Automated Precision Tuning using Semidefinite Programming

Victor Magron, RA Imperial College

joint work with G. Constantinides and A. Donaldson

British-French-German Conference on Optimization
15 June 2015

Errors and Proofs

- Mathematicians and Computer Scientists want to eliminate all the uncertainties on their results. Why?

Errors and Proofs

- Mathematicians and Computer Scientists want to eliminate all the uncertainties on their results. Why?

M. Lecat, Erreurs des Mathématiciens des origines à nos jours, 1935.

$\rightsquigarrow$ 130 pages of errors! (Euler, Fermat, Sylvester, ...)

Errors and Proofs

- Mathematicians and Computer Scientists want to eliminate all the uncertainties on their results. Why?

M. Lecat, Erreurs des Mathématiciens des origines à nos jours, 1935.

$\rightsquigarrow$ 130 pages of errors! (Euler, Fermat, Sylvester, ...)

Ariane 5 launch failure, Pentium FDIV bug

Errors and Proofs

Guaranteed Optimization

Input : Linear problem  (LP), geometric, semidefinite  (SDP)

Output : solution + **certificate**  numeric-symbolic $\rightsquigarrow$  formal

Errors and Proofs

Guaranteed Optimization

Input : Linear problem  (LP), geometric, semidefinite  (SDP)

Output : solution + **certificate**  numeric-symbolic $\rightsquigarrow$  formal

Verification of critical systems

Reliable software/hardware embedded codes

Aerospace control


molecular biology, robotics, code synthesis, . . .

Errors and Proofs

Guaranteed Optimization

Input : Linear problem  (LP), geometric, semidefinite  (SDP)

Output : solution + **certificate**  numeric-symbolic $\rightsquigarrow$  formal

Verification of critical systems

Reliable software/hardware embedded codes

Aerospace control

molecular biology, robotics, code synthesis, . . .

Efficient Verification of Nonlinear Systems

- Automated precision tuning of systems/programs
  analysis/synthesis
- Efficiency sparsity correlation patterns
- Certified approximation algorithms

Rounding Error Bounds

Real : $p(\mathbf{x}) := x_1 \times x_2 + x_3$

Floating-point : $\hat{p}(\mathbf{x}, \mathbf{e}) := [x_1 x_2 (1 + e_1) + x_3](1 + e_2)$

Input variable uncertainties $\mathbf{x} \in \mathbf{S}$
Finite precision $\rightsquigarrow$ bounds over $\mathbf{e}$

$| e_i | \leqslant 2^{-m}$ $\quad m = 24$ (single) or 53 (double)

Guarantees on absolute round-off error $| \hat{p} - p |$?

Nonlinear Programs

- Polynomials programs : $+, -, \times$

$$x_2x_5 + x_3x_6 + x_1(-x_1 + x_2 + x_3 - x_4 + x_5 + x_6)$$

Nonlinear Programs

- Polynomials programs : $+, -, \times$

$$x_2x_5 + x_3x_6 + x_1(-x_1 + x_2 + x_3 - x_4 + x_5 + x_6)$$

- Semialgebraic programs: $|\cdot|, \sqrt{}, /, \sup, \inf$

$$\frac{4x}{1 + \frac{x}{1.11}}$$

Nonlinear Programs

- Polynomials programs : $+, -, \times$

$$x_2x_5 + x_3x_6 + x_1(-x_1 + x_2 + x_3 - x_4 + x_5 + x_6)$$

- Semialgebraic programs: $|\cdot|, \sqrt{}, /, \sup, \inf$

$$\frac{4x}{1 + \frac{x}{1.11}}$$

- Transcendental programs: $\arctan, \exp, \log, \ldots$

$$\log(1 + \exp(x))$$

Existing Frameworks

Classical methods:

- Abstract domains [Goubault-Putot 11]

  FLUCTUAT: intervals, octagons, zonotopes

- Interval arithmetic [Daumas-Melquiond 10]

  GAPPA: interface with COQ proof assistant

Existing Frameworks

Recent progress:

- Affine arithmetic + SMT [Darulova 14]

  `rosa`: sound compiler for reals (in SCALA)

- Symbolic Taylor expansions [Solovyev 15]

`FPTaylor`: certified optimization (in OCAML and HOL-LIGHT)

Contributions

Maximal Rounding error of the program implementation of f:

$$r^\star := \max |\hat{f}(\mathbf{x}, \mathbf{e}) - f(\mathbf{x})|$$

Decomposition: **linear** term l w.r.t. $\mathbf{e}$ + nonlinear term h

$$r^\star \leqslant \max |l(\mathbf{x}, \mathbf{e})| + \max |h(\mathbf{x}, \mathbf{e})|$$

- Sparse SDP bounds for l
- Coarse bound of h with interval arithmetic

Contributions

1. Comparison with SMT and linear/affine arithmetic:
   $\rightsquigarrow$ More **Efficient** optimization $\oplus$ **Tighter** upper bounds
2. Extensions to transcendental/conditional programs
3. Formal verification of SDP bounds 
4. Open source tool `Real2Float` (in OCaml and Coq)

Introduction

Semidefinite Programming for Polynomial Optimization

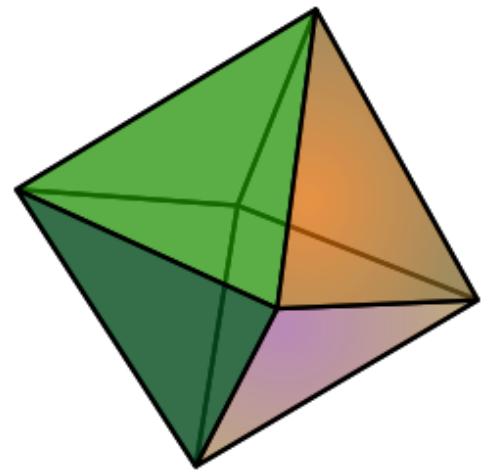
Rounding Error Bounds with Sparse SDP

Conclusion

What is Semidefinite Programming?

