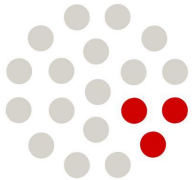


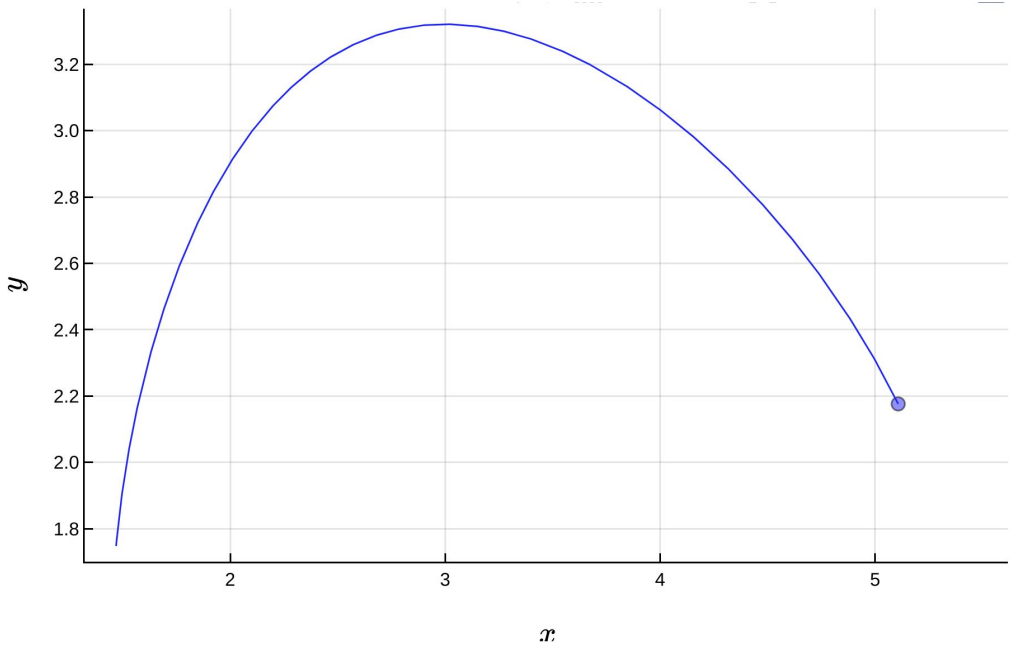
Verificación formal de sistemas controlados por reloj

Marcelo Forets, CURE, UdelaR
Daniel Freire, IFFC, UdelaR
Christian Schilling, IST Austria, Austria

arXiv: https://arxiv.org/abs/2006.12325

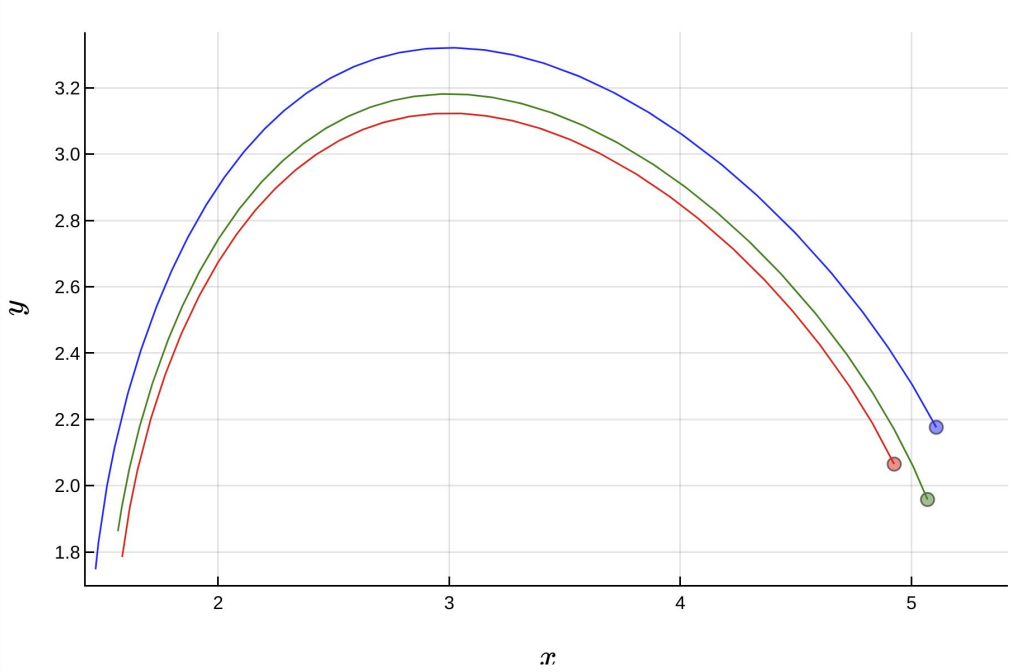
Parte 1:
¿Qué es Reachability Analysis?

