Optimization

More advanced topics in optimization

Integer programs

- Many combinatorial optimization problems can be represented as **integer programs**
- We already thought about finding large independent sets:
  - Given a graph $G = (V, E)$, find $S \subseteq V$ of largest size such that $\forall uv \in E$ at most one of u, v is in S
- Encode S via $x_u = 1$ for $u \in S$ and $x_u = 0$ otherwise

Maximize $\sum_u x_u$ such that $0 \leq x \leq 1$, $\forall uv \in E.\ x_u + x_v \leq 1$, and $x \in \mathbb{Z}$

This looks just like an LP in the sense that the objective and constraint are linear

But the variables are not a real vector x, the entries are forced to be in $\mathbb{Z}$ (in fact, in $\{0,1\}$)

Example integer programs

Max matching: maximize $\sum_{uv\in E} x_{uv}$ such that $x \in \{0,1\}^E$ and $\forall u \in V, \sum_{v\in N(u)} x_{uv} \leq 1$

Max independent set: maximize $\sum_{u\in V} x_u$ such that $x \in \{0,1\}^V$ and $\forall uv \in E, x_u + x_v \leq 1$

Chromatic number: minimize $\sum_{I\in\Omega} w_I$ such that $w \in \{0,1\}^\Omega$ and $\forall v \in V, \sum_{I\in\Omega \text{ s.t. } v\in I} w_I \geq 1$
(where Ω is the set of all independent sets in G)

Max cut: maximize $\sum_u \sum_v \frac{1-x_u x_v}{2}$ such that $x_u \in \{-1,1\}$

These are not necessarily easy to! But stating optimization problems such as this in simple form gives us some ideas…

Relaxation

- An **integer linear program (ILP)** has the form $\max x$ such that $x \geq 0$, $Ax \leq b$, and $x \in \mathbb{Z}^n$
- The **linear relaxation** of an ILP is the LP you get by dropping the condition $x \in \mathbb{Z}^n$
- This can **enlarge** the feasible set, so the solution to the LP is always at least the solution to the ILP
- Sometimes it is useful to solve the relaxation and then **round** the solution back to an integer
- Sometimes no rounding is necessary
- Let's explore our examples

Max bipartite matching

Let $G = (V, E)$ be a graph

Let $M = \max \sum_{uv \in E} x_{uv}$ such that $\forall u \in V. \sum_{v \in N(u)} x_{uv} \leq 1$ and $x \in \{0,1\}^E$

This integer program should be scary, but what happens if we drop the integrality constraint?

Let $M^* = \max \sum_{uv \in E} x_{uv}$ such that $\forall u \in V. \sum_{v \in N(u)} x_{uv} \leq 1$ and $x \geq 0$ (the fact that $x_{uv} \in [0,1]$ follows from these)

The **integrality gap** is M^*/M: the ratio by which you hurt the objective when you relax the integrality constraint.

One indication that the bipartite matching problem is easy to solve is that we can prove that the integrality gap is 1, which means there is no gap!

The fractional matching polytope

A **polytope** is the multidimensional generalization of a polygon or polyhedron. It is a shape bounded by hyperplanes.

- A **hyperplane** is the set of vectors in $\mathbb{R}^n$ which satisfy a single linear equation, e.g. $x + y = 1$ in $\mathbb{R}^2$
- A **polytope** is the set of vectors in $\mathbb{R}^n$ satisfying $Ax \leq b$ for some $m \times n$ matrix A and $b \in \mathbb{R}^m$
- Each constraint equation gives us that the polytope lies on one side of a hyperplane
- This looks just like linear programming!
- An LP is an optimization problem whose feasible set is a polytope and whose objective function is linear

Given a graph $G = (V, E)$, the **fractional matching polytope** is the feasible set of the maximum fractional matching LP:

$$Ax \leq 1$$

Where $x \in \mathbb{R}^E$, $x \geq 0$ and A is the **incidence** matrix indexed by $V \times E$ such that $A_{u,e} = 1 \Leftrightarrow u \in e$

Extreme points

You probably know what we mean by the corner of a polygon, and perhaps a polyhedron.

The multidimensional generalization is an **extreme point**. This is a point of the polytope that cannot be written as a *convex combination* of any other points in the polyhedron.

This means x^* is an extreme point of the polytope P if and only if whenever we can find an equation

$$x^* = \sum_i t_i y_i$$

Where we have $t_i \in [0,1]$, $\sum_i t_i = 1$, and $y_i \in P$, it turns out that some $y_j = x^*$ and $t_j = 1$.

