MURI Meeting — June 20, 2001

# Estimation and Optimization:

# Gaps and Bridges

Laurent El Ghaoui

EECS

UC Berkeley

# goals

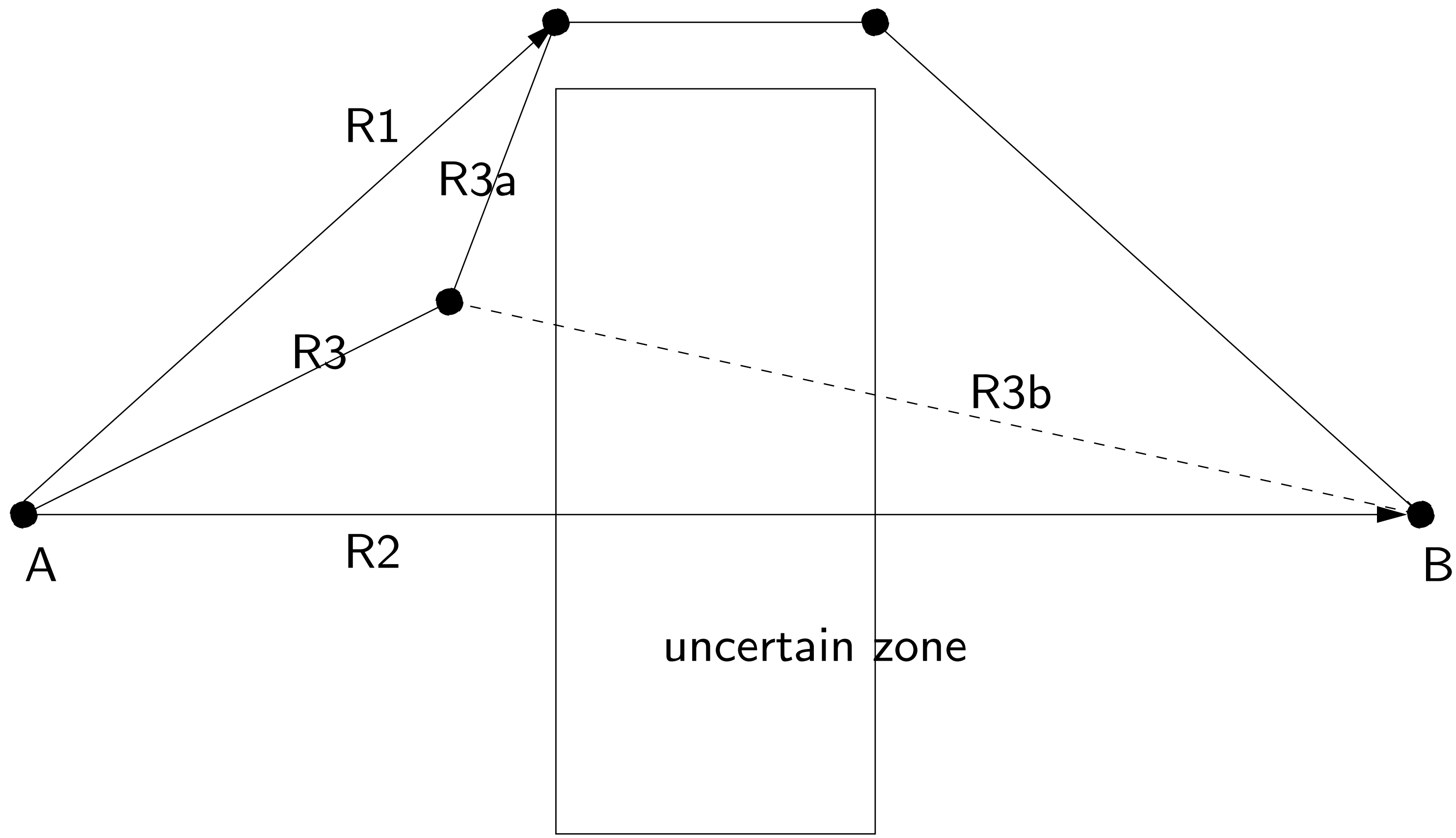
- currently, estimation (of model parameters) and optimization (of decision variables) are done **separately** in decision-making
- our goal is to integrate these two steps into one:
  - better handling of uncertainties (estimation errors)
  - "data-driven" optimization
- testbed involves a routing problem for aerial vehicles