El problema de verificación



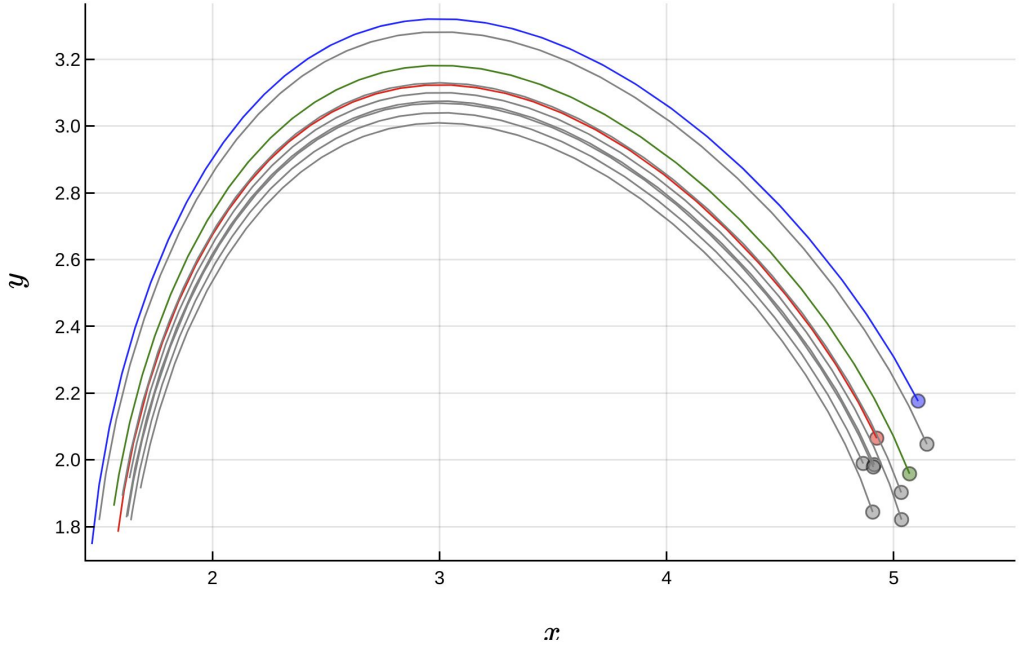
$$x'(t) = f(x(t), u(t), p(t))$$

El problema de verificación



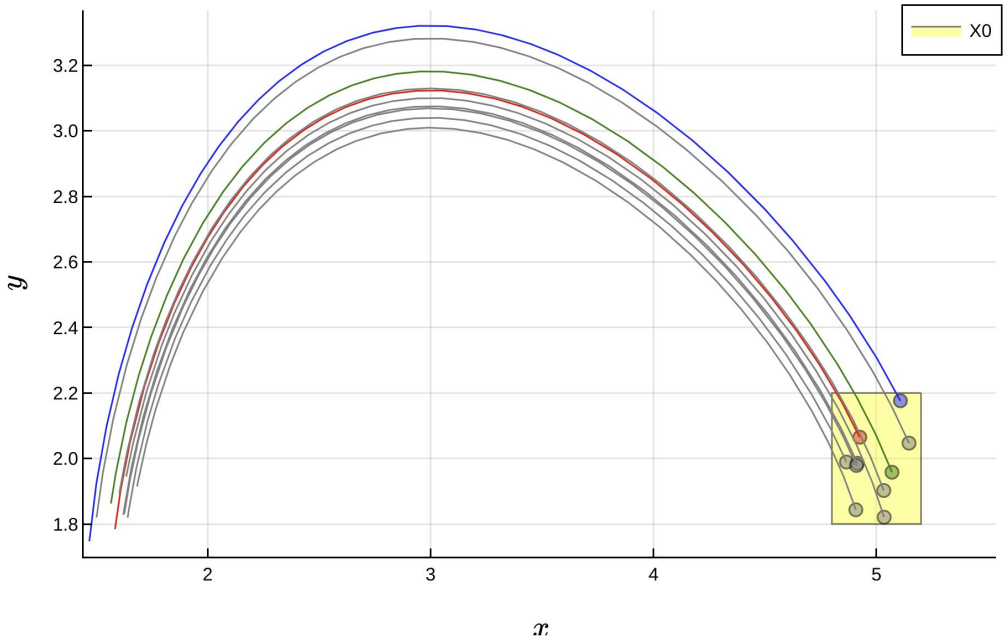
$$x'(t) = f(x(t), u(t), p(t))$$

El problema de verificación



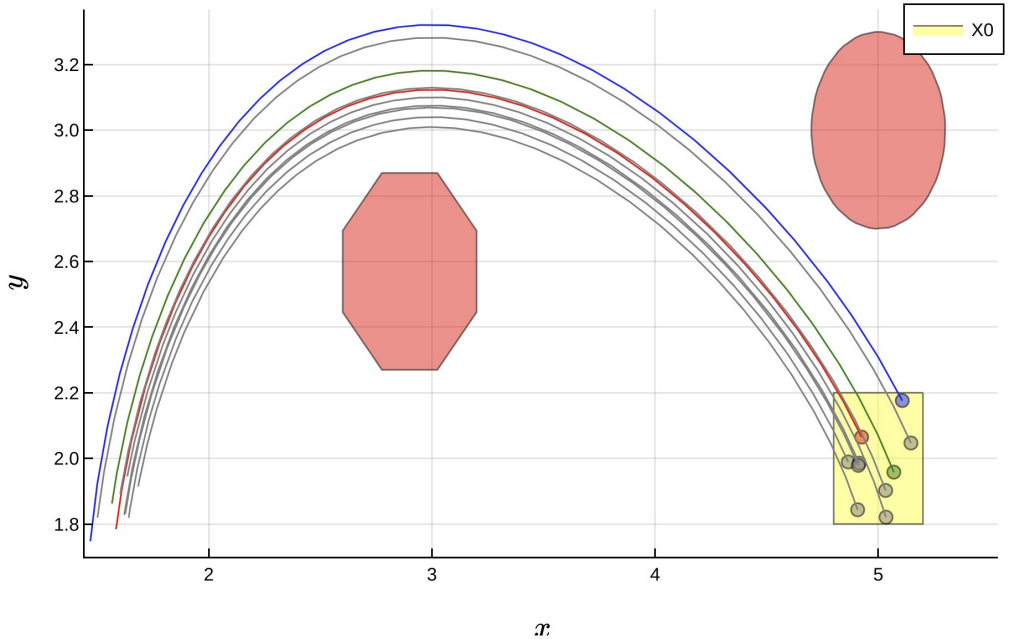
$$x'(t) = f(x(t), u(t), p(t))$$

El problema de verificación



$$x'(t) = f(x(t), u(t), p(t))$$

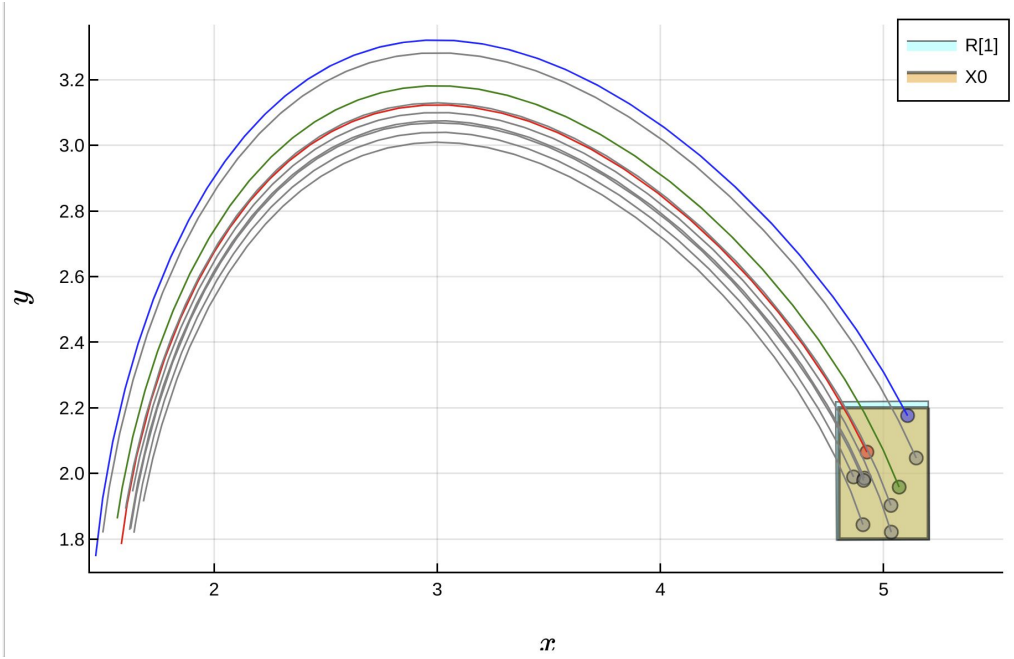
El problema de verificación



$$x'(t) = f(x(t), u(t), p(t))$$

Si el conjunto de estados alcanzables **no** se intersecta con ciertos estados “malos”, el sistema se dice seguro.

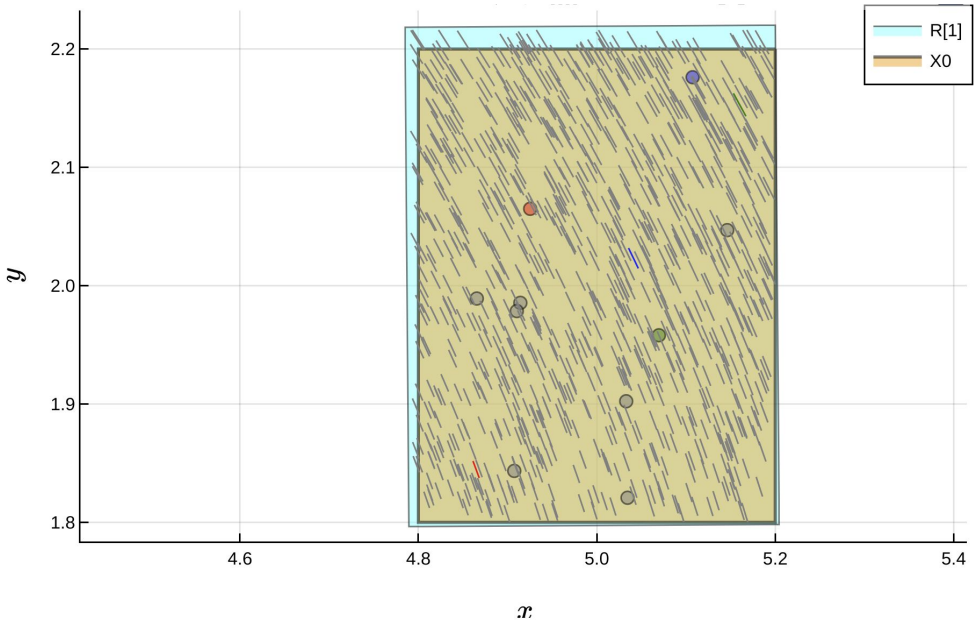
El problema de verificación



$$x'(t) = f(x(t), u(t), p(t))$$

El método de *reachability analysis* permite calcular R[1], que contiene todos los estados alcanzables para un intervalo de tiempo [0, dt], partiendo del conjunto inicial X0.

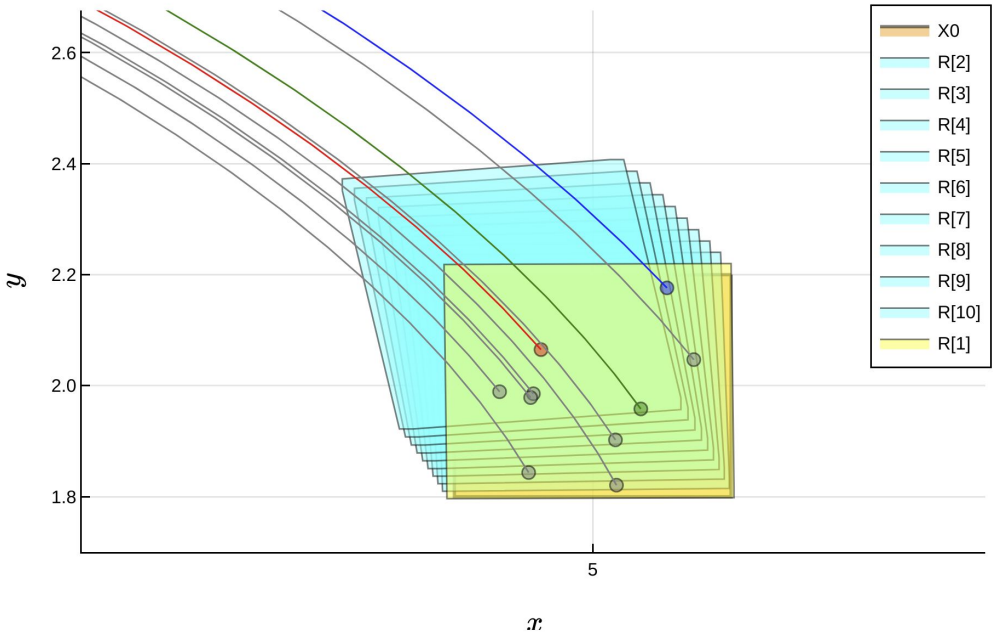
El problema de verificación



$$x'(t) = f(x(t), u(t), p(t))$$

El método de *reachability analysis* permite calcular R[1], que contiene todos los estados alcanzables para un intervalo de tiempo [0, dt], partiendo del conjunto inicial X0.

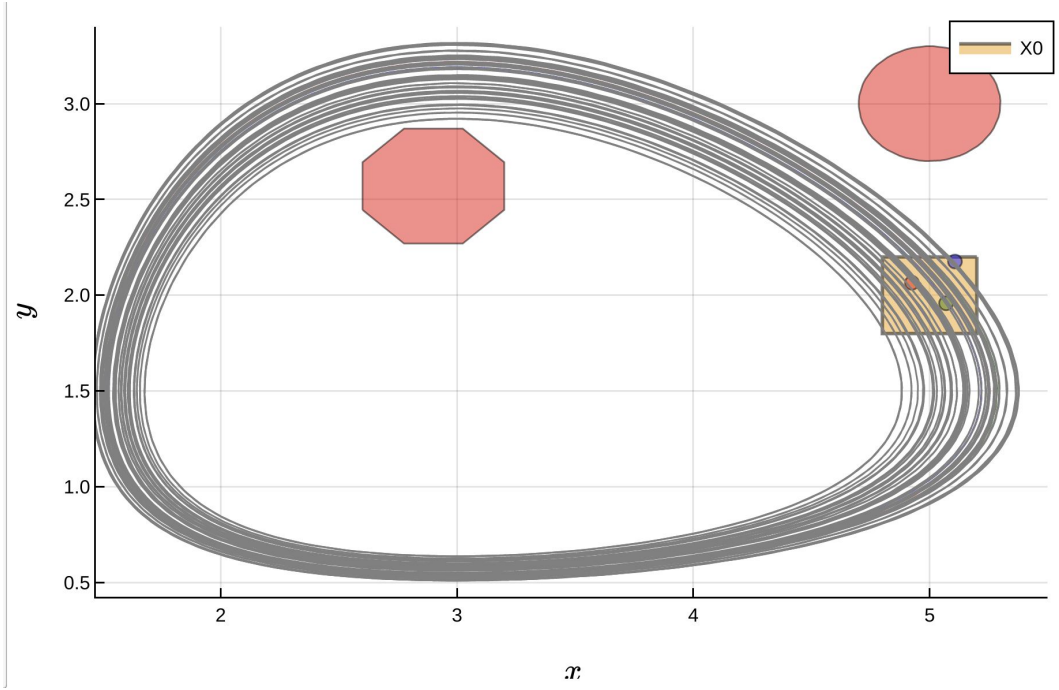
El problema de verificación



$$x'(t) = f(x(t), u(t), p(t))$$

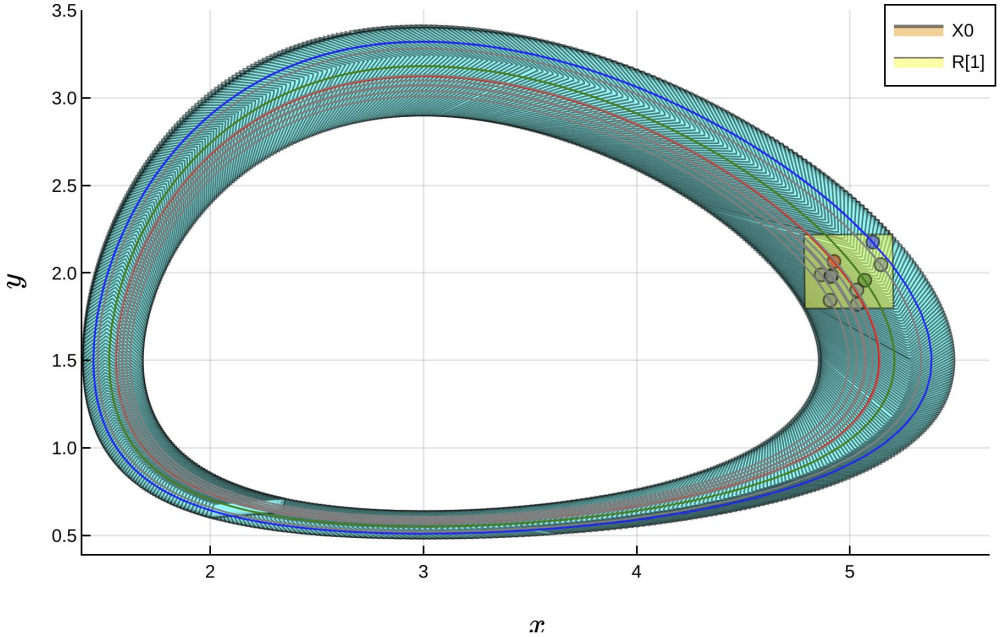
Cada *reach-set* R[i] contiene los estados alcanzables para el intervalo de tiempo [dti, dti'].

