Formal Verification of Roundoff Error Bounds using Semidefinite Programming

Victor Magron, CNRS VERIMAG

Jointly Certified Upper Bounds with **G. Constantinides** and **A. Donaldson**
Jointly Certified Lower Bounds with **M. Farid**

GTVerif
IRIF, 16 June 2016

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, ...)

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, ...)

Ariane 5 launch failure, Pentium FDIV bug

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, ...)

Ariane 5 launch failure, Pentium FDIV bug

- U.S. Patriot missile killed 28 soldiers from the U.S. Army's
- Internal clock: 0.1 sec intervals
- Roundoff error on the binary constant "0.1"

Roundoff Error Bounds

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

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

Input variable constraints $\mathbf{x} \in \mathbf{X}$
Finite precision $\rightsquigarrow$ bounds over $\mathbf{e} \in \mathbf{E}$: $| e_i | \leqslant 2^{-53}$ (double)
Guarantees on absolute round-off error $| \hat{f} - f |$?

$$\max \hat{f} - f \quad \begin{array}{c} \downarrow \textbf{Upper} \text{ Bounds} \downarrow \\ \hline \uparrow \textbf{Lower} \text{ Bounds} \uparrow \end{array} \quad \max \hat{f} - f$$

$$\min \hat{f} - f \quad \begin{array}{c} \downarrow \textbf{Lower} \text{ Bounds} \downarrow \\ \hline \uparrow \textbf{Upper} \text{ Bounds} \uparrow \end{array} \quad \min \hat{f} - f$$

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 (SCALA)

- Symbolic Taylor expansions [Solovyev 15]

  `FPTaylor`: certified optimization (OCAML/HOL-LIGHT)

- Guided random testing `s3fp` [Chiang 14]

Contributions

Maximal Roundoff error of the program implementation of f:

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

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

$$\max |l| + \max |h| \geqslant r^\star \geqslant \max |l| - \max |h|$$

- Coarse bound of h with interval arithmetic
- **Semidefinite programming** (SDP) bounds for l:

Contributions

Maximal Roundoff error of the program implementation of f:

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

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

$$\max |l| + \max |h| \geqslant r^\star \geqslant \max |l| - \max |h|$$

- Coarse bound of h with interval arithmetic
- **Semidefinite programming** (SDP) bounds for l:

$\downarrow$ **Upper** Bounds $\downarrow$

$\uparrow$ **Lower** Bounds $\uparrow$
$\downarrow$ **Lower** Bounds $\downarrow$

$\uparrow$ **Upper** Bounds $\uparrow$

Sparse SDP relaxations **Robust SDP relaxations**

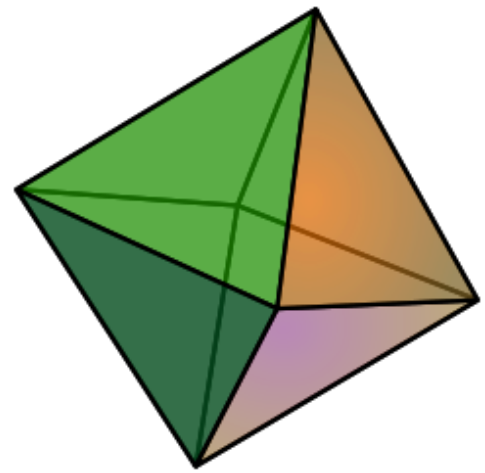
Contributions

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

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

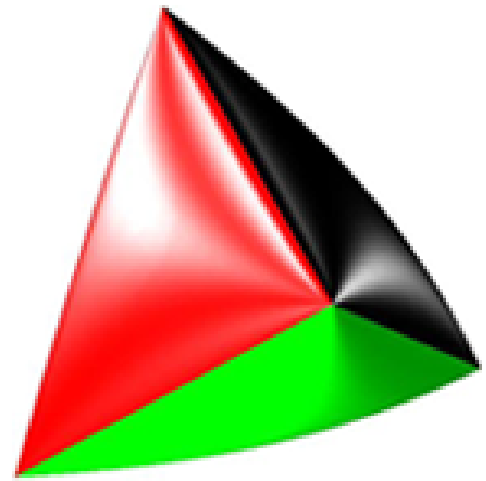
  - Linear cost $\mathbf{c}$
  - Linear inequalities "$\sum_i A_{ij}\, z_j \geqslant d_i$"

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)



