TDA206 - Discrete Optimization
2017-01-16

What is the course about?

Consider a pharmacy company, which produces x boxes of medication. They want to sell these boxes in order to maximise their profit.

Market price: $p = 900 - 0.4x$
Production cost: $c = 100 + 0.6x$
Profit: $x(p - c) = f(x)$

Objective: $\underset{x \in \mathbb{Z}}{max}\, x(800 - x)$

Given that:
$x \leq 500$
$x \geq 0$
$10 \mid x$

In the general form, we have want to: $\underset{x \in \psi}{opt}\, f(x)$, where ψ is the domain, $f(x)$ is the objective function and we either want to min or max said function. Most usually, we will consider min, since $max\, f(x) = -min\, f(x)$. We also concern ourselves with constraints $P[x]$ (predicates or propositions). $\{x \mid p \in P[x]\}$ is called the feasible set.

Convex Functions

Wikipeia: *In mathematics, a real-valued function defined on an interval is called convex (or convex downward or concave upward) if the line segment between any two points on the graph of the function lies above or on the graph, in a Euclidean space (or more generally a vector space) of at least two dimensions.*

$\exists.x, y$: "line between x and y" $\subsetneq N$, where N is the region.
$\forall.x, y$: line $\subseteq N$, where N is the region.

In order to figure out if every dot on the line L between x and y is in N, we take: $\lambda x + (1 - \lambda)y$, where $\lambda \in [0, 1]$. The interval $[0, 1]$ ensures we only

check the line L between x and y.

A set is $C \subseteq \mathbb{R}^n$ is convex iff: $\forall .x, y \in C, \lambda \in [0,1] : \lambda x + (1-\lambda)y \subseteq C$

Examples of Convex Sets

Norm Balls

The norm of a vector v, $||v||_L$. We will mostly be concerned with l^2-norms: $\sqrt{x \cdot x} = x^T x = xx =< x, x >= \sqrt{x_1^2 + x_2^2 + x_3^2 + ...}$.
More general: $l^p = (\sum_i |x_i|^p)^{1/p}$, and we're pretty much only concerned with: $l^1 = \sum_i |x_i|$ and l^∞.

An l^2-ball is a circle on the plane.
An l^1-ball is a diamond, inscribed in the l^2-ball.
An l^∞-ball is a square, surrounding the l^2-ball.

Properties of Norms

Positivity: $||x|| \geq 0$
Scalability: $||\lambda x|| = |\lambda| \, ||x||$
Triangle in-eq (TIQ): $||x + y|| \leq ||x|| + ||y||$

Def: Norm Ball

$\{x \,|\, ||x|| \leq R\}$, where R is some radius.
We want to show that: $||\lambda x + (1-\lambda)y|| \leq R$

$$
\begin{aligned}
&||\lambda x + (1-\lambda)y|| \leq R \\
&\leq ||\lambda x|| + ||(1-\lambda)y|| \\
&= |\lambda| \, ||x|| + |1-\lambda| \, ||y|| \\
&\leq \lambda R + (1-\lambda)R \\
&= (\lambda + 1 - \lambda)R = R
\end{aligned}
$$

Def: Hyperplanes

$\{\vec{x} \,|\, \vec{a} \cdot \vec{x} = b\}$

Hyperplanes are all convex sets.

Def: Half spaces

$\{\vec{x} \,|\, \vec{a} \cdot \vec{x} \leq b\}$ (also $\geq$)

Convex Functions

One of the simplest convex functions is $f(x) = x^2$. We look at the epigraph of f. If you draw a line between two points on f we get a line: $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$.

Example - Norm

In order to show that $f(x) = ||x||$ is a convex function, we must show that: $||\lambda x + (1-\lambda)y|| \leq \lambda||x|| + (1-\lambda)||y||$. We can use the same approach as above. Since $\lambda \in [0,1]$ we get: $\lambda\,||x|| + (1-\lambda)\,||y||$.

Example - Alternate way

Given that $f(x)$ is differentiable (you can take a derivative of it), which ever derivative you take will be below the actual function. We can write this as: $f(y) \geq f(x) + \bigtriangledown f(x)(y-x)$.

2017-01-18

Gradient

$\bigtriangledown f(x) = \frac{df(x)}{dx}$

$$\bigtriangledown f(\vec{x}) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ . \\ . \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$$

Example

$f(x) = 2x_1^3 + 8x_2^2 - x_3 + 1$
$\frac{\partial f(x)}{\partial x_1} = 6x_1^2$
$\frac{\partial f(x)}{\partial x_2} = 16x_2$
$\frac{\partial f(x)}{\partial x_3} = -1$

$$\bigtriangledown f(\vec{x}) = \begin{pmatrix} 6x_1^2 \\ 16x_2 \\ -1 \end{pmatrix}$$

Recap: Convex Function

Criteria for f to be convex:

1. $\forall x_1, x_2 \in \mathbb{R}^n,\ \lambda \in [0,1] :\ f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$
2. If f is differentiable, all the tangents of f will be below f.
   $\forall x, x_0\ f(x) \geq f(x_0) + \nabla f(x_0) \cdot (x - x_0)$
3. If the change of the slope is always increasing.
   If f is twice differentiable, then f is convex if $\nabla^2 f(x) \geq 0$.