- Linear Programming (LP):

$$\min_{\mathbf{z}} \quad \mathbf{c}^\top \mathbf{z}$$
$$\text{s.t.} \quad \mathbf{A}\,\mathbf{z} \geqslant \mathbf{d} \ .$$



Polyhedron

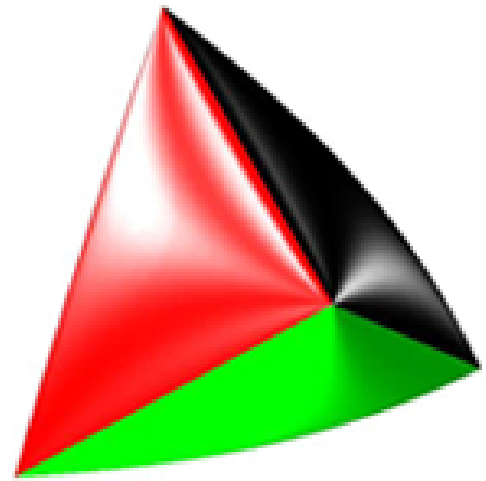
  - Linear cost $\mathbf{c}$
  - Linear inequalities "$\sum_i A_{ij}\, z_j \geqslant d_i$"

What is Semidefinite Programming?

- Semidefinite Programming (SDP):

$$\min_{\mathbf{z}} \quad \mathbf{c}^\top \mathbf{z}$$
$$\text{s.t.} \quad \sum_i \mathbf{F}_i z_i \succcurlyeq \mathbf{F}_0 \ .$$

  - Linear cost $\mathbf{c}$
  - Symmetric matrices $\mathbf{F}_0, \mathbf{F}_i$
  - Linear matrix inequalities "$\mathbf{F} \succcurlyeq 0$" ($\mathbf{F}$ has nonnegative eigenvalues)



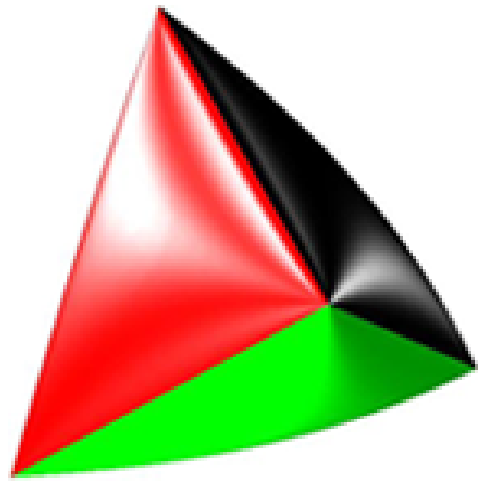
Spectrahedron

What is Semidefinite Programming?

- Semidefinite Programming (SDP):

$$\min_{\mathbf{z}} \quad \mathbf{c}^\top \mathbf{z}$$
$$\text{s.t.} \quad \sum_i \mathbf{F}_i z_i \succcurlyeq \mathbf{F}_0 \ , \quad \mathbf{A}\,\mathbf{z} = \mathbf{d} \ .$$

  - Linear cost $\mathbf{c}$
  - Symmetric matrices $\mathbf{F}_0, \mathbf{F}_i$
  - Linear matrix inequalities "$\mathbf{F} \succcurlyeq 0$" ($\mathbf{F}$ has nonnegative eigenvalues)



Spectrahedron

Applications of SDP

- Combinatorial optimization
- Control theory
- Matrix completion
- Unique Games Conjecture (Khot '02) :
  "A *single concrete algorithm* provides **optimal guarantees** among all efficient algorithms for a large class of computational problems."
  (Barak and Steurer survey at ICM'14)
- Solving polynomial optimization (Lasserre '01)

SDP for Polynomial Optimization

- Prove **polynomial inequalities** with SDP:

$$p(a,b) := a^2 - 2ab + b^2 \geqslant 0 \ .$$

- Find $\mathbf{z}$ s.t. $p(a,b) = \begin{pmatrix} a & b \end{pmatrix} \underbrace{\begin{pmatrix} z_1 & z_2 \\ z_2 & z_3 \end{pmatrix}}_{\succcurlyeq 0} \begin{pmatrix} a \\ b \end{pmatrix}$.

- Find $\mathbf{z}$ s.t. $a^2 - 2ab + b^2 = z_1 a^2 + 2 z_2 ab + z_3 b^2 \qquad (\mathbf{A}\,\mathbf{z} = \mathbf{d})$

- $\begin{pmatrix} z_1 & z_2 \\ z_2 & z_3 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}}_{\mathbf{F}_1} z_1 + \underbrace{\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}}_{\mathbf{F}_2} z_2 + \underbrace{\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}}_{\mathbf{F}_3} z_3 \succcurlyeq \underbrace{\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}}_{\mathbf{F}_0}$

SDP for Polynomial Optimization

- Choose a cost $\mathbf{c}$ e.g. $(1,0,1)$ and solve:

$$\begin{aligned} \min_{\mathbf{z}} \quad & \mathbf{c}^\top \mathbf{z} \\ \text{s.t.} \quad & \sum_i \mathbf{F}_i z_i \succcurlyeq \mathbf{F}_0 \ , \quad \mathbf{A}\,\mathbf{z} = \mathbf{d} \ . \end{aligned}$$

- Solution $\begin{pmatrix} z_1 & z_2 \\ z_2 & z_3 \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \succcurlyeq 0$ (eigenvalues 0 and 1)

- $a^2 - 2ab + b^2 = \begin{pmatrix} a & b \end{pmatrix} \underbrace{\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}}_{\succcurlyeq 0} \begin{pmatrix} a \\ b \end{pmatrix} = (a-b)^2 \, .$

- Solving **SDP** $\Longrightarrow$ Finding **SUMS OF SQUARES** certificates

SDP for Polynomial Optimization

General case:

- Semialgebraic set $\mathbf{S} := \{\mathbf{x} \in \mathbb{R}^n : g_1(\mathbf{x}) \geqslant 0, \ldots, g_m(\mathbf{x}) \geqslant 0\}$
- $p^* := \min_{\mathbf{x} \in \mathbf{S}} p(\mathbf{x})$: NP hard
- Sums of squares (SOS) $\Sigma[\mathbf{x}]$ (e.g. $(x_1 - x_2)^2$)
- $\mathcal{Q}(\mathbf{S}) := \left\{ \sigma_0(\mathbf{x}) + \sum_{j=1}^m \sigma_j(\mathbf{x}) g_j(\mathbf{x}), \text{ with } \sigma_j \in \Sigma[\mathbf{x}] \right\}$
- Fix the degree $2k$ of sums of squares
  $\mathcal{Q}_k(\mathbf{S}) := \mathcal{Q}(\mathbf{S}) \cap \mathbb{R}_{2k}[\mathbf{x}]$

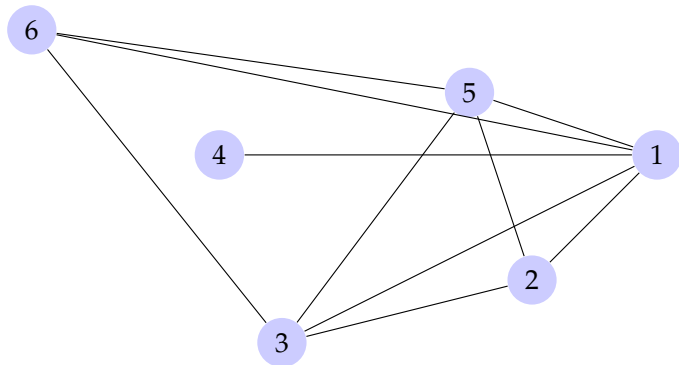
SDP for Polynomial Optimization

- **Hierarchy of SDP relaxations**:
  $\lambda_k := \sup_{\lambda} \left\{ \lambda : p - \lambda \in \mathcal{Q}_k(\mathbf{S}) \right\}$
- Convergence guarantees $\lambda_k \uparrow p^*$ [Lasserre 01]
- Can be computed with SDP solvers (CSDP, SDPA)
- **"No Free Lunch" Rule**: $\binom{n+2k}{n}$ SDP variables
- Extension to semialgebraic functions $r(\mathbf{x}) = p(\mathbf{x}) / \sqrt{q(\mathbf{x})}$ [Lasserre-Putinar 10]

Sparse SDP Optimization [Waki, Lasserre 06]

- Correlative sparsity pattern (csp) of variables

$$x_2x_5 + x_3x_6 - x_2x_3 - x_5x_6 + x_1(-x_1 + x_2 + x_3 - x_4 + x_5 + x_6)$$



1. Maximal cliques $C_1, \ldots, C_l$
2. Average size $\kappa \rightsquigarrow \binom{\kappa+2k}{\kappa}$ variables

$C_1 := \{1, 4\}$
$C_2 := \{1, 2, 3, 5\}$
$C_3 := \{1, 3, 5, 6\}$
Dense SDP: 210 variables
Sparse SDP: 115 variables

Introduction

Semidefinite Programming for Polynomial Optimization

Rounding Error Bounds with Sparse SDP

Conclusion

Polynomial Programs

Input: exact $f(\mathbf{x})$, floating-point $\hat{f}(\mathbf{x}, \mathbf{e})$, $\mathbf{x} \in \mathbf{S}$, $\mid e_i \mid \leqslant 2^{-m}$
Output: Bound for $f - \hat{f}$

1: Error $r(\mathbf{x}, \mathbf{e}) := f(\mathbf{x}) - \hat{f}(\mathbf{x}, \mathbf{e}) = \sum_{\alpha} r_\alpha(\mathbf{e}) \mathbf{x}^\alpha$

2: Decompose $r(\mathbf{x}, \mathbf{e}) = l(\mathbf{x}, \mathbf{e}) + h(\mathbf{x}, \mathbf{e})$, l **linear** in $\mathbf{e}$

3: $l(\mathbf{x}, \mathbf{e}) = \sum_{i=0}^{n'} s_i(\mathbf{x}) e_i$

4: Maximal cliques correspond to $\{\mathbf{x}, e_1\}, \ldots, \{\mathbf{x}, e_{n'}\}$

5: Bound $l(\mathbf{x}, \mathbf{e})$ with sparse SDP relaxations (and h with IA)

Dense relaxation: $\binom{n+n'+2k}{n+n'}$ SDP variables
Sparse relaxation: $n' \binom{n+1+2k}{n+1}$ SDP variables

