Automatic Abstraction of Transistor Level Circuits to Hybrid Automata

Ahmad Tarraf, Lars Hedrich
University of Frankfurt. Germany

FAC' 18, Wien

Outline

- Introduction
- Overview
- Abstraction of analog behavior
- Experimental results
- Conclusion
- Future Goals

Introduction

Circuits controlling safety-critical systems:

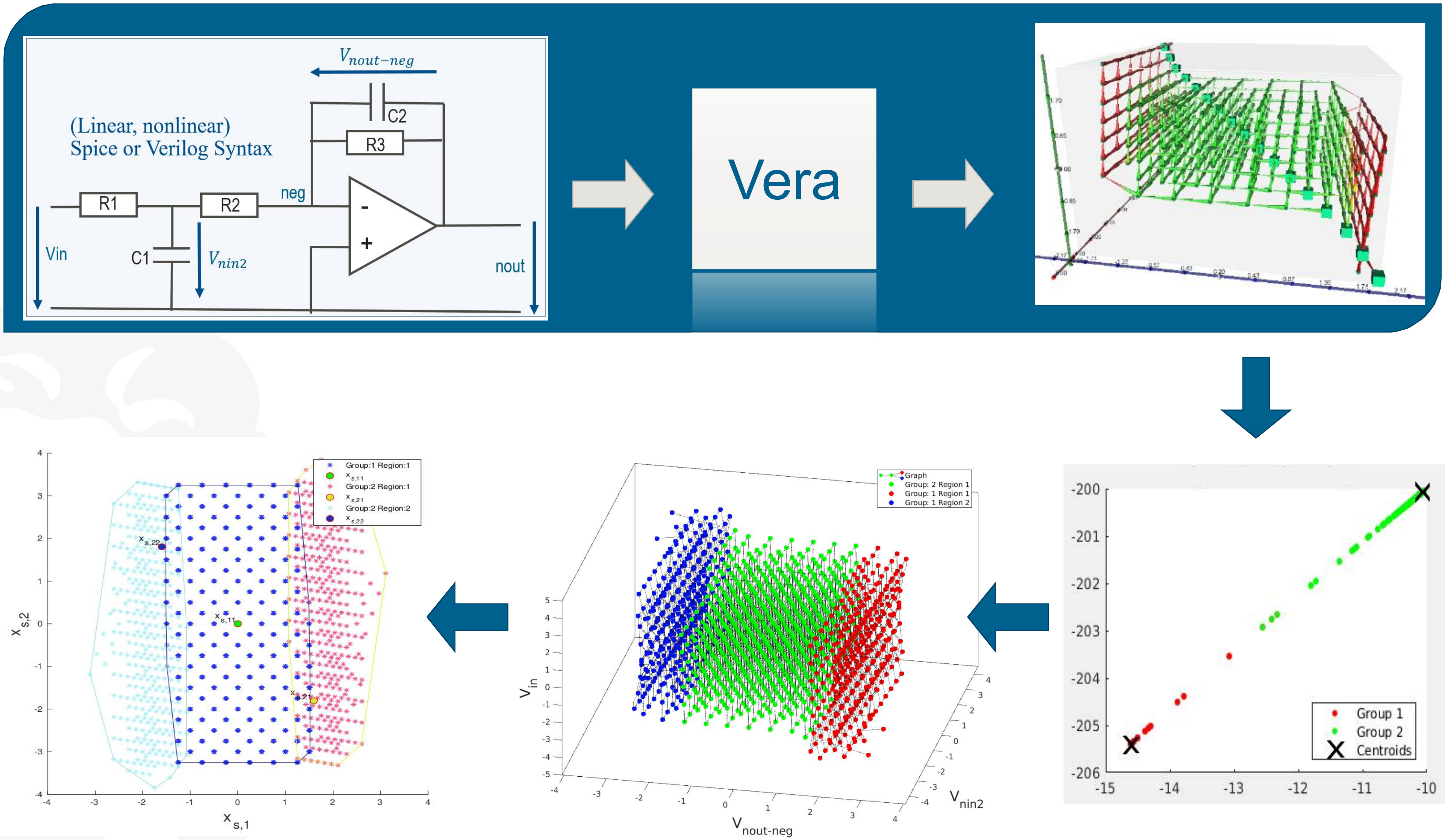
- Automated vehicles
- Robotic surgery
- Automatic systems in emergency rooms
- Human-robot collaboration

Formal verification: digital domain $\neq$ hybrid domain $\neq$ transistor level

Approaches for formal verification on transistor level :

- Not scalable (5-40 Transistors)
- Exception: formal verification of linear circuits
- The nonlinear behavior often causes system failure
- Misses an automatic and formal process for abstract model generation