El problema de verificación



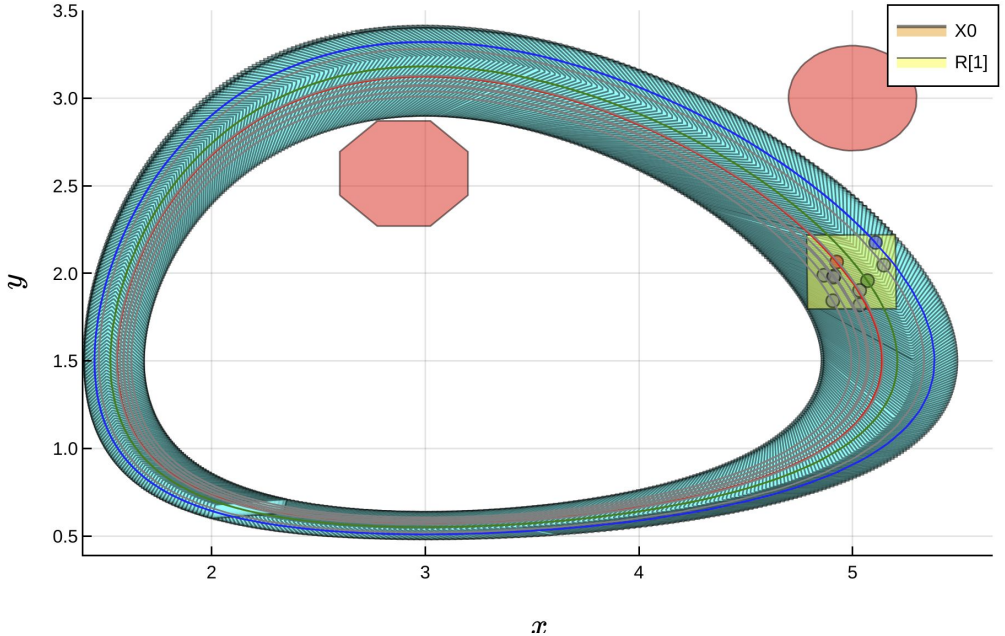
$$x'(t) = f(x(t), u(t), p(t))$$

El problema de verificación



$$x'(t) = f(x(t), u(t), p(t))$$

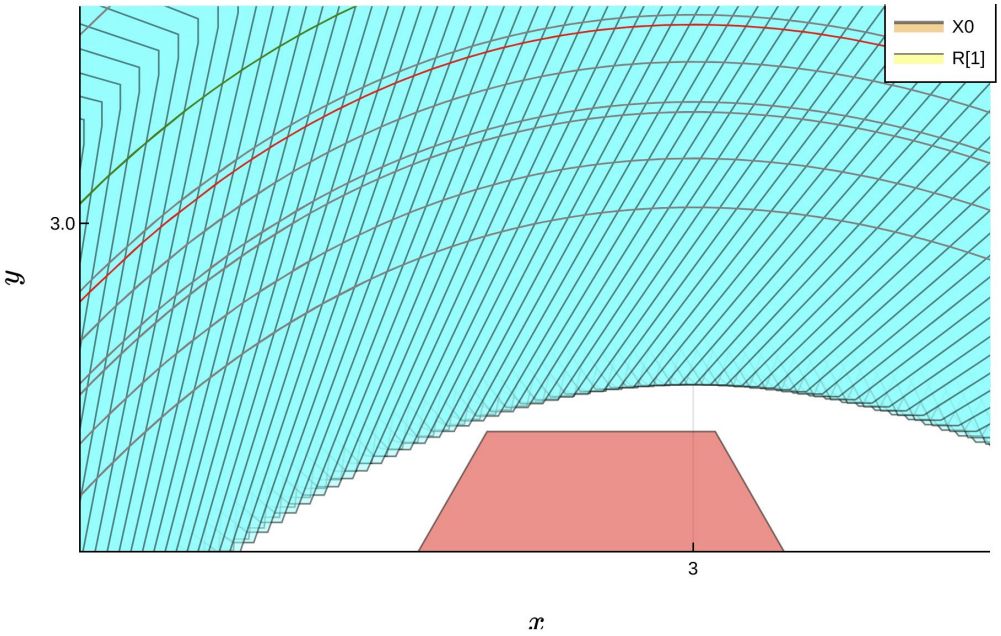
El problema de verificación



$$x'(t) = f(x(t), u(t), p(t))$$

Si el conjunto de estados alcanzables **no** se intersecta con los estados “malos”, el sistema se dice seguro.

El problema de verificación



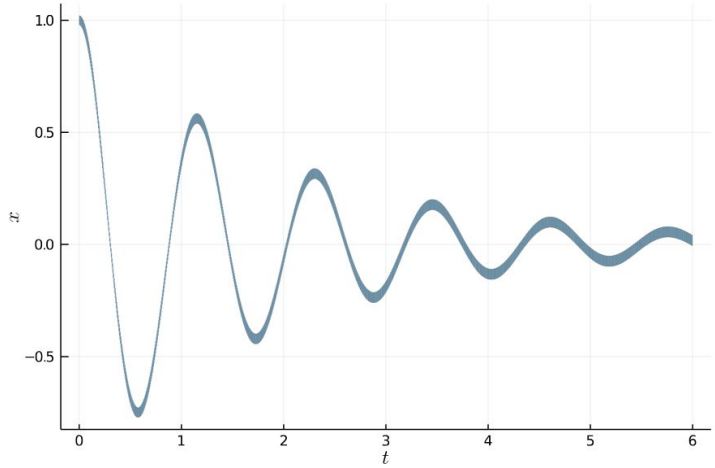
$$x'(t) = f(x(t), u(t), p(t))$$

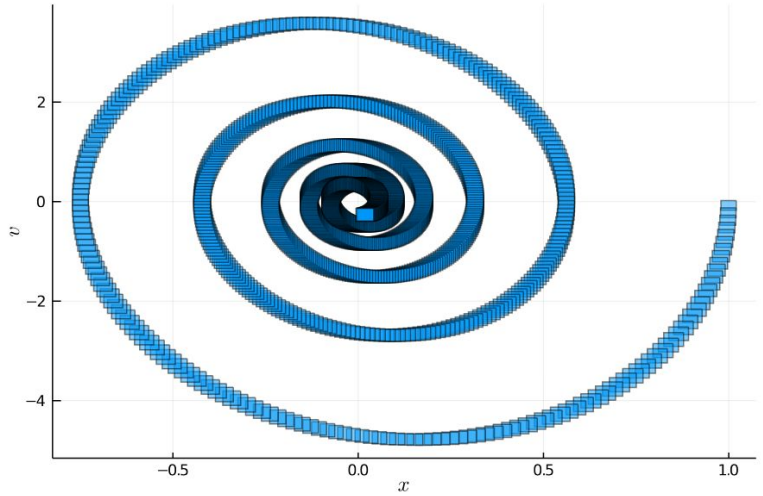
El método de *reachability analysis* es exhaustivo (cubre todos los comportamientos admitidos) y riguroso (en sentido matemático, incluyendo robustez numérica respecto a errores de punto flotante).

Trayectorias, reach-sets y flowpipes

$$\dot{x}(t) = Ax(t) + Bu(t), x(0) \in \mathcal{X}_0, u(t) \in \mathcal{U}(t)$$

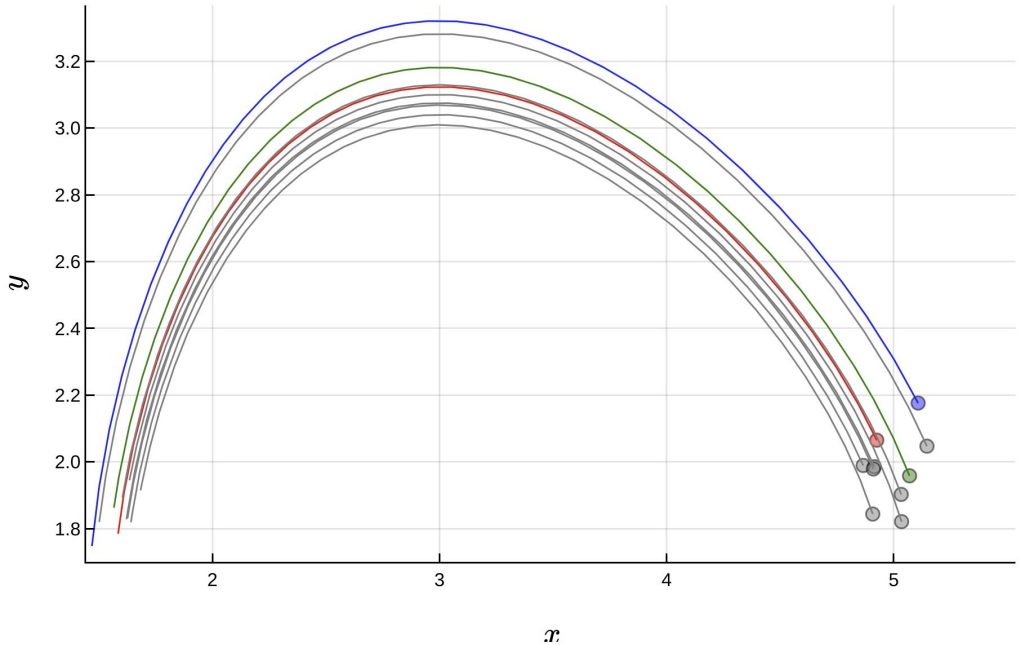
$$x \in \mathbb{R}^n, u \in \mathbb{R}^m, A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times m}, \mathcal{X}_0 \subset \mathbb{R}^n, \mathcal{U}(t) \subset \mathbb{R}^m$$





Trayectorias, reach-sets y flowpipes

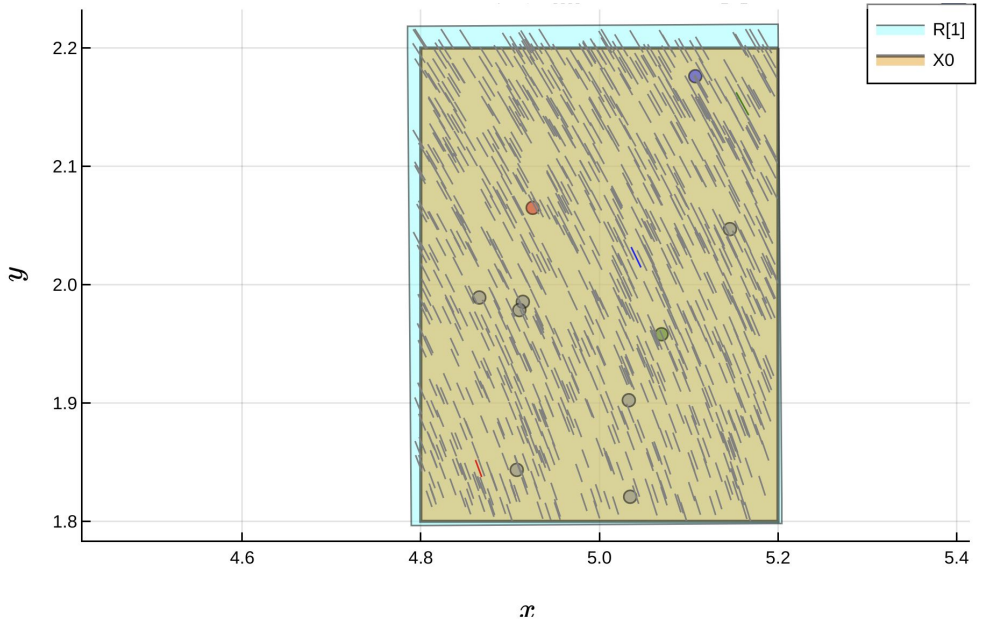
$$\dot{x}(t) = Ax(t) + Bu(t), x(0) \in \mathcal{X}_0, u(t) \in \mathcal{U}(t)$$



$$\mathbf{x}_{\mathbf{x}_0,\mathbf{u}}(t) = e^{At}\mathbf{x}_0 + \int_0^t e^{A(t-s)}B\mathbf{u}(s)\,ds.$$

Trayectorias, reach-sets y flowpipes

$$\mathcal{R}^e(\mathcal{X}_0, \mathcal{U}, t) := \bigcup_{\mathbf{u} \in \mathcal{U}} \left\{ \mathbf{x}_{\mathbf{x}_0, \mathbf{u}}(t) : \mathbf{x}_0 \in \mathcal{X}_0, \mathbf{u}(s) \in \mathcal{U}(s) \ \forall \ s \in [0, t] \right\}$$



