Optimisation et apprentissage.

Alexandre d'Aspremont, *CNRS & École Polytechnique*.

Introduction

Complexity.

In the course. . .

- Randomness helps. Getting a solution with a small probability of failure is often much easier than solving the problem exactly.
- Random instances of some optimization problems are easier to solve.

Today. . .

- Focus on **convexity** and its impact on complexity.
- Convex approximations, duality.
- Applications in learning.

Introduction

In optimization.

Twenty years ago. . .

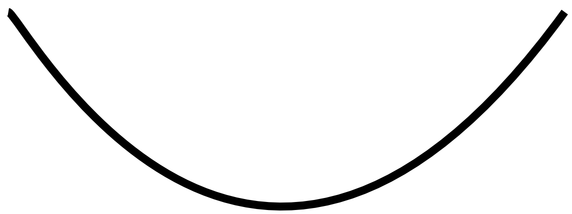
- Solve realistic large-scale problems using naive algorithms.
- Solve small, naive problems using serious algorithms.

Twenty years later. . .

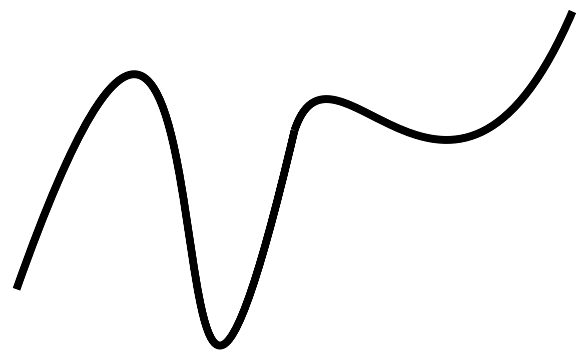
- Solve realistic problems in e.g. statistics, signal processing, using efficient algorithms with explicit complexity bounds.
- Statisticians have started to care about complexity.
- Optimizers have started to care about statistics.

Introduction

Convexity.



Convex



Not convex

Key message from **complexity theory:** as the problem dimension gets large

- all **convex** problems are easy,
- most nonconvex problems are hard.

Introduction

Convex problem.

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & a_i^T x = b_i, \quad i = 1, \ldots, p \end{array}$$

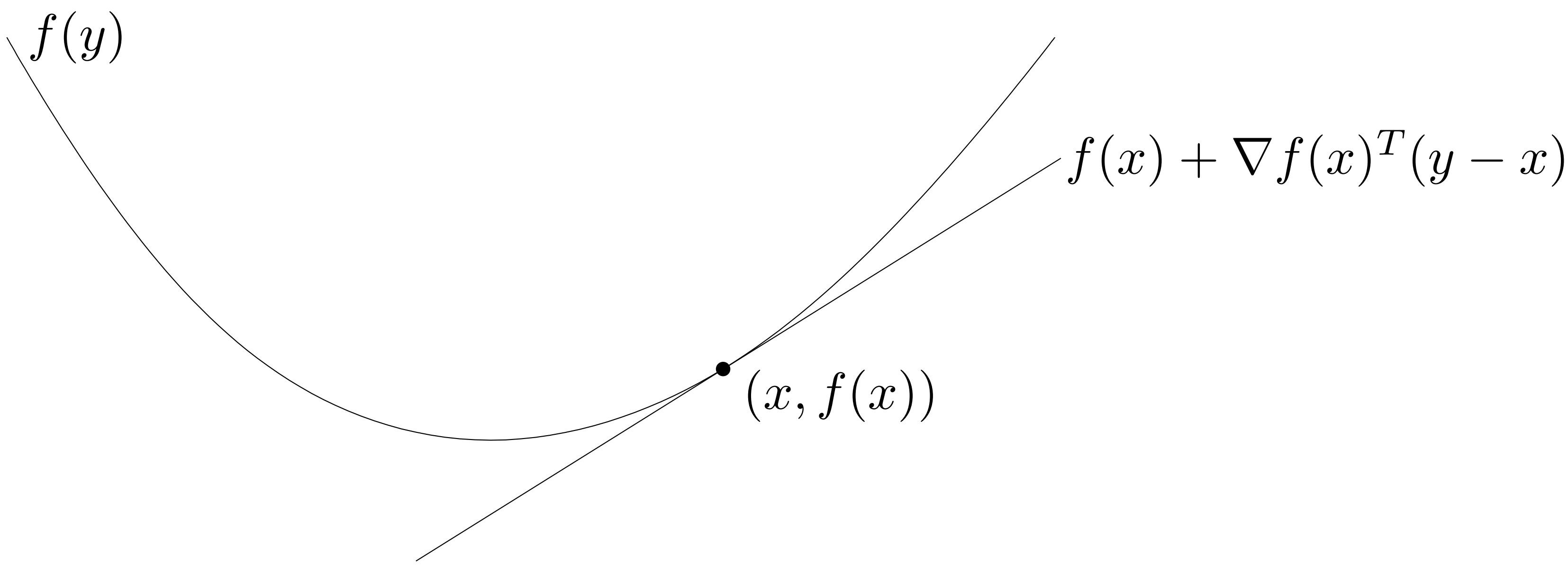
f_0, f_1, . . . , f_m are convex functions, the equality constraints are all affine.

- Strong assumption, yet **surprisingly expressive.**
- Good convex approximations of nonconvex problems.

Introduction

First-order condition. Differentiable f with convex domain is convex iff

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) \quad \text{for all } x, y \in \mathbf{dom}\, f$$

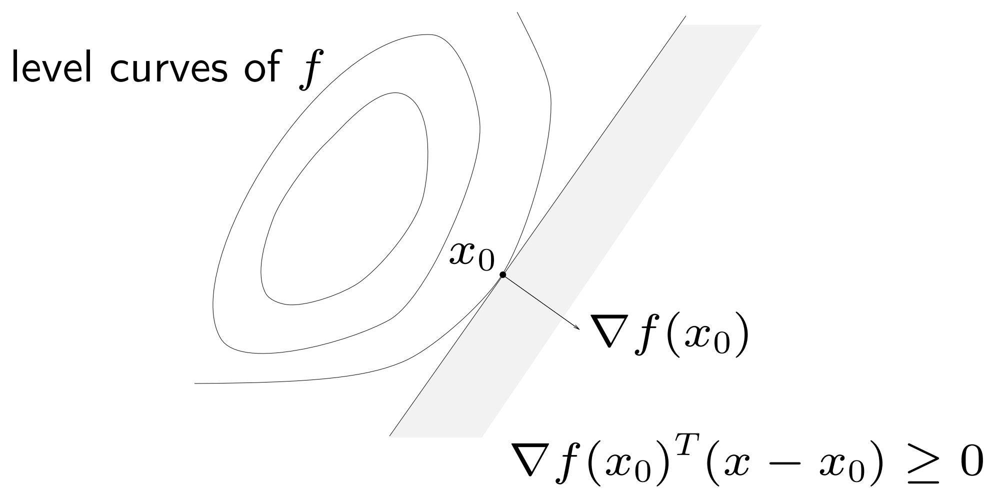


First-order approximation of f is global underestimator

Ellipsoid method

Ellipsoid method. Developed in 70s by Shor, Nemirovski and Yudin.

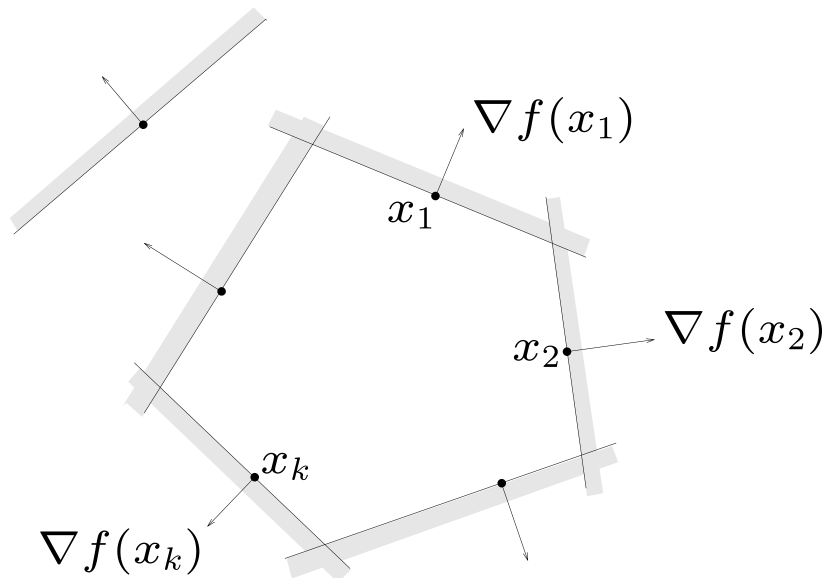
- Function $f : \mathbb{R}^n \to \mathbb{R}$ convex (and for now, differentiable)
- **problem:** minimize f
- **oracle model:** for any x we can evaluate f and $\nabla f(x)$ (at some cost)



By evaluating ∇f we rule out a halfspace in our search for $x^\star$.

Ellipsoid method

Suppose we have evaluated $\nabla f(x_1), \ldots, \nabla f(x_k)$,

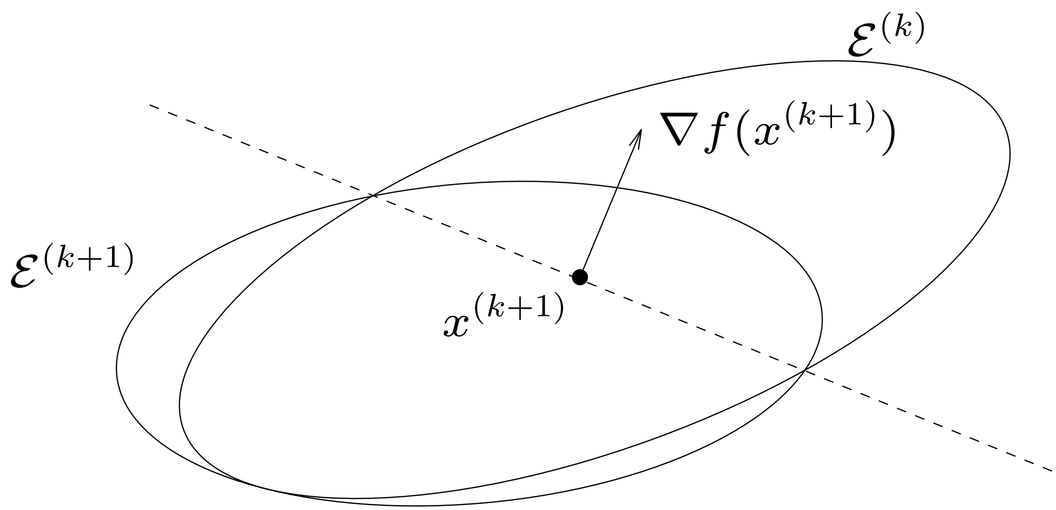


on the basis of $\nabla f(x_1), \ldots, \nabla f(x_k)$, we have **localized** $x^\star$ to a polyhedron.

