Interpolation over Nonlinear Arithmetic

Towards Program Reasoning and Verification

Mingshuai Chen

Lehrstuhl für Informatik 2 Softwaremodellierung und Verifikation | RWTH AACHEN UNIVERSITY

chenms@cs.rwth-aachen.de moves.rwth-aachen.de/people/chenms

—Joint work with J. Wang, B. Zhan, N. Zhan, D. Kapur, J. An, T. Gan, L. Dai, and B. Xia—

MOVES · November 2019

What Is Interpolation ?

Interpolation /ɪntəːpəˈleɪʃ(ə)n/

MATHEMATICS

> *"the insertion of an intermediate value or term into a series by estimating or calculating it from surrounding known values."*

[OXFORD Dictionary]

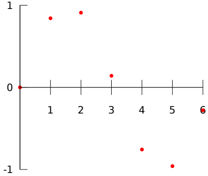
What Is Interpolation?

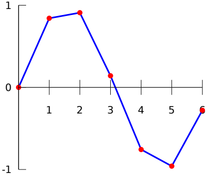
Interpolation /ɪntəːpəˈleɪʃ(ə)n/

MATHEMATICS

"the insertion of an intermediate value or term into a series by estimating or calculating it from surrounding known values."

[OXFORD Dictionary]





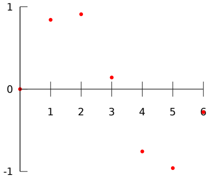
What Is Interpolation?

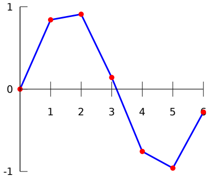
Interpolation /ɪntəːpəˈleɪʃ(ə)n/

MATHEMATICS

"the insertion of an intermediate value or term into a series by estimating or calculating it from surrounding known values."

[OXFORD Dictionary]





LOGICAL REASONING

$$P \models Q \qquad P \models \color{blue}{R} \models Q$$

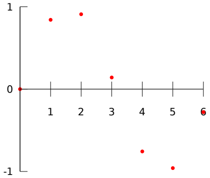
What Is Interpolation?

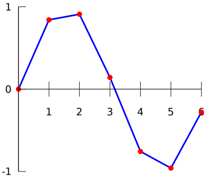
Interpolation /ɪntəːpəˈleɪʃ(ə)n/

MATHEMATICS

"the insertion of an intermediate value or term into a series by estimating or calculating it from surrounding known values."

[OXFORD Dictionary]





LOGICAL REASONING

$$P \models Q \qquad P \models R \models Q$$

$$P \wedge Q \models \bot \qquad P \models R \text{ and } R \wedge Q \models \bot$$

Interpolants as Loop Invariants

Example ([Sharma et al., CAV '12])

$x := 0; y := 0;$
while $(*)$
  $\{x := x + 1; y := y + 1; \}$
while $(x \neq 0)$
  $\{x := x - 1; y := y - 1; \}$
if $(y \neq 0)$
  error ();

Interpolants as Loop Invariants

Example ([Sharma et al., CAV '12])

$x := 0; y := 0;$

while $(*)$

$\quad \{x := x + 1; y := y + 1; \}$

_ _ _ _ _ _ _ _ _ _ _

while $(x \neq 0)$

$\quad \{x := x - 1; y := y - 1; \}$

if $(y \neq 0)$

$\quad$ **error ()**;

Interpolants as Loop Invariants

Example ([Sharma et al., CAV'12])

$x := 0; y := 0;$
while $(*)$
$\{x := x + 1; y := y + 1; \}$
$- - - - - - - - - - -$
while $(x \neq 0)$
$\{x := x - 1; y := y - 1; \}$
if $(y \neq 0)$
error ();

$$A \mathrel{\widehat{=}} x_1 = 0 \wedge y_1 = 0 \wedge \mathit{ite}\,(b,\; x = x_1 \wedge y = y_1,\; x = x_1 + 1 \wedge y = y_1 + 1)$$

Interpolants as Loop Invariants

Example ([Sharma et al., CAV '12])

$x := 0; y := 0;$
while $(*)$
  $\{x := x + 1; y := y + 1;\}$
$- - - - - - - - - - - -$
while $(x \neq 0)$
  $\{x := x - 1; y := y - 1;\}$
if $(y \neq 0)$
  error ();

$$
\begin{aligned}
A \mathrel{\widehat{=}} {} & x_1 = 0 \wedge y_1 = 0 \wedge \\
& \mathit{ite}\,(b, \\
& \quad x = x_1 \wedge y = y_1, \\
& \quad x = x_1 + 1 \wedge y = y_1 + 1) \\
B \mathrel{\widehat{=}} {} & \mathit{ite}\,(x = 0, \\
& \quad x_2 = x \wedge y_2 = y, \\
& \quad x_2 = x - 1 \wedge y_2 = y - 1) \wedge \\
& x_2 = 0 \wedge \neg(y_2 = 0)
\end{aligned}
$$

Interpolants as Loop Invariants

Example ([Sharma et al., CAV '12])

$$
\begin{array}{l}
x := 0; y := 0;\\
\text{while } (*)\\
\quad \{x := x+1; y := y+1;\}\\
\text{- - - - - - - - - - - -}\\
\text{while } (x \neq 0)\\
\quad \{x := x-1; y := y-1;\}\\
\text{if } (y \neq 0)\\
\quad \textbf{error ()};
\end{array}
$$

$$
\begin{array}{rl}
A \mathrel{\widehat{=}} & x_1 = 0 \wedge y_1 = 0 \wedge\\
& \mathit{ite}\,(b,\\
& \quad x = x_1 \wedge y = y_1,\\
& \quad x = x_1 + 1 \wedge y = y_1 + 1)\\
B \mathrel{\widehat{=}} & \mathit{ite}\,(x = 0,\\
& \quad x_2 = x \wedge y_2 = y,\\
& \quad x_2 = x - 1 \wedge y_2 = y - 1) \wedge\\
& x_2 = 0 \wedge \neg(y_2 = 0)
\end{array}
$$

$A \wedge B \models \bot.$ $\quad I(x, y) \mathrel{\widehat{=}} x = y$ s.t. $A \models I$ and $I \wedge B \models \bot.$

Interpolants as Loop Invariants

Example ([Sharma et al., CAV'12])

$x := 0; y := 0;$
while $(*)$
  $\{x := x + 1; y := y + 1; \}$
$- - - - - - - - - - -$
while $(x \neq 0)$
  $\{x := x - 1; y := y - 1; \}$
if $(y \neq 0)$
  error ();

$$
\begin{aligned}
A \mathrel{\widehat{=}}\ & x_1 = 0 \wedge y_1 = 0 \wedge \\
& ite\,(b, \\
& \quad x = x_1 \wedge y = y_1, \\
& \quad x = x_1 + 1 \wedge y = y_1 + 1) \\
B \mathrel{\widehat{=}}\ & ite\,(x = 0, \\
& \quad x_2 = x \wedge y_2 = y, \\
& \quad x_2 = x - 1 \wedge y_2 = y - 1) \wedge \\
& x_2 = 0 \wedge \neg(y_2 = 0)
\end{aligned}
$$

$$A \wedge B \models \bot. \qquad I(x, y) \mathrel{\widehat{=}} x = y \ \text{ s.t. } \ A \models I \text{ and } I \wedge B \models \bot.$$

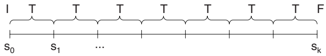
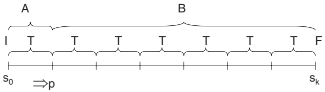


Figure – Bounded model checking.



©K. L. McMillan, 2003

Figure – Computing image by interpolation.

Interpolation-based Verification

☹ The bottleneck of existing formal verification techniques lies in scalability.

Interpolation-based Verification

☹ The bottleneck of existing formal verification techniques lies in scalability.

☺ Interpolation helps in scaling these verification techniques due to its inherent capability of local and modular reasoning :

- Nelson-Oppen method : equivalently decomposing a formula of a composite theory into formulas of its component theories;
- SMT : combining different decision procedures to verify programs with complicated data structures;
- Bounded model-checking : generating invariants to verify infinite-state systems due to McMillan;
- ...

Interpolation-based Verification

Interpolant synthesis plays the central role in interpolation-based techniques :

Interpolation-based Verification

Interpolant synthesis plays the central role in interpolation-based techniques :

☺ Well-established methods to synthesize interpolants for various theories, e.g., decidable fragments of FOL, LA, multi-sets, etc., and combinations thereof.