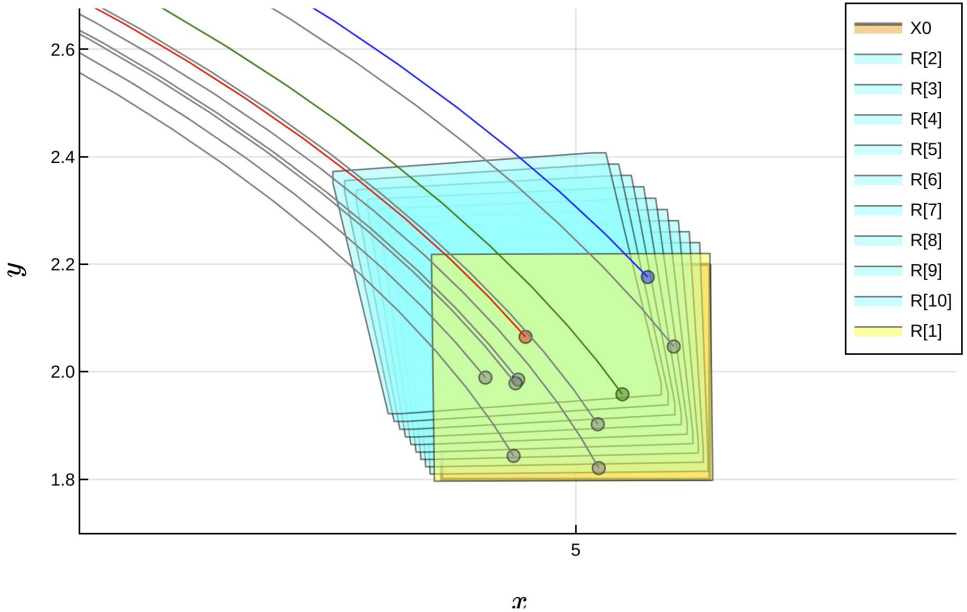
$$\mathcal{R}^e(\mathcal{X}_0, \mathcal{U}, t) \subseteq$$

$$\mathcal{R}(\mathcal{X}_0, \mathcal{U}, [t_1, t_2]) \subset \mathbb{R}^n$$

$$t \in [t_1, t_2]$$

Trayectorias, reach-sets y flowpipes

$$\mathcal{F}^e(\mathcal{X}_0, \mathcal{U}, [t_1, t_2]) := \bigcup_{s \in [t_1, t_2]} \mathcal{R}^e(\mathcal{X}_0, \mathcal{U}, s).$$

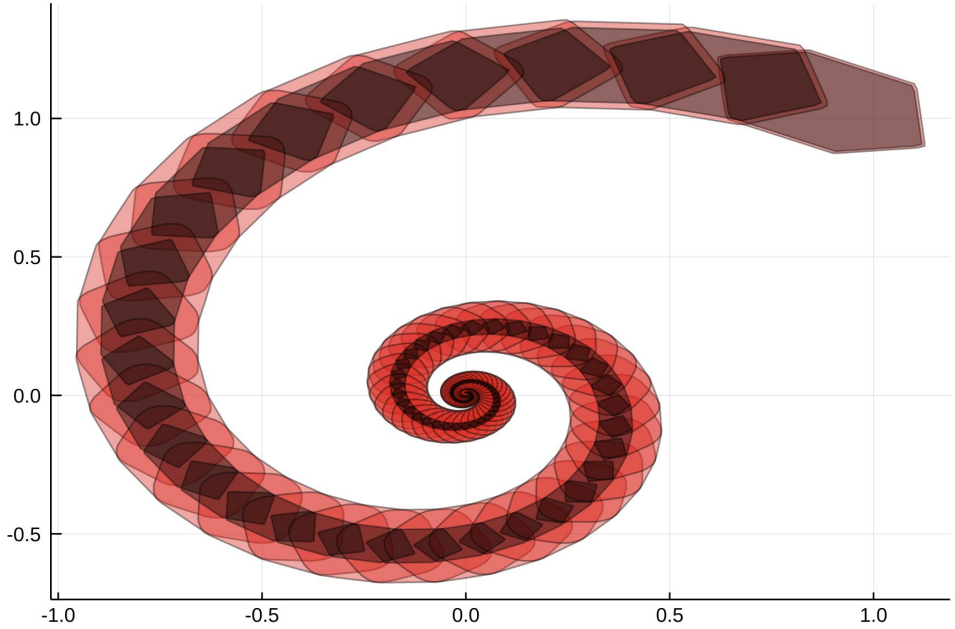


$$\mathcal{F}^e(\mathcal{X}_0, \mathcal{U}, [t_1, t_2]) \subseteq$$

$$\mathcal{F}(\mathcal{X}_0, \mathcal{U}, [t_1, t_2])$$

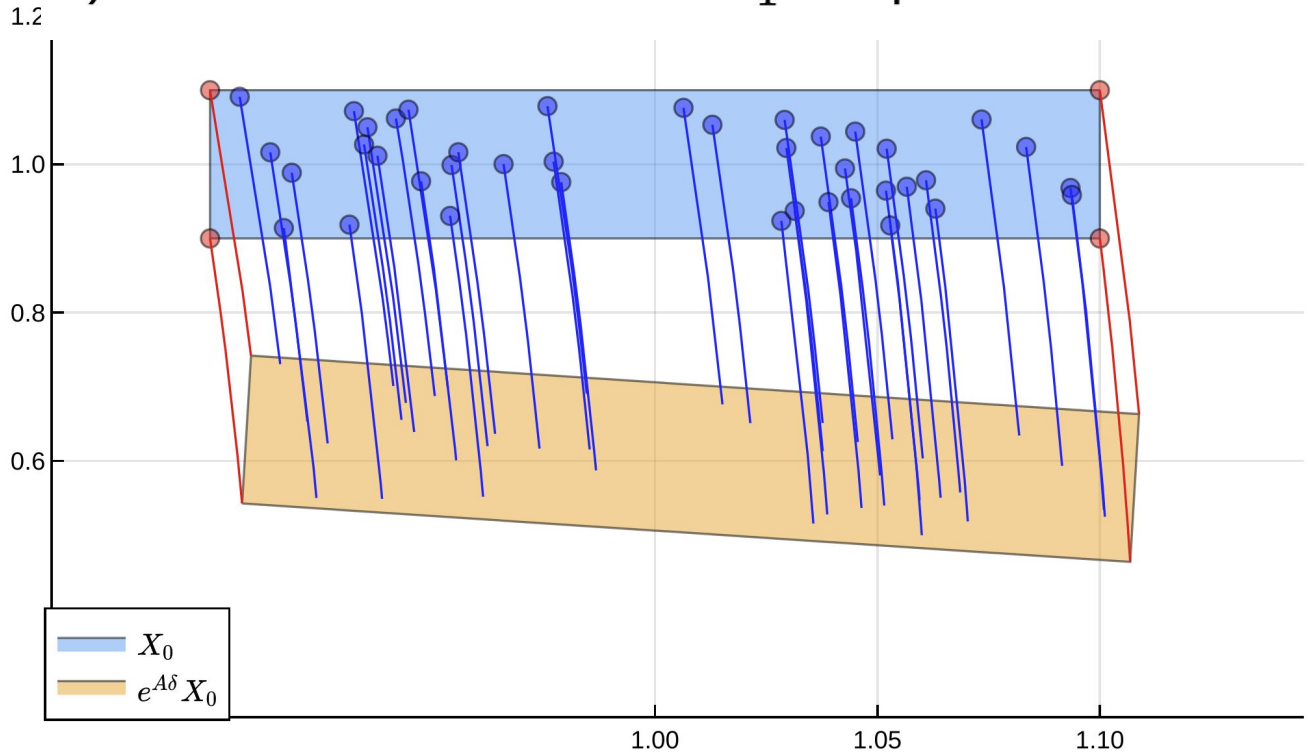
Cómo se calcula un flowpipe

$$\dot{x}(t) = Ax(t) + Bu(t), x(0) \in \mathcal{X}_0, u(t) \in \mathcal{U}(t)$$



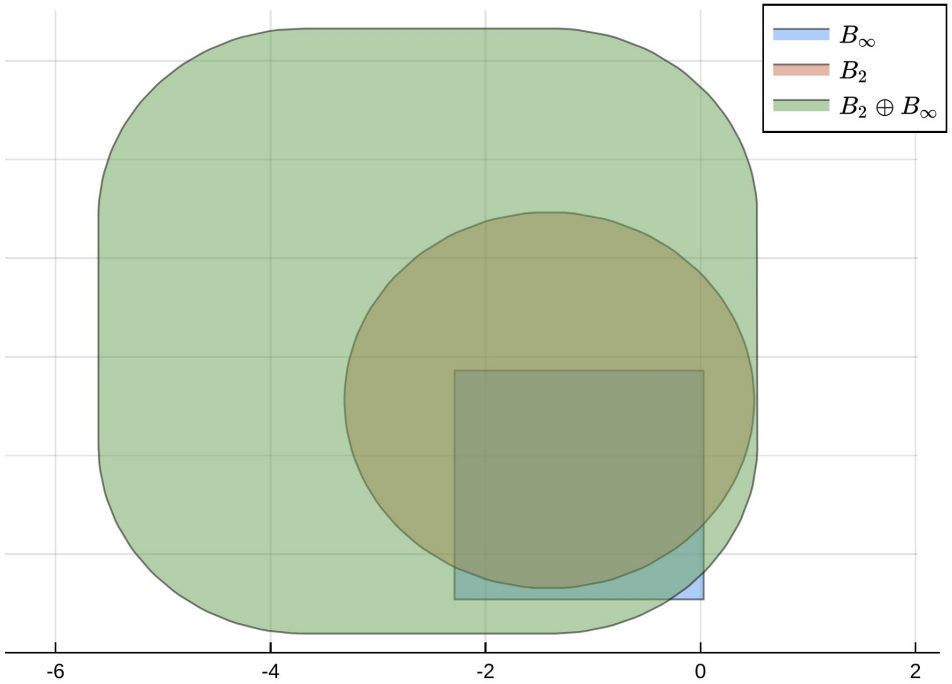
Cómo se calcula un flowpipe

1) *Discretización:* Calcular R_1 tal que contiene a $\mathcal{F}^e(\mathcal{X}_0, \mathcal{U}, [0, \delta]) \subseteq R_1$.



Cómo se calcula un flowpipe

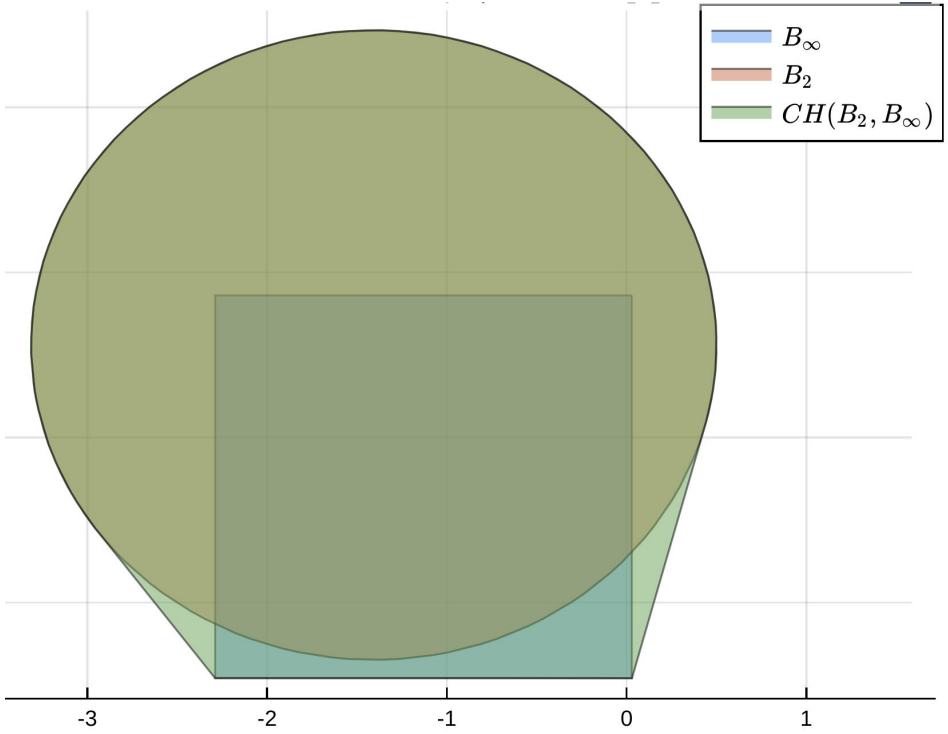
Precisamos operar con conjuntos..



