Certified Roundoff Error bounds using Bernstein Expansions and Sparse Krivine-Stengle Representations

A. Rocca[1,2] with V. Magron[1] and T. Dang[1]

[1]CNRS-Verimag, [2]TIMC-IMAG, Grenoble

July 31, 2017

UNIVERSITÉ Grenoble Alpes

Table of Contents

Introduction

Bernstein Expansions
- Preliminaries
- Error bounds computation

Krivine-Stengle (K.S.)
- Preliminaries
- Error bounds computation

Results

Context

- Roundoff error bounds in computation of **polynomial functions**
- Sound bounds, or certifiable bounds

Context

- Roundoff error bounds in computation of **polynomial functions**
- Sound bounds, or certifiable bounds
  - ☞ polynomial optimization problem = NP-Hard!

Context

- Roundoff error bounds in computation of **polynomial functions**
- Sound bounds, or certifiable bounds
  - ☞ polynomial optimization problem = NP-Hard!
- Simple Rounding Model

Context

- Roundoff error bounds in computation of **polynomial functions**
- Sound bounds, or certifiable bounds
  - polynomial optimization problem = NP-Hard!
- Simple Rounding Model
  - $\mathrm{rnd}(x) = x(1+e)+u$
  - e: error variable, $|e| \leq 2^{-\mathtt{prec}}$
  - u: very small (single: $|e| \leq 2^{-24}, |u| \leq 2^{-150}$)

Related Works

Feature	`Real2Float` [5]	`Rosa` [1]	`FPTaylor` [6]	`Gappa` [2]	`Fluctuat` [3]
Basic FP operations/formats	√	√	√	√	√
Special values ($\pm\infty$, NaN)					√
Input uncertainties	√	√	√	√	√
Transcendental functions	√		√		
Discontinuity errors	√	√			√
Proof certificates	√		√	√	
Methods	SDP/SOS (Sparse)	SMT & Int.Arith	Symbolic Taylor Expans.	Int. Arith.	Static Analysis

Contributions

- Two methods for roundoff error upper bounds:

Contributions

- Two methods for roundoff error upper bounds:
  - Bernstein expansions.
  - Sparse Krivine-Stengle representations.

Contributions

- Two methods for roundoff error upper bounds:
  - Bernstein expansions.
  - Sparse Krivine-Stengle representations.
- Three implementations:

Contributions

- Two methods for roundoff error upper bounds:
  - Bernstein expansions.
  - Sparse Krivine-Stengle representations.
- Three implementations:
  - `FPBern`(a): C++ double precision implementation [4].
  - `FPBern`(b): `Matlab` rational arithmetic implementation [4].
  - `FPKriSten`: `Matlab` & `Cplex` Sparse Krivine-Stengle representations [7].

Simple Running Example

➢ A simple example:

$$f(x) := x^2 - x\ , \quad \forall x \in X = [0,1].$$

Floating point approximation:

$$\hat{f}(x,\mathbf{e}) = (((1+e_1)x(1+e_1)x)(1+e_2) - x(1+e_1))(1+e_3).$$

Roundoff error upper bound:

$$r(x,\mathbf{e}) := \max_{\substack{x\in[0,1]\\ \mathbf{e}\in[-\varepsilon,\varepsilon]^3}} (|\hat{f}(x,\mathbf{e}) - f(x)|) \quad .$$

$$|\hat{f}(x,\mathbf{e}) - f(x)| \leq |I(x,\mathbf{e})| + |h(x,\mathbf{e})|$$

Simple Running Example

- A simple example:

$$f(x) := x^2 - x\ , \quad \forall x \in X = [0, 1].$$

Floating point approximation:

$$\hat{f}(x, \mathbf{e}) = (((1 + e_1)x(1 + e_1)x)(1 + e_2) - x(1 + e_1))(1 + e_3).$$

Roundoff error upper bound of:

$$|\hat{f}(x, \mathbf{e}) - f(x)| \leq \underbrace{|I(x, \mathbf{e})|}_{\text{Our Focus}} + \overbrace{|h(x, \mathbf{e})|}^{\text{Interval Arithmetic or } O(|\mathbf{e}|^2)}$$

Simple Running Example

- A simple example:

$$f(x) := x^2 - x\ , \quad \forall x \in X = [0, 1].$$

Floating point approximation:

$$\hat{f}(x, \mathbf{e}) = (((1 + e_1)x(1 + e_1)x)(1 + e_2) - x(1 + e_1))(1 + e_3).$$

Roundoff error upper bound of:

$$I(x, \mathbf{e}) = (2x^2 - x)e_1 + x^2 e_2 + (x^2 - x)e_3.$$

Bernstein Expansions - Notions

Bernstein Basis

Given $\gamma, \alpha, \mathbf{d} \in \mathbb{N}^n$, and $\mathbf{x} \in [0,1]^n$:

$$f(\mathbf{x}) = \sum_{\gamma} a_\gamma \mathbf{x}^\gamma = \sum_{\alpha \leq \mathbf{d}} b_\alpha^{(f)} \mathbf{B}_{\mathbf{d},\alpha}(\mathbf{x}).$$

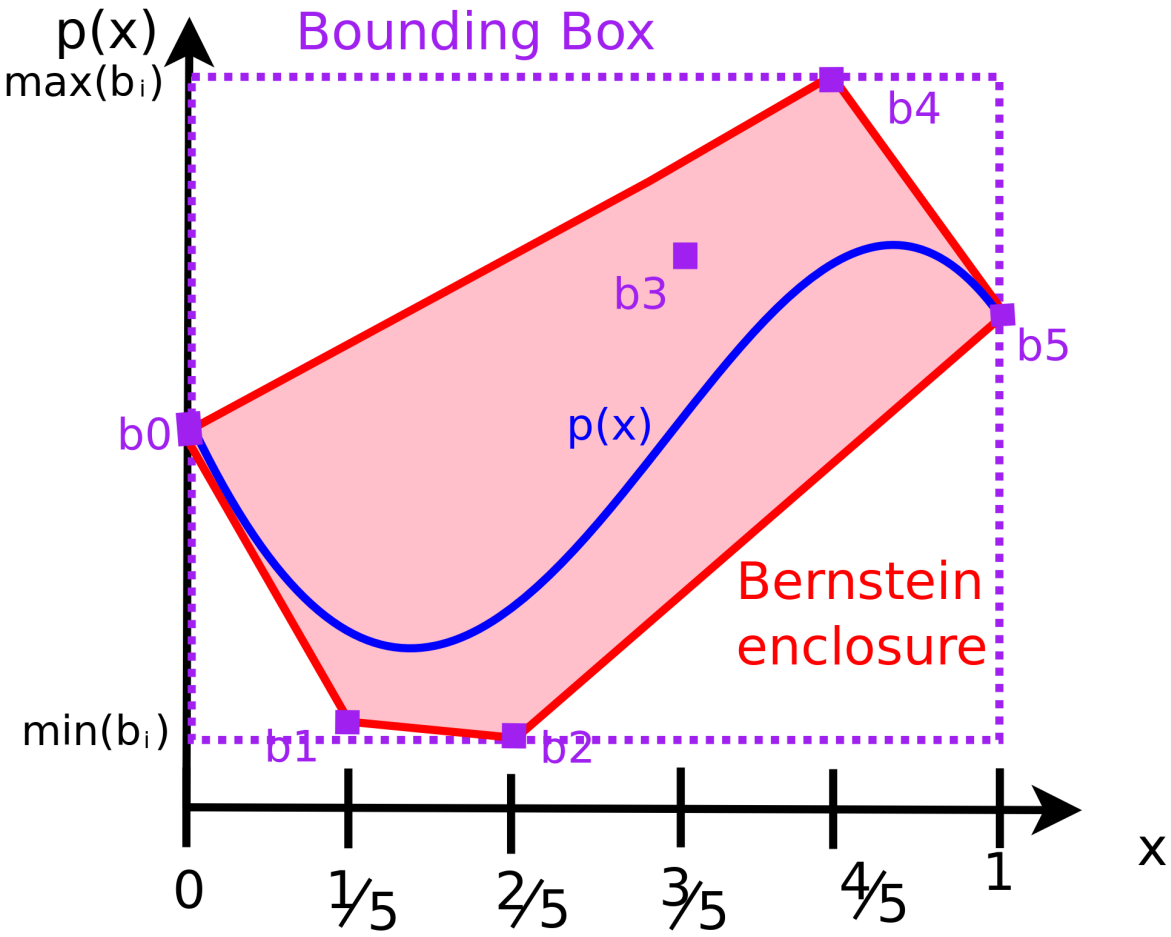
$$b_\alpha^{(f)} = \sum_{\beta < \alpha} \frac{\binom{\alpha}{\beta}}{\binom{\mathbf{d}}{\beta}} a_\beta, \quad \mathbf{0} \leq \alpha \leq \mathbf{d}.$$

