

# Reachability computation using set integration

Thao Dang

VERIMAG, CNRS (France)

# Plan

1. Reachability analysis using set integration
2. Reachability analysis of linear systems
3. Reachability analysis of polynomial systems

# Introduction

## Reachable set computation for differential equations

- often requires symbolic computation (to handle non-determinism)

We propose to use ‘set integration’ which combines ‘traditional’ numerical integration schemes with set computations

- Numerical integration can be applied to general differential equations and provides efficient error control mechanisms
- Convenient to approximate single solutions, but we need to characterize sets of all possible solutions

# Reachability analysis

## Direct methods

- **Track the evolution** of the reachable set under the flow of the system.  
Various **set representations**: *e.g.* polyhedra, zonotopes, ellipsoids, level sets
- **Exact** results, or **accurate approximations** with error bounds. Using **symbolic** or **numerical** computations
- **Tools**: Coho, CheckMate, d/dt, HysDel, VeriShift, Vertdict, Requiem, HJ-solver toolbox, ..

## Indirect methods

- **Abstraction methods**: reducing to a simpler system that preserves the property (*e.g.* [Tiwari & Khanna 02; Alur et al. 02; Clarke et al. 03])
- Achieve a proof of the property without computing the reachable set:  
*e.g.* **Barrier certificates** [Prajna & Jadbabaie04], **polynomial invariants** [Tiwari & Khanna 04].

★ **Scalability** is still challenging (complexity and size of **real-life systems**)

# Reachable set computation using numerical integration

$$\dot{\mathbf{x}}(t) = g(\mathbf{x}(t)); \quad \mathbf{x}(0) = \mathbf{x}_0$$

## Main idea

- A typical numerical scheme:

$$\mathbf{x}_{k+1} = \mathcal{Y}_k(g, h, \mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_k)$$

where  $h$  is the step size.

- Set integration: computing such schemes with sets, that is  $\mathbf{x}_k$  is **not a single point** in  $\mathbb{R}^n$  **but a subset** of  $\mathbb{R}^n$

In this presentation, we apply this idea to linear and polynomial differential equations.

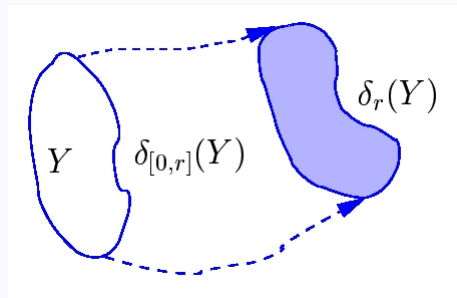
These methods were implemented in the reachability analysis tool **d/dt**.

# Reachability operators

Continuous system  $\dot{\mathbf{x}} = f(\mathbf{x})$  where  $\mathbf{x} \in \mathcal{X}$ ;  $f : \mathcal{X} \rightarrow \mathbb{R}^n$  continuous vector field. Let  $\phi_{\mathbf{x}}(t)$  be the solution of the diff eq with  $\mathbf{x}$  as initial condition.

Given a time interval  $I$  and a set of states  $Y$ , **successor operator**  $\delta_I(Y) = \{\mathbf{y} \mid \exists \mathbf{x} \in Y \exists t \in I \mathbf{y} = \phi_{\mathbf{x}}(t)\}$ .

The **reachable set** from  $Y$  is  $\delta(Y) = \delta_{[0, \infty)}(Y)$  (all states reachable after any non-negative amount of time).

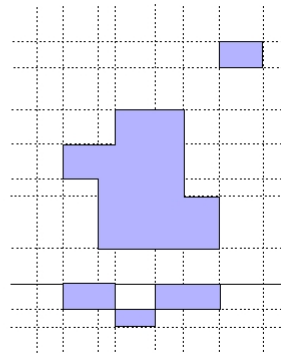
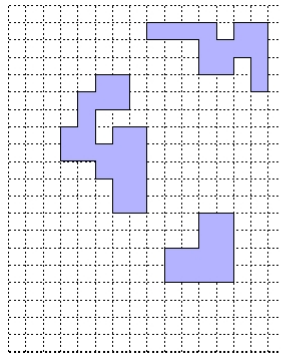