Suma de Minkowski

Cómo se calcula un flowpipe

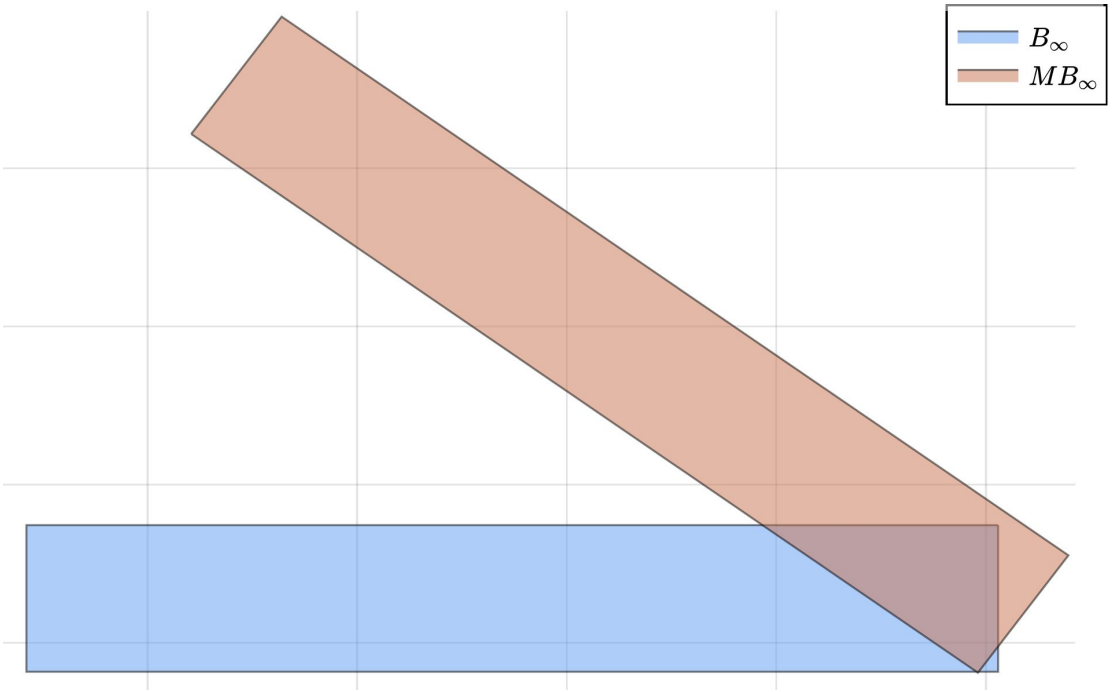
Precisamos operar con conjuntos..



Envolvente convexa

Cómo se calcula un flowpipe

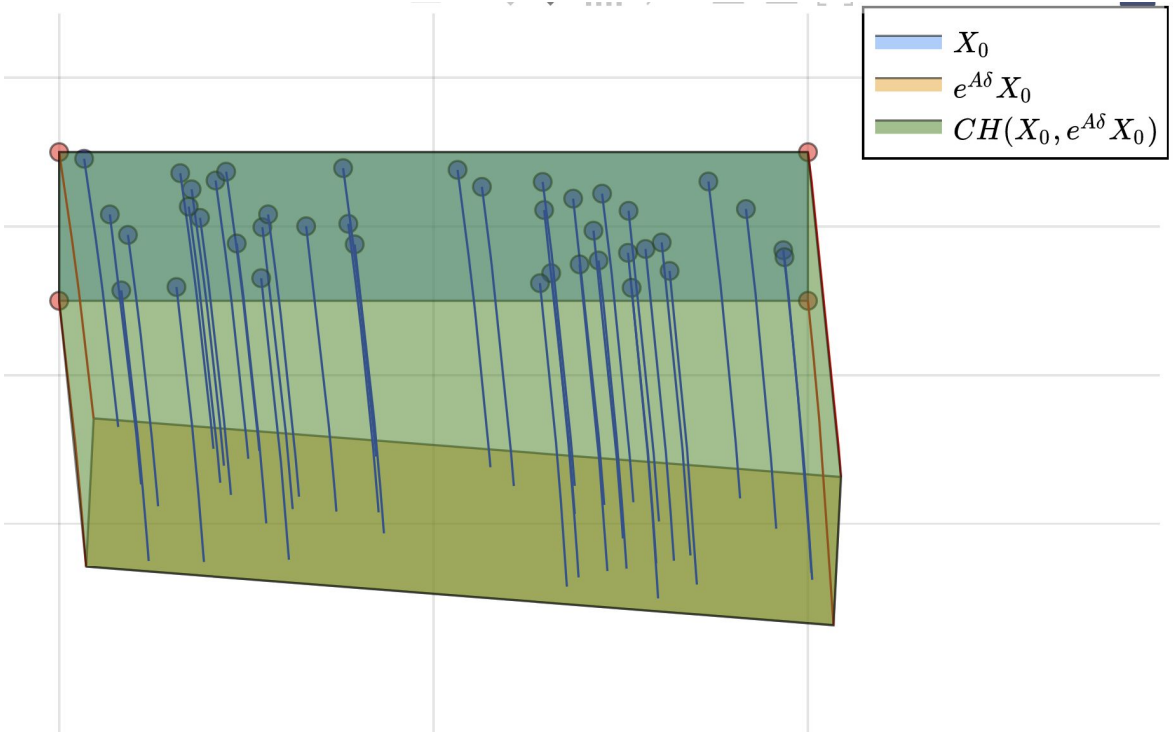
Precisamos operar con conjuntos..



Transformación lineal

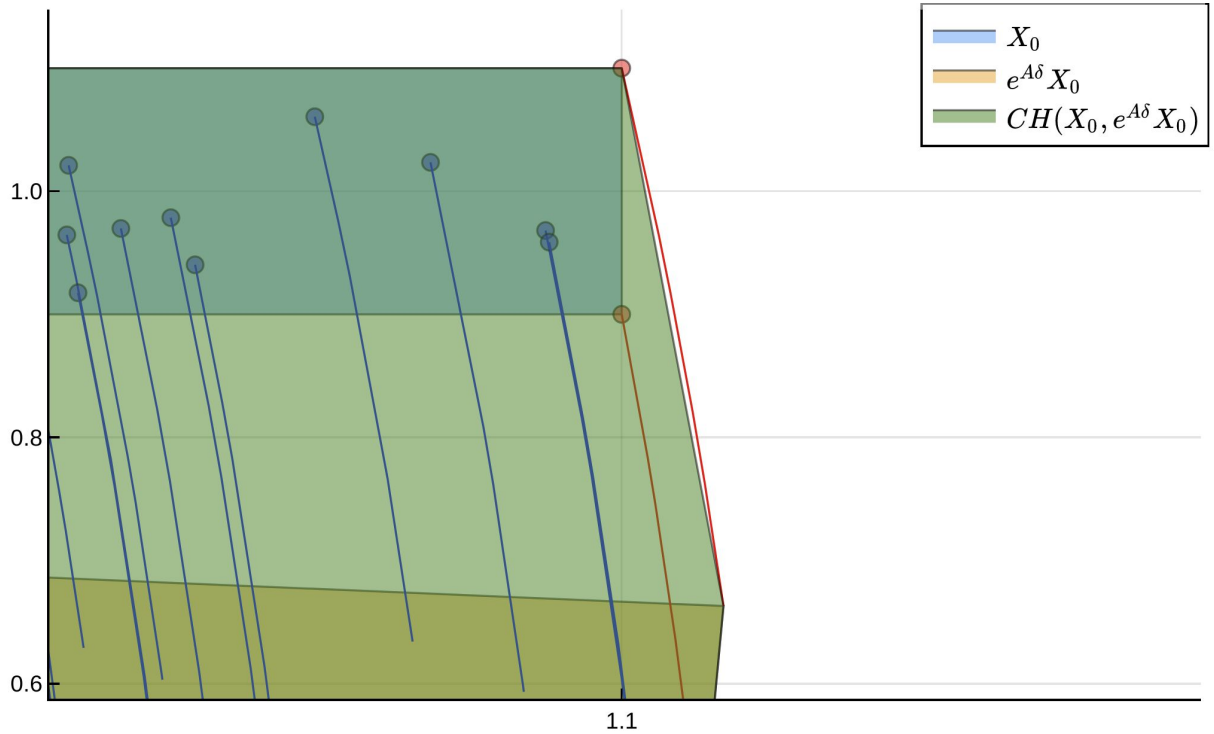
Cómo se calcula un flowpipe

1) *Discretización:* Calcular R_1 tal que contiene a $\mathcal{F}^e(\mathcal{X}_0, \mathcal{U}, [0, \delta]) \subseteq R_1$.



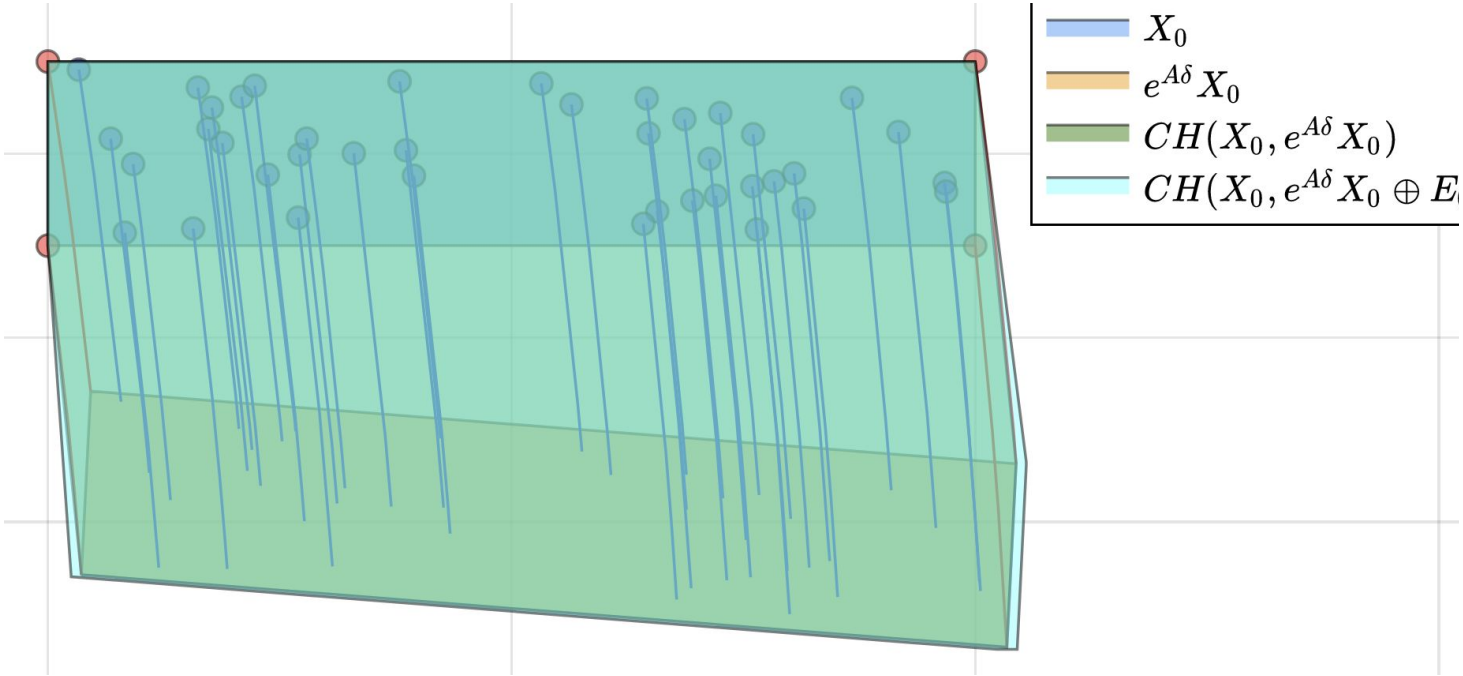
Cómo se calcula un flowpipe

1) *Discretización:* Calcular R_1 tal que contiene a $\mathcal{F}^e(\mathcal{X}_0, \mathcal{U}, [0, \delta]) \subseteq R_1$.