Some properties:

$$\min_{\alpha \leq \mathbf{d}} b_\alpha^{(f)} \leq f(\mathbf{x}) \leq \max_{\alpha \leq \mathbf{d}} b_\alpha^{(f)}.$$

$$(\mathbf{d}+\mathbf{1})^{\mathbf{1}} = \prod_{i=1}^{n} (d_i + 1).$$

Bernstein Expansions - A picture



Rounding with Bernstein Expansions - Theory

Re-writing $l(\mathbf{x}, \mathbf{e}) := \sum_{j=1}^{m} s_j(\mathbf{x}) e_j$.

Given $\mathbf{e} \in [-1, 1]^m$ and $x \in [0, 1]^n$:

$$\overline{l'_{\mathbf{d}}} := \max_{\alpha \leq \mathbf{d}} \sum_{j=1}^{m} |b_{\alpha}^{(s_j)}|, \quad \Rightarrow \text{ Sparse decomposition}$$

and $\underline{l'_{\mathbf{d}}} := -\overline{l'_{\mathbf{d}}}$.

Roundoff upper bounds - Bernstein

$$\underline{l'_{\mathbf{d}}} \leq l'(\mathbf{x}, \mathbf{e}) \leq \overline{l'_{\mathbf{d}}} \quad \forall (\mathbf{x}, \mathbf{e}) \in \mathbf{X} \times \mathbf{E}\,.$$

Rounding with Bernstein Expansions - Example

$$l(x, \mathbf{e}) = \underbrace{(2x^2 - x)}_{s_1(x)} e_1 + \overbrace{x^2}^{s_2(x)} e_2 + \underbrace{(x^2 - x)}_{s_3(x)} e_3, \quad \mathbf{d} = (2)$$

$$\mathbf{b}^{s_1} = [0, -0.5, 1], \quad \mathbf{b}^{s_2} = [0, 0, 1], \quad \mathbf{b}^{s_3} = [0, -0.5, 0]$$

Bernstein coefficients:

- $\max_{\alpha \leq 2} \sum_{j=1}^{3} |b_\alpha^{(s_j)}| = 2$
  - ☞ $\overline{l'_{\mathbf{d}}} = 2 \Rightarrow l^* \leq 2\varepsilon$

Rounding with Bernstein Expansions - Example

$$l(x, \mathbf{e}) = \underbrace{(2x^2 - x)}_{s_1(x)} e_1 + \overbrace{x^2}^{s_2(x)} e_2 + \underbrace{(x^2 - x)}_{s_3(x)} e_3, \quad \mathbf{d} = (2)$$

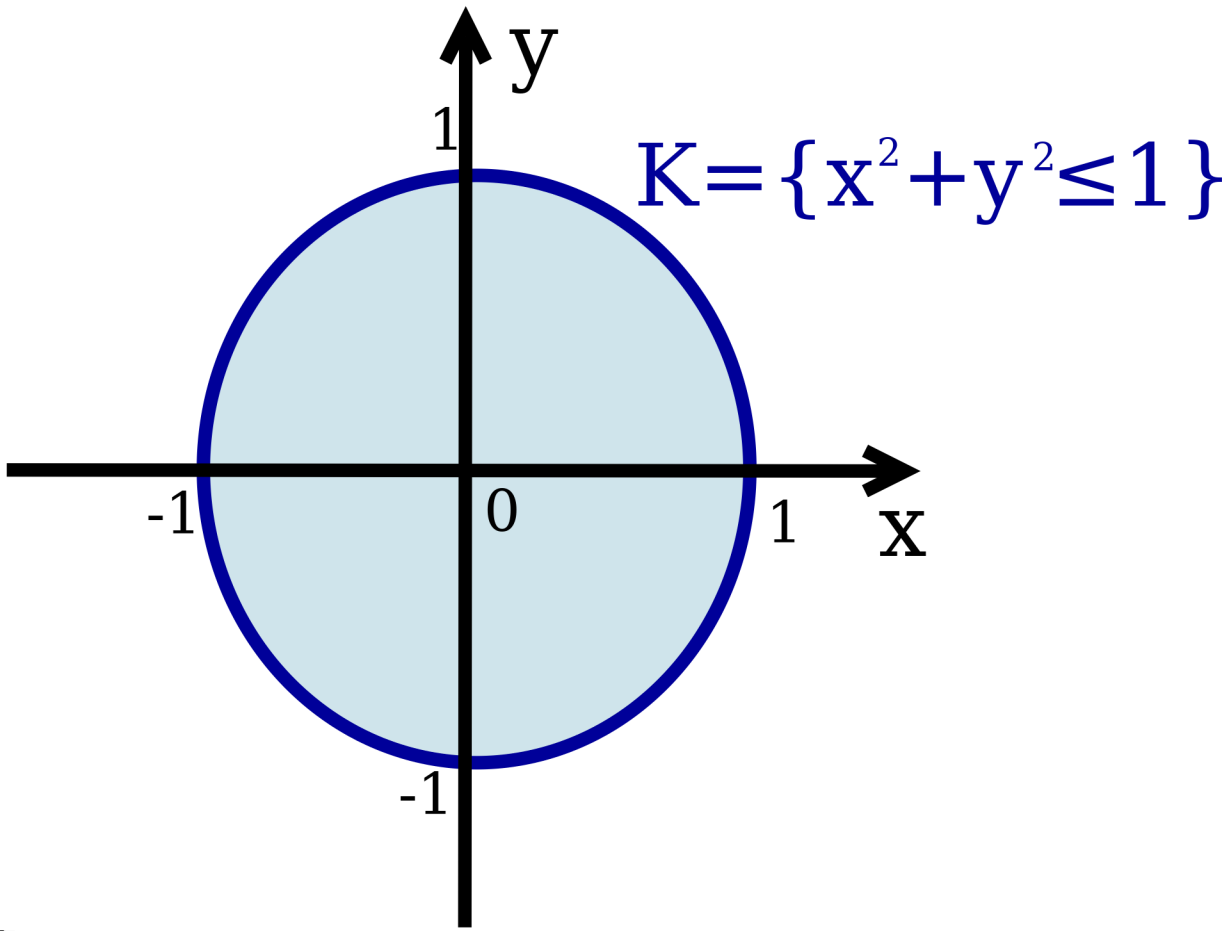
$$\mathbf{b}^{s_1} = [0, -0.5, 1], \quad \mathbf{b}^{s_2} = [0, 0, 1], \quad \mathbf{b}^{s_3} = [0, -0.5, 0]$$

- $\max_{\alpha \leq 2} \sum_{j=1}^{3} |b_\alpha^{(s_j)}| = 2$
  - ☞ $\overline{l'_{\mathbf{d}}} = 2 \Rightarrow l^* \leq 2\varepsilon$

\# Bernstein coefficients:

- ☞ SPARSE: $m(\mathbf{d} + \mathbf{1})^n = 9$
- ☞ DENSE: $(\mathbf{d} + \mathbf{1})^{n+m} = 24$

Semi-algebraic set **K**



Krivine-Stengle - Notions

$\mathbf{K} = \{\mathbf{x} \in \mathbb{R}^n : 0 \leq g_i(\mathbf{x}) \leq 1,\ i = 1, \ldots, p\}$, with $g_1, \ldots, g_p \in \mathbb{R}[\mathbf{x}]$.

