Optimization

CPS 271
Ron Parr

Outline

• Why?
• Unconstrained optimization
• Convex optimization
• Duality
• Lagrange multipliers
Why talk about optimization?

- Many machine learning problems involve some kind of optimization
  - Maximized posterior: $P(H|D)$
  - Maximum likelihood: $P(D|H)$
  - Minimize error on training set
  - Maximize margin of solution

- Optimization techniques are really important and useful in many areas

Unconstrained Optimization

- Often we need to find the minimum (or maximum) of some function

- Solving analytically for all possible local optima (zero gradient) is usually impractical

- What do we do?
Differentiable Objectives

- With differentiable objectives, we don’t need to “probe”

- Suppose objective function is: \( f(x_1, y_1, x_2, y_2, x_3, y_3) \)

- Gradient tells us direction and steepness of change

\[
\nabla f = \left( \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial y_1}, \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial y_2}, \frac{\partial f}{\partial x_3}, \frac{\partial f}{\partial y_3} \right)
\]

Following the Gradient

\[
\mathbf{x} = (x_1, y_1, x_2, y_2, x_3, y_3)
\]

\[
\mathbf{x} \leftarrow \mathbf{x} - \alpha \nabla f(\mathbf{x})
\]

For sufficiently small step sizes, this will converge to a region around a local optimum.

If gradient is hard to compute: Compute empirical gradient
Second Order Methods

- Gradient descent works (up to local optima), but can be slow
- Second order or quasi-Newton methods use second derivatives (Hessian) to speed things up, often dramatically
- A lot of machinery exists to help you do this, e.g., fminunc in matlab (uses BFGS; no apparent relationship to BFG9000)

- Some caveats:
  - Still only hits local minima
  - The Hessian is $O(n^2)$

What’s a Hessian?

$$H_f = \begin{pmatrix}
\frac{\partial^2 f}{\partial^2 x_1} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\
\cdots & \cdots & \cdots \\
\frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial^2 x_n}
\end{pmatrix}$$
Constrained Optimization

- Quite often, our variables must be constrained:
  - Must be in some range, e.g., probabilities
  - Must conform to requirements of the problem, e.g., be a linear combination of inputs

- Unconstrained optimization isn’t enough 😞

Convexity

- What is a convex set?

- What is a convex function?

\[ f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha) f(y) \]
Convex Optimization

minimize : $f_0(x)$
subject to: $f_i(x) \leq 0, i = 1, \ldots, m$

: $a_i^T x = b_i, i = 1, \ldots, p$

- $f_0, \ldots, f_m$ must be convex
- Equality constraints are affine

Examples

- Experiment design
  (measurements that greatest reduction in uncertainty)
- Zero sum games
- Production planning
  (Mix of goods to make given mix of materials)
- Network flow
  (choose links to maximize flow through a network)
- A few others from Boyd
  - Processor scheduling
  - Grasp force optimization
  - Phased array antenna beamforming
Why Convexity?

• Unique global optimum value
  (not necessarily unique argmax)

• Reasonably efficient algorithms exist
  – Polynomial cost per iteration
  – Solution improves with each iteration
  – No more than 10s or 100s of iterations

• In most cases, duality ensures that there is a corresponding max problem for all min problems

A Generic Algorithm for Convex Optimization

• Given: Convex objective and convex constraint set
• Start with a feasible point
• Repeat until gradient of objective is ~0
  – Take a gradient step in the direction that improves the objective function (as if we were in the unconstrained case)
  – If we leave the feasible set, project back onto the feasible set (find closest feasible point to current one)
  – If objective gets worse after projection, then reduce step size and try again

• This is probably the slowest possible non-stupid algorithm, but it is generic and gives insight into why such problems are solvable
Linear Programs

minimize : \( c^T x \)
subject to: \( Ax = b \)
\( : x \geq 0 \)

• Easily tweaked to handle \( \geq \) inequalities
• Multiply by -1 to reverse inequalities
• Easily tweaked to handle different x domain

Linear Programs (max formulation)

maximize : \( c^T x \)
subject to: \( Ax \leq b \)
\( : x \geq 0 \)

• Multiply by -1 to reverse inequalities
• Easily tweaked to handle different x domain
Linear programs: example

- Make reproductions of 2 paintings

\[
\begin{align*}
\text{maximize} & \quad 3x + 2y \\
\text{subject to} & \quad 4x + 2y \leq 16 \\
& \quad x + 2y \leq 8 \\
& \quad x + y \leq 5 \\
& \quad x \geq 0 \\
& \quad y \geq 0
\end{align*}
\]

- Painting 1:
  - Sells for $30
  - Requires 4 units of blue, 1 green, 1 red
- Painting 2
  - Sells for $20
  - Requires 2 blue, 2 green, 1 red
- We have 16 units blue, 8 green, 5 red

