Outline

Introduction to optimization

Types of optimization problems, convexity

Solving optimization problems
Logistics

HW0, some unintentional ambiguity about “no late days” criteria

To be clear, in all future assignments, the policy is:

You have 5 late days, no more than 2 on any assignment
If you use up your five late days, you will receive 20% off per day for these two days
If you submit any homework more than 2 days late, you will receive zero credit

All homework, both programming and written portions, must be written up independently

All students who submitted HW0 have been taken off waitlist
Outline

Introduction to optimization

Types of optimization problems, convexity

Solving optimization problems
Continuous optimization

The problems we have seen so far (i.e., search) in class involve making decisions over a discrete space of choices.

An amazing property:

<table>
<thead>
<tr>
<th></th>
<th>Discrete search</th>
<th>(Convex) optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables</td>
<td>Discrete</td>
<td>Continuous</td>
</tr>
<tr>
<td># Solutions</td>
<td>Finite</td>
<td>Infinite</td>
</tr>
<tr>
<td>Solution complexity</td>
<td>Exponential</td>
<td>Polynomial</td>
</tr>
</tbody>
</table>

One of the most significant trends in AI in the past 15 years has been the integration of optimization methods throughout the field.
Optimization definitions

We’ll write optimization problems like this:

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad x \in \mathcal{C}
\end{align*}
\]

which should be interpreted to mean: we want to find the value of \(x\) that achieves the smallest possible value of \(f(x)\), out of all points in \(\mathcal{C}\)

Important terms:

- \(x \in \mathbb{R}^n\) – optimization variable (vector with \(n\) real-valued entries)
- \(f: \mathbb{R}^n \rightarrow \mathbb{R}\) – optimization objective
- \(\mathcal{C} \subseteq \mathbb{R}^n\) – constraint set
- \(x^* \equiv \arg\min_{x \in \mathcal{C}} f(x)\) – optimal solution
- \(f^* \equiv f(x^*) \equiv \min_{x \in \mathcal{C}} f(x)\) – optimal objective
Example: Weber point

Given a collection of cities (assume on 2D plane) how can we find the location that minimizes the sum of distances to all cities?

Denote the locations of the cities as \( y^{(1)}, \ldots, y^{(m)} \)

Write as the optimization problem:

\[
\min_x \sum_{i=1}^{m} \|x - y^{(m)}\|_2
\]
Example: image deblurring

Given corrupted image $Y \in \mathbb{R}^{m \times n}$, reconstruct image by solving optimization problem:

$$\min_X \sum_{i,j} |Y_{i,j} - (K \ast X)_{i,j}| + \lambda \sum_{i,j} \left( (X_{i,j} - X_{i,j+1})^2 + (X_{i+1,j} - X_{i,j})^2 \right)^{\frac{1}{2}}$$

where $K \ast$ denotes convolution with a blurring filter.
Example: robot trajectory planning

Many robotic planning tasks are more complex than shortest path, e.g. have robot dynamics, require “smooth” controls.

Common to formulate planning problem as an optimization task.

Robot state $x_t$ and control inputs $u_t$

$$
\begin{align*}
\text{minimize} & \quad \sum_{i=1}^{T} \|u_t\|_2^2 \\
\text{subject to} & \quad x_{t+1} = f_{\text{dynamics}}(x_t, u_t) \\
& \quad x_t \in \text{FreeSpace}, \forall t \\
& \quad x_1 = x_{\text{init}}, \quad x_T = x_{\text{goal}}
\end{align*}
$$

Figure from (Schulman et al., 2014)
As we will see in much more detail shortly, virtually all (supervised) machine learning algorithms boil down to solving an optimization problem

$$\min_{\theta} \sum_{i=1}^{m} \ell(h_{\theta}(x^{(i)}), y^{(i)})$$

Where $x^{(i)} \in X$ are inputs, $y^{(i)} \in Y$ are outputs, $\ell$ is a loss function, ad $h_{\theta}$ is a hypothesis function parameterized by $\theta$, which are the parameters of the model we are optimizing over.

Much more on this soon
One of the key benefits of looking at problems in AI as optimization problems: we separate out the **definition** of the problem from the **method for solving it**.