Question: what is a 'good' point x_{k+1} at which to evaluate ∇f?

Ellipsoid algorithm

Idea: localize $x^\star$ in an **ellipsoid** instead of a polyhedron.



Compared to cutting-plane method:

- localization set doesn't grow more complicated
- easy to compute query point
- but, we add unnecessary points in step 4

Ellipsoid Method

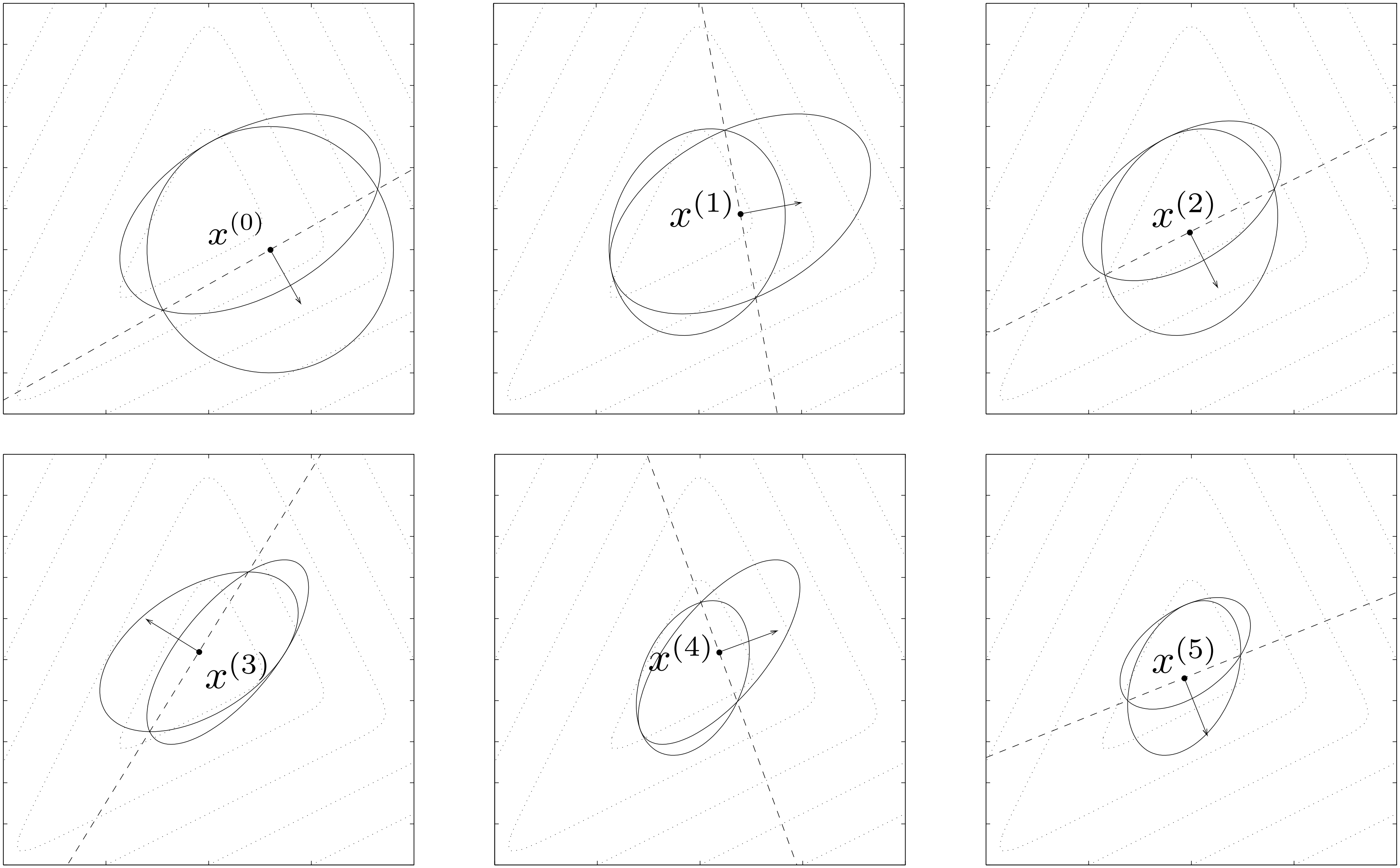
Challenges in cutting-plane methods:

- can be difficult to compute appropriate next query point
- localization polyhedron grows in complexity as algorithm progresses

Ellipsoid method:

- Simple formula for $\mathcal{E}^{(k+1)}$ given $\mathcal{E}^{(k)}$
- $\mathbf{vol}(\mathcal{E}^{(k+1)}) < e^{-\frac{1}{2n}} \mathbf{vol}(\mathcal{E}^{(k)})$

Ellipsoid Method: example



Duality

A **linear program** (LP) is written

$$\begin{array}{ll}\text{minimize} & c^T x\\ \text{subject to} & Ax = b\\ & x \geq 0\end{array}$$

where $x \geq 0$ means that the coefficients of the vector x are nonnegative.

- Starts with Dantzig's simplex algorithm in the late 40s.
- First proofs of polynomial complexity by Nemirovskii and Yudin [1979] and Khachiyan [1979] using the ellipsoid method.
- First efficient algorithm with polynomial complexity derived by Karmarkar [1984], using interior point methods.

Duality

Duality. The two linear programs

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \geq 0 \end{array} \qquad \begin{array}{ll} \text{maximize} & y^T b \\ \text{subject to} & c - A^T y \geq 0 \end{array}$$

have the same optimal values.

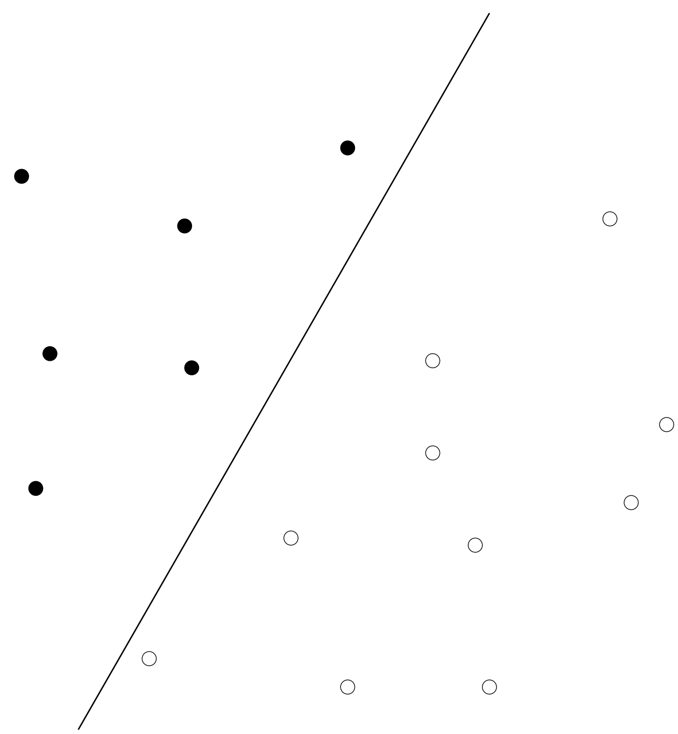
- Similar results hold for most **convex** problems.
- Usually both primal and dual have a natural interpretation.
- Many algorithms solve both problems simultaneously.

Support Vector Machines

Support Vector Machines

Simplest version. . .

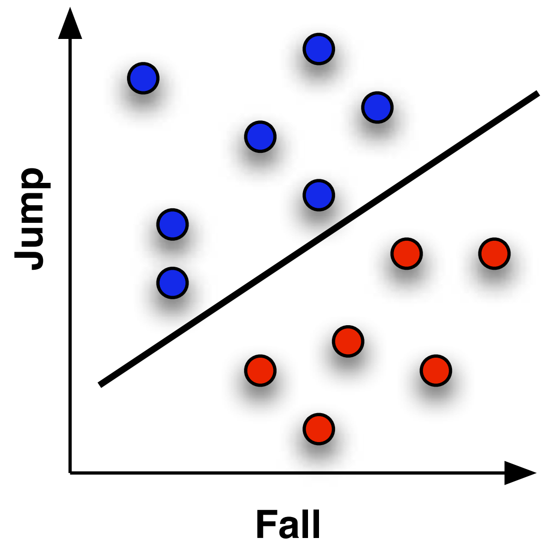
- **Input**: A set of **points** (in 2D here) and **labels** (black & white).
- **Output**: A linear classifier separating the two groups.



Text Classification

Example: word frequencies.

- In **blue**: good news
- In **red**: bad news.



Improving these results. . .

- Are we restricted to **linear** classifiers?
- What happens when the two classes are not perfectly **separable**?

Linear Classification

The **linear separation** problem.

Inputs:

- Data **points** $x_j \in \mathbb{R}^n, \quad j = 1, \ldots, m.$
- Binary **Labels** $y_j \in \{-1, 1\}, \quad j = 1, \ldots, m.$

Problem:

$$\begin{array}{ll} \text{find} & w \in \mathbb{R}^n \\ \text{such that} & \langle w, x_j \rangle \geq 1 \quad \text{for all } j \text{ such that } y_j = 1 \\ & \langle w, x_j \rangle \leq -1 \quad \text{for all } j \text{ such that } y_j = -1 \end{array}$$

Output:

- The classifier vector w.

Linear Classification

Nonlinear classification.

