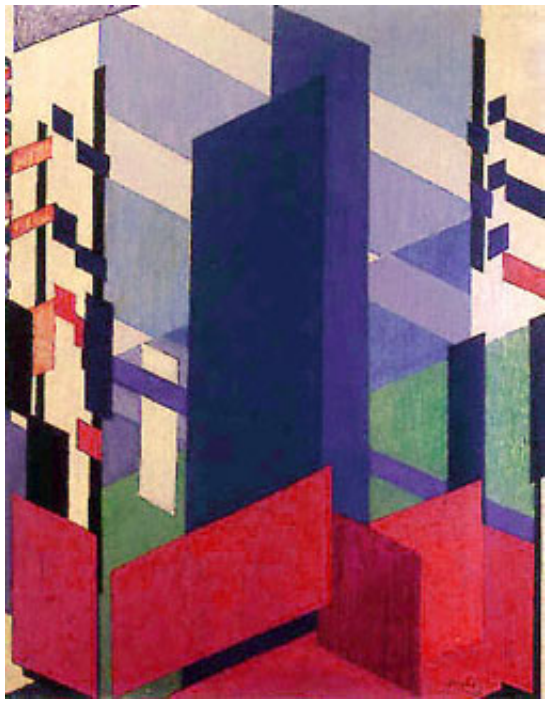
COURSE ON LMI OPTIMIZATION
WITH APPLICATIONS IN CONTROL
PART I.4

# SOLVING LMIs

Didier HENRION
www.laas.fr/∼henrion
henrion@laas.fr



Architecture Philosophique (1913)
František Kupka (1871-1957)

November 2003

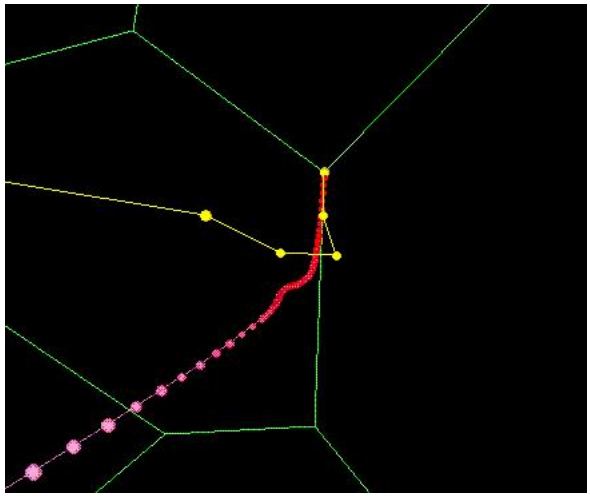
# History

## Convex programming

- logarithmic barrier function (Frisch 1955)
- method of centers (Huard 1967)

## Interior-point (IP) methods

- ellipsoid algorithm (Khachiyan 1979)
polynomial bound on worst-case iteration count
- IP methods for LP (Karmarkar 1984)
improved complexity bound and efficiency - now about 50% of commercial LP solvers
- self-concordant barrier functions (Nesterov, Nemirovski 1988, Alizadeh 1991) - IP methods for general convex programs, in particular SDP and LMI



Steve Wright's artistic view of interior point methods in action
Iterates (yellow) LP edges (green) central path (red)

## Logarithmic barrier function

For the optimization problem

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \geq 0 \end{array}$$

where the $f_i(x)$ are twice continuously differentiable convex functions, we define the logarithmic barrier function

$$\phi(x) = -\sum_i \log f_i(x) = \log \prod_i f_i(x)^{-1}$$

which is convex in the interior $f_i(x) > 0$ of the feasible set

Then we solve the unconstrained convex problem

$$\min f_0(x) + \mu\phi(x)$$

where $\mu > 0$ and the term $\mu\phi(x)$ acts as a "repellent" of the boundary

The minimum is attained in the interior = interior-point method

# Descent methods

To solve an unconstrained optimization problem

$$\min f(x)$$

for $x \in \mathbb{R}^n$ we produce a minimizing sequence

$$x_{k+1} = x_k + t_k \Delta x_k$$

where $\Delta x_k \in \mathbb{R}^n$ is the step or search direction
and $t^{(k)} \geq 0$ is the step size or step length