# agenda

- a driving application
- classical approach to decision-making
- advances in optimization
- robust optimization
- moment-based optimization
- joint estimation and optimization

# driving application

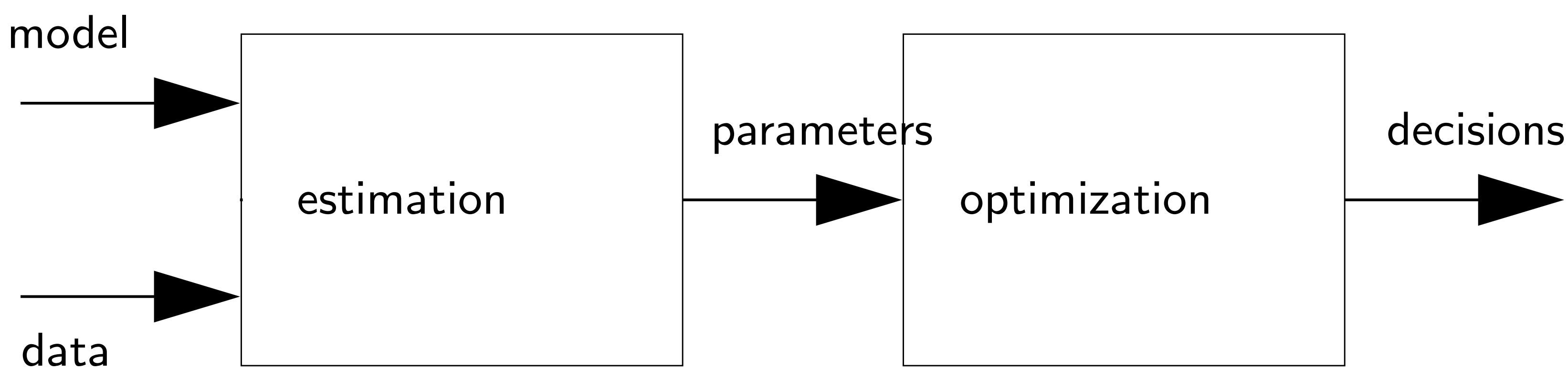
dynamic routing of aerial vehicles under uncertainty:



# driving application: challenges

- this is a problem with **recourse**
  (information is revealed as process evolves)
- it has **stochastic uncertainty** (network capacity is random)
- the model parameters (transition matrix governing state of nodes) are full of **estimation errors**
- a robust solution, which is not too conservative, is needed
  (trade-off risk vs. reward)

# classical approach to decision-making




- **estimation** phase: compute an estimate of model parameters
  *fields:* identification, estimation, data-fitting, etc
- **decision** phase: choose decision variables in an "optimal" way
  *fields:* optimization, control, etc

above processes can be done iteratively ("adaptive control")

# optimization under uncertainty

**traditional optimization** assumes perfect knowledge of data

- in effect, **tunes** solution to (imperfect) data
- hence, approach is **not reliable** in practice

recent study (Ben-Tal, Nemirovski, 2000) found nearly $90\%$ of all linear programs proposed in NETLIB to be *extremely sensitive* to implementation errors

## advances in optimization under uncertainty

recent approaches account for data uncertainty:

- **stochastic optimization:** distribution of (random) uncertainty is known
- **robust optimization:** bounds on uncertainty are known
- **moment-based optimization:** use partial knowledge (eg, moments) on distribution of random uncertainties

*main challenge: obtain tractable, yet accurate, approximations to these hard problems*

# robust optimization

**principle:**

- assume data is only known within given bounds
- find a solution that is guaranteed to meet the constraints whatever the uncertainty within its bounds
- similar to a **game** between decision-maker and uncertainty
- leads to very **hard** problems in general

# robust optimization: results

- efficient approximations, ie, solutions with **guaranteed** robustness, based on convex optimization

  (uses a tractable sufficient condition for robustness)