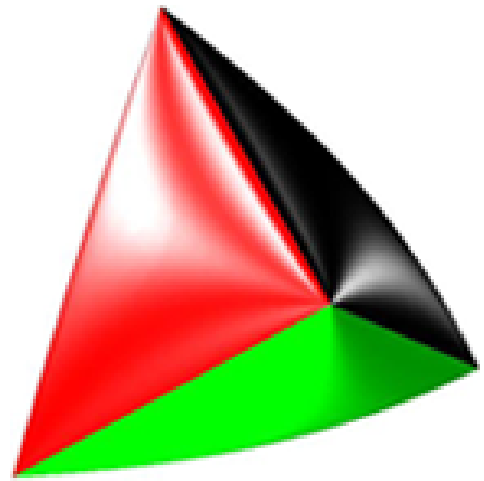
Spectrahedron

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
- Solving polynomial optimization (Lasserre '01)

SDP for Polynomial Optimization

- Prove **polynomial inequalities** with SDP:

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

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

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

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

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 2)

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

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

SDP for Polynomial Optimization

Hierarchy of SDP relaxations:

$$f^* := \min_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) \geqslant \lambda_k := \sup_{\lambda} \left\{ \lambda : f - \lambda \text{ SOS of degree 2k} \right\}$$

SDP for Polynomial Optimization

Hierarchy of SDP relaxations:

$$f^* := \min_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) \geqslant \lambda_k := \sup_{\lambda} \Big\{ \lambda : f - \lambda \text{ SOS of degree 2k} \Big\}$$

Theorem [Lasserre 01]

$\lambda_k \uparrow f^*$

SDP for Polynomial Optimization

Hierarchy of SDP relaxations:

$$f^* := \min_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) \geqslant \lambda_k := \sup_{\lambda} \left\{ \lambda : f - \lambda \text{ SOS of degree 2k} \right\}$$

Theorem [Lasserre 01]

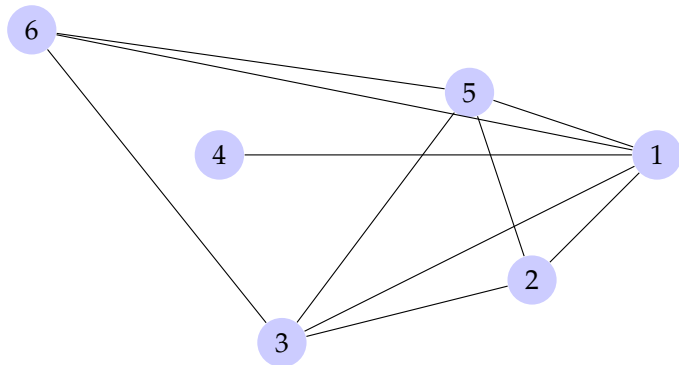
$\lambda_k \uparrow f^*$

"No Free Lunch" Rule: $\binom{n+2k}{n}$ SDP variables

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

Polynomial Programs

Input: exact $f(\mathbf{x})$, floating-point $\hat{f}(\mathbf{x}, \mathbf{e})$, $\mathbf{x} \in \mathbf{X}$, $\mid e_i \mid \leqslant 2^{-53}$
Output: Bound for $f - \hat{f}$

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

2: $l(\mathbf{x}, \mathbf{e}) = s_1(\mathbf{x})e_1 + \cdots + s_m(\mathbf{x})e_m$

3: Maximal cliques correspond to $\{\mathbf{x}, e_1\}, \ldots, \{\mathbf{x}, e_m\}$

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

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

Preliminary Comparisons

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

$$\mathbf{x} \in [4.00, 6.36]^6, \quad \mathbf{e} \in [-\epsilon, \epsilon]^{15}, \quad \epsilon = 2^{-53}$$

- Dense SDP: $\binom{6+15+4}{6+15} = 12650$ variables $\rightsquigarrow$ **Out of memory**

Preliminary Comparisons

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

$$\mathbf{x} \in [4.00, 6.36]^6, \quad \mathbf{e} \in [-\epsilon, \epsilon]^{15}, \quad \epsilon = 2^{-53}$$

- Dense SDP: $\binom{6+15+4}{6+15} = 12650$ variables $\rightsquigarrow$ **Out of memory**
- Sparse SDP `Real2Float` tool: $15\binom{6+1+4}{6+1} = 4950 \rightsquigarrow 759\epsilon$

Preliminary Comparisons

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

$$\mathbf{x} \in [4.00, 6.36]^6, \quad \mathbf{e} \in [-\epsilon, \epsilon]^{15}, \quad \epsilon = 2^{-53}$$

- Dense SDP: $\binom{6+15+4}{6+15} = 12650$ variables $\rightsquigarrow$ **Out of memory**
- Sparse SDP `Real2Float` tool: $15\binom{6+1+4}{6+1} = 4950 \rightsquigarrow 759\epsilon$
- Interval arithmetic: 922ϵ ($10\times$ less CPU)