Interpolation-based Verification

Interpolant synthesis plays the central role in interpolation-based techniques :

☺ Well-established methods to synthesize interpolants for various theories, e.g., decidable fragments of FOL, LA, multi-sets, etc., and combinations thereof.

- SAT-based : generate interpolants for LA from (resolution) unsatisfiability proofs.
  - ⇒ McMillan, K. L. : *Interpolation and SAT-based model checking.* CAV '03.

Interpolation-based Verification

Interpolant synthesis plays the central role in interpolation-based techniques :

☺ Well-established methods to synthesize interpolants for various theories, e.g., decidable fragments of FOL, LA, multi-sets, etc., and combinations thereof.

- SAT-based : generate interpolants for LA from (resolution) unsatisfiability proofs.
  - ⇒ McMillan, K. L. : *Interpolation and SAT-based model checking*. CAV '03.
- Constraint solving-based : reduce interpolation for LA to linear programming by Motzkin's transposition theorem.
  - ⇒ Rybalchenko, A., Sofronie-Stokkermans, V. : *Constraint solving for interpolation*. J. Symb. Comput. '10.

Interpolation-based Verification

Interpolant synthesis plays the central role in interpolation-based techniques :

☺ Well-established methods to synthesize interpolants for various theories, e.g., decidable fragments of FOL, LA, multi-sets, etc., and combinations thereof.

- SAT-based : generate interpolants for LA from (resolution) unsatisfiability proofs.
  - ⇒ McMillan, K. L. : *Interpolation and SAT-based model checking.* CAV '03.
- Constraint solving-based : reduce interpolation for LA to linear programming by Motzkin's transposition theorem.
  - ⇒ Rybalchenko, A., Sofronie-Stokkermans, V. : *Constraint solving for interpolation.* J. Symb. Comput. '10.

☹ Little work on synthesizing nonlinear ones : [Kupferschmid & Becker, FORMATS '11], [Dai et al., CAV '13], [Gao & Zufferey, TACAS '16], [Okudono et al., APLAS '17].

Interpolation-based Verification

Interpolant synthesis plays the central role in interpolation-based techniques :

☺ Well-established methods to synthesize interpolants for various theories, e.g., decidable fragments of FOL, LA, multi-sets, etc., and combinations thereof.

- SAT-based : generate interpolants for LA from (resolution) unsatisfiability proofs.
  - ⇒ McMillan, K. L. : *Interpolation and SAT-based model checking.* CAV '03.
- Constraint solving-based : reduce interpolation for LA to linear programming by Motzkin's transposition theorem.
  - ⇒ Rybalchenko, A., Sofronie-Stokkermans, V. : *Constraint solving for interpolation.* J. Symb. Comput. '10.

☹ Little work on synthesizing nonlinear ones : [Kupferschmid & Becker, FORMATS '11], [Dai et al., CAV '13], [Gao & Zufferey, TACAS '16], [Okudono et al., APLAS '17].

- Reduce interpolation for concave quadratic polynomial inequalities to semi-definite programming. Tool : `NLFIntp`.
  - ⇒ Gan, T., Dai, L., Xia, B., Zhan, N., Kapur, D., Chen, M. : *Interpolation synthesis for quadratic polynomial inequalities and combination with EUF.* IJCAR '16.

Interpolation-based Verification

Interpolant synthesis plays the central role in interpolation-based techniques :

☺ Well-established methods to synthesize interpolants for various theories, e.g., decidable fragments of FOL, LA, multi-sets, etc., and combinations thereof.

- SAT-based : generate interpolants for LA from (resolution) unsatisfiability proofs.
  - ⇒ McMillan, K. L. : *Interpolation and SAT-based model checking.* CAV'03.
- Constraint solving-based : reduce interpolation for LA to linear programming by Motzkin's transposition theorem.
  - ⇒ Rybalchenko, A., Sofronie-Stokkermans, V. : *Constraint solving for interpolation.* J. Symb. Comput.'10.

☹ Little work on synthesizing nonlinear ones : [Kupferschmid & Becker, FORMATS'11], [Dai et al., CAV'13], [Gao & Zufferey, TACAS'16], [Okudono et al., APLAS'17].

- Reduce interpolation for concave quadratic polynomial inequalities to semi-definite programming. Tool : `NLFIntp`.
  - ⇒ Gan, T., Dai, L., Xia, B., Zhan, N., Kapur, D., Chen, M. : *Interpolation synthesis for quadratic polynomial inequalities and combination with EUF.* IJCAR'16.
- Counterexample-guided learning of polynomial interpolants for the general quantifier-free theory of NLA. Tool : `NIL`.
  - ⇒ Chen, M., Wang, J., An, J., Zhan, B., Kapur, D., Zhan, N. : *NIL : Learning nonlinear interpolants.*. CADE'19.

Interpolation-based Verification

Interpolant synthesis plays the central role in interpolation-based techniques :

☺ Well-established methods to synthesize interpolants for various theories, e.g., decidable fragments of FOL, LA, multi-sets, etc., and combinations thereof.

- SAT-based : generate interpolants for LA from (resolution) unsatisfiability proofs.
  - ⇒ McMillan, K. L. : *Interpolation and SAT-based model checking.* CAV'03.
- Constraint solving-based : reduce interpolation for LA to linear programming by Motzkin's transposition theorem.
  - ⇒ Rybalchenko, A., Sofronie-Stokkermans, V. : *Constraint solving for interpolation.* J. Symb. Comput.'10.

☹ Little work on synthesizing nonlinear ones : [Kupferschmid & Becker, FORMATS'11], [Dai et al., CAV'13], [Gao & Zufferey, TACAS'16], [Okudono et al., APLAS'17].

- Reduce interpolation for concave quadratic polynomial inequalities to semi-definite programming. Tool : `NLFIntp`.
  - ⇒ Gan, T., Dai, L., Xia, B., Zhan, N., Kapur, D., Chen, M. : *Interpolation synthesis for quadratic polynomial inequalities and combination with EUF.* IJCAR'16.
- Counterexample-guided learning of polynomial interpolants for the general quantifier-free theory of NLA. Tool : `NIL`.
  - ⇒ Chen, M., Wang, J., An, J., Zhan, B., Kapur, D., Zhan, N. : *NIL : Learning nonlinear interpolants.*. CADE'19.

Outline

Outline

Craig Interpolation

Craig Interpolant

Given ϕ and ψ in a theory $\mathcal{T}$ s.t. $\phi \wedge \psi \models_{\mathcal{T}} \bot$, a formula I is a *(reverse) interpolant* of ϕ and ψ if (1) $\phi \models_{\mathcal{T}} I$; (2) $I \wedge \psi \models_{\mathcal{T}} \bot$; and (3) $var(I) \subseteq var(\phi) \cap var(\psi)$.

Craig Interpolation

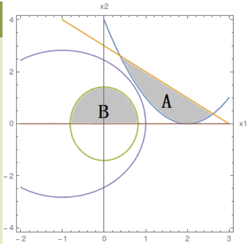
Craig Interpolant

Given ϕ and ψ in a theory $\mathcal{T}$ s.t. $\phi \wedge \psi \models_{\mathcal{T}} \bot$, a formula I is a *(reverse) interpolant* of ϕ and ψ if (1) $\phi \models_{\mathcal{T}} I$; (2) $I \wedge \psi \models_{\mathcal{T}} \bot$; and (3) $var(I) \subseteq var(\phi) \cap var(\psi)$.

Example (over nonlinear $\mathcal{T}$)

$$A \mathrel{\widehat{=}} -{x_1}^2 + 4x_1 + x_2 - 4 \geq 0 \wedge -x_1 - x_2 + 3 - y^2 > 0$$
$$B \mathrel{\widehat{=}} -3{x_1}^2 - {x_2}^2 + 1 \geq 0 \wedge x_2 - z^2 \geq 0$$

$$I \mathrel{\widehat{=}} -3 + 2x_1 + {x_1}^2 + \frac{1}{2}{x_2}^2 > 0$$



Binary Classification

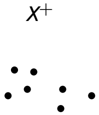
Binary Classification

Given a training dataset $X = X^{+} \uplus X^{-}$ of positive/negative sample points, find a classifier $C\colon X \mapsto \{\top, \bot\}$, s.t. (1) $\forall \vec{x} \in X^{+}.\ C(\vec{x}) = \top$; and (2) $\forall \vec{x} \in X^{-}.\ C(\vec{x}) = \bot$.