Krivine-Stengle Representations

Let $\psi \in \mathbb{R}[\mathbf{x}]$ positive over $\mathbf{K}$. Then $\exists k \in \mathbb{N}$ and $\lambda_{\alpha,\beta} \geq 0$ such that:

$$\psi(\mathbf{x}) = \sum_{|\alpha+\beta| \leq k} \lambda_{\alpha,\beta} h_{\alpha,\beta}(\mathbf{x}), \quad \forall \mathbf{x} \in \mathbb{R}^n.$$

$$h_{\alpha,\beta}(\mathbf{x}) = \mathbf{g}^{\alpha}(\mathbf{1} - \mathbf{g})^{\beta} = \prod_{i=1}^{p} g_i^{\alpha_i}(1 - g_i)^{\beta_i}$$

Krivine-Stengle - Sparsity (I)

Example on $\mathbf{K} = [0,1]^3$

$$f(\mathbf{x}) = x_1x_2 + x_1^2x_3 \quad \Rightarrow \quad J_1 = I_1 = \{1,2\},\ J_2 = I_2 = \{1,3\}$$

Given $m \in \mathbb{N}$, $I_j \subseteq \{1,\dots,n\}$, and $J_j \subseteq \{1,\dots,p\}$ for all $j = 1,\dots,m$ $\{I_j\}$,$\{J_j\}$ define a sparsity pattern of f positive on $\mathbf{K}$ if:

Krivine-Stengle - Sparsity (I)

Example on $\mathbf{K} = [0,1]^3$

$$f(\mathbf{x}) = x_1x_2 + x_1^2x_3 \quad \Rightarrow \quad J_1 = I_1 = \{1,2\},\ J_2 = I_2 = \{1,3\}$$

Given $m \in \mathbb{N}$, $I_j \subseteq \{1,\dots,n\}$, and $J_j \subseteq \{1,\dots,p\}$ for all $j = 1,\dots,m$ $\{I_j\}$,$\{J_j\}$ define a sparsity pattern of f positive on $\mathbf{K}$ if:

- $f = \sum_{j=1}^{m} f_j$ with $f_j \in \mathbb{R}[\mathbf{x}, I_j]$,

Krivine-Stengle - Sparsity (I)

Example on $\mathbf{K} = [0,1]^3$

$$f(\mathbf{x}) = x_1x_2 + x_1^2x_3 \quad \Rightarrow \quad J_1 = I_1 = \{1,2\},\ J_2 = I_2 = \{1,3\}$$

Given $m \in \mathbb{N}$, $I_j \subseteq \{1,\dots,n\}$, and $J_j \subseteq \{1,\dots,p\}$ for all $j = 1,\dots,m$ $\{I_j\}$,$\{J_j\}$ define a sparsity pattern of f positive on $\mathbf{K}$ if:

- $f = \sum_{j=1}^{m} f_j$ with $f_j \in \mathbb{R}[\mathbf{x}, I_j]$, $\quad \Rightarrow \quad x_1x_2 + x_1^2x_3$

Krivine-Stengle - Sparsity (I)

Example on $\mathbf{K} = [0,1]^3$

$$f(\mathbf{x}) = x_1x_2 + x_1^2x_3 \quad \Rightarrow \quad J_1 = I_1 = \{1,2\},\ J_2 = I_2 = \{1,3\}$$

Given $m \in \mathbb{N}$, $I_j \subseteq \{1,\dots,n\}$, and $J_j \subseteq \{1,\dots,p\}$ for all $j = 1,\dots,m$ $\{I_j\}$,$\{J_j\}$ define a sparsity pattern of f positive on $\mathbf{K}$ if:

- $f = \sum_{j=1}^{m} f_j$ with $f_j \in \mathbb{R}[\mathbf{x}, I_j]$, $\quad \Rightarrow \quad x_1x_2 + x_1^2x_3$
- $g_i \in \mathbb{R}[\mathbf{x}, I_j]\ \forall i \in J_j,\ \forall j = 1,\dots,m,$

Krivine-Stengle - Sparsity (I)

Example on $\mathbf{K} = [0,1]^3$

$$f(\mathbf{x}) = x_1x_2 + x_1^2x_3 \quad \Rightarrow \quad J_1 = I_1 = \{1,2\},\ J_2 = I_2 = \{1,3\}$$

Given $m \in \mathbb{N}$, $I_j \subseteq \{1,\dots,n\}$, and $J_j \subseteq \{1,\dots,p\}$ for all $j = 1,\dots,m$ $\{I_j\}$,$\{J_j\}$ define a sparsity pattern of f positive on $\mathbf{K}$ if:

- $f = \sum_{j=1}^{m} f_j$ with $f_j \in \mathbb{R}[\mathbf{x}, I_j]$, $\Rightarrow$ $x_1x_2 + x_1^2x_3$
- $g_i \in \mathbb{R}[\mathbf{x}, I_j]\ \forall i \in J_j,\ \forall j = 1,\dots,m$, $\Rightarrow$ $g_i = x_i,\ i \leq 3$

Krivine-Stengle - Sparsity (I)

Example on $\mathbf{K} = [0,1]^3$

$$f(\mathbf{x}) = x_1x_2 + x_1^2x_3 \quad \Rightarrow \quad J_1 = I_1 = \{1,2\},\ J_2 = I_2 = \{1,3\}$$

Given $m \in \mathbb{N}$, $I_j \subseteq \{1,\dots,n\}$, and $J_j \subseteq \{1,\dots,p\}$ for all $j = 1,\dots,m$ $\{I_j\}$,$\{J_j\}$ define a sparsity pattern of f positive on $\mathbf{K}$ if:

- $f = \sum_{j=1}^m f_j$ with $f_j \in \mathbb{R}[\mathbf{x}, I_j]$, $\Rightarrow$ $x_1x_2 + x_1^2x_3$
- $g_i \in \mathbb{R}[\mathbf{x}, I_j]\ \forall i \in J_j,\ \forall j = 1,\dots,m$, $\Rightarrow$ $g_i = x_i,\ i \leq 3$
- $\bigcup_{j=1}^m I_j = \{1,\dots,n\}$ and $\bigcup_{j=1}^m J_j = \{1,\dots,p\}$,

Krivine-Stengle - Sparsity (I)

Example on $\mathbf{K} = [0,1]^3$

$$f(\mathbf{x}) = x_1x_2 + x_1^2x_3 \quad \Rightarrow \quad J_1 = I_1 = \{1,2\},\ J_2 = I_2 = \{1,3\}$$