Function: $f(x) = x^2$

Criteria 1:$(\lambda x_1 + (1-\lambda)x_2)^2 \leq \lambda x_1^2 + (1-\lambda)x_2^2$

$(\lambda x_1 + (1-\lambda)x_2)^2 = \lambda^2 x_1^2 + 2\lambda(1-\lambda)x_1x_2 + (1-\lambda)^2 x_2^2 \leq \lambda x_1^2 + (1-\lambda)x_2^2$
$0 \leq \lambda x_1^2 - \lambda^2 x_1^2 - 2\lambda(1-\lambda)x_1x_2 + (1-\lambda)x_2^2 - (1-\lambda)^2 x_2^2$
$0 \leq \lambda x_1^2(1-\lambda) - 2\lambda(1-\lambda)x_1x_2 + (1-\lambda)x_2^2(1-(1-\lambda))$
$0 \leq \lambda(1-\lambda)x_1^2 - \lambda(1-\lambda)x_1x_2 + \lambda(1-\lambda)x_2^2$
$0 \leq \lambda(1-\lambda)(x_1^2 - 2x_1x_2 + x_2^2)$
$0 \leq \lambda(1-\lambda)(x_1 - x_2)^2$

Criteria 2: $x^2 \geq x_0^2 + \frac{dx_0^2}{dx} \cdot (x - x_0)$

$x^2 \geq x_0^2 + 2x_0 \cdot (x - x_0)$
$x^2 \geq x_0^2 + 2xx_0 - 2x_0^2$
$x^2 - 2xx_0 + x_0^2 \geq 0$
$(x - x_0)^2 \geq 0$

Criteria 3: $\nabla^2 f(x) \geq 0$

$\nabla^2 f(x) = \nabla^2 x^2 = 2 \geq 0$

Optimization

Wikipedia
An optimization problem is a problem on the form:

$$\begin{aligned} &\underset{x \in \psi}{opt}\, f(x) \\ s.t.\ &P[x] \end{aligned}$$

$\Omega \subseteq \psi$, the feasible set

Def: A convex opt. problem is one in which:
1. $\psi = \mathbb{R}^n$
2. $f(x)$ is convex
3. Ω is convex

Note: Make sure that your SW either checks 2 and 3, or that you know that beforehand.

Also, an opt. problem is convex if either:

a) $g_i(x) \leq b_i$ and g_i is convex

If $g_i(x)$ is convex, we're good, but if g_i is not convex, we might be okay if Ω is convex (a continuous line). You must prove that b_i "gives you" a convex Ω.

b) $h_j(x) = b_i$ and h_j is affine

Theorem

The family of convex sets is closed under intersection.

If we have to convex sets A and B, and in the intersection between A and B we have two points x and y. We know that: $x, y \in A$ and $x, y \in B$, as well as $\mathcal{L}_A \subseteq A$ and $\mathcal{L}_B \subseteq B$. Since a line segment between two points in $\mathbb{R}$ is unique, we know that $\mathcal{L}_A = \mathcal{L}_B = \mathcal{L} \subseteq A \cap B$.

If we want to define a convex region, without using complex schtuff, we can use half spaces – which are always convex. Since a half space is always convex, and an intersection between arbitrary many convex sets is convex, we can always guarantee a convex feasible region (convex polytope). Using this method we can create reasonable feasible regions with very few constraints.

2017-01-23

Clarification regarding Laplacian and Hessian.

Linear Programming

Generally we would write our optimization problem as:

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{opt} \quad & f \\ s.t \quad & P[x] \end{aligned}$$

In linear programming we wolud have:
$f = c^\intercal \vec{x}$ (linear function)
$P[x]$: $a_i^\intercal \vec{x} \leq b_i$ and $d_j^\intercal \vec{x} = e$

Our feasible region will be given by the constraints.

Example: Diet Problem

We want to minimize the cost for feeding the army.

$$\begin{aligned} min \quad & c^\intercal x = c_1x_1 + c_2x_2 + ... + c_nx_n \\ s.t \quad & k_1x_1 + k_2x_2 + ... \geq cal \\ & 3000 \geq b_1x_1 + b_2x_2 + ... \geq 800 \\ & x_i \geq 0 \end{aligned}$$

Add constraints for protein, fat, vitamins, etc...

Different ways to program linearly

$$\begin{array}{cc} \underset{x\in\mathbb{R}^n}{opt} & c^\intercal x \\ s.t & d_i^\intercal x \leq b_i \overset{\rightarrow}{\Leftrightarrow} Ax\} \overset{\rightarrow}{\leq} b \\ & a_j^\intercal x \geq q_i \\ & d^\intercal x = e \end{array}$$

This is not unique, we could for insteance multiply d with a constant. There are various ways of writing this:

1. Standard/augmented form.
2. Canonical form

Standard / Augmented Form

$$\begin{array}{cc} max & c^\intercal x \\ s.t & Dx = e \\ & x \geq 0 \end{array}$$

Canonical Form

$$\begin{array}{cc} max & c^\intercal x \\ s.t & Ax \leq b \\ & x \geq 0 \end{array}$$

Go from min $\rightarrow$ max

You negate the objective function.

Invert the $\geq$

$a_i x \geq b \rightarrow -a_i x \leq -b$

Std to Can

$d_i x = e \rightarrow d_i x \leq e$ and $d_i x \geq e$

Unbounded $\rightarrow$bounded

$x \rightarrow x^+ - x^- = x,\ x^+, x^- \geq 0$

Can to Std

$a_i x \leq b_i \rightarrow a_i x = b_i - z_i,\ z_i \geq 0 \Rightarrow a_i x + z_i = b_i$

Duality

$$
\begin{aligned}
max \quad & x_1 + 2x_2 + x_3 + x_4 \\
s.t. \quad & x_1 + 2x_2 + x_3 \leq 2 \\
& x_2 + x_4 \leq 1 \\
& x_1 + 2x_3 \leq 1 \\
& x \geq 0
\end{aligned}
$$

Some solver claims: $x* = \begin{pmatrix} 1 \\ 0.5 \\ 0 \\ 0.5 \end{pmatrix}$, $f* = 2.5$, but how do we verify this?