Preliminary Comparisons

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

$$\mathbf{x} \in [4.00, 6.36]^6, \quad \mathbf{e} \in [-\epsilon, \epsilon]^{15}, \quad \epsilon = 2^{-53}$$

- Dense SDP: $\binom{6+15+4}{6+15} = 12650$ variables $\rightsquigarrow$ **Out of memory**
- Sparse SDP `Real2Float` tool: $15\binom{6+1+4}{6+1} = 4950 \rightsquigarrow 759\epsilon$
- Interval arithmetic: 922ϵ ($10\times$ less CPU)
- Symbolic Taylor `FPTaylor` tool: 721ϵ ($21\times$ more CPU)

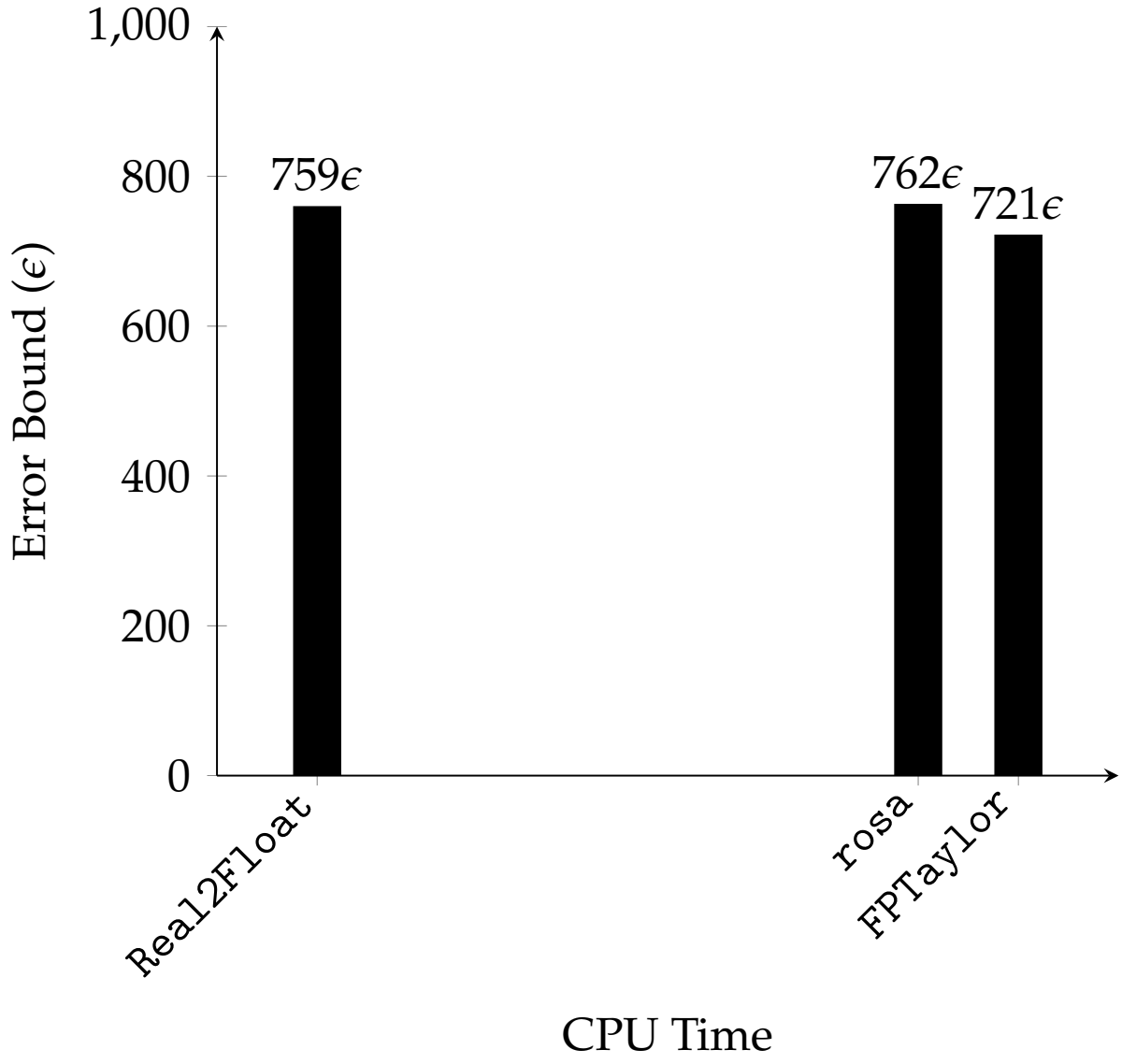
Preliminary Comparisons