Overview

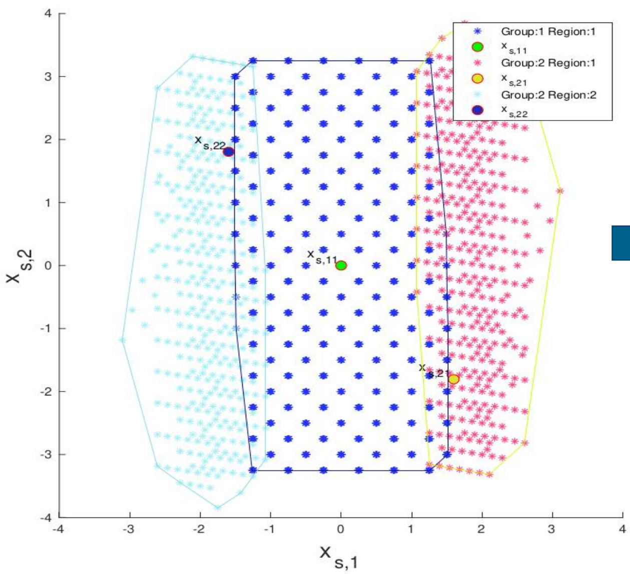


Hybrid states identification

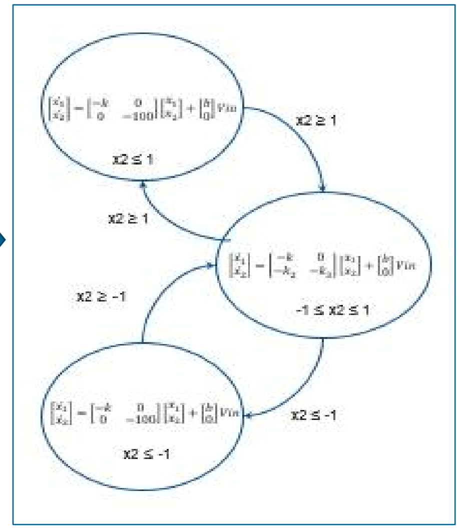
Region identification

Eigenvalues clustering

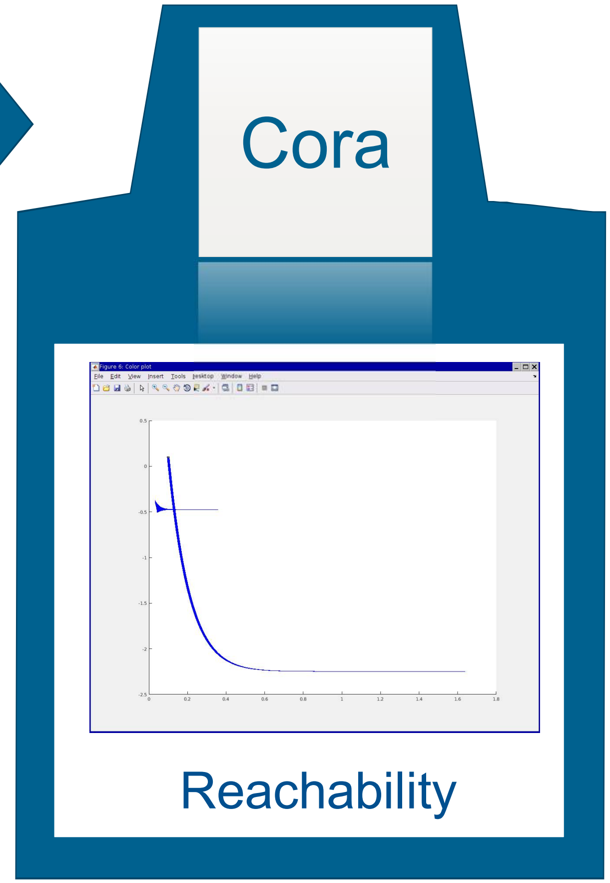
Overview

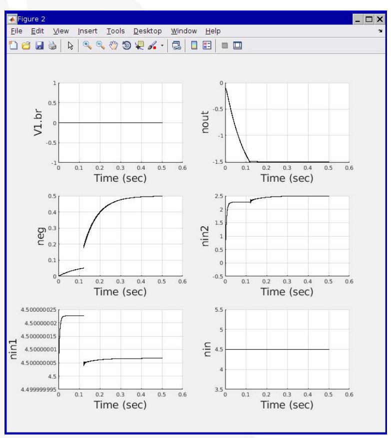


Hybrid states identification



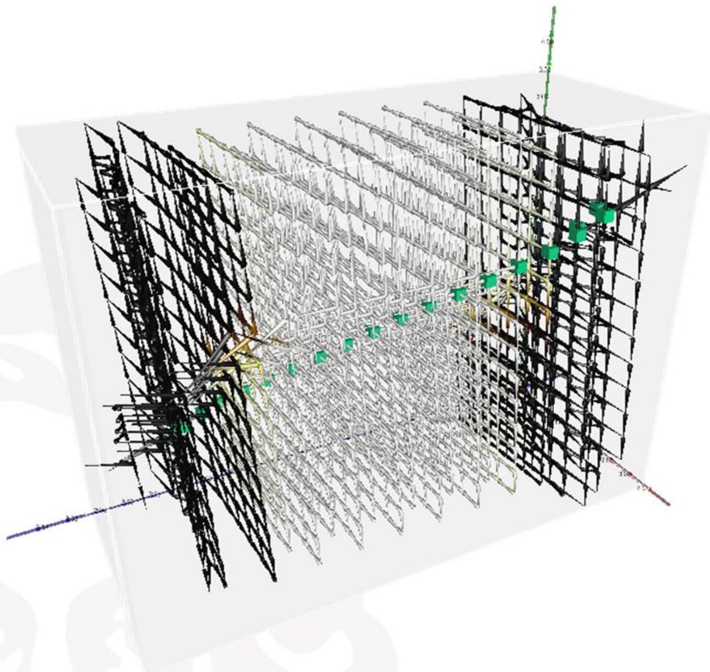
Hybrid automata



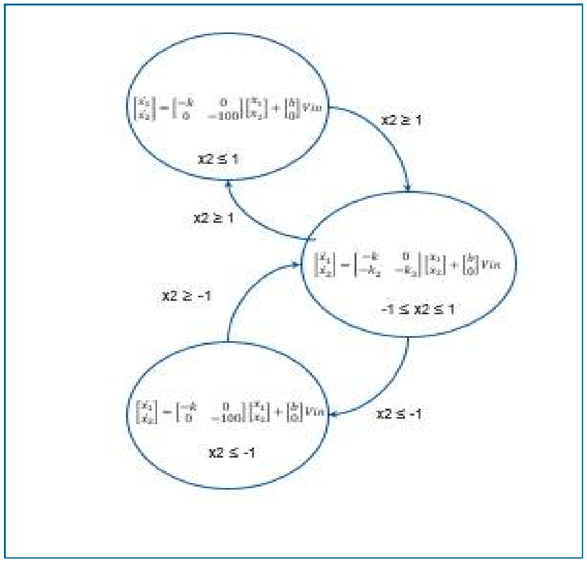


Back transformation

Abstraction of Analog Behavior:



Discretized state space



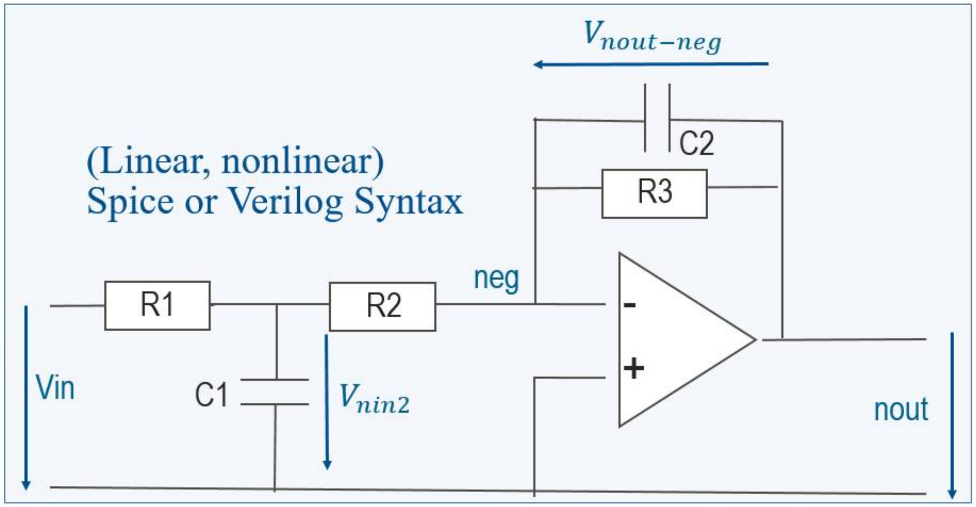
Hybrid automata

A State Space Sampler: Vera

X → Vector of unknowns

e.g: $[i_{in}\ V_{nout}\ V_{neg}\ V_{nin2}\ V_{in}]$‘

Charge oriented equation:



$$\underline{A} \cdot \dot{\vec{q}} + \vec{f}(\vec{x}) = \vec{0}$$

$$\vec{q} = \vec{f}_q(\vec{x})$$

DC and all other operating points

$f(x) = 0$:

$x_{new} = x_{old} + {J_f}^{-1} f(x)$

linearization

$$\underline{J}_f = \frac{\partial \vec{f}(\vec{x})}{\partial \vec{x}}$$

$$\underline{J}_{f_q} = \frac{\partial \vec{f}_q(\vec{x})}{\partial \vec{x}}$$

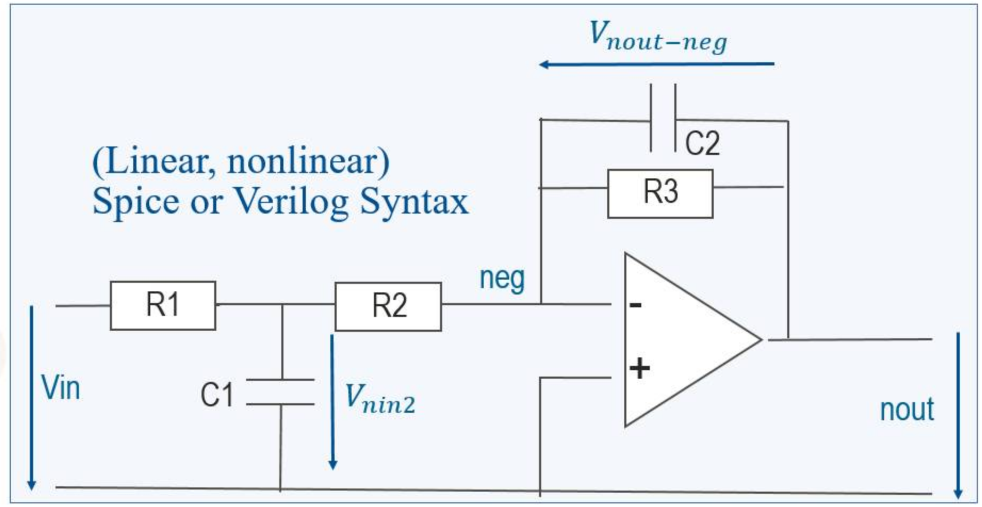
$$\underline{G} = \underline{J}_f$$

$$\underline{C} = \underline{A} \cdot \underline{J}_{f_q}$$

$$\underline{C} \cdot \dot{\vec{x}} + \underline{G} \cdot \vec{x} = \vec{0}$$

State Space Sampler: Vera

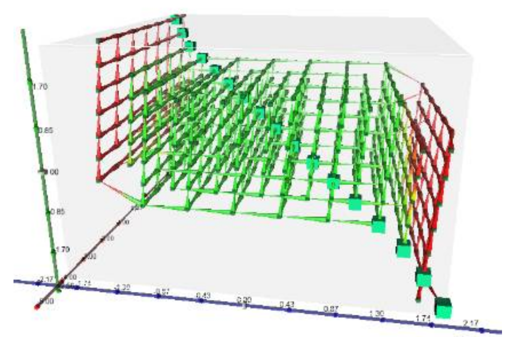
$$\underline{C} \cdot \dot{\vec{x}} + \underline{G} \cdot \vec{x} = \vec{b} \cdot u$$
$$y = \vec{r}^T \cdot \vec{x}$$



$$s \cdot \underline{E} \cdot \underline{C} \cdot \underline{F} \cdot \dot{\vec{x}}_s + \underline{E} \cdot \underline{G} \cdot \underline{F} \cdot \vec{x}_s = \underline{E} \cdot \vec{b} \cdot u$$
$$y = \vec{r}^T \cdot \underline{F} \cdot \vec{x}_s$$

Vera

Kronecker form + order reduction



$$s \cdot \begin{bmatrix} \underline{I} & \underline{0} \\ \underline{0} & \underline{0} \end{bmatrix} \begin{bmatrix} \dot{\vec{x}}_{s,Re} \\ \dot{\vec{x}}_{s,\infty} \end{bmatrix} + \begin{bmatrix} \underline{\lambda}_{red} & \underline{0} \\ \underline{0} & \underline{I} \end{bmatrix} \begin{bmatrix} \vec{x}_{s,Re} \\ \vec{x}_{s,\infty} \end{bmatrix} = \begin{bmatrix} \vec{\tilde{b}}_{Re} \\ \vec{\tilde{b}}_{\infty} \end{bmatrix} \cdot u$$

$$y = \begin{bmatrix} \vec{\tilde{r}}^T_{Re} \vec{\tilde{r}}^T_{\infty} \end{bmatrix} \begin{bmatrix} \vec{x}_{s,Re} \\ \vec{x}_{s,\infty} \end{bmatrix}$$