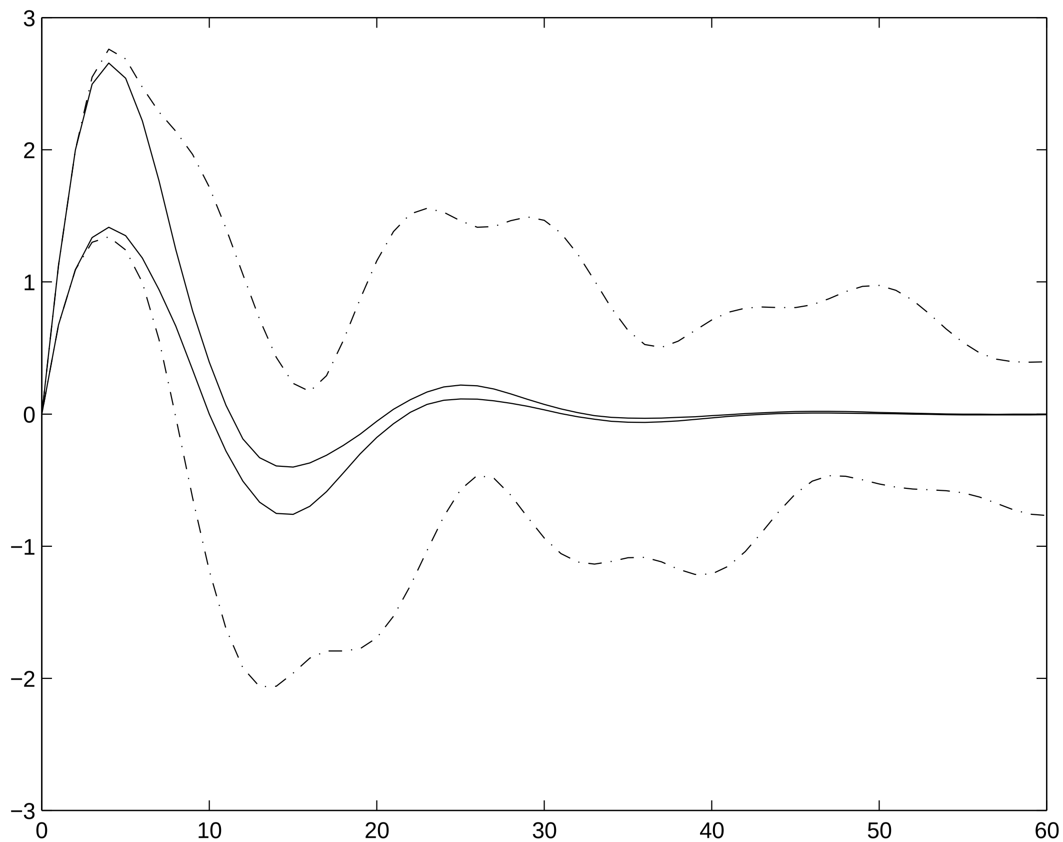
- evaluation of **quality** of results

  (how conservative is the sufficient condition?)

*vast array of success stories: control systems, circuit design, network design, etc*

# example: worst-case simulation

simulate state bounds for an uncertain dynamical system



# robust optimization: challenges

- handling **random** uncertainty

  *possible answer:* moment-based optimization

- obtain the **bounds** on uncertainty from data

  *possible answer:* joint estimation and optimization

- handling **recourse**

## moment-based optimization

- assume uncertainty is random, with **partially** known distribution (typically, moments, such as expected values)
- design against worst-case distribution
- results in an estimate of the (worst-case) distribution that is most prudent to choose
- hard problems in general but efficient approximations based on convex optimization

