Verification of Nonlinear Hybrid Systems with Ariadne

Luca Geretti and Tiziano Villa

June 2, 2016

Outline

Outline

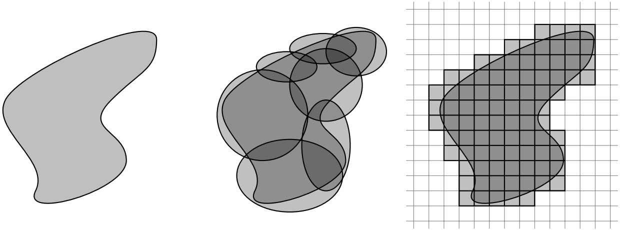
The issue with nonlinearity

Reachable sets cannot be represented in an effective and efficient way

- Most set operations on accurate representations are undecidable.
- Coarse approximations are ultimately needed to recover decidability.
- Set representations play a particularly important role, as a tradeoff between effectiveness and efficiency.

Representing regions of space

- Subsets of $\mathbb{R}^n$ are approximated by finite unions of basic sets:
  - intervals, simplices, cuboids, parallelotopes, zonotopes, polytopes, spheres and ellipsoids, **Taylor sets**.
- Finite unions of basic sets of a given type are called *denotable sets*.



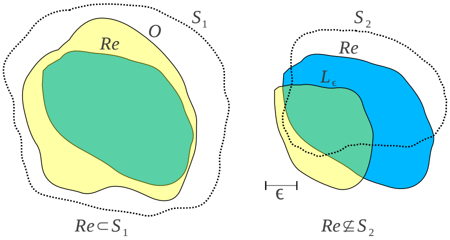
Approximating regions

Approximating Re with A

1. **Inner approximation:** Re strictlay contains A.
2. **Outer approximation:** Re is strictly contained in A.
3. ε**-lower approximation:** every point of A is at distance less than ε from a point of Re.

- Inner approximation is used for specification of system's properties ($System \subseteq Property_{inner} \subseteq Property$), but it is not computable in general.
- Outer and ε-lower approximations can be used to verify property satisfaction.

Property satisfaction in terms of sets



- Re is the (exact) reachable set.
- O is the outer approximation.
- L_ε is the ε-lower approximation.
- S_1, S_2 are sets within which a property is satisfied.

Hybrid regions and Hybrid grid sets

Definition (Hybrid sets)

Hybrid sets are subsets of the space $\mathcal{V} \times \mathbb{R}^n$.

- We start from *hybrid basic sets* that pair a location of the automaton with a single basic set:
  - hybrid intervals, hybrid simplices, hybrid cuboids, hybrid parallelotopes, hybrid zonotopes, hybrid polytopes, hybrid spheres and hybrid ellipsoids, **hybrid Taylor sets**.
- Finite unions of hybrid basic sets are called *hybrid denotable sets*.
- Hybrid sets are approximated by hybrid denotable sets.

Hybrid regions and Hybrid grid sets

Definition (Hybrid grids)

A grid for every location of the automaton. Hybrid sets are approximated by marking the cells on the grids.

Hybrid grid sets are practical but coarse:

- Union, intersection, difference and inclusion can be performed efficiently.
- They introduce large over-approximations.
- They do not scale well when more precision is required.

Outline

Introduction to Ariadne

- Developed by a joint team including the University of Verona, CWI/University of Maastricht, the University of Udine and the company PARADES/ALES (Rome).
- Based on a rigorous mathematical semantics for the numerical analysis of continuous and hybrid systems.
- Exploits reachability analysis to prove properties of nonlinear systems, especially for safety verification.
- Released as an open source distribution.

Approximate Reachability Analysis

Given a hybrid automaton H, an initial set I, ARIADNE can compute:

- an outer approximation of the states reached by H starting from I.
- for a given $\varepsilon > 0$, an ε-lower approximation of the states reached by H starting from I.

The reachability algorithm of ARIADNE for O

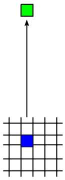
1. Start from the initial Taylor set and compute the continuous evolution of the automaton within the bounding set, until no new states are reached. Mark the cells of the projection of the reach set on the grid, until no more cells can be marked.

