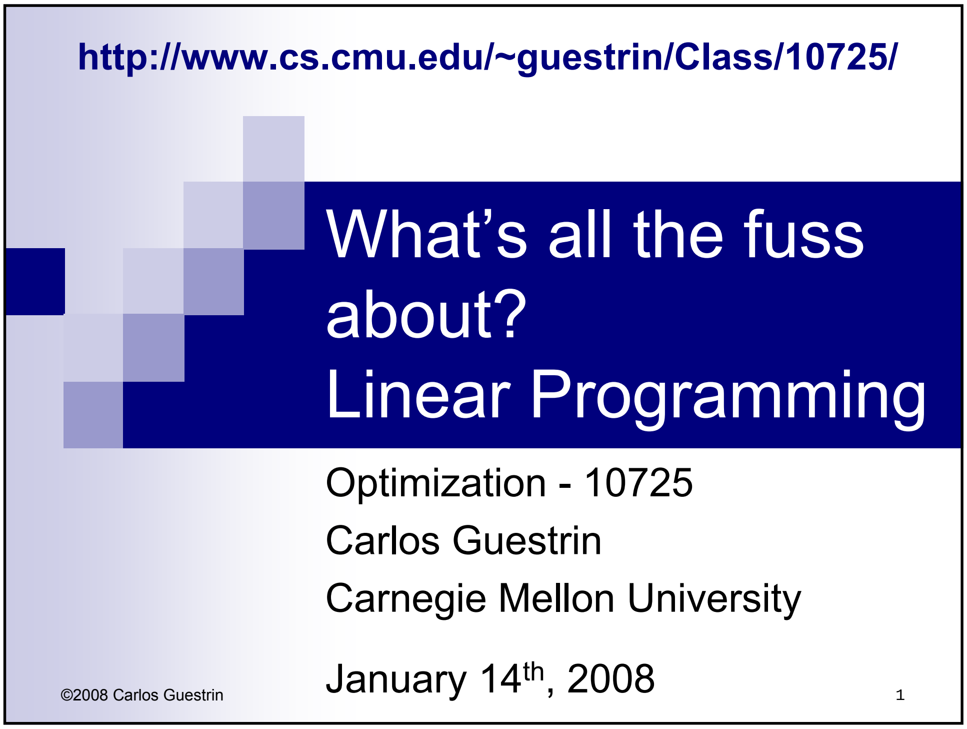


http://www.cs.cmu.edu/~guestrin/Class/10725/

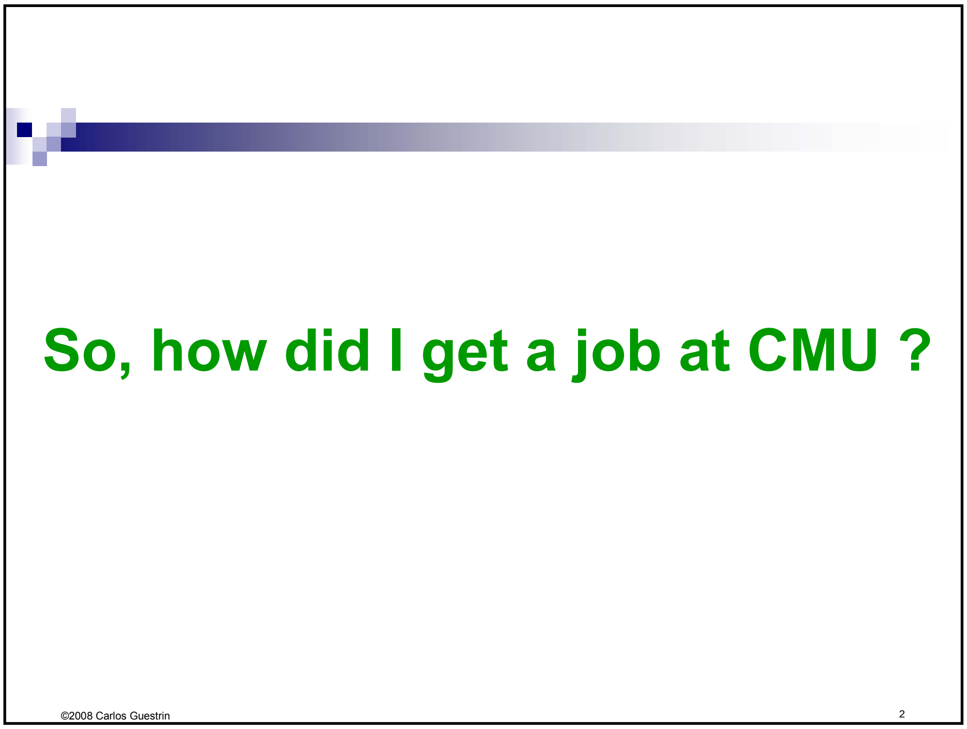
What's all the fuss about? Linear Programming

Optimization - 10725

Carlos Guestrin

Carnegie Mellon University

January 14th, 2008



So, how did I get a job at CMU ?

How? How? How? Hooooow?

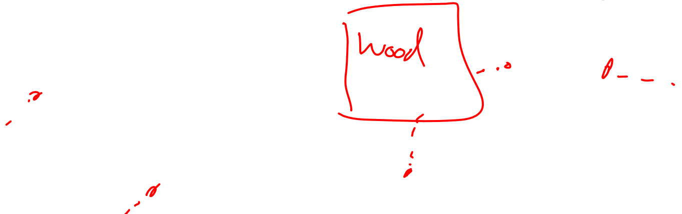
- OK, we all know it was a mistake, but how?
- Multiagent coordination…
  - □ There were many heuristics out there
    - "first you do this, than you do that, then you pray…"
  - □ Formulated an optimization problem
    - linear program
    - Unique optimum, no local minima, find it in polytime on the number of variables and constraints
  - □ Polynomial number of variables
  - □ One constraint for each state and action

how actions?

k^n

LP decomposition technique

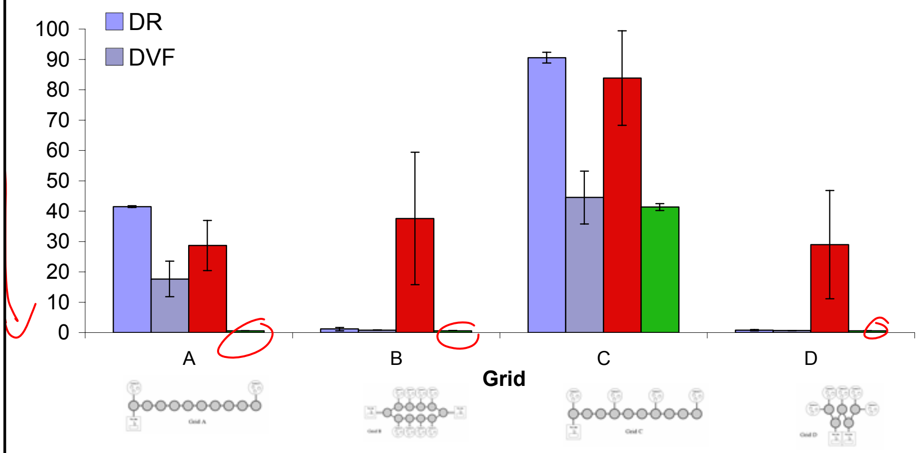
- Need to solve multiagent problems with linear program with exponentially-many constraints
- Problem has structure (similar to graphical models)



LP decomposition technique

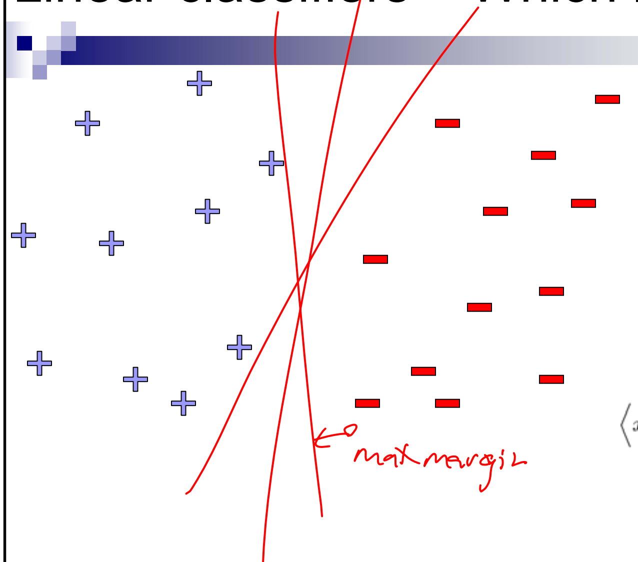
exp many constraints → equivalent poly time LP

Power Grid Problem



Lower is better!