## example: minimax probability classifier

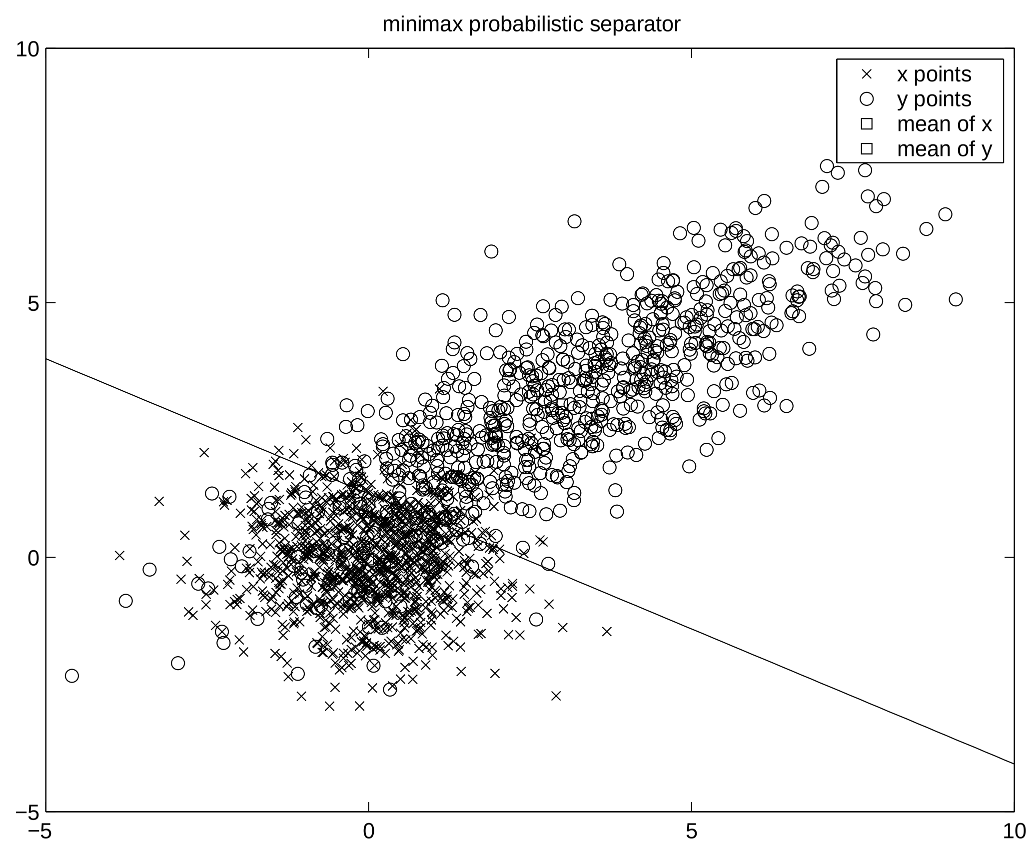
(with G. Lanckriet, C. Bhattacharyya, M. Jordan)

**problem:** classify data points in two separate clusters

**approach:**

- assume data points are distributed according to a distribution of which we only known moments (mean and covariances)
- minimize the *worst-case* probability of misclassification, over all distributions having the observed mean and covariances
- problem is solved exactly as a simple convex optimization problem (second-order cone program)

# minimax probability classifier



# joint estimation and optimization

- robust optimization requires bounds on model parameters
  (a point estimate is not sufficient anymore!)
- how do we get these bounds from data?
  (these bounds should be accurate!)

we'll show two examples of joint approach

# worst-case identification/simulation

for dynamical systems, in a sliding window fashion:

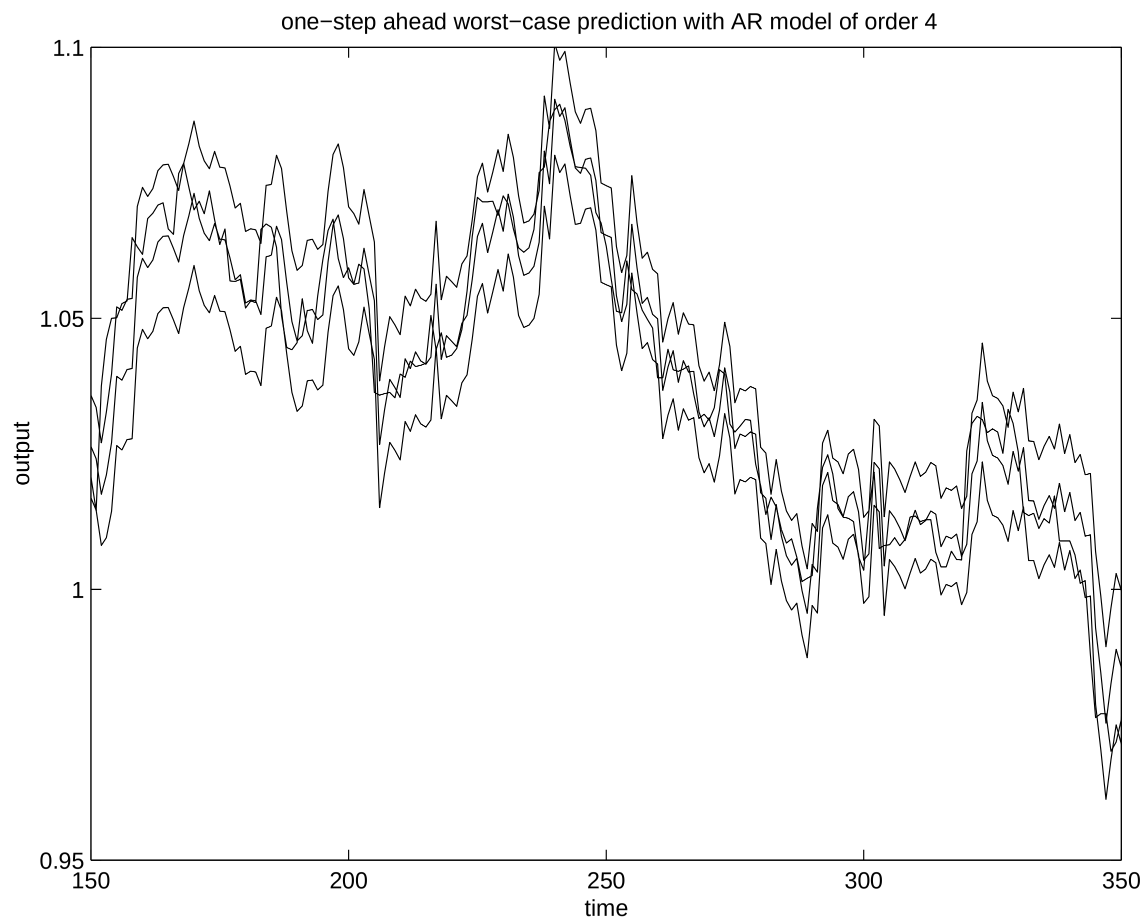
- build an "uncertain system" model

  for example, a set of recursive, linear equations with unknown-bout-bounded coefficients

- do a one-step ahead prediction based on above model

*above can be solved using convex optimization*

# example


one−step ahead worst−case prediction with AR model of order 4
1.1
1.05
1
0.95
output
150
200
250
300
350
time

## estimating a transition matrix

- many applications require the knowledge of the transition matrix of a Markov chain
- this matrix often arises in the context of stochastic dynamic programming, Markov decision processes, etc
- in driving application (optimal routing of aerial vehicles), transition matrix describes (say, weather) uncertainty

# estimating the transition matrix

to estimate transition matrix, solve a maximum-likelihood problem:

$$\text{maximize } L(P) \text{ subject to } P \in \mathcal{P}$$

where

- $P$ is the transition matrix
- set $\mathcal{P}$ describes bounds (a priori knowledge) on $P$ (such as, sign)
- $L(P)$ is the (concave) log-likelihood function (depends on data)