Parsing State Space Data

parser

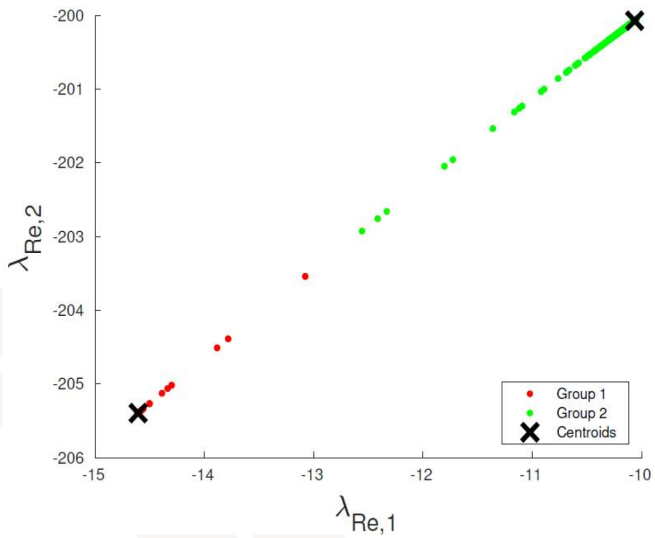
Vera → .acv file ➡ matlab structure

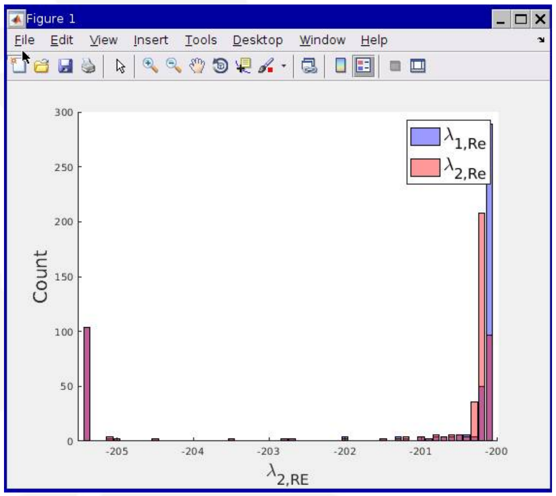
```
id{106}
leafid{105}
inputexpand{1}
time{0.0}
}
transition{
id{1}
leafid{0}
inputexpand{0}
time{0.0}
}
}
center{0 1.5 -0.333333 -2.16667 -4 -4 0 0 -14.6061 0 -205.394 0 0
}
p2ptransitions{
count{1}
p2ptransition{
start{0 1.5 -0.333333 -2.16667 -4 -4 0 0 -14.6061 0 -205.394 0 0
end{0 1.5 -0.333333 -2.16667 -4 -4 0 0 -14.6061 0 -205.394 0 0
time{1e+99}
length{0}
}
```

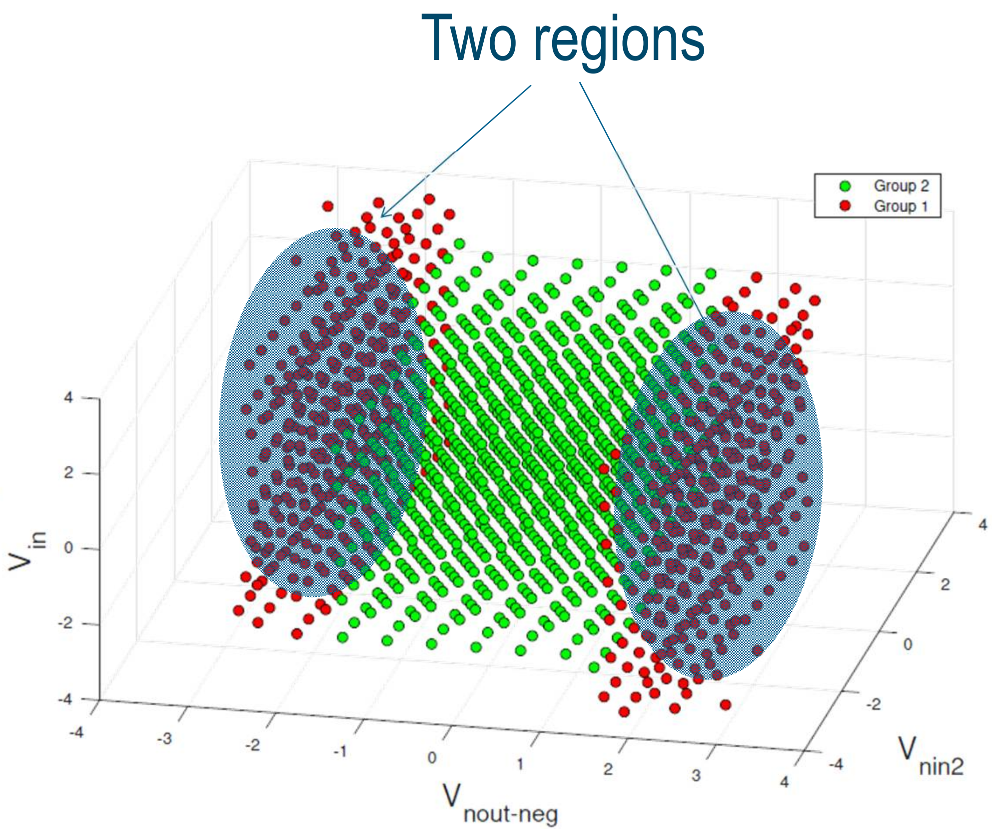
```
space.center(1,:)  =  [ 0;  1;  -0.333333;  -1.66667;  -3;
;  0;  1;  0;  1;  1.33333;  -1.66667;  -3;   ] ;
space.Eigvright(:,1,1) =  [ 0;  -4.60608e-09;  1;  0.539392
space.Eigvright(:,2,1) =  [ 0;  -1.05394e-08;  0.0539392;
space.Eigvright(:,3,1) =  [ 1;  0;  0;  0;  0;  0;   ] ;
space.Eigvright(:,4,1) =  [ 0;  0;  0;  0;  0;  1;   ] ;
space.Eigvright(:,5,1) =  [ 0;  -1;  -1;  3.19022e-15;  7.2
space.Eigvright(:,6,1) =  [ 0;  0;  0;  0;  1;  0;   ] ;
space.Eigvleft(:,1,1) =  [ 5.24142;  -4.47586e-05;  9717.28
space.Eigvleft(:,2,1) =  [ -97.1728;  -0.00102414;  5241.42
space.id(2)=2;
space.Graph=addedge(space.Graph,2,1);
space.Graph=addedge(space.Graph,2,369);
space.center(2,:)  =  [ 0;  1;  -0.533333;  -1.77455;  -3;
 0;  0;  0;  0;  1.33;  1.53333;  -1.77455;  -3;   ] ;
space.Eigvright(:,1,2) =  [ 0;  -4.60608e-09;  1;  0.539392
space.Eigvright(:,2,2) =  [ 0;  -1.05394e-08;  0.0539392;
space.Eigvright(:,3,2) =  [ 1;  0;  0;  0;  0;  0;   ] ;
space.Eigvright(:,4,2) =  [ 0;  0;  0;  0;  0;  1;   ] ;
space.Eigvright(:,5,2) =  [ 0;  -1;  -1;  3.19022e-15;  7.2
space.Eigvright(:,6,2) =  [ 0;  0;  0;  0;  1;  0;   ] ;
```

State space Clustering using Eigenvalues

Create cluster with k-means + The silhouette method







Region Identification

Three approaches to find the regions inside a group:

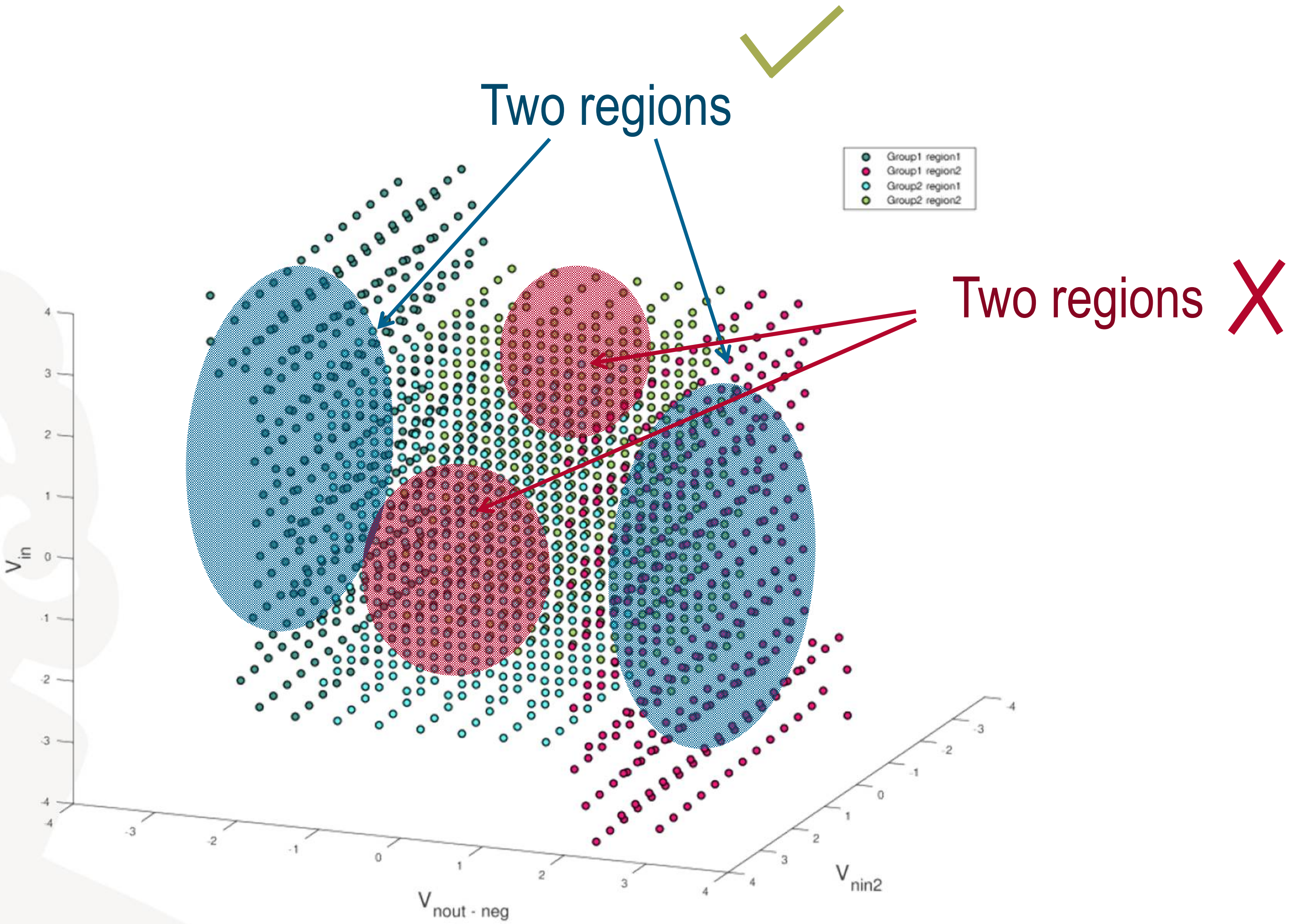
Approach 1: using k-means on the energy states