Binary Classification

Binary Classification

Given a training dataset $X = X^+ \uplus X^-$ of positive/negative sample points, find a classifier $C\colon X \mapsto \{\top, \bot\}$, s.t. (1) $\forall \vec{x} \in X^+ .\ C(\vec{x}) = \top$; and (2) $\forall \vec{x} \in X^- .\ C(\vec{x}) = \bot$.



Binary Classification

Binary Classification

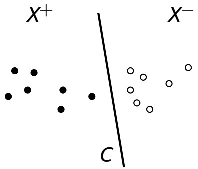
Given a training dataset $X = X^{+} \uplus X^{-}$ of positive/negative sample points, find a classifier $C\colon X \mapsto \{\top, \bot\}$, s.t. (1) $\forall \vec{x} \in X^{+}.\ C(\vec{x}) = \top$; and (2) $\forall \vec{x} \in X^{-}.\ C(\vec{x}) = \bot$.

X^{+} X^{-}

Binary Classification

Binary Classification

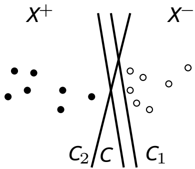
Given a training dataset $X = X^+ \uplus X^-$ of positive/negative sample points, find a classifier $C\colon X \mapsto \{\top, \bot\}$, s.t. (1) $\forall \vec{x} \in X^+ .\ C(\vec{x}) = \top$; and (2) $\forall \vec{x} \in X^- .\ C(\vec{x}) = \bot$.



Binary Classification

Binary Classification

Given a training dataset $X = X^+ \uplus X^-$ of positive/negative sample points, find a classifier $C\colon X \mapsto \{\top, \bot\}$, s.t. (1) $\forall \vec{x} \in X^+ .\ C(\vec{x}) = \top$; and (2) $\forall \vec{x} \in X^- .\ C(\vec{x}) = \bot$.

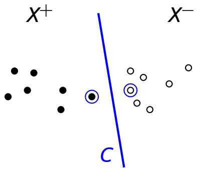


There could be (infinitely) many valid classifiers.

Binary Classification

Binary Classification

Given a training dataset $X = X^+ \uplus X^-$ of positive/negative sample points, find a classifier $C\colon X \mapsto \{\top, \bot\}$, s.t. (1) $\forall \vec{x} \in X^+ .\ C(\vec{x}) = \top$; and (2) $\forall \vec{x} \in X^- .\ C(\vec{x}) = \bot$.



Support Vector Machine (SVM) finds a separating hyperplane that yields the largest distance (functional margin) to the nearest positive and negative samples (support vectors), which boils down to convex optimizations.

Interpolation vs. Classification

☺ Linear interpolants can be viewed as hyperplane classifiers, [Sharma et al., CAV'12] : sampling from $[\![\phi]\!]$ and $[\![\psi]\!]$ $\rightarrow$ building a hyperplane classifier $\rightarrow$ refining by CEs.

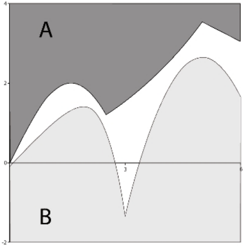
Interpolation vs. Classification

☺ Linear interpolants can be viewed as hyperplane classifiers, [Sharma et al., CAV'12] : sampling from $[\![\phi]\!]$ and $[\![\psi]\!]$ $\rightarrow$ building a hyperplane classifier $\rightarrow$ refining by CEs.

☹ X^+ and X^- might not be linearly separable (often the case when sampled from nonlinear ϕ and ψ, resp.) :

$$A \quad \widehat{=} \quad (x < 2.5 \Rightarrow y \geq 2\sin(x)) \wedge (x \geq 2.5 \wedge x < 5 \Rightarrow y \geq 0.125x^2 + 0.41) \wedge (x \geq 5 \wedge x \leq 6 \Rightarrow y \geq 6.04 - 0.5x)$$

$$B \quad \widehat{=} \quad (x < 3 \Rightarrow y \leq x\cos(0.1e^x) - 0.083) \wedge (x \geq 3 \wedge x \leq 6 \Rightarrow y \leq -x^2 + 10x - 22.35)$$



©Kupferschmid & Becker, FORMATS'11

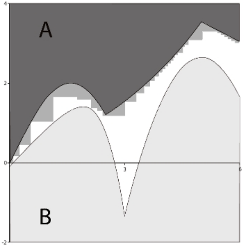
Interpolation vs. Classification

☺ Linear interpolants can be viewed as hyperplane classifiers, [Sharma et al., CAV'12] : sampling from $[\![\phi]\!]$ and $[\![\psi]\!]$ $\rightarrow$ building a hyperplane classifier $\rightarrow$ refining by CEs.

☹ X^+ and X^- might not be linearly separable (often the case when sampled from nonlinear ϕ and ψ, resp.) :

$$A \;\widehat{=}\; (x < 2.5 \Rightarrow y \geq 2\sin(x)) \wedge(x \geq 2.5 \wedge x < 5 \Rightarrow y \geq 0.125x^2 + 0.41) \wedge(x \geq 5 \wedge x \leq 6 \Rightarrow y \geq 6.04 - 0.5x)$$

$$B \;\widehat{=}\; (x < 3 \Rightarrow y \leq x\cos(0.1e^x) - 0.083) \wedge(x \geq 3 \wedge x \leq 6 \Rightarrow y \leq -x^2 + 10x - 22.35)$$



©Kupferschmid & Becker, FORMATS'11

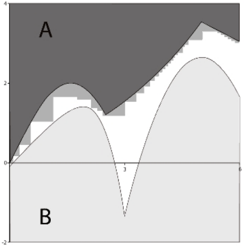
☺ Encoding interpolants as logical combinations of linear constraints.

Interpolation vs. Classification

☺ Linear interpolants can be viewed as hyperplane classifiers, [Sharma et al., CAV'12] : sampling from $[\![\phi]\!]$ and $[\![\psi]\!]$ $\rightarrow$ building a hyperplane classifier $\rightarrow$ refining by CEs.

☹ X^+ and X^- might not be linearly separable (often the case when sampled from nonlinear ϕ and ψ, resp.) :

$$
\begin{aligned}
A \quad &\widehat{=} \quad (x < 2.5 \Rightarrow y \geq 2\sin(x)) \\
&\quad \wedge (x \geq 2.5 \wedge x < 5 \Rightarrow y \geq 0.125x^2 + 0.41) \\
&\quad \wedge (x \geq 5 \wedge x \leq 6 \Rightarrow y \geq 6.04 - 0.5x) \\
B \quad &\widehat{=} \quad (x < 3 \Rightarrow y \leq x\cos(0.1e^x) - 0.083) \\
&\quad \wedge (x \geq 3 \wedge x \leq 6 \Rightarrow y \leq -x^2 + 10x - 22.35)
\end{aligned}
$$



©Kupferschmid & Becker, FORMATS'11

☺ Encoding interpolants as logical combinations of linear constraints.

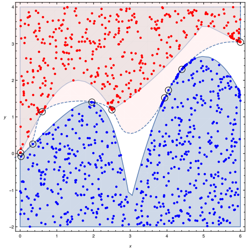
☹ Yielding rather complex interpolants (even of an infinite length in the worst case).

Interpolation vs. Classification

☺ Linear interpolants can be viewed as hyperplane classifiers, [Sharma et al., CAV'12] : sampling from $[\![\phi]\!]$ and $[\![\psi]\!]$ $\rightarrow$ building a hyperplane classifier $\rightarrow$ refining by CEs.

☹ X^+ and X^- might not be linearly separable (often the case when sampled from nonlinear ϕ and ψ, resp.) :

$$
\begin{aligned}
A \quad \widehat{=} \quad & (x < 2.5 \Rightarrow y \geq 2\sin(x)) \\
& \wedge (x \geq 2.5 \wedge x < 5 \Rightarrow y \geq 0.125x^2 + 0.41) \\
& \wedge (x \geq 5 \wedge x \leq 6 \Rightarrow y \geq 6.04 - 0.5x) \\
B \quad \widehat{=} \quad & (x < 3 \Rightarrow y \leq x\cos(0.1e^x) - 0.083) \\
& \wedge (x \geq 3 \wedge x \leq 6 \Rightarrow y \leq -x^2 + 10x - 22.35)
\end{aligned}
$$



©Chen et al., CADE'19

☺ Encoding interpolants as logical combinations of linear constraints.

☹ Yielding rather complex interpolants (even of an infinite length in the worst case).

☺ NIL : learning nonlinear interpolants.