Given $m \in \mathbb{N}$, $I_j \subseteq \{1,\dots,n\}$, and $J_j \subseteq \{1,\dots,p\}$ for all $j = 1,\dots,m$ $\{I_j\}$,$\{J_j\}$ define a sparsity pattern of f positive on $\mathbf{K}$ if:

- $f = \sum_{j=1}^m f_j$ with $f_j \in \mathbb{R}[\mathbf{x}, I_j]$, $\quad \Rightarrow \quad x_1x_2 + x_1^2x_3$
- $g_i \in \mathbb{R}[\mathbf{x}, I_j]$ $\forall i \in J_j$, $\forall j = 1,\dots,m$, $\quad \Rightarrow \quad g_i = x_i, \quad i \leq 3$
- $\bigcup_{j=1}^m I_j = \{1,\dots,n\}$ and $\bigcup_{j=1}^m J_j = \{1,\dots,p\}$,
- (Running Intersection Property) $\forall j = 1,\dots,m-1$, $\exists s \leq j$ s.t. $I_{j+1} \cap \bigcup_{i=1}^j I_i \subseteq I_s$,

Krivine-Stengle - Sparsity (I)

Example on $\mathbf{K} = [0,1]^3$

$$f(\mathbf{x}) = x_1x_2 + x_1^2x_3 \quad \Rightarrow \quad J_1 = I_1 = \{1,2\},\ J_2 = I_2 = \{1,3\}$$

Given $m \in \mathbb{N}$, $I_j \subseteq \{1,\dots,n\}$, and $J_j \subseteq \{1,\dots,p\}$ for all $j = 1,\dots,m$ $\{I_j\}$,$\{J_j\}$ define a sparsity pattern of f positive on $\mathbf{K}$ if:

- $f = \sum_{j=1}^m f_j$ with $f_j \in \mathbb{R}[\mathbf{x}, I_j]$, $\quad \Rightarrow \quad x_1x_2 + x_1^2x_3$
- $g_i \in \mathbb{R}[\mathbf{x}, I_j]\ \forall i \in J_j,\ \forall j = 1,\dots,m$, $\quad \Rightarrow \quad g_i = x_i,\ \ i \leq 3$
- $\bigcup_{j=1}^m I_j = \{1,\dots,n\}$ and $\bigcup_{j=1}^m J_j = \{1,\dots,p\}$,
- (Running Intersection Property) $\forall j = 1,\dots,m-1,\ \exists s \leq j$ s.t. $I_{j+1} \cap \bigcup_{i=1}^j I_i \subseteq I_s$, $\quad \Rightarrow \quad I_2 \cap I_1 = \{1\} \subset I_1$

Krivine-Stengle - Sparsity (II)

$\mathbf{K}_j = \{\mathbf{x} \in \mathbb{R}^{n_j} : 0 \leq g_i(\mathbf{x}) \leq 1,\ i \in J_j\}$, with $j = 1, \ldots, m$

Sparse K.S. representations

If f positive over $\mathbf{K}$, and there exists some sparsity pattern $\{I_j\}, \{J_j\}$:

$$f = \sum_{j=1}^{m} \phi_j \text{ and } \phi_j > 0 \text{ over } \mathbf{K}_j$$

$$\phi_j = \sum_{|\alpha_j + \beta_j| \leq k} \lambda_{\alpha_j, \beta_j} h_{\alpha_j, \beta_j}\ , \quad j = 1, \ldots, m, \quad \lambda_{\alpha_j, \beta_j} \geq 0.$$

$$h_{\alpha_j, \beta_j} := \mathbf{g}^{\alpha_j} (\mathbf{1} - \mathbf{g})^{\beta_j},\ \text{with } \alpha_j, \beta_j \in \mathbb{N}^{n_j}$$

Rounding with Sparse K.S - Theory

$$l(\mathbf{x},\mathbf{e}) := \sum_{j=1}^{m} \frac{\partial r(\mathbf{x},\mathbf{e})}{\partial e_j}(\mathbf{x},0)e_j = \sum_{j=1}^{m} s_j(\mathbf{x})e_j$$

$$\mathbf{K} = \{\mathbf{y} \in \mathbb{R}^{n+m} : 0 \le g_j(\mathbf{y}) \le 1\,, \quad j = 1,\ldots,n+m\}\,,$$

with (for $\mathbf{x} \in [0,1]^n$): $g_j(\mathbf{y}) := x_j, \quad j = 1,\ldots,n$

and: $g_j(\mathbf{y}) := \frac{1}{2} + \frac{e_j}{2}, \quad j = n+1,\ldots,n+m.$

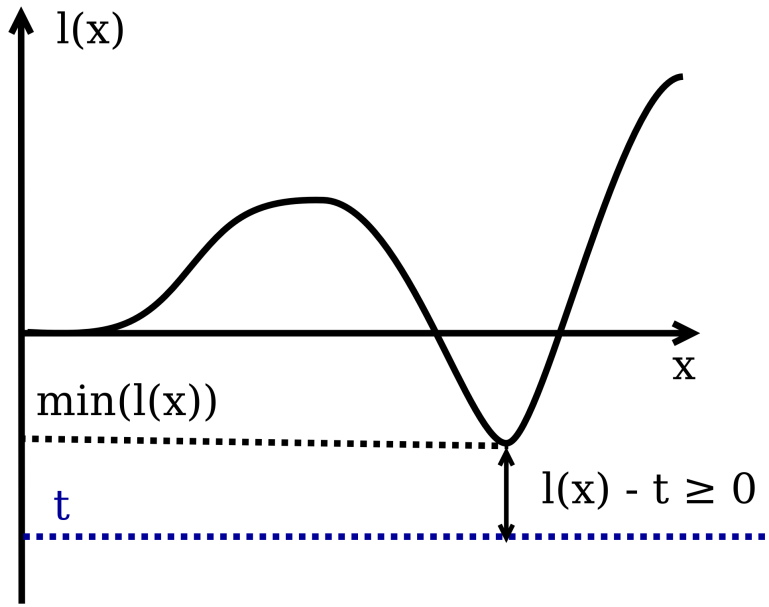
$$J_j = I_j = \{1,\ldots,n,n+j\}$$

➢ $h_{\alpha_j,\beta_j}(\mathbf{y}) = h_{\alpha'_j,\beta'_j,\gamma_j,\delta_j}(\mathbf{x},\mathbf{e}) = \mathbf{x}^{\alpha'_j}(\mathbf{1}-\mathbf{x})^{\beta'_j}(\frac{1}{2}+\frac{e_j}{2})^{\gamma_j}(\frac{1}{2}-\frac{e_j}{2})^{\delta_j}\,.$

Rounding with Sparse K.S - Theory

- $l(\mathbf{x},\mathbf{e}) := \sum_{j=1}^{m} \frac{\partial r(\mathbf{x},\mathbf{e})}{\partial e_j}(\mathbf{x},0)e_j = \sum_{j=1}^{m} s_j(\mathbf{x})e_j$
- $h_{\alpha_j,\beta_j}(\mathbf{y}) = \mathbf{x}^{\alpha'_j}(\mathbf{1}-\mathbf{x})^{\beta'_j}(\frac{1}{2}+\frac{e_j}{2})^{\gamma_j}(\frac{1}{2}-\frac{e_j}{2})^{\delta_j}$.

$$\begin{aligned} \underline{l'_k} := \max_{t\in\mathbb{R}} \quad & t, \\ \text{s.t} \quad & l'(\mathbf{x},\mathbf{e}) - t \geq 0, \\ & \forall(\mathbf{x},\mathbf{e}) \in \mathbf{K}. \end{aligned}$$