Approach 2: breadth-first search

Approach 3: connection Graph

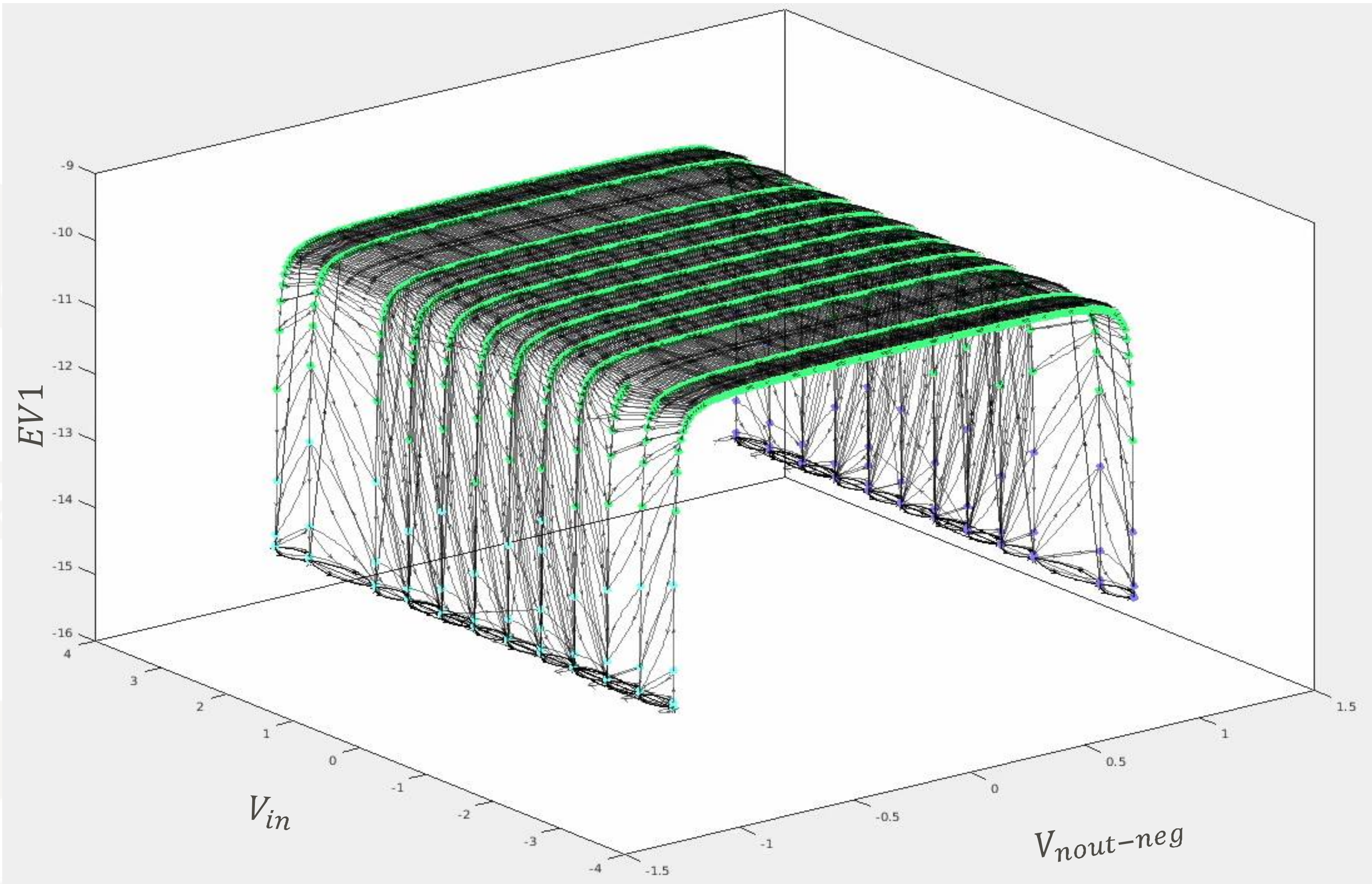
Region Identification: Approach 1

Using k-means on the states



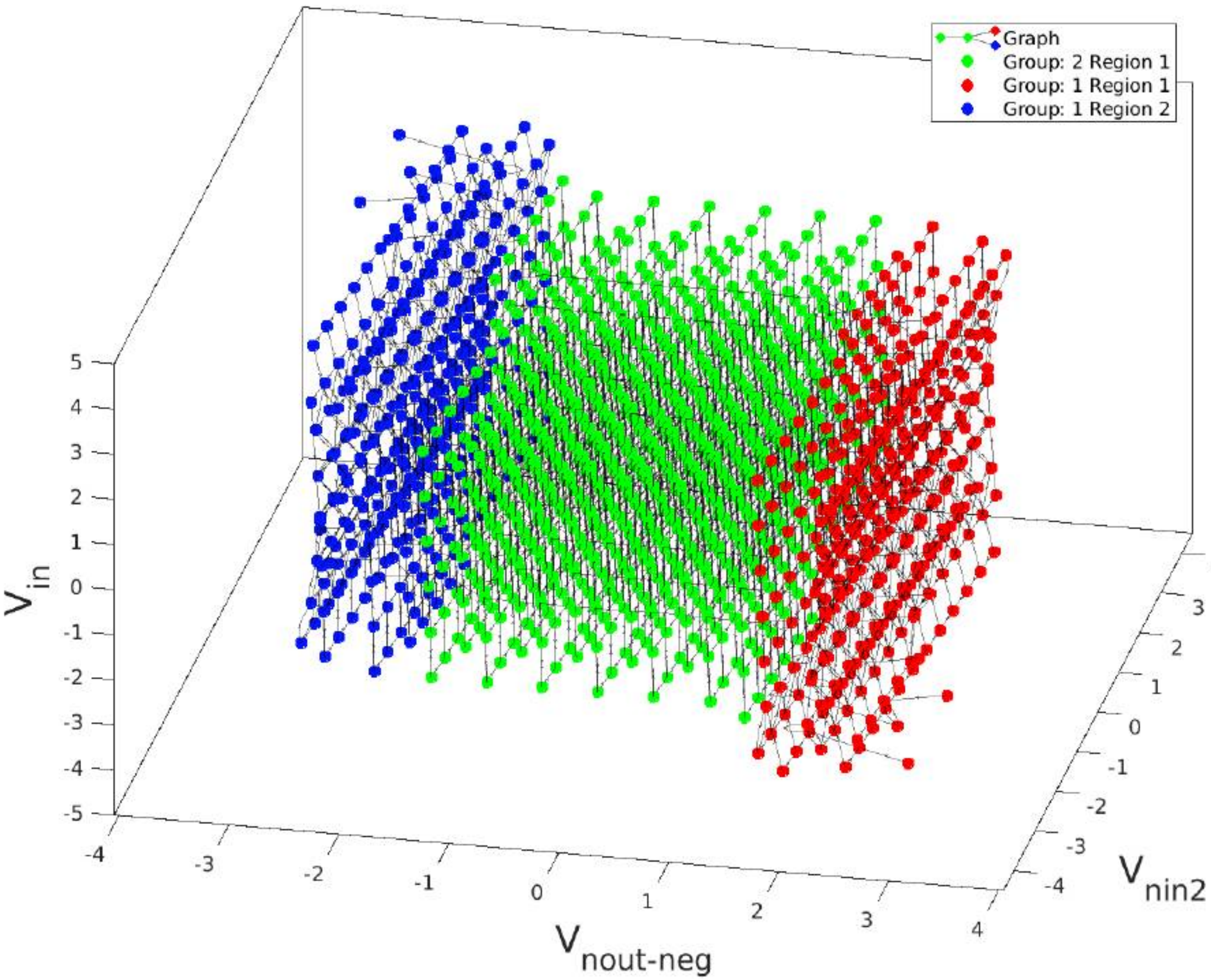
Region Identification: Approach 2

Breadth-first search on connection graph



Region Identification: Approach 2 (cont.)