- The problem:

$$\begin{array}{ll} \text{find} & w \\ \text{such that} & \langle w, x_j \rangle \geq 1 \quad \text{for all } j \text{ such that } y_j = 1 \\ & \langle w, x_j \rangle \leq -1 \quad \text{for all } j \text{ such that } y_j = -1 \end{array}$$

  is linear in the variable w. Solving it amounts to solving a **linear program**.

- Suppose we want to add quadratic terms in x:

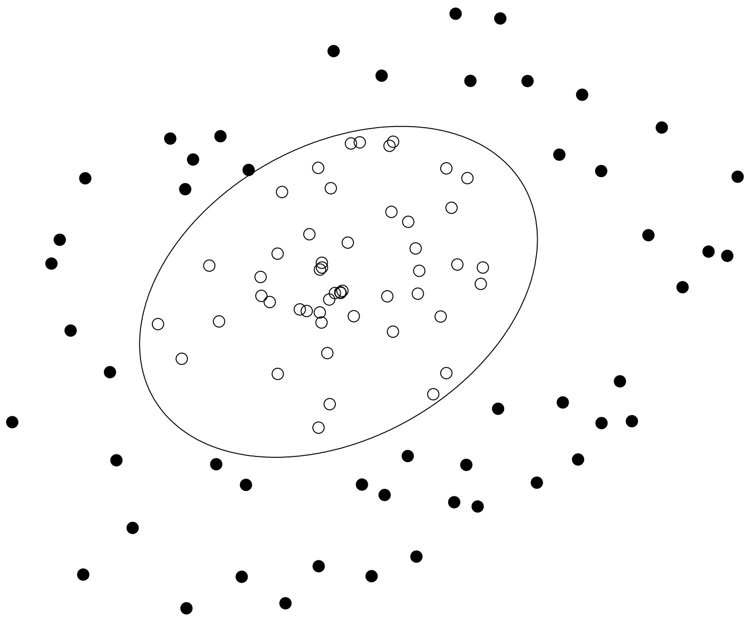
$$\begin{array}{ll} \text{find} & w \\ \text{such that} & \langle w, (x_j, x_j^2) \rangle \geq 1 \quad \text{for all } j \text{ such that } y_j = 1 \\ & \langle w, (x_j, x_j^2) \rangle \leq -1 \quad \text{for all } j \text{ such that } y_j = -1 \end{array}$$

  this is still a (larger) **linear** program in the variable w.

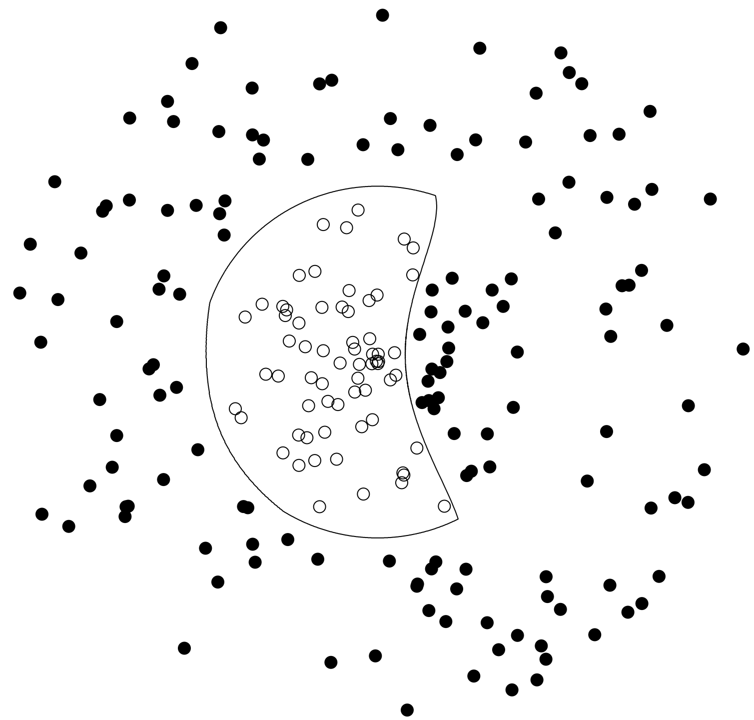
Nonlinear classification is **as easy** as linear classification.

Classification

This trick means that we are not limited to linear classifiers:



Separation by ellipsoid



Separation by 4th degree polynomial

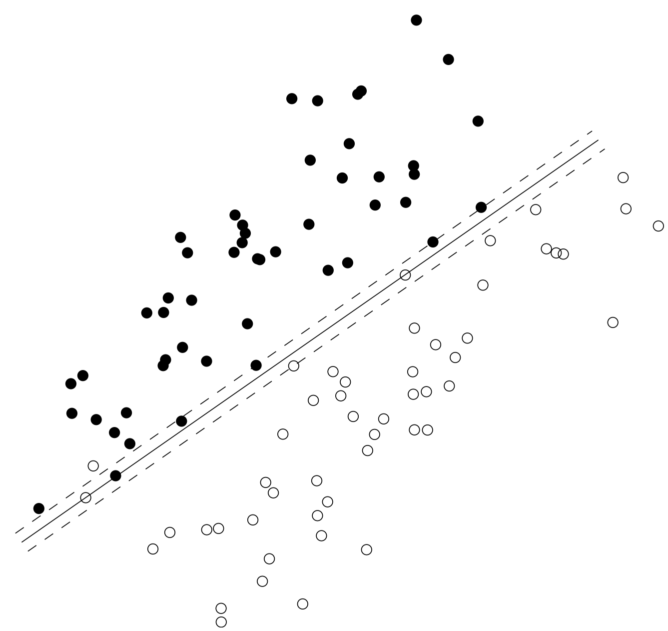
Both are **equivalent** to linear classification. . . just increase the dimension.

Classification: margin

Suppose the two sets are not **separable**. We solve instead

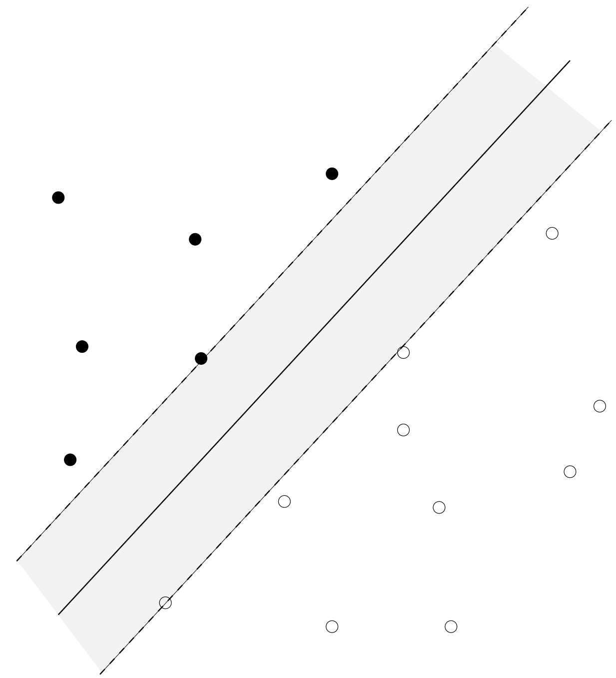
$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T u + \mathbf{1}^T v \\ \text{subject to} & \langle w, x_j \rangle \geq 1 - u_j \quad \text{for all } j \text{ such that } y_j = 1 \\ & \langle w, x_j \rangle < -(1 - v_j) \quad \text{for all } j \text{ such that } y_j = -1 \\ & u \succeq 0, \quad v \succeq 0 \end{array}$$

Can be interpreted as a heuristic for minimizing the number of misclassified points.



Robust linear discrimination

Suppose instead that the two data sets are well **separated.**



(Euclidean) distance between hyperplanes

$$\begin{aligned} \mathcal{H}_1 &= \{z \mid a^T z + b = 1\} \\ \mathcal{H}_2 &= \{z \mid a^T z + b = -1\} \end{aligned}$$

is $\mathbf{dist}(\mathcal{H}_1, \mathcal{H}_2) = 2/\|a\|_2$

to separate two sets of points by maximum margin,

$$\begin{array}{ll} \text{minimize} & (1/2)\|a\|_2 \\ \text{subject to} & a^T x_i + b \geq 1, \quad i = 1, \ldots, N \\ & a^T y_i + b \leq -1, \quad i = 1, \ldots, M \end{array} \tag{1}$$

(after squaring objective) a QP in a, b

Classification

In practice. . .

- The data has very high dimension.
- The classifier is highly nonlinear.
- **Overfitting is a problem:** in high dimensional spaces it is always possible to find a classifier, but the classifier itself can become somewhat meaningless.
  - Maximizing the **margin** helps.
  - Determine the tradeoff between error and margin by **cross-validation**.

Support Vector Machines: Duality

Given m data points $x_i \in \mathbb{R}^n$ with labels $y_i \in \{-1, 1\}$.

- The maximum margin classification problem can be written

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|w\|_2^2 + C\mathbf{1}^T z \\ \text{subject to} & y_i(w^T x_i) \geq 1 - z_i, \quad i = 1, \ldots, m \\ & z \geq 0 \end{array}$$

  in the variables w, $z \in \mathbb{R}^n$, with parameter $C > 0$.

- The Lagrangian is written

$$L(w, z, \alpha) = \frac{1}{2}\|w\|_2^2 + C\mathbf{1}^T z + \sum_{i=1}^{m} \alpha_i(1 - z_i - y_i w^T x_i)$$

  with dual variable $\alpha \in \mathbb{R}_+^m$.

Support Vector Machines: Duality

- The Lagrangian can be rewritten

$$L(w,z,\alpha) = \frac{1}{2}\left(\left\|w - \sum_{i=1}^{m}\alpha_i y_i x_i\right\|_2^2 - \left\|\sum_{i=1}^{m}\alpha_i y_i x_i\right\|_2^2\right) + (C\mathbf{1}-\alpha)^T z + \mathbf{1}^T\alpha$$

with dual variable $\alpha \in \mathbb{R}^n_+$.