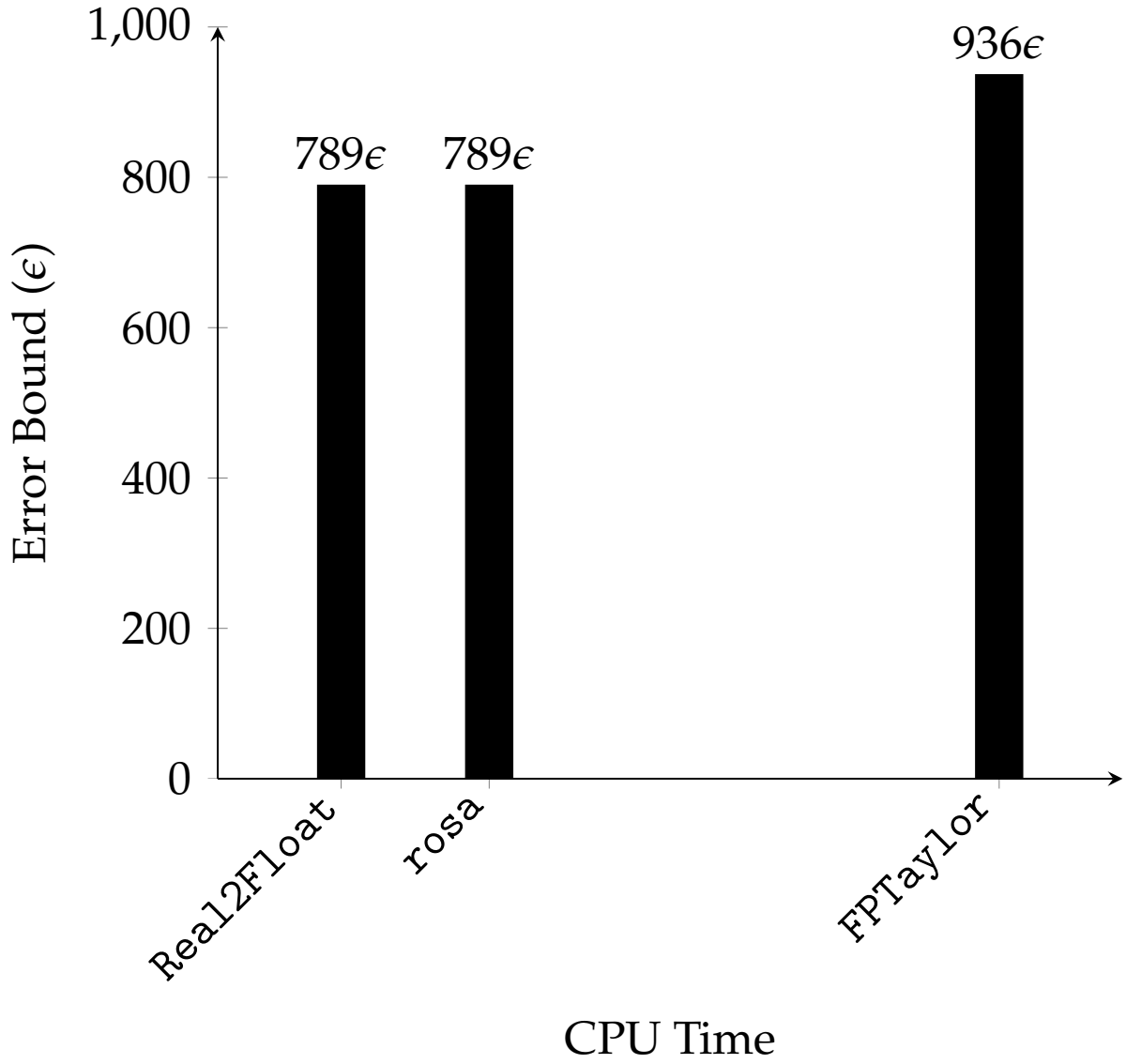
Preliminary Comparisons

$$f(\mathbf{x}) := x_2x_5 + x_3x_6 - x_2x_3 - x_5x_6 + x_1(-x_1 + x_2 + x_3 - x_4 + x_5 + x_6)$$

$$\mathbf{x} \in [4.00, 6.36]^6, \quad \mathbf{e} \in [-\epsilon, \epsilon]^{15}, \quad \epsilon = 2^{-24}$$

- Dense SDP: $\binom{6+15+4}{6+15} = 12650$ variables $\rightsquigarrow$ **Out of memory**
- Sparse SDP `Real2Float` tool: $15\binom{6+1+4}{6+1} = 4950 \rightsquigarrow 789\epsilon$
- Interval arithmetic: 2023ϵ ($17\times$ less CPU)
- Symbolic Taylor `FPTaylor` tool: 936ϵ ($16.3\times$ more CPU)
- SMT-based `rosa` tool: 789ϵ ($4.6\times$ more CPU)

Preliminary Comparisons



Extensions: Transcendental Programs

Given $\mathbf{K}$ a compact set and f a transcendental function, bound $f^* = \inf_{\mathbf{x}\in\mathbf{K}} f(\mathbf{x})$ and prove $f^* \geqslant 0$

- f is under-approximated by a semialgebraic function f_{sa}
- Reduce the problem $f^*_{\text{sa}} := \inf_{\mathbf{x}\in\mathbf{K}} f_{\text{sa}}(\mathbf{x})$ to a polynomial optimization problem (POP)

Extensions: Transcendental Programs

Approximation theory: Chebyshev/Taylor models

- mandatory for non-polynomial problems
- hard to combine with Sum-of-Squares techniques (degree of approximation)