A descent method consists in finding
a sequence $\{x_k\}$ such that

$$f(x^\star) \leq \cdots f(x_{k+1}) < f(x_k)$$

where $x^\star$ is the optimum

## General descent method

0. given starting point $x$
1. determine descent direction $\Delta x$
2. line search: choose step size $t > 0$
3. update: $x = x + t\Delta x$
4. go to step 1 until a stopping criterion is satisfied

## Newton's method

A particular choice of search direction is the Newton step

$$\Delta x = -\nabla^2 f(x)^{-1} \nabla f(x)$$

where $\nabla f(x)$ is the gradient
and $\nabla^2 f(x)$ is the Hessian

This step $y = \Delta x$ minimizes the second-order Taylor approximation

$$\widehat{f}(x+y) = f(x) + \nabla f(x)^T y + y^T \nabla^2 f(x) y/2$$

and it is the steepest descent direction for the quadratic norm defined by the Hessian

Quadratic convergence near the optimum

## Self-concordance

Shortcomings of Newton's method:
- number of required Newton steps

hardly estimated in practice
- analysis depends on used coordinate system

Theory of self-concordant functions:
- number of Newton steps easily estimated
- affine-invariant property

Smooth convex functions with 2nd derivatives Lipschitz continuous with respect to the metric induced by the Hessian:

$$|f'''(x)| \leq 2f''(x)^{3/2}$$

include many logarithmic barrier functions

For LP, QP or SDP 1st and 2nd derivatives of standard self-concordant barriers can be found easily in closed form

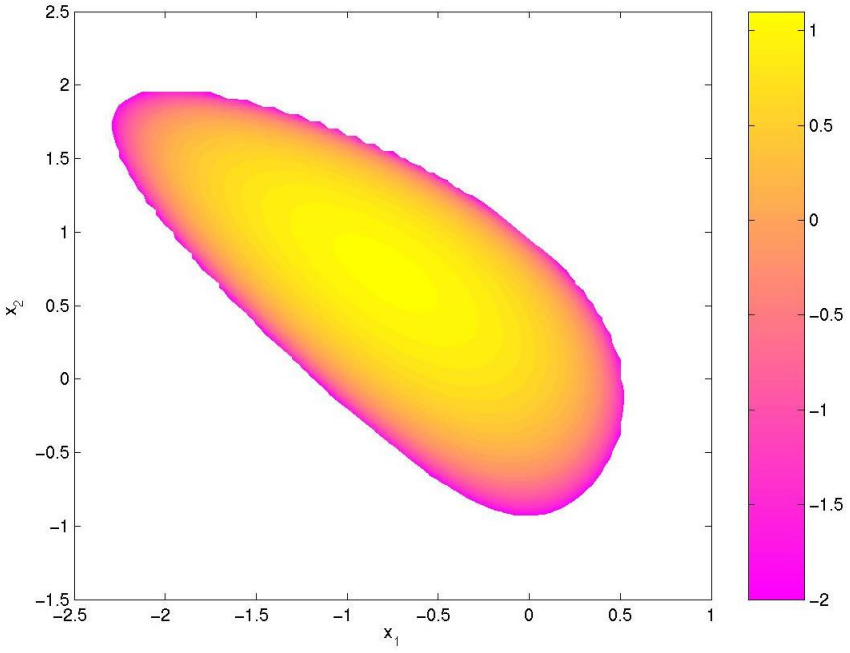
## Barrier function for an LMI

Given an LMI constraint $F(x) \succeq 0$ we define its logarithmic barrier function