Rounding with Sparse K.S - Theory

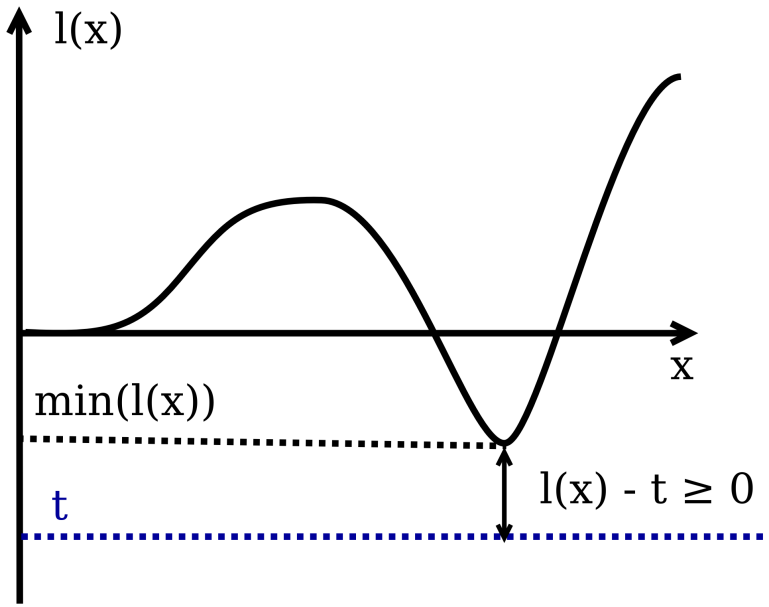
- $l(\mathbf{x}, \mathbf{e}) := \sum_{j=1}^{m} \frac{\partial r(\mathbf{x},\mathbf{e})}{\partial e_j}(\mathbf{x}, 0) e_j = \sum_{j=1}^{m} s_j(\mathbf{x}) e_j$
- $h_{\alpha_j,\beta_j}(\mathbf{y}) = \mathbf{x}^{\alpha'_j}(\mathbf{1} - \mathbf{x})^{\beta'_j}(\frac{1}{2} + \frac{e_j}{2})^{\gamma_j}(\frac{1}{2} - \frac{e_j}{2})^{\delta_j}$.

$$\underline{l'_k} := \max_{t,\lambda_{\alpha_j,\beta_j}} \quad t\,,$$

$$\text{s.t} \quad l' - t = \sum_{j=1}^{m} \phi_j\,,$$

$$\phi_j = \sum_{|\alpha_j+\beta_j|\leq k} \lambda_{\alpha_j,\beta_j} h_{\alpha_j,\beta_j}\,,$$

$$\lambda_{\alpha_j,\beta_j} \geq 0\,, \quad j = 1, \ldots, m\,.$$



Rounding with Sparse K.S - Linear Programming

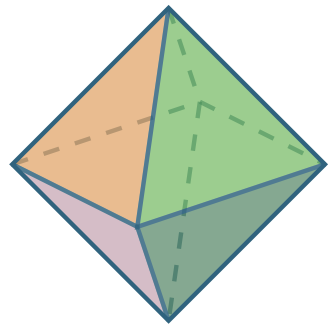
$$\underline{l'_k} := \max_{t,\lambda_{\alpha_j,\beta_j}} \quad t\,,$$

$$\text{s.t} \quad l' - t = \sum_{j=1}^{m} \phi_j\,,$$

$$\phi_j = \sum_{|\alpha_j+\beta_j|\leq k} \lambda_{\alpha_j,\beta_j} h_{\alpha_j,\beta_j}\,, \quad j = 1,\ldots,m\,,$$

$$\lambda_{\alpha_j,\beta_j} \geq 0\,, \quad j = 1,\ldots,m\,.$$

- Equivalent to a Linear Program (LP).
  - Optimization with linear cost function & Polytopic feasible set.
- Used LP solver in implementation: `Cplex`.



Rounding with Sparse K.S - Example

$$\underline{l'_3} := \max_{t,\lambda_{\alpha_j,\beta_j}} \quad t\,,$$

$$\text{s.t} \quad (2x^2 - x)e_1 + x^2 e_2 + (x^2 - x)e_3 - t = \sum_{j=1}^{m} \phi_j\,,$$

$$\phi_j = \sum_{|\alpha_j+\beta_j|\leq 3} \lambda_{\alpha_j,\beta_j} h_{\alpha_j,\beta_1}(x, e_j)\,,$$

$$\lambda_{\alpha_j,\beta_j} \geq 0\,, \quad j = 1,\ldots,3\,.$$

Solving a LP of:

➢ SPARSE (var): $m\binom{2(n+1)+k}{k} + 1 = 106$

☞ $l'^* \leq \max(|\underline{l'_3}|, |\overline{l'_3}|) = 2$

Rounding with Sparse K.S - Example

$$\underline{l'_3} := \max_{t,\lambda_{\alpha_j,\beta_j}} \quad t\,,$$

$$\text{s.t} \quad (2x^2 - x)e_1 + x^2 e_2 + (x^2 - x)e_3 - t = \sum_{j=1}^{m} \phi_j\,,$$

$$\phi_j = \sum_{|\alpha_j+\beta_j|\leq 3} \lambda_{\alpha_j,\beta_j} h_{\alpha_j,\beta_1}(x, e_j)\,,$$

$$\lambda_{\alpha_j,\beta_j} \geq 0\,, \quad j = 1,\ldots,3\,.$$

☞ $l'^* \leq \max(|\underline{l'_3}|, |\overline{l'_3}|) = 2$

Solving a LP of:

- ➢ SPARSE (var): $m\binom{2(n+1)+k}{k} + 1 = 106$
- ➢ DENSE (var): $\binom{2(n+m)+k}{k} + 1 = 165$

Rounding with Sparse K.S - Example

$$\underline{l'_3} := \max_{t,\lambda_{\alpha_j,\beta_j}} \quad t\,,$$

$$\text{s.t} \quad (2x^2 - x)e_1 + x^2 e_2 + (x^2 - x)e_3 - t = \sum_{j=1}^{m} \phi_j\,,$$

$$\phi_j = \sum_{|\alpha_j+\beta_j|\leq 3} \lambda_{\alpha_j,\beta_j} h_{\alpha_j,\beta_1}(x, e_j)\,,$$

$$\lambda_{\alpha_j,\beta_j} \geq 0\,, \quad j = 1,\ldots,3\,.$$

☞ $l'^* \leq \max(|\underline{l'_3}|, |\overline{l'_3}|) = 2$

Solving a LP of:

- ➢ SPARSE (var): $m\binom{2(n+1)+k}{k} + 1 = 106$
- ➢ DENSE (var): $\binom{2(n+m)+k}{k} + 1 = 165$
- ➢ SPARSE (constraint): $[\frac{mk}{n+1} + 1]\binom{n+k}{k} = 22$

Rounding with Sparse K.S - Example

$$\underline{l'_3} := \max_{t,\lambda_{\alpha_j,\beta_j}} \quad t\,,$$

$$\text{s.t} \quad (2x^2 - x)e_1 + x^2 e_2 + (x^2 - x)e_3 - t = \sum_{j=1}^{m} \phi_j\,,$$

$$\phi_j = \sum_{|\alpha_j+\beta_j|\leq 3} \lambda_{\alpha_j,\beta_j} h_{\alpha_j,\beta_1}(x, e_j)\,,$$

$$\lambda_{\alpha_j,\beta_j} \geq 0\,, \quad j = 1,\ldots,3\,.$$

☞ $l'^* \leq \max(|\underline{l'_3}|, |\overline{l'_3}|) = 2$

Solving a LP of:

- SPARSE (var): $m\binom{2(n+1)+k}{k} + 1 = 106$
- DENSE (var): $\binom{2(n+m)+k}{k} + 1 = 165$
- SPARSE (constraint): $[\frac{mk}{n+1} + 1]\binom{n+k}{k} = 22$
- DENSE (constraint): $\binom{n+m+k}{k} = 35$

Compared tools

`Real2Float`	`Ocaml/Coq`	Certificates
`Rosa`	`Java`	No Certificate
`FPTaylor`	`Ocaml/Hol Light`	Certificates

- ➢ Using default features: rounding of polynomial functions over boxes.
- ➢ No convergence speed study.
- ✎ Additional custom made benchmarks:

$$\texttt{ex-n-nSum-deg}(\mathbf{x}) := \sum_{j=0}^{\texttt{nSum}} \left(\prod_{k=1}^{\texttt{deg}} \left(\sum_{i=1}^{\texttt{n}} x_i \right) \right).$$

Performances (seconds) - Classical Benchmarks

				C++	Matlab	Matlab	Ocaml	Java	OCaml
Benchmark	n	m	d	FPBern(a)	FPBern(b)	FPKriSten	Real2Float	Rosa	FPTaylor
rigidBody1	3	10	3	**5e-4**	0.88	0.22(0.02)	0.58	0.13	1.84
rigidBody2	3	15	5	**2e-3**	1.87	2.78(0.47)	0.26	2.17	3.01
kepler0	6	21	3	**4e-3**	9.62	1.93(0.18)	0.22	3.78	4.93
kepler1	4	28	4	**6e-3**	6.91	3.93(0.53)	17.6	63.1	9.33
kepler2	6	42	4	**5e-2**	64.9	20.5(3.75)	16.5	106	19.1
sineTaylor	1	13	8	**6e-4**	0.50	0.92(0.27)	1.05	3.50	2.91
sineOrder3	1	6	4	**2e-4**	0.27	0.08(0.01)	0.40	0.48	1.90
sqroot	1	15	5	**2e-4**	0.34	0.24(0.02)	0.14	0.77	2.70
himmilbeau	2	11	5	**1e-3**	1.72	0.77(0.22)	0.20	2.51	3.28
schwefel	3	15	5	**2e-3**	3.04	2.90(0.56)	0.23	3.91	0.53
magnetism	7	27	3	**9e-2**	176	3.07(0.26)	0.29	1.95	5.91
caprasse	4	34	5	**6e-3**	6.03	18.8(4.89)	3.63	17.6	12.2

☞ FPBern(a): **not certified but fast.**

Performances (seconds) - Classical Benchmarks

				C++	Matlab	Matlab	Ocaml	Java	OCaml
Benchmark	n	m	d	FPBern(a)	FPBern(b)	FPKriSten	Real2Float	Rosa	FPTaylor
rigidBody1	3	10	3	**5e−4**	0.88	0.22(0.02)	0.58	0.13	1.84
rigidBody2	3	15	5	**2e−3**	1.87	2.78(0.47)	0.26	2.17	3.01
kepler0	6	21	3	**4e−3**	9.62	1.93(0.18)	0.22	3.78	4.93
kepler1	4	28	4	**6e−3**	6.91	3.93(0.53)	17.6	63.1	9.33
kepler2	6	42	4	**5e−2**	64.9	20.5(3.75)	16.5	106	19.1
sineTaylor	1	13	8	**6e−4**	0.50	0.92(0.27)	1.05	3.50	2.91
sineOrder3	1	6	4	**2e−4**	0.27	0.08(0.01)	0.40	0.48	1.90
sqroot	1	15	5	**2e−4**	0.34	0.24(0.02)	0.14	0.77	2.70
himmilbeau	2	11	5	**1e−3**	1.72	0.77(0.22)	0.20	2.51	3.28
schwefel	3	15	5	**2e−3**	3.04	2.90(0.56)	0.23	3.91	0.53
magnetism	7	27	3	**9e−2**	176	3.07(0.26)	0.29	1.95	5.91
caprasse	4	34	5	**6e−3**	6.03	18.8(4.89)	3.63	17.6	12.2

☞ FPBern(a): **not certified but fast.**

☞ FPBern(b): **sound & "not too slow".**

Performances (seconds) - Classical Benchmarks

				C++	Matlab	Matlab	Ocaml	Java	OCaml
Benchmark	*n*	*m*	*d*	FPBern(a)	FPBern(b)	FPKriSten	Real2Float	Rosa	FPTaylor
rigidBody1	3	10	3	**5e−4**	0.88	0.22(0.02)	0.58	0.13	1.84
rigidBody2	3	15	5	**2e−3**	1.87	2.78(0.47)	0.26	2.17	3.01
kepler0	6	21	3	**4e−3**	9.62	1.93(0.18)	0.22	3.78	4.93
kepler1	4	28	4	**6e−3**	6.91	3.93(0.53)	17.6	63.1	9.33
kepler2	6	42	4	**5e−2**	64.9	20.5(3.75)	16.5	106	19.1
sineTaylor	1	13	8	**6e−4**	0.50	0.92(0.27)	1.05	3.50	2.91
sineOrder3	1	6	4	**2e−4**	0.27	0.08(0.01)	0.40	0.48	1.90
sqroot	1	15	5	**2e−4**	0.34	0.24(0.02)	0.14	0.77	2.70
himmilbeau	2	11	5	**1e−3**	1.72	0.77(0.22)	0.20	2.51	3.28
schwefel	3	15	5	**2e−3**	3.04	2.90(0.56)	0.23	3.91	0.53
magnetism	7	27	3	**9e−2**	176	3.07(0.26)	0.29	1.95	5.91
caprasse	4	34	5	**6e−3**	6.03	18.8(4.89)	3.63	17.6	12.2

☞ FPBern(a): **not certified but fast.**

☞ FPBern(b): **sound & "not too slow".**

☞ FPKriSten: **certificates & LP solving fast.**

Performances (seconds) - Generated Benchmarks

				C++	Matlab	Matlab	Ocaml	Java	OCaml
Benchmark	*n*	*m*	*d*	FPBern(a)	FPBern(b)	FPKriSten	Real2Float	Rosa	FPTaylor
ex-2-2-5	2	9	3	**4e−4**	0.69	0.12(0.01)	0.07	4.20	2.30
ex-2-2-10	2	14	3	**5e−4**	0.71	0.17(0.01)	0.35	4.75	3.42
ex-2-2-15	2	19	3	**6e−4**	0.72	0.23(0.02)	9.75	5.33	4.91
ex-2-2-20	2	24	3	**8e−4**	0.73	0.28(0.02)	TIMEOUT	6.28	6.27
ex-2-5-2	2	9	6	**2e−2**	2.34	1.23(0.26)	0.27	4.26	2.53
ex-2-10-2	2	14	11	**2e−2**	7.34	96.9(58.5)	49.2	9.37	5.07
ex-5-2-2	5	12	3	**8e−3**	18.3	0.70(0.08)	0.21	4.45	12.3
ex-10-2-2	10	22	3	**2.50**	TIMEOUT	6.11(0.6)	30.7	5.34	34.6

☞ No major effect of the error variables.