For many classes of problems, there are off-the-shelf solvers that will let you solve even large, complex problems, once you have put them in the right form.
Outline

Introduction to optimization

Types of optimization problems, convexity

Solving optimization problems
Classes of optimization problems

Many different names for types of optimization problems: linear programming, quadratic programming, nonlinear programming, semidefinite programming, integer programming, geometric programming, mixed linear binary integer programming (the list goes on and on, can all get a bit confusing)

We’re instead going to focus on two dimensions: convex vs. nonconvex and constrained vs. unconstrained
In unconstrained optimization, every point $x \in \mathbb{R}^n$ is feasible, so singular focus is on minimizing $f(x)$

In contrast, for constrained optimization, it may be difficult to even find a point $x \in C$

Often leads to very different methods for optimization (more next lecture)
Convex vs. nonconvex optimization

Originally, researchers distinguished between linear (easy) and nonlinear (hard) problems.

But in 80s and 90s, it became clear that this wasn’t the right distinction, key difference is between convex and nonconvex problems.

Convex problem:

\[
\min x f(x)
\]

subject to \( x \in \mathcal{C} \)

Where \( f \) is a convex function and \( \mathcal{C} \) is a convex set.
Convex sets

A set $\mathcal{C}$ is convex if, for any $x, y \in \mathcal{C}$ and $0 \leq \theta \leq 1$

$$\theta x + (1 - \theta)y \in \mathcal{C}$$

Examples:

- All points $\mathcal{C} = \mathbb{R}^n$
- Intervals $\mathcal{C} = \{ x \in \mathbb{R}^n \mid l \leq x \leq u \}$ (elementwise inequality)
- Linear equalities $\mathcal{C} = \{ x \in \mathbb{R}^n \mid Ax = b \}$ (for $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$)
- Intersection of convex sets $\mathcal{C} = \bigcap_{i=1}^{m} \mathcal{C}_i$
Convex functions

A function \( f: \mathbb{R}^n \to \mathbb{R} \) is convex if, for any \( x, y \in \mathbb{R}^n \) and \( 0 \leq \theta \leq 1 \)
\[
f(\theta x + (1 - \theta) y) \leq \theta f(x) + (1 - \theta) f(y)
\]

Convex functions “curve upwards” (or at least not downwards)

If \( f \) is convex then \(-f\) is concave

If \( f \) is both convex and concave, it is affine, must be of form:
\[
f(x) = \sum_{i=1}^{n} a_i x_i + b
\]
Examples of convex functions

Exponential: \( f(x) = \exp(ax), \ a \in \mathbb{R} \)

Negative logarithm: \( f(x) = -\log x \), with domain \( x > 0 \)

Squared Euclidean norm: \( f(x) = \|x\|_2^2 \equiv x^T x \equiv \sum_{i=1}^{n} x_i^2 \)

Euclidean norm: \( f(x) = \|x\|_2 \)

Non-negative weighted sum of convex functions

\[
f(x) = \sum_{i=1}^{m} w_i f_i(x), \quad w_i \geq 0, \ f_i \text{ convex}
\]
Poll: convex sets and functions

Which of the following functions or sets are convex

1. A union of two convex sets \( C = C_1 \cup C_2 \)

2. The set \( \{ x \in \mathbb{R}^2 | x \geq 0, x_1 x_2 \geq 1 \} \)

3. The function \( f: \mathbb{R}^2 \rightarrow \mathbb{R}, f(x) = x_1 x_2 \)

4. The function \( f: \mathbb{R}^2 \rightarrow \mathbb{R}, f(x) = x_1^2 + x_2^2 + x_1 x_2 \)
Convex optimization

The key aspect of convex optimization problems that make them tractable is that all local optima are global optima.

**Definition:** a point $x$ is globally optimal if $x$ is feasible and there is no feasible $y$ such that $f(y) < f(x)$.

**Definition:** a point $x$ is locally optimal if $x$ is feasible and there is some $R > 0$ such that for all feasible $y$ with $\|x - y\|_2 \leq R$, $f(x) \leq f(y)$.

**Theorem:** for a convex optimization problem all locally optimal points are globally optimal.
Proof of global optimality

**Proof:** Given a locally optimal $x$ (with optimality radius $R$), and suppose there exists some feasible $y$ such that $f(y) < f(x)$

Now consider the point

$$z = \theta x + (1 - \theta) y, \quad \theta = 1 - \frac{R}{2\|x - y\|_2}$$

1) Since $x, y \in C$ (feasible set), we also have $z \in C$ (by convexity of $C$)

2) Furthermore, since $f$ is convex:

$$f(z) = f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta) f(y) < f(x) \quad \text{and}$$

$$\|x - z\|_2 = \left\|x - \left(1 - \frac{R}{2\|x-y\|_2}\right)x + \frac{R}{2\|x-y\|_2} y\right\|_2 = \left\|\frac{R(x-y)}{2\|x-y\|_2}\right\|_2 = \frac{R}{2}$$

