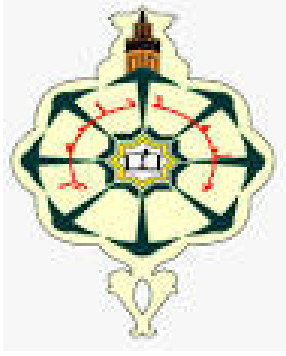
Improving the convergence aspect in the static analysis field.

Yassamine Seladji.

FEANICSES WORKSHOP 2022

December 5, 2022



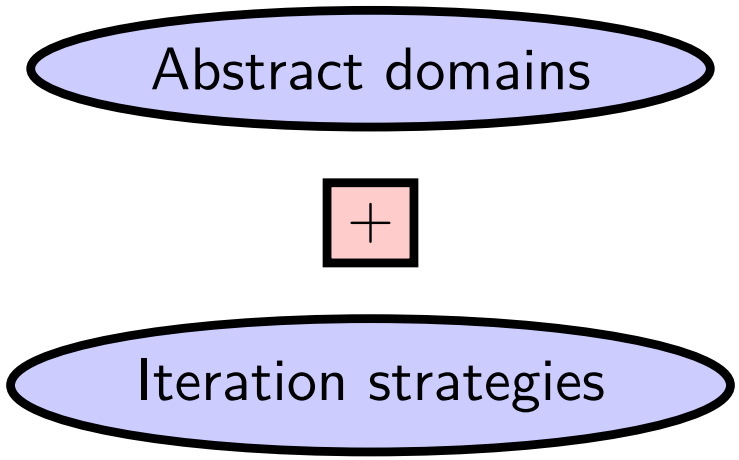
The context

Program verification

- The formal verification proves that the program semantic verifies a given specification.
- The abstract interpretation does an automatically verification.

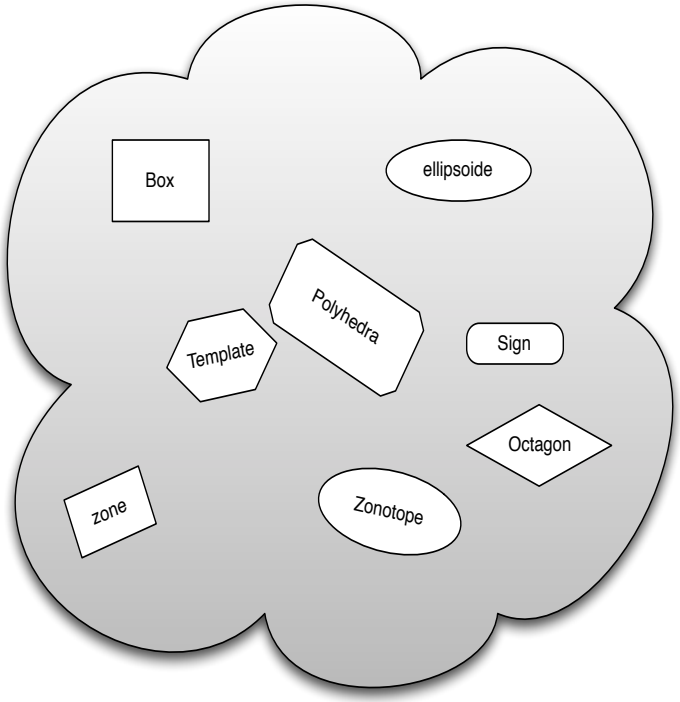
The context

Abstract Interpretation



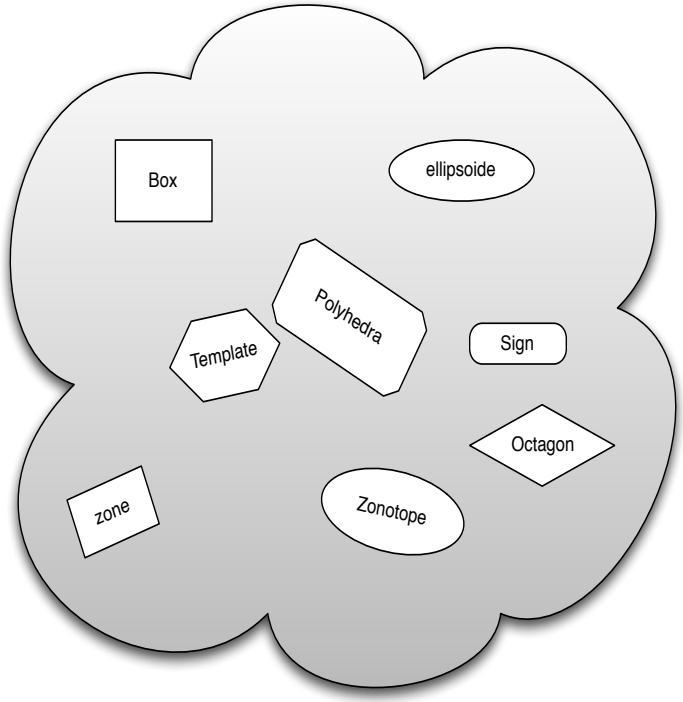
The context

Abstract Interpretation



The context

Abstract Interpretation

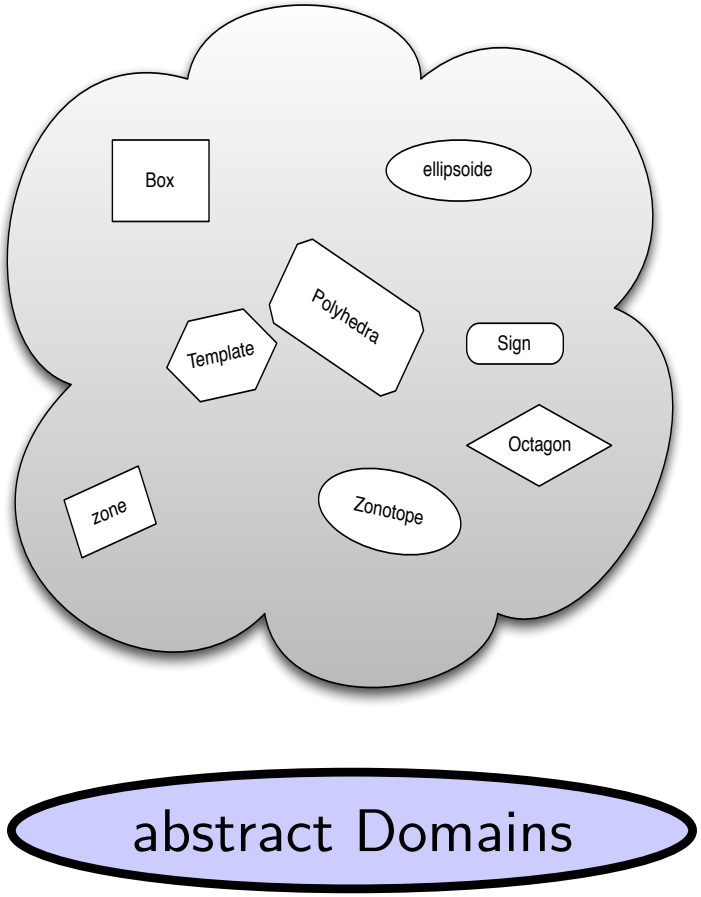


Algorithm 2 The Kleene Algorithm

1: $\mathbb{D}_i := \bot$
2: **repeat**
3: $\quad \mathbb{D}_i := \mathbb{D}_{i-1} \sqcup F(\mathbb{D}_{i-1})$
4: **until** $\mathbb{D}_i \sqsubseteq \mathbb{D}_{i-1}$

The context

Abstract Interpretation

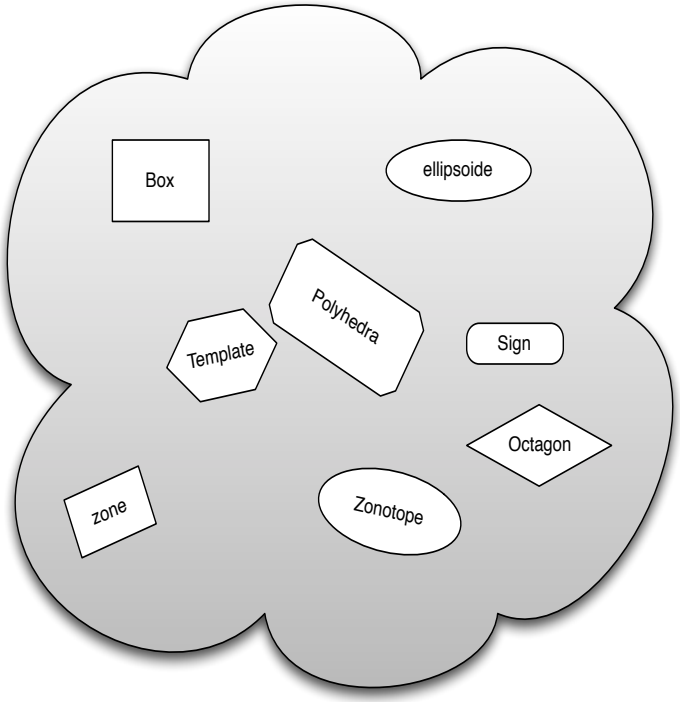


Algorithm 3 The Kleene Algorithm

1: $\mathbb{D}_i := \bot$
2: **repeat**
3: $\quad \mathbb{D}_i := \mathbb{D}_{i-1} \sqcup F(\mathbb{D}_{i-1})$
4: **until** $\mathbb{D}_i \sqsubseteq \mathbb{D}_{i-1}$