Performances (seconds) - Generated Benchmarks

				C++	Matlab	Matlab	Ocaml	Java	OCaml
Benchmark	n	m	d	FPBern(a)	FPBern(b)	FPKriSten	Real2Float	Rosa	FPTaylor
ex-2-2-5	2	9	3	**4e−4**	0.69	0.12(0.01)	0.07	4.20	2.30
ex-2-2-10	2	14	3	**5e−4**	0.71	0.17(0.01)	0.35	4.75	3.42
ex-2-2-15	2	19	3	**6e−4**	0.72	0.23(0.02)	9.75	5.33	4.91
ex-2-2-20	2	24	3	**8e−4**	0.73	0.28(0.02)	TIMEOUT	6.28	6.27
ex-2-5-2	2	9	6	**2e−2**	2.34	1.23(0.26)	0.27	4.26	2.53
ex-2-10-2	2	14	11	**2e−2**	7.34	96.9(58.5)	49.2	9.37	5.07
ex-5-2-2	5	12	3	**8e−3**	18.3	0.70(0.08)	0.21	4.45	12.3
ex-10-2-2	10	22	3	**2.50**	TIMEOUT	6.11(0.6)	30.7	5.34	34.6

☞ No major effect of the error variables.

☞ Bernstein appropriate for "high" degree: $O(d^n)$ (fixed n)

Performances (seconds) - Generated Benchmarks

				C++	Matlab	Matlab	Ocaml	Java	OCaml
Benchmark	n	m	d	FPBern(a)	FPBern(b)	FPKriSten	Real2Float	Rosa	FPTaylor
ex-2-2-5	2	9	3	**4e-4**	0.69	0.12(0.01)	0.07	4.20	2.30
ex-2-2-10	2	14	3	**5e-4**	0.71	0.17(0.01)	0.35	4.75	3.42
ex-2-2-15	2	19	3	**6e-4**	0.72	0.23(0.02)	9.75	5.33	4.91
ex-2-2-20	2	24	3	**8e-4**	0.73	0.28(0.02)	TIMEOUT	6.28	6.27
ex-2-5-2	2	9	6	**2e-2**	2.34	1.23(0.26)	0.27	4.26	2.53
ex-2-10-2	2	14	11	**2e-2**	7.34	96.9(58.5)	49.2	9.37	5.07
ex-5-2-2	5	12	3	**8e-3**	18.3	0.70(0.08)	0.21	4.45	12.3
ex-10-2-2	10	22	3	**2.50**	TIMEOUT	6.11(0.6)	30.7	5.34	34.6

- ☞ No major effect of the error variables.
- ☞ Bernstein appropriate for "high" degree: $O(d^n)$ (fixed n)
- ☞ K.S. appropriate for "high" dimension: $O(n^d)$ (fixed d)

Accuracy - Classical Benchmarks

				C++	Matlab	Matlab	Ocaml	Java	OCaml
Benchmark	*n*	*m*	*d*	FPBern(a)	FPBern(b)	FPKriSten	Real2Float	Rosa	FPTaylor
rigidBody1	3	10	3	5.33e−13	5.33e−13	5.33e−13	5.33e−13	5.08e−13	**3.87e−13**
rigidBody2	3	15	5	6.48e−11	6.48e−11	6.48e−11	6.48e−11	6.48e−11	**5.24e−11**
kepler0	6	21	3	1.08e−13	1.08e−13	1.08e−13	1.18e−13	1.16e−13	**1.05e−13**
kepler1	4	28	4	4.23e−13	**4.04e−13**	4.23e−13	4.47e−13	6.49e−13	4.49e−13
kepler2	6	42	4	**2.03e−12**	**2.03e−12**	**2.03e−12**	2.09e−12	2.89e−12	2.10e−12
sineTaylor	1	13	8	5.51e−16	**5.48e−16**	5.51e−16	6.03e−16	9.56e−16	6.75e−16
sineOrder3	1	6	4	1.35e−15	1.35e−15	1.25e−15	1.19e−15	1.11e−15	**9.97e−16**
sqroot	1	15	5	1.29e−15	1.29e−15	1.29e−15	1.29e−15	8.41e−16	**7.13e−16**
himmilbeau	2	11	5	2.00e−12	2.00e−12	1.97e−12	1.43e−12	1.43e−12	**1.32e−12**
schwefel	3	15	5	1.48e−11	1.48e−11	1.48e−11	1.49e−11	1.49e−11	**1.03e−11**
magnetism	7	27	3	1.27e−14	1.27e−14	1.27e−14	1.27e−14	1.27e−14	**7.61e−15**
caprasse	4	34	5	4.49e−15	4.49e−15	4.49e−15	5.63e−15	5.96e−15	**3.04e−15**

☞ Similar accuracy to Real2Float & Rosa.

Accuracy - Classical Benchmarks

				C++	Matlab	Matlab	Ocaml	Java	OCaml
Benchmark	*n*	*m*	*d*	FPBern(a)	FPBern(b)	FPKriSten	Real2Float	Rosa	FPTaylor
rigidBody1	3	10	3	5.33e−13	5.33e−13	5.33e−13	5.33e−13	5.08e−13	**3.87e−13**
rigidBody2	3	15	5	6.48e−11	6.48e−11	6.48e−11	6.48e−11	6.48e−11	**5.24e−11**
kepler0	6	21	3	1.08e−13	1.08e−13	1.08e−13	1.18e−13	1.16e−13	**1.05e−13**
kepler1	4	28	4	4.23e−13	**4.04e−13**	4.23e−13	4.47e−13	6.49e−13	4.49e−13
kepler2	6	42	4	**2.03e−12**	**2.03e−12**	**2.03e−12**	2.09e−12	2.89e−12	2.10e−12
sineTaylor	1	13	8	5.51e−16	**5.48e−16**	5.51e−16	6.03e−16	9.56e−16	6.75e−16
sineOrder3	1	6	4	1.35e−15	1.35e−15	1.25e−15	1.19e−15	1.11e−15	**9.97e−16**
sqroot	1	15	5	1.29e−15	1.29e−15	1.29e−15	1.29e−15	8.41e−16	**7.13e−16**
himmilbeau	2	11	5	2.00e−12	2.00e−12	1.97e−12	1.43e−12	1.43e−12	**1.32e−12**
schwefel	3	15	5	1.48e−11	1.48e−11	1.48e−11	1.49e−11	1.49e−11	**1.03e−11**
magnetism	7	27	3	1.27e−14	1.27e−14	1.27e−14	1.27e−14	1.27e−14	**7.61e−15**
caprasse	4	34	5	4.49e−15	4.49e−15	4.49e−15	5.63e−15	5.96e−15	**3.04e−15**

☞ Similar accuracy to Real2Float & Rosa.

Accuracy - Classical Benchmarks

				C++	Matlab	Matlab	Ocaml	Java	OCaml
Benchmark	*n*	*m*	*d*	FPBern(a)	FPBern(b)	FPKriSten	Real2Float	Rosa	FPTaylor
rigidBody1	3	10	3	5.33e−13	5.33e−13	5.33e−13	5.33e−13	5.08e−13	**3.87e−13**
rigidBody2	3	15	5	6.48e−11	6.48e−11	6.48e−11	6.48e−11	6.48e−11	**5.24e−11**
kepler0	6	21	3	1.08e−13	1.08e−13	1.08e−13	1.18e−13	1.16e−13	**1.05e−13**
kepler1	4	28	4	4.23e−13	**4.04e−13**	4.23e−13	4.47e−13	6.49e−13	4.49e−13
kepler2	6	42	4	**2.03e−12**	**2.03e−12**	**2.03e−12**	2.09e−12	2.89e−12	2.10e−12
sineTaylor	1	13	8	5.51e−16	**5.48e−16**	5.51e−16	6.03e−16	9.56e−16	6.75e−16
sineOrder3	1	6	4	1.35e−15	1.35e−15	1.25e−15	1.19e−15	1.11e−15	**9.97e−16**
sqroot	1	15	5	1.29e−15	1.29e−15	1.29e−15	1.29e−15	8.41e−16	**7.13e−16**
himmilbeau	2	11	5	2.00e−12	2.00e−12	1.97e−12	1.43e−12	1.43e−12	**1.32e−12**
schwefel	3	15	5	1.48e−11	1.48e−11	1.48e−11	1.49e−11	1.49e−11	**1.03e−11**
magnetism	7	27	3	1.27e−14	1.27e−14	1.27e−14	1.27e−14	1.27e−14	**7.61e−15**
caprasse	4	34	5	4.49e−15	4.49e−15	4.49e−15	5.63e−15	5.96e−15	**3.04e−15**

☞ Similar accuracy to `Real2Float` & `Rosa`.