Outline

Space Transformation & Kernel Trick

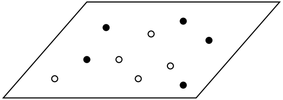


Figure – 2-dimensional input space

Space Transformation & Kernel Trick

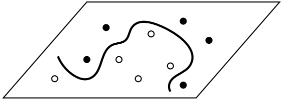


Figure – 2-dimensional input space

Space Transformation & Kernel Trick

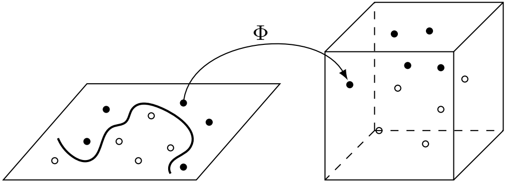


Figure – 2-dimensional input space $\mapsto$ 3-dimensional feature (monomial) space with linear separation.

Space Transformation & Kernel Trick

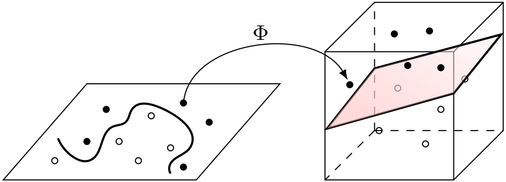


Figure – 2-dimensional input space $\mapsto$ 3-dimensional feature (monomial) space with linear separation.

Space Transformation & Kernel Trick

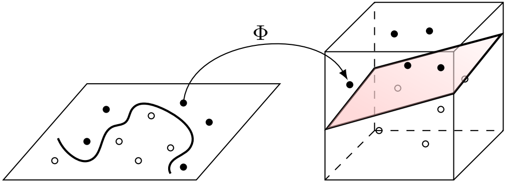


Figure – 2-dimensional input space $\mapsto$ 3-dimensional feature (monomial) space with linear separation.

Optimal-margin classifier I:

$$\sum_{i=1}^{n} \alpha_i \kappa(\vec{x}_i, \mathbf{x}) \qquad = 0$$

Space Transformation & Kernel Trick

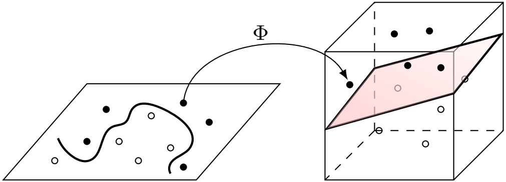


Figure – 2-dimensional input space $\mapsto$ 3-dimensional feature (monomial) space with linear separation.

Optimal-margin classifier I:

kernel function

$$\sum_{i=1}^{n} \alpha_i \kappa(\vec{x}_i, \mathbf{x}) = 0$$

support vectors

Space Transformation & Kernel Trick

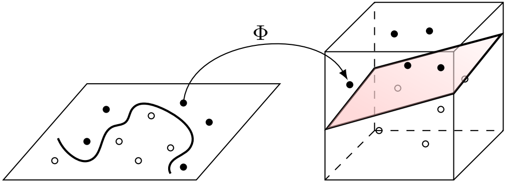


Figure – 2-dimensional input space ↦ 3-dimensional feature (monomial) space with linear separation.

Optimal-margin classifier I:

kernel function

$$\sum_{i=1}^{n} \alpha_i \kappa(\vec{x}_i, \mathbf{x}) = \Phi(\vec{x}_i)^{\mathrm{T}} \Phi(\mathbf{x}) \qquad = 0$$

support vectors

Space Transformation & Kernel Trick

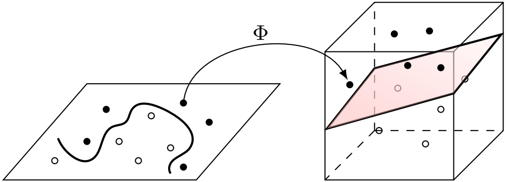


Figure – 2-dimensional input space $\mapsto$ 3-dimensional feature (monomial) space with linear separation.

Optimal-margin classifier I:

kernel function

$$\sum_{i=1}^{n} \alpha_i \kappa(\vec{x}_i, \mathbf{x}) = \Phi(\vec{x}_i)^{\mathrm{T}} \Phi(\mathbf{x}) = (\beta \vec{x}_i^{\mathrm{T}} \mathbf{x} + \theta)^m = 0$$

support vectors

Space Transformation & Kernel Trick

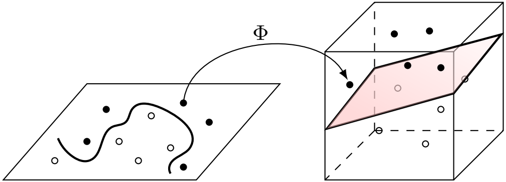


Figure – 2-dimensional input space $\mapsto$ 3-dimensional feature (monomial) space with linear separation.

Optimal-margin classifier I:

kernel function

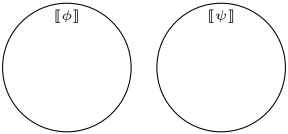
$$\sum_{i=1}^{n} \alpha_i \kappa(\vec{x}_i, \mathbf{x}) = \Phi(\vec{x}_i)^{\mathrm{T}} \Phi(\mathbf{x}) = (\beta \vec{x}_i^{\mathrm{T}} \mathbf{x} + \theta)^m = 0$$

support vectors

polynomial degree describing
complexity of the monomial space

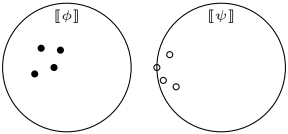
The NIL Algorithm

1. Given mutually contradictory nonlinear ϕ and ψ over common variables $\mathbf{x}$.
2. Generate sample points by, e.g., (uniformly) scattering random points.
3. Find a classifier by SVMs (with kernel-degree *m*) as a candidate interpolant.
4. Refine the candidate by CEs till it being verified as a true interpolant.



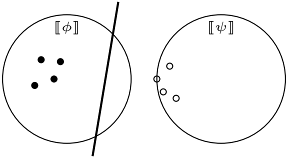
The NIL Algorithm

1. Given mutually contradictory nonlinear ϕ and ψ over common variables $\mathbf{x}$.
2. Generate sample points by, e.g., (uniformly) scattering random points.
3. Find a classifier by SVMs (with kernel-degree *m*) as a candidate interpolant.
4. Refine the candidate by CEs till it being verified as a true interpolant.



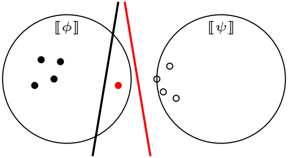
The NIL Algorithm

1. Given mutually contradictory nonlinear ϕ and ψ over common variables $\mathbf{x}$.
2. Generate sample points by, e.g., (uniformly) scattering random points.
3. Find a classifier by SVMs (with kernel-degree *m*) as a candidate interpolant.
4. Refine the candidate by CEs till it being verified as a true interpolant.



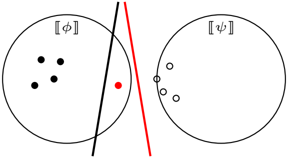
The NIL Algorithm

1. Given mutually contradictory nonlinear ϕ and ψ over common variables $\mathbf{x}$.
2. Generate sample points by, e.g., (uniformly) scattering random points.
3. Find a classifier by SVMs (with kernel-degree *m*) as a candidate interpolant.
4. Refine the candidate by CEs till it being verified as a true interpolant.



The NIL Algorithm

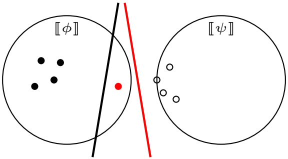
1. Given mutually contradictory nonlinear ϕ and ψ over common variables $\mathbf{x}$.
2. Generate sample points by, e.g., (uniformly) scattering random points.
3. Find a classifier by SVMs (with kernel-degree *m*) as a candidate interpolant.
4. Refine the candidate by CEs till it being verified as a true interpolant.



☺ Sound, and complete when $[\![\phi]\!]$ and $[\![\psi]\!]$ are bounded sets with positive functional margin.

The NIL Algorithm

1. Given mutually contradictory nonlinear ϕ and ψ over common variables $\mathbf{x}$.
2. Generate sample points by, e.g., (uniformly) scattering random points.
3. Find a classifier by SVMs (with kernel-degree *m*) as a candidate interpolant.
4. Refine the candidate by CEs till it being verified as a true interpolant.



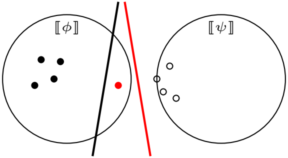
☺ Sound, and complete when $[\![\phi]\!]$ and $[\![\psi]\!]$ are bounded sets with positive functional margin.