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

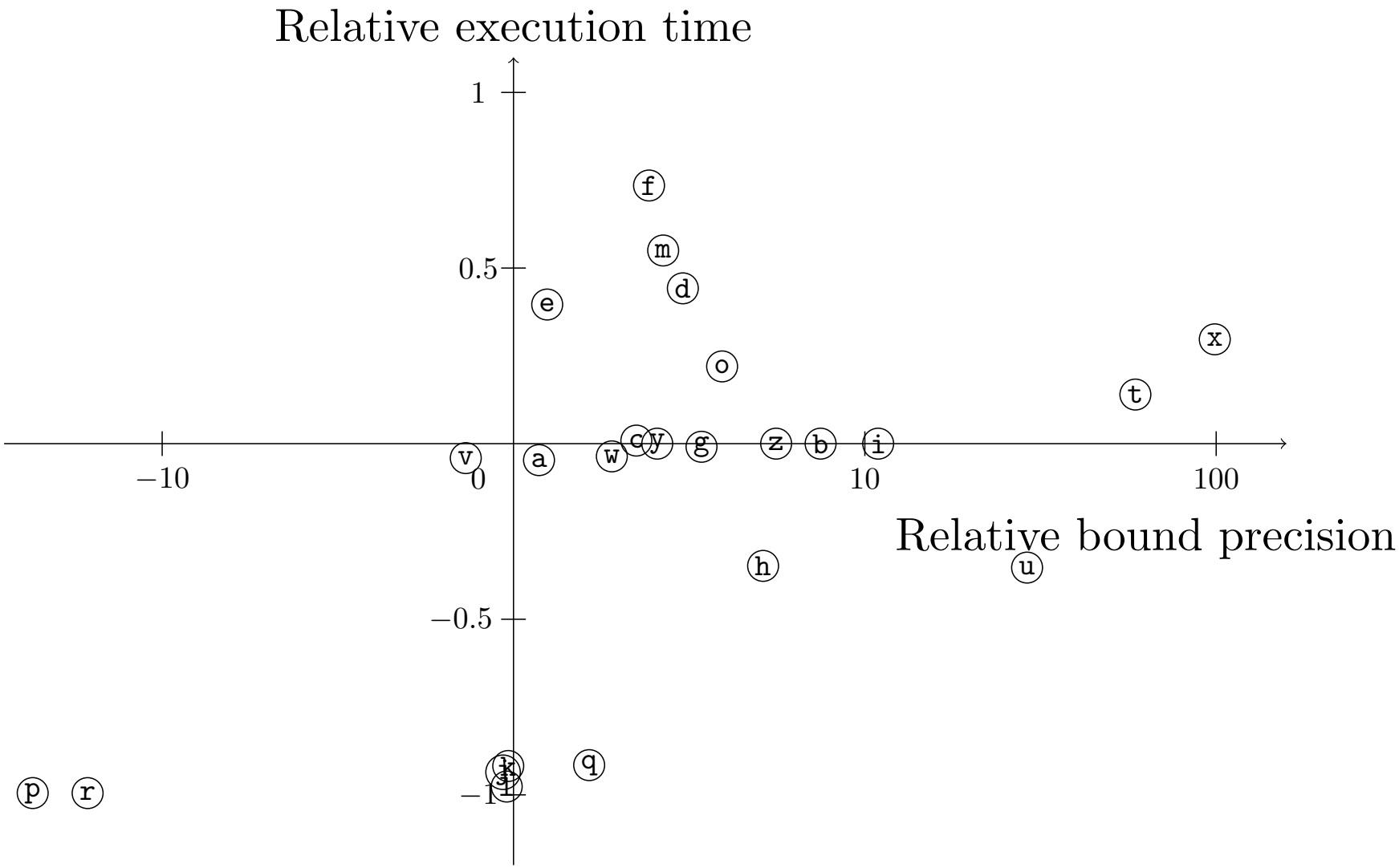
$$\mathbf{x} \in [4.00, 6.36]^6, \quad \mathbf{e} \in [-\epsilon, \epsilon]^{15}, \quad \epsilon = 2^{-53}$$

- Dense SDP: $\binom{6+15+4}{6+15} = 12650$ variables $\rightsquigarrow$ **Out of memory**
- Sparse SDP `Real2Float` tool: $15\binom{6+1+4}{6+1} = 4950 \rightsquigarrow 759\epsilon$
- Interval arithmetic: 922ϵ ($10\times$ less CPU)
- Symbolic Taylor `FPTaylor` tool: 721ϵ ($21\times$ more CPU)
- SMT-based `rosa` tool: 762ϵ ($19\times$ more CPU)