We have these properties:
$a \leq b \wedge c \leq d \rightarrow a + c \leq b + d$
$ma \leq mb \wedge nc \leq nd \rightarrow ma + nc \leq mb + nd$

our goal is to find an upper bound on $f*$.

$$
\begin{aligned}
& \frac{1}{2}(x_1 + 2x_2 + x_3) \leq \frac{1}{2} * 2 \\
& x_2 + x_4 \leq 1 \\
& \frac{1}{2}(x_1 + 2x_3) \leq \frac{1}{2} \\
& \frac{1}{2}x_1 + \frac{1}{2}x_1 + x_2 + x_2 + \frac{1}{2}x_3 + x_3 + x_4 \leq 2.5 \\
& x_1 + 2x_2 + x_3 + x_4 \leq x_1 + 2x_2 + \frac{3}{2} + x_4 \leq 2.5
\end{aligned}
$$

$y_i =$ factors
$y_1(a_{11}x_1 + a_{12}x_2 + a_{13}x_{13}) \leq y_1 b_1$
$2(a_{21}x_1 + a_{22}x_2 + a_{23}x_{13}) \leq y_2 b_2$
$y \geq 0$

$y_1 a_{11}x_1 + y_2 a_{21}x_1 + y_1 a_{12}x_2 + y_2 a_{22}x_2 + y_1 a_{13}x_3 + y_2 a_{23}x_{23} \leq y^\intercal b$
- make upper bound to $c^\intercal x$
- make bound tight

$c_1x_1 + c_2x_2 + c_3x_3 \leq (y_1a_{11} + y_2a_{21})x_1 + (y_1a_{12} + y_2a_{22})x_2 + (y_1a_{13} + y_2a_{23})x_3$

This gives us new constraints!

$c_1 \leq y_1a_{11} + y_2a_{21}$

$c_2 \leq y_1 a_{12} + y_2 a_{22}$
$c_3 \leq y_1 a_{13} + y_2 a_{23}$

We get a new problem:

$$\begin{aligned} min \quad & y^\intercal b \\ s.t. \quad & c \leq y^\intercal A \\ & y \geq 0 \end{aligned}$$

This is a dual problem of primal.

Primal

If your primal problem is:

$$\begin{aligned} max \quad & c^\intercal x \\ s.t \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

then your dual is derived as follows:

$$\begin{aligned} c^\intercal x \leq \quad & y^\intercal A x \leq y^\intercal b \\ & y \geq 0 \\ \text{so...} & \\ c^\intercal x \leq \quad & (A^\intercal y)^\intercal x \leq y^\intercal b \\ c^\intercal \leq \quad & (A^\intercal y)^\intercal \\ c \leq \quad & A^\intercal y \end{aligned}$$

So the dual becomes:

$$\begin{aligned} min \quad & b^\intercal y \\ s.t \quad & A^\intercal y \geq c \\ & y \geq 0 \end{aligned}$$

2017-01-25

A note on duals

$$\begin{array}{cc} min & b^\intercal y \\ s.t & A^\intercal y \geq c \\ & y \geq 0 \end{array}$$

If $c \leq A^\intercal y$ is true, it must be the case that $c^\intercal \leq (A^\intercal y)^\intercal \Leftrightarrow c^\intercal \leq y^\intercal A$. We also have that $Ax \leq b$, hence: for any $y \geq 0$, $y^\intercal Ax \leq y^\intercal b$. Looking at our previous expression we have, for any $x \geq 0$: $c^\intercal x \leq y^\intercal Ax$. Combining these we get: $c^\intercal x \leq y^\intercal Ax \leq y^\intercal b$.

Theorem: Weak duality theorem

For a primal linear program $\mathcal{P}$, and its dual $\mathcal{D}$, the following holds: If x is a feasible solution in $\mathcal{P}$ and y is a feasible solution in $\mathcal{D}$, then $c^\intercal x \leq b^\intercal y$.

Def

$b^\intercal y - c^\intercal x$ is called the duality gap.

Theorem: Strong duality theorem

If the primal $\mathcal{P}$ has an optimal solution $x*$ the dual $\mathcal{D}$ has an optimal solution $y*$ s.t. the duality gap $c^\intercal x* = b^\intercal y*$. (I.e. the duality gap is zero.)

Complementary Slackness

Given a primal $\mathcal{P}$ and a dual $\mathcal{D}$, we have the following constraints:

$\mathcal{D}\backslash\mathcal{P}$	$x_1 \geq 0$	$x_2 \geq 0$	$x_3 \geq 0$	
$y_1 \geq 0$	a_{11}	a_{12}	a_{13}	$\leq b_1$
$y_2 \geq 0$	a_{21}	a_{22}	a_{23}	$\leq b_2$
	$\vee$	$\vee$	$\vee$	
min	c_1	c_2	c_3	

$c^\intercal x* = y*^\intercal Ax* = b^\intercal y*$
$c^\intercal x* = y*^\intercal Ax*$
$c^\intercal x* - y*^\intercal Ax* = 0$
$(c^\intercal - y*^\intercal A)x* = 0$
$\sum_j (c^\intercal - y*^\intercal A)_j * x_j = 0$
if $x_j* > 0 \Rightarrow (c^\intercal - y*^\intercal A) = 0$.

If a primal variable is non-zero, its corresponding dual constaint is binding/active (it is equal, not "inequal"). If the dual constraint has slack ($\neq$), then the primal variable is 0.

The minimum cost flow problem

- We have a set of vertices $\mathcal{V}$.
- Each vertex $v \in \mathcal{V}$ has a balance b.
- Each arc has a capacity c and a cost (price) p.
- Each arch has a flow x (the actual amount that is pushed through the arc).