Integrality for bipartite matchings

The fractional matching polytope: $P = \{x \in \mathbb{R}^E : x \geq 0 \text{ and } \forall u. \sum_{v \in N(u)} x_{uv} \leq 1\}$

Theorem. If G is a bipartite graph, then every extreme point x^* in the fractional matching polytope has integer coordinates.

Proof. By contradiction. Suppose not and let $x^* \in P$ be extreme with some non-integer entry. Let $F \subset E$ be the nonempty set of coordinates where the entries are in $(0,1)$.

We claim that a vertex cannot be in only one edge of F. If there is a vertex u incident to exactly one fractional edge uv then all the other edges incident to u must have value 0 by the constraint. But then we have $0 < x^*_{uv} < 1$ and must have some wiggle room: there is some small $\epsilon > 0$ such that increasing or decreasing x^*_{uv} by ϵ keeps us in P. This contradicts x^* being extreme.

Then F must be a nonempty union of cycles. Since G is bipartite there are no odd cycles.

Let $C = \{e_1, \dots e_{2k}\}$ be an even cycle whose edges are in F. Let $\epsilon > 0$ be s.t. $\forall e \in C$ we have $\epsilon < x^*_e < 1 - \epsilon$.

Then if we change the x^* entry by $+\epsilon$ on even edges and $-\epsilon$ on odd edges we have not violated any constraints. Again, this contradicts x^* being extremal. □

Non-integrality for general matchings

The fractional matching polytope: $P = \{x \in \mathbb{R}^E : \ x \geq 0 \text{ and } \forall u. \sum_{v \in N(u)} x_{uv} \leq 1\}$

Example. There are non-bipartite graphs whose fractional matching polytope is not integral.

Augmenting paths

The following theorem underpins the correctness of the bipartite matching algorithm we discussed.

Theorem. A matching in **any** graph G is maximum if and only if there are no M**-**augmenting paths.

Proof. The easy direction is ”only if”. The contrapositive is that if there is an M-augmenting path then M is not maximum. But this is obvious because you can flip the edges on the path and increase the size of the matching.

The if direction is tougher. Suppose for contradiction that M is maximum but there are no M-augmenting paths. The trick is to consider a maximum matching M^* and the symmetric difference $D = M \oplus M^*$.

Since M and M^* are matchings, D must be a subgraph of G of maximum degree 2.

Then D consists of paths and cycles, and these paths and cycles must alternate between edges of M and M^*.

Then any cycle must be even.

We have $|M^*| > |M|$ so there must be some path in D with more edges of M^* than of M.