Linear classifiers – Which line is better?



Data:

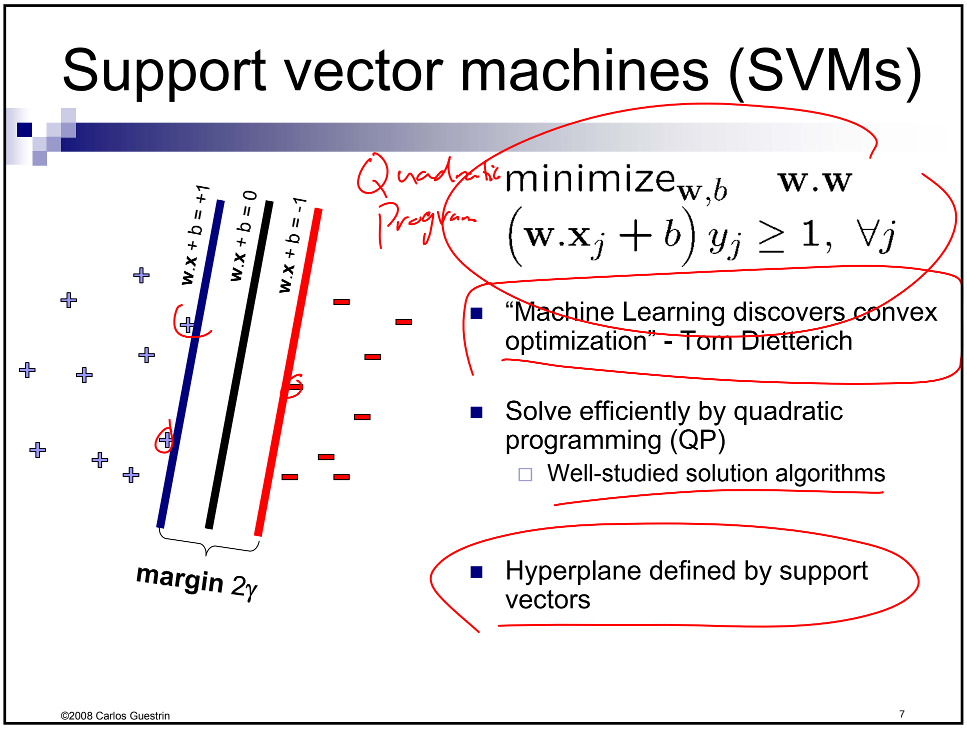
$$\left\langle x_1^{(1)}, \ldots, x_1^{(m)}, y_1 \right\rangle$$
$$\vdots$$
$$\left\langle x_n^{(1)}, \ldots, x_n^{(m)}, y_n \right\rangle$$

Example i:

$\left\langle x_i^{(1)}, \ldots, x_i^{(m)} \right\rangle$ — m features

$y_i \in \{-1, +1\}$ — class

w.x = $\sum_j w^{(j)} x^{(j)}$



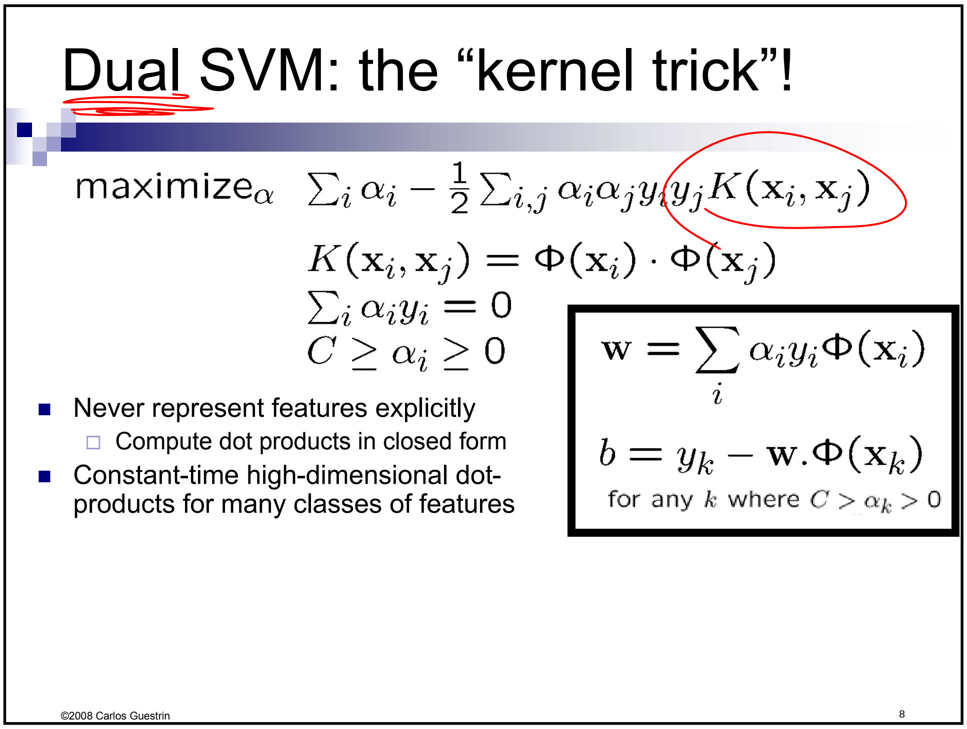
Support vector machines (SVMs)

Quadratic Program

$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w}$$

$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \ \forall j$$

- "Machine Learning discovers convex optimization" - Tom Dietterich
- Solve efficiently by quadratic programming (QP)
  - Well-studied solution algorithms
- Hyperplane defined by support vectors



Dual SVM: the "kernel trick"!

$$\text{maximize}_\alpha \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

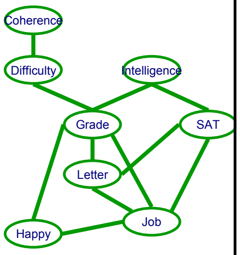
$$\sum_i \alpha_i y_i = 0$$

$$C \geq \alpha_i \geq 0$$

$$\boxed{\begin{array}{l} \mathbf{w} = \sum_i \alpha_i y_i \Phi(\mathbf{x}_i) \\ b = y_k - \mathbf{w}.\Phi(\mathbf{x}_k) \\ \text{for any } k \text{ where } C > \alpha_k > 0 \end{array}}$$

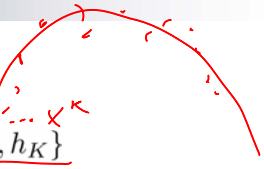
- Never represent features explicitly
  - Compute dot products in closed form
- Constant-time high-dimensional dot-products for many classes of features

Inference in graphical models



- Want to compute posterior distribution
  P(Job=t | Grade = "A")
- Very hard problem… #P-complete
- There were many heuristics…
  - □ Unstable, local optima, no guarantees…
  - □ But can we find "best approximation"?
    - E.g., lowest KL divergence?
- No, but…
  - □ Can formulate convex approximation
    - Variational methods
  - □ Stable, no local optima, often very effective
- "Graphical models discover convexity"

The regression problem



- **Instances:** <$\mathbf{x}_j$, t_j>
- **Learn:** Mapping from x to t(**x**)
- **Hypothesis space:**
  - □ Given, basis functions
  - □ Find coeffs **w**={w_1,…,w_k}

1, x, x², … x^k

$$H = \{h_1, \ldots, h_K\}$$

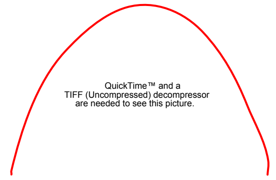
$$\underbrace{t(\mathbf{x})}_{\text{data}} \approx \widehat{f}(\mathbf{x}) = \textstyle\sum_i w_i h_i(\mathbf{x})$$