Accuracy - Generated Benchmarks

				C++	Matlab	Matlab	Ocaml	Java	OCaml
Benchmark	*n*	*m*	*d*	FPBern(a)	FPBern(b)	FPKriSten	Real2Float	Rosa	FPTaylor
ex-2-2-5	2	9	3	2.23e-14	2.23e-14	2.23e-14	2.23e-14	2.23e-14	**1.96e-14**
ex-2-2-10	2	14	3	5.33e-14	5.33e-14	5.33e-14	5.33e-15	5.33e-14	**4.85e-14**
ex-2-2-15	2	19	3	9.55e-14	9.55e-14	9.55e-14	9.55e-14	9.55e-14	**8.84e-14**
ex-2-2-20	2	24	3	1.49e-13	1.49e-13	1.49e-13	TIMEOUT	1.49e-13	**1.40e-13**
ex-2-5-2	2	9	6	1.67e-13	1.67e-13	1.67e-13	1.67e-13	1.67e-13	**1.41e-13**
ex-2-10-2	2	14	11	1.05e-11	1.05e-11	1.34e-11	1.05e-11	1.05e-11	**8.76e-12**
ex-5-2-2	5	12	3	8.55e-14	8.55e-14	8.55e-14	8.55e-14	8.55e-14	**7.72e-14**
ex-10-2-2	10	22	3	5.16e-13	TIMEOUT	5.16e-13	5.16e-13	5.16e-13	**4.82e-13**

☞ Optimal Solutions on the custom benchmarks !

Result Overview

- `FPBern` & `FPKriSten`: similar accuracy with `Real2Float` & `Rosa`.

Result Overview

- `FPBern` & `FPKriSten`: similar accuracy with `Real2Float` & `Rosa`.
- `FPBern`(a): Not sound BUT very fast.

Result Overview

- `FPBern` & `FPKriSten`: similar accuracy with `Real2Float` & `Rosa`.
- `FPBern`(a): Not sound BUT very fast.
- `FPBern`(b): Sound BUT a bit slower $\Rightarrow$ `Matlab`.

Result Overview

- `FPBern` & `FPKriSten`: similar accuracy with `Real2Float` & `Rosa`.
- `FPBern`(a): Not sound BUT very fast.
- `FPBern`(b): Sound BUT a bit slower $\Rightarrow$ `Matlab`.
- `FPKriSten`:

Result Overview

- `FPBern` & `FPKriSten`: similar accuracy with `Real2Float` & `Rosa`.
- `FPBern`(a): Not sound BUT very fast.
- `FPBern`(b): Sound BUT a bit slower $\Rightarrow$ `Matlab`.
- `FPKriSten`:
  - Generate certificates.

Result Overview

- `FPBern` & `FPKriSten`: similar accuracy with `Real2Float` & `Rosa`.
- `FPBern`(a): Not sound BUT very fast.
- `FPBern`(b): Sound BUT a bit slower $\Rightarrow$ `Matlab`.
- `FPKriSten`:
  - Generate certificates.
  - LP built in `Matlab` THEN solved with `Cplex` $\Rightarrow$ Speed up possible.

Conclusions

- ☞ Similar accuracy.

Conclusions

- ☞ Similar accuracy.
- ☞ Computational cost due to error variables **e** drastically reduced.

Conclusions

- Similar accuracy.
- Computational cost due to error variables **e** drastically reduced.
- Bernstein appropriate for “high” degree/low dimensions.

Conclusions

- ☞ Similar accuracy.
- ☞ Computational cost due to error variables **e** drastically reduced.
- ☞ Bernstein appropriate for “high” degree/low dimensions.
- ☞ K.S. appropriate for “higher” dimensions/low degree.

Perspectives

- `FPKriSten` in `C++`.

Perspectives

- `FPKriSten` in `C++`.
- Extend `FPKriSten` to algebraic compact set.

Perspectives

- `FPKriSten` in `C++`.
- Extend `FPKriSten` to algebraic compact set.
- `FPBern` with rational arithmetic in `C++`.

Perspectives

- `FPKriSten` in `C++`.
- Extend `FPKriSten` to algebraic compact set.
- `FPBern` with rational arithmetic in `C++`.
- Polytopic domains for `FPBern`.

Perspectives

- `FPKriSten` in `C++`.
- Extend `FPKriSten` to algebraic compact set.
- `FPBern` with rational arithmetic in `C++`.
- Polytopic domains for `FPBern`.
- Set splitting for better accuracy (or degree increase).

Perspectives

- `FPKriSten` in `C++`.
- Extend `FPKriSten` to algebraic compact set.
- `FPBern` with rational arithmetic in `C++`.
- Polytopic domains for `FPBern`.
- Set splitting for better accuracy (or degree increase).
- Handling Loops, Handling conditional statements.

Perspectives

- `FPKriSten` in `C++`.
- Extend `FPKriSten` to algebraic compact set.
- `FPBern` with rational arithmetic in `C++`.
- Polytopic domains for `FPBern`.
- Set splitting for better accuracy (or degree increase).
- Handling Loops, Handling conditional statements.
- ...

Thank You !

- ☞ FPKriSten: https://github.com/roccaa/FPKriSten
- ☞ FPBern: https://github.com/roccaa/FPBern
- ☞ Paper: https://arxiv.org/abs/1610.07038

Bibliography I


E. Darulova and V. Kuncak.
Sound compilation of reals.
Acm Sigplan Notices, 49(1):235–248, 2014.


M. Daumas and G. Melquiond.
Certification of Bounds on Expressions Involving Rounded Operators.
ACM TMS, 37(1):2:1–2:20, Jan. 2010.

Bibliography II


D. Delmas, E. Goubault, S. Putot, J. Souyris, K. Tekkal, and F. Védrine.
Towards an industrial use of fluctuat on safety-critical avionics software.
In *FMICS*, volume 5825 of *LNCS*, pages 53–69. 2009.


T. Dreossi and T. Dang.
Parameter synthesis for polynomial biological models.
In *HSCC*, pages 233–242. ACM, 2014.

Bibliography III


V. Magron, G. Constantinides, and A. Donaldson.
Certified roundoff error bounds using semidefinite programming.
ACM (TOMS), 43(4):34, 2017.


A. Solovyev, C. Jacobsen, Z. Rakamarić, and G. Gopalakrishnan.
Rigorous Estimation of Floating-Point Round-off Errors with Symbolic Taylor Expansions.
In *FM*, volume 9109 of *LNCS*. Springer.

Bibliography IV

T. Weisser, J. B. Lasserre, and K.-C. Toh.
Sparse-bsos: a bounded degree sos hierarchy for large scale polynomial optimization with sparsity.
Math. Prog. Comp., pages 1–32, 2017.