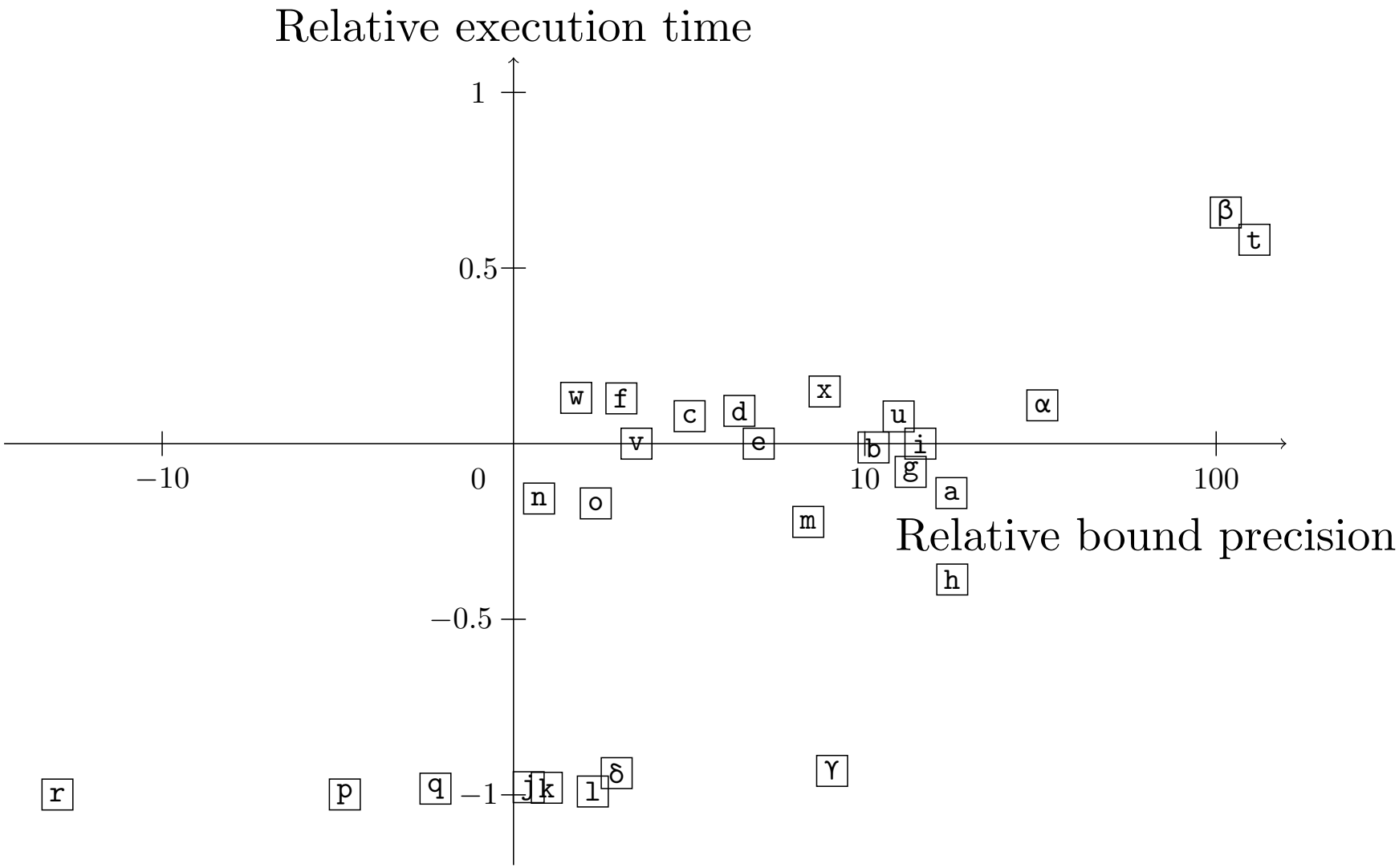
Preliminary Comparisons



Comparison with `rosa`



Comparison with `FPTaylor`



Method 1: `geneig` [Lasserre 11]

Generalized eigenvalue problem:

$$f^* := \min_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) \leqslant \lambda_k := \sup_{\lambda} \quad \lambda$$
$$\text{s.t.} \quad \mathbf{M}_k(f\,\mathbf{y}) \succcurlyeq \lambda \mathbf{M}_k(\mathbf{y}).$$