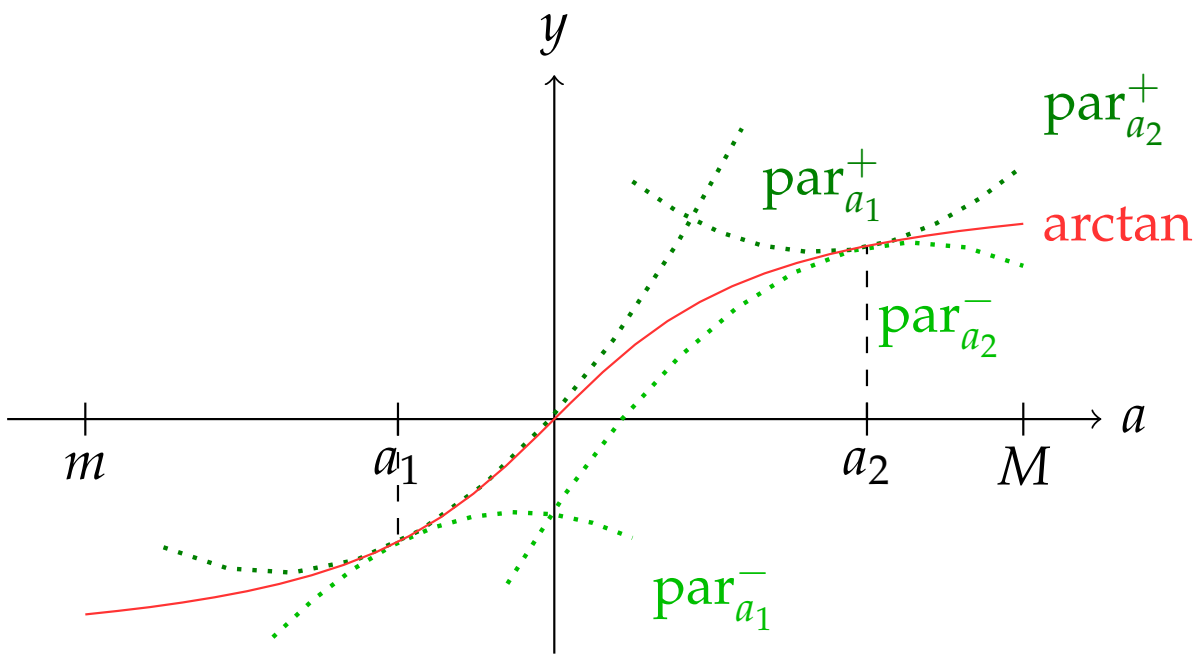
Extensions: Transcendental Programs

- Initially introduced to solve Optimal Control Problems [Fleming-McEneaney 00]
- **Curse of dimensionality** reduction [McEaneney Kluberg, Gaubert-McEneaney-Qu 11, Qu 13].
  Allowed to solve instances of dim up to 15 (inaccessible by grid methods)
- In our context: approximate transcendental functions

Extensions: Transcendental Programs

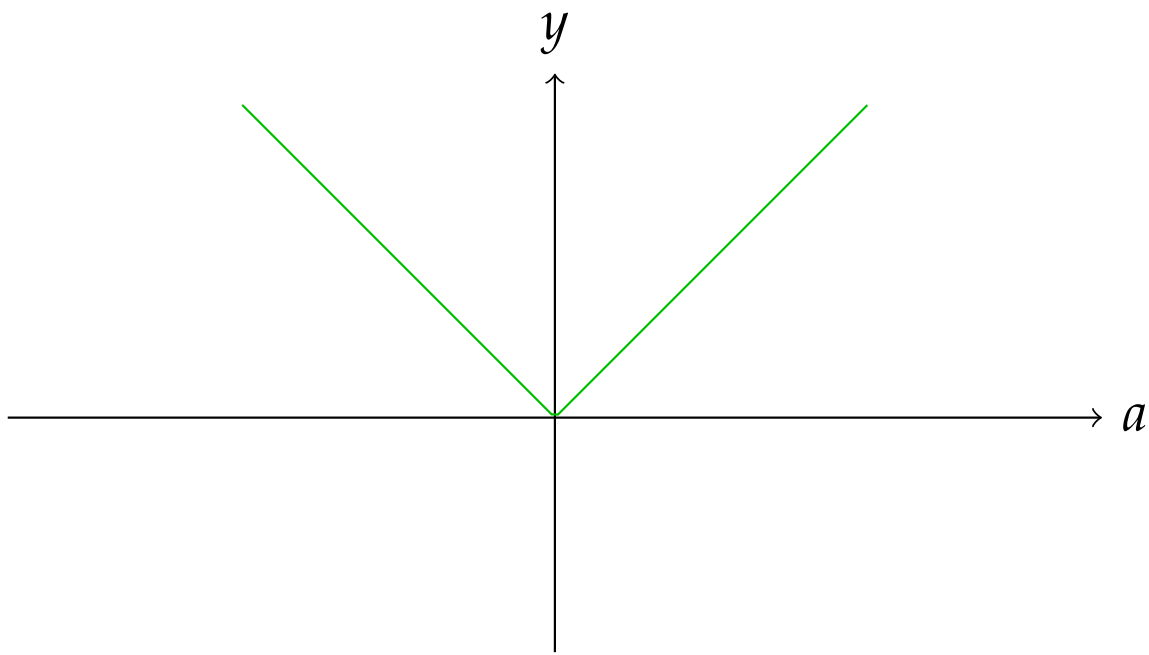
Definition

Let $\gamma \geqslant 0$. A function $\phi : \mathbb{R}^n \to \mathbb{R}$ is said to be γ-*semiconvex* if the function $\mathbf{x} \mapsto \phi(\mathbf{x}) + \frac{\gamma}{2}\|\mathbf{x}\|_2^2$ is convex.