  - □ Why is this called linear regression???
    - model is linear in the parameters
- Precisely, minimize the residual squared error:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_j \left(t(\mathbf{x}_j) - \sum_i w_i h_i(\mathbf{x}_j) \right)^2$$

convex ≡ Convex Quadratic Program (no constraints)
matrix ops

Overfitting and Regulatization


QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

X^0 *	0.5978945307598738
X^1 *	2.340201179939286
X^2 *	-0.2518661564155983

X^0 *	2186.1840042933895
X^1 *	-16469.863162589896
X^2 *	52512.21531414815
X^3 *	-95140.78449247137
X^4 *	110662.42418814528
X^5 *	-88277.46566570339
X^6 *	50266.25653442256
X^7 *	-20945.151255487497
X^8 *	6480.3454987076375
X^9 *	-1498.2612106472752
X^10 *	258.5241805962563
X^11 *	-32.9723696102697
X^12 *	3.0455789461745604
X^13 *	-0.196424297260513
X^14 *	0.00829064821420466
X^15 *	-2.0200453027411646E-4
X^16 *	2.082281731309957E-6

Penalize large weights (Regularization)

L_2

$$\underset{w}{\text{argmin}} \underline{\sum_j \left(t(\mathbf{x}_j) - \sum_i w_i h_i(\mathbf{x}_j)\right)^2} + \lambda \sum_i w_i^2$$

still convex :)

still solve with matrix inversion

every basis a little mass

$\sum_i w_i^2$

w	opt 1	option 2
w_1	0.5	0
w_2	0.5	0
w_3	0.75	1
w_4	0.75	1

better L_2

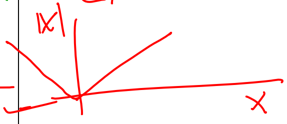
option 2 better for L_1

L_1 LASSO

$$\sum_j \left(t(\mathbf{x}_j) - \sum_i w_i h_i(\mathbf{x}_j)\right)^2 + \underline{\underline{\lambda \sum_i |w_i|}}$$

favors sparse solutions

L_1 not smoooooooth



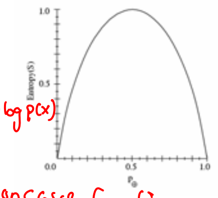
not differentiable at ∅

it is convex

Quadratic program with constraints

Maximum Entropy Principle

- Suppose you know:
  - □ "On average 1/4 of the people smoke and have cancer"
    - Expected value of a feature: $E_x[\mathbb{1}(\text{smoke}=\text{true}, \text{Cancer}=\text{true})] = \frac{1}{4}$
  - □ More generally, given a set of features $f_1,\dots,f_n$, you know:
    $E_x[f_i(x)] = \alpha_i$



- What's a good representation for the distribution? $H(p) = -\sum_x p(x)\log p(x)$
  - □ If you have commitment problems, maximum entropy!

concave function

- Solve optimization problem: $\max_p H(p)$
  $E_x[f_i(x)] = \alpha_i \quad \forall i$

- Seems hard, but… efficient solutions

I only know maximum likelihood estimation

- Maximum entropy problem:
  $\max_p H(p)$
  $E[f_i] = \alpha_i$

- Here is a completely different problem:
  - □ Maximize data likelihood for a log-linear model
    $P(x|\theta) \propto e^{\sum_i \theta_i f_i(x)}$
    want to find MLE

- two problems are equivalent
  dual

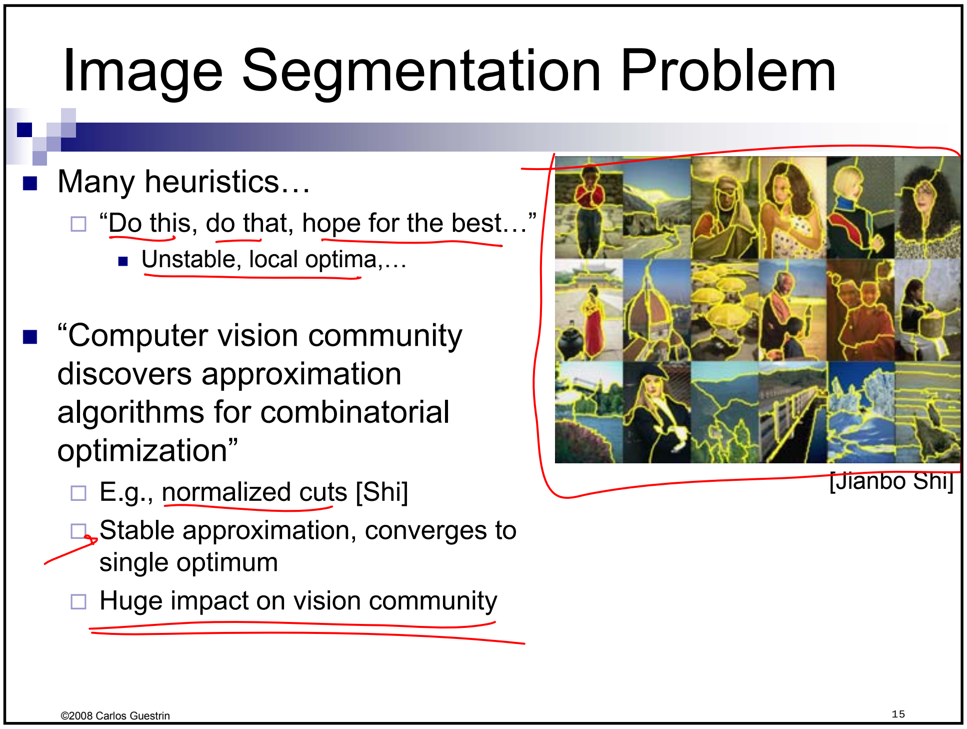
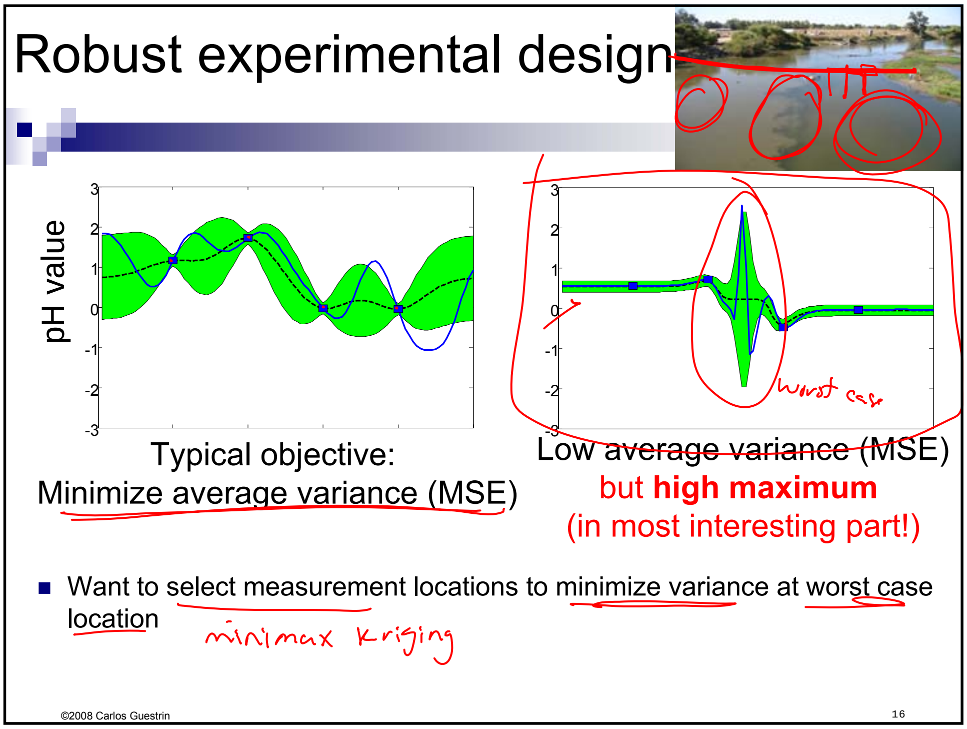


Image Segmentation Problem

- Many heuristics…
  - □ "Do this, do that, hope for the best…"
    - Unstable, local optima,…
- "Computer vision community discovers approximation algorithms for combinatorial optimization"
  - □ E.g., normalized cuts [Shi]
  - □ Stable approximation, converges to single optimum
  - □ Huge impact on vision community

[Jianbo Shi]

©2008 Carlos Guestrin



Robust experimental design

Typical objective:
Minimize average variance (MSE)

Low average variance (MSE)
but high maximum
(in most interesting part!)