☹ Quantifier Elimination (QE) is involved in checking interpolants and generating CEs [1].

1. SMT-solving techniques over nonlinear arithmetic do not suffice.

The NIL Algorithm

1. Given mutually contradictory nonlinear ϕ and ψ over common variables $\mathbf{x}$.
2. Generate sample points by, e.g., (uniformly) scattering random points.
3. Find a classifier by SVMs (with kernel-degree *m*) as a candidate interpolant.
4. Refine the candidate by CEs till it being verified as a true interpolant.



☺ Sound, and complete when $[\![\phi]\!]$ and $[\![\psi]\!]$ are bounded sets with positive functional margin.

☹ Quantifier Elimination (QE) is involved in checking interpolants and generating CEs [1].

☹ May not terminate in cases with zero functional margin.

1. SMT-solving techniques over nonlinear arithmetic do not suffice.

Comparison with Naïve QE-Based Method

	QE-based method	NIL
Logical strength	strongest : $\cancel{\exists \mathbf{y}}.\ \phi(\mathbf{x}, \mathbf{y})$ weakest : $\cancel{\forall \mathbf{z}}.\ \neg\psi(\mathbf{x}, \mathbf{z})$	medium $\Rightarrow$ robust
Complexity of I	direct projection $\Rightarrow$ complex	single polynomial $\Rightarrow$ simple
Efficiency	doubly exponential	$n \times$ doubly exponential

Comparison with Naïve QE-Based Method

	QE-based method	**NIL**
Logical strength	strongest : $\cancel{\exists \mathbf{y}}.\ \phi(\mathbf{x}, \mathbf{y})$ weakest : $\cancel{\forall \mathbf{z}}.\ \neg\psi(\mathbf{x}, \mathbf{z})$	medium $\Rightarrow$ robust
Complexity of I	direct projection $\Rightarrow$ complex	single polynomial $\Rightarrow$ simple
Efficiency	doubly exponential	$n \times$ doubly exponential

Comparison with Naïve QE-Based Method

	QE-based method	NIL
Logical strength	strongest : $\cancel{\exists \mathbf{y}}.\ \phi(\mathbf{x}, \mathbf{y})$ weakest : $\cancel{\forall \mathbf{z}}.\ \neg\psi(\mathbf{x}, \mathbf{z})$	medium $\Rightarrow$ robust
Complexity of I	direct projection $\Rightarrow$ complex	single polynomial $\Rightarrow$ simple
Efficiency	doubly exponential	$n \times$ doubly exponential

Comparison with Naïve QE-Based Method

	QE-based method	NIL
Logical strength	strongest : $\cancel{\exists \mathbf{y}}.\ \phi(\mathbf{x}, \mathbf{y})$ weakest : $\cancel{\forall \mathbf{z}}.\ \neg\psi(\mathbf{x}, \mathbf{z})$	medium $\Rightarrow$ robust
Complexity of *I*	direct projection $\Rightarrow$ complex	single polynomial $\Rightarrow$ simple
Efficiency	doubly exponential	$n \times$ doubly exponential

Comparison with Naïve QE-Based Method

	QE-based method	**NIL**
Logical strength	strongest : $\cancel{\exists \mathbf{y}}.\ \phi(\mathbf{x}, \mathbf{y})$ weakest : $\cancel{\forall \mathbf{z}}.\ \neg\psi(\mathbf{x}, \mathbf{z})$	medium $\Rightarrow$ robust
Complexity of I	direct projection $\Rightarrow$ complex	single polynomial $\Rightarrow$ simple
Efficiency	doubly exponential	$n \times$ doubly exponential

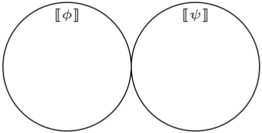
QE + template ?

Comparison with Naïve QE-Based Method

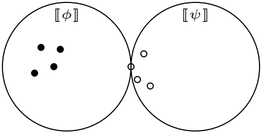
	QE-based method	**NIL**
Logical strength	strongest : $\cancel{\exists \mathbf{y}}.\ \phi(\mathbf{x},\mathbf{y})$ weakest : $\cancel{\forall \mathbf{z}}.\ \neg\psi(\mathbf{x},\mathbf{z})$	medium $\Rightarrow$ robust
Complexity of I	direct projection $\Rightarrow$ complex	single polynomial $\Rightarrow$ simple
Efficiency	doubly exponential	$n \times$ doubly exponential

QE + template ? $\Rightarrow$ Too many unknown parameters.

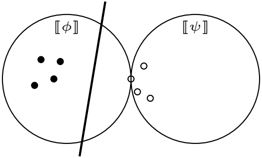
NIL$_\delta$: For Cases with Zero Functional Margin



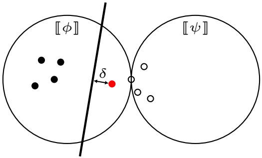
NIL$_\delta$: For Cases with Zero Functional Margin



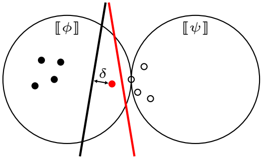
NIL$_\delta$: For Cases with Zero Functional Margin



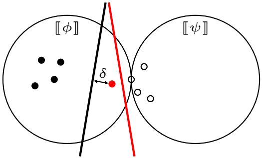
NIL_δ : For Cases with Zero Functional Margin



NIL_δ : For Cases with Zero Functional Margin

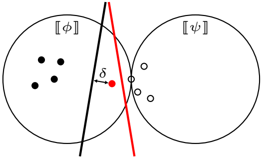


NIL$_\delta$: For Cases with Zero Functional Margin



☺ δ-sound, and δ-complete if $[\![\phi]\!]$ and $[\![\psi]\!]$ are bounded sets even with zero functional margin.

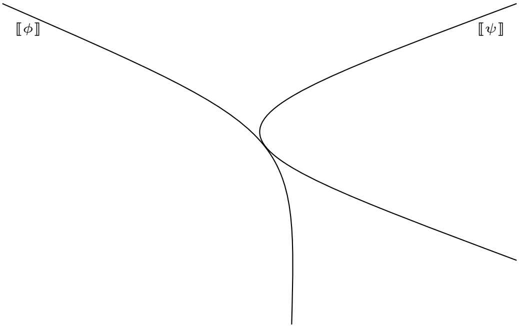
NIL$_\delta$: For Cases with Zero Functional Margin



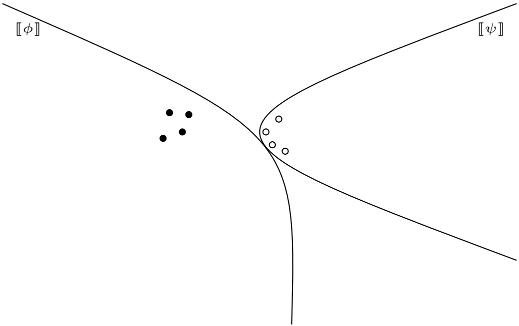
☺ δ-sound, and δ-complete if $[\![\phi]\!]$ and $[\![\psi]\!]$ are bounded sets even with zero functional margin.

☹ May not converge to an actual interpolant when $[\![\phi]\!]$ or $[\![\psi]\!]$ is unbounded.

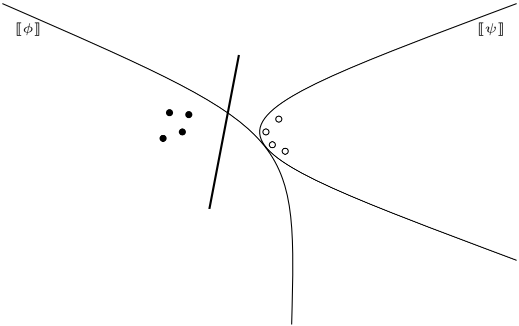
$\mathrm{NIL}^{*}_{\delta,B}$: For Unbounded Cases with Varying Tolerance



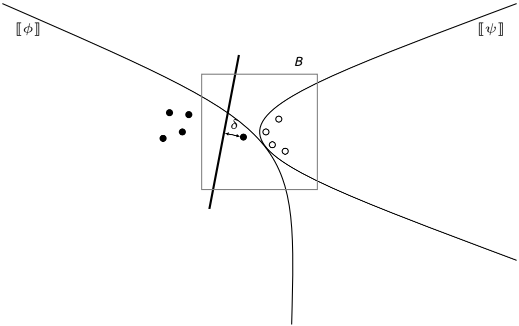
$\mathrm{NIL}^*_{\delta,B}$: For Unbounded Cases with Varying Tolerance