Solving the linear program graphically

\[
\begin{align*}
\text{maximize} & \quad 3x + 2y \\
\text{subject to} & \quad 4x + 2y \leq 16 \\
& \quad x + 2y \leq 8 \\
& \quad x + y \leq 5 \\
& \quad x \geq 0 \\
& \quad y \geq 0
\end{align*}
\]

optimal solution: \( x=3, y=2 \)
Modified LP

\[ \text{maximize } 3x + 2y \]
\[ \text{subject to} \]
\[ 4x + 2y \leq 15 \]
\[ x + 2y \leq 8 \]
\[ x + y \leq 5 \]
\[ x \geq 0 \]
\[ y \geq 0 \]

Optimal solution: \( x = 2.5, y = 2.5 \)
Solution value = 7.5 + 5 = 12.5

Half paintings?

Integer (linear) program

\[ \text{maximize } 3x + 2y \]
\[ \text{subject to} \]
\[ 4x + 2y \leq 15 \]
\[ x + 2y \leq 8 \]
\[ x + y \leq 5 \]
\[ x \geq 0, \text{ integer} \]
\[ y \geq 0, \text{ integer} \]

Optimal LP solution: \( x=2, y=3 \) (objective 12)
Optimal LP solution: \( x=2.5, y=2.5 \) (objective 12.5)
Mixed integer (linear) program

\[
\begin{align*}
\text{maximize} & \quad 3x + 2y \\
\text{subject to} & \quad 4x + 2y \leq 15 \\
& \quad x + 2y \leq 8 \\
& \quad x + y \leq 5 \\
& \quad x \geq 0 \\
& \quad y \geq 0, \text{ integer}
\end{align*}
\]

optimal LP solution: \(x=2.5, y=2.5\) (objective 12)
optimal IP solution: \(x=2, y=3\) (objective 12)
optimal MIP solution: \(x=2.75, y=2\) (objective 12.25)

Sudoku

- Sudoku can be solved as an integer program with no objective function
- Let \(x_{ijk}\) (binary) = 1 correspond to \(k\) being in row \(i\), column \(j\), \(1 \leq k \leq 9\)
- Define 9 sets of variables \(b_1 \ldots b_9\), corresponding to the 9 blocks

\[
\begin{align*}
\forall i, j & : \sum_k x_{ijk} = 1 \\
\forall i, k & : \sum_j x_{ijk} = 1 \\
\forall j, k & : \sum_i x_{ijk} = 1 \\
\forall k, l & : \sum_{i,j \in b_l} x_{ijk} = 1 \\
\forall ij & : x_{ijk} \in \{0, 1\}
\end{align*}
\]
Sudoku

Each square has exactly one number in it:
\[ \forall i, j : \sum_{k} x_{ijk} = 1 \]

Each number appears exactly once in each row:
\[ \forall i, k : \sum_{j} x_{ijk} = 1 \]

Each number appears exactly once in each column:
\[ \forall j, k : \sum_{i} x_{ijk} = 1 \]

Each number appears exactly once in each block:
\[ \forall k, l : \sum_{i,j \in B_{i,j}} x_{ijk} = 1 \]

All variables are binary:
\[ \forall i, j, k : x_{ijk} \in \{0,1\} \]

Solving linear/integer programs

- Linear programs can be solved efficiently
  - Simplex, ellipsoid, interior point methods...
  - LPs are "weakly polynomial"

- Standard packages for solving these
  - GNU Linear Programming Kit, CPLEX, ...
  - Algorithms built in to matlab optimization toolbox (linprog)

- (Mixed) integer programs are NP-hard to solve
  - Quite easy to model many standard NP-complete problems as integer programs (try it!)
  - Search type algorithms such as branch and bound

- Some packages will try to solve (mixed)integer programs too, but they are heuristic – fast sometimes, exponential in worst case
Example LP Prestidigitation

- Suppose you want to minimize max norm:
  \[ \min_x \|C^T x - b\|_\infty. \]
- This doesn’t look linear, but:
  
  minimize: \( \varepsilon \)

  subject to: \( C^T x - b = e \)
  
  \[ \varepsilon \geq e_i \forall i \]
  
  \[ \varepsilon \leq -e_i \forall i \]

Quadratic Program

\[
\text{minimize:} \quad \frac{1}{2} x^T P_0 x + q_0^T x \\
\text{subject to:} \quad x \geq 0 \\
\quad A x = b
\]