- Want to select measurement locations to minimize variance at worst case location

minimax kriging

©2008 Carlos Guestrin

Is it possible to find near optimal solution?

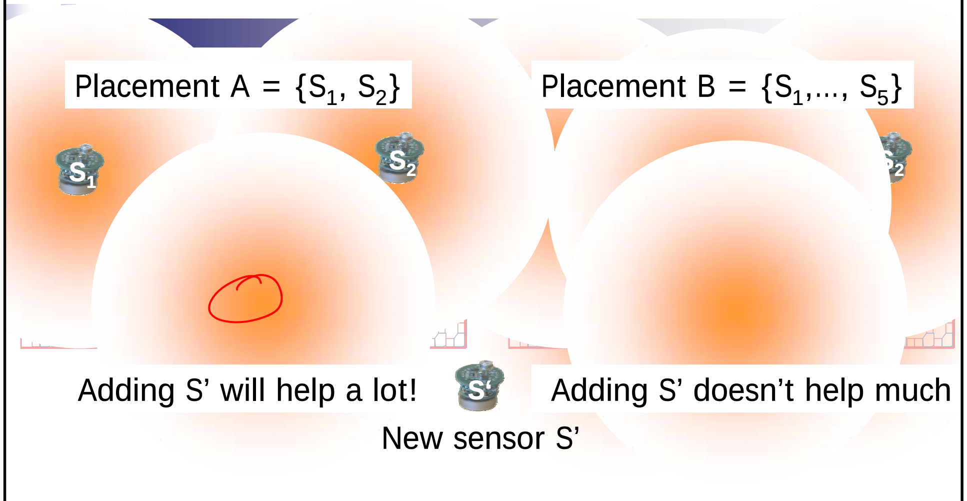
- Unless P=NP, there is no approximation algorithm

if alg finds $\hat{A}$ $F(\hat{A}) \geq \gamma(n) \cdot F(A_{opt})$

$\gamma(n)$ anything, e^{-n}

- State of the art:
  - □ Simulated annealing with 7 hand-tuned parameters

Key observation: Diminishing returns



Placement A = $\{S_1, S_2\}$

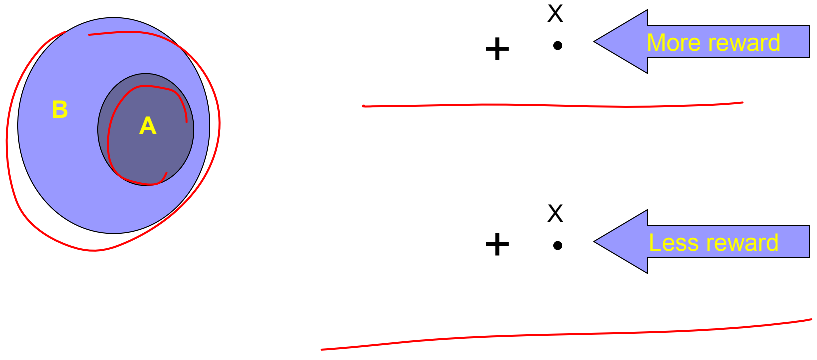
Placement B = $\{S_1, \ldots, S_5\}$

Adding S' will help a lot!

Adding S' doesn't help much

New sensor S'

Submodular Functions



$F(A \cup x) - F(A) \geq F(B \cup x) - F(B)$

Exploiting submodularity [Krause et al. '07]

- Unless P=NP, there is no polytime *approximation algorithm* for:
  - $\max_A \min_i F_i(A)$
    - **for |A|=k**

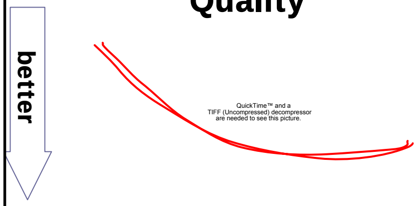
- **SATURATE** achieves optimal value with only slightly more than k sensors
  - $\min_i F_i(A_S) \geq \min_i F_i(A_{OPT})$
    - for $|A_S| \geq \alpha k$
    - $\boldsymbol{\alpha = 1 + \log(\max_{s \in V} \sum_i F_i(s))}$

relaxation

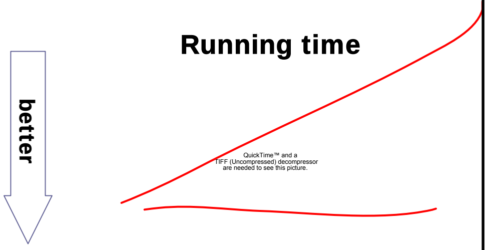
- Unless **NP ⊂ Dtime($n^{\log \log n}$)**, no polytime algorithm achieves
  - $\min_i F_i(A_S) \geq \min_i F_i(A_{OPT})$
    - for $|A_S| \geq \beta k$
    - $\boldsymbol{\beta < 1 + \log(\max_{s \in V} \sum_i F_i(s))}$

Comparison to state of the art
(minimax Kriging on sensor data)

Quality



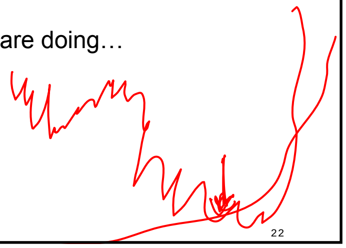
Running time

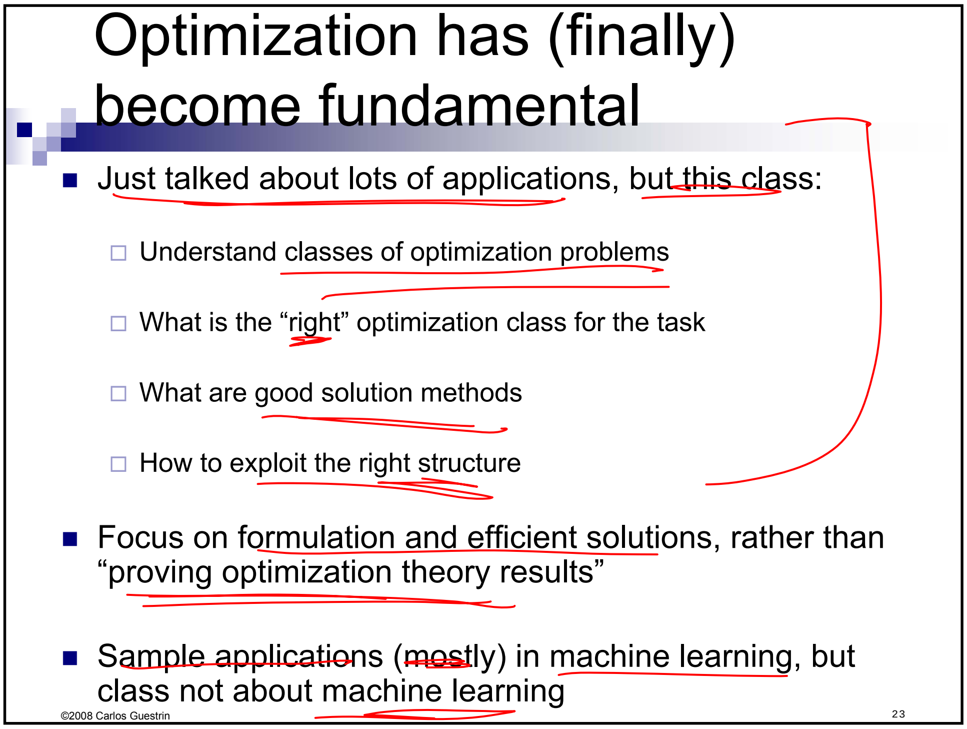


Where we are going…

- Have a complex problem
- Not too long ago:
  - Design "algorithm"
    - Not even know what is being optimized
    - Hope for the best