$$\phi(x) = -\log\det F(x) = \log\det F(x)^{-1}$$

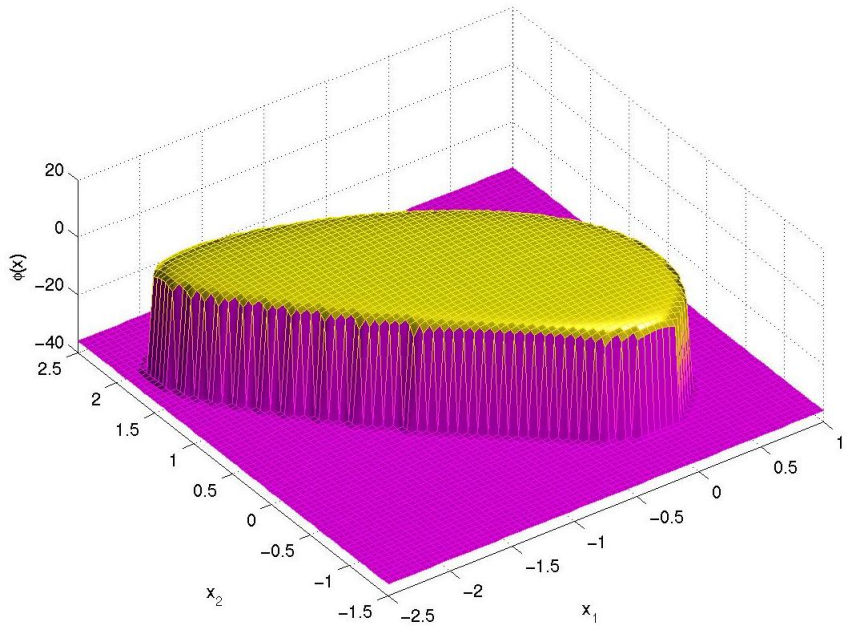
This function is analytic, convex and self-concordant on $\{x \,:\, F(x) \succ 0\}$

The optimum of $\min \phi(x)$ is called the analytic center of the LMI



## Gradient and Hessian for an LMI

The barrier function $\phi(x)$ is flat in the interior of the feasible set and sharply increases toward the boundary




Closed-form expressions for gradient

$$\begin{aligned}(\nabla\phi(x))_i &= -\text{trace}\, F(x)^{-1}F_i \\ &= -\text{trace}\, F(x)^{-1/2}F_iF(x)^{-1/2}\end{aligned}$$

and Hessian

$$\begin{aligned}(\nabla^2\phi(x))_{ij} &= \text{trace}\, F(x)^{-1}F_iF(x)^{-1}F_j \\ &= \text{trace}\left(F(x)^{-1/2}F_iF(x)^{-1/2}\right)\left(F(x)^{-1/2}F_jF(x)^{-1/2}\right)\end{aligned}$$

# IP methods for SDP

Primal / dual SDP

$$\begin{array}{ll} \min_X & \text{trace}\, CX \\ \text{s.t.} & \text{trace}\, A_i X = b_i \\ & X \succeq 0 \end{array} \qquad \begin{array}{ll} \max_y & b^T y \\ \text{s.t.} & Z = C - \sum_i y_i A_i \\ & Z \succeq 0 \end{array}$$

## Primal methods

$$\begin{array}{ll} \min_X & \text{trace}\, CX - \mu \log\det X \\ \text{s.t.} & \text{trace}\, A_i X = b_i \end{array}$$

where parameter $\mu$ is sequentially decreased to zero and iterates $X_k$ are always primal feasible

## Dual methods

$$\begin{array}{ll} \max_{y,Z} & b^T y + \mu \log\det Z \\ \text{s.t.} & Z = C - \sum_i y_i A_i \end{array}$$

where parameter $\mu$ is sequentially decreased to zero and iterates $y_k, Z_k$ are always dual feasible

$X_k \succeq 0$ or $Z_k \succeq 0$ ensured via Newton process:

- large decreases of $\mu$ require damped Newton steps
- small updates allow full (deep) Newton steps

# Primal-dual IP methods for SDP

## Primal-dual methods

$$\begin{array}{ll} \min_{x,y,Z} & \text{trace}\, XZ - \mu \log\det XZ \\ \text{s.t.} & \text{trace}\, A_i X = b_i \\ & Z = C - \sum_i y_i A_i \end{array}$$

Minimizers satisfy optimality conditions

$$\begin{aligned} \text{trace}\, A_i X &= b_i \\ \textstyle\sum_i y_i A_i + Z &= C \\ XZ &= \mu I \\ X, Z &\succeq 0 \end{aligned}$$