2. When no more cells can be marked, compute a single discrete evolution step from the reached Taylor set. Mark the new cells of the grid that are reached by the discrete step.

3. If new cells are reached, go to (1). Otherwise, stop.

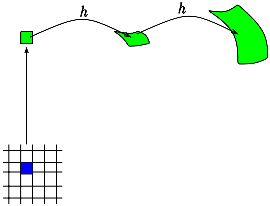
The grid and the bounding set are essential for termination!

Computing the continuous evolution (1)



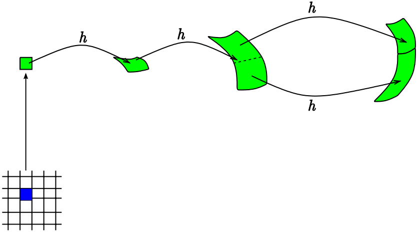
1. Convert the cells to basic sets (Taylor sets + error term).
2. Integrate the continuous dynamics with integration step h for a user-given number of steps.
3. If a set becomes too large, split it.
4. Project back to the grid.

Computing the continuous evolution (1)



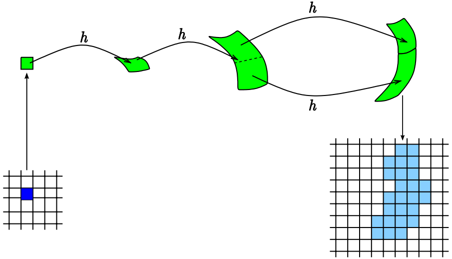
1. Convert the cells to basic sets (Taylor sets + error term).
2. Integrate the continuous dynamics with integration step *h* for a user-given number of steps.
3. If a set becomes too large, split it.
4. Project back to the grid.

Computing the continuous evolution (1)



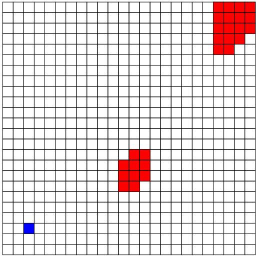
1. Convert the cells to basic sets (Taylor sets + error term).
2. Integrate the continuous dynamics with integration step *h* for a user-given number of steps.
3. If a set becomes too large, split it.
4. Project back to the grid.

Computing the continuous evolution (1)



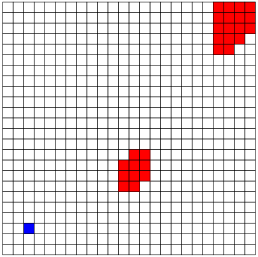
1. Convert the cells to basic sets (Taylor sets + error term).
2. Integrate the continuous dynamics with integration step h for a user-given number of steps.
3. If a set becomes too large, split it.
4. Project back to the grid.

Computing the continuous evolution (2)



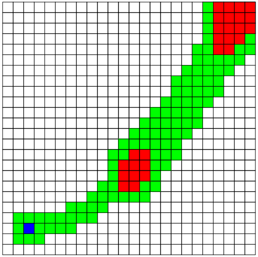
5. Check if new grid cells have been reached.
6. If so, go back to (1).
7. The snapshot is discretised and added to the set of cells. When a new snapshot does not provide additional contribution, the current set of cells is returned as the reachability result.

Computing the continuous evolution (2)



5. Check if new grid cells have been reached.
6. If so, go back to (1).
7. The snapshot is discretised and added to the set of cells. When a new snapshot does not provide additional contribution, the current set of cells is returned as the reachability result.

Computing the continuous evolution (2)



5. Check if new grid cells have been reached.
6. If so, go back to (1).
7. The snapshot is discretised and added to the set of cells. When a new snapshot does not provide additional contribution, the current set of cells is returned as the reachability result.

Computing the discrete evolution

Computing the discrete evolution is simpler:

1. For every control switch $e \in \mathcal{E}$, determine the set of cells that intersect with $Act(e)$.
2. If such set is not empty, apply the reset function $Reset(e)$ to obtain the set of cells reached by the transition.