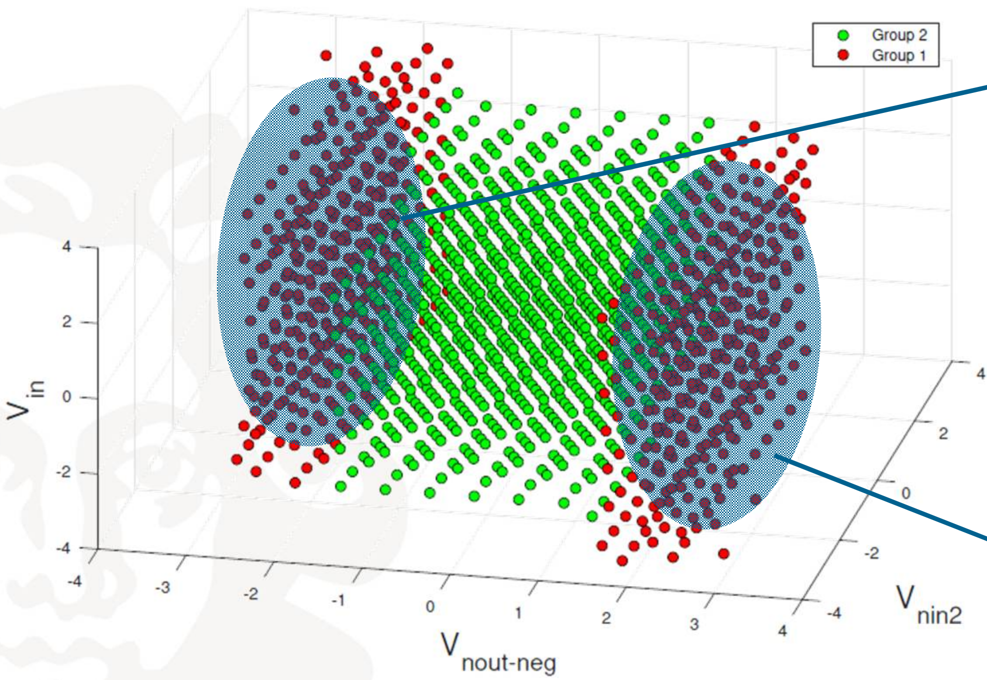
Breadth-first search on connection graph

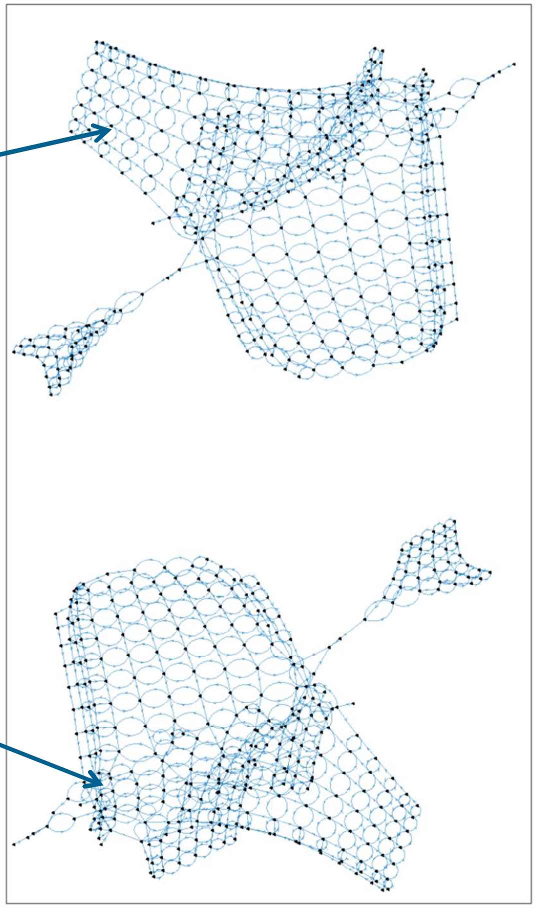


Region Identification: Approach 3

Connection Graph

+ correction: distance smaller than discretization step?





Locations of Hybrid Automata

Guards and invariants ?

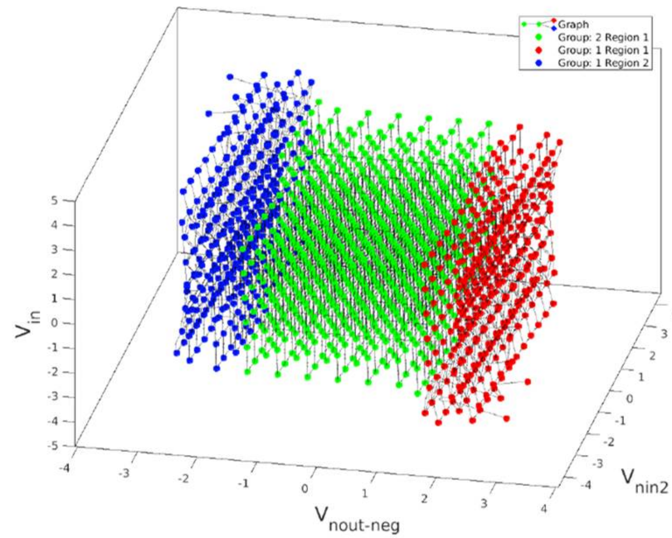
For each region:

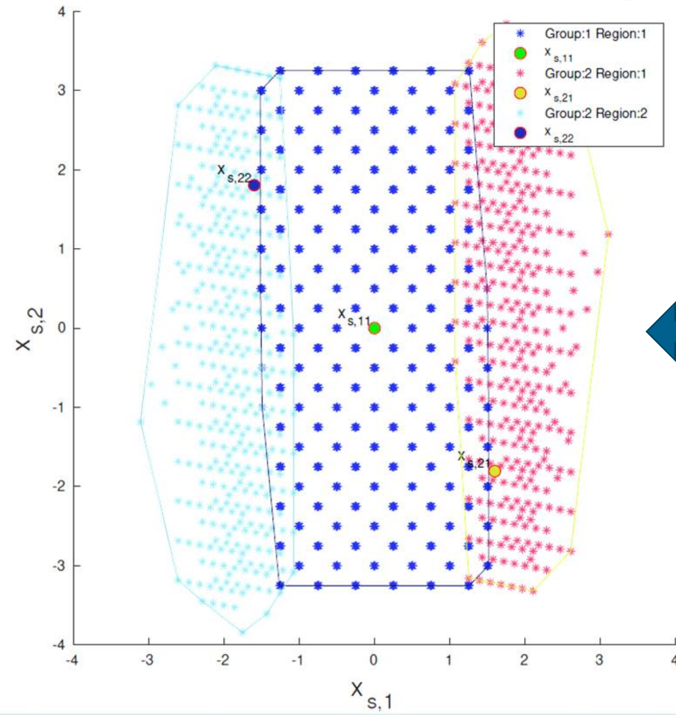
1. Calculate the operating point in the region In the X_s Domain (Reduced state space)

2. Based on the o.p. transform all points from the X domain to the X_s domain

3. Use convex hulls on the regions
   ➡ Invariants

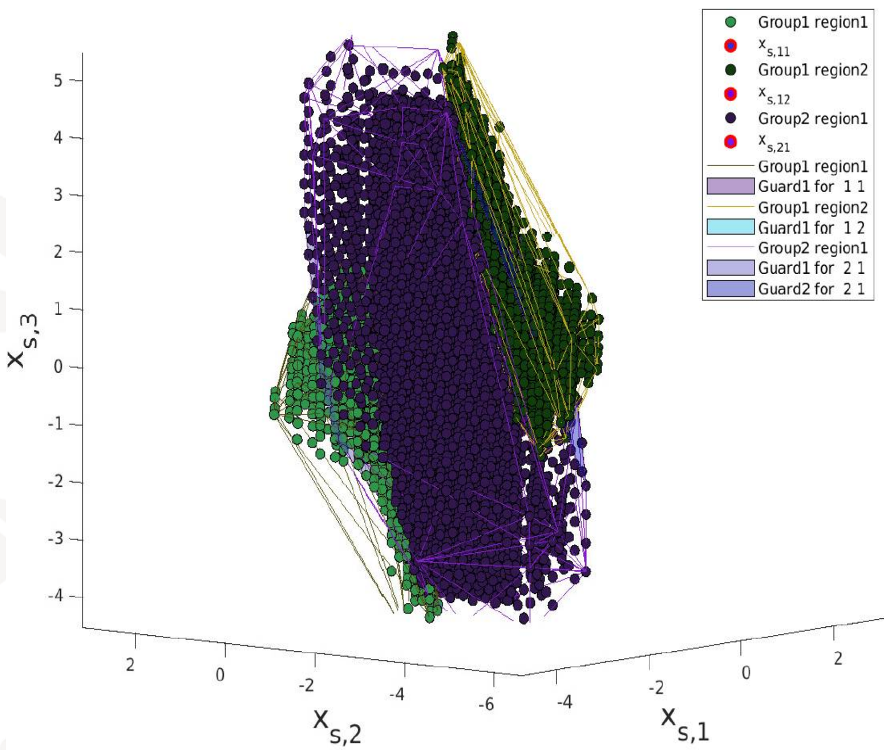
4. Edges of the convex hulls to the other Groups
   ➡ Guards