Duality gap

$$\text{trace}\, CX - b^T y = \text{trace}\, XZ \succeq 0$$

is minimized along the central path of solutions $X(\mu)$, $y(\mu)$, $Z(\mu)$, a smooth curve parametrized by scalar $\mu$

For this reason, logarithmic barrier methods are also called path-following methods

## Newton step for LMI

For the SDP in LMI form

$$\begin{array}{ll} \text{min} & c^T x \\ \text{s.t.} & F(x) = F_0 + \sum_i x_i F_i \succeq 0 \end{array}$$

the centering problem is

$$\text{min}\, c^T x - \mu \log\det F(x)$$

and at each iteration Newton step $\Delta x$ satisfies the linear system of equations (LSE)

$$H\Delta x = -g$$

where gradient $g$ and Hessian $H$ are given by

$$\begin{aligned} H_{ij} &= \text{trace}\, F(x)^{-1} F_i F(x)^{-1} F_i \\ g_i &= c_i/\mu + \text{trace}\, F(x)^{-1} F_i \end{aligned}$$

LSE typically solved via Cholesky factorization or QR decomposition (near the optimum)

## Newton step for primal-dual methods

For primal-dual IP methods, primal and dual directions $\Delta X$, $\Delta y$ and $\Delta Z$ satisfy non-linear KKT optimality conditions

$$\begin{aligned} \text{trace}\, A_i \Delta X &= 0 \\ \textstyle\sum_i \Delta y_i A_i + \Delta Z &= 0 \\ (X + \Delta X)(Z + \Delta Z) &= \mu I \end{aligned}$$

Key point is in linearizing and symmetrizing the latter equation


Long list of primal-dual search directions, the most known of which is Nesterov-Todd's

Dynamic updates of $\mu$ result in predictor-corrector methods

# Complexity

For the $n$-by-$n$ LMI $F(x) \succeq 0$ with $m$ variables
the flops count of IP methods for SDP is as follows:

For each iteration:
$(a)$ $\mathcal{O}(n^2 m)$ to form $F(x)$
$(b)$ $\mathcal{O}(n^3 m)$ to form $F(x)^{-1} F_i F(x)^{-1} F_j$
$(c)$ $\mathcal{O}(n^2 m^2)$ to form $F(x)^{-1} F_i$
$(d)$ $\mathcal{O}(m^3)$ to solve Newton LSE with Cholesky

Dominating terms are $(b)$ and $(c)$ so the complexity for solving one Newton step is:

$$\mathcal{O}(n^3 m + n^2 m^2)$$

..but structure can be exploited in these steps !

Number of iterations with Newton's method:

$$\mathcal{O}(\sqrt{n} \log \varepsilon^{-1})$$

where $\varepsilon$ is the desired accuracy

In general, it is assumed that $m = \mathcal{O}(n^2)$ otherwise redundant constraints can be removed, so the global worst-case complexity for a dense SDP is

$$\mathcal{O}(n^{6.5} \log \varepsilon^{-1})$$

Much less in practice !

## Primal-dual and barrier methods

In contrast with barrier methods,
in primal-dual methods..

- At each step both primal and dual variables are updated simultaneously
- Search direction obtained from Newton's method applied to modified KKT equations
- Work when problem is not strictly feasible
- Iterates are primal and dual infeasible except when converging

Generally for LP, QP or SDP primal-dual methods outperform barrier methods

## IP methods in general

General characteristics of IP methods:

- Efficiency: about 5 to 50 iterations, almost independent of input data (problem), each iteration is a least-squares problem (well established linear algebra)
- Theory: worst-case analysis of IP methods yields polynomial computational time
- Structure: tailored SDP solvers can exploit problem structure

For more information see the Linear, Cone and SDP section at

www.optimization-online.org

and the Optimization and Control section at

fr.arXiv.org/archive/math

## Penalty/augmented Lagrangian methods

Use similar ideas, but cannot be considered as an interior-point method

When applied to LMI problem

$$\min c^T x \text{ s.t. } F(x) = F_0 + \sum_i x_i F_i \succeq 0$$

- penalty method - some eigenvalues of $F(x)$ can be negative
- barrier method - no eigenvalue of $F(x)$ can be negative (use of logarithm)