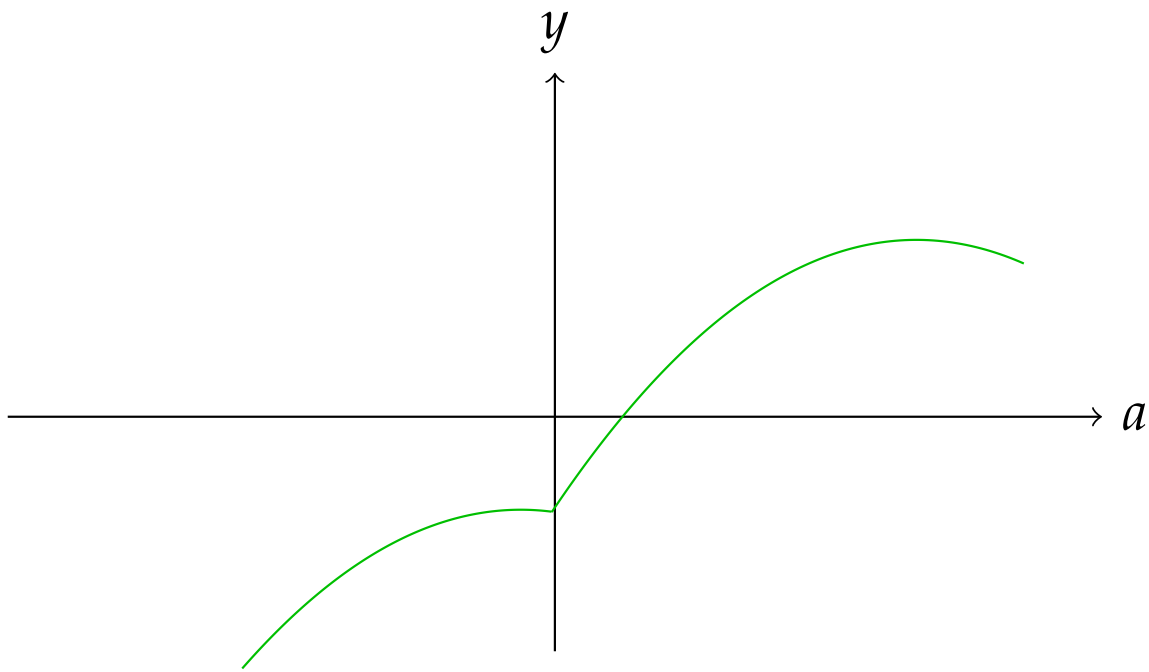
Extensions: Transcendental Programs

Exact parsimonious maxplus representations



Extensions: Transcendental Programs

Exact parsimonious maxplus representations



Extensions: Programs with Conditionals

`if` $(p(\mathbf{x}) \leqslant 0)$ $f(\mathbf{x})$; `else` $g(\mathbf{x})$;

DIVERGENCE PATH ERROR:

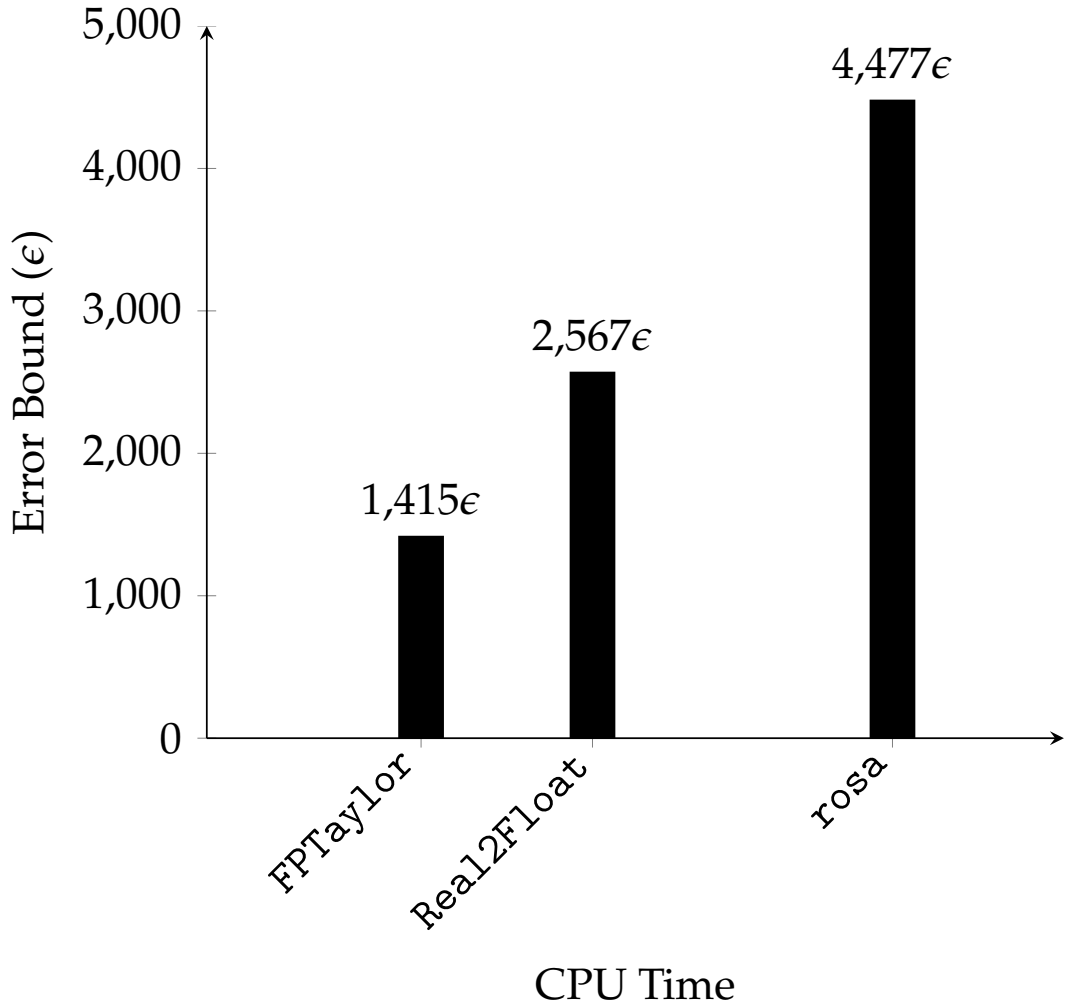
$$
\begin{aligned}
r^\star := \max\{ & \\
& \max_{p(\mathbf{x})\leqslant 0, p(\mathbf{x},\mathbf{e})\geqslant 0} \mid \hat{f}(\mathbf{x},\mathbf{e}) - g(\mathbf{x}) \mid \\
& \max_{p(\mathbf{x})\geqslant 0, p(\mathbf{x},\mathbf{e})\leqslant 0} \mid \hat{g}(\mathbf{x},\mathbf{e}) - f(\mathbf{x}) \mid \\
& \max_{p(\mathbf{x})\geqslant 0, p(\mathbf{x},\mathbf{e})\geqslant 0} \mid \hat{f}(\mathbf{x},\mathbf{e}) - f(\mathbf{x}) \mid \\
& \max_{p(\mathbf{x})\leqslant 0, p(\mathbf{x},\mathbf{e})\leqslant 0} \mid \hat{g}(\mathbf{x},\mathbf{e}) - g(\mathbf{x}) \mid \\
\} &
\end{aligned}
$$