Upper Semantics:

When there are multiple enabled transitions, or when the system exhibits grazing (tangential contact between a reached region and an activation set), all possible transitions are taken.

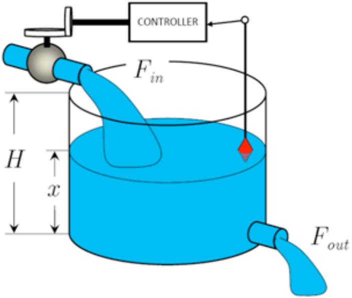
Computing ε-lower evolution

The computation of the ε-lower approximation uses a different algorithm:

1. Start from the initial set and compute the continuous evolution for a time-step t. Do not discretize the result.
2. Compute a single discrete evolution step.
3. If the width of the flow tube is smaller than a given ε, go to (1). Otherwise, discretize the computed set and stop.

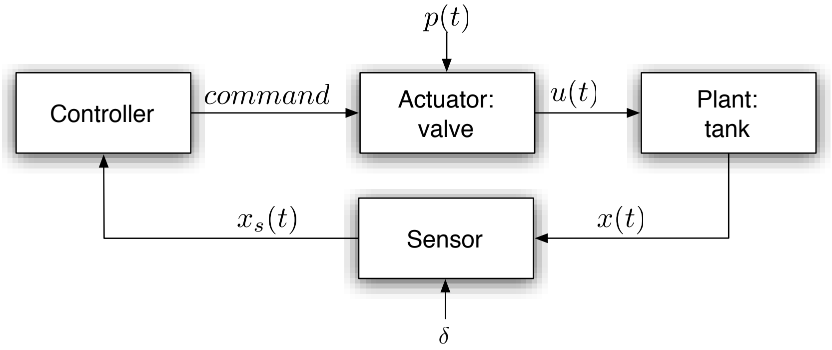
Outline

The watertank example



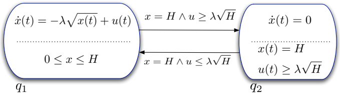
- Outlet flow F_{out} depends on the water level ($F_{out}(t) = \lambda\sqrt{x(t)}$).
- Inlet flow F_{in} is controlled by the valve position ($F_{in}(t) = u(t)$).
- The controller senses the water level and sends the appropriate commands to the valve.

The watertank control loop

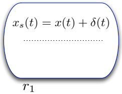


Modeling the water tank

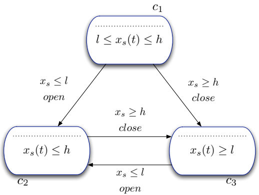
Tank automaton



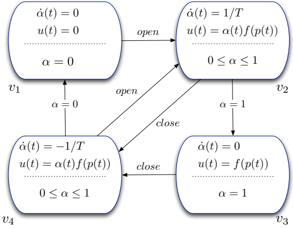
Sensor automaton



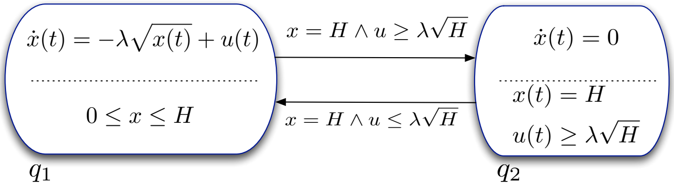
Controller automaton



Valve automaton



The water tank automaton

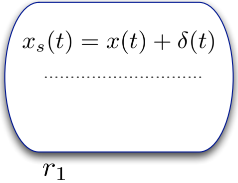


- $x(t)$ it the water level, $u(t)$ is the inlet flow.
- q_1 represents the situation when there is no water overflow.
- q_2 represents the situation when there is water overflow.