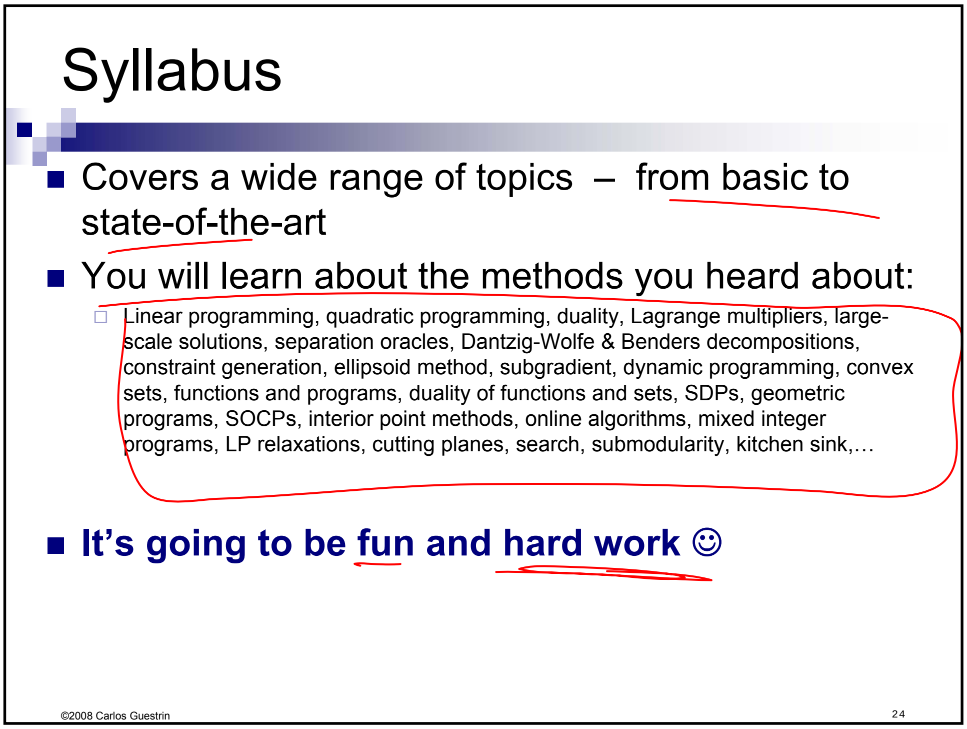
- Now: Formulate optimization problem and find efficient solution, exploiting structure
  - Convexity, locality, decomposition, submodularity,…
- But be careful, you still need to understand what you are doing…





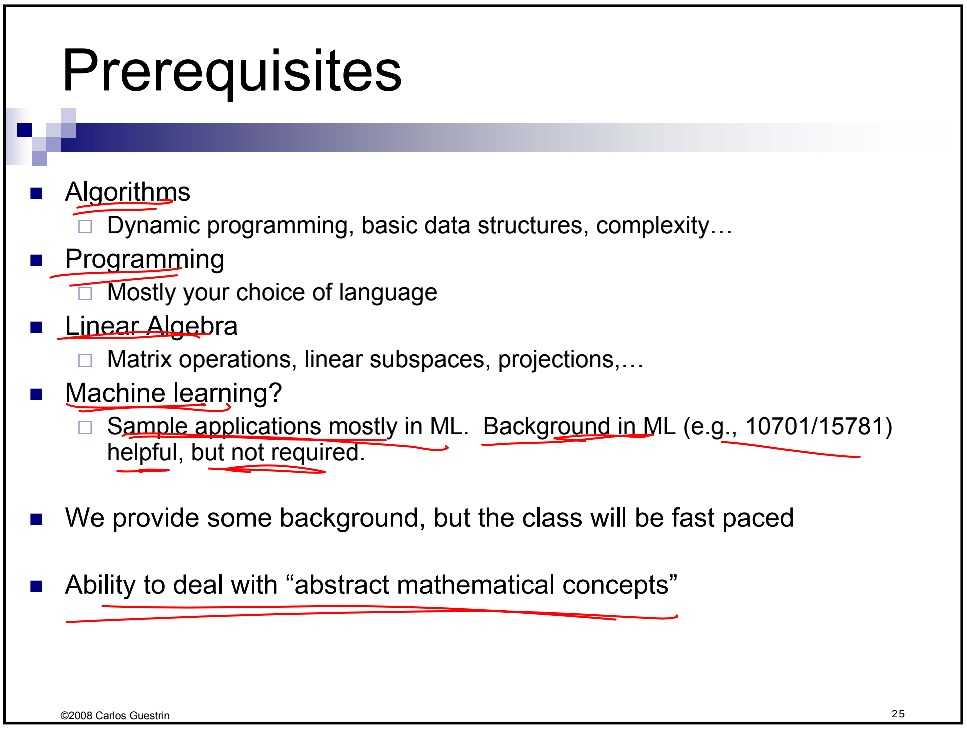
Optimization has (finally) become fundamental

- Just talked about lots of applications, but this class:
  - □ Understand classes of optimization problems
  - □ What is the "right" optimization class for the task
  - □ What are good solution methods
  - □ How to exploit the right structure
- Focus on formulation and efficient solutions, rather than "proving optimization theory results"
- Sample applications (mostly) in machine learning, but class not about machine learning



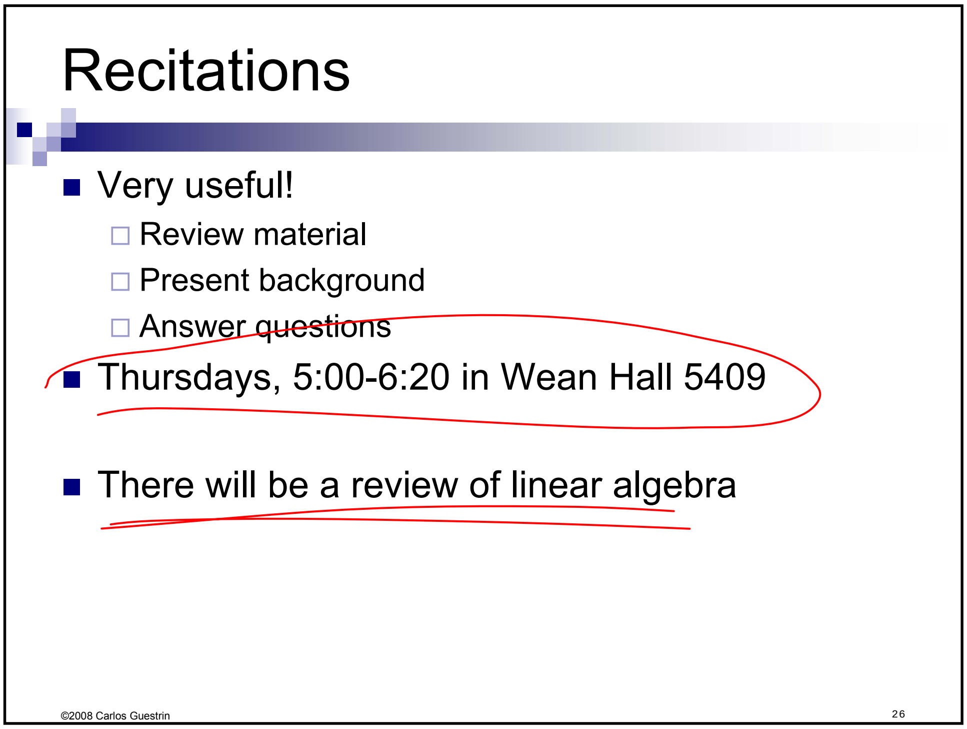
Syllabus

- Covers a wide range of topics – from basic to state-of-the-art
- You will learn about the methods you heard about:
  - □ Linear programming, quadratic programming, duality, Lagrange multipliers, large-scale solutions, separation oracles, Dantzig-Wolfe & Benders decompositions, constraint generation, ellipsoid method, subgradient, dynamic programming, convex sets, functions and programs, duality of functions and sets, SDPs, geometric programs, SOCPs, interior point methods, online algorithms, mixed integer programs, LP relaxations, cutting planes, search, submodularity, kitchen sink,…
- **It's going to be fun and hard work ☺**