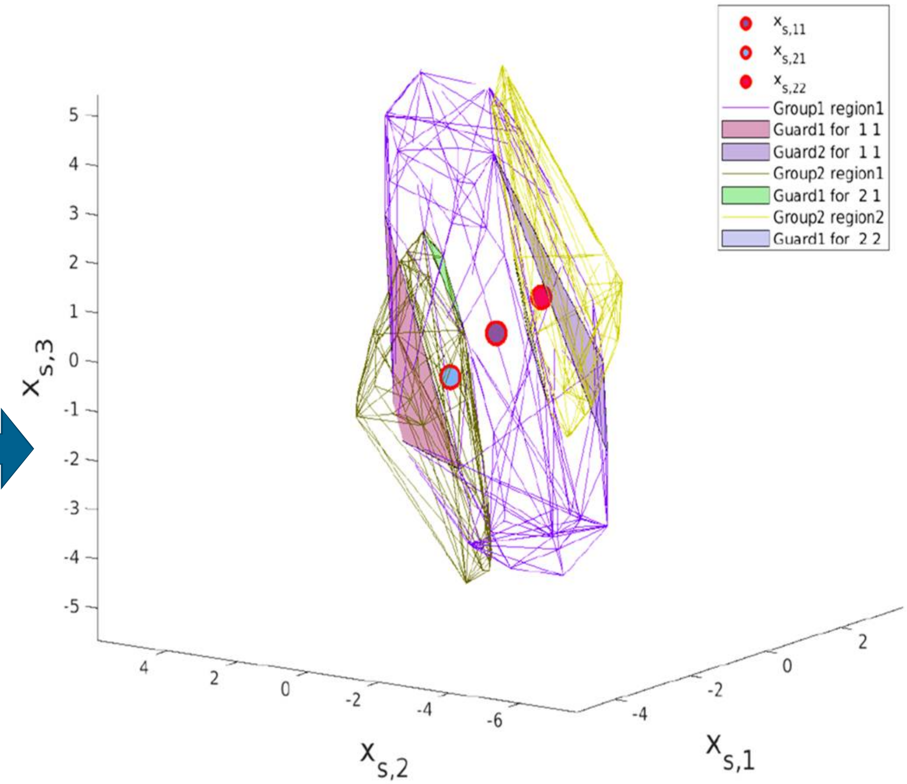




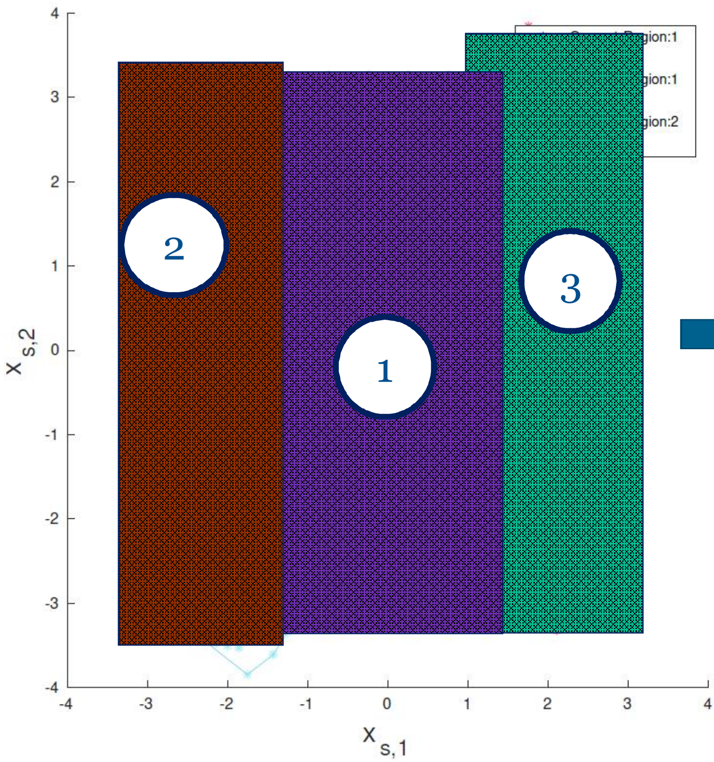
Guards and Invariants

Works for n dimensions: e.g 3rd order lowpass





Guards and Invariants



$$\dot{\vec{x}}_{hyb,3} = \begin{bmatrix} -14.8 & 0 \\ 0 & -205.4 \end{bmatrix} \vec{x}_{hyb,3} + \begin{bmatrix} 5.1 \\ -97.2 \end{bmatrix} \cdot (u + 3.5)$$

3

$$\text{reset} := x_{DC,3}$$

$$[1, 0]' \cdot \vec{x}_s \leq -1.5$$

...

$$\dot{\vec{x}}_{hyb,1} = \begin{bmatrix} -10.1 & 0 \\ 0 & -200.2 \end{bmatrix} \vec{x}_{hyb,1} + \begin{bmatrix} 5.1 \\ -99.9 \end{bmatrix} \cdot (u)$$

1

$$\text{reset} := x_{DC,2}$$

$$[-1, 0]' \cdot \vec{x}_s \leq -1.43$$

...

$$\dot{\vec{x}}_{hyb,2} = \begin{bmatrix} -14.8 & 0 \\ 0 & -205.4 \end{bmatrix} \vec{x}_{hyb,2} + \begin{bmatrix} 5.2 \\ -97.2 \end{bmatrix} \cdot (u - 3.5)$$

2

Experimental Results

$$\dot{\vec{x}}_{hyb,3} = \begin{bmatrix} -14.8 & 0 \\ 0 & -205.4 \end{bmatrix} \vec{x}_{hyb,3} + \begin{bmatrix} 5.1 \\ -97.2 \end{bmatrix} \cdot (u + 3.5)$$

3

reset $:= x_{DC,3}$

$[1,0]' \cdot \vec{x}_s \leq -1.5$

...

$$\dot{\vec{x}}_{hyb,1} = \begin{bmatrix} -10.1 & 0 \\ 0 & -200.2 \end{bmatrix} \vec{x}_{hyb,1} + \begin{bmatrix} 5.1 \\ -99.9 \end{bmatrix} \cdot (u)$$

1

reset $:= x_{DC,2}$

$[-1,0]' \cdot \vec{x}_s \leq -1.43$

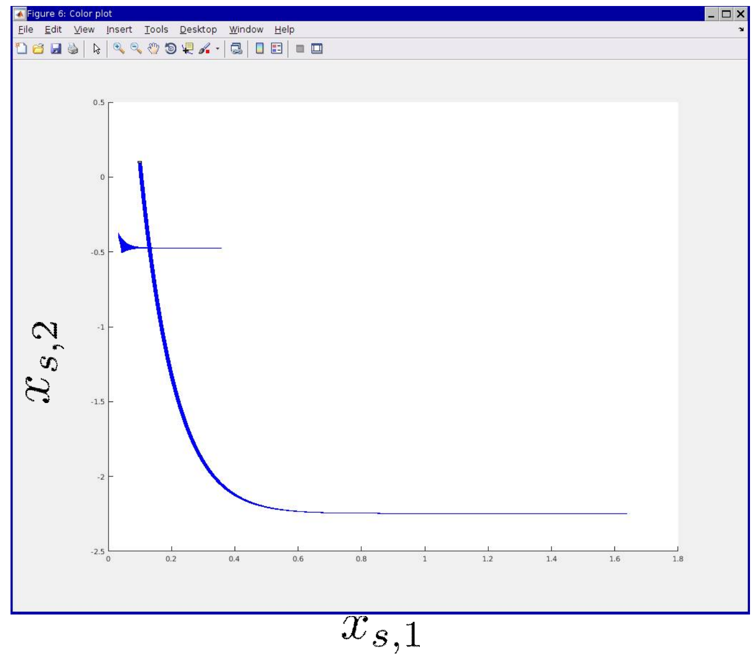
...

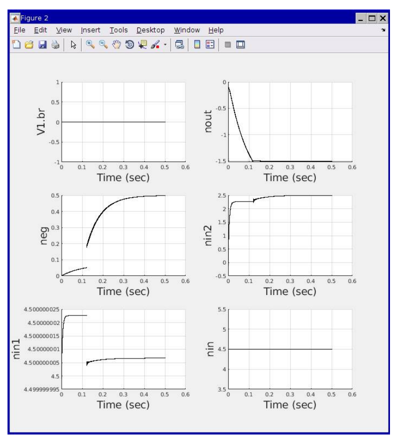
$$\dot{\vec{x}}_{hyb,2} = \begin{bmatrix} -14.8 & 0 \\ 0 & -205.4 \end{bmatrix} \vec{x}_{hyb,2} + \begin{bmatrix} 5.2 \\ -97.2 \end{bmatrix} \cdot (u - 3.5)$$