# Polyhedral approximation

To represent reachable sets, we use **convex polyhedra** and **orthogonal polyhedra** (unions of closed full-dimensional hyper-rectangles)

## Orthogonal polyhedra

- **Canonical representation**  $\Rightarrow$  effective computations of Boolean operations, equivalence and emptiness checking, membership testing, and other geometric operations (face detection, etc.).
- Appropriate for **over-** and **under-approximations** of **non-convex** sets



# Polynomial differential equations

We consider a polynomial differential equation:

$$\dot{\mathbf{x}}(t) = g(\mathbf{x}(t)) = A\mathbf{x}(t) + f(\mathbf{x}(t))$$

$f$  is the non-linear part

Considering  $f(\mathbf{x}(t))$  as independent input, we can write:

$$\mathbf{x}_{k+1} = e^{Ah}\mathbf{x}_k + \int_0^h e^{A(h-\tau)} f(\mathbf{x}(t_k + \tau)) d\tau.$$

Approximate  $\mathbf{x}(t_k + \tau)$  by  $\alpha(t_k + \tau) = \mathbf{x}_k + g(\mathbf{x}_k)\tau$ . Then,

$$\bar{\mathbf{x}}_{k+1} = e^{Ah}\mathbf{x}_k + \int_0^h e^{A(h-\tau)} g(\alpha(t_k + \tau)) d\tau = e^{Ah}\mathbf{x}_k + Q(\mathbf{x}_k)$$

The map  $Q(\mathbf{x}_k)$  is a polynomial in  $\mathbf{x}_k$ . We shall compute this map using Bézier techniques

## Second order approximation scheme



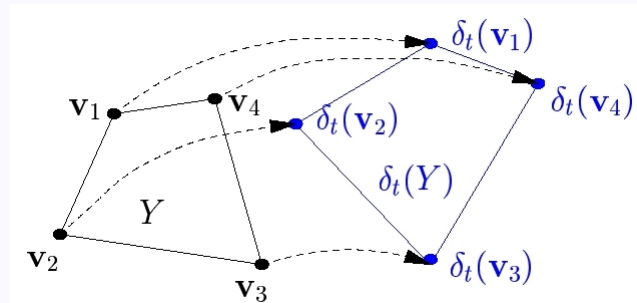
# Plan

1. Reachability analysis using set integration
2. Reachability analysis of linear systems
3. Reachability analysis of polynomial systems

# Reachability analysis of linear systems

A continuous linear system  $\dot{\mathbf{x}} = A\mathbf{x}$ . Initial set  $Y$  is a convex bounded polyhedron  $Y = \text{conv}(V)$  where  $V = \{\mathbf{v}_1, \dots, \mathbf{v}_m\}$  is a finite set of vertices

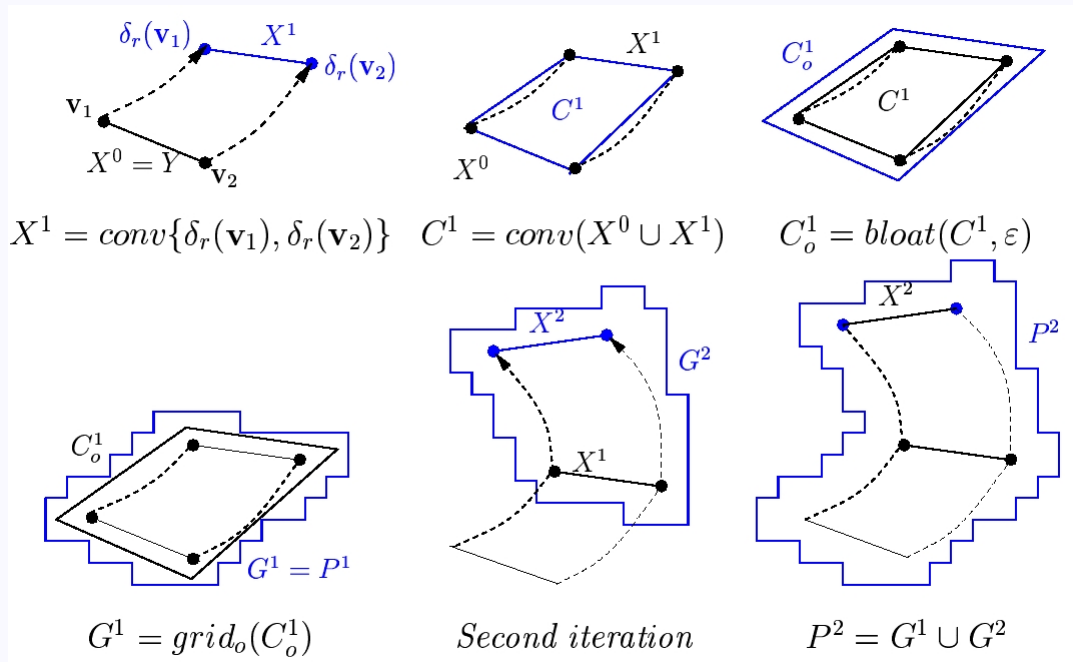
- Reachable set at time  $r$   $\delta_t(Y) = \text{conv}\{\delta_t(\mathbf{v}_1), \dots, \delta_t(\mathbf{v}_m)\}$ , and the successor of a point  $\mathbf{v}$  is  $\delta_t(\mathbf{v}) = e^{At}\mathbf{v}$