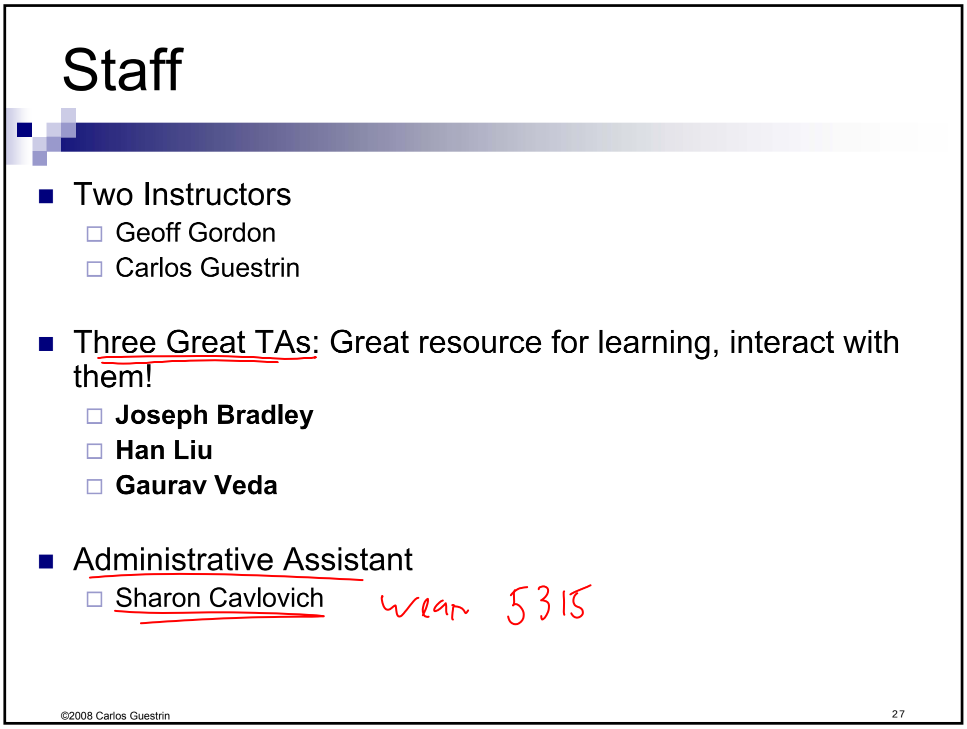
Prerequisites

- Algorithms
  - □ Dynamic programming, basic data structures, complexity…
- Programming
  - □ Mostly your choice of language
- Linear Algebra
  - □ Matrix operations, linear subspaces, projections,…
- Machine learning?
  - □ Sample applications mostly in ML. Background in ML (e.g., 10701/15781) helpful, but not required.
- We provide some background, but the class will be fast paced
- Ability to deal with “abstract mathematical concepts”



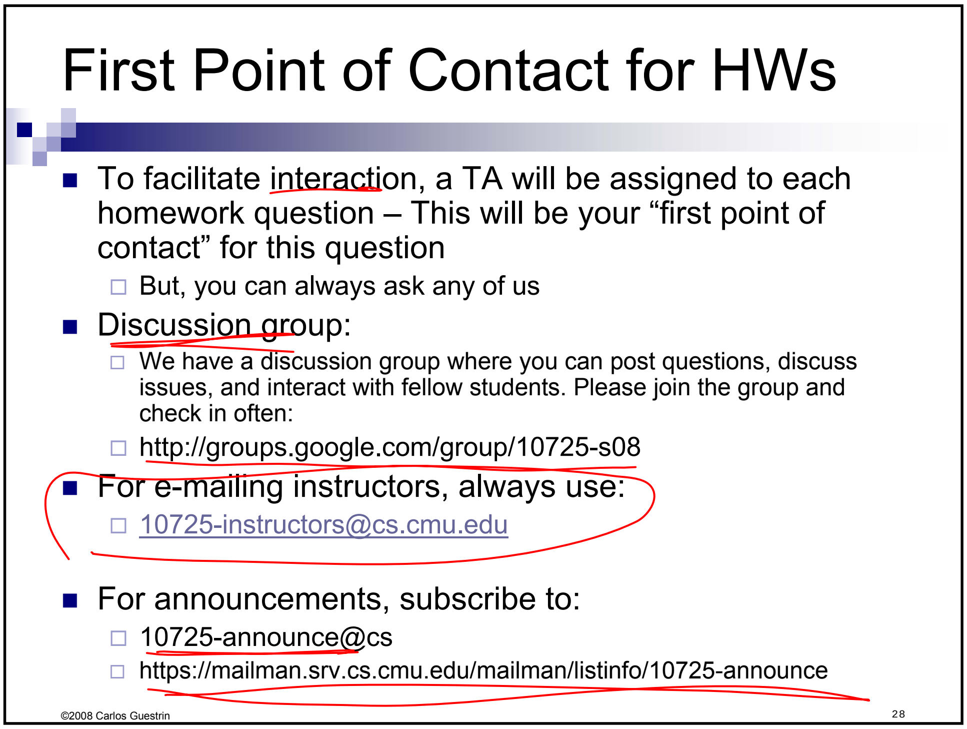
Recitations

- Very useful!
  - □ Review material
  - □ Present background
  - □ Answer questions
- Thursdays, 5:00-6:20 in Wean Hall 5409
- There will be a review of linear algebra



Staff

- Two Instructors
  - □ Geoff Gordon
  - □ Carlos Guestrin
- Three Great TAs: Great resource for learning, interact with them!
  - □ **Joseph Bradley**
  - □ **Han Liu**
  - □ **Gaurav Veda**
- Administrative Assistant
  - □ Sharon Cavlovich wean 5315



First Point of Contact for HWs

- To facilitate interaction, a TA will be assigned to each homework question – This will be your “first point of contact” for this question
  - □ But, you can always ask any of us
- Discussion group:
  - □ We have a discussion group where you can post questions, discuss issues, and interact with fellow students. Please join the group and check in often:
  - □ http://groups.google.com/group/10725-s08
- For e-mailing instructors, always use:
  - □ 10725-instructors@cs.cmu.edu
- For announcements, subscribe to:
  - □ 10725-announce@cs
  - □ https://mailman.srv.cs.cmu.edu/mailman/listinfo/10725-announce

Text Books

- Textbook:
  - □ Convex Optimization, Boyd and Vandenberghe, which is available online for free.
- Optional textbooks:

  ① Linear Optimization Bertsimas & Tsitsiklis
  - □ Combinatorial Optimization: Algorithms and Complexity, Christos Papadimitriou and Kenneth Steiglitz.
  - □ Combinatorial Optimization, Alexander Schrijver.
  - □ Nonlinear Programming, Dimitri Bertsekas.
  - □ Approximation Algorithms, Vijay Vazirani

Grading

- 5 homeworks (50%)
  - □ First one goes out 1/23
    - Start early, Start early, Start early, Start early, Start early, Start early, Start early, Start early, Start early, Start early
- Final project (30%)
  - □ Your chance to do something cool with optimization
  - □ Projects done individually, or groups of two students
- Final (20%)
  - □ Take home
    - Out: Monday, May 5 (time TBD)
    - Due: Friday, May 9 (time TBD)

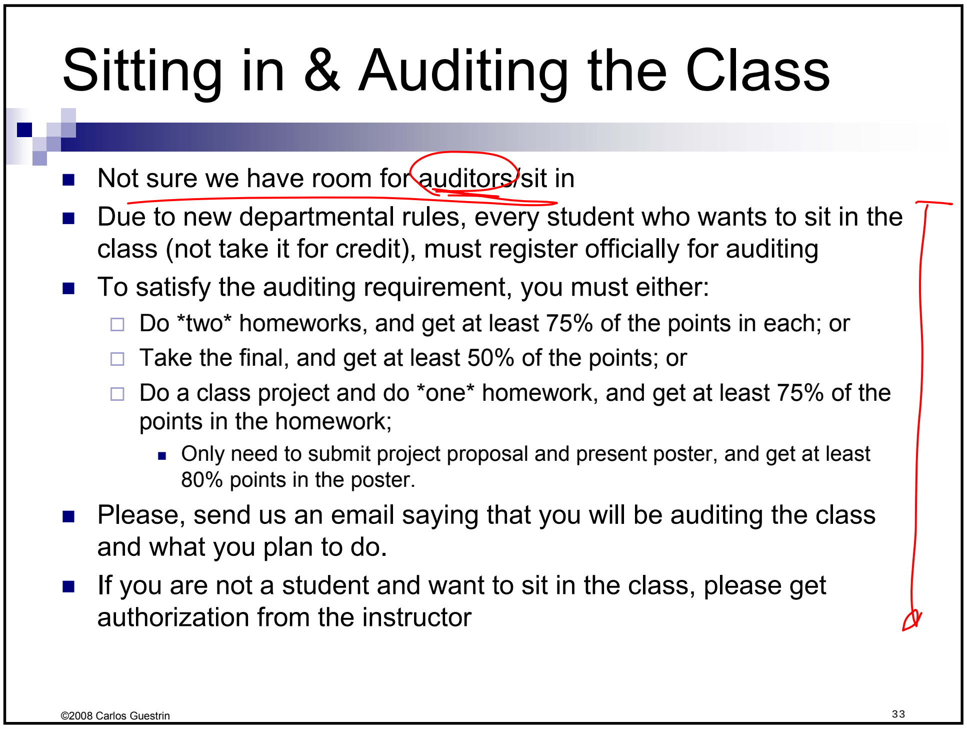
Homeworks

- Homeworks are hard, start early ☺
- Due in the beginning of class
- 3 late days for the semester
- After late days are used up:
  - □ Half credit within 48 hours
  - □ Zero credit after 48 hours
- All homeworks **must be handed in**, even for zero credit
- Late homeworks handed in to Sharon Cavlovich, WEH 5315□