We can describe this as an LP:

$$\begin{aligned} min \quad & \sum_a p_a \times x_a \\ s.t \quad & \forall v \in \mathcal{V} : \sum x_{\text{in}} - \sum x_{\text{out}} = b_v \\ & \forall a : x_a \leq c_a \\ & \forall a : x_a \geq 0 \end{aligned}$$

2017-01-30

Integer Linear Programs

ILPs are LPs with $\mathbb{Z}^n$ as domain instead of $\mathbb{R}^n$. However, ILPs are not convex – we cannot draw lines between dots in $\mathbb{Z}$! There is however ways of solving ILPs.

Solve ILPs using LP

One of them is LP Relaxations. Another way of solving ILPs is to ensure that all vertices in the LP polytop are integer, then we know that the opitmal solution $x*$ is integer.

We can ensure that each vertex is the intersection of n linearly independent hyperplanes. In 2D, this means that we need two planes, in $\mathbb{R}^3$ we need 3 planes, etc.

Formalizing

$Mx = b$, we want to make sure that for all integer b, x has an integer solution.

We use Cramer's Rule, which states that $M_i = M$ with column i replaced by b and $x_i = \frac{|M_i|}{|M|}$. We can ensure that this holds by making $|m| = \pm 1$.

Def:

A matrix M is called unimodular iff $|M| = \pm 1$.

Construct a polytope of integer vertices

Given the primal for an LP:

$$\begin{aligned} max \quad & c^\intercal x \\ s.t \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

we can rewrite this as:

$$\begin{bmatrix} A \\ -\varepsilon \end{bmatrix} x \leq \begin{bmatrix} b \\ 0 \end{bmatrix}$$

But, this is not the matrix we're looking for.

$\mathcal{S} \sqsubseteq \begin{bmatrix} A \\ -\varepsilon \end{bmatrix}$ is however the matrix we want.

We know that: $\mathcal{S}$ has to be a unimodular matrix. If we make sure that all invertible sub-matrices are unimodular we know that we get an integer polytope.

Def:

A matrix A is called totally unimodular (tum) iff all invertible sub-matrices $\mathcal{S} \sqsubseteq A$ are unimodular. The transpose is also tum. Equivalently: for all square $\mathcal{S}$, $|\mathcal{S}| \subseteq \{-1, 0, 1\}$.

Since this must hold for a 1×1 matrix, the entries in this matrix can only be one of $\{-1, 0, 1\}$.

Conclusion!

If ILP has tum $A \rightarrow$ Feasible region has only integer vertices $\rightarrow$ solve LP relation and $x_{ILP}* = x_{LP}*$.

Incidence Matrix

We define a matrix such that:

	sv	vt	st
s	1	0	1
v	-1	1	0
t	0	-1	-1

describes a graph where there is an edge (sv) that starts in s and ends in v.

Max flow problem

Given a flow graph we have some constraints:
volume: $x_{sv} - x_{vt} = 0$
flow: $x_{ij} \leq u_{ij}$
$x \geq 0$

Solve it using ILP

Incidence Matrix:

$$
\begin{aligned}
max \quad & c^\intercal x \\
s.t \quad & 0 \leq x_{sv} \leq 3 \\
& 0 \leq x_{vt} \leq 1 \\
& 0 \leq x_{st} \leq 2 \\
& 0 \leq x_{ts} \leq \infty
\end{aligned}
$$

$$
\begin{bmatrix} 1 & 1 & 0 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} x_{sv} \\ x_{st} \\ x_{vt} \\ x_{ts} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}
$$

$$
\varepsilon x \leq \begin{bmatrix} 3 \\ 2 \\ 1 \\ \infty \end{bmatrix}
$$

The dual variables for $\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$ is $\begin{bmatrix} \Pi_s \\ \Pi_v \\ \Pi_t \end{bmatrix}$ and the dual variables for $\begin{bmatrix} 3 \\ 2 \\ 1 \\ \infty \end{bmatrix}$ is $\begin{bmatrix} \alpha_{sv} \\ \alpha_{st} \\ \alpha_{vt} \\ \alpha_{vs} \end{bmatrix}$,
hence the dual becomes:

$$
\begin{aligned}
min \quad & (\vec{0}\vec{u}) \cdot (\vec{\Pi}\vec{\alpha}) = \vec{u}^\intercal \vec{\alpha} \\
s.t \quad & [A^\intercal E^\intercal] \begin{bmatrix} \Pi \\ \alpha \end{bmatrix} \geq 0
\end{aligned}
$$