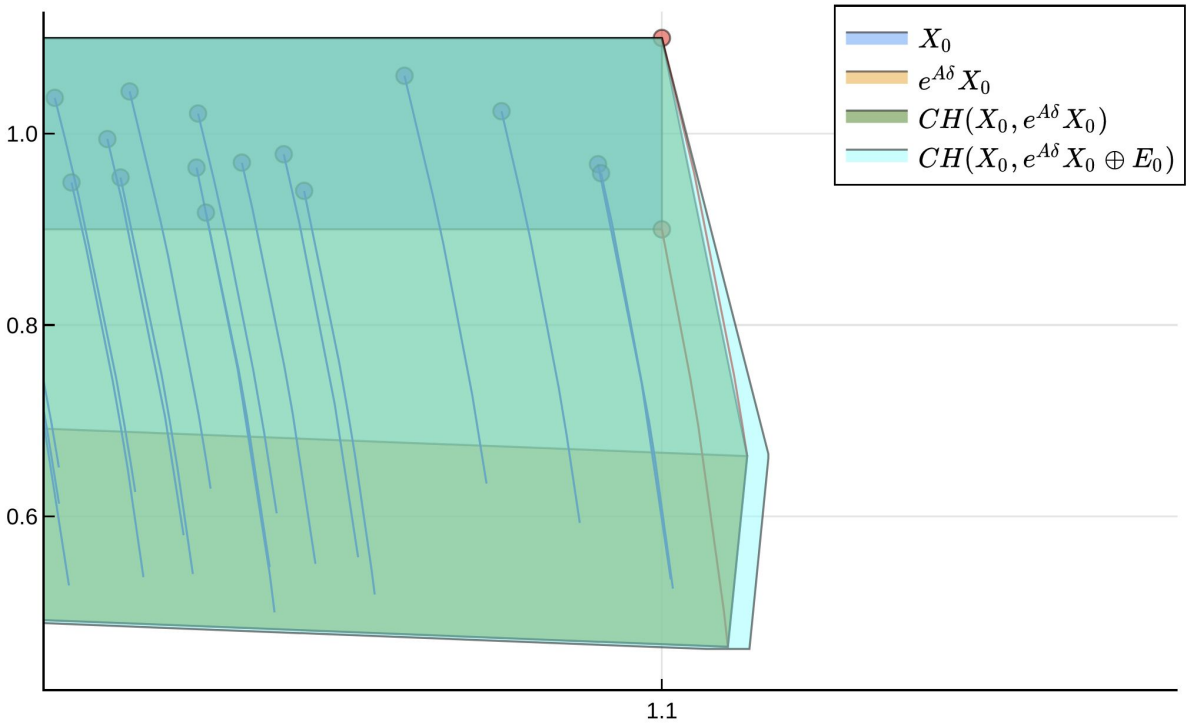
Cómo se calcula un flowpipe

1) *Discretización:* Calcular R_1 tal que contiene a $\mathcal{F}^e(\mathcal{X}_0, \mathcal{U}, [0, \delta]) \subseteq R_1$.



Cómo se calcula un flowpipe

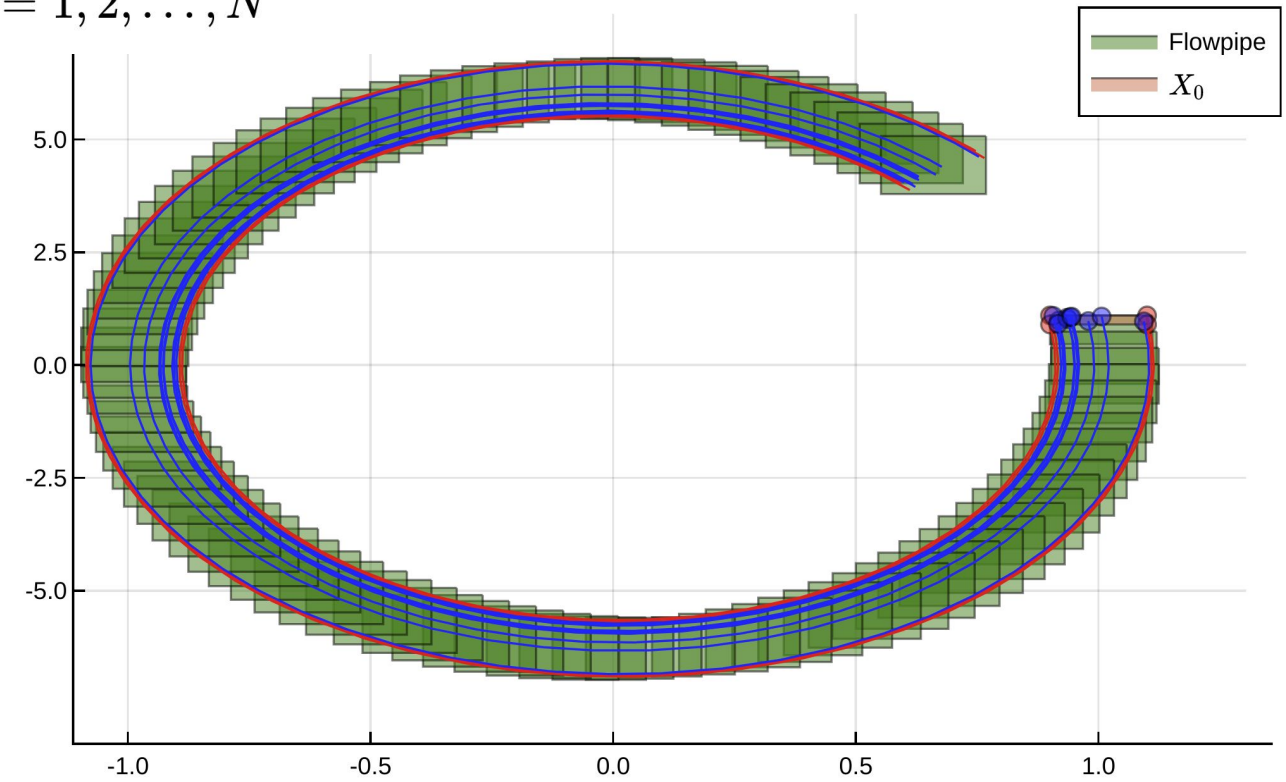
1) *Discretización:* Calcular R_1 tal que contiene a $\mathcal{F}^e(\mathcal{X}_0, \mathcal{U}, [0, \delta]) \subseteq R_1$.



Cómo se calcula un flowpipe

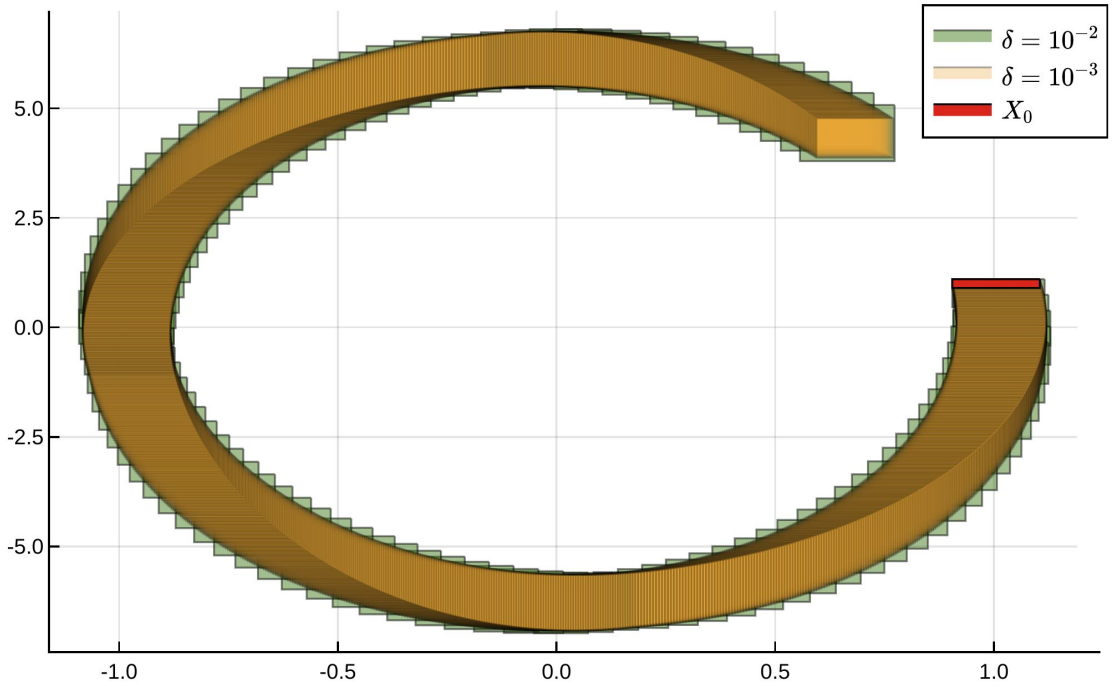
2) *Integración con conjuntos:* Resolver la recurrencia

$$R_{k+1} = \Phi R_k \oplus V_k, \qquad k = 1, 2, \ldots, N$$



Cómo se calcula un flowpipe

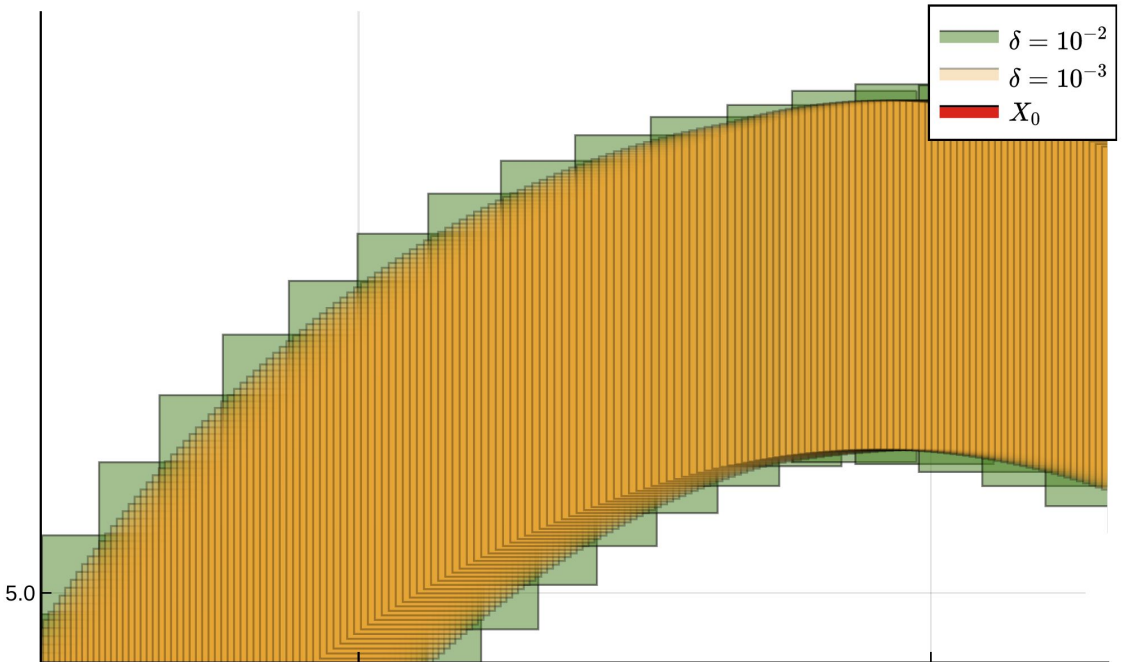
3) *Solución:* Devolver el flowpipe $F := \{R_1, R_2, \ldots, R_N\}$.