The context

Abstract Interpretation



abstract Domains

Algorithm 4 The Kleene Algorithm

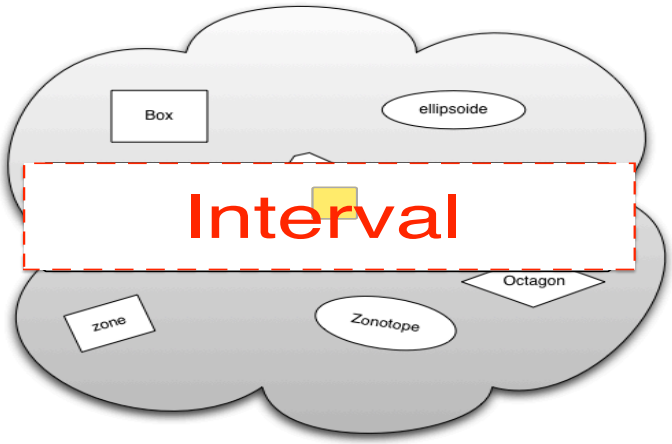
1: $\mathbb{D}_i := \bot$
2: **repeat**
3: $\quad \mathbb{D}_i := \mathbb{D}_{i-1} \sqcup F(\mathbb{D}_{i-1})$
4: **until** $\mathbb{D}_i \sqsubseteq \mathbb{D}_{i-1}$

Iteration strategies

Outline

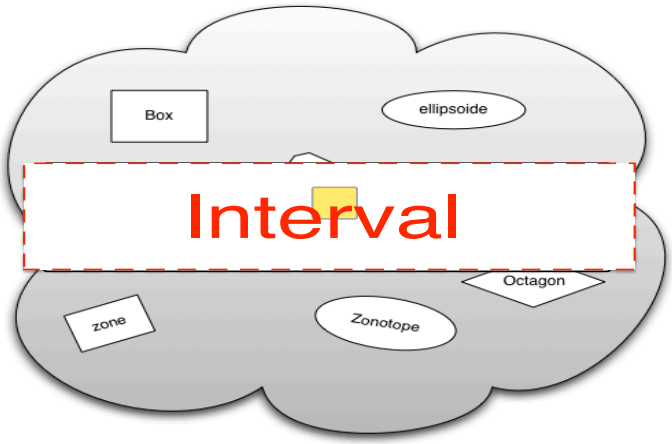
1. The Interval analysis
   - Numerical Analysis
   - Example
   - Acceleration Process
   - Experimentation
2. The Polyhedral Analysis
   - Template Analysis Based support function
   - Define relevant templates
   - Eliminate constraints redundancy
   - The acceleration process
   - Experimentations

The Interval analysis



abstract Domains

The Interval analysis



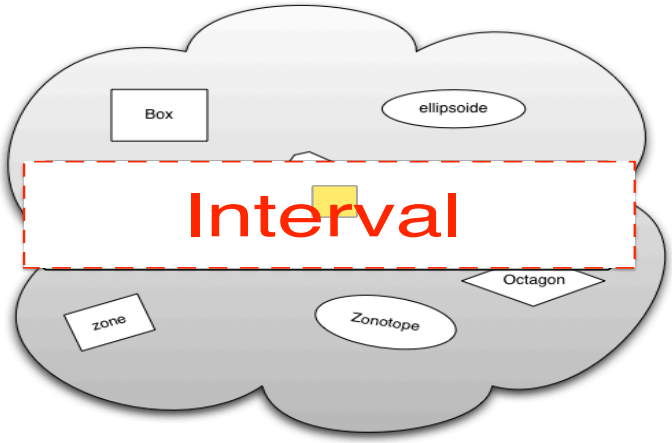
Algorithm 6 The Kleene Algorithm

1: $\mathbb{I}_i := \bot$
2: **repeat**
3: $\quad \mathbb{I}_i := \mathbb{I}_{i-1} \sqcup F(\mathbb{I}_{i-1})$
4: **until** $\mathbb{I}_i \sqsubseteq \mathbb{I}_{i-1}$

abstract Domains

Iteration strategies

The Interval analysis



abstract Domains

Algorithm 7 The Kleene Algorithm

1: $\mathbb{I}_i := \bot$
2: **repeat**
3: $\quad \mathbb{I}_i := \mathbb{I}_{i-1} \sqcup F(\mathbb{I}_{i-1})$
4: **until** $\mathbb{I}_i \sqsubseteq \mathbb{I}_{i-1}$

Iteration strategies

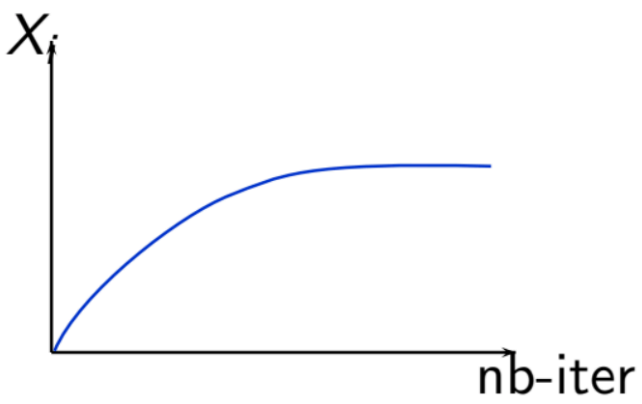
Acceleration

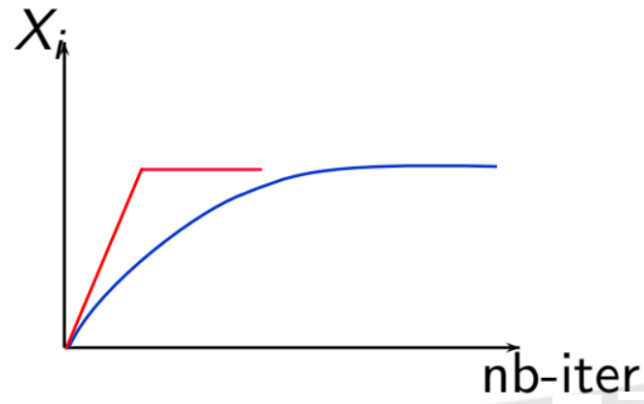
Numerical Analysis

Sequence transformation

The Transformation Methods

A sequence transformation is a function $T : \mathbb{R}^{\mathbb{N}} \to \mathbb{R}^{\mathbb{N}}$ such that, for all converging sequences $(X_i)_{i\in\mathbb{N}} \in \mathbb{R}^{\mathbb{N}}$, the sequence $(X'_i)_{i\in\mathbb{N}}$ defined by $x'_i = T(x_i)$ is convergent with $\lim_{i\to\infty} X'_i = \lim_{i\to\infty} X_i$. The sequence $(X'_i)_{i\in\mathbb{N}}$ is said to be accelerated if $\lim_{i\to\infty} \frac{X'_i - X_\infty}{X_i - X_\infty} = 0$.





Numerical Analysis

Sequence transformation

The Aitken Δ^2-Method

Let $(X_i)_{i\in\mathbb{N}}$ be a numerical sequence and T the Aitken Δ^2-Method, such that: $\forall n \in \mathbb{N}, T(X_{n+1}) = \dfrac{X_{n+1} - X_n}{X_{n+2} - 2X_{n+1} + X_n}$.

Exemple: $X_n = 1 + \dfrac{1}{n+1}, \forall n \geq \mathbb{N}$ avec $\lim_{n\to+\infty} X_n = 1$

ε_n^0	ε_n^2	ε_n^4	ε_n^6	ε_n^8
2.0000000				
1.5000000	1.2500000			
1.3333333	1.1666667	1.1111109		
1.2500000	1.1249999	1.0833337	1.0624931	
1.2000000	1.1000001	1.0666663	1.0500028	1.0399799
1.1666667	1.0833333	1.0555556	1.0416545	1.0334257
1.1428571	1.0714287	1.0476161	1.0357504	
1.1250000	1.0624998	1.0416761		
1.1111111	1.0555557			
1.1000000				

Interval Analysis

Example

The program

```
while (1) {*

xn1 = -0.4375*x1+0.0625*x2+0.2652*x3+0.1*u1;

xn2 = 0.0625*x1+0.4375*x2+0.2652*x3+0.1*u2;

xn3 = -0.2652*x1+0.2652*x2+0.375*x3+0.1*u3;

x1 = xn1;

x2 = xn2;

x3 = xn3;

}
```