Augmented Lagrangian

$$L(x, Z, p) = c^T x + \text{trace } Z\Phi(x, p)$$

with dual variable $Z$ and suitable penalty function, for example

$$\Phi(x, p) = p^2(F(x) + pI)^{-1} - pI$$

with penalty parameter $p$

## Penalty/augmented Lagrangian methods (2)

General algorithm

1. find $x_{k+1}$ such that $\|\nabla_x L(x, Z_k, p_k)\| \leq \epsilon_k$
2. update dual variables: $Z_{k+1} = f(x_{k+1}, Z_k)$
3. update penalty parameter: $p_{k+1} < p_k$
4. go to step 1 until a stopping criterion is satisfied

Can be considered as a primal-dual method, but dual variables are obtained in closed-form at step 2

Complexity for the $n$-by-$n$ LMI $F(x) \succeq 0$ with $m$ variables depends mostly on Newton step 1 in $O(n^3m + n^2m^2)$, same as IP methods

Can be improved to $O(m^2K^2)$ where $K$ is the max number of non-zero terms in the $F_i$

## SDP solvers

Available under the Matlab environment

Primal-dual path-following predictor-corrector algorithms:

- SeDuMi (Sturm)
- SDPT3 (Toh, Tütüncü, Todd)
- CSDP (Borchers)
- SDPA (Kojima and colleagues)

parallel version available

Primal-dual potential reduction:

- MAXDET (Wu, Vandenberghe, Boyd)

explicit max det terms in objective function

Dual-scaling path-following algorithms:

- DSDP (Benson, Ye, Zhang)

exploits structure for combinatorics

Barrier method and augmented Lagrangian:

- PENSDP (Kočvara, Stingl)

## Matrices as variables

Generally, in control problems we do not encounter the LMI in canonical or semidefinite form but rather with matrix variables

Lyapunov's inequality

$$A^T P + PA < 0 \quad P = P^T > 0$$

can be written in canonical form

$$F(x) = F_0 + \sum_{i=1}^{m} F_i x_i > 0$$

with the notations

$$F_0 = 0 \quad F_i = -A^T B_i - B_i A$$

where $B_i$, $i = 1, \ldots, n(n+1)/2$ are matrix bases for symmetric matrices of size $n$

Most software packages for solving LMIs however work with canonical or semidefinite forms, so that a (sometimes time-consuming) pre-processing step is required

## LMI solvers

Available under the Matlab environment

Projective method: project iterate on ellipsoid within PSD cone = least squares problem

- LMI Control Toolbox (Gahinet, Nemirovski)

exploits structure with rank-one linear algebra warm-start + generalized eigenvalues
originally developed for INRIA's Scilab

LMI interface to SDP solvers

- LMITOOL (Nikoukah, Delebecque, El Ghaoui)

for both Scilab and Matlab

- SeDuMi Interface (Peaucelle)
- YALMIP (Löfberg)

See Helmberg's page on SDP
`www-user.tu-chemnitz.de/~helmberg/semidef.html`
and Mittelmann's page on optimization software with benchmarks
`plato.la.asu.edu/guide.html`

## LMI relaxation software

GloptiPoly is written as an open-source, general purpose and user-friendly Matlab software

Optionally, problem definition made easier with Matlab Symbolic Math Toolbox, gateway to Maple kernel

Gloptipoly solves small to medium non-convex global optimization problems with multivariate real-valued polynomial objective functions and constraints

Software and documentation available at

`www.laas.fr/~henrion/software/gloptipoly`

## Methodology

GloptiPoly builds and solves a hierarchy of successive convex linear matrix inequality (LMI) relaxations of increasing size, whose optima are guaranteed to converge asymptotically to the global optimum

Relaxations are build from LMI formulation of sum-of-squares (SOS) decomposition of multivariate polynomials (see last chapter)