Benchmarks

Doppler

$u \in [-100, 100]$
$v \in [20, 20000]$
$T \in [-30, 50]$

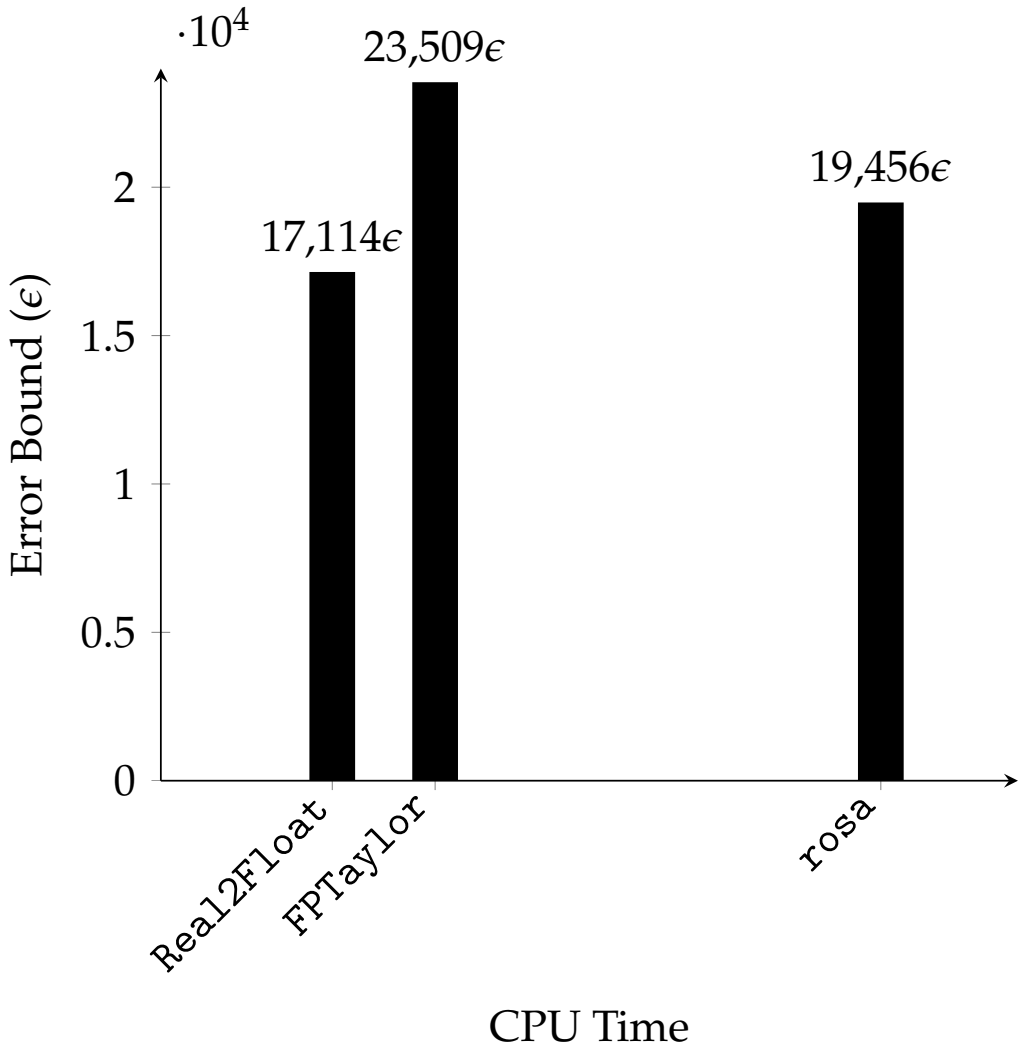
$$\texttt{let}\ \ t_1 = 331.4 + 0.6T\ \ \texttt{in}\ \ \frac{-t_1 v}{(t_1 + u)^2}$$



Benchmarks

Kepler2

$$\mathbf{x} \in [4, 6.36]^6$$
$$x_1x_4(-x_1 + x_2 + x_3 - x_4 + x_5 + x_6) + x_2x_5(x_1 - x_2 + x_3 + x_4 - x_5 + x_6) + x_3x_6(x_1 + x_2 - x_3 + x_4 + x_5 - x_6) - x_2x_3x_4 - x_1x_3x_5 - x_1x_2x_6 - x_4x_5x_6$$