- Minimizing in (w,z) we form the dual problem

$$\begin{array}{ll}\mbox{maximize} & -\frac{1}{2}\left\|\sum_{i=1}^{m}\alpha_i y_i x_i\right\|_2^2 + \mathbf{1}^T\alpha\\ \mbox{subject to} & 0 \leq \alpha \leq C\end{array}$$

- At the optimum, we must have

$$w = \sum_{i=1}^{m}\alpha_i y_i x_i \quad \mbox{and} \quad \alpha_i = C \mbox{ if } z_i > 0$$

(this is the representer theorem).

Support Vector Machines: the kernel trick

- If we write X the data matrix with columns x_i, the dual can be rewritten

$$\begin{array}{ll} \text{maximize} & -\frac{1}{2}\alpha^T \mathbf{diag}(y) X^T X \,\mathbf{diag}(y)\alpha + \mathbf{1}^T\alpha \\ \text{subject to} & 0 \leq \alpha \leq C \end{array}$$

- This means that the data only appears in the dual through the gram matrix

$$K = X^T X$$

  which is called the **kernel** matrix.

- In particular, the original **dimension n does not appear in the dual.**
- SVM complexity only grows with **the number of samples**, typically $O(m^{1.5})$.
- For linear classifiers: the magnitude of w_i gives a hint on the importance of variable i (for text: important words).

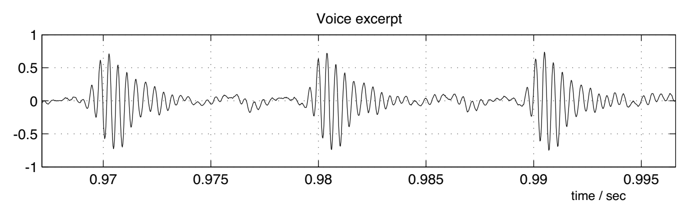
Support Vector Machines: the kernel trick

Kernels.

- All matrices written $K = X^T X$ can be kernel matrices.
- Easy to construct from highly diverse data types.

Examples. . .

- Kernels for **voice recognition**



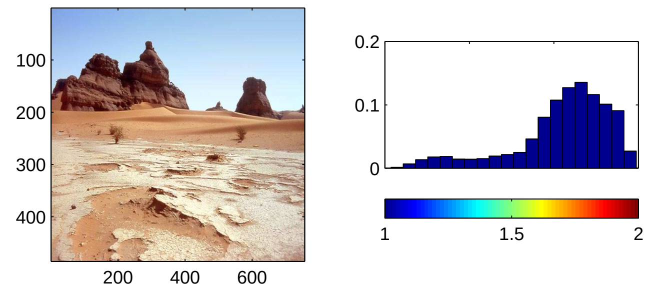
- Kernels for **gene sequence alignment**

```
AAB24882       TYHMCQFHCRYVNNHSGEKLYECNERSKAFSCPSHLQCHKRRQIGEKTHEHNQCGKAFPT 60
AAB24881       --------------------YECNQCGKAFAQHSSLKCHYRTHIGEKPYECNQCGKAFSK 40
                                   ****: .***:  * *:** * :****.:* *******..

AAB24882       PSHLQYHERTHTGEKPYECHQCGQAFKKCSLLQRHKRTHTGEKPYE-CNQCGKAFAQ- 116
AAB24881       HSHLQCHKRTHTGEKPYECNQCGKAFSQHGLLQRHKRTHTGEKPYMNVINMVKPLHNS 98
                **** *:************:***:**.: .***************    :  *.: :
```

Support Vector Machines: the kernel trick

- Kernels for **images**



- Kernels for **text classification**

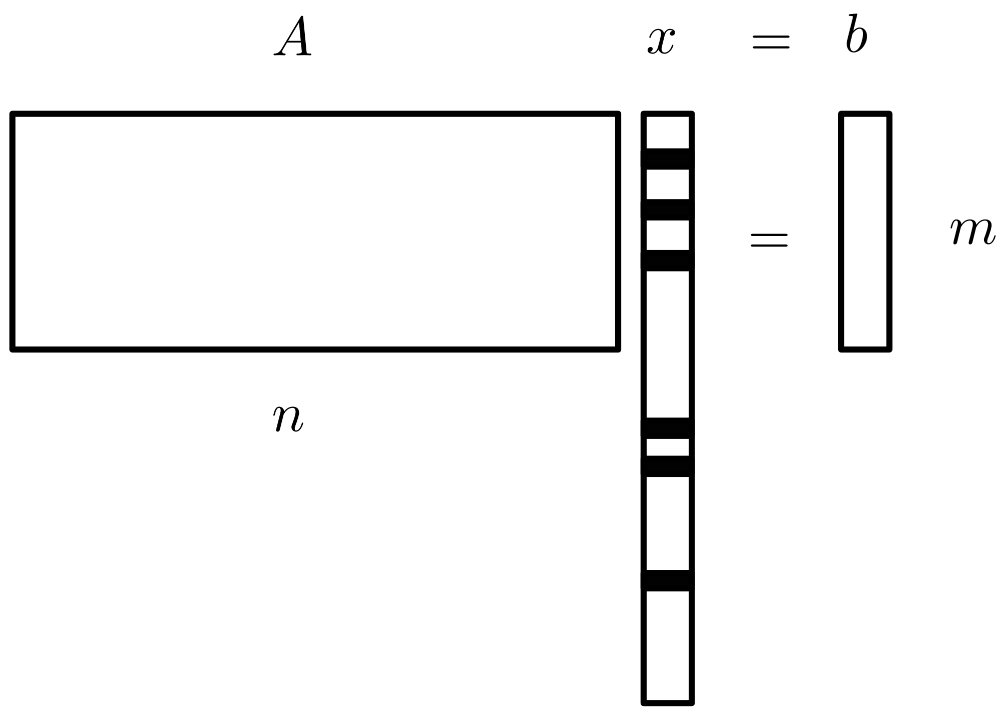
Ryanair Q3 ***profit up*** *30%,* ***stronger*** *than expected. (From Reuters.) DUBLIN, Feb 5 (Reuters) - Ryanair (RYA.I: Quote, Profile , Research) posted a 30 pct* ***jump*** *in third-quarter net* ***profit*** *on Monday, confounding analyst* ***expectations*** *for a* ***fall****, and* ***ramped up*** *its full-year* ***profit*** *goal while predicting big fuel-cost* ***savings*** *for the following year (. . .).*

profit	loss	up	down	jump	fall	below	expectations	ramped up
3	0	2	0	1	1	0	1	1

Compressed Sensing

Compressed Sensing

Consider the following underdetermined linear system



where $A \in \mathbb{R}^{m \times n}$, with $n \gg m$.

Can we find the **sparsest** solution?

Compressed Sensing

- **Signal processing:** We make a few measurements of a high dimensional signal, which admits a sparse representation in a well chosen basis (e.g. Fourier, wavelet). Can we reconstruct the signal exactly?

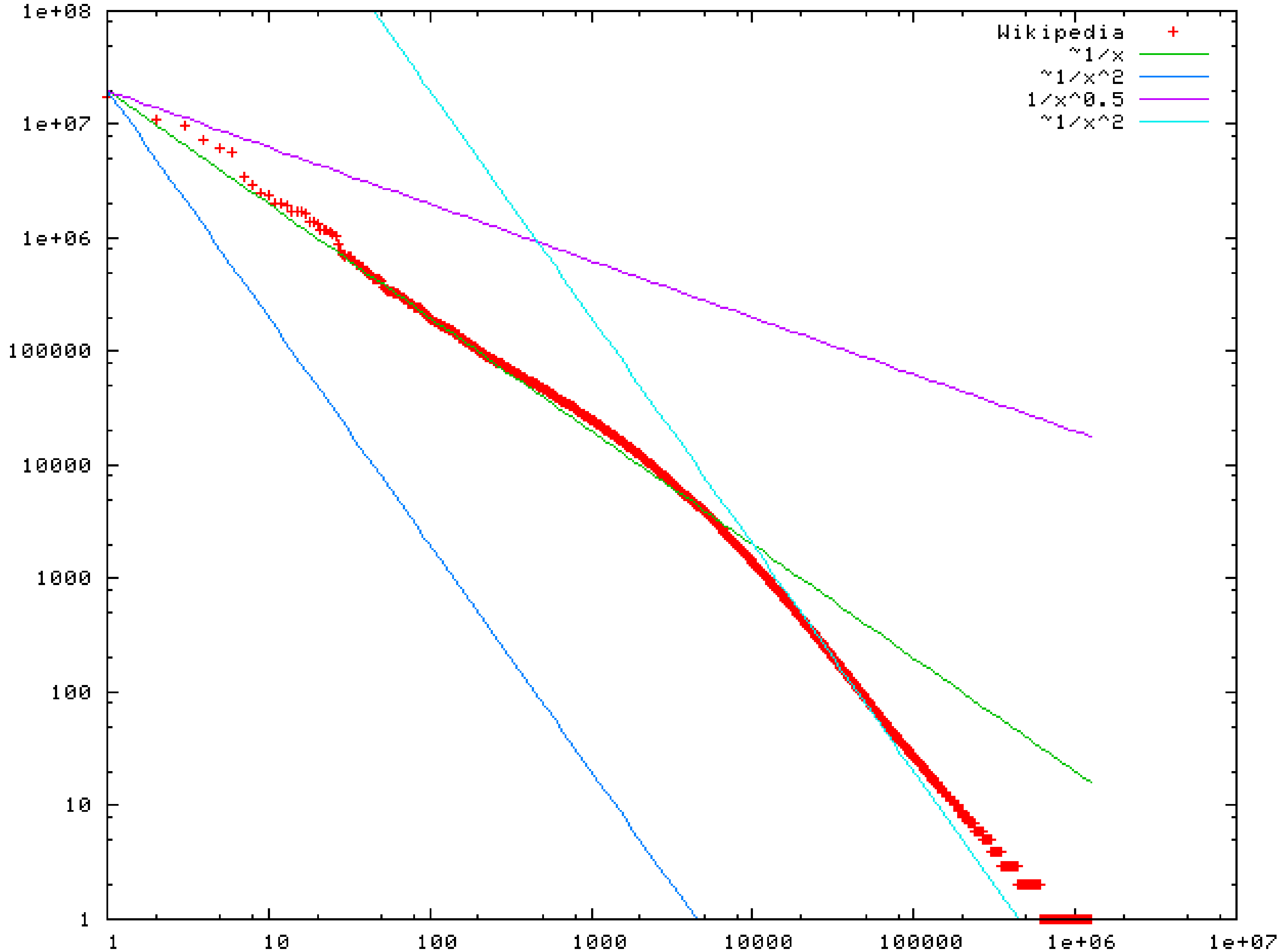
- **Coding:** Suppose we transmit a message which is corrupted by a few errors. How many errors does it take to start losing the signal?