$\Longrightarrow$

Kleene iterations

x_1

$[1.000000, 2.000000]$
$[-0.447300, 5.716500]$
$[-2.291255, 6.573381]$

$\Downarrow\Downarrow$ 127 Kleene iterations

$[-5.197505, 8.873306]$

Interval Analysis

Example

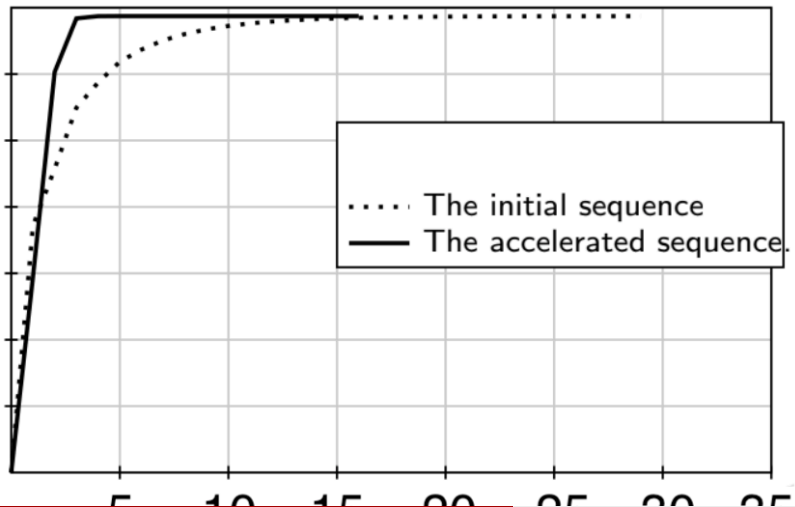
Kleene iterations

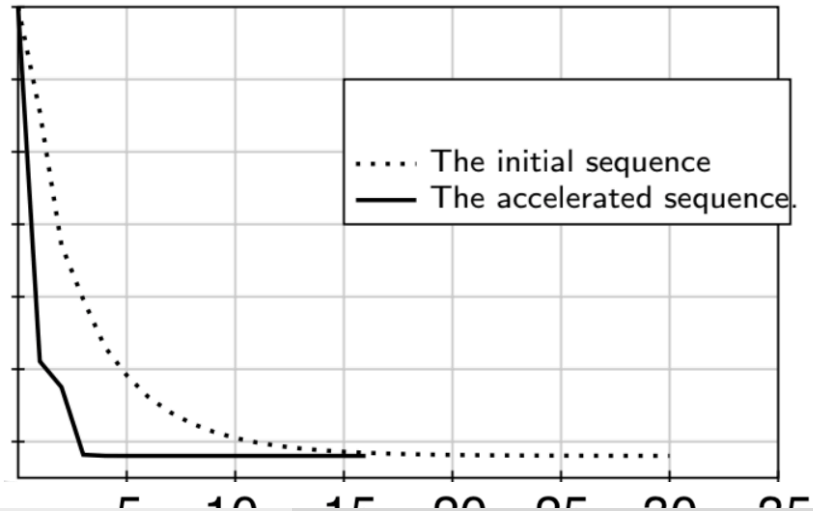
x_1

$[1.000000, 2.000000]$
$[-0.447300, 5.716500]$
$[-2.291255, 6.573381]$

⇊ 127 itérations

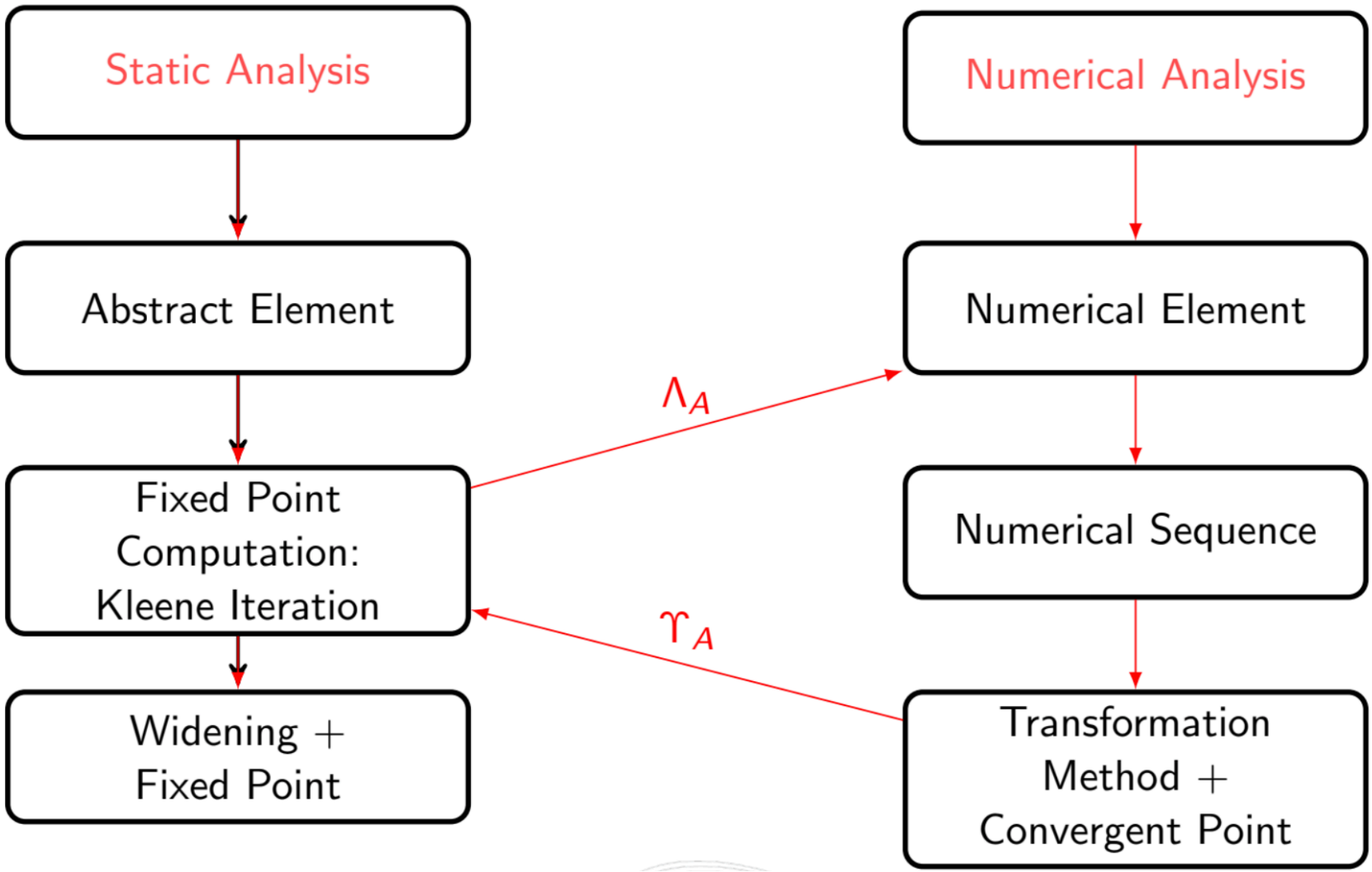
$[-5.197505, 8.873306]$





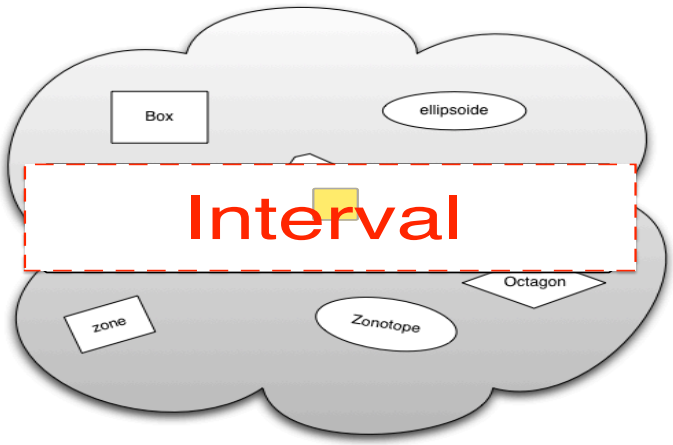
Interval Analysis

Acceleration Process



The Interval analysis

Acceleration Process



abstract Domains

Algorithm 8 The accelerated Kleene Algorithm

1: $\mathbb{I}_i := \bot$
2: **repeat**
3: $\quad \mathbb{I}_i := \mathbb{I}_{i-1} \sqcup F(\mathbb{I}_{i-1})$
4: $\quad Y_i := Accelerate(\Lambda(\mathbb{I}_0), ..., \Lambda(\mathbb{I}_I))$
5: $\quad$ **if** $\| Y_i - Y_{i-1} \| \leq \epsilon$ **then**
6: $\quad\quad \mathbb{I}_i := \mathbb{I}_{i-1} \sqcup \Upsilon(Y_i)$
7: $\quad$ **end if**
8: **until** $\mathbb{I}_i \sqsubseteq \mathbb{I}_{i-1}$

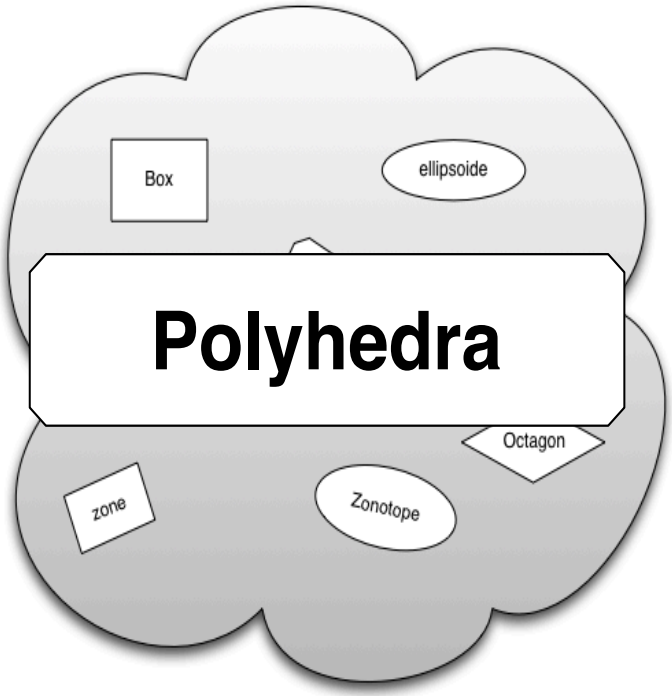
Iteration strategies

Interval Analysis

Experimentation

	Efficiency (Iterations - Computation time)	
Program name	Kleene	Accel.
`contraction.c`	127 - 0.014s	17 - 0.009s
`butter1.c`	346 - 0.018s	8 - 0.003s
`butter2.c`	180 - 0.012s	16 - 0.006s
`gauss-seidl.c`	24 - 0.004s	12 - 0.004s

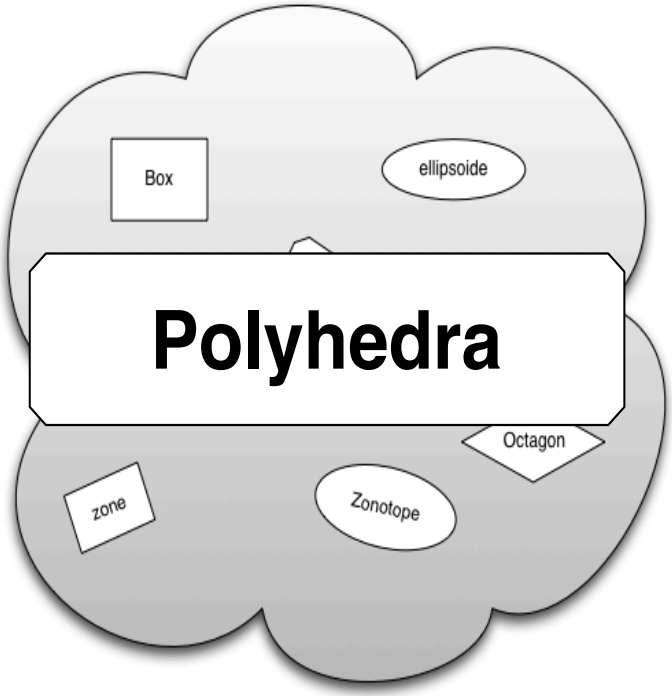
The Polyhedral Analysis



Algorithm 9 The Kleene Algorithm

1: $\mathbb{P}_i := \bot$
2: **repeat**
3: $\quad \mathbb{P}_i := \mathbb{P}_{i-1} \sqcup F(\mathbb{P}_{i-1})$
4: **until** $\mathbb{P}_i \sqsubseteq \mathbb{P}_{i-1}$