2

Cora

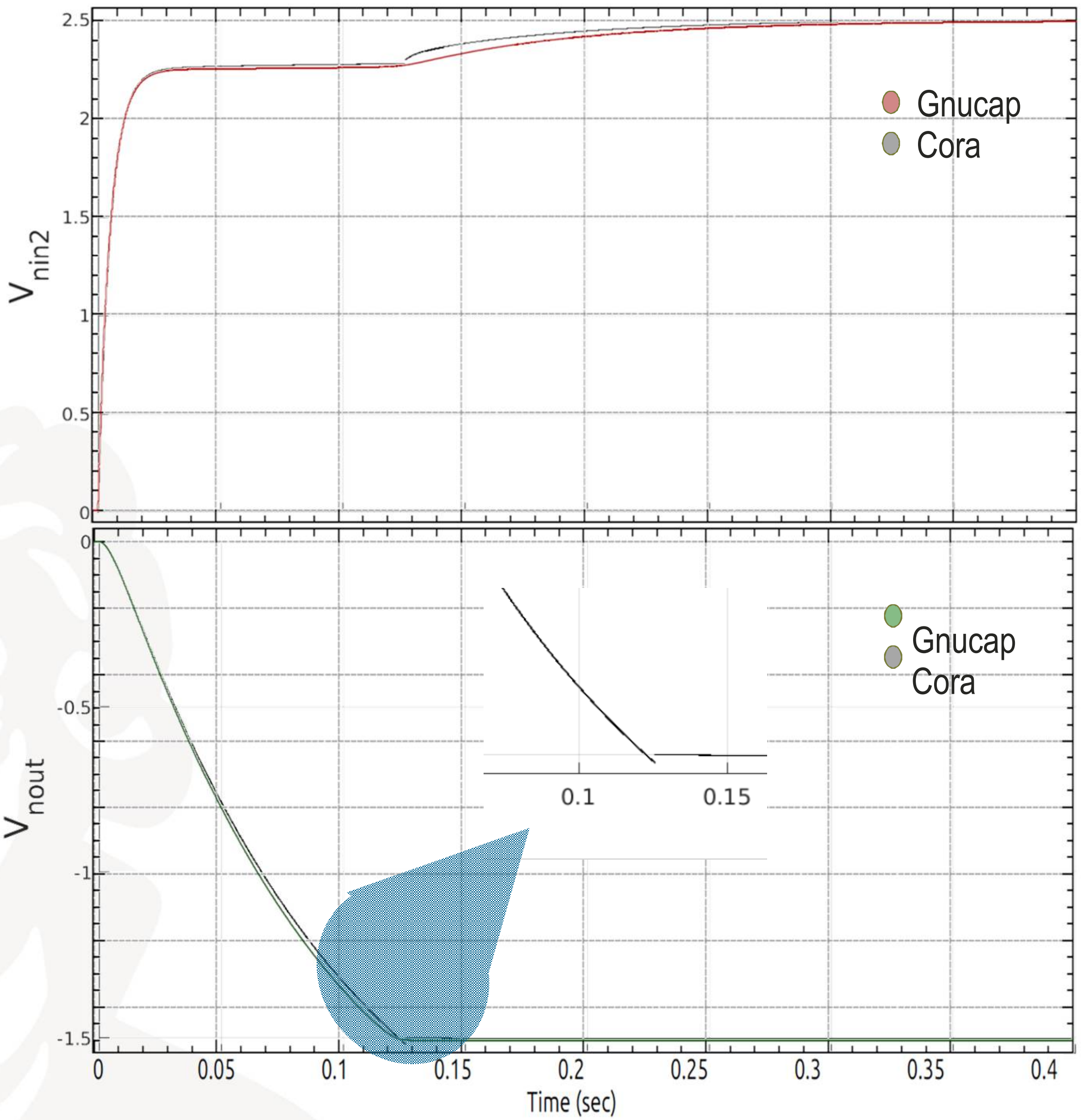
Reachability



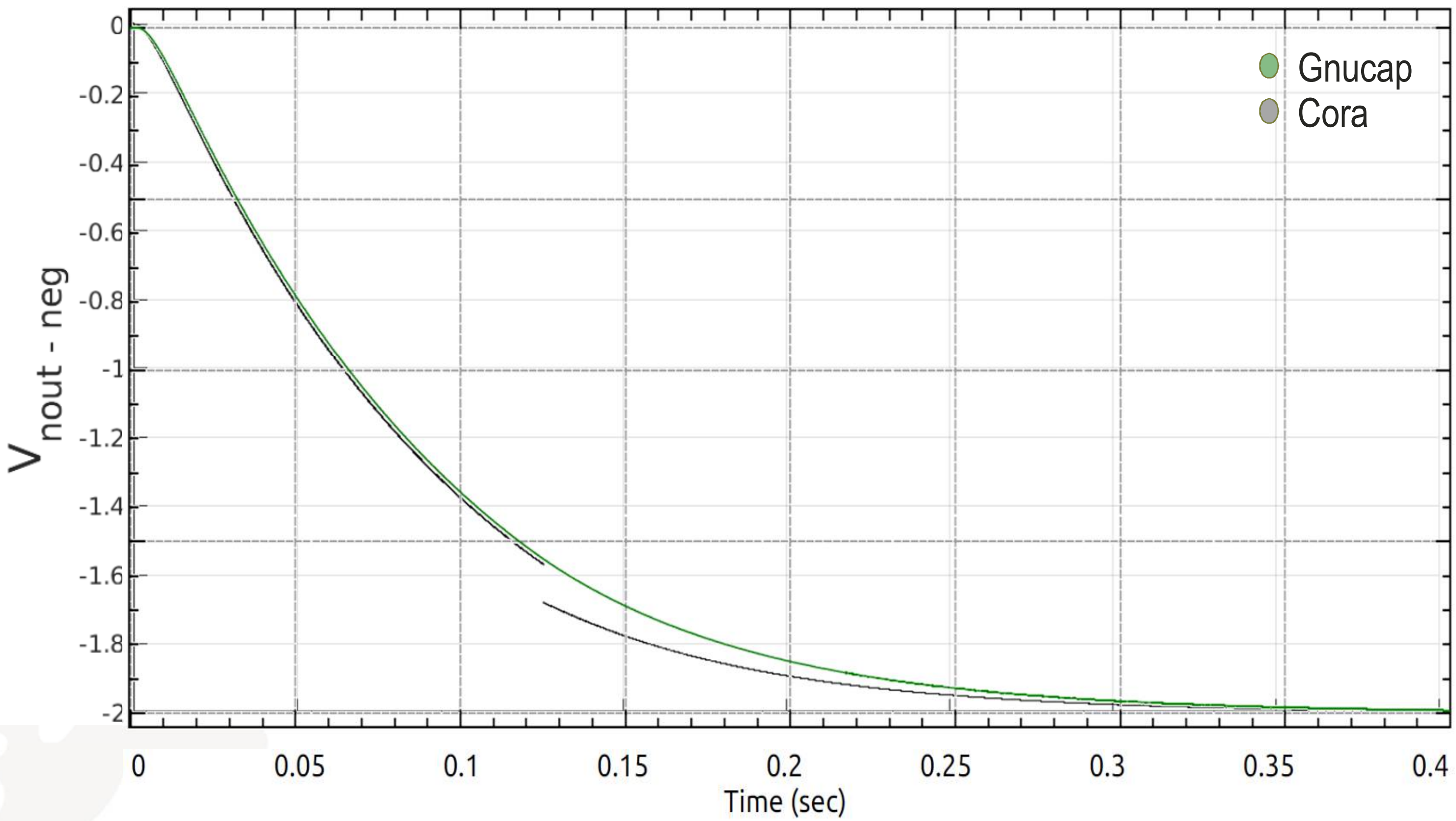


Back transformation

Comparison Netlist vs. Hybrid Automata



Comparison Netlist vs. Hybrid Automata (cont.)



Experimental results

	2nd order low pass	3rd order low pass	RCLD	Bandpass
Sampled points	1619	12256	30716	3801
Reachable points	501	1043	1437	2428
Order	2	3	2	2
Dim $\underline{G}$	6x6	7x7	6x6	17x17
Points with silhouette > 0.8	10	12	26	590
Percentage of outcasts silhouette	0.74 %	2%	1.8 %	24 %
Parser Time	0.72	5.99	21.17	1.66
Matlab read time	4	56.5	74.84	15.85
Abstraction time	2.65	10.83	2.66	3.97
Abstraction time with plot	4.753	13.32	5.6	6.51
Max edges of convex hull	8	56	10	12
Locations	3	3	2	2

Conclusion

Automatic abstraction of analog circuits and systems to hybrid automata

- makes use of system properties (eigenvalues)
- clusters a discretized state space into linear systems
- builds a hybrid automata
- allows performing a reachability analysis

Main Properties:

- States of the hybrid automata are linear
- Large amount of data was clustered into few regions
- Fully automated
- Can be used on nonlinear as well as on linear systems
- Discontinuities in back transformed results

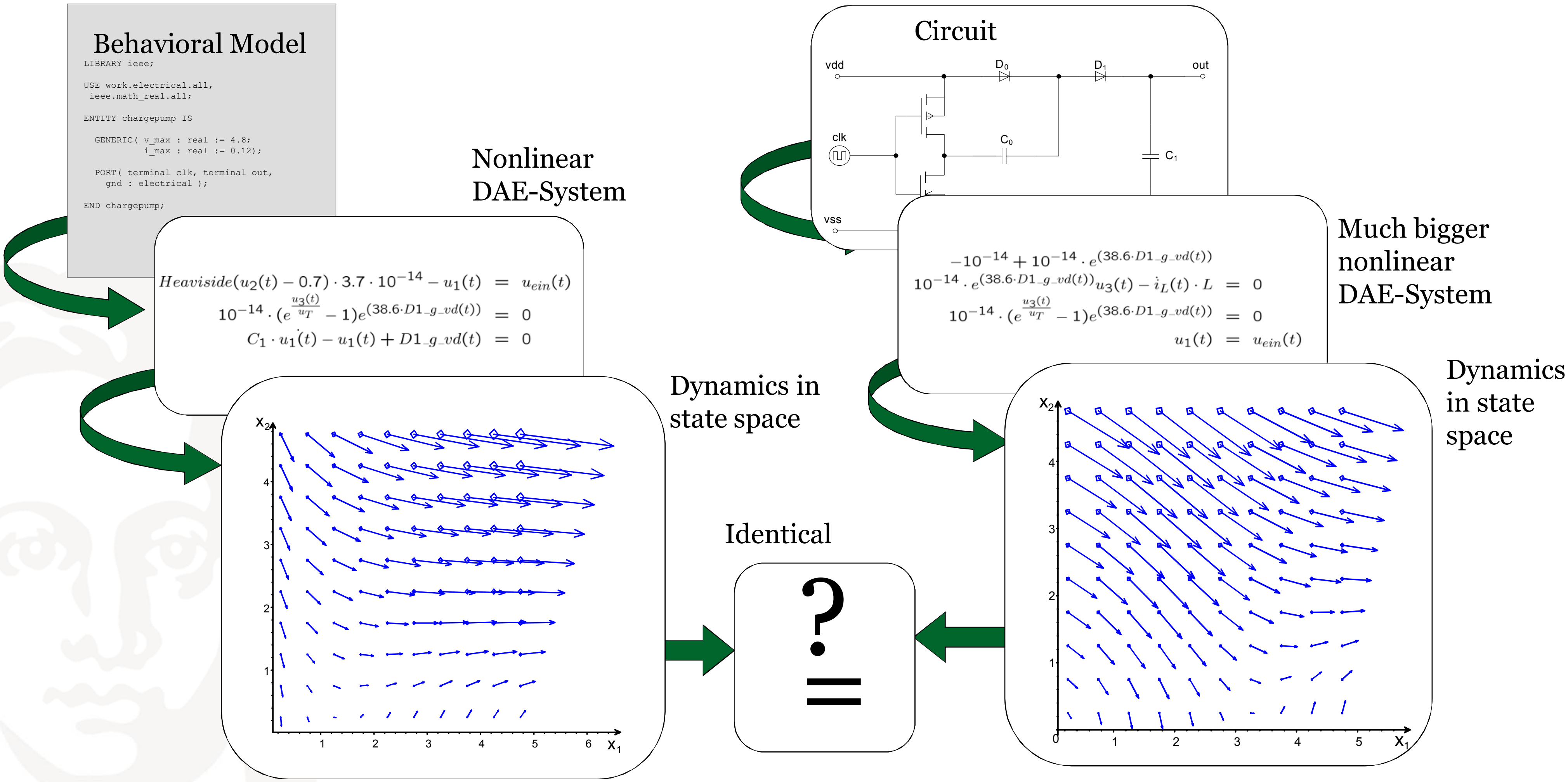
Future Goals

- Compositional hybrid automata
- Higher order examples
- Big netlists
- Error measure

Thank you for your attention

Questions?