- **Statistics:** Variable selection in regression (LASSO, etc).

Why **sparsity**?

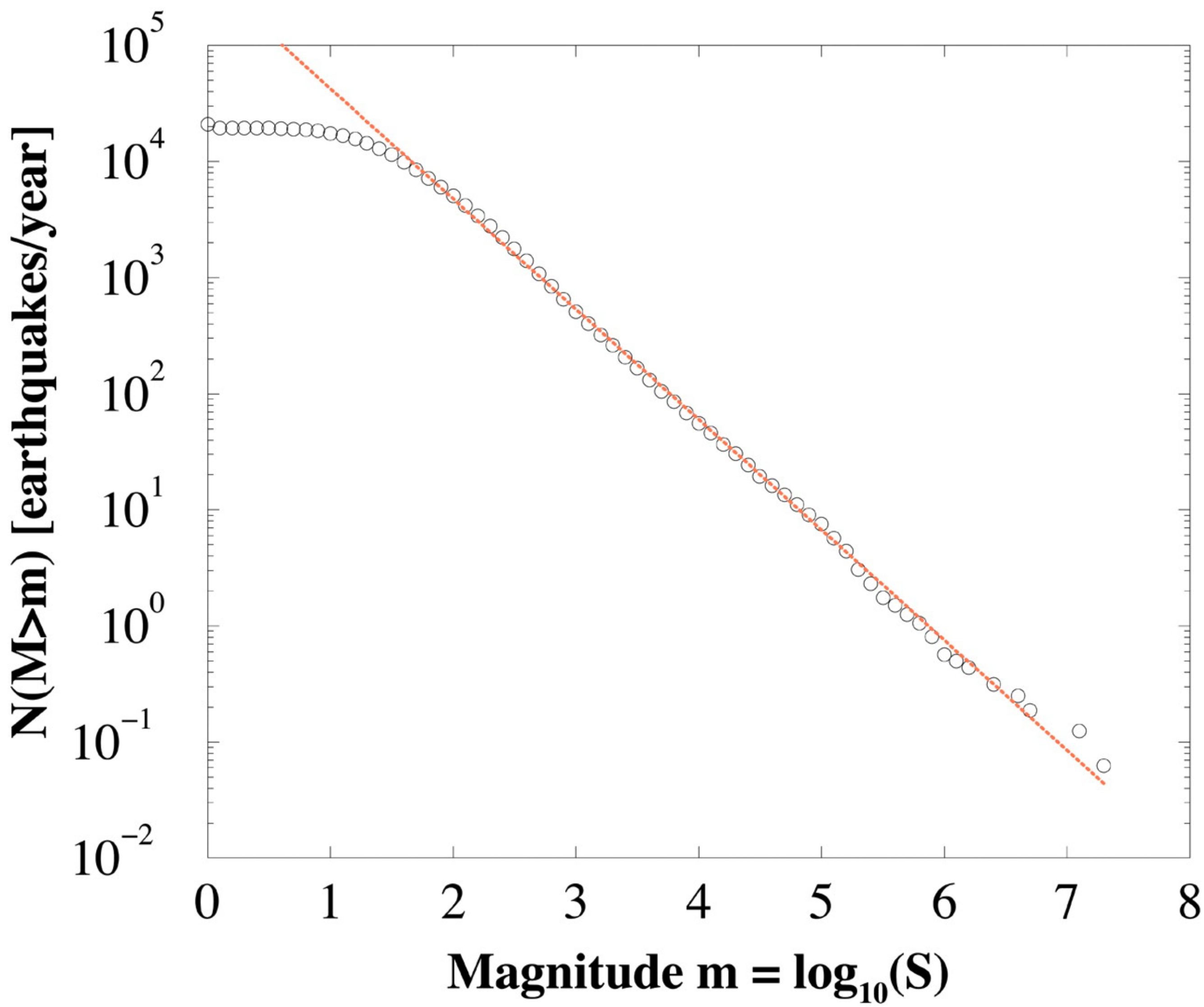
- Sparsity is a proxy for **power laws**. Most results stated here on sparse vectors apply to vectors with a power law decay in coefficient magnitude.
- Power laws appear everywhere...
  - Zipf law: word frequencies in natural language follow a power law.
  - Ranking: pagerank coefficients follow a power law.
  - Signal processing: $1/f$ signals
  - Social networks: node degrees follow a power law.
  - Earthquakes: Gutenberg-Richter power laws
  - River systems, cities, net worth, etc.

Compressed Sensing



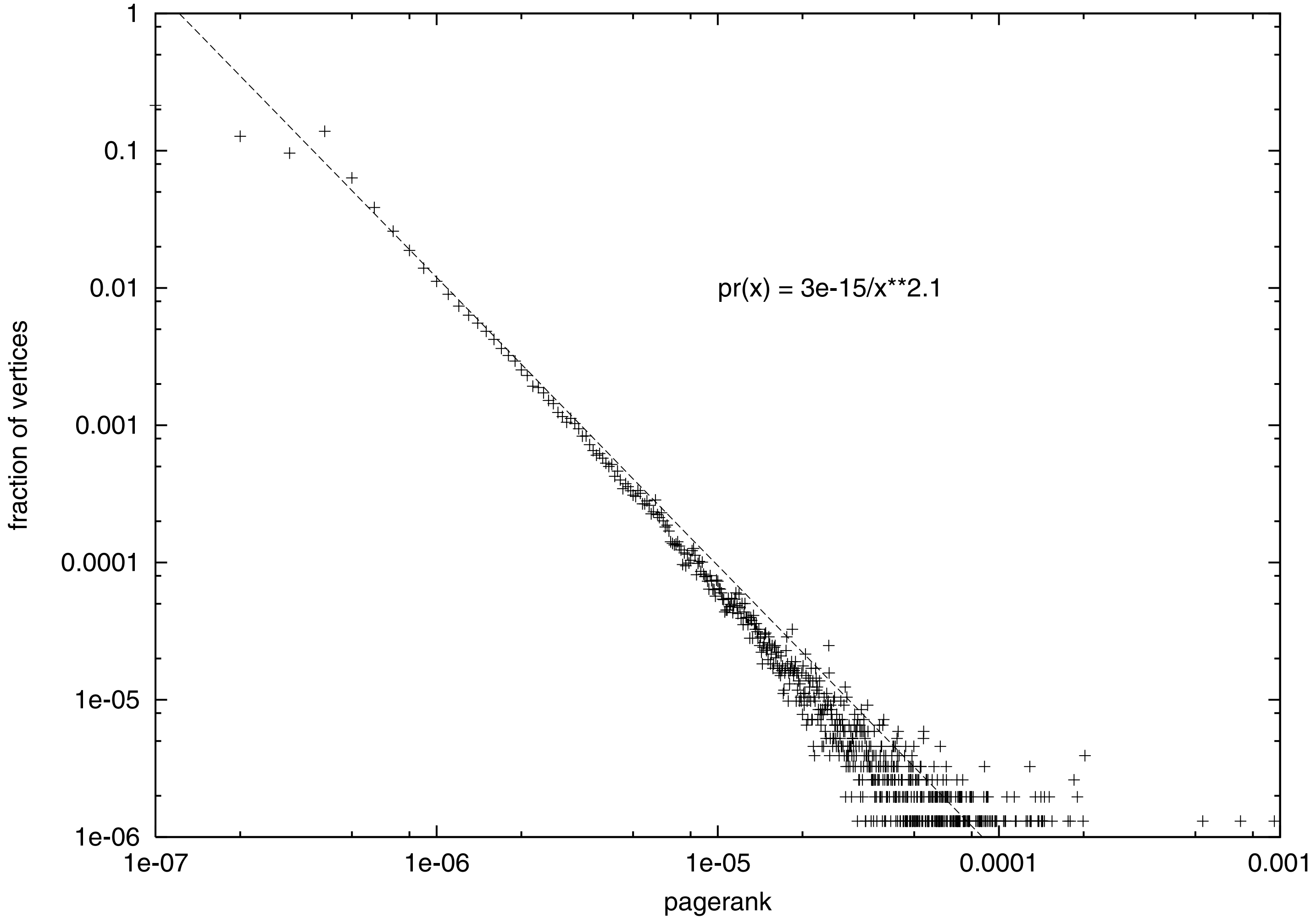
Frequency vs. word in Wikipedia (from Wikipedia).