The Polyhedral Analysis



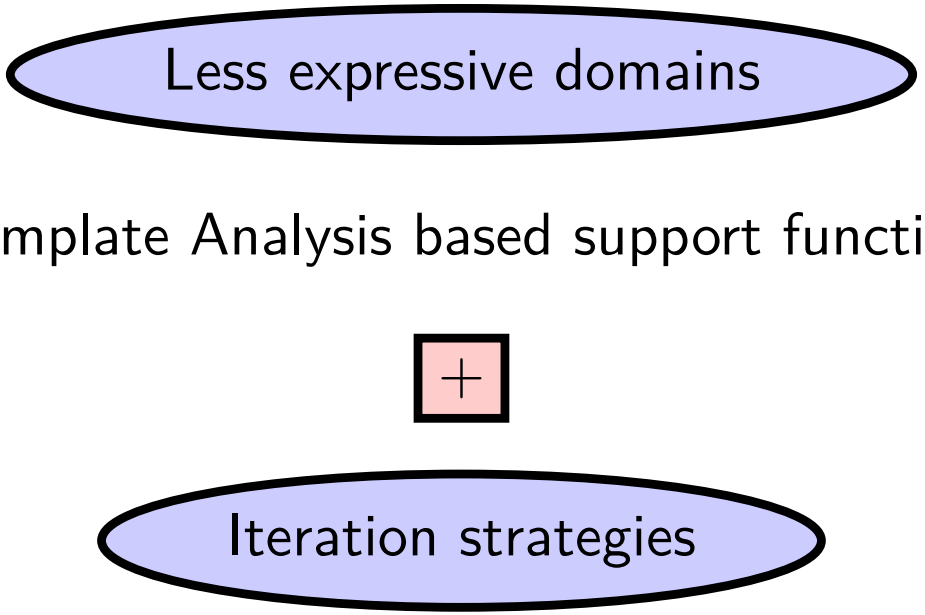
Algorithm 10 The Kleene Algorithm

1: $\mathbb{P}_i := \bot$
2: **repeat**
3: $\quad \mathbb{P}_i := \mathbb{P}_{i-1} \sqcup F(\mathbb{P}_{i-1})$
4: **until** $\mathbb{P}_i \sqsubseteq \mathbb{P}_{i-1}$

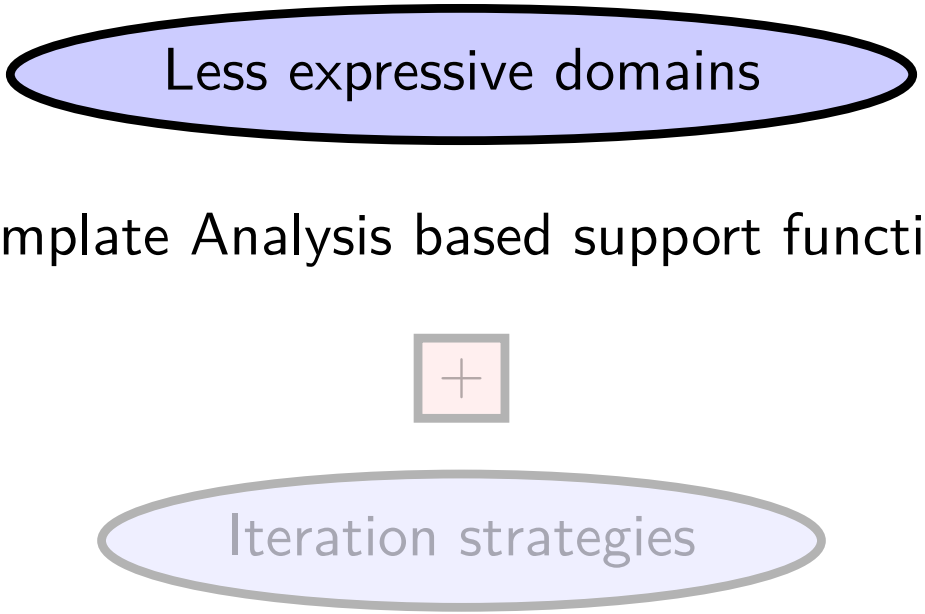
Iteration strategies

Less Expressive Domains

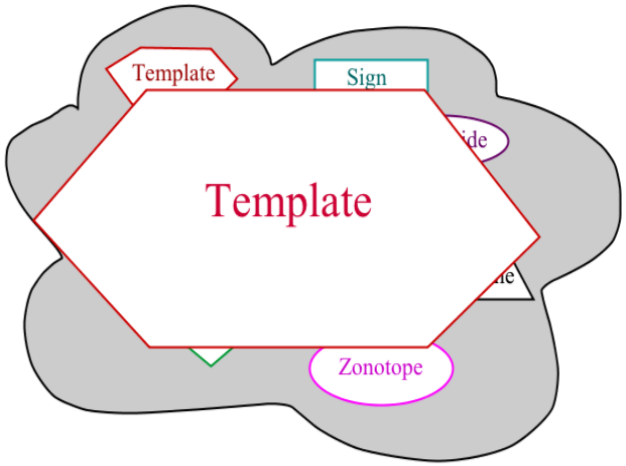
The Polyhedral Analysis



The Polyhedral Analysis



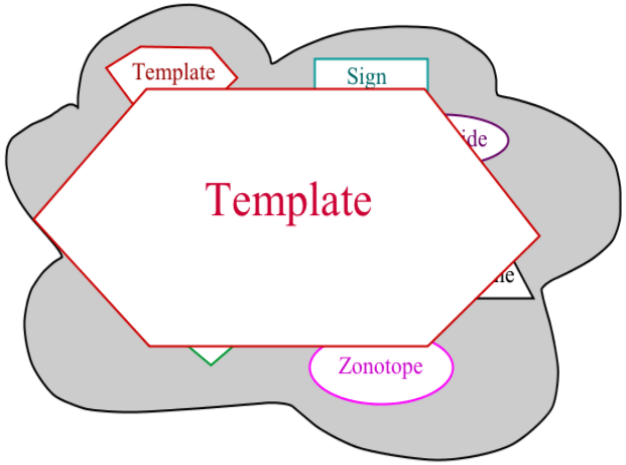
Template Analysis Based support function



$$P = \bigcap_{i \in [1,m]} \{ x \in R^n : \langle x, a_i \rangle \leq c_i \}$$

$$X \in R^n, \begin{pmatrix} a_{11} & \dots & a_{n1} \\ \vdots & \ddots & \vdots \\ a_{1m} & \dots & a_{nm} \end{pmatrix} X \leq \begin{pmatrix} c_1 \\ \vdots \\ c_m \end{pmatrix}$$

Template Analysis Based support function



$$P = \bigcap_{i \in [1,m]} \{ x \in R^n : \langle x, a_i \rangle \leq c_i \}$$

$$X \in R^n, \underbrace{\begin{pmatrix} a_{11} & \dots & a_{n1} \\ \vdots & \ddots & \vdots \\ a_{1m} & \dots & a_{nm} \end{pmatrix}}_{Known} X \leq \underbrace{\begin{pmatrix} c_1 \\ \vdots \\ c_m \end{pmatrix}}_{unKnown}$$

Template Analysis based support function

$$X \in R^n, \underbrace{\begin{pmatrix} a_{11} & \dots & a_{n1} \\ \vdots & \ddots & \vdots \\ a_{1m} & \dots & a_{nm} \end{pmatrix}}_{Known} X \leq \underbrace{\begin{pmatrix} c_1 \\ \vdots \\ c_m \end{pmatrix}}_{unKnown}$$

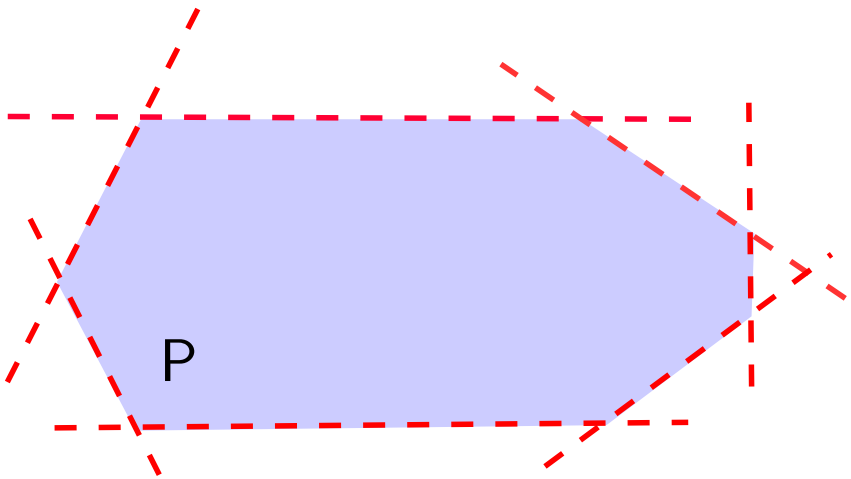
Template Analysis based support function

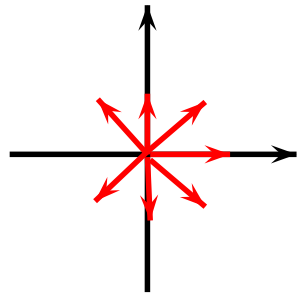
$$X \in R^n, \underbrace{\begin{pmatrix} a_{11} & \dots & a_{n1} \\ \vdots & \ddots & \vdots \\ a_{1m} & \dots & a_{nm} \end{pmatrix}}_{Known} X \leq \underbrace{\begin{pmatrix} c_1 \\ \vdots \\ c_m \end{pmatrix}}_{unKnown}$$

$$\underbrace{\begin{pmatrix} a_{11} & \dots & a_{n1} \\ \vdots & \ddots & \vdots \\ a_{1m} & \dots & a_{nm} \end{pmatrix}}_{\text{Set of Directions}} \qquad \underbrace{\begin{pmatrix} c_1 \\ \vdots \\ c_m \end{pmatrix}}_{\text{Support functions}}$$