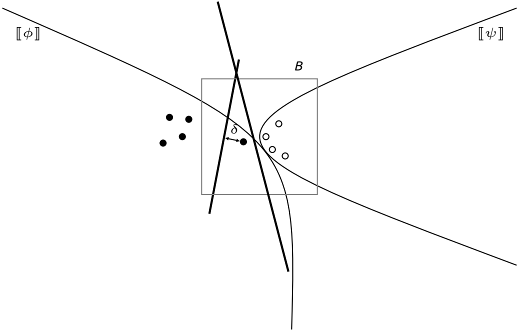
$\text{NIL}^*_{\delta,B}$: For Unbounded Cases with Varying Tolerance



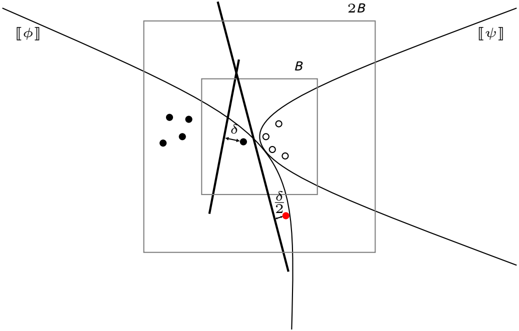
$\mathrm{NIL}^*_{\delta,B}$: For Unbounded Cases with Varying Tolerance



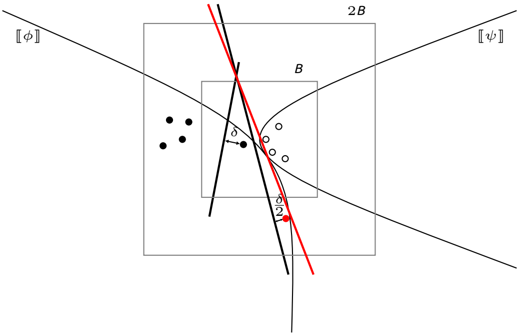
$\text{NIL}^*_{\delta,B}$: For Unbounded Cases with Varying Tolerance



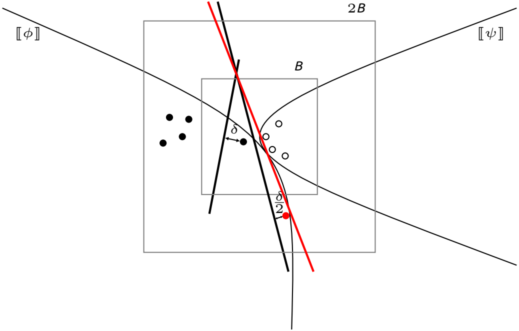
$\text{NIL}^*_{\delta,B}$: For Unbounded Cases with Varying Tolerance



$\mathrm{NIL}^*_{\delta,B}$: For Unbounded Cases with Varying Tolerance



$\mathrm{NIL}^*_{\delta,B}$: For Unbounded Cases with Varying Tolerance



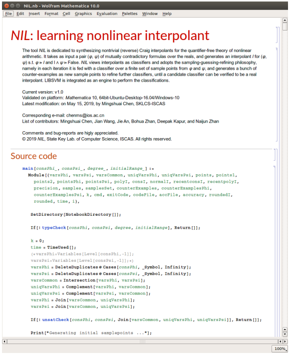
☺ The sequence of candidate interpolants converges to an actual interpolant.

Outline

Implementation Issues

NIL : an open-source tool in Wolfram Mathematica.

- LIBSVM : SVM classifications;
- Reduce [2] : verification of candidate interpolants;
- FindInstance : generation of counterexamples;
- Rational recovery : rounding off floating-point computations [Lang, Springer NY '12].



©*NIL*, 2019

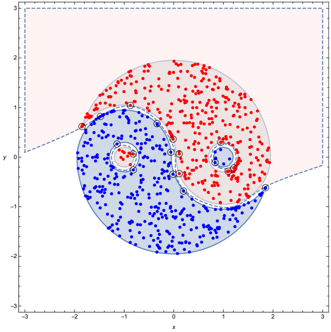
2. CAD implementation for quantifier-free fragment of a first-order theory of polynomials over the reals and its appropriate extension to transcendental functions [Strzeboński, J. Symb. Comput. '11].