But this path is therefore an M-augmenting path. □

This theorem is suspiciously similar to the integrality one…

Examples

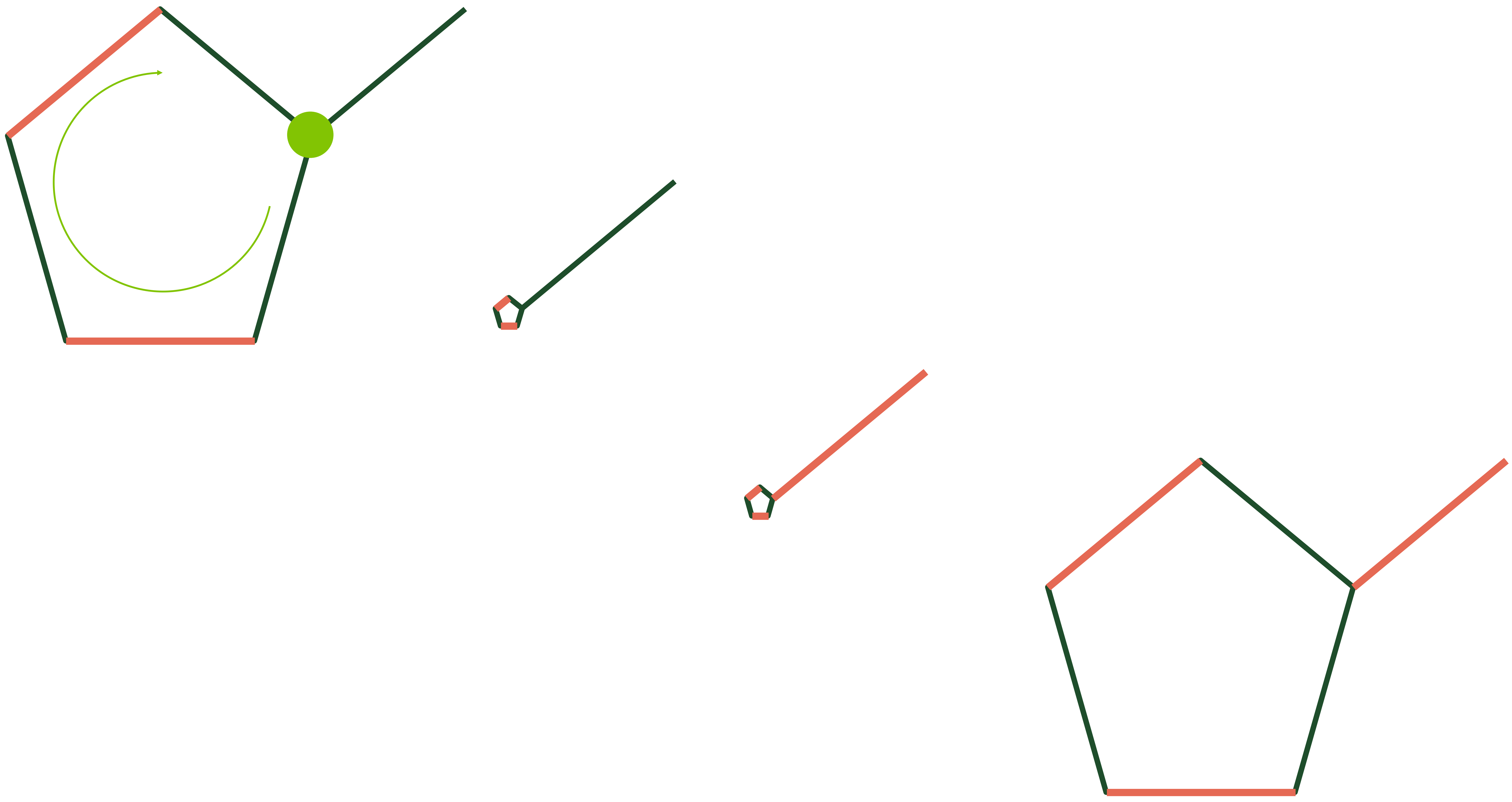
Non-bipartite graphs I

- It turns out that we can efficiently find maximum matchings in non-bipartite graph
- There are a few useful perspectives on how to do this
- The simplest is to try to fix the LP for matchings
- Recall that on an odd cycle we can end up with a fractional extreme point where $x_e = 1/2$ for all edges
- We can "chop off" these extreme points with **blossom inequalities**
- For any set of vertices $S \subseteq V$ with odd size, we add $\sum_{e \in E(S)} x_e \leq \frac{|S|-1}{2}$
- It should be clear that this does get rid of the non-integral extreme points of the type we identified
- But it's quite a bit harder to prove that the extreme points of this new **general matching polytope** are integral

Non-bipartite graphs II

- Worse, there are now exponentially many constraints, so solving this LP will be slow.
- We know that we can still increase the size of the matching via augmenting paths, but it’s now no longer clear how to find augmenting paths efficiently.
- Edmonds gave an algorithm for general maximum matching which essentially shows that you can contract odd cycles into vertices while you search for augmenting paths.
- The key is that when you search for augmenting paths from a free vertex v (e.g. with BFS), if you find an odd cycle then you immediately contract it

Examples



What about independent sets?

- Independent sets are quite a lot like matchings.
- We have the program max $\sum_u x_u$ such that $x_u \in \{0,1\}$ and $\forall uv \in E\ x_u + x_v \leq 1.$
- We have the same problems with fractional independent sets as for fractional matchings
- But there is no blossom algorithm for independent sets: it genuinely is NP-hard to solve this ILP

What about max-cut?

- The max-cut program was much more interesting, it's objective was **quadratic** (and there are no constraints other than the domain)

$$\text{maximize } \sum_u \sum_v \frac{1 - x_u x_v}{2} \text{ such that } x_u \in \{-1, 1\}$$

It's NP-hard to compute max-cut, but we can approximate it with more serious ideas from optimization.

Quadratic optimization

You probably know that some quadratic optimization problems are easily solvable

- minimize $x^2 - 2x + 1$ over $x \in \mathbb{R}$

- minimize $x^2 + 2x + 1$ over $x \geq 0$

Semidefinite programs I

To approximate max-cut we can move to higher dimensions.