Se sabe que esta sucesión converge a la solución exacta para $\delta \to 0$.

Cómo se calcula un flowpipe

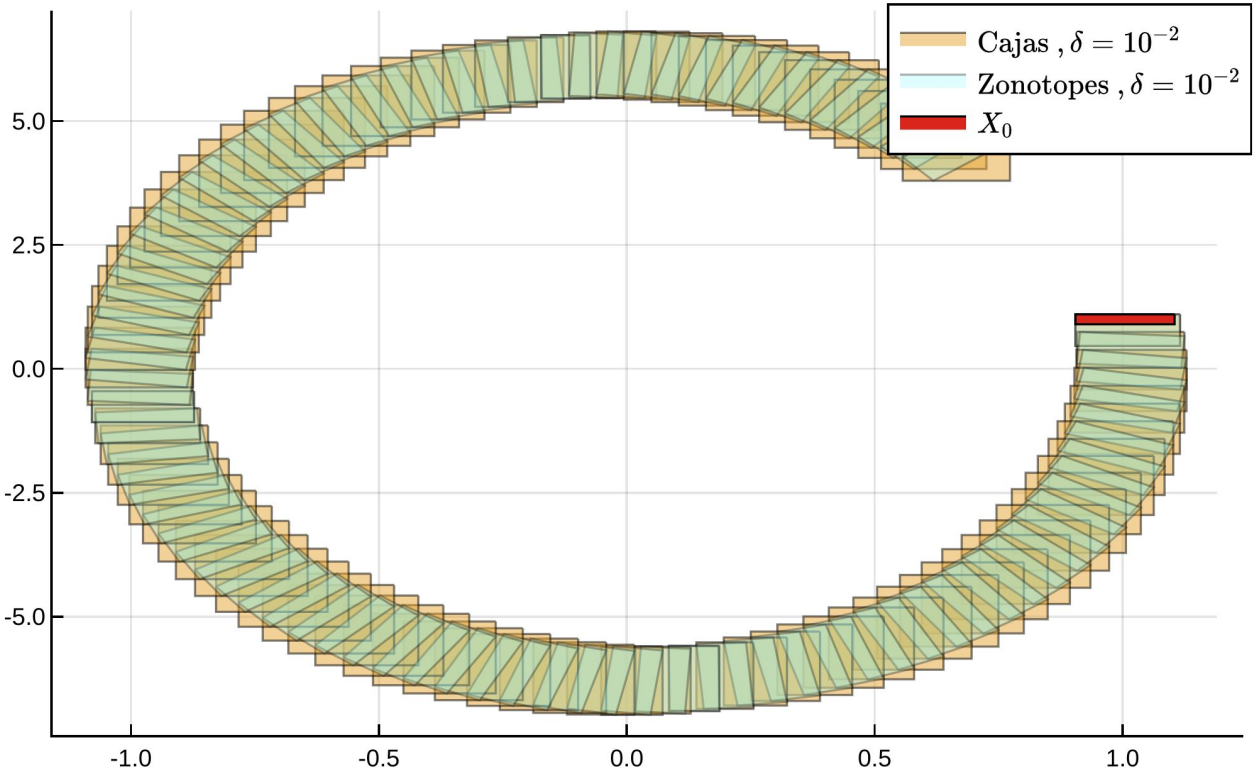
3) *Solución:* Devolver el flowpipe $F := \{R_1, R_2, \ldots, R_N\}$.



Se sabe que esta sucesión converge a la solución exacta para $\delta \to 0$.

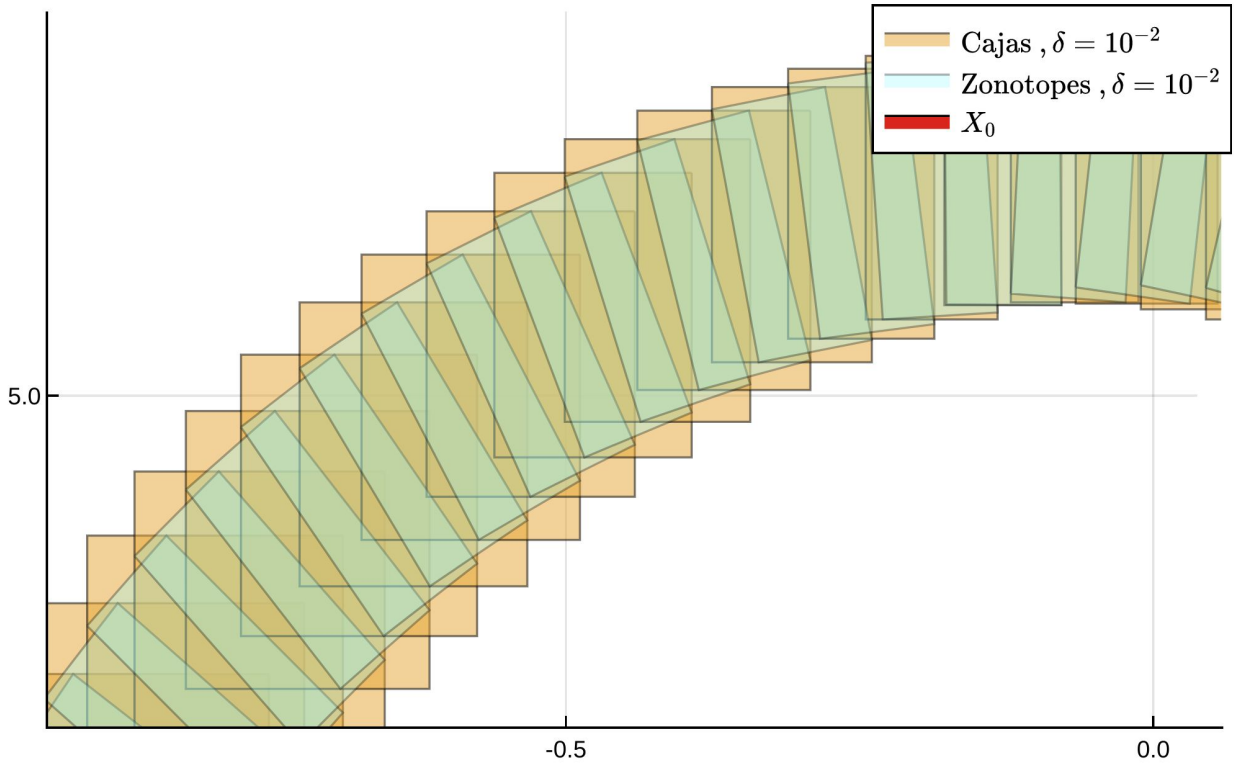
Cómo se calcula un flowpipe

3) *Solución:* Devolver el flowpipe $F := \{R_1, R_2, \ldots, R_N\}$.



Cómo se calcula un flowpipe

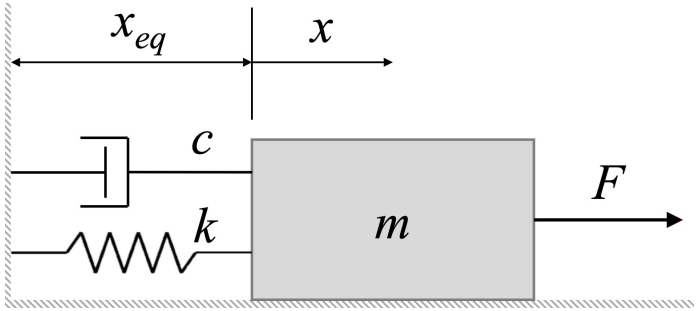
3) *Solución:* Devolver el flowpipe $F := \{R_1, R_2, \ldots, R_N\}$.



Ejemplo: oscilador armónico forzado

$$m\ddot{x} = -kx - c\dot{x} + F(t)$$

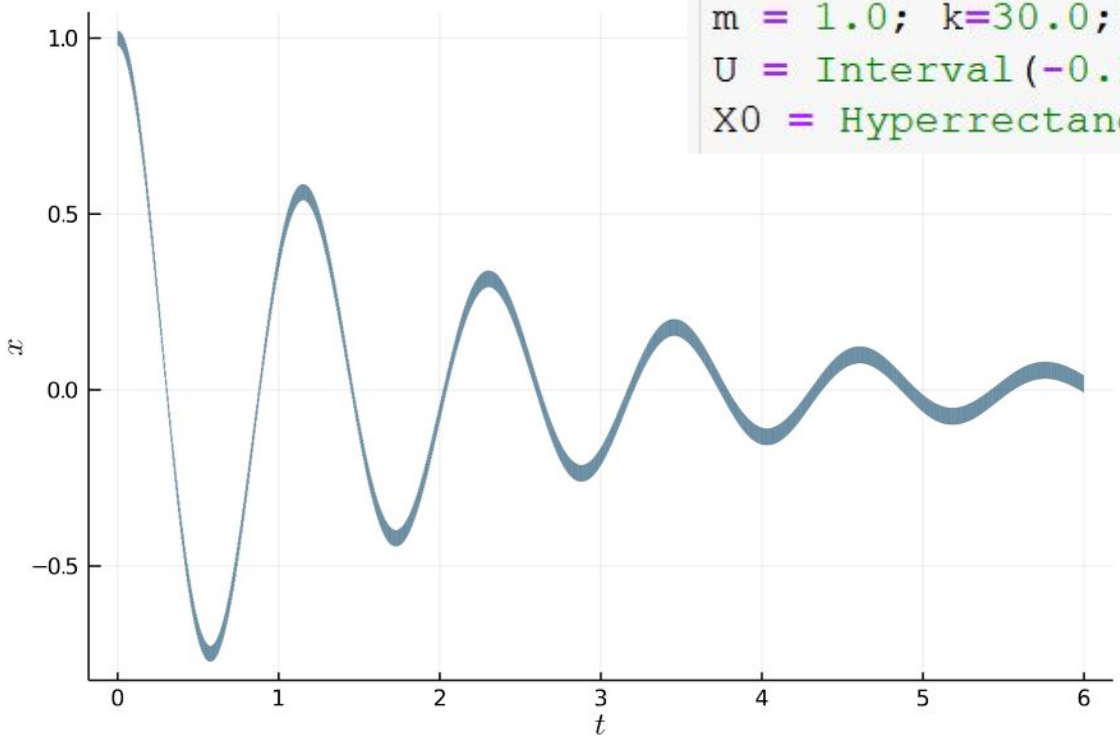
$$(x(0), \dot{x}(0)) \in X_0$$



$$\frac{d}{dt}\begin{bmatrix} x \\ v \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -c/m \end{bmatrix}\begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ F(t)/m \end{bmatrix}$$