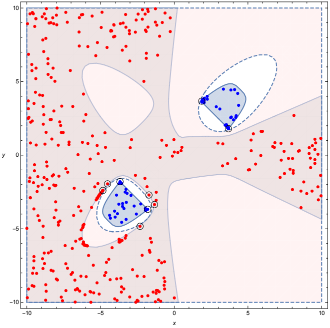
Benchmark Examples

Category	ID	Name	ϕ	ψ	I	Time/s
	1	Dummy	$x < -1$	$x \geq 1$	$x < 0$	0.11
	2	Necklace	$y - x^2 - 1 = 0$	$y + x^2 + 1 = 0$	$-y < 0$	0.21
	3	Face	$(x+4)^2 + y^2 - 1 \leq 0 \vee$ $(x-4)^2 + y^2 - 1 \leq 0$	$x^2 + y^2 - 64 \leq 0 \wedge$ $(x+4)^2 + y^2 - 9 \geq 0 \wedge$ $(x-4)^2 + y^2 - 9 \geq 0$	$\frac{x^4}{223} - \frac{x^3 y}{356} + x^2(\frac{y^2}{45} - \frac{y}{170} - \frac{2}{9}) +$ $x(\frac{y^3}{89} + \frac{y^2}{68} - \frac{y}{74} - \frac{1}{55}) + \frac{y^4}{146} +$ $\frac{y^3}{95} + \frac{y^2}{37} + \frac{y}{366} + 1 < 0$	0.33
	4	Twisted	$x^2 - 2xy^2 + 3xz - y^2$ $- yz + z^2 - 1 \geq 0 \wedge$ $\frac{1}{120}\left(-x^6 - y^6\right) + x^2 z^2 -$ $x^2 + \frac{1}{6}\left(x^4 + 2x^2 y^2 + y^4\right) +$ $y^2 z^2 - y^2 - 4 \leq 0$	$w^2 + 4(x-y)^4 + (x+y)^2 - 80 \leq 0 \wedge$ $- w^2 (x-y)^4 + 100(x+y)^2 - 3000 \geq 0$	$-\frac{x^4}{160} + x^3\left(\frac{y}{170} - \frac{1}{113}\right) + x^2\left(-\frac{y^2}{225} + \frac{y}{76} + \frac{2}{27}\right) +$ $x\left(\frac{y^3}{259} + \frac{y^2}{63} + \frac{5y}{51} - \frac{1}{316}\right) - \frac{y^4}{183} - \frac{y^3}{94} + \frac{y^2}{14} + \frac{y}{255} - 1 < 0$	140.62
with/without rounding	5	Ultimate	$(x^2 + y^2 - 3.8025 \leq 0 \wedge y \geq 0 \vee$ $(x-1)^2 + y^2 - 0.9025 \leq 0) \wedge$ $(x-1)^2 + y^2 - 0.09 > 0 \wedge$ $(x+1)^2 + y^2 - 1.1025 \geq 0 \vee$ $(x+1)^2 + y^2 - \frac{1}{25} \leq 0$	$(-3.8025 + x^2 + y^2 \leq 0 \wedge -y \geq 0 \vee$ $- 0.9025 + (-1-x)^2 + y^2 \leq 0) \wedge$ $- 0.09 + (-1-x)^2 + y^2 > 0 \wedge$ $- 1.1025 + (1-x)^2 + y^2 \geq 0 \vee$ $- \frac{1}{25} + (1-x)^2 + y^2 \leq 0$	$\frac{x^7}{27} + x^6(-\frac{y}{5} - \frac{1}{96}) + x^5(\frac{2y^2}{9} - \frac{y}{32} - \frac{1}{2}) +$ $x^4(-\frac{2y^3}{9} + \frac{y}{3} + \frac{1}{31}) + x^3(\frac{y^4}{11} + \frac{y^3}{10} - \frac{10y^2}{13} + \frac{y}{18} + \frac{15}{16}) +$ $x^2(-\frac{y^5}{25} - \frac{y^4}{18} - \frac{y^3}{3} + \frac{y^2}{10} - \frac{1}{32}) +$ $x\left(\frac{y^6}{71} + \frac{2y^4}{11} - \frac{y^3}{25} - y^2 - \frac{y}{45} - \frac{3}{8}\right) +$ $\frac{y^6}{48} - \frac{y^5}{7} + \frac{y^4}{6} - \frac{y^3}{2} - \frac{y^2}{6} - \frac{y}{59} + \frac{1}{85} < 0$	48.82
	6	IJCAR16-1	$- x_1^2 + 4x_1 + x_2 - 4 \geq 0 \wedge$ $- x_1 - x_2 + 3 - y^2 > 0$	$-3x_1^2 - x_2^2 + 1 \geq 0 \wedge x_2 - z^2 \geq 0$	$1 - \frac{3x_1}{4} - \frac{x_2}{2} < 0$	0.16
	7	CAV13-1	$1 - a^2 - b^2 > 0 \wedge a^2 + b - 1 - x = 0 \wedge$ $b + bx + 1 - y = 0$	$x^2 - 2y^2 - 4 > 0$	$-1 + \frac{x^2}{2} - \frac{y}{3} + \frac{xy}{3} - \frac{y^2}{4} < 0$	3.25
	8	CAV13-2	$x^2 + y^2 + z^2 - 2 \geq 0 \wedge$ $1.2x^2 + y^2 + xz = 0$	$20 - 3x^2 - 4y^3 - 10z^2 \geq 0 \wedge$ $x^2 + y^2 - z - 1 = 0$	$105x^4 + x^2(140y^2 + 24y(5z+7) + 35z(3z+8)) +$ $2(70y^3 z + 5y^2(12z^2 + 21z + 28) - 14y(6z^3 + 5z^2 +$ $10) - 35(3z^4 + 8z^2 + 4z - 9)) < 14x(20x^2(z+1) +$ $10y^2(z+2) - 3y(4z^2 - 5z + 4) - 20z(z^2 + 2))$	3857.89
	9	CAV13-3	$vc < 49.61 \wedge fa = 0.5418vc^2 \wedge$ $fr = 1000 - fa \wedge ac = 0.0005fr \wedge$ $vc_1 = vc + ac$	$vc_1 \geq 49.61$	$-1 + \frac{2vc_1}{99} < 0$	40.63
	10	Parallel parabola	$y - x^2 - 1 \geq 0$	$y - x^2 < 0$	$\frac{1}{2} + x^2 < y$	4.50
	11	Parallel halfplane	$y - x - 1 \geq 0$	$y - x + 1 < 0$	$x < y$	2.46
	12	Sharper-1	$y + 1 < 0$	$x^2 + y^2 - 1 \leq 0$	$2 + y < y^2$	2.19
	13	Sharper-2	$y - x > 0 \wedge x + y > 0$	$y + x^2 <= 0$	$y > 0$	2.38
	14	Coincident	$x + y > 0 \vee x + y < 0$	$x + y = 0$	$(x+y)^2 > 0$	0.18
	15	Adjacent	$y - x^2 > 0$	$y - x^2 <= 0$	$x^2 < y$	0.25
with rounding	16	IJCAR16-2	$- y_1 + x_1 - 2 \geq 0 \wedge 2x_2 - x_1 - 1 > 0 \wedge$ $- y_1^2 - x_1^2 + 2x_1 y_1 - 2y_1 + 2x_1 \geq 0 \wedge$ $- y_2^2 - y_1^2 - x_2^2 - 4y_1 + 2x_2 - 4 \geq 0$	$- z_1 + 2x_2 + 1 \geq 0 \wedge 2x_1 - x_2 - 1 > 0 \wedge$ $- z_1^2 - 4x_2^2 + 4x_2 z_1 + 3z_1 - 6x_2 - 2 \geq 0 \wedge$ $- z_2^2 - x_1^2 - x_2^2 + 2x_1 + z_1 - 2x_2 - 1 \geq 0$	$x_1 < x_2$	12.33
	17	CAV13-4	$xa_1 + 2ya_1 \geq 0 \wedge xa_1 + 2ya_1 - x_1 = 0 \wedge$ $- 2xa_1 + ya_1 - y_1 = 0 \wedge x - x_1 - 1 = 0 \wedge$ $y = y_1 + x \wedge xa = x - 2y \wedge ya = 2x + y$	$xa + 2ya < 0$	$2xa + 4ya > 5$	3.10
beyond polynomials	18	TACAS16	$y - x^2 \geq 0$	$y + \cos x - 0.8 \leq 0$	$15x^2 < 4 + 20y$	12.71
	19	Transcendental	$\sin x \geq 0.6$	$\sin x \leq 0.4$	SVM failed	–
unbalanced	20	Unbalanced	$x > 0 \vee x < 0$	$x = 0$	$x^2 > 0$	0.11

Visualizations in *NIL*

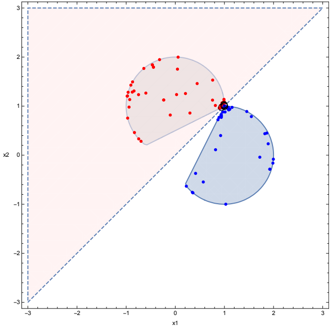
Beyond the scope of concave quadratic formulas as required in [Gan et al., IJCAR '16] :

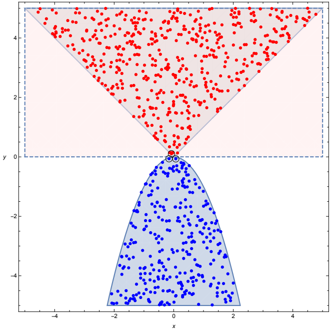




Visualizations in *NIL*

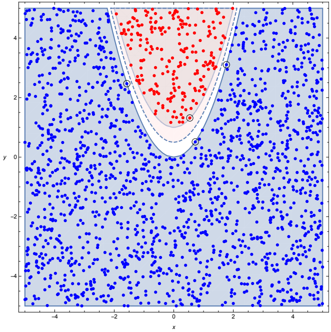
Adjacent and sharper cases as in [Okudono et al., APLAS'17] :

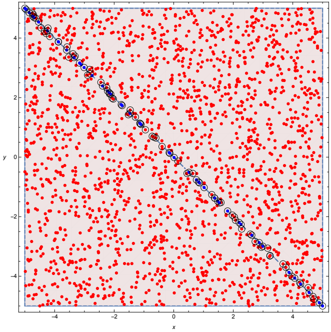




Visualizations in *NIL*

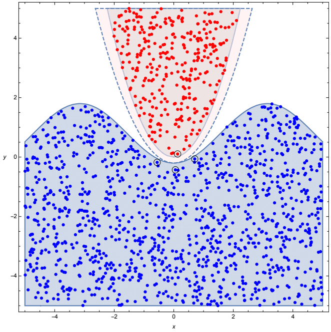
Formulas sharing parallel or coincident boundaries :

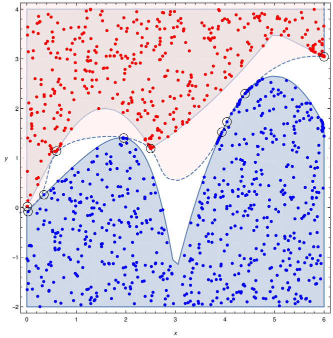




Visualizations in *NIL*

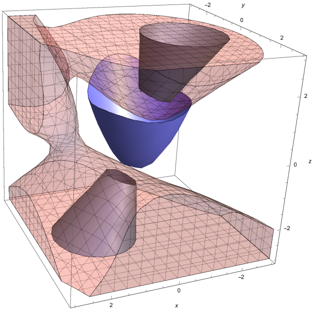
Transcendental cases from [Gao & Zufferey, TACAS '16] and [Kupferschmid & Becker, FORMATS '11], yet with simpler interpolants :





Visualizations in *NIL*

Three-dimensional case from [Dai et al., CAV'13], yet with simpler interpolants :