Method 1: `geneig` [Lasserre 11]

Generalized eigenvalue problem:

$$f^* := \min_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) \leqslant \lambda_k := \sup_{\lambda} \quad \lambda$$
$$\text{s.t.} \quad \mathbf{M}_k(f\,\mathbf{y}) \succcurlyeq \lambda \mathbf{M}_k(\mathbf{y}).$$

Uniform distribution moments: $\mathbf{y}_\alpha := \int_{\mathbf{X}} \mathbf{x}^\alpha d\mathbf{x}$
Localizing matrices $\mathbf{M}_k(f\,\mathbf{y})$:

$$\mathbf{M}_1(f\,\mathbf{y}) = \begin{array}{c} \\ 1 \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & x_1 & x_2 \end{array} \\ \begin{pmatrix} \int_{\mathbf{X}} f(\mathbf{x})d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_1 d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_2 d\mathbf{x} \\ \int_{\mathbf{X}} f(\mathbf{x})x_1 d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_1^2 d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_1x_2 d\mathbf{x} \\ \int_{\mathbf{X}} f(\mathbf{x})x_2 d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_2x_1 d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_2^2 d\mathbf{x} \end{pmatrix} \end{array}$$

Method 1: `geneig` [Lasserre 11]

Generalized eigenvalue problem:

$$f^* := \min_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) \leqslant \lambda_k := \sup_{\lambda} \quad \lambda$$
$$\text{s.t.} \quad \mathbf{M}_k(f\,\mathbf{y}) \succcurlyeq \lambda \mathbf{M}_k(\mathbf{y}).$$

Uniform distribution moments: $\mathbf{y}_\alpha := \int_{\mathbf{X}} \mathbf{x}^\alpha d\mathbf{x}$
Localizing matrices $\mathbf{M}_k(f\,\mathbf{y})$:

$$\mathbf{M}_1(f\,\mathbf{y}) = \begin{array}{c} \\ 1 \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & x_1 & x_2 \end{array} \\ \begin{pmatrix} \int_{\mathbf{X}} f(\mathbf{x})d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_1 d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_2 d\mathbf{x} \\ \int_{\mathbf{X}} f(\mathbf{x})x_1 d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_1^2 d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_1x_2 d\mathbf{x} \\ \int_{\mathbf{X}} f(\mathbf{x})x_2 d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_2x_1 d\mathbf{x} & \int_{\mathbf{X}} f(\mathbf{x})x_2^2 d\mathbf{x} \end{pmatrix} \end{array}$$

Theorem [Lasserre 11]

$\lambda_k \downarrow f^*$

Method 2: `mvbeta` [DeKlerk et al. 16]

Elementary calculation with $f(\mathbf{x}) = \sum_{\alpha} f_\alpha \mathbf{x}^\alpha$:

$$f^* := \min_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) \leqslant f_k^H := \min_{|\eta+\beta|\leqslant 2k} \sum_{\alpha} f_\alpha \frac{\gamma_{\eta+\alpha,\beta}}{\gamma_{\eta,\beta}}$$

Method 2: `mvbeta` [DeKlerk et al. 16]

Elementary calculation with $f(\mathbf{x}) = \sum_\alpha f_\alpha \mathbf{x}^\alpha$:

$$f^* := \min_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) \leqslant f_k^H := \min_{|\eta+\beta|\leqslant 2k} \sum_\alpha f_\alpha \frac{\gamma_{\eta+\alpha,\beta}}{\gamma_{\eta,\beta}}$$

Multivariate beta distribution moments:

$$\gamma_{\eta,\beta} := \int_{\mathbf{X}} \mathbf{x}^\eta (1-\mathbf{x})^\beta d\mathbf{x}\,.$$

Method 2: `mvbeta` [DeKlerk et al. 16]

↑ **Lower** Bounds ↑
↓ **Lower** Bounds ↓

Elementary calculation with $f(\mathbf{x}) = \sum_\alpha f_\alpha \mathbf{x}^\alpha$:

$$f^* := \min_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) \leqslant f_k^H := \min_{|\eta+\beta|\leqslant 2k} \sum_\alpha f_\alpha \frac{\gamma_{\eta+\alpha,\beta}}{\gamma_{\eta,\beta}}$$

Multivariate beta distribution moments:

$$\gamma_{\eta,\beta} := \int_{\mathbf{X}} \mathbf{x}^\eta (1-\mathbf{x})^\beta d\mathbf{x}\,.$$

Theorem [DeKlerk et al. 16]