The water tank automaton

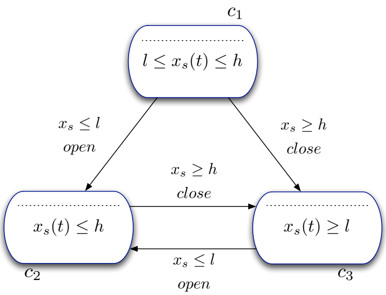
- $\lambda\sqrt{H}$ is the largest outflow $\lambda\sqrt{x}$ when $x = H$.
- $x = H \wedge u > \lambda\sqrt{H}$ is the case when the water is at the top level H and the inflow u is larger than the largest outflow $\lambda\sqrt{H}$.
- when in q_2, if (by the action of the controller) the valve angle decreases, then u decreases to the point that the invariant $x = H \wedge u > \lambda\sqrt{H}$ is not true anymore, and so the transition to q_1 is taken under the guard $x = H \wedge u \leq \lambda\sqrt{H}$.

The sensor automaton



- The input is the real water level $x(t)$ provided by the tank.
- The output is the measured water level $x_s(t) = x(t) + \delta$ for the controller (where δ is an interval $(-\delta_1, \delta_1)$).

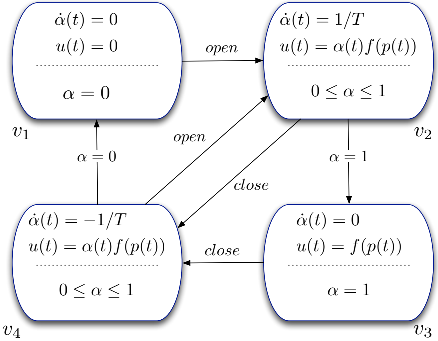
A simple hysteresis controller



- The input is the measured water level $x_s(t)$ provided by the sensor.
- The output is the command signal *open* or *close* for the valve.

- The controller produces the *open* command when $x_s(t) \le l$, and it produces the *close* command when $x_s(t) \ge h$.
- l and h are lower and upper water levels.

The valve automaton

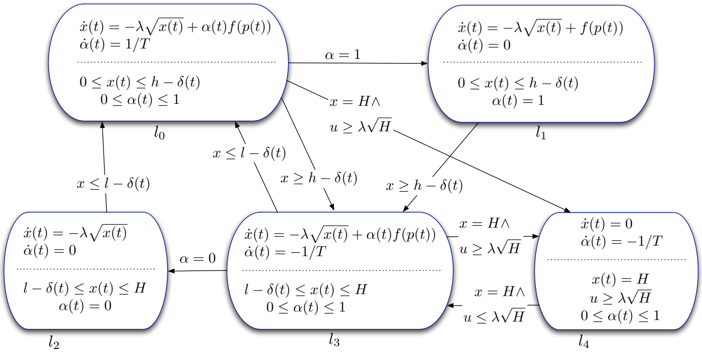


- In response to a command, the valve aperture changes linearly in time with rate $1/T$.

The valve automaton

- The pressure $p(t)$ is assumed to be any constant value in an interval $[p_1, p_2]$, where p_1 and p_2 are respectively the minimum and the maximum of $p(t)$ over a time interval of interest.
- One may assume $f(p(t)) = k\sqrt{p}$, where p is a constant value from an interval (see above) and so it can be used also in a linear model.

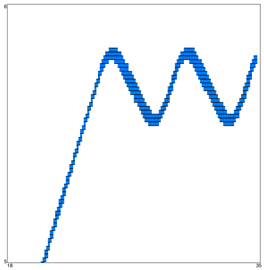
The complete watertank automaton



The complete watertank automaton

- The automaton is obtained by the composition and reduction of all the automata of the system.
- The locations l_0 and l_3 model respectively when the valve is opening and when the valve is closing.
- The locations l_1 and l_2 model respectively when the valve is open and when the valve is closed.
- The location l_4 models when there is overflow; the transitions between l_4 and l_3 handle the passage between overflow and decrease of water below the top level.

Evolution of the watertank



- Horizontal axis: time t; vertical axis: water level $x(t)$.
- The water level in the tank oscillates widely periodically between the lower level l and the upper level h.