We can think of $\{-1,1\}$ as the surface of the **1-dimensional unit sphere:** the set of points at distance one from zero

What if we let x_u take values in a d-dimensional sphere? That is, $x_u \in \mathbb{R}^d$ with $\|x_u\| = 1$.

For an edge uv consider scalar product $x_u \cdot x_v$. If $x_u = x_v$ then this is just 1 by our constraint, and if they're on opposite sides of the sphere we get -1. This means we can generalize the cut value $\sum_{uv\in E} \frac{1 - x_u \cdot x_v}{2}$

Then we want to maximize $\sum_{uv\in E} \frac{1 - x_u \cdot x_v}{2}$ such that $\forall u\ \|x_u\| = 1$.

But we can put all this in a matrix. Let X be the $n \times n$ matrix with $X_{u,v} = x_u \cdot x_v$. Now our objective and constraint are linear:

maximize $\sum_{uv\in E} \frac{1 - X_{u,v}}{2}$ such that $X_{u,u} = 1$.

This still doesn't work, because allowing X to be any matrix with these properties is too loose.

Semidefinite programs II

maximize $\sum_{uv \in E} \frac{1 - X_{u,v}}{2}$ such that $X_{u,u} = 1$.

It turns out that there's a special property of all square matrices X formed by vectors x_u such that $X_{u,v} = x_u \cdot x_v$

These matrices are called **Gram matrices**, and they are **positive semidefinite (PSD)**.

Definition. A symmetric matrix A is **positive semidefinite** if, for all nonzero vectors x we have $x^T A x \geq 0$.

Theorem. A symmetric matrix is PSD if and only if all its eigenvalues are nonnegative.

Theorem. Any Gram matrix is PSD.

Proof. Let $y \neq 0$.

Then $y^T X y = \sum_{u,v} X_{u,v} y_u y_v = \sum_{u,v} (x_u \cdot x_v) y_u y_v = \sum_{u,v} (y_u x_u) \cdot (y_v x_v) = (\sum_u y_u x_u) \cdot (\sum_v y_v x_v) = \|\sum_u y_u x_u\|^2 \geq 0.$ □

We write $X \succcurlyeq 0$ to mean that X is PSD.

Semidefinite program for max-cut

maximize $\sum_{uv \in E} \frac{1 - X_{u,v}}{2}$ such that $X \succcurlyeq 0$ and $\forall u \; X_{u,u} = 1$.

There are general-purpose algorithms that efficiently solve SDPs, analogous to the ones for LPs.

The intuition is that an LP for max-cut would smear $\{-1,1\}$ onto the line $[-1,1]$ and but then 0 is valid but useless.

The SDP blows the problem up to the d-dimensional sphere, and the best choice of d is the number of vertices.

The problem: The optimum in the SDP is not a cut, we need to make a binary decision for each vertex.

The solution: Randomized rounding!

Take X that solves the SDP and then choose a hyperplane through the origin uniformly at random. Vertices on one side of the plane get 1, the others get -1.

Cholesky decomposition and rounding

- There is a fancy procedure called Cholesky decomposition that, given the Gram matrix X that solves the SDP, tells us the underlying vectors x_u that we want.
- For the rounding we choose r such that each entry is an independent standard Gaussian
- Then $r/\|r\|$ is uniform over the sphere
- If $x_u \cdot r \geq 0$ set $u \in S$, otherwise $u \notin S$
- The key thing we have to compute is the probability that an edge is cut

Goemans–Williamson rounding

- Let θ_{uv} be the angle between x_u and x_v, this is $\arccos x_u \cdot x_v$
- Intuitively, the larger the angle (up to π or $180°$) the more likely uv is to be cut by the random direction
- The true probability is $\frac{\theta_{uv}}{\pi}$

In terms of the angles, the SDP objective is $opt^* = \sum_{uv} \frac{1-\cos\theta_{uv}}{2}$ but the actual cut we get is $\sum_{uv} \frac{\theta_{uv}}{\pi}$

Let opt be the optimal max-cut in G and let $\alpha = \min_{\theta} \frac{2\theta}{\pi(1-\cos\theta)} \approx 0.878$.

Since the true max-cut is feasible in the SDP we have $opt^* \geq opt$.

Then the expected size of our cut is at least $\alpha\, opt^* \geq \alpha\, opt$.