- Quadratic objective
- Linear contraints
- Convex if \( P_0 \) is PSD
- Can be solved fairly efficiently
QCQP

minimize: $\frac{1}{2} x^T P_0 x + q_0^T x$

subject to: $\frac{1}{2} x^T P_i x + q_i^T x + r_i \leq 0, i = 1,...,m$

: $Ax = b$

- Quadratically constrained quadratic program
- Not convex in general (NP-hard in general)
- Convex if $P_0...P_n$ are positive semi-definite
- Special convex cases:
  - QP: $P_i=0$ for all $i > 0$
  - Linear program: $P_i=0$ for all $i$
Visualizing Why Convexity Matters

Duality

- Under some mild assumptions, convex minimization problems have a corresponding dual maximization problem with the same objective value.
- Solution to primal can be used to reconstruct solution to dual, and vice versa.
- LP duality:

  \[
  \begin{align*}
  \text{minimize} : & \quad c^T x \\
  \text{subject to} : & \quad A x = b \\
  \quad & \quad x \geq 0
  \end{align*}
  \]

  \[
  \begin{align*}
  \text{maximize} : & \quad b^T y \\
  \text{subject to} : & \quad A^T y = c \\
  \quad & \quad y \geq 0
  \end{align*}
  \]
Lagrange Multipliers

• Suppose you want to maximize $f(x)$
• Subject to an equality constraint $g(x)$

• Lagrange multipliers allow us to:
  – Convert this into a minimization problem
  – Simplify the problem by replacing $f(x)$ with constraints on the gradient of $f(x)$

The Constraint Surface I

• From Bishop:

• Observation 1:
  – Gradient of $g$ is normal to the constraint surface
  – Don’t forget: $g$ is a surface defined by an equality
  – $x^2+y^2=c$ is very different from $z=x^2+y^2$
Constraint Surface II

- From Bishop:

- Gradient of objective must also be normal at optimum
- Proof by contradiction: If it were not, you could take a step in the direction of the gradient, then project onto the constraint surface to improve your objective function

Gradients must be parallel

- From Bishop:

- For any optimal solution: \( \nabla f + \lambda \nabla g = 0, \lambda \neq 0 \)
Defining the Lagrangian

- We have: \( \nabla f + \lambda \nabla g = 0, \lambda \neq 0 \)
- Define: \( L(x, \lambda) = f(x) + \lambda g(x) \)
- Objective:

\[
\nabla L(x, \lambda) = 0
\]

\[
\downarrow
\]

\[
\nabla L_x = \nabla f + \lambda \nabla g = 0
\]

\[
\nabla L_\lambda = \frac{\partial L}{\partial \lambda} = g(x) = 0
\]

Lagrangian Example (from Bishop)

- Maximize: \( f(x_1, x_2) = 1 - x_1^2 - x_2^2 \)
- Subject to: \( g(x_1, x_2) = x_1 + x_2 - 1 = 0 \)
- Lagrangian: \( L(x, \lambda) = 1 - x_1^2 - x_2^2 + \lambda (x_1 + x_2 - 1) \)
- Solve for:

\[
\begin{align*}
x_1 + x_2 - 1 &= 0 \\
-2x_1 + \lambda &= 0 \\
-2x_2 + \lambda &= 0
\end{align*}
\]
Lagrange Multiplier Summary

• Converts a max to a min (generalization of this is called taking the Lagrange Dual)

• Can convert quadratic objectives to linear constraints

• Very useful trick!

KKT Conditions

• KKT conditions generalize Lagrange multipliers to include inequality constraints
• Given:
  – Maximize f(x)
  – Subject to: g_i(x)=0, 1≤i≤k
  – Subject to: h_j(x)≥0, 1≤j≤k
• Transformed problem:
  – Minimize:
    $$f(x) + \sum_{i=1}^{k_1} \lambda_i g_i(x) + \sum_{j=1}^{k_2} \mu_j h_j(x)$$
  – Subject to: $\mu_i ≥ 0$
  – Subject to: $\mu_k h_k=0$
Understanding KKT conditions

\[ f(x) + \sum_{i=1}^{k_1} \lambda_i g_i(x) + \sum_{j=1}^{k_2} \mu_j h_j(x) \]

- When no inequality constraints (h) or when inequality constraints are tight, same as Lagrange multipliers
- \( \mu_k \geq 0; \mu_k h_k = 0 \) enforce “complementarity”
- If constraint is not tight, \( \mu_k \) must be 0

Optimization Summary

- Optimization techniques are very powerful tools to have in your toolbox
- Constrained optimization problems occur frequently in science and engineering, \textit{not just machine learning}
- Convex optimization problems can be solved efficiently
- Many useful tricks for creating, manipulating convex optimization problems
  - Duality
  - Lagrange multipliers
  - KKT conditions
  - etc.
What You Need to Know for CPS 271

• Essential:
  – LP, QP
  – Duality

• Used, but not exercised/quizzed:
  – Lagrange multipliers
  – KKT conditions

• General Knowledge:
  – QCQPs
  – (M)Ips
  – General convex optimization issues