Outline

Assume-guarantee system specification

- The system is specified as a set of components.
- Every component is annotated with a pair (A_i, G_i) of assumptions and guarantees.
- The requirements (A, G) of the whole system are decomposed into a set of simpler requirements (A_i, G_i) that, if satisfied, guarantee that the overall requirements (A, G) are satisfied.

Safety checking

Let C be a component of the system, annotated with assumptions A and guarantees G. With ARIADNE we can verify whether the component C respects the safety guarantees G or not given the assumptions A.

- Represent the component C by a hybrid automaton H with inputs and outputs.
- Assumptions A are represented by a hybrid automaton H_A that specifies the possible inputs U for H.
- Guarantees G specify the possible outputs Y of automaton H.

This is a reachability analysis problem:

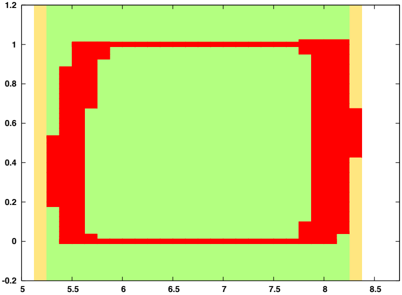
$$Reach(H \| H_A) \subseteq Sat(G).$$

Safety checking by grid refinement

1. Compute an outer-approximation O of $Reach(H \| H_A)$ using a grid of a given size.
2. If $O \subseteq Sat(G)$, the system is verified to be safe. Exit with success.
3. Otherwise, compute an ε-lower approximation L_ε of $Reach(H \| H_A)$. The value of ε depends on the size of the grid (typically, ε is a small multiple of the size of a grid cell).
4. If there exists at least a point in L_ε that is outside $Sat(G)$ by more than ε, the system is verified to be unsafe. Exit with failure.
5. Otherwise, set the grid to a finer size and restart from point 1.

Verifying the water tank

Safety property: the water level between 5.25 and 8.25 meters.



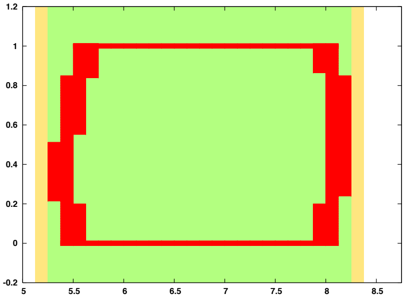
First iteration: grid $1/8 \times 1/80$ (x-axis: $x(t)$, y-axis: $\alpha(t)$).

Outer reach is not safe, try lower reach.

Green: safe set Orange: ε-tolerance Red: computed set

Verifying the water tank

Safety property: the water level between 5.25 and 8.25 meters.



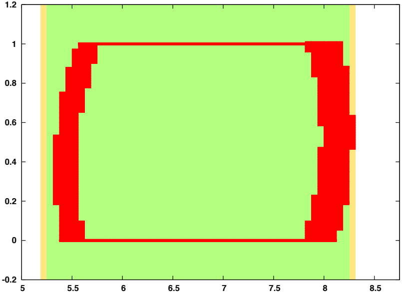
First iteration: grid $1/8 \times 1/80$ (x-axis: $x(t)$, y-axis: $\alpha(t)$).

Lower reach is not unsafe, refine grid.

Green: safe set Orange: ε-tolerance Red: computed set

Verifying the water tank

Safety property: the water level between 5.25 and 8.25 meters.



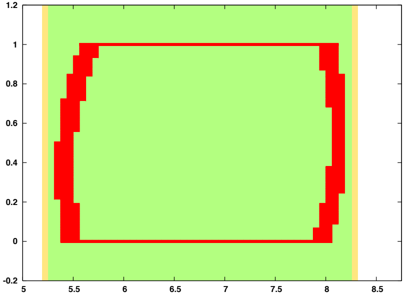
Second iteration: grid $1/16 \times 1/160$ (x-axis: $x(t)$, y-axis: $\alpha(t)$).

Outer reach is not safe, try lower reach.

Green: safe set Orange: ε-tolerance Red: computed set

Verifying the water tank

Safety property: the water level between 5.25 and 8.25 meters.