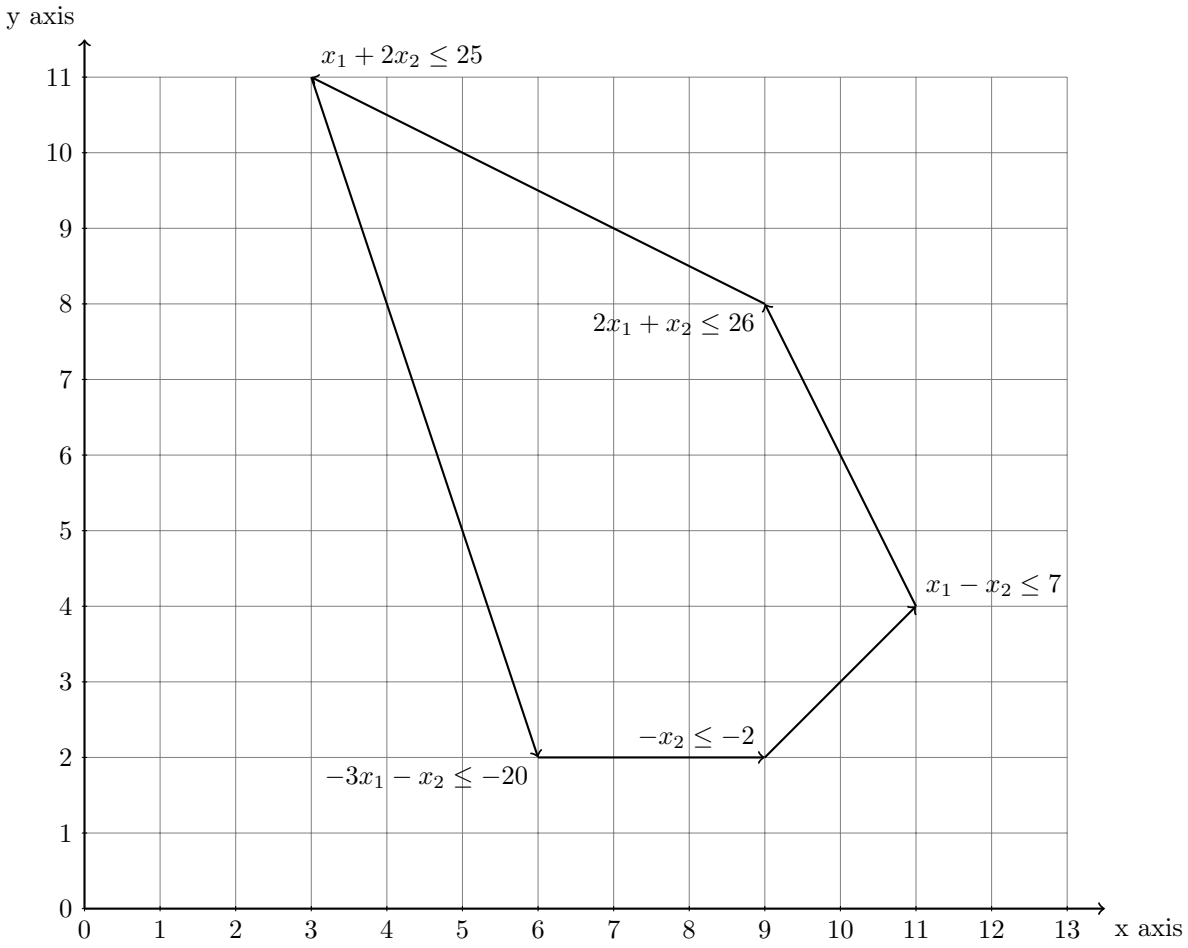
$\alpha_{ts} + \Pi_t - \Pi_s \geq 1$
$\forall ij : \alpha_{ij} + \Pi_i - \Pi_j \geq 0$

$\alpha_{ts} = 0$ by complementary slackness. $\Pi_t \geq \Pi_s + 1 = 1$.

1. The dual of the max s-t flow ILP is solvable as (0,1)-ILP.
2. Strong integer duality $\rightarrow$ primal = max flow, dual = min cut. Since the incident matrix is unimodular, these are the same.

2017-02-01

Recap on Integer Polytopes



We want to solve:

$$\begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 1 & -1 \\ 0 & -1 \\ -3 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 25 \\ 26 \\ 7 \\ -2 \\ -20 \end{bmatrix}$$

ILP Solution to 3-SAT

We want to know if there exist an assignment of x s.t. the 3-SAT is satisfied. We also want to do this using ILP.

We have the example: $(x_1 \vee \neg x_3 \vee x_4) \wedge (x_2 \vee x_3 \vee x_5) \wedge (\neg x_1 \vee \neg x_3 \vee \neg x_4)$

$$\begin{aligned} max \quad & x \\ s.t \quad & x_i \in \mathbb{Z} \\ & x_i \geq 0 \\ & x_i \leq 1 \\ & x_1 + (1 - x_3) + x_4 \geq 1 \\ & x_1 - x_3 + x_4 \geq 0 \\ & . \\ & . \end{aligned}$$

Matching

We have two different kinds of mathings – maximum and maximal. A maximum matching is "the largest number of edges that meets the constraint". A maximal matching is "a matching s.t. adding another edge is impossible". A maximum match is a maximum maximal match. We can check a graph's bi-partiteness using matching.

We can solve the maximum matching problem using an algorithm for network flow – using ILP.

ILP for Maximum Matching

note that $c = 1$

$$
\begin{array}{lll}
max & c^\intercal x & x_{ij}\, edge\, i-j \\
s.t & \forall i: \sum_{j \in \delta(i)} x_{ij} \leq 1 & \\
 & \forall j: \sum_{i \in \delta(j)} x_{ij} \leq 1 & \\
 & x_{ij} \geq 0 &
\end{array}
$$

Let A be a matrix with nods $\times$ edges.
Dual:

$$
\begin{array}{ll}
min & y \\
s.t & A^\intercal y \geq 1_i
\end{array}
$$

The dual of the maximum matching problem is the vertex cover problem. MMP $\in \mathcal{P}$, but VC $\in \mathcal{NPH}$!

2017-02-20

Königs Theorem

In the mathematical area of graph theory, König's theorem describes an equivalence between the maximum matching problem and the minimum vertex cover problem in bipartite graphs.

The maximum matching problem can be expressed as:

$$\begin{aligned} max \quad & \sum y_e \\ s.t \quad & \forall v \in V : \sum_{w \in \delta(v)} y_w \leq 1 \\ & y_e \in \{0, 1\} \end{aligned}$$

And the feasible region (for bipartite graphs) (i.e. the polytope) is integer.

Vertex Cover

$$\begin{aligned} min \quad & \sum_v x_v \\ s.t \quad & \forall e = (v, w) : x_v + x_w \geq 1 \end{aligned}$$

We know that we can solve the maximum matching by its LP-relaxation. For bipartite graphs, the equivalence between vertex cover and maximum matching described by König's theorem allows the bipartite vertex cover problem to be solved in polynomial time.