$f_k^H \downarrow f^*$

Method 3: `robustsdp`

Robust SDP with $l(\mathbf{x}, \mathbf{e}) = \sum_{i=1}^{m} s_i(\mathbf{x}) e_i$:

$$l^* := \min_{(\mathbf{x},\mathbf{e}) \in \mathbf{X} \times \mathbf{E}} l(\mathbf{x}, \mathbf{e}) \leqslant \lambda'_k := \sup_{\lambda} \quad \lambda$$
$$\text{s.t.} \quad \forall \mathbf{e} \in \mathbf{E}, \ \mathbf{M}_k(l\,\mathbf{y}) \succcurlyeq \lambda \mathbf{M}_k(\mathbf{y}).$$

Method 3: `robustsdp`

Robust SDP with $l(\mathbf{x},\mathbf{e}) = \sum_{i=1}^{m} s_i(\mathbf{x}) e_i$:

$$l^* := \min_{(\mathbf{x},\mathbf{e})\in\mathbf{X}\times\mathbf{E}} l(\mathbf{x},\mathbf{e}) \leqslant \lambda'_k := \sup_{\lambda} \quad \lambda$$
$$\text{s.t.} \quad \forall \mathbf{e} \in \mathbf{E},\ \mathbf{M}_k(l\,\mathbf{y}) \succcurlyeq \lambda \mathbf{M}_k(\mathbf{y}).$$

Linearity $\rightsquigarrow \mathbf{M}_k(l\,\mathbf{y}) = \sum_{i=1}^{m} e_i \mathbf{M}_k(s_i\,\mathbf{y})$

Factorize $\mathbf{M}_k(s_i\,\mathbf{y}) = \mathbf{L}_k^i \mathbf{R}_k^i$, $\mathbf{L}_k := [\mathbf{L}_k^1 \cdots \mathbf{L}_k^m]$, $\mathbf{R}_k := [\mathbf{R}_k^1 \cdots \mathbf{R}_k^m]^T$

Method 3: `robustsdp`

Robust SDP with $l(\mathbf{x},\mathbf{e}) = \sum_{i=1}^{m} s_i(\mathbf{x}) e_i$:

$$l^* := \min_{(\mathbf{x},\mathbf{e})\in\mathbf{X}\times\mathbf{E}} l(\mathbf{x},\mathbf{e}) \leqslant \lambda'_k := \sup_{\lambda} \quad \lambda$$
$$\text{s.t.} \quad \forall \mathbf{e} \in \mathbf{E}\,,\ \mathbf{M}_k(l\,\mathbf{y}) \succcurlyeq \lambda \mathbf{M}_k(\mathbf{y}).$$

Linearity $\rightsquigarrow \mathbf{M}_k(l\,\mathbf{y}) = \sum_{i=1}^{m} e_i \mathbf{M}_k(s_i\,\mathbf{y})$

Factorize $\mathbf{M}_k(s_i\,\mathbf{y}) = \mathbf{L}_k^i \mathbf{R}_k^i$, $\mathbf{L}_k := [\mathbf{L}_k^1 \cdots \mathbf{L}_k^m]$, $\mathbf{R}_k := [\mathbf{R}_k^1 \cdots \mathbf{R}_k^m]^T$