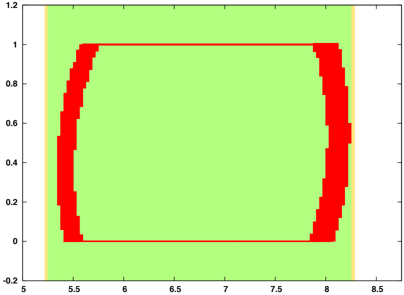
Second iteration:
grid $1/16 \times 1/160$
(x-axis: $x(t)$, y-axis: $\alpha(t)$).

Lower reach is not unsafe, refine grid.

Green: safe set Orange: ε-tolerance Red: computed set

Verifying the water tank

Safety property: the water level between 5.25 and 8.25 meters.



Third iteration: grid $1/32 \times 1/320$ (x-axis: $x(t)$, y-axis: $\alpha(t)$).

Outer reach is safe, system is proved safe.

Green: safe set Orange: ε-tolerance Red: computed set

Verifying the water tank

1. In this example, we could prove safety by outer reach.
2. Variations of the parameters could yield systems where lower reach would prove unsafety or where no conclusions could be drawn (smallest precision of the parameters reached without proving safety or unsafety).

Dominance checking

Definition

Given two components C_1 and C_2, with assumptions and guarantees (A_1, G_1) and (A_2, G_2), we say that C_1 dominates C_2 if and only if under weaker assumptions ($A_2 \subseteq A_1$), stronger promises are guaranteed ($G_1 \subseteq G_2$).

If this is the case, the component C_2 can be replaced with C_1 in the system without affecting the whole system behaviour.

Intuitively, the component C_1 dominates C_2 if it issues sharper outputs ($G_1 \subseteq G_2$) with looser inputs ($A_2 \subseteq A_1$), e.g., a dominating controller can issue a subset of the control commands to cope with an environment which is allowed more freedom.

Dominance checking by reachability analysis

1. Represent the two components by two hybrid automata H_1 and H_2 with inputs and outputs.
2. Assumptions A_1 and A_2 are represented by hybrid automata H_{A_1} and H_{A_2} that specify the possible inputs U_1, U_2 for the components.
3. Guarantees G_1 and G_2 specify the possible outputs Y_1, Y_2 of the automata H_1 and H_2.
4. H_1 dominates H_2 if and only if $G_1 \subseteq G_2$ and $A_2 \subseteq A_1$.

This is a reachability analysis problem:

$$Reach(H_{A_1} \| H_1)|_{Y_1} \subseteq Reach(H_{A_2} \| H_2)|_{Y_2}.$$

Dominance checking in ARIADNE

The approximate reachability routines of ARIADNE can be used to test dominance of components:

1. Compute an ε-lower approximation L_2^ε of $Reach(H_{A_2} \| H_2)|_{Y_2}$.
2. Remove a border of size ε from L_2^ε.
3. Compute an outer approximation O_1 of $Reach(H_{A_1} \| H_1)|_{Y_1}$.
4. If $O_1 \subseteq L_2^\varepsilon - \varepsilon$ then $Reach(H_{A_1} \| H_1)|_{Y_1} \subseteq Reach(H_{A_2} \| H_2)|_{Y_2}$ and thus H_1 dominates H_2.
5. If not, we cannot say anything about H_1 and H_2, and we retry with a finer approximation.

Dominance checking in Ariadne

The proof of correctness of the procedure relies on the following steps:

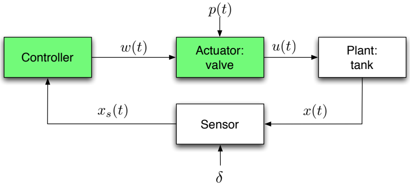
1. $Reach(H_{A_1} \| H_1)|_{Y_1} \subseteq O_1$ by definition.
2. $O_1 \subseteq L_2^{\varepsilon} - \varepsilon$ to be verified.
3. $L_2^{\varepsilon} - \varepsilon \subseteq Inner_2$ under suitable hypotheses.
4. $Inner_2 \subseteq Reach(H_{A_2} \| H_2)|_{Y_2}$ by definition.