Compressed Sensing



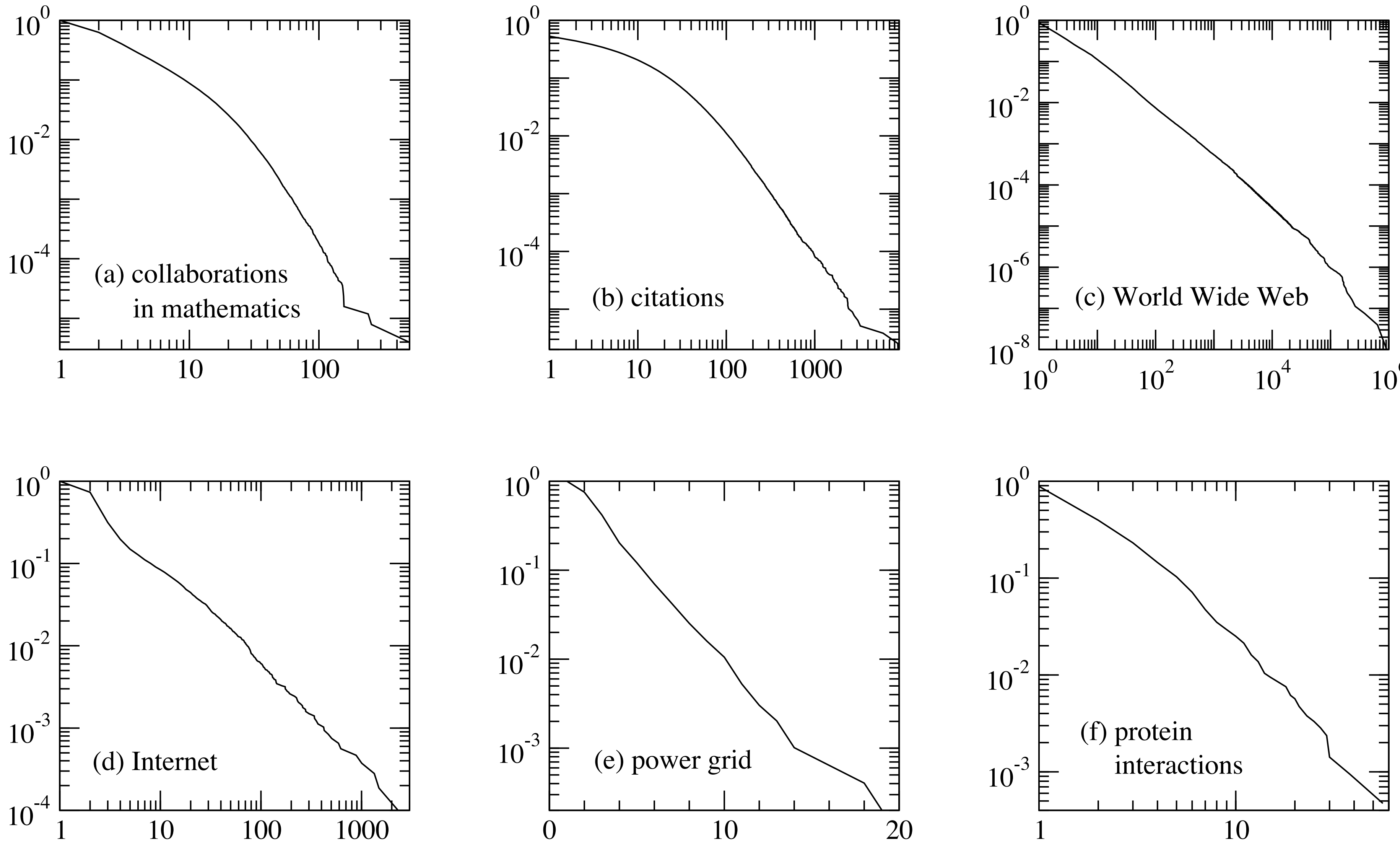
Frequency vs. magnitude for earthquakes worldwide. [Christensen et al., 2002]

Compressed Sensing



Pages vs. Pagerank on web sample. [Pandurangan et al., 2006]

Compressed Sensing



Cumulative degree distribution in networks. [Newman, 2003]

Compressed Sensing

- Getting the sparsest solution means solving:

$$\begin{array}{ll} \text{minimize} & \mathbf{Card}(x) \\ \text{subject to} & Ax = b \end{array}$$

which is a (hard) **combinatorial** problem in $x \in \mathbb{R}^n$.

- A classic heuristic is to solve instead:

$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & Ax = b \end{array}$$

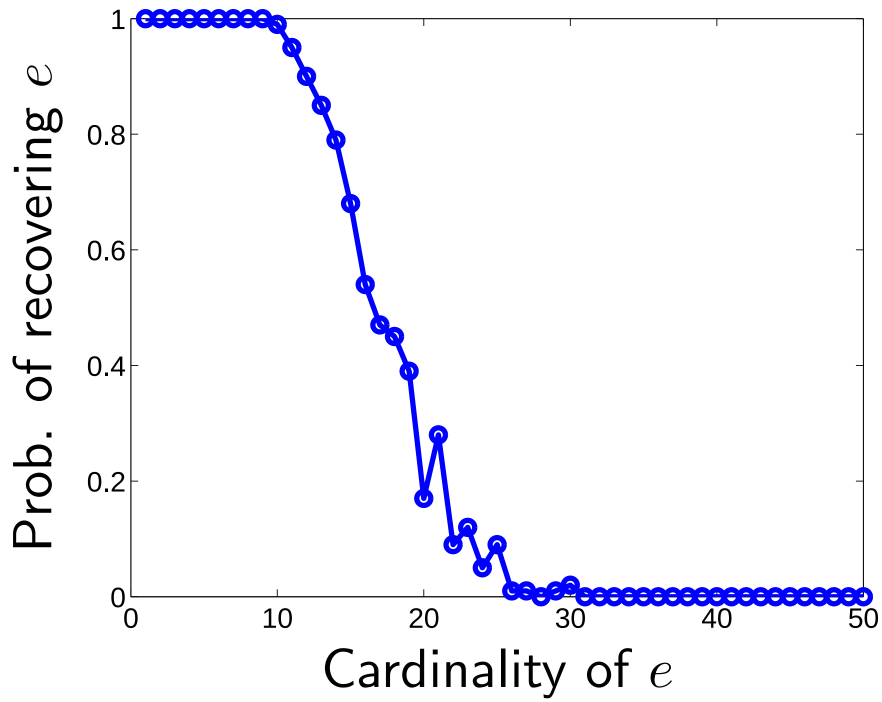
which is equivalent to an (easy) **linear program**.

Compressed Sensing

Example: we fix A, we draw many **sparse** signals e and plot the probability of perfectly recovering e by solving

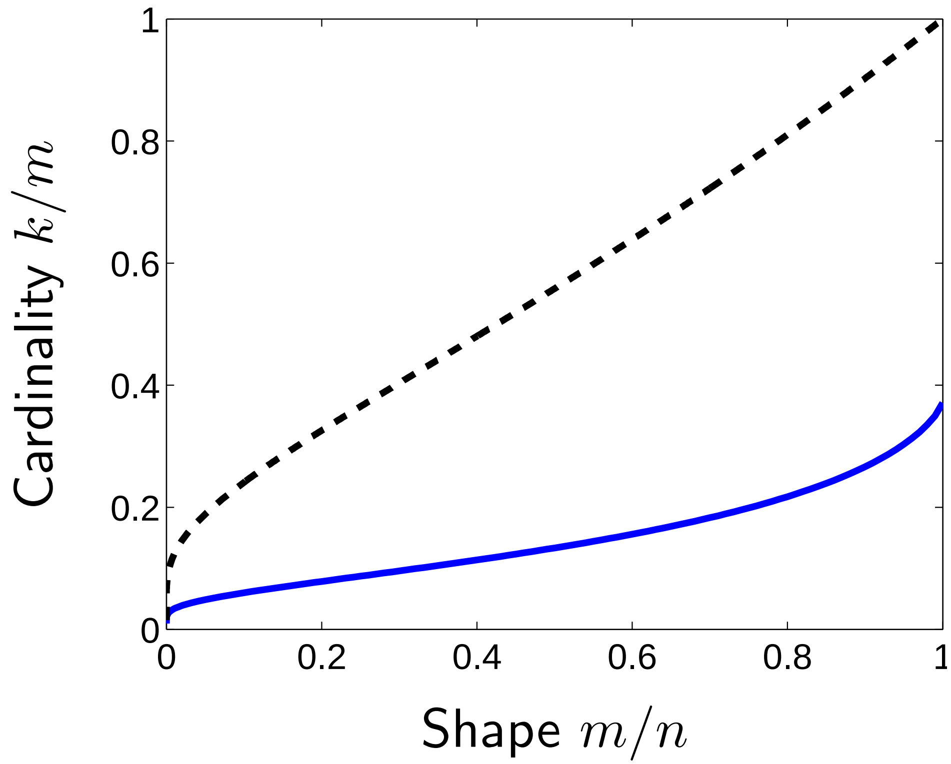
$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & Ax = Ae \end{array}$$

in $x \in \mathbb{R}^n$, with $n = 50$ and $m = 30$.



Compressed Sensing

- For some matrices A, when the solution e is sparse enough, the solution of the **linear program** problem is also the **sparsest** solution to $Ax = Ae$. [Donoho and Tanner, 2005, Candès and Tao, 2005]
- Let $k = \mathbf{Card}(e)$, this happens even when $\mathbf{k = O(m)}$ asymptotically, which is provably optimal.



Semidefinite Programming

Semidefinite Programming

A **linear program** (LP) is written

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \geq 0 \end{array}$$

where $x \geq 0$ means that the coefficients of the vector x are nonnegative.

Semidefinite Programming

A **semidefinite program** (SDP) is written

$$\begin{array}{ll} \text{minimize} & \mathbf{Tr}(CX) \\ \text{subject to} & \mathbf{Tr}(A_i X) = b_i, \quad i = 1, \ldots, m \\ & X \succeq 0 \end{array}$$

where $X \succeq 0$ means that the matrix variable $X \in \mathbf{S}_n$ is **positive semidefinite**.

- Nesterov and Nemirovskii [1994] showed that the **interior point algorithms** used for linear programs could be extended to semidefinite programs.
- Key result: **self-concordance** analysis of Newton's method (affine invariant smoothness bounds on the Hessian).