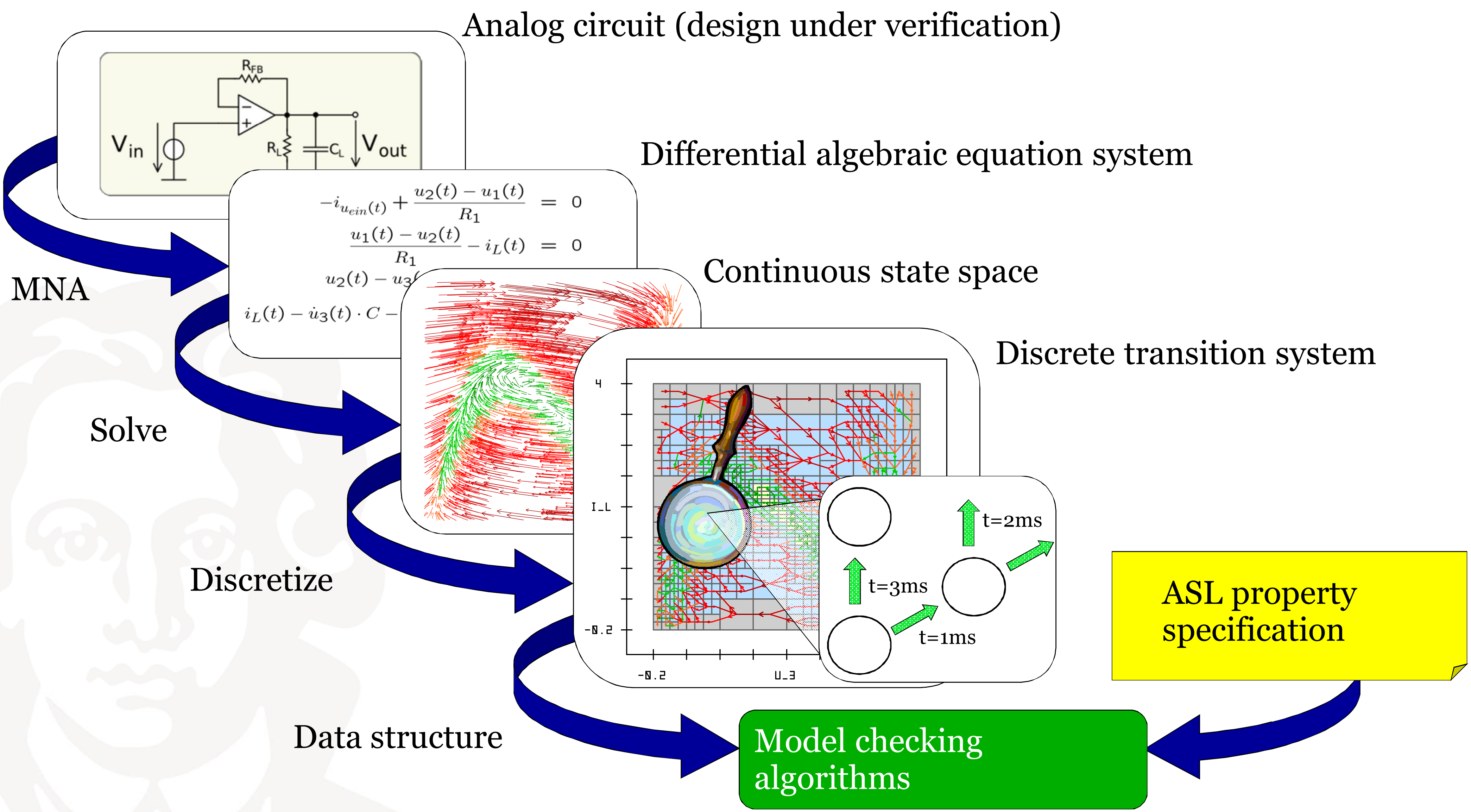
Equivalence Checking Concept



- Use gnucap simulator as back end.

- 50 equations/transistor
- Implicit, strongly nonlinear equations
- Use of numerical evaluation

Model Checking Flow



Vera - Concept

Alog:

Read netlist

For all input voltages

find the DC operating point

Calculate G & C Matrices

Calculate the F matrix

For all point in the state space

shift the point

find a consistent solution

Calculate G & C Matrices

Calculate the F matrix

Bandpass

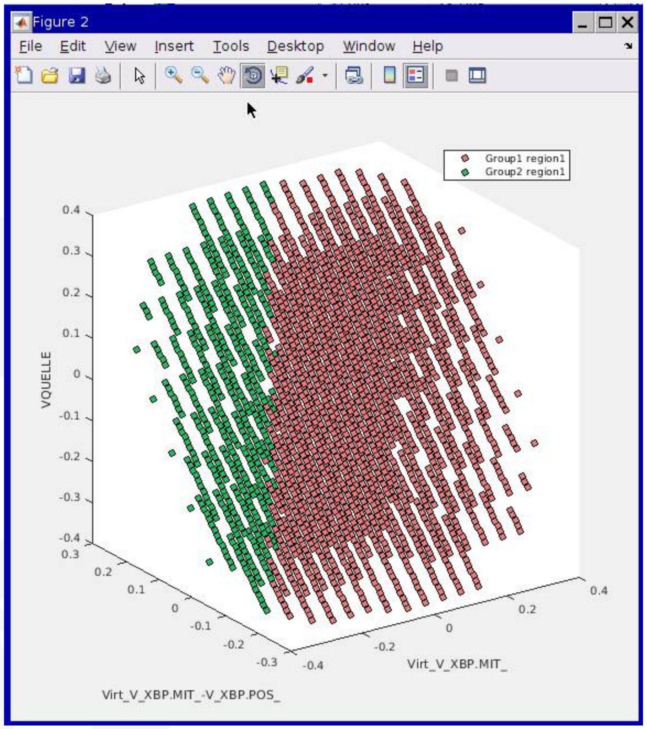


Figure 2
File Edit View Insert Tools Desktop Window Help
Group1 region1
Group2 region1
VQUELLE
Virt_V_XBP.MIT_
Virt_V_XBP.MIT_-V_XBP.POS_

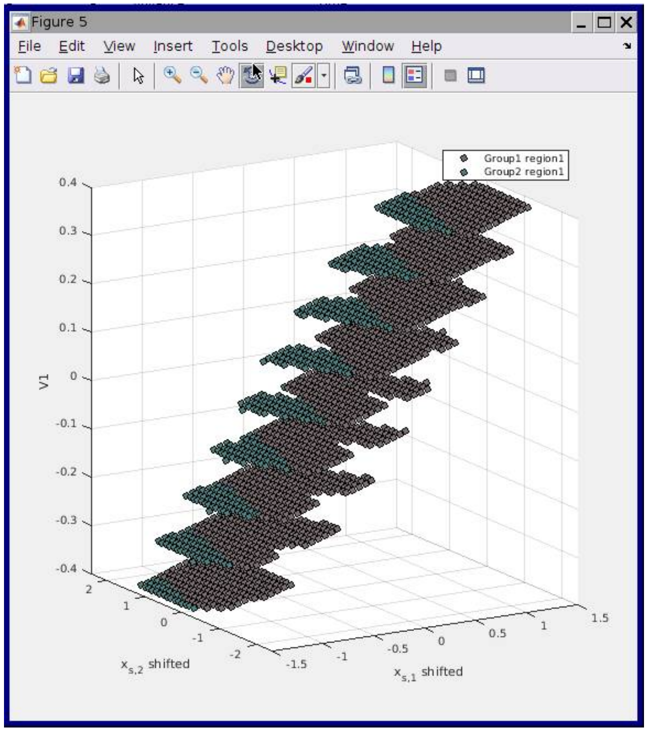
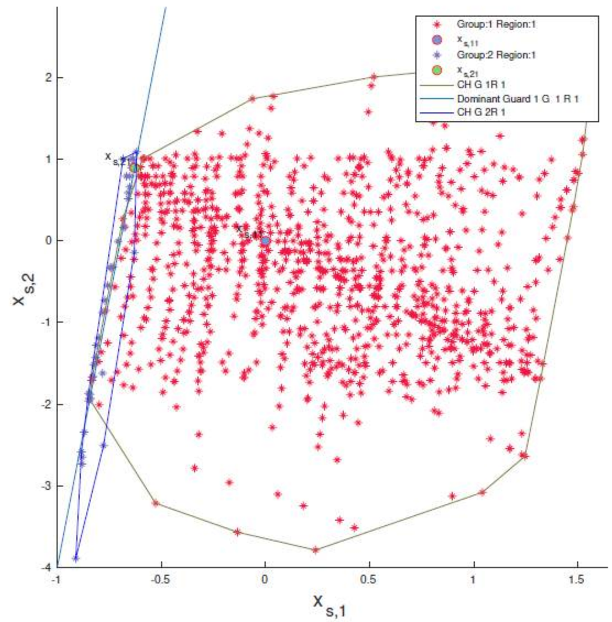


Figure 5
File Edit View Insert Tools Desktop Window Help
Group1 region1
Group2 region1
V1
$x_{s,2}$ shifted
$x_{s,1}$ shifted

RCLD



Group:1 Region:1
x s,11
Group:2 Region:1
x s,21
CH G 1R 1
Dominant Guard 1 G 1 R 1
CH G 2R 1
2
1
0
-1
-2
-3
-4
-1
-0.5
0
0.5
1
1.5
x s,2
x s,1