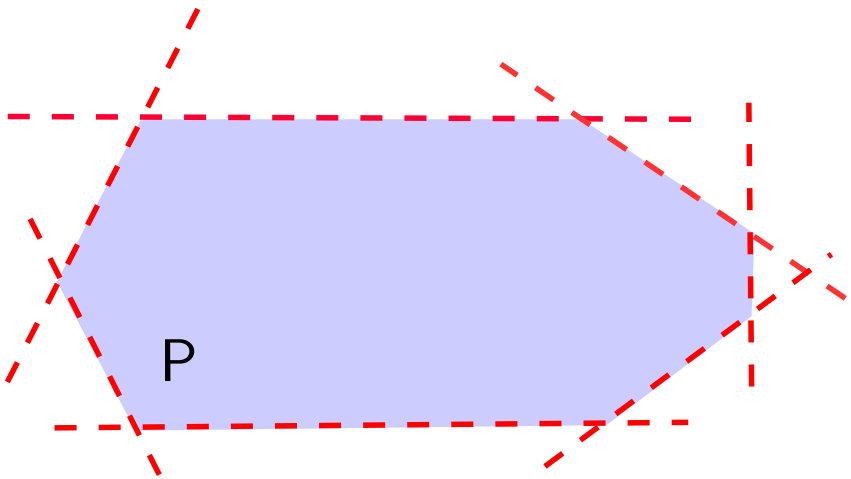
$$\forall i \in [1, m], d_i = (a_{1i}, \dots, a_{ni}) \qquad \forall i \in [1, m], C_i = \delta_P(d_i)$$

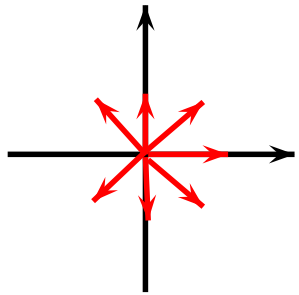
Template Analysis based support function





Template Analysis based support function





$$P = \bigcap_{i \in [1,m]} \left\{ x \in R^n : \langle x, d_i \rangle \leq \delta_P(d_i) \right\}$$

Template Analysis based support function

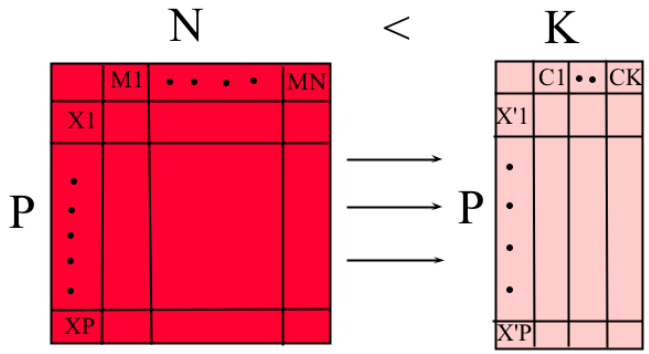
- Advantages :
  - The analysis are done in linear time.
  - The result is close to the polyhedral analysis accuracy
- Limits :
  - Define relevant templates.
  - Eliminate constraints redundancy.

Template Analysis based support function

- Limits :
  - Define relevant templates $\Rightarrow$ Use the principal components analysis (VMCAI'17).
  - Eliminate constraints redundancy $\Rightarrow$ Use the K-median Algorithm (IJCM'18).

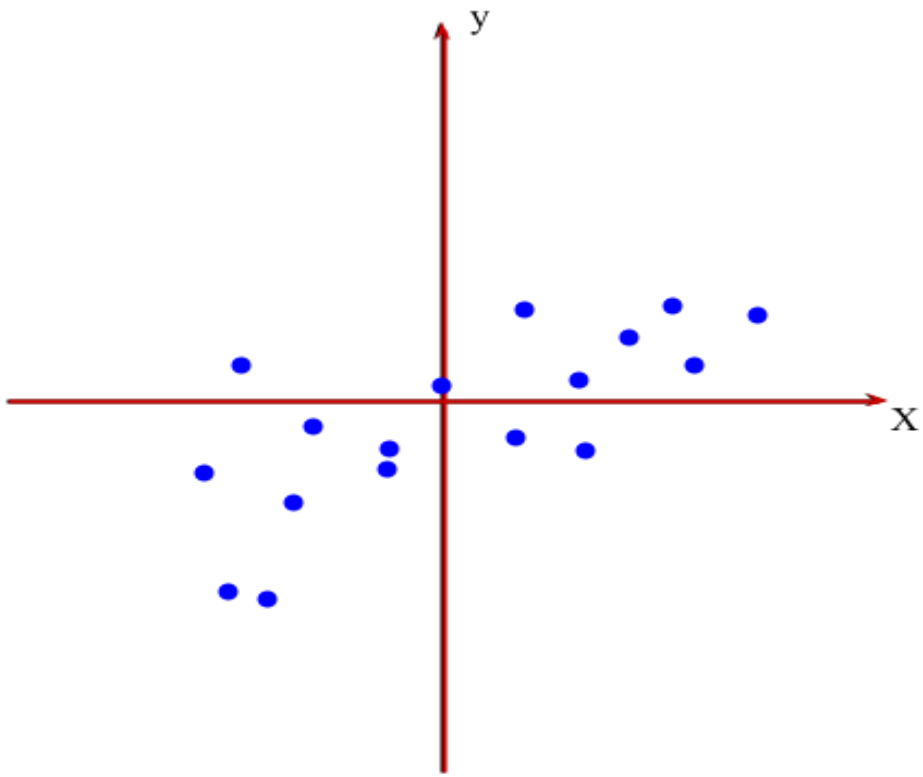
The Principal Component Analysis

- Data dimensionality reduction.



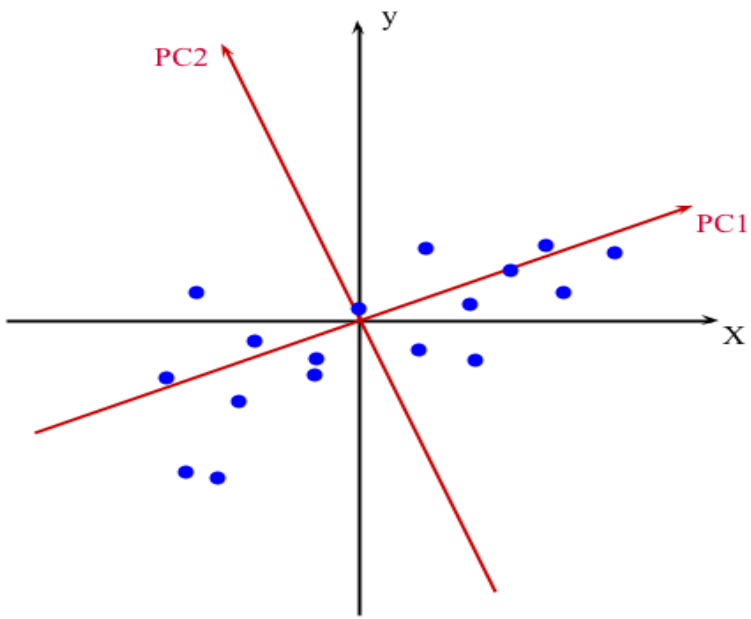
- The K components display as much as possible of the variation among data.
- The PCA is concerned with identifying correlation in data.

The Principal Component Analysis



- Cloud of data : points x_i with $i \in [1, P]$.

The Principal Component Analysis



- Cloud of data : points x_i with $i \in [1, P]$.
- PC1 and PC2 maximize the variance of X_i.

The Principal Component Analysis

1. Generate data (points).
2. Calculate the covariance matrix.
3. Find the eigenvalues and eigenvectors of the covariance matrix.
4. The eigenvectors with the highest eigenvalues are chosen as principal components.

The PCA Point Collection

- The PCA point collection.
  - The center projection method.
    - Perform the template analysis based support function, using a set Δ of directions uniformly distributed on the unit sphere.
    - Compute the Chebyshev center (x_c, r) of P.

$$\max\ r$$

$$st : \forall d_i \in \Delta, \langle x_c, d_i \rangle + r\|d_i\|_2 \leq \delta_P(d_i)$$

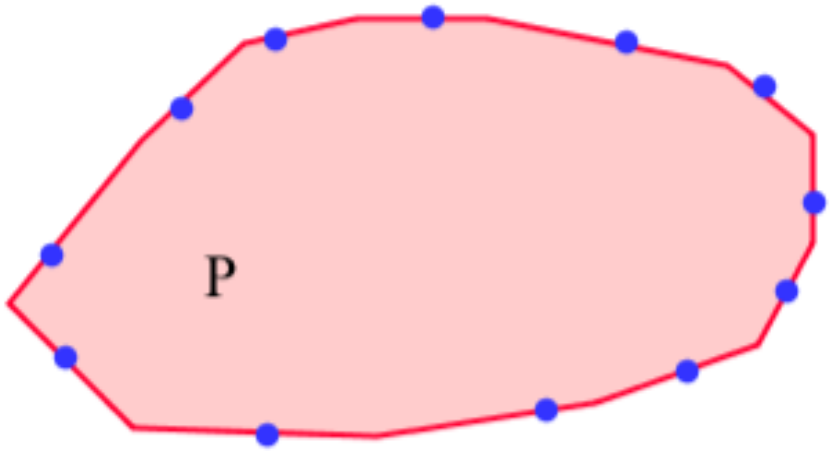
    - $\forall d_i \in \Delta$, compute the orthogonal projection of x_c on H_{d_i}:$\langle x_c, d_i \rangle = \delta_P(d_i)$, such that:

$$proj(x_c, H_{d_i}) = x_c + \left(\frac{\delta_P(d_i) - \langle x_c, d_i \rangle}{\|d_i\|^2} \right) d_i$$

    - $proj(x_c, H_{d_i}) \in H_{d_i}$ and may belong to P.