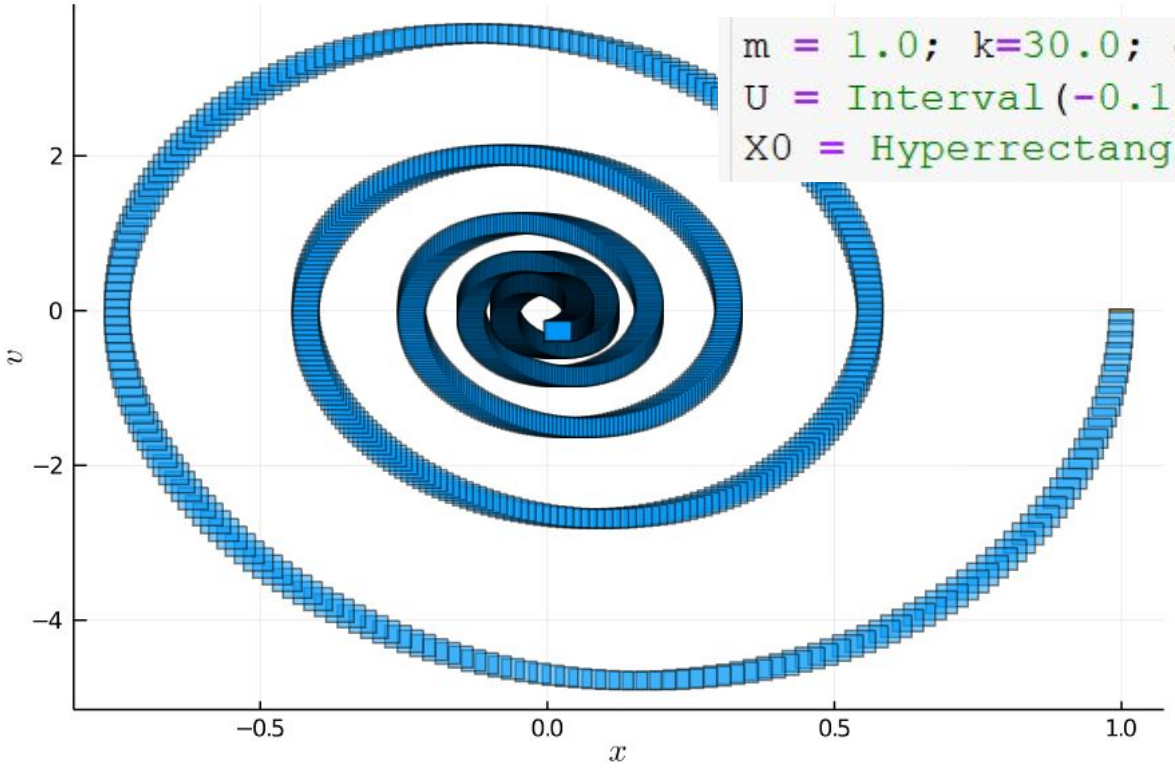
$$\mathbf{x}_{\mathbf{x}_0,\mathbf{u}}(t) = e^{At}\mathbf{x}_0 + \int_0^t e^{A(t-s)}B\mathbf{u}(s)\,ds.$$

Ejemplo: oscilador armónico forzado



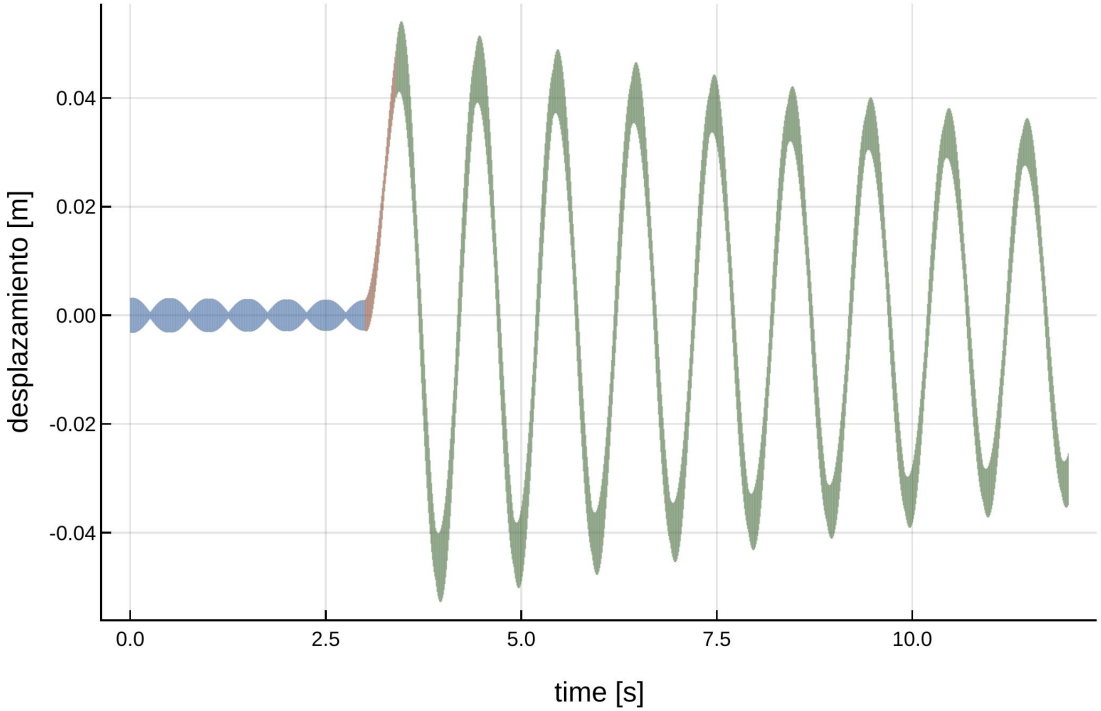

```
m = 1.0; k=30.0; c=1.0
U = Interval(-0.1, +0.1)
X0 = Hyperrectangle([1.0, 0.0],[0.02, 0.02])
```

Ejemplo: oscilador armónico forzado




```
m = 1.0; k=30.0; c=1.0
U = Interval(-0.1, +0.1)
X0 = Hyperrectangle([1.0, 0.0],[0.02, 0.02])
```

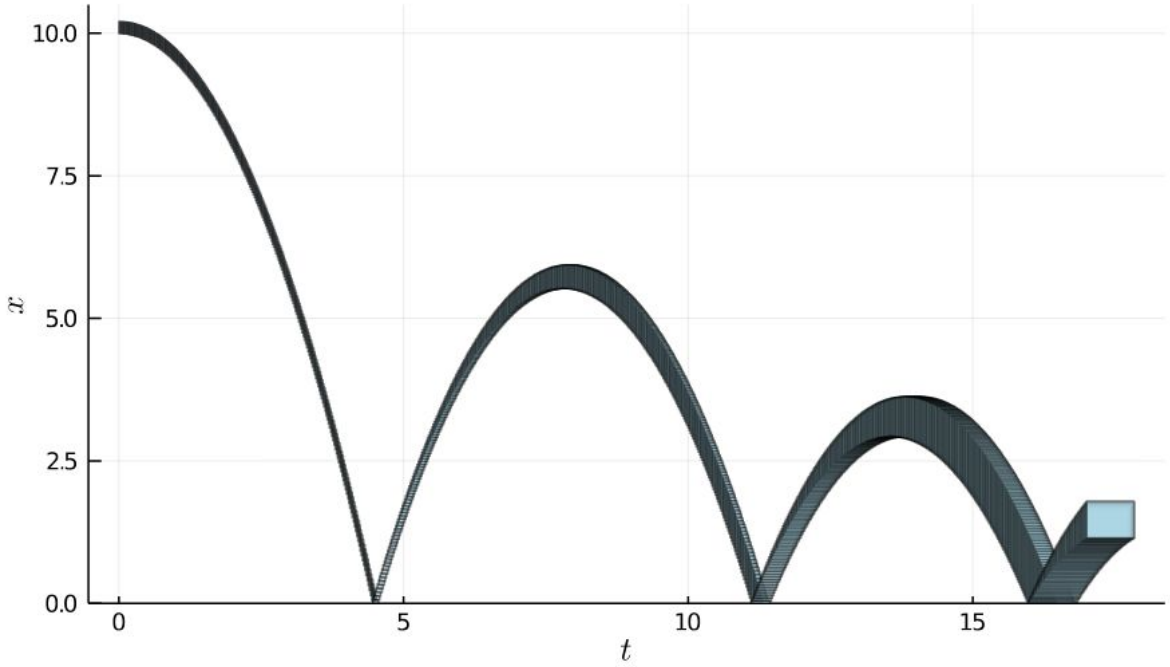
Ejemplo: oscilador armónico forzado



También se pueden modelar cambios en la dinámica...

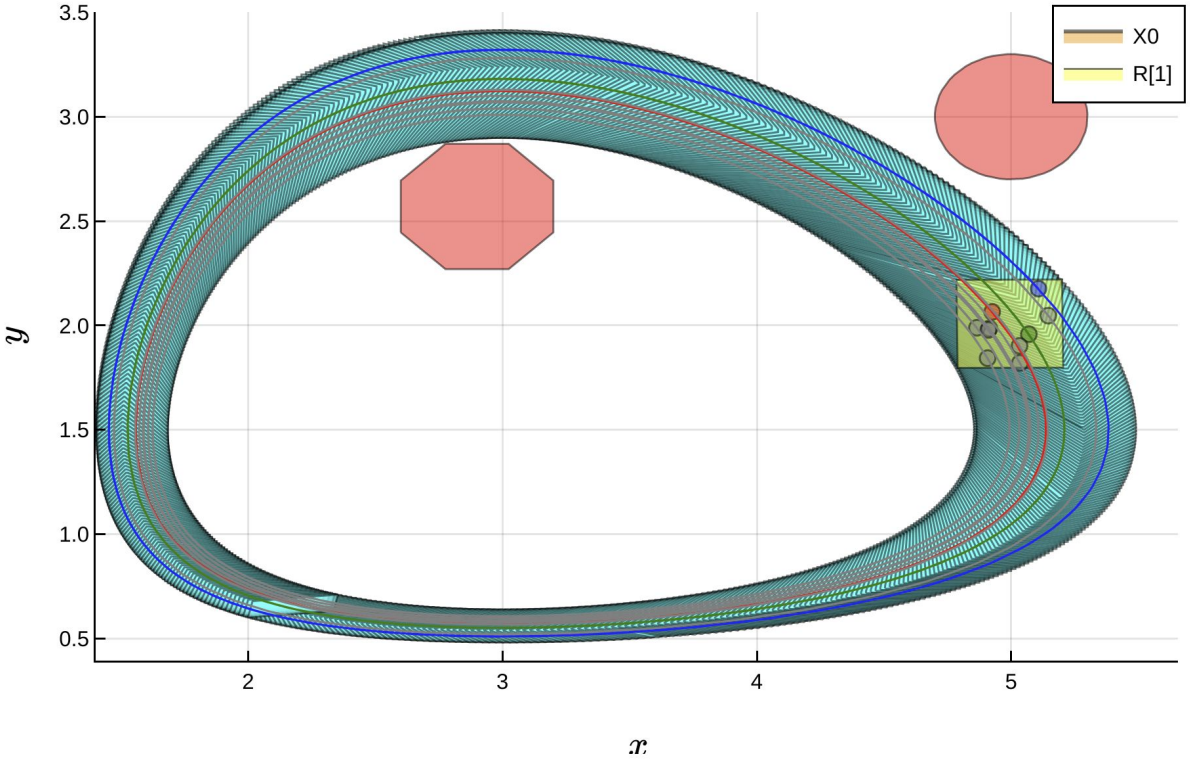
Esto da lugar a los sistemas híbridos (Parte 2 de la charla)

Reachability para sistemas híbridos



Eg. “bouncing ball”

Reachability para sistemas no-lineales



Eg. Lotka-Volterra usando modelos de Taylor

Próxima parte: Aplicación a un freno EM

- Código disponible: **JuliaReach**
- Eficiente para sistemas lineales con miles utilizando métodos de descomposición.
- Diversas aplicaciones: control robusto, redes neuronales, vehículos autónomos, verificación de circuitos electrónicos, estabilidad, etc.

JuliaReach

Reachability Computations for Dynamical Systems in Julia

http://juliareach.com/

- *Planeamos ofrecer un curso de grado / posgrado sobre métodos formales y aplicaciones en Ciencias e Ingeniería.*

¡Gracias!
¿Preguntas?

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

CURE Centro Universitario Regional del Este

FACULTAD DE CIENCIAS UDELAR fcien.edu.uy

FACULTAD DE INGENIERIA

IST AUSTRIA Institute of Science and Technology

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Massachusetts Institute of Technology

Continuará...