In bipartite graphs the size of the maximum matching equals the size of the minimum vertex cover, by strong LP-duality.

If we're not using bipartite graphs it can get a bit nastier. This is because the incidence matrix is not TUM. In general graphs, Vertex Cover is $\mathcal{NP} - hard$.

Heuristics

Naïve Way

Given a graph we take the node with the highest degree and choose it. We procede until we cover all edges.

$cost(v) := \sum_{w \in \delta(v)} \frac{1}{deg(v)}$

We know that we started of with m edges, for each iteration we know that we have at least $m-j+1$ edges left Where j is the edge covered in the kth iteration. That can be covered using $\leq |opt|$ nodes. In the pidgeon lemma, we have $m-j+1$ pidgeons left. There always exist an edge $\exists v \, s.t. \, d(v) \geq ceil(\frac{m-j+1}{|opt|}) \geq \frac{m-j+1}{|opt|}$. This implies that at iteration k the cost of edge e_j is $\leq \frac{|opt|}{m-j+1}$.

Hence, the total cost: $\sum_e cost(e) \leq |opt| \sum_e \frac{1}{m-e+1} = H_m |opt| = \sum_{i=1}^{m} \frac{1}{i} |opt|$.
$H_m \in \Theta(log\, m) \in \mathcal{O}(log(n^2)) = \mathcal{O}(2log(n)) = \mathcal{O}(log(n))$

Dual Way

Given a linear program primal that is a minimization problem we get a dual lp that is a maximization problem.

If our min problem has a optimum solution x we know that the optimal solution for the dual is also x. However, looking at the $\mathbb{R}$-line the dual's feasible solutions are $< x$ and for the primal the feasible solutions are $> x$.

The primal ILP opt solution will be a solution "a little bit worse" than x. Since we want to minimize, it will be a solution larger than x. Analogly the ILP opt dual is a solution less than x.

In the Vertex Cover

Vertex cover is an ILP. We have some opt solution $x*$. If we want a lower bound for $x*$ we can take any feasible solution to the dual. This is the case for any minimization problem!

This leads us to an unintutive, but good, aproximation heuristics!

By creating *any* matching, we have a lower bound for $x*$. We pick an arbitrary edge e, the vertices that are connected by e (u and v) are added to the solution set. We remove all edges incident to u and v and do the same thing repeat until we have a solution to the matching.

We maintained feasability in the dual, but not in the primal. However, the *final* solution is feasible in the primal!

Primal Dual Method

1. Start Algorithm

2. Acquire a feasible dual solution y
3. Find primal solution x that minimizes violation of complementary slackness
4. Is complementary slackness satisfied for all constraints?
   (a) If yes: we have an opt LP solution.
   (b) If no: Find a new feasible dual solution y, with better objective value. We then go back to Finding a new x.... (3)

2017-02-22

Original Primal-Dual Method for Linear Programs

Starting with a feasible dual solution y, we take this y and attempt at finding a feasible primal solution x that satisfies complementary slackness. If we find such a solution we know that y and x are optimal! If we do not have complementary slackness, we change y in order to improve the dual objective.

Recap of Complementary Slackness:

Strong duality states that if x^* and y^* are optimal solutions to an $\mathcal{LP}$. Then it is the case that: $c^\intercal x^* = y^* A x^* = b^\intercal y^*$.

From this it follows that we have complementary slackness if:

1. For the primal complementary slackness: $x_j > 0 \rightarrow \sum_i a_{ij} y_i = c_j$
2. For the dual complemantary slackness: $y_i > 0 \rightarrow \sum_j a_{ij} x_j = b$

Setup

Given a primal and a dual:

$$\begin{array}{lll} min & c^\intercal x & \\ s.t & Ax \geq b & \\ & x \geq 0 & \end{array} \qquad\qquad \begin{array}{lll} max & b^\intercal y \\ s.t & A^\intercal y \leq c \\ & y \geq 0 \end{array}$$

We define the sets $\mathcal{I} := \{i \,|\, y_i = 0\}$, containing all the tight dual variables, and $\mathcal{J} := \{j \,|\, \sum_i a_{ij} y_{ij} = c_j\}$, containing all the binding dual constraints.

What we do next?

We have no gotten ourselves a new LP.

$$
\begin{aligned}
min \quad & \sum_{i \in \mathcal{I}^{\mathcal{C}}} s_i + \sum_{j \in \mathcal{J}^{\mathcal{C}}} x_j \\
s.t \quad & x \geq 0 \\
& s \geq 0 \\
& i \in \mathcal{I} : \sum_j a_{ij} x_j \geq b_i \\
& i \in \mathcal{I}^{\mathcal{C}} : \sum_j a_{ij} x_j - s_i = b_i
\end{aligned}
$$

Note that $\mathcal{I}^{\mathcal{C}}$ is the complement of $\mathcal{I}$.

If the solution to the above problem is 0, we have found a y such that complementary slackness is fullfilled!

Why do we even?!

We have gone from solving the primal to potentially solving a bunch of linear programs. We have however gotten rid of the original c in the primal. The original problem is a weighted LP, but we have replaced it by an LP that does not have a cost associated with it.