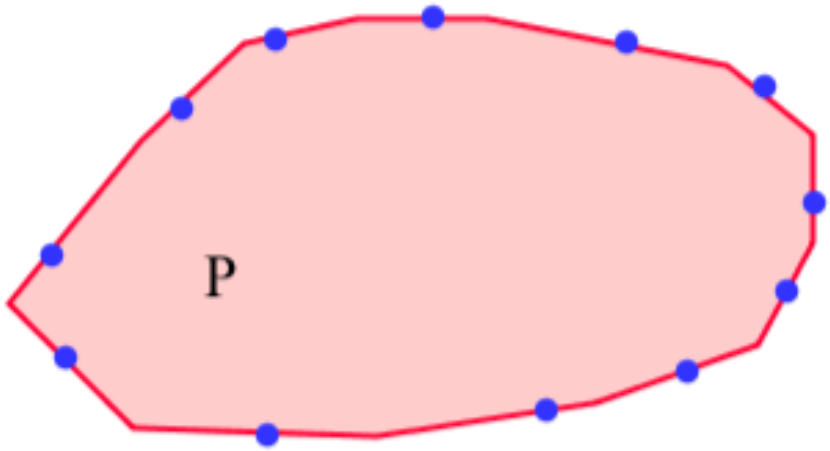
Define relevant templates

- The PCA Point Collection : D_{PCA}.



Define relevant templates

- The PCA Point Collection : D_{PCA}.
- The PCA Computation

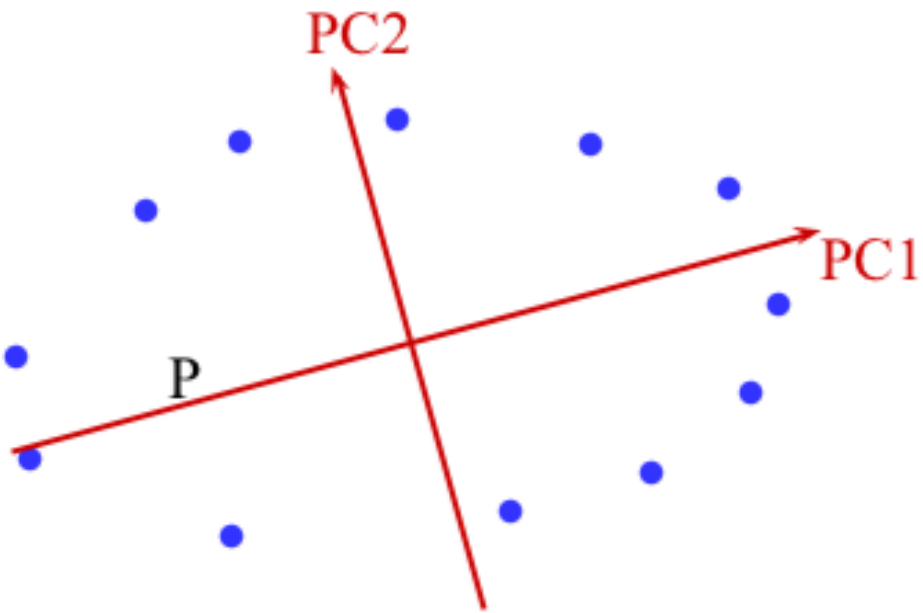


Define relevant templates

- The PCA Point Collection : D_{PCA}.
- The PCA Computation
  - Compute the PCA of D_{PCA}.

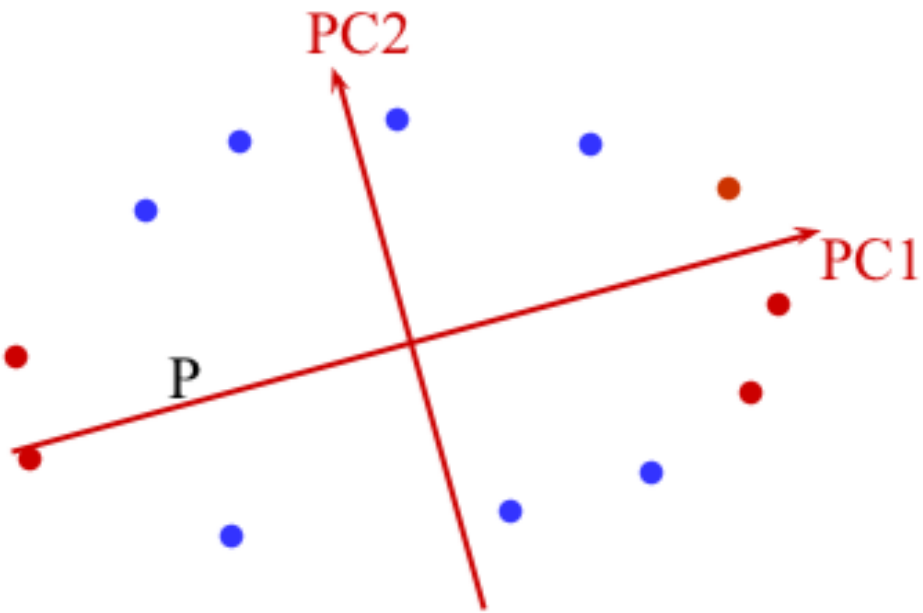
Define relevant templates

- The PCA Point Collection : D_{PCA}.
- The PCA Computation
  - Compute the PCA of D_{PCA}.



Define relevant templates

- The PCA Point Collection : D_{PCA}.
- The PCA Computation
  - Compute the PCA of D_{PCA}.
  - Delete K points that mostly contribute in the PCA computation.

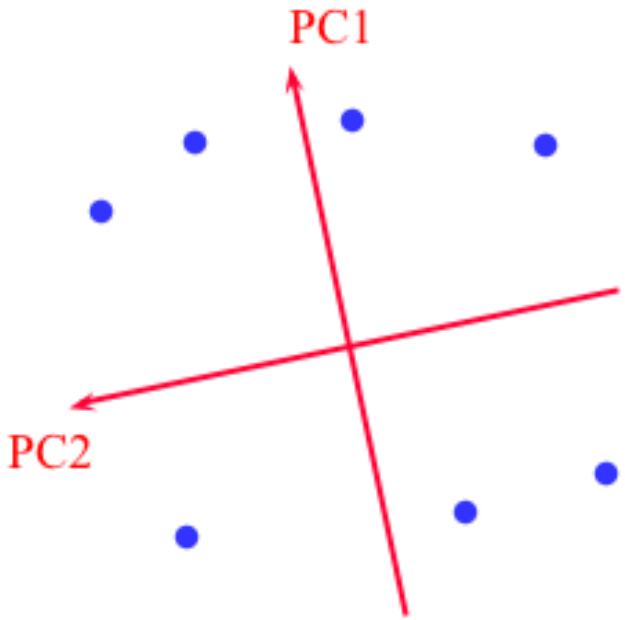


Define relevant templates

- The PCA Point Collection : D_{PCA}.
- The PCA Computation
  - Compute the PCA of D_{PCA}.
  - Delete K points that mostly contribute in the PCA computation.

Define relevant templates

- The PCA Point Collection : D_{PCA}.
- The PCA Computation
  - Compute the PCA of D_{PCA}.
  - Delete K points that mostly contribute in the PCA computation.

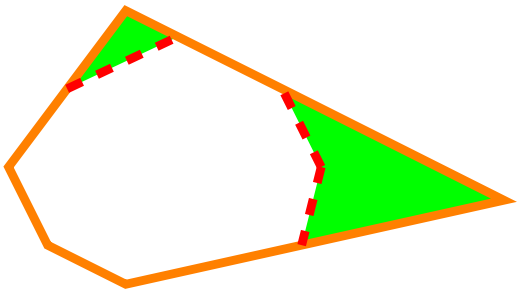


Template Analysis based support function

- Limits :
  - Define relevant templates $\Rightarrow$ Use the principal components analysis (VMCAI'17).
  - Eliminate constraints redundancy $\Rightarrow$ Use the K-median Algorithm (IJCM'18).

Eliminate constraints redundancy

We want to approximate $\mathbb{P}^n$ by keeping only $k \leq n$ constraints, such that: $\mathbb{P}^n \subseteq \mathbb{P}^k$, with $\mathbb{P}^k$ is the best approximation of $\mathbb{P}^n$.



$$vol(P^k) - vol(P^n),$$

Eliminate constraints redundancy

The following combinatorial optimization problem is known to be a hard problem:

$$\min_{\substack{S \subset [n] \\ |S|=k}} vol(P^K) - vol(P^n)$$

Eliminate constraints redundancy

The following combinatorial optimization problem is known to be a hard problem:

$$\min_{\substack{S \subset [n] \\ |S|=k}} vol(P^K) - vol(P^n)$$

The discrete approximation :

- Define the volume difference approximation.

Eliminate constraints redundancy

The following combinatorial optimization problem is known to be a hard problem:

$$\min_{\substack{S \subset [n] \\ |S|=k}} vol(P^K) - vol(P^n)$$

The discrete approximation :

- Define the volume difference approximation.

$$\min_{\substack{S \subset [n] \\ |S|=k}} \sum_{x \in X} \min_{j \in [n]} d_j(x)$$

Eliminate constraints redundancy

The following combinatorial optimization problem is known to be a hard problem:

$$\min_{\substack{S \subset [n] \\ |S|=k}} vol(P^K) - vol(P^n)$$

The discrete approximation :

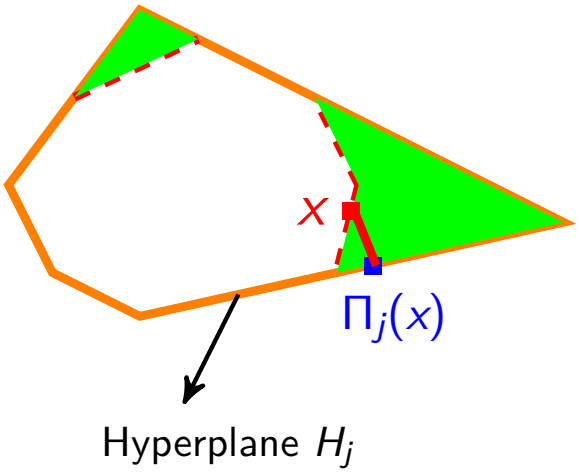
- Define the volume difference approximation.

$$\min_{\substack{S \subset [n] \\ |S|=k}} \sum_{x \in X} \min_{j \in [n]} d_j(x)$$

- Define distance functions.