Therefore $Reach(H_{A_1} \| H_1)|_{Y_1} \subseteq Reach(H_{A_2} \| H_2)|_{Y_2}$ and thus H_1 dominates H_2.

A sufficient hypothesis to guarantee that $L_2^{\varepsilon} - \varepsilon \subseteq Inner_2$ is that $Reach(H_{A_2} \| H_2)|_{Y_2}$ is a ε-regular set, i.e., there are no holes "smaller than ε" in the set.

The water tank again

We want to replace the controller and the valve.



- The valve is slower than the previous one
- The controller is smarter and can fix the valve aperture to any value $w(t) \in [0, 1]$

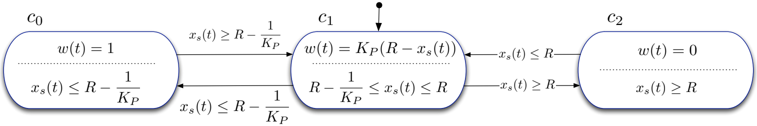
Does the system still operate correctly?

The water tank again

Application of dominance relation in this example:

1. The automaton H_1 represents the whole system with new components (proportional controller, slower valve, sensor, plant).
2. The automaton H_2 represents the whole system with old components (hysteresis controller, original faster valve, sensor, plant).
3. A_1 and A_2 specify the same external input $U_1 = U_2 = p(t)$, i.e. the pressure on the valve, so it is $A_2 = A_1$.
4. G_1 and G_2 specify the same output $Y_1 = Y_2 = x(t)$, i.e., the water level of the tank, for which it is requested $G_1 \subseteq G_2$.

A proportional controller



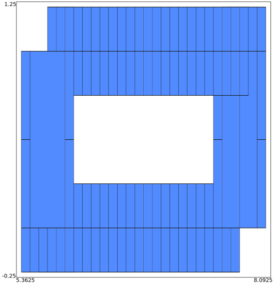
- The input is the measured water level $x_s(t)$ provided by the sensor.
- The output is a command signal $w(t) \in [0, 1]$ for the valve position regulation.
- The controller computes the output $w(t)$ from the measured level $x_s(t)$ and the water level reference R.
- In response to a command $w(t)$ the valve aperture $a(t)$ varies with the first-order linear dynamics $\dot{a}(t) = \frac{1}{\tau}(w(t) - a(t))$.

A proportional controller

1. Location c_0 models when the controller saturates the opening valve command to $w(t) = 1$.
2. Location c_1 models when the controller tracks the water reference level R.
3. Location c_2 models when the controller saturates the closing valve command to $w(t) = 0$.

Results

ε-lower approximation of the reachable set of the hysteresis controller:



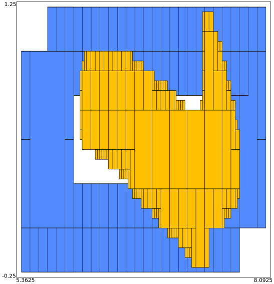
Assumptions:

- Inlet pressure p between 50 and 60 KPa (KiloPascal)
- Sensor's error between -0.05 and 0.05 m

The proportional controller dominates the hysteresis controller.

Results

Outer approximation of the reachable set of the proportional controller:



Assumptions:

- Inlet pressure p between 50 and 60 KPa (KiloPascal)
- Sensor's error between -0.05 and 0.05 m

The proportional controller dominates the hysteresis controller.

Parametric verification

A system can be partially specified

- environmental parameters outside the control of the designer and for which there may be imperfect knowledge
- design parameters that can be fixed by the designer, but whose admissible values are not necessarily known a priori

Ariadne allows parametric verification

- exhaustively check all possible values of the parameters
- determine the value for the design parameters for which the component respects the guarantees, for all possible values of the environmental parameters

Parametric dominance results

Obtained for different values of two parameters: the gain K_P and the reference height R of the proportional controller.

- Green: proportional dominates hysteretic for all points;
- Red: proportional does not dominate hysteretic in at least one point;
- Yellow: insufficient accuracy to obtain a result.