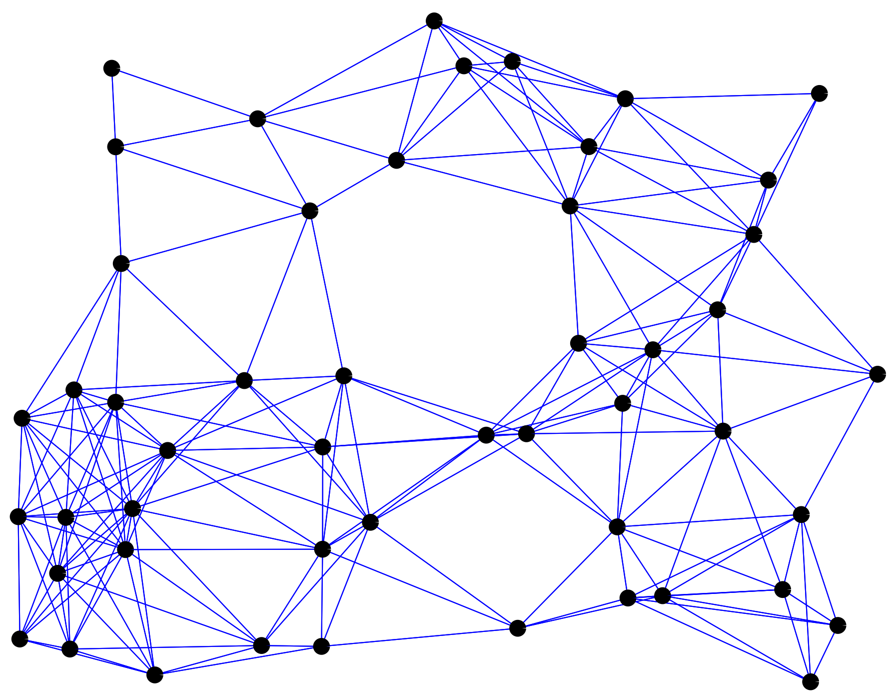
Semidefinite Programming

- Modeling
  - Linear programming started as a toy problem in the 40s, many applications followed.
  - Semidefinite programming has much stronger expressive power, many new applications being investigated today (cf. this talk).
  - Similar conic duality theory.
- Algorithms
  - Robust solvers for solving large-scale linear programs are available today (e.g. MOSEK, CPLEX, GLPK).
  - Not (yet) true for semidefinite programs. Very active work now on first-order methods, motivated by applications in statistical learning (matrix completion, NETFLIX, structured MLE, . . .).

Mixing rates for Markov chains & maximum variance unfolding

Mixing rates for Markov chains & unfolding

- Let $G = (V, E)$ be an **undirected graph** with n vertices and m edges.
- We define a **Markov chain** on this graph, and let $w_{ij} \geq 0$ be the transition rate for edge $(i, j) \in V$.



Mixing rates for Markov chains & unfolding

- Let $\pi(t)$ be the state distribution at time t, its evolution is governed by the heat equation

$$d\pi(t) = -L\pi(t)dt$$

with

$$L_{ij} = \begin{cases} -w_{ij} & \text{if } i \neq j,\ (i,j) \in V \\ 0 & \text{if } (i,j) \notin V \\ \sum_{(i,k)\in V} w_{ik} & \text{if } i = j \end{cases}$$

the **graph Laplacian** matrix, which means

$$\pi(t) = e^{-Lt}\pi(0).$$

Mixing rates for Markov chains & unfolding

[Sun, Boyd, Xiao, and Diaconis, 2006]

- Maximizing the mixing rate of the Markov chain means solving

$$\begin{array}{ll} \mbox{maximize} & t \\ \mbox{subject to} & L(w) \succeq t(\mathbf{I} - (1/n)\mathbf{1}\mathbf{1}^T) \\ & \sum_{(i,j)\in V} d_{ij}^2 w_{ij} \leq 1 \\ & w \geq 0 \end{array}$$

  in the variable $w \in \mathbb{R}^m$, with (normalization) parameters $d_{ij}^2 \geq 0$.
- Since $L(w)$ is an affine function of the variable $w \in \mathbb{R}^m$, this is a **semidefinite program** in $w \in \mathbb{R}^m$.

Mixing rates for Markov chains & unfolding

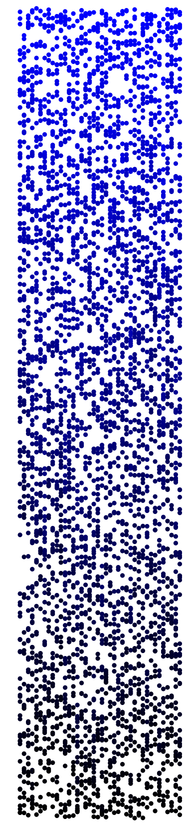
[Weinberger and Saul, 2006, Sun et al., 2006]

- The **dual** means solving

$$\begin{array}{ll} \text{maximize} & \mathbf{Tr}(X(\mathbf{I}-(1/n)\mathbf{1}\mathbf{1}^T)) \\ \text{subject to} & X_{ii}-2X_{ij}+X_{jj} \leq d_{ij}^2, \quad (i,j) \in V \\ & X \succeq 0, \end{array}$$

  in the variable $X \in \mathbf{S}_n$.
- This is a **maximum variance unfolding problem.**

Mixing rates for Markov chains & unfolding



From [Sun et al., 2006]: we are given pairwise 3D distances for k-nearest neighbors in the point set on the right. We plot the maximum variance point set satisfying these pairwise distance bounds on the right.

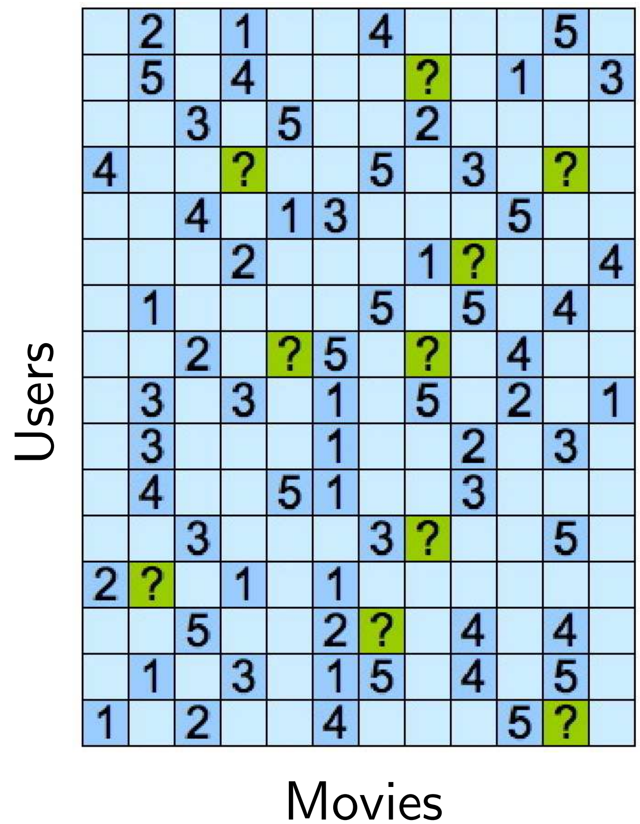
The NETFLIX challenge

NETFLIX

- **Video On Demand** and DVD by mail service in the United States, Canada, Latin America, the Caribbean, United Kingdom, Ireland, Sweden, Denmark, Norway, Finland.
- About 25 million users and 60,000 films.
- Unlimited streaming, DVD mailing, cheaper than CANAL+ :)
- Online movie recommendation engine.

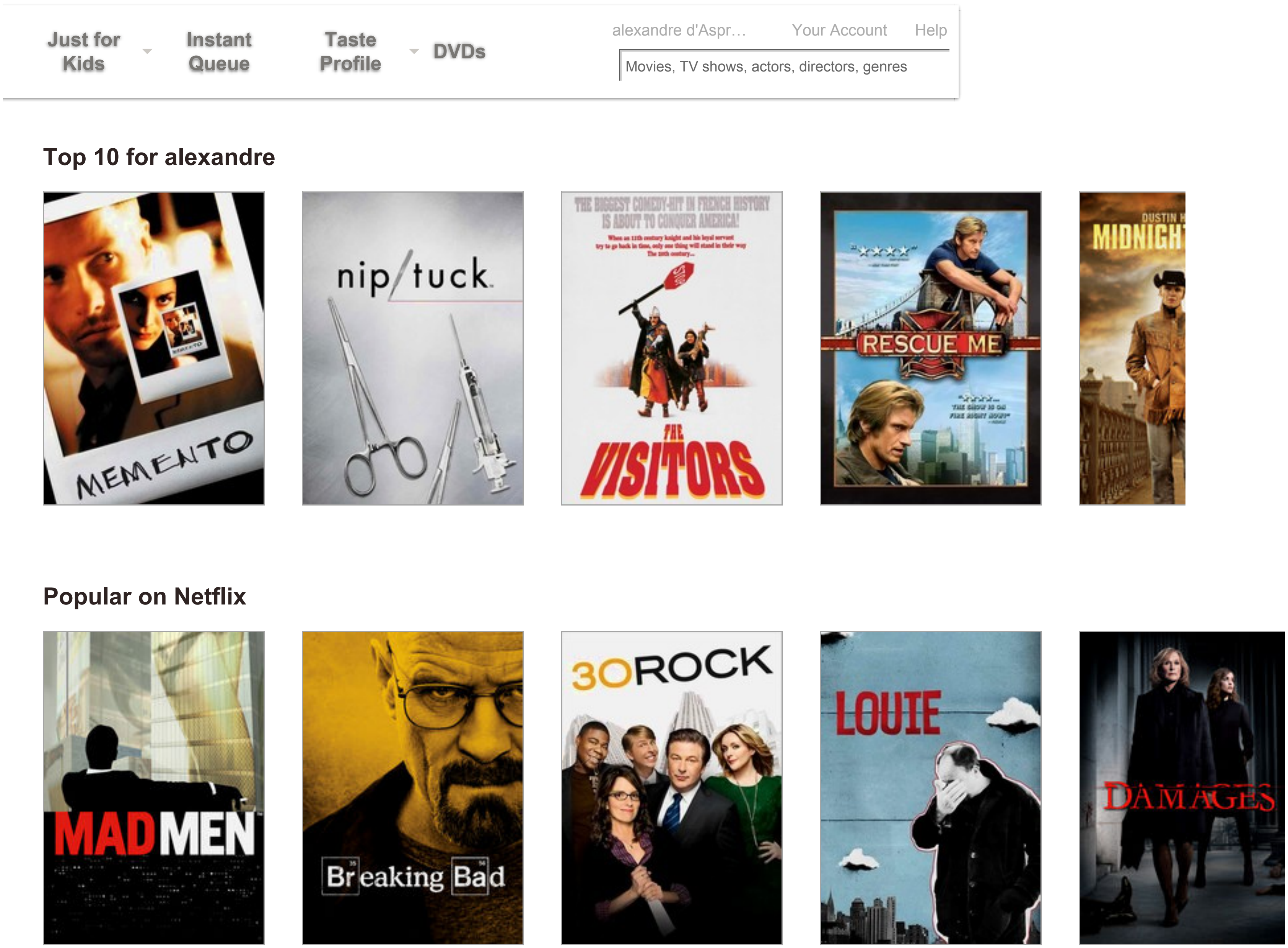
Collaborative prediction

- Users assign **ratings** to a certain number of movies:



- Objective: make recommendations for other movies. . .