The distance functions

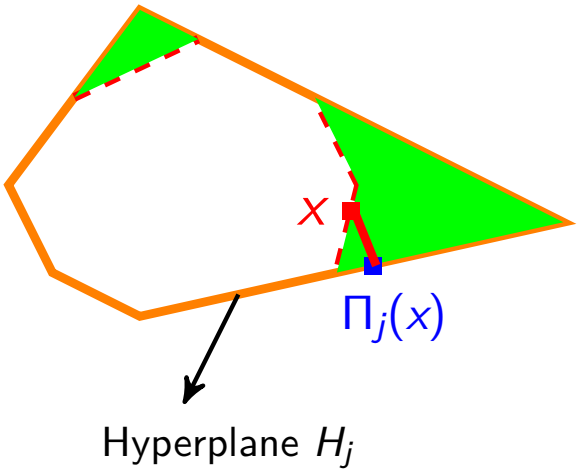
- The projective distance :



$$\Pi_j(x) := \arg\min\{\|x - y\| : y \in H_j\}.$$

The distance functions

- The projective distance :

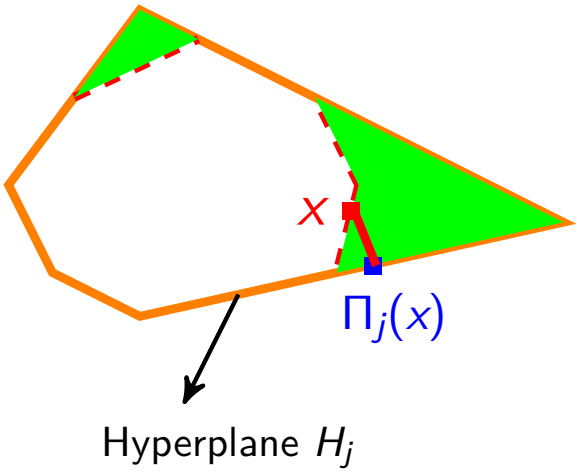


$$\Pi_j(x) := \arg\min\{\|x - y\| : y \in H_j\}.$$

$$p_j(x) = b_j - \langle a_j, x\rangle, \quad j \in [n], x \in bd(P^n).$$

The distance functions

- The projective distance :



$$\Pi_j(x) := \arg\min\{\|x - y\| : y \in H_j\}.$$

$$p_j(x) = b_j - \langle a_j, x\rangle, \quad j \in [n], x \in bd(P^n).$$

Where : $bd(\mathbb{P}^n) = \bigcup_{i=1}^{n} (\mathbb{P}^n \bigcap H_i()$ with $H_i := \{x \in \mathbb{R}^n : \langle a_i, x\rangle = b_i\}$

Eliminate constraints redundancy

The following combinatorial optimization problem is known to be a hard problem:

$$\min_{\substack{S\subset[n]\\|S|=k}} vol(P^K) - vol(P^n)$$

The discrete approximation :

- Define the volume difference approximation.

$$\min_{\substack{S\subset[n]\\|S|=k}} \sum_{x\in X} \min_{j\in[n]} d_j(x)$$

- Define distance functions.

$$p_j(x) = b_j - \langle a_j, x\rangle, \quad j \in [n], x \in bd(\mathbb{P}^n)$$

Eliminate constraints redundancy

The following combinatorial optimization problem is known to be a hard problem:

$$\min_{\substack{S\subset[n]\\|S|=k}} vol(P^K) - vol(P^n)$$

The discrete approximation :

- Define the volume difference approximation.

$$\min_{\substack{S\subset[n]\\|S|=k}} \sum_{x\in X} \min_{j\in[n]} d_j(x)$$

- Define distance functions.

$$p_j(x) = b_j - \langle a_j, x\rangle, \quad j \in [n], x \in bd(\mathbb{P}^n)$$

- Solve the K-median problem.

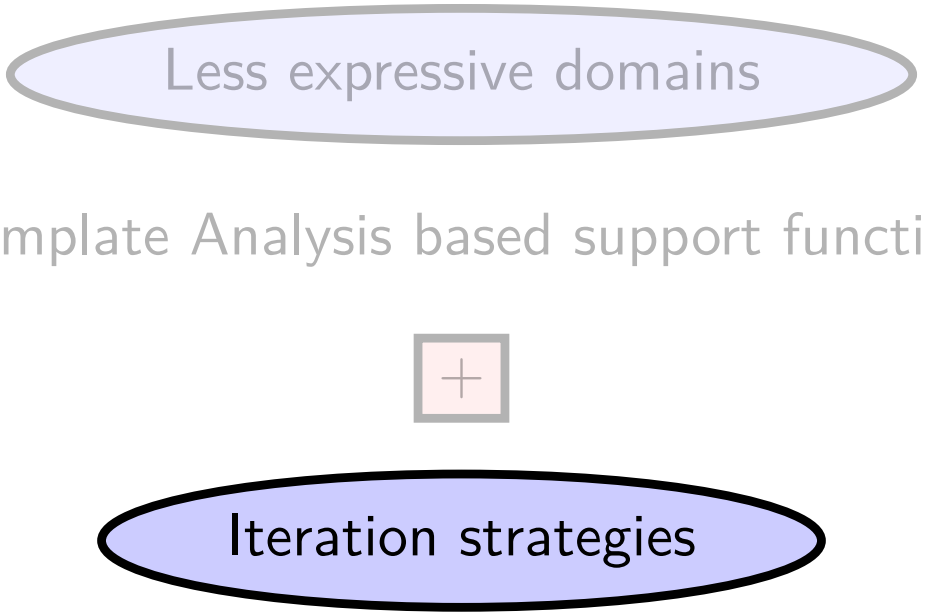
The discrete approximation

$$\min_{\substack{S\subset[n]\\|S|=k}} \sum_{x\in X} \min_{j\in[n]} d_j(x)$$

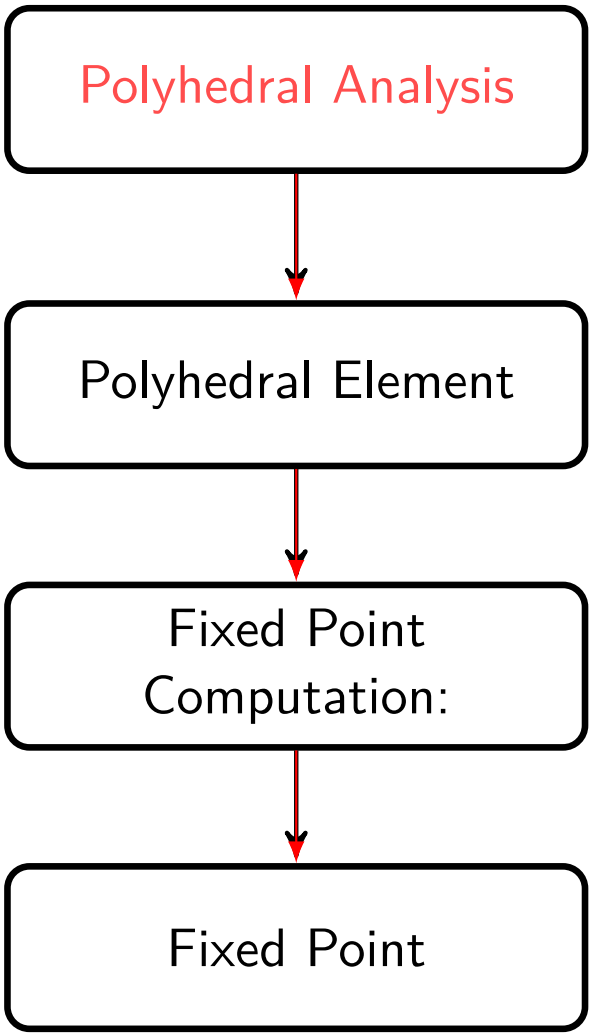
- It's a k-median problem.
- X is the set of cities.
- The hyperplanes $H_1, \cdots, H_m$ is the set facilities .
- $d(j, x)$ is the distance between a city $x \in X$ and a facility H_j.
- We apply the algorithm of Jain and Vazirani [1], for approximately solving k-median problems in polynomial time.

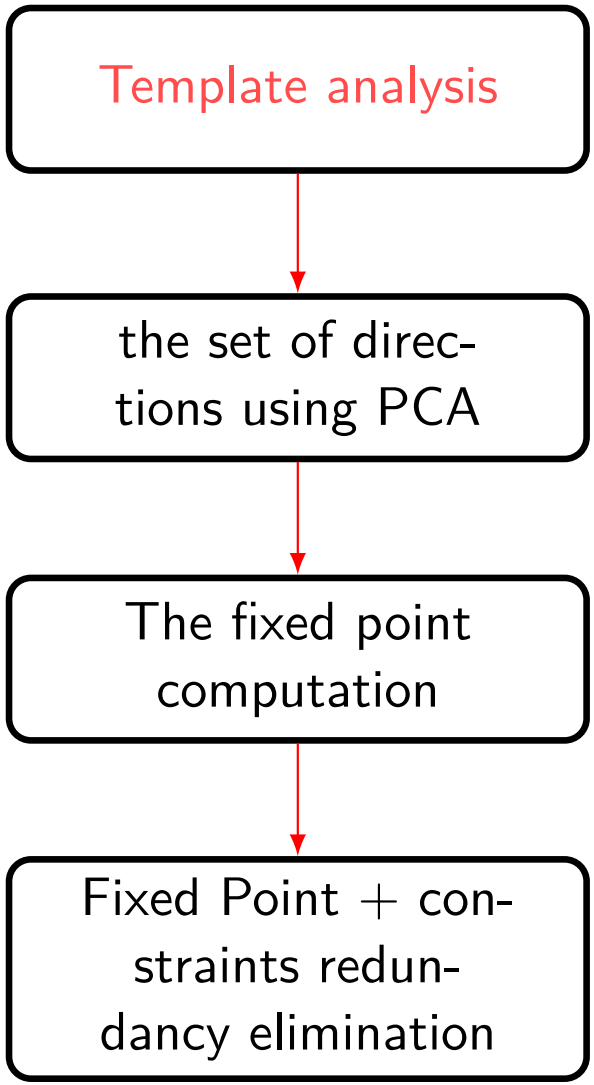
[1] V. V. Vazirani, Approximation algorithms, Springer-Verlag, 2001