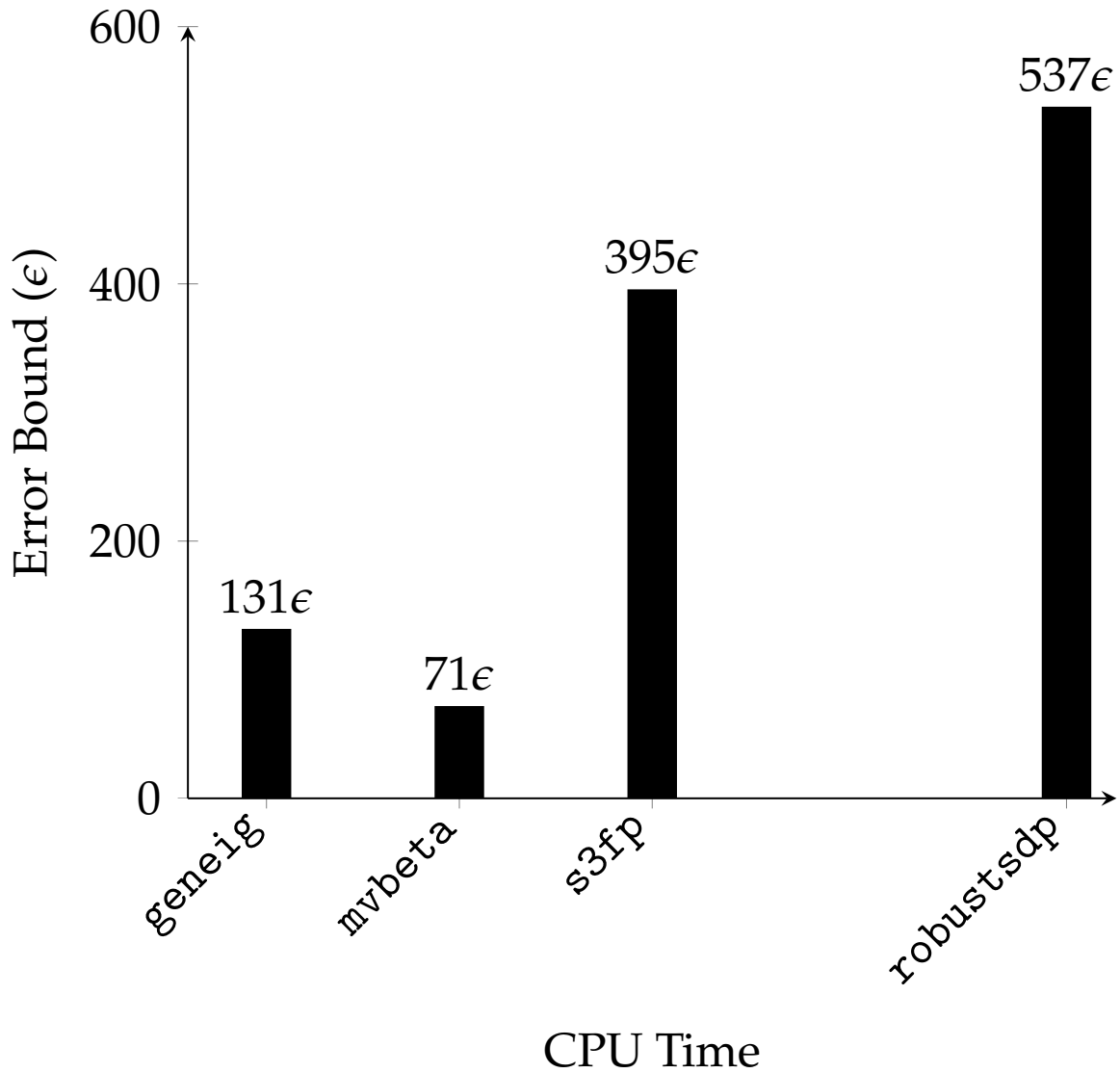
Theorem following from [El Ghaoui et al. 98]

$\lambda'_k \downarrow l^*$ and $\lambda'_k = \sup_{\lambda,\mathbf{S},\mathbf{G}} \quad \lambda$

$$\text{s.t.} \quad \begin{pmatrix} -\lambda \mathbf{M}_k(\mathbf{y}) - \mathbf{L}_k\,\mathbf{S}\,\mathbf{L}_k{}^T & \mathbf{R}_k{}^T + \mathbf{L}_k\,\mathbf{G} \\ \mathbf{R}_k - \mathbf{G}\,\mathbf{L}_k{}^T & \mathbf{S} \end{pmatrix} \succcurlyeq 0\,,$$
$$\mathbf{S}^T = \mathbf{S}\,,\ \mathbf{G}^T = -\mathbf{G}\,.$$

Benchmark `kepler0` with $k = 2$

↑ **Lower** Bounds ↑
↓ **Lower** Bounds ↓



Conclusion

Sparse/Robust SDP relaxations for **NONLINEAR PROGRAMS**:

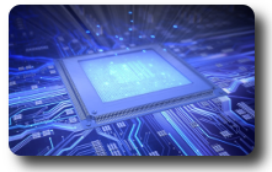
- Polynomial and transcendental programs
- Certified  $\rightsquigarrow$ Formal  roundoff error bounds
  (Joint work with T. Weisser and B. Werner)
- `Real2Float` open source tool:

  `http://nl-certify.forge.ocamlcore.org/real2float.html`

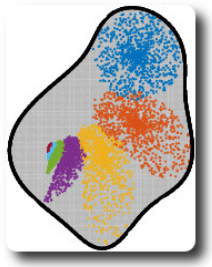
Conclusion

Further research:

- Automatic **FPGA** code generation



- Roundoff error analysis with `while/for` loops



Master / PhD Positions Available !

End

Thank you for your attention!

`http://www-verimag.imag.fr/~magron`

- V. Magron, G. Constantinides, A. Donaldson. Certified Roundoff Error Bounds Using Semidefinite Programming, `arxiv.org/abs/1507.03331`