Thus, $z$ is feasible, within radius $R$ of $x$, and has lower objective value, a contradiction of supposed local optimality of $x$
Outline

Introduction to optimization

Types of optimization problems, convexity

Solving optimization problems
A key concept in solving optimization problems is the notation of the gradient of a function (multi-variate analogue of derivative)

For $f: \mathbb{R}^n \rightarrow \mathbb{R}$, gradient is defined as vector of partial derivatives

$$\nabla_x f(x) \in \mathbb{R}^n = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \frac{\partial f(x)}{\partial x_2} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix}$$

Points in “steepest direction” of increase in function $f$
Gradient motivates a simple algorithm for minimizing $f(x)$: take small steps in the direction of the negative gradient

**Algorithm:** Gradient Descent  
**Given:**  
Function $f$, initial point $x_0$, step size $\alpha > 0$  
**Initialize:**  
$x \leftarrow x_0$  
**Repeat until convergence:**  
$x \leftarrow x - \alpha \nabla_x f(x)$

“Convergence” can be defined in a number of ways
Gradient descent works

**Theorem:** For differentiable \( f \) and small enough \( \alpha \), at any point \( x \) that is not a (local) minimum

\[
    f(x - \alpha \nabla_x f(x)) < f(x)
\]

i.e., gradient descent algorithm will decrease the objective

**Proof:** Any differentiable function \( f \) can be written in terms of its Taylor expansion: \( f(x + v) = f(x) + \nabla_x f(x)^T v + O(\|v\|^2_2) \)
Gradient descent works (cont)

Choosing \( v = -\alpha \nabla_x f(x) \), we have

\[
\begin{align*}
    f(x - \alpha \nabla_x f(x)) &= f(x) - \alpha \nabla_x f(x)^T \nabla_x f(x) + O(\|\alpha \nabla_x f(x)\|_2^2) \\
    &\leq f(x) - \alpha \|\nabla_x f(x)\|^2_2 + C \|\alpha \nabla_x f(x)\|^2_2 \\
    &= f(x) - (\alpha - \alpha^2 C) \|\nabla_x f(x)\|^2_2 \\
    &< f(x) \quad \text{(for } \alpha < 1/C \text{ and } \|\nabla_x f(x)\|^2_2 > 0) 
\end{align*}
\]

(Watch out: a bit of subtlety of this line, only holds for small \( \alpha \nabla_x f(x) \))

We are guaranteed to have \( \|\nabla_x f(x)\|^2_2 > 0 \) except at optima

Works for both convex and non-convex functions, but with convex functions guaranteed to find global optimum
Poll: modified gradient descent

Consider an alternative version of gradient descent, where instead of choosing an update $x - \alpha \nabla_x f(x)$, we choose some other direction $x + \alpha v$ where $v$ has a negative inner product with the gradient $\nabla_x f(x)^T v < 0$

Will this update, for suitably chosen $\alpha$, still decrease the objective?

1. No, not necessarily (for either convex or nonconvex functions)
2. Only for convex functions
3. Only for nonconvex functions
4. Yes, for both convex and nonconvex functions
Gradient descent in practice

Choice of $\alpha$ matters a lot in practice:

$$\min_{x} 2x_1^2 + x_2^2 + x_1 x_2 - 6x_1 - 5x_2$$

$\alpha = 0.05$

$\alpha = 0.2$

$\alpha = 0.42$
Dealing with constraints, non-differentiability

For settings where we can easily project points onto the constraint set $\mathcal{C}$, can use a simple generalization called \textit{projected gradient descent}

$$\text{Repeat: } x \leftarrow P_{\mathcal{C}} \left( x - \alpha \nabla_x f(x) \right)$$

If $f$ is not differentiable, but continuous, it still has what is called a \textit{subgradient}, can replace gradient with subgradient in all cases (but theory/practice of convergence is quite different)
Optimization in practice

We won’t discuss this too much yet, but one of the beautiful properties of optimization problems is that there exists a wealth of tools that can solve them using very simple notation

Example: solving Weber point problem using cvxpy (http://cvxpy.org)

```python
import numpy as np
import cvxpy as cp

n,m = (5,10)
y = np.random.randn(n,m)
x = cp.Variable(n)
f = sum(cp.norm2(x - y[:,i]) for i in range(m))
prob = cp.Problem(cp.Minimize(f), []
prob.solve()
```