Polyhedral Analysis

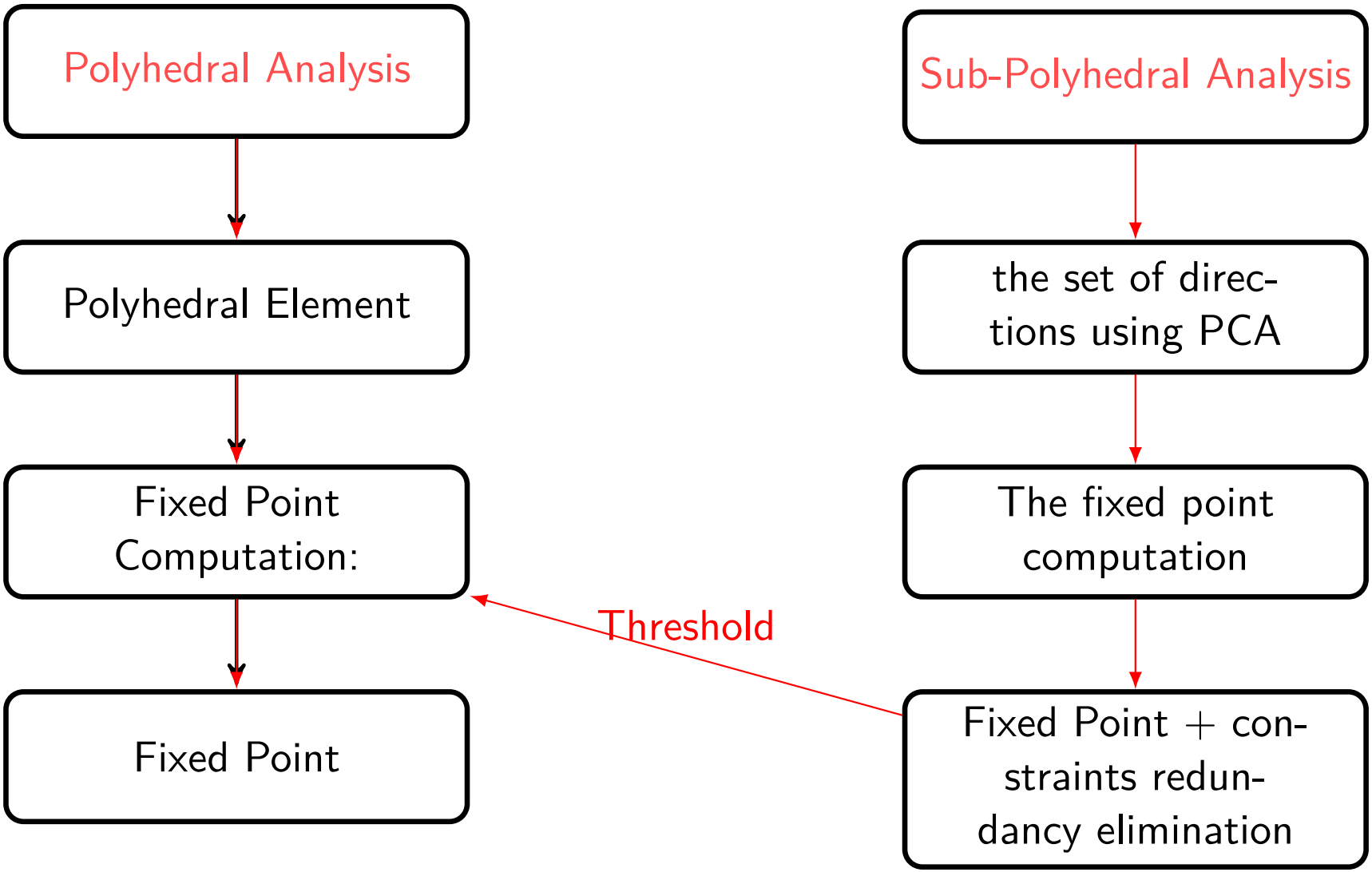


The acceleration process

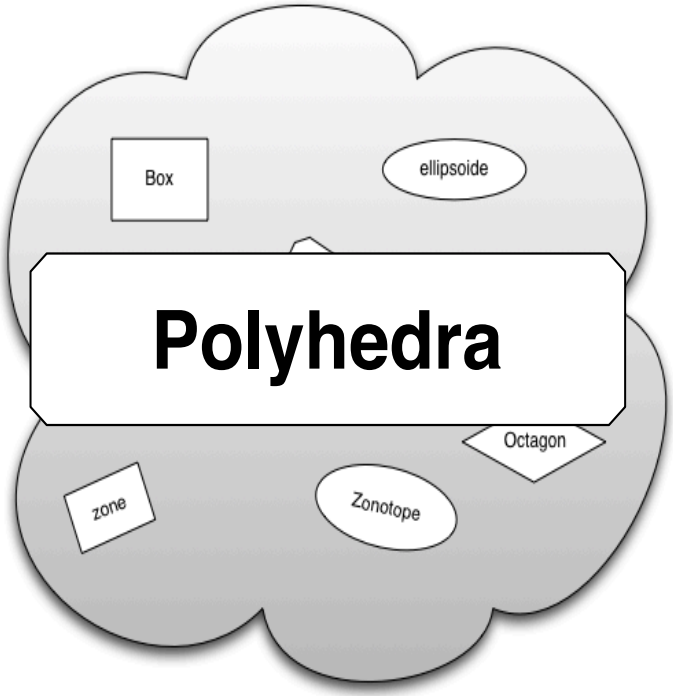




The acceleration process



The acceleration process



Algorithm 11 The Kleene Algorithm

1: $\mathbb{P}_i := \bot$
2: **repeat**
3: $\quad \Delta = \mathit{PCAdirection}(\mathbb{P}_{i-1})$
4: $\quad \mathbb{P}_\Delta = \mathit{acceleration}(\mathbb{P}_{i-1}, \Delta)$
5: $\quad \mathbb{P}_i := \mathbb{P}_{i-1} \sqcup F(\mathbb{P}_\Delta)$
6: **until** $\mathbb{P}_i \sqsubseteq \mathbb{P}_{i-1}$

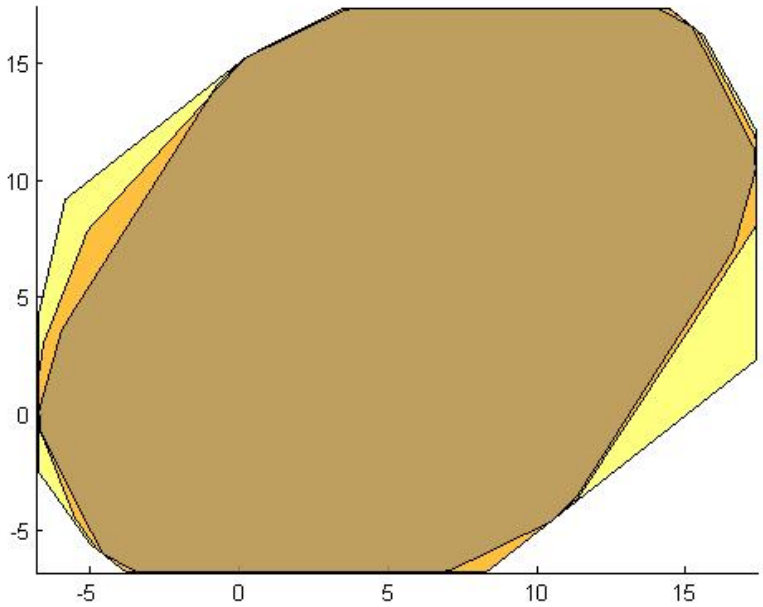
Less Expressive Domains

Iteration strategies

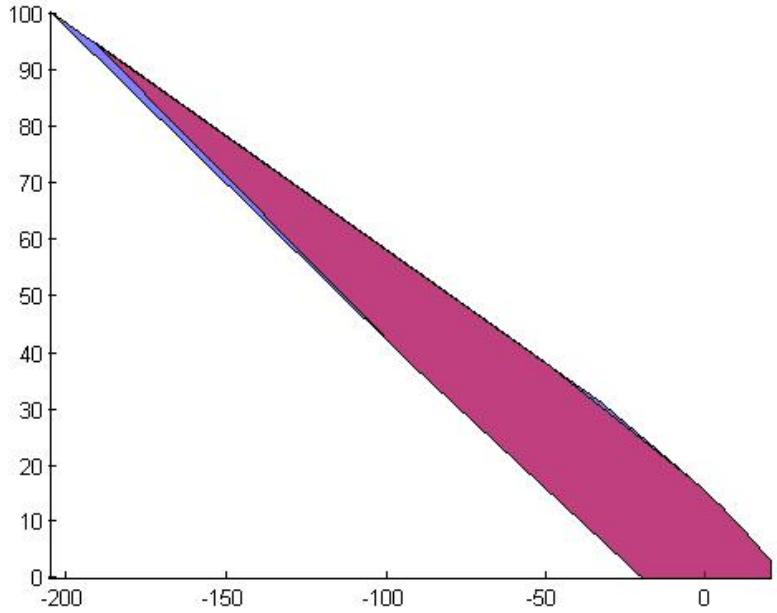
Experimentations

Program		Acceleration process		Polyhedral analysis
Name	$\|V\|$	$\|\Delta\|$	t(s)	t(s)
`prog`	2	308	2.202	52.27
`filter1`	4	332	2.202	TO
`filter2`	4	332	6.264	TO
`filter3`	4	332	59.764	TO
`filter4`	5	350	2m10.953	TO
`lead_leg_controller`	5	350	5m58.748	TO
`Dampened_oscillator`	6	332	15.291	TO
`Harmonic_oscillator`	6	332	4.882	TO
`lp_iir_9600_2`	6	372	1m13.537	TO

The Experimentation



The program called `filter2`



The program called `prog`

Conclusion and Perspective

Thank you for your attention !