- Collaboration
  - □ You may **discuss** the questions
  - □ Each student writes their own answers
  - □ Write on your homework anyone with whom you collaborate
  - □ Each student must write their own code for the programming part
  - □ **Please don't search for answers on the web, Google, other classes' homeworks, etc.**
    - please ask us if you are not sure if you can use a particular reference

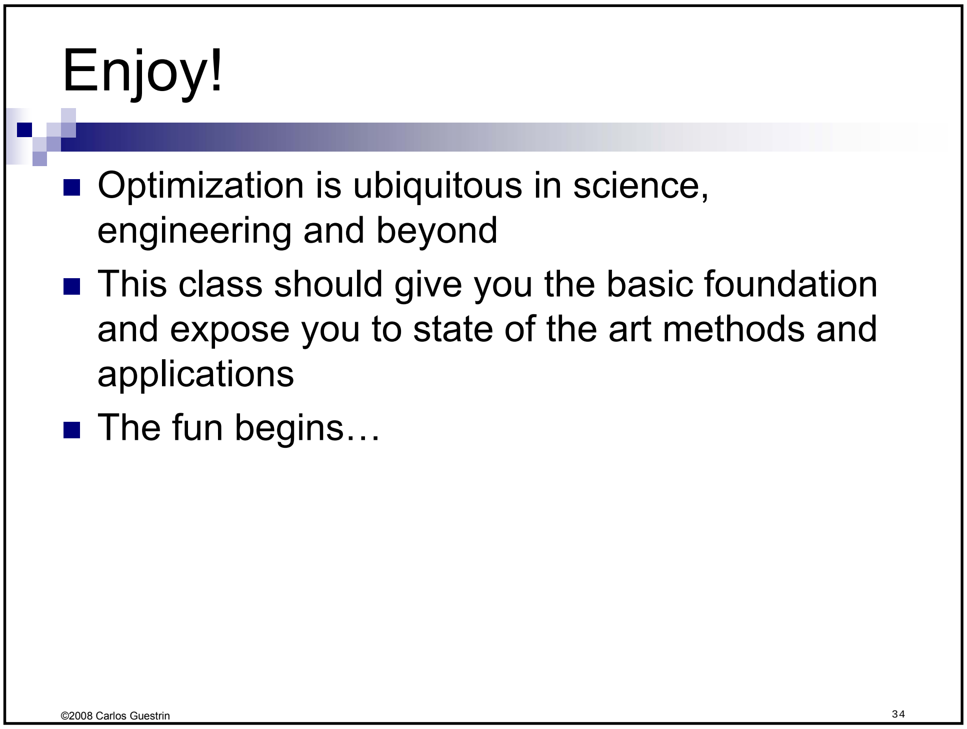
Waiting list

- If you are on the waiting list, sign list after class



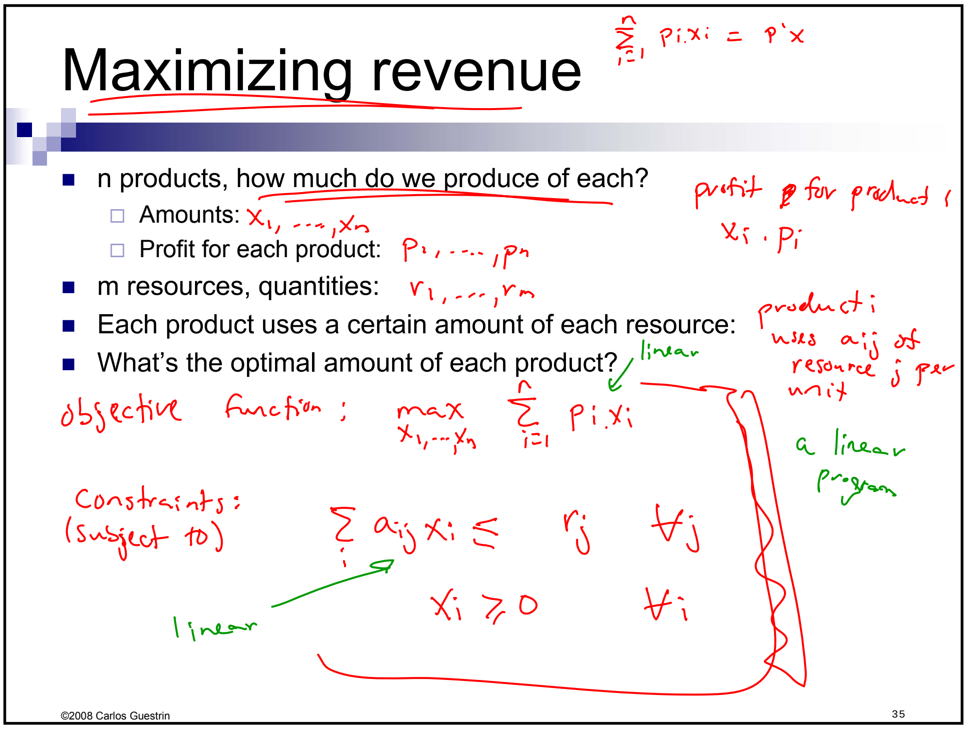
Sitting in & Auditing the Class

- Not sure we have room for auditors/sit in
- Due to new departmental rules, every student who wants to sit in the class (not take it for credit), must register officially for auditing
- To satisfy the auditing requirement, you must either:
  - □ Do *two* homeworks, and get at least 75% of the points in each; or
  - □ Take the final, and get at least 50% of the points; or
  - □ Do a class project and do *one* homework, and get at least 75% of the points in the homework;
    - Only need to submit project proposal and present poster, and get at least 80% points in the poster.
- Please, send us an email saying that you will be auditing the class and what you plan to do.
- If you are not a student and want to sit in the class, please get authorization from the instructor



Enjoy!

- Optimization is ubiquitous in science, engineering and beyond
- This class should give you the basic foundation and expose you to state of the art methods and applications
- The fun begins…



Maximizing revenue

$\sum_{i=1}^{n} p_i x_i = p'x$

- n products, how much do we produce of each?
  - □ Amounts: $x_1, \ldots, x_n$
  - □ Profit for each product: $p_1, \ldots, p_n$
- m resources, quantities: $r_1, \ldots, r_m$
- Each product uses a certain amount of each resource:
- What's the optimal amount of each product?