NETFLIX


Just for Kids
Instant Queue
Taste Profile
DVDs
alexandre d'Aspr…
Your Account
Help
Movies, TV shows, actors, directors, genres
Top 10 for alexandre
MEMENTO
nip/tuck
THE BIGGEST COMEDY-HIT IN FRENCH HISTORY IS ABOUT TO CONQUER AMERICA!
THE VISITORS
RESCUE ME
MIDNIGHT
Popular on Netflix
MAD MEN
Breaking Bad
30 ROCK
LOUIE
DAMAGES

Collaborative prediction

Infer **user preferences** and **movie features** from user ratings.

- A **linear prediction model**

$$\text{rating}_{ij} = u_i^T v_j$$

  where u_i represents user characteristics and v_j movie features.

- This makes collaborative prediction a **matrix factorization** problem, We look for a linear model by factorizing $M \in \mathbb{R}^{n \times m}$ as:

$$M = U^T V$$

  where $U \in \mathbb{R}^{n \times k}$ represents user characteristics and $V \in \mathbb{R}^{k \times m}$ movie features.

- Overcomplete representation. . . We want k to be as small as possible, i.e. we seek a **low rank** approximation of M.

Collaborative prediction

- We would like to solve

$$\text{minimize} \quad \mathbf{Rank}(X) + c \sum_{(i,j)\in S} \max(0, 1 - X_{ij}M_{ij})$$

non-convex and numerically hard...

- Relaxation result in Fazel et al. [2001]: replace $\mathbf{Rank}(X)$ by its convex envelope on the spectahedron to solve:

$$\text{minimize} \quad \|X\|_* + c \sum_{(i,j)\in S} \max(0, 1 - X_{ij}M_{ij})$$

where $\|X\|_*$ is the **nuclear norm**, $i.e.$ sum of the singular values of X.

- This is a convex **semidefinite program** in X.

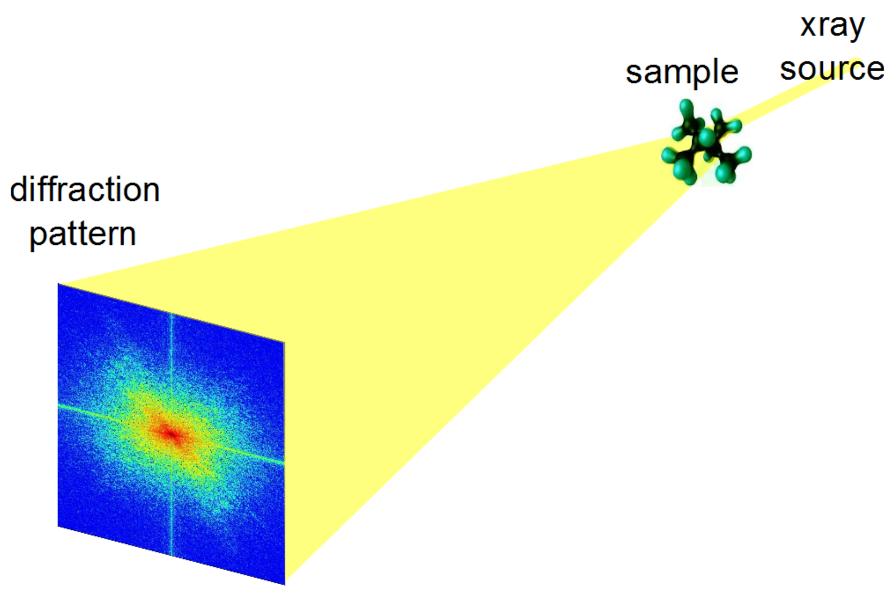
Collaborative prediction

NETFLIX challenge.

- NETFLIX offered \$1 million to the team who could improve the quality of its ratings by 10%, and \$50.000 to the first team to improve them by 1%.
- It took two weeks to beat the 1% mark, and three years to reach 10%.
- Very large number of scientists, students, postdocs, etc. working on this.
- The story could end here. But all this work had surprising outcomes. . .

Phase Recovery

Molecular imaging



(from [Candes et al., 2011b])

- CCD sensors only record the **magnitude** of diffracted rays, and loose the **phase**
- **Fraunhofer diffraction:** phase is required to invert the 2D Fourier transform

Phase Recovery

Focus on the **phase retrieval** problem, i.e.

$$\begin{array}{ll} \text{find} & x \\ \text{such that} & |\langle a_i, x\rangle|^2 = b_i^2, \quad i = 1, \ldots, n \end{array}$$

in the variable $x \in \mathbf{C}^p$.

- [Shor, 1987, Lovász and Schrijver, 1991] write

$$|\langle a_i, x\rangle|^2 = b_i^2 \quad \Longleftrightarrow \quad \mathbf{Tr}(a_i a_i^* x x^*) = b_i^2$$

- [Chai et al., 2011] and [Candes et al., 2011a] formulate phase recovery as a **matrix completion** problem

$$\begin{array}{ll} \text{Minimize} & \mathbf{Rank}(X) \\ \text{such that} & \mathbf{Tr}(a_i a_i^* X) = b_i^2, \quad i = 1, \ldots, n \\ & X \succeq 0 \end{array}$$

Phase Recovery

[Recht et al., 2007, Candes and Recht, 2008, Candes and Tao, 2010] show that under certain conditions on A and x_0, it suffices to solve

$$\begin{array}{ll}\text{Minimize} & \mathbf{Tr}(X)\\ \text{such that} & \mathbf{Tr}(a_i a_i^* X) = b_i^2, \quad i = 1, \ldots, n\\ & X \succeq 0\end{array}$$

which is a (convex) **semidefinite program** in $X \in \mathbf{H}_p$.

- Solving the **convex** semidefinite program yields a solution to the combinatorial, hard reconstruction problem.
- Apply results from **collaborative filtering** (NETFLIX) to **molecular imaging.**

Merci!

References

O. Bunk, A. Diaz, F. Pfeiffer, C. David, B. Schmitt, D.K. Satapathy, and JF Veen. Diffractive imaging for periodic samples: retrieving one-dimensional concentration profiles across microfluidic channels. *Acta Crystallographica Section A: Foundations of Crystallography*, 63 (4):306–314, 2007.

E. J. Candès and T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

E. J. Candes, T. Strohmer, and V. Voroninski. Phaselift : exact and stable signal recovery from magnitude measurements via convex programming. *To appear in Communications in Pure and Applied Mathematics*, 2011a.

E.J. Candes and B. Recht. Exact matrix completion via convex optimization. *preprint*, 2008.

E.J. Candes and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *Information Theory, IEEE Transactions on*, 56(5): 2053–2080, 2010.

E.J. Candes, Y. Eldar, T. Strohmer, and V. Voroninski. Phase retrieval via matrix completion. *Arxiv preprint arXiv:1109.0573*, 2011b.

A. Chai, M. Moscoso, and G. Papanicolaou. Array imaging using intensity-only measurements. *Inverse Problems*, 27:015005, 2011.

K. Christensen, L. Danon, T. Scanlon, and P. Bak. Unified scaling law for earthquakes, 2002.

D. L. Donoho and J. Tanner. Sparse nonnegative solutions of underdetermined linear equations by linear programming. *Proc. of the National Academy of Sciences*, 102(27):9446–9451, 2005.

M. Fazel, H. Hindi, and S. Boyd. A rank minimization heuristic with application to minimum order system approximation. *Proceedings American Control Conference*, 6:4734–4739, 2001.

J.R. Fienup. Phase retrieval algorithms: a comparison. *Applied Optics*, 21(15):2758–2769, 1982.

R. Gerchberg and W. Saxton. A practical algorithm for the determination of phase from image and diffraction plane pictures. *Optik*, 35: 237–246, 1972.

D. Griffin and J. Lim. Signal estimation from modified short-time fourier transform. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 32(2):236–243, 1984.

R.W. Harrison. Phase problem in crystallography. *JOSA A*, 10(5):1046–1055, 1993.

N. K. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

L. G. Khachiyan. A polynomial algorithm in linear programming (in Russian). *Doklady Akademiia Nauk SSSR*, 224:1093–1096, 1979.

L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1(2):166–190, 1991.

J. Miao, T. Ishikawa, Q. Shen, and T. Earnest. Extending x-ray crystallography to allow the imaging of noncrystalline materials, cells, and single protein complexes. *Annu. Rev. Phys. Chem.*, 59:387–410, 2008.

A. Nemirovskii and D. Yudin. Problem complexity and method efficiency in optimization. *Nauka (published in English by John Wiley, Chichester, 1983)*, 1979.

Y. Nesterov and A. Nemirovskii. *Interior-point polynomial algorithms in convex programming*. Society for Industrial and Applied Mathematics, Philadelphia, 1994.

MEJ Newman. The structure and function of complex networks. *Arxiv preprint cond-mat/0303516*, 2003.

G. Pandurangan, P. Raghavan, and E. Upfal. Using pagerank to characterize web structure. *Internet Mathematics*, 3(1):1–20, 2006.

B. Recht, M. Fazel, and P.A. Parrilo. Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization. *Arxiv preprint arXiv:0706.4138*, 2007.

H. Sahinoglou and S.D. Cabrera. On phase retrieval of finite-length sequences using the initial time sample. *Circuits and Systems, IEEE Transactions on*, 38(8):954–958, 1991.

N.Z. Shor. Quadratic optimization problems. *Soviet Journal of Computer and Systems Sciences*, 25:1–11, 1987.

N. Srebro. *Learning with Matrix Factorization*. PhD thesis, Massachusetts Institute of Technology, 2004.

J. Sun, S. Boyd, L. Xiao, and P. Diaconis. The fastest mixing Markov process on a graph and a connection to a maximum variance unfolding problem. *SIAM Review*, 48(4):681–699, 2006.

K.Q. Weinberger and L.K. Saul. Unsupervised Learning of Image Manifolds by Semidefinite Programming. *International Journal of Computer Vision*, 70(1):77–90, 2006.