In practice convergence is ensured fast, typically at 2nd or 3rd LMI relaxation

## Features

General features of GloptiPoly:

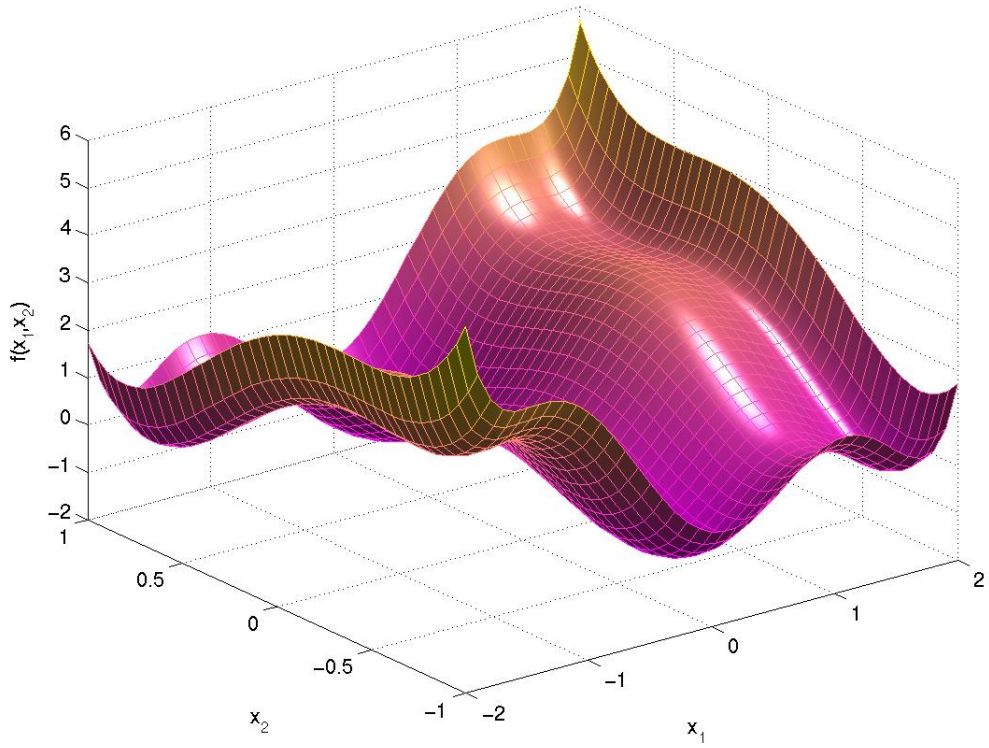
- Certificate of global optimality (rank checks)
- Automatic extraction of globally optimal solutions (multiple eigenvectors)
- 0-1 or $\pm 1$ integer constraints on some of the decision variables (combinatorial optimization problems)
- Generation of input and output data in SeDuMi's format
- Generation of moment matrices associated with LMI relaxations (rank checks)
- User-defined scaling of decision variables (to improve numerical behavior)
- Exploits sparsity of polynomial data

Major update of Gloptipoly planned (hopefully !) for winter 2003

## Benchmark examples
## Continuous problems

Mostly from Floudas/Pardalos 1999 handbook

About 80 % of pbs solved with LMI relaxation of small order (typically 2 or 3) in less than 3 seconds on a PC Pentium IV at 1.6 MHz with 512 Mb RAM