profit ~~p~~ for product i
$x_i \cdot p_i$

product i uses a_{ij} of resource j per unit

linear

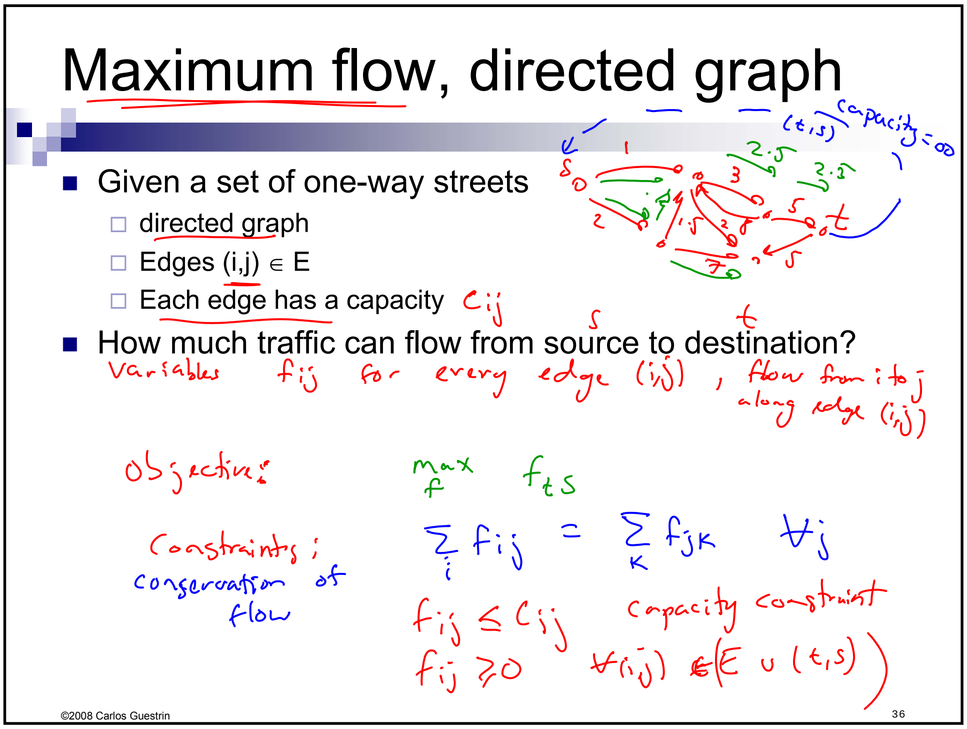
objective function: $\max_{x_1,\ldots,x_n} \sum_{i=1}^{n} p_i x_i$

Constraints: (subject to) $\sum_i a_{ij} x_i \leq r_j \quad \forall j$

$x_i \geq 0 \quad \forall i$

linear

a linear program



Maximum flow, directed graph

- Given a set of one-way streets
  - □ directed graph
  - □ Edges (i,j) ∈ E
  - □ Each edge has a capacity c_{ij}
- How much traffic can flow from source to destination?

s
t
1
2
3
2.5
2.5
1.5
2
5
5
7
(t,s)
capacity = ∞
s
t

Variables f_{ij} for every edge (i,j), flow from i to j along edge (i,j)

objective: $\max_{f} f_{ts}$

Constraints; conservation of flow $\sum_i f_{ij} = \sum_k f_{jk} \quad \forall j$

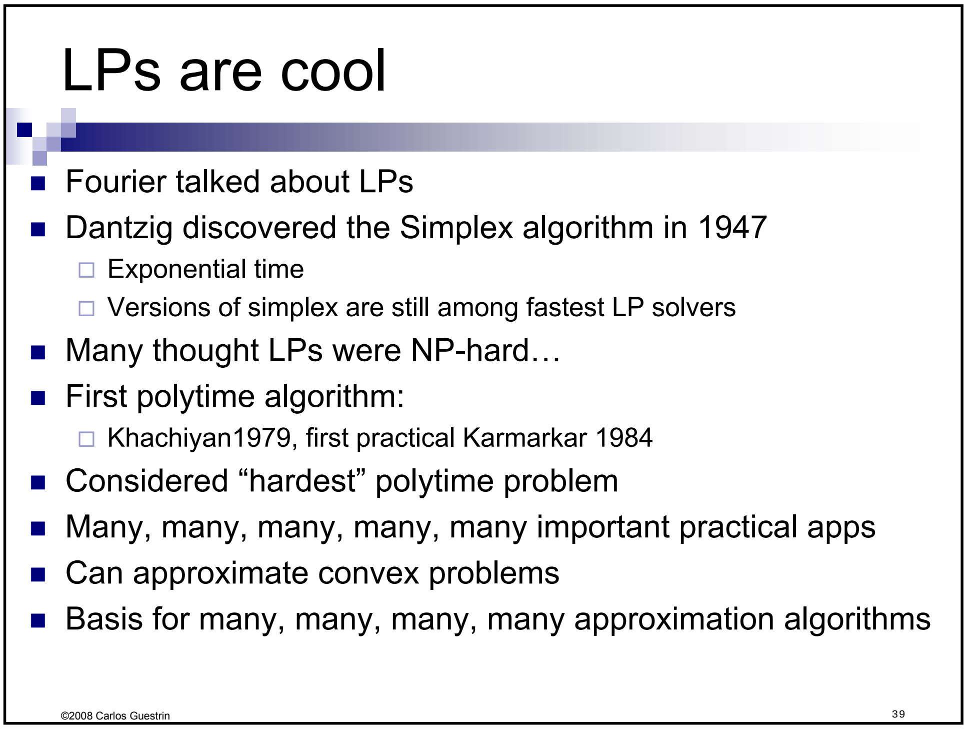
$f_{ij} \leq c_{ij}$ capacity constraint

$f_{ij} \geq 0 \quad \forall (i,j) \in (E \cup (t,s))$

L_1 Regression

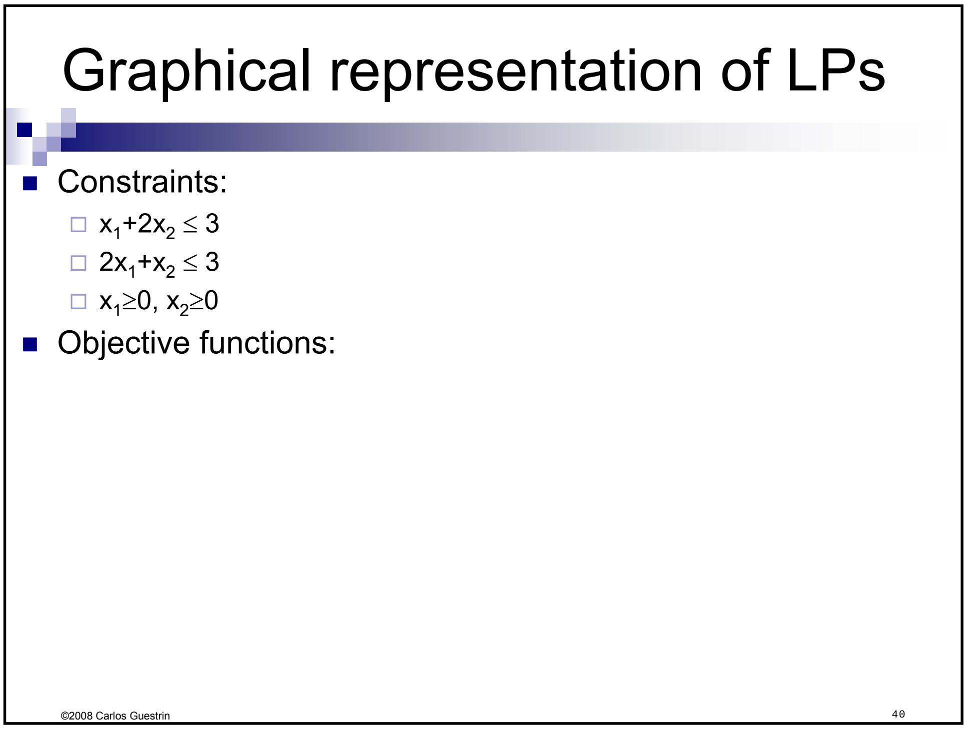
- Least-squares regression: $\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_j \left(t(\mathbf{x}_j) - \sum_i w_i h_i(\mathbf{x}_j) \right)^2$
- L_1 regression:
- Absolute values, not linear… how can this be solved???

General Form of Linear Program



LPs are cool

- Fourier talked about LPs
- Dantzig discovered the Simplex algorithm in 1947
  - □ Exponential time
  - □ Versions of simplex are still among fastest LP solvers
- Many thought LPs were NP-hard…
- First polytime algorithm:
  - □ Khachiyan1979, first practical Karmarkar 1984
- Considered "hardest" polytime problem
- Many, many, many, many, many important practical apps
- Can approximate convex problems
- Basis for many, many, many, many approximation algorithms



Graphical representation of LPs

- Constraints:
  - □ $x_1+2x_2 \leq 3$
  - □ $2x_1+x_2 \leq 3$
  - □ $x_1 \geq 0$, $x_2 \geq 0$
- Objective functions: