lambda_{max}/\lambda_{min}$ measures how much they differ
  - Gradient descent performs poorly when H has a poor condition no.
  - Because in one direction derivative increases rapidly while in another direction it increases slowly
  - Step size must be small so as to avoid overshooting the minimum, but it will be too small to make progress in other directions with less curvature

Gradient Descent without *H*

- *H* with condition no, 5
  - Direction of most curvature has five times more curvature than direction of least curvature
- Due to small step size Gradient descent wastes time
- Algorithm based on Hessian can predict that steepest descent is not promising



Newton's method uses Hessian

- Another second derivative method
  - Using Taylor's series of $f(\boldsymbol{x})$ around current $\boldsymbol{x}^{(0)}$

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}^{(0)}) + (\boldsymbol{x} - \boldsymbol{x}^{(0)})^T \nabla_x f(\boldsymbol{x}^{(0)}) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}^{(0)})^T H(f)(\boldsymbol{x} - \boldsymbol{x}^{(0)})(\boldsymbol{x} - \boldsymbol{x}^{(0)})$$

    - solve for the critical point of this function to give $\boldsymbol{x}^* = \boldsymbol{x}^{(0)} - H(f)(\boldsymbol{x}^{(0)})^{-1} \nabla_x f(\boldsymbol{x}^{(0)})$
  - When f is a quadratic (positive definite) function use solution to jump to the minimum function directly
  - When not quadratic apply solution iteratively
- Can reach critical point much faster than gradient descent
  - But useful only when nearby point is a minimum

Summary of Gradient Methods

- First order optimization algorithms: those that use only the gradient
- Second order optimization algorithms: use the Hessian matrix such as Newton's method
- Family of functions used in ML is complicated, so optimization is more complex than in other fields
  - No guarantees
- Some guarantees by using Lipschitz continuous functions,

$$\boxed{|f(\boldsymbol{x}) - f(\boldsymbol{y})| \le L \|\boldsymbol{x} - \boldsymbol{y}\|_2}$$

  - with Lipschitz constant *L*

Convex Optimization

- Applicable only to convex functions – functions which are well-behaved,
  - e.g., lack saddle points and all local minima are global minima
- For such functions, Hessian is positive semi-definite everywhere
- Many ML optimization problems, particularly deep learning, cannot be expressed as convex optimization

Constrained Optimization

- We may wish to optimize $f(\boldsymbol{x})$ when the solution $\boldsymbol{x}$ is constrained to lie in set S
  - Such values of $\boldsymbol{x}$ are feasible solutions
- Often we want a solution that is small, such as $||\boldsymbol{x}|| \leq 1$
- Simple approach: modify gradient descent taking constraint into account (using Lagrangian formulation)

Ex: Least squares with Lagrangian

- We wish to minimize $f(\mathrm{x}) = \frac{1}{2} \| A\mathrm{x} - \mathrm{b} \|^2$
  - Subject to constraint $\boldsymbol{x}^T\boldsymbol{x} \leq 1$
- We introduce the Lagrangian $L(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) + \lambda\left(\boldsymbol{x}^T\boldsymbol{x} - 1\right)$
  - And solve the problem $\min_{\boldsymbol{x}} \max_{\lambda, \lambda \geq 0} L(\boldsymbol{x}, \lambda)$
- For the unconstrained problem (no Lagrangian) the smallest norm solution is x=A+b
  - If this solution is not feasible, differentiate Lagrangian wrt x to obtain $A^T Ax - A^T b + 2\lambda x = 0$
  - Solution takes the form $x = (A^T A + 2\lambda I)^{-1} A^T b$
  - Choosing λ: continue solving linear equation and increasing λ until $\boldsymbol{x}$ has the correct norm

Generalized Lagrangian: KKT

- More sophisticated than Lagrangian
- Karush-Kuhn-Tucker is a very general solution to constrained optimization
- While Lagrangian allows equality constraints, KKT allows both equality and inequality constraints
- To define a generalized Lagrangian we need to describe S in terms of equalities and inequalities

Generalized Lagrangian

- Set *S* is described in terms of *m* functions *g(i)* and *n* functions *h(j)* so that

$$S = \left\{ \boldsymbol{x} \mid \forall i, g^{(i)}(\boldsymbol{x}) = 0 \text{ and } \forall j, h^{(j)}(\boldsymbol{x}) \leq 0 \right\}$$

  - Functions of *g* are equality constraints and functions of *h* are inequality constraints
- Introduce new variables λ_i and α_j for each constraint (called KKT multipliers) giving the generalized Lagrangian

$$\boxed{L(\boldsymbol{x}, \lambda, \alpha) = f(\boldsymbol{x}) + \sum_i \lambda_i g^{(i)}(\boldsymbol{x}) + \sum_j \alpha_j h^{(j)}(\boldsymbol{x})}$$

- We can now solve the unconstrained optimization problem