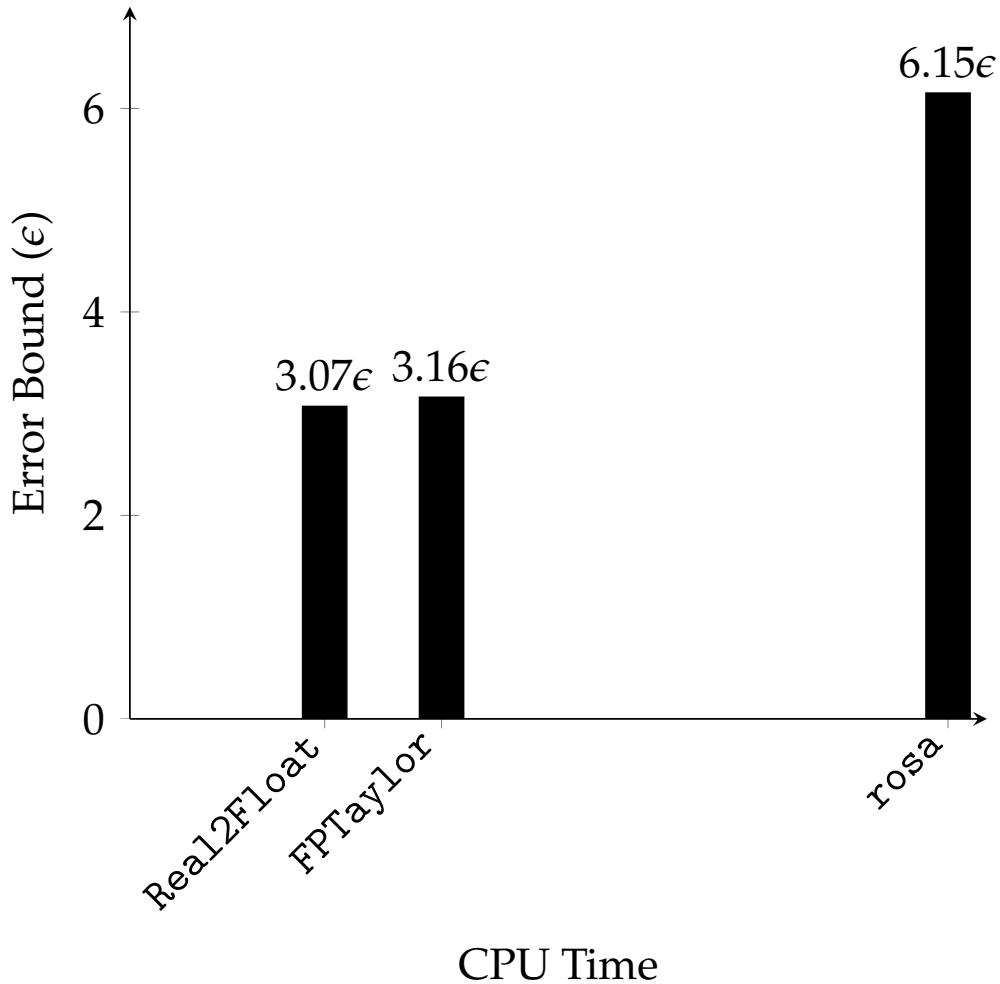


Benchmarks

verhulst

$x \in [0.1, 0.3]$

$$\frac{4x}{1 + \frac{x}{1.11}}$$

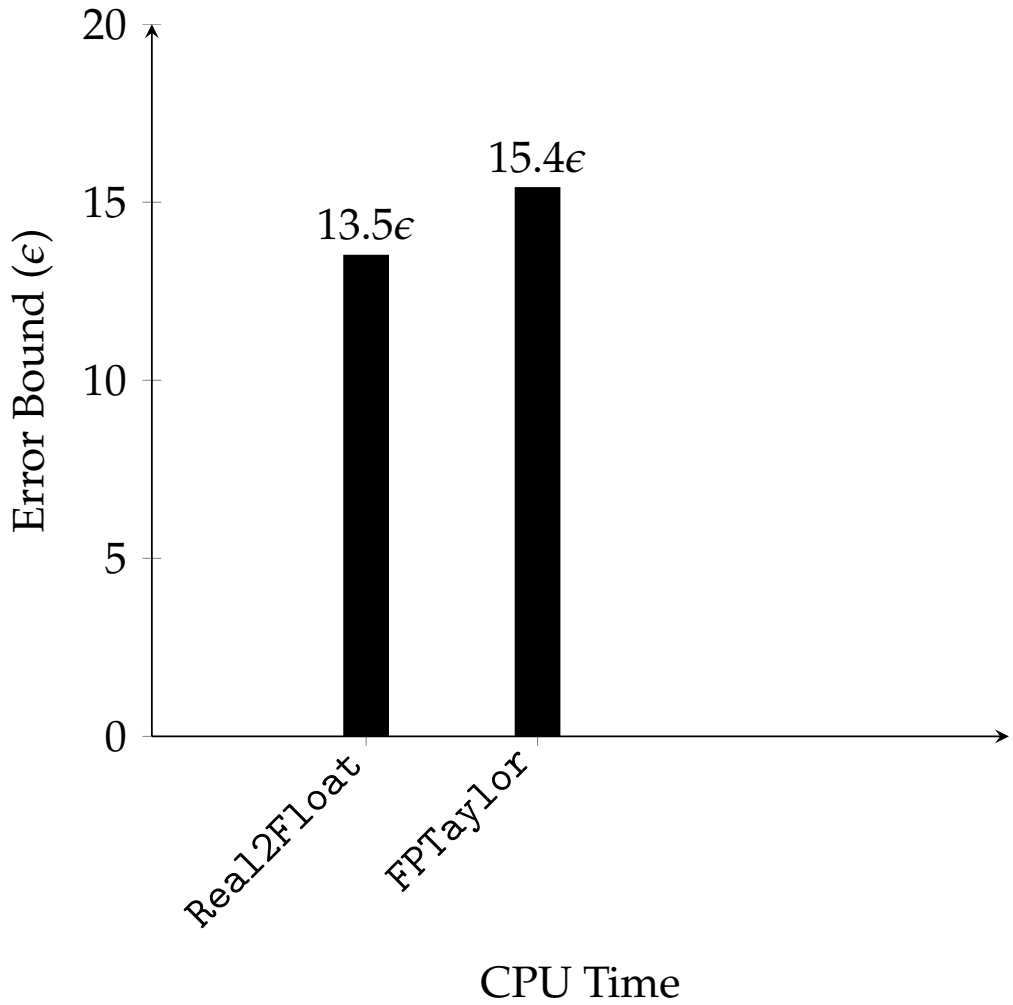


Benchmarks

logexp

$x \in [-8, 8]$

$\log(1 + \exp(x))$



Introduction

Semidefinite Programming for Polynomial Optimization

Rounding Error Bounds with Sparse SDP

Conclusion

Conclusion

Sparse SDP relaxations analyze **NONLINEAR PROGRAMS**:

- Polynomial and transcendental programs
- Handles conditionals, input uncertainties, ...
- Certified  $\leadsto$ Formal  rounding error bounds
- `Real2Float` open source tool:

  `https://forge.ocamlcore.org/projects/nl-certify`

Conclusion

Further research:

- Improve formal polynomial checker 
- Alternative polynomial bounds using geometric programming (T. de Wolff, S. Iliman)
- Mixed linear/SDP certificates (trade-off CPU/precision)
- More program verification: while loops
- Automatic **FPGA** code generation 

End

Thank you for your attention!

cas.ee.ic.ac.uk/people/vmagron