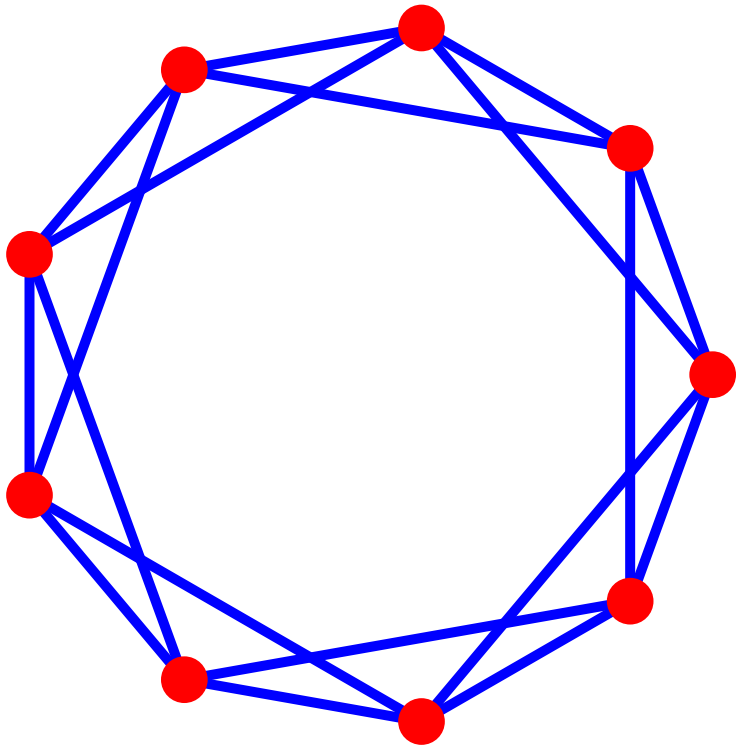



Six-hump camel back function

## Benchmark exmaples
## Discrete problems

From Floudas/Pardalos handbook and also Anjos' Ph.D (Univ Waterloo)

By perturbing criterion (destroys symmetry) global convergence ensured on 80 % of pbs in less than 4 seconds



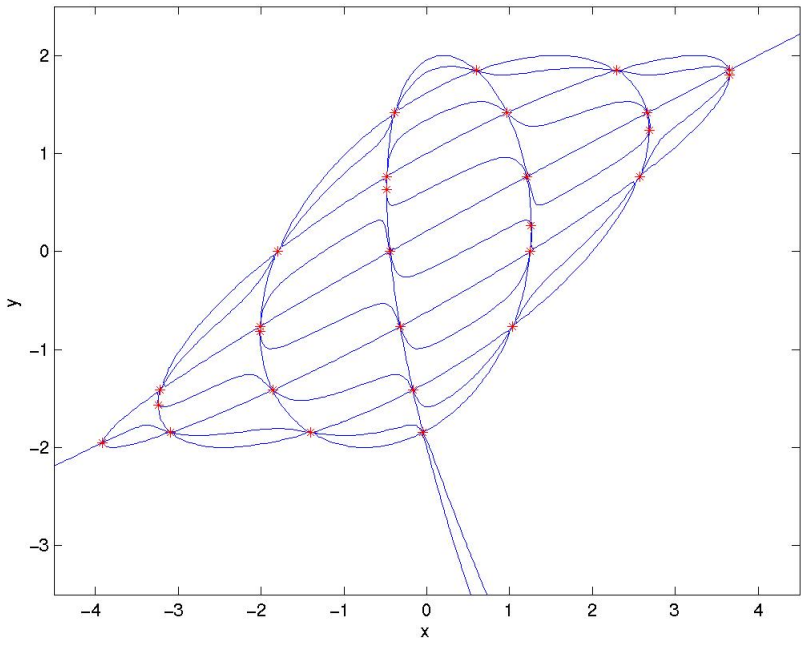
MAXCUT on antiweb $AW_9^2$ graph

# Benchmark examples
# Polynomial systems of equations

From Verschelde's and Posso databasis
Real coefficients & coeffs only

Out of 59 systems:
- 61 % solved in t $<$ 10 secs
- 20 % solved in 10 $<$ t $<$ 100 secs
- 10 % solved in t $\geq$ 100 secs
- 9 % out of memory

No criterion optimized
No enumeration of all solutions




Intersections of seventh and eighth degree polynomial curves

## GloptiPoly: summary

GloptiPoly is a general-purpose software with a user-friendly interface

Pedagogical flavor, black-box approach, no expert tuning required to cope with very distinct applied maths and engineering pbs

Not a competitor to highly specialized codes for solving polynomial systems of equations or large combinatorial optimization pbs

Numerical conditioning (Chebyshev basis) deserves further study

See also the SOSTOOLS software

www.cds.caltech.edu/sostools