# estimating bounds

- *classical* approach: estimate intervals of confidence on $P$

  $\rightarrow$ leads to very conservative results

- *confidence region* approach: describe uncertainty by a set of the form

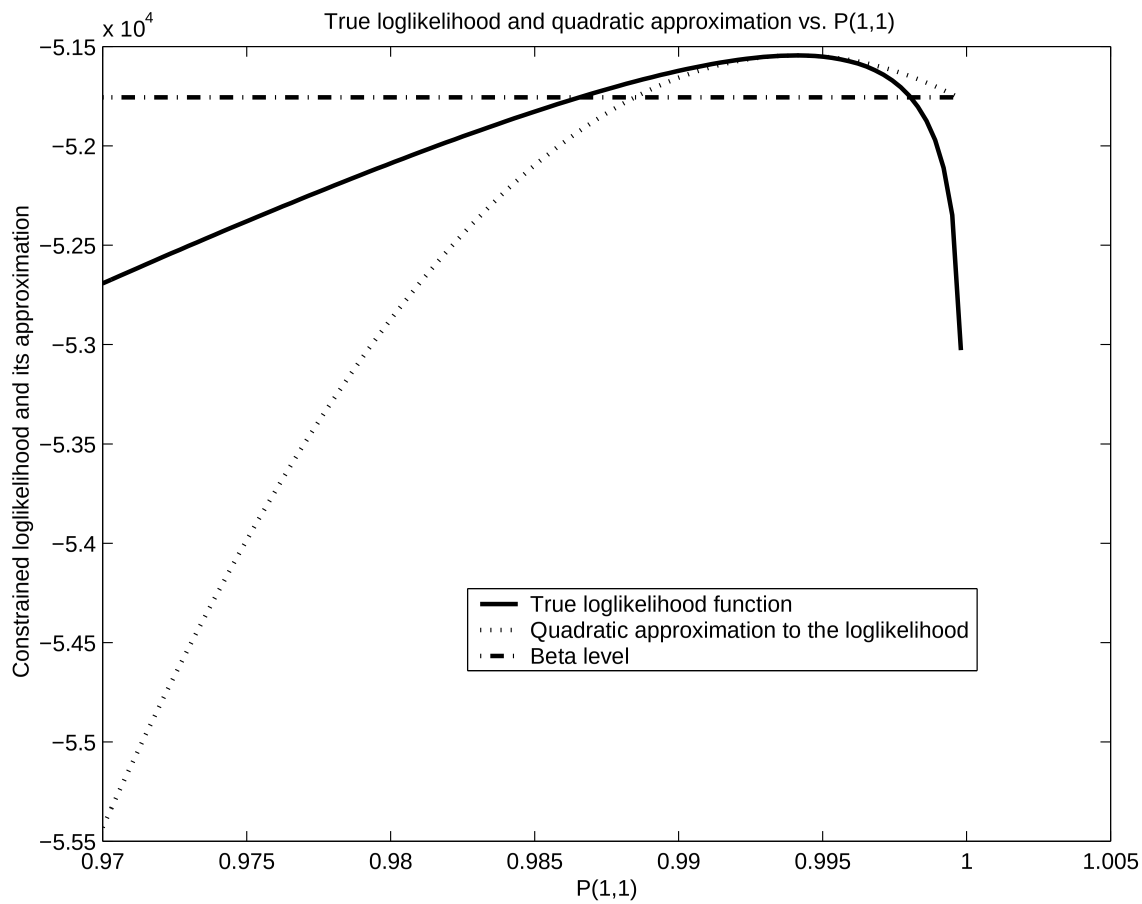
$$\{P \in \mathcal{P} \mid L(P) \geq \beta\}$$

  where the level $\beta$ is appropriately chosen (below the maximal value of $L$ over $\mathcal{P}$)

- *Fisher information matrix* approach: use quadratic approximation of the above set

  (turns out, not necessary to do this approximation)

# example



True loglikelihood and quadratic approximation vs. P(1,1)
x 10^4
−5.15
−5.2
−5.25
−5.3
−5.35
−5.4
−5.45
−5.5
−5.55
Constrained loglikelihood and its approximation
0.97
0.975
0.98
0.985
0.99
0.995
1
1.005
P(1,1)
True loglikelihood function
Quadratic approximation to the loglikelihood
Beta level

# results

maximal and average difference between the ML estimate and its bounds: Ell (Ellipsoid approach) and CL (using CL-regions)

| | $\max(P^* - P_l)$ | mean$(P^* - P_l)$ | $\max(P_u - P^*)$ | mean$(P_u - P^*)$ |
|---|---|---|---|---|
| CL | 0.0381 | 0.0035 | 0.0339 | 0.0060 |
| Ell | 0.1485 | 0.0114 | 0.0407 | 0.0052 |

# associated decision problem

often, we are interested in optimal resource allocation problems involving a transition matrix

**classical approach:**

- get "best estimate" of transition matrix, $P^*$
- solve decision problem

$$\min_{V \in \mathcal{V}} V \cdot P^*$$

  where $\cdot$ means scalar product, $\mathcal{V}$ is decision set

# robust decision problem

**robust version:**

- estimate confidence region (*i.e.*, parameter $\beta$)
- solve game

$$\min_{V \in \mathcal{V}} \max_{P \in C_\beta} V \cdot P$$

above problem can be expressed as a classical convex program (easy to solve)

# summary of results

- extended robust optimization to random uncertainty with **moment-based optimization**, and applied it to a new **classification** problem
- developed a framework for a better (less conservative) treatment of uncertainty in optimization: **joint estimation and optimization**
- obtained specific results (algorithm) for joint estimation/optimization of a **transition matrix**
  (problem forms the basis of many others)
- developed a **joint identification and prediction** scheme for dynamical systems with bounded uncertainty
  $\rightarrow$ obtained an $O(n^3)$ algorithm solving this problem

students: A. Varma, G. Lanckriet

## conclusions

- a better integration of estimation and optimization is **necessary** to address real-life decision-making challenges
- our goal is to make this a unified, data-driven process
- next steps: robust dynamic programming, robust optimization problems with recourse, robust combinatorial optimization