Restricted Dual

Given the LP above, we have the primal variables x and s. We know that both x and $s \geq 0$. x is split into two sets ($\mathcal{J}$ and $\mathcal{J}^{\mathcal{C}}$). The cost vector therefore becomes $c = (0 \ldots 0, 1 \ldots 1, 1 \ldots 1)$. Due to complementary slackness for $\mathcal{J}$. Due to the sets $\mathcal{I}$ and $\mathcal{I}^{\mathcal{C}}$ we also split the b vector into being equal or "greater than, equal". The $\mathcal{I}$s and xs makes out the A matrix. The part of s corresponding to $\geq b$ is 0, but the part corresponding to $= b$ is the negative identity matrix (E, for einz). The upper left part gives us: $0 \geq y$, the lower left gives us a free y which alligns with $\mathcal{I}$ and $\mathcal{I}^{\mathcal{C}}$.

The Revival – Primal-Dual for approximation

- Enforce the primal complementary slackness
- Relax the dual complementary slackness

What does it mean to relax complementary slackness?

Relaxed $\mathcal{PCS}$: $x_j > 0 \rightarrow \sum_j a_{ij} y_i \geq \frac{c_j}{\alpha}$, we are binding $\frac{c_j}{\alpha} \leq \sum a_{ij} y_i \leq c_j$.

Relaxed $\mathcal{DCS}$: $y_i > 0 \rightarrow \sum_j a_{ij} x_j \leq \beta b_i$

$\alpha, \beta \geq 1$

We now get the bound: $b^\intercal y \leq c^\intercal x \leq \alpha\beta b^\intercal y$, where $\alpha\beta$ is the approximation factor.

Back to the Vertex Cover

Given the node-edge incidence matrix A (columns = nodes, rows = edges) we must have at least one 1 per row, and atmost one 1 per column.

$$\begin{aligned} min \quad & \sum_v x_v \\ s.t \quad & x_v + x_w \geq 1 \quad (v,w) \in E \end{aligned}$$

We also have the dual:

$$\begin{aligned} max \quad & \sum y_e \\ s.t \quad & \sum y_e \leq 1 \end{aligned}$$

We know want to enforce the $\mathcal{PCS}$ by enforcing $\sum y_e = 1$ in the dual. We then have our $\beta = 2 = x_v + x_w \geq 1$.

2017-02-27

Branch and Bound

We have the following setting:

- The optimization over the feasible region Ω is hard.
- The bounds on f^* are easy.

Our approach is to split Ω into two subsets: Ω_1 and Ω_2. We know have four bounds instead of two (l_{Ω_1}, u_{Ω_1} and the same for Ω_2). We can check which bound yields the best result. We compare $u_{\Omega_1}, l_{\Omega_2}$. If $u_{\Omega_1} < l_{\Omega_2}$. We can throw away Ω_2 since even the lowest value in Ω_2 was worse than the largest one in Ω_1. If $u_{\Omega_1} \nleq l_{\Omega_2}$we cannot really say anything, hence we must continue searching.

Algorithm for minimization using BB

- We start with some feasible solution x^*
- f^* is set to ∞
- f_u^* is also set to the value given by $f(x^*)$
- $\mathcal{A}$ =$\{\Omega\}$
- While $|\mathcal{A}| > 0$
  - $\Omega = pop(A)$
  - (f_u, x_u)= $upperBound(\Omega)$
  - $(f_l, x_l) = lowerBound(\Omega)$
  - $f_u^* = min(f_u^*, f_u)$
  - if $f_l \leq f_u^*$
    - if x_l is feasible and $f_l < f^*$
      - $f^* = f_l$
      - $x^* = x_l$
    - else
      - partition Ω into Ω_1, Ω_2, ...
      - add Ω_i to A
- Return (f^*, x^*)

Bounds for minimizing ILP

Upper Bound:
A primal feasible x for the ILP

Lower Bonds:
The x^* from the LP-relaxation
A dual feasible x (LP or ILP)

Travelling Salesman Problem

Given a graph $G = (V, E)$ where all $e \in E$ have nonnegative cost c_e, we want to find the least costly Hamiltonian Path (round tour which visits each vertex exactly ones). (Euler Path: Round tour that visits all edges exactly once.)

Euler Tour

We can find the tour (if it exists) in polynomial time.

Hamilton Cycle

The existance of $\mathcal{HP}$ in a graph is $\mathcal{NP} - hard$.

Lower Bound

We can easily find the lower bound of the problem by calculating the cost of the MST of the graph. $c(MST) \leq c(H^* \backslash e) \leq c(H^*)$.

Complete Metric TSP

If we try to solve the Metric TSP we get a little bit easier problem.

MST Heuristcs

Given a graph $G = (V, E)$, we:

1. Find an MST.
2. Start on any node and traverse the graph s.t. you walk on every edge twice. This costs $2 \times c(MST)$. This is not a hamiltonian cycle but it is a cycle.
3. Redo the MST walk. At some occasion you cannot "go back", because this would result in a double tap on a vertex. Then you go to the vertex you would have gone too "after the next vertex".

Due to the triangle in-equality (in the metric version of the problem) we know that what we just did is at most as expensive as the $MST - walk$. **It is not generally the case that the triangle in-equality holds!**

2017-03-01

Branch and Bound – Example

We start with a feasible region Ω. We can calculate a lower bound for the problem and we find it to be 2. A feasible solution tells us a upper bound is 100.

We now split Ω into two parts: Ω_1, Ω_2. We get new bounds: 2–91, 30–70.

We split Ω_1 into two parts: Ω_{11}, Ω_{12} and get: 80–90, 40–60. Our best upper bound this far is 60, so we know that the other set Ω_{11} (where the lower bound is 80) must not contain the solution! Cut it off.

We repeat this until we can no longer split Ω_i.

Euclidian TSP

Until recently the Christofieds Heuristics was the state of the art heuristics. The Christofides' algorithm follows a similar outline to the MST approach but combines the minimum spanning tree with a solution of another problem, minimum-weight perfect matching. This gives a TSP tour which is at most 1.5 times the optimal.