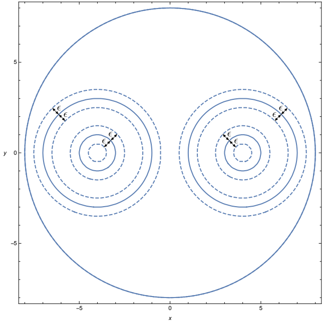
Interpolants of Simpler Forms

Name	Interpolants by *NIL*	Interpolants from the sources
IJCAR16-1	$1 - \frac{3x_1}{4} - \frac{x_2}{2} < 0$	$-3 + 2x_1 + x_1^2 + \frac{1}{2}x_2^2 > 0$
CAV13-1	$-1 + \frac{x^2}{2} - \frac{y}{3} + \frac{xy}{3} - \frac{y^2}{4} < 0$	$436.45(x^2 - 2y^2 - 4) + \frac{1}{2} \leq 0$
CAV13-2	$105x^4 + x^2(140y^2 + 24y(5z+7) + 35z(3z+8)) + 2(70y^3z + 5y^2(12z^2 + 21z + 28) - 14y(6z^3 + 5z^2 + 10) - 35(3z^4 + 8z^2 + 4z - 9)) < 14x(20x^2(z+1) + 10y^2(z+2) - 3y(4z^2 - 5z + 4) - 20z(z^2 + 2))$	$-14629.26 + 2983.44x_3 + 10972.97x_3^2 + 297.62x_2 + 297.64x_2x_3 + 0.02x_2x_3^2 + 9625.61x_2^2 - 1161.80x_2^2x_3 + 0.01x_2^2x_3^2 + 811.93x_2^3 + 2745.14x_2^4 - 10648.11x_1 + 3101.42x_1x_3 + 8646.17x_1x_3^2 + 511.84x_1x_2 - 1034x_1x_2x_3 + 0.02x_1x_2x_3^2 + 9233.66x_1x_2^2 + 1342.55x_1x_2^2x_3 - 138.70x_1x_2^3 + 11476.61x_1^2 - 3737.70x_1^2x_3 + 4071.65x_1^2x_3^2 - 2153.00x1_2x_2 + 373.14x_1^2x_2x_3 + 7616.18x_1^2x_2^2 + 8950.77x_1^3 + 1937.92x_1^3x_3 - 64.07x_1^3x_2 + 4827.25x_1^4 > 0$
CAV13-3	$-1 + \frac{2vc_1}{99} < 0$	$-1.3983vc_1 + 69.358 > 0$
Sharper-1	$2 + y < y^2$	$34y^2 - 68y - 102 \geq 0$
Sharper-2	$y > 0$	$8y + 4x^2 > 0$
IJCAR16-2	$x_1 < x_2$	$-x_1 + x_2 > 0$
CAV13-4	$2xa + 4ya > 5$	$716.77 + 1326.74(ya) + 1.33(ya)^2 + 433.90(ya)^3 + 668.16(xa) - 155.86(xa)(ya) + 317.29(xa)(ya)^2 + 222.00(xa)^2 + 592.39(xa)^2(ya) + 271.11(xa)^3 > 0$
TACAS16	$15x^2 < 4 + 20y$	$y > 1.8 \vee (0.59 \leq y \leq 1.8 \wedge -1.35 \leq x \leq 1.35) \vee (0.09 \leq y < 0.59 \wedge -0.77 \leq x \leq 0.77) \vee (y \geq 0 \wedge -0.3 \leq x \leq 0.3)$

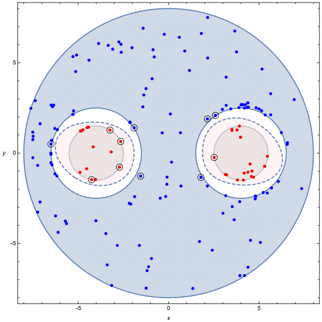
Interpolants of Simpler Forms

Name	Interpolants by *NIL*	Interpolants from the sources
IJCAR16-1	$1 - \frac{3x_1}{4} - \frac{x_2}{2} < 0$	$-3 + 2x_1 + x_1^2 + \frac{1}{2}x_2^2 > 0$
CAV13-1	$-1 + \frac{x^2}{2} - \frac{y}{3} + \frac{xy}{3} - \frac{y^2}{4} < 0$	$436.45(x^2 - 2y^2 - 4) + \frac{1}{2} \leq 0$
CAV13-2	$105x^4 + x^2(140y^2 + 24y(5z+7) + 35z(3z+8)) + 2(70y^3z + 5y^2(12z^2 + 21z + 28) - 14y(6z^3 + 5z^2 + 10) - 35(3z^4 + 8z^2 + 4z - 9)) < 14x(20x^2(z+1) + 10y^2(z+2) - 3y(4z^2 - 5z + 4) - 20z(z^2+2))$	$-14629.26 + 2983.44x_3 + 10972.97x_3^2 + 297.62x_2 + 297.64x_2x_3 + 0.02x_2x_3^2 + 9625.61x_2^2 - 1161.80x_2^2x_3 + 0.01x_2^2x_3^2 + 811.93x_2^3 + 2745.14x_2^4 - 10648.11x_1 + 3101.42x_1x_3 + 8646.17x_1x_3^2 + 511.84x_1x_2 - 1034x_1x_2x_3 + 0.02x_1x_2x_3^2 + 9233.66x_1x_2^2 + 1342.55x_1x_2^2x_3 - 138.70x_1x_2^3 + 11476.61x_1^2 - 3737.70x_1^2x_3 + 4071.65x_1^2x_3^2 - 2153.00x1_2x_2 + 373.14x_1^2x_2x_3 + 7616.18x_1^2x_2^2 + 8950.77x_1^3 + 1937.92x_1^3x_3 - 64.07x_1^3x_2 + 4827.25x_1^4 > 0$
CAV13-3	$-1 + \frac{2vc_1}{99} < 0$	$-1.3983vc_1 + 69.358 > 0$
Sharper-1	$2 + y < y^2$	$34y^2 - 68y - 102 \geq 0$
Sharper-2	$y > 0$	$8y + 4x^2 > 0$
IJCAR16-2	$x_1 < x_2$	$-x_1 + x_2 > 0$
CAV13-4	$2xa + 4ya > 5$	$716.77 + 1326.74(ya) + 1.33(ya)^2 + 433.90(ya)^3 + 668.16(xa) - 155.86(xa)(ya) + 317.29(xa)(ya)^2 + 222.00(xa)^2 + 592.39(xa)^2(ya) + 271.11(xa)^3 > 0$
TACAS16	$15x^2 < 4 + 20y$	$y > 1.8 \vee (0.59 \leq y \leq 1.8 \wedge -1.35 \leq x \leq 1.35) \vee (0.09 \leq y < 0.59 \wedge -0.77 \leq x \leq 0.77) \vee (y \geq 0 \wedge -0.3 \leq x \leq 0.3)$

Perturbation-Resilient Interpolants



(a) ϵ-perturbations in the radii



(b) Interpolant resilient to ϵ-perturbations

Figure – Introducing ϵ-perturbations (say with ϵ up to 0.5) in ϕ and ψ. The synthesized interpolant is hence resilient to any ϵ-perturbation in the radii satisfying $-0.5 \leq \epsilon \leq 0.5$.

Outline

Concluding Remarks

Problem : We face that

- polynomial constraints have been shown useful to express invariant properties for programs and hybrid systems,
- little work on synthesizing nonlinear interpolants, which either restricts the input formulae or yields complex results.

Concluding Remarks

Problem : We face that

- polynomial constraints have been shown useful to express invariant properties for programs and hybrid systems,
- little work on synthesizing nonlinear interpolants, which either restricts the input formulae or yields complex results.

Status : We present

- a unified, counterexample-guided method for generating polynomial interpolants over the general quantifier-free theory of nonlinear arithmetic,
- soundness of NIL, and sufficient conditions for its completeness and convergence,
- Experimental results indicating that our method suffices to address more interpolation tasks, including those with perturbations in parameters, and in many cases synthesizes simpler interpolants.

Concluding Remarks

Problem : We face that

- polynomial constraints have been shown useful to express invariant properties for programs and hybrid systems,
- little work on synthesizing nonlinear interpolants, which either restricts the input formulae or yields complex results.

Status : We present

- a unified, counterexample-guided method for generating polynomial interpolants over the general quantifier-free theory of nonlinear arithmetic,
- soundness of NIL, and sufficient conditions for its completeness and convergence,
- Experimental results indicating that our method suffices to address more interpolation tasks, including those with perturbations in parameters, and in many cases synthesizes simpler interpolants.

Future Work : We plan to

- improve the efficiency of NIL by substituting the general purpose QE procedure with alternative methods,
- combine nonlinear arithmetic with EUFs, by resorting to, e.g., predicate-abstraction techniques,
- investigate the performance of NIL over different classification techniques, e.g., the widespread regression-based methods.

Probabilistic Craig Interpolants ?

Probabilistic Craig Interpolants ?

- Generalized Craig Interpolation for stochastic-SAT : resolution-based BMC of MDPs.
  - ⇒ Teige, T., Fränzle, M. : *Generalized Craig Interpolation for Stochastic Boolean Satisf. Prob.*. TACAS '11.
- Generalized Craig Interpolation for stochastic-SMT : resolution-based UMC of PHA.
  - ⇒ Mahdi, A., Fränzle, M. : *Generalized Craig Interpolation for Stochastic Satisf. Modulo Theory Prob.*. RP '14.