Algorithm

1. Find a minimum spanning tree for the problem
2. Create a matching for the problem with the set of cities of odd order.
3. Find an Eulerian tour for this graph
4. Convert to TSP using shortcuts.

This is a 3/2-approximation for Eucledian MST.

General TSP

Can we approximate the general TSP?

Unless $\mathcal{P} = \mathcal{NP}$ there exists no α-approximation to TSP for any constant α. This – likely – means that there is no polynomial way of approximate this within reasonable bounds of the TSP.

Proof

Includes Hamiltonian Cycles and Euler Tours. The former is $\mathcal{NPH}$ even in the decision problem and the latter is $\mathcal{P}$.

Given an unweighted graph $G = (V, E)$, we create another graph $G' = (V', E')$ with the following properties:

- if some edge $e \in E \rightarrow c(e') \in E' = 1$
- if some edge $e \notin E \rightarrow$ add an edge e' to E' and let $c(e') = \alpha n$, where α is some constant and $n = |V|$.

If we say that we have an algorithm that can give us a hamiltonian cycle of cost $< \alpha n$ in polynomial time for any (weighted) graph, we have two scenarios we need to look at:

1. If our original graph contain a hamiltonian cycle the weight of that cycle will be n, since we have one edge per node. This means that in the second graph we would not have to pick any αn edge.
2. We now have a weighted problem. If we have no hamiltonian cycle in G we must pick at least one αn edge in G'. Since at least one edge in our solution costs αn we could not live up to what we claimed.

ILP for TSP

We need to ensure that:

- All edges are connected. (The resulting graph is connected.)
- Each vertex in the original graph is in the solution. (Each vertex is visited at least once.)
- Each vertex is visited at most one (apart from the first vertex).
- The first vertex is also the last vertex.

This sums up to:

- Degree constraint:
  - $deg(v_i) = 2$
- Sub-tour elimination:
  - avoid "too short trips"

$$
\begin{aligned}
&min && c^\intercal x \\
&s.t && \forall v : \sum_{e \in \delta(v)} x_e = 2 \\
& && x \in \mathbb{B}
\end{aligned}
$$

NOT DONE!! See next week's notes.

Ω is "all the round trips" aka. all hamiltonian cycles.

2017-03-06

Recap of last weeks min

We have a minimisation problem from last class:

$$\begin{aligned} min \quad & c^\intercal x \\ s.t \quad & \forall v \in V : \sum_{e \in \delta(v)} x_e = 2 \\ & x \in \mathbb{B} \end{aligned}$$

This is missing a way to eliminate cycles of node sets that are not the enteire set. We introduce a subtour elimination constraint stating that:

Inner SEC: $\forall\big(\emptyset \subset \mathcal{S} \subset V\big) : \sum_{e=(u,w)\, u,w \in \mathcal{S}} x_e \leq |\mathcal{S}| - 1$

Outer SEC: $\emptyset \subset \mathcal{S} \subset V : \sum_{e \in \delta(\mathcal{S})\, v \in \mathcal{S}} x_e \geq 2$

Lagrange Relaxation

$$\begin{aligned} min \quad & c^\intercal x \\ s.t \quad & Ax \leq b \end{aligned}$$

The constraint above is violated if $Ax - b > 0$. If we add that term to the objective function, we can create:

$$\begin{aligned} min \quad & c^\intercal x + \lambda^\intercal (Ax - b) \\ s.t \quad & \lambda > 0 \end{aligned}$$

These things are called "lagrangian multipliers".

1-Tree bound

$? \leq c(H^*) = c(e) + c(f) + c(P)$

where $c(e) + c(f) = A$ and $c(P) = B$

$A = 2$ cheapest edges incident to v_1.
$B =$ MST on $V \backslash \{v_1\}$

Held-Karp Bound

One of the problems with the MST is that we are not constrained to the nodes having degree 2. The idea with HKB is that we change the weights s.t. the tree is closer to being a path.

Objective: Change all cycle costs by the same constant amount, while changing MST to something that is path-like.

Idea: Add a cost to each node.

Problem: We have a problem working with edge costs.

Solution: Add the "node cost" for each node to its incident edges instead. Meaning that: $c(e) \rightarrow \lambda_v + \lambda_w + c(e)$.

This changes the objective value, but our solution (as in what edges to walk on) will NOT change. We can use this to "punish" certain edges.

Lagrangian Relaxation

We lagrange relax all nodes but v_1: $\sum x_e = 2$

$min\ c^\intercal x - \sum_v \lambda_v (\sum_{e \in \delta(v)} x_e - 2) = min\ c^\intercal x - \sum_v \lambda_v \sum_{e \in \delta(v)} x_e + 2\sum_v \lambda_v =$
$min \sum_e (c_e - \lambda_u - \lambda_v) x_e$ whereas this is the MST on $V \backslash \{v_1\}$, aka the HKB aka the lagrangian relaxation.

Cutting Plane Method

Solve the relaxation. Cut inbetween that solution and the optimal solution. Repeat until we either get the optimal solution or a good bound.

We want to solve:

$$\begin{array}{ll} min & c^\intercal x \\ s.t & Ax \leq b \end{array} \rightarrow$$

$$\begin{array}{ll} min & c^\intercal x \\ s.t & Ax \leq \lfloor b \rfloor \end{array} \rightarrow$$

find λ s.t. $\lambda^\intercal Ax \in \mathbb{Z} \rightarrow \lambda^\intercal Ax \leq \lfloor \lambda^\intercal b \rfloor$

Our goal: Solv LP-relaxation of TSP without too many constraints.

Approach

- We start without any SECs.
- Find a min cut with a weight of less than 2. This implies that we violated an outer SEC.
  - We do this using the max flow/min cut theorem.