- Reachable set during time interval  $[0, r]$ ,

**Lemma:** Given a time step  $r \geq 0$ , there exists  $\varepsilon = \mathcal{O}(r^2)$  such that  $\delta_{[0,r]}(Y) \subseteq \text{conv}(Y \cup \delta_r(Y)) \oplus \varepsilon B$  ( $\varepsilon$ -neighborhood of the convex hull of  $Y$  and  $\delta_r(Y)$ ).

# Reachability analysis of linear systems (cont'd)

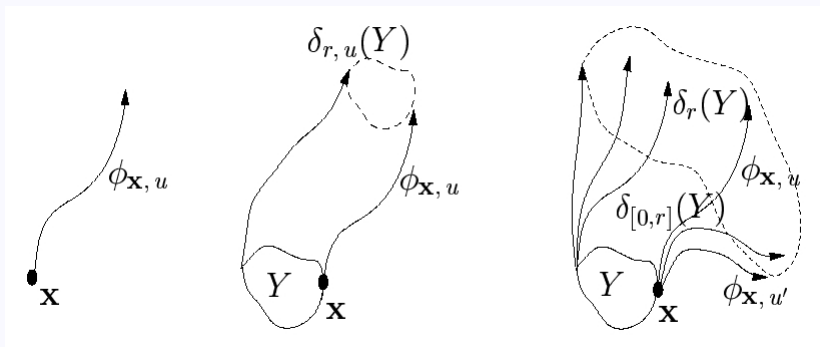


No accumulation of error, approximation error is of order  $\mathcal{O}(r^2)$ .

# Linear systems with uncertain input (cont'd)

Computing reachable set  $\delta_t(Y)$  at time  $r$  using the Maximal Principle

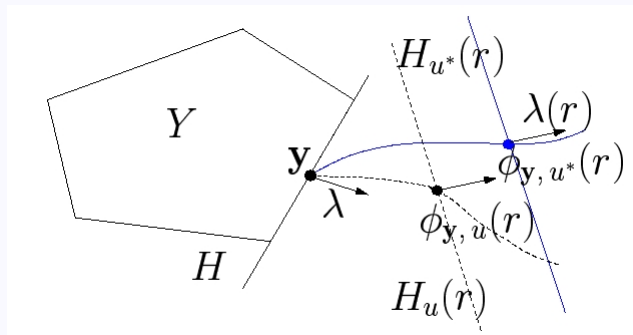
- The initial polyhedron can be written as intersection of half-spaces. Each half-space  $H = \{\mathbf{x} \mid \langle \lambda, \mathbf{x} \rangle \leq \langle \lambda, \mathbf{y} \rangle\}$ ;  $\lambda$ : normal vector,  $\mathbf{y}$ : supporting point
- For every half-space  $H$ , there exists an input  $u^*$  s.t. calculating its successors under  $u^*$  is sufficient to derive a *tight polyhedral approximation* of  $\delta_t(Y)$ .
- Evolution of normal vector  $\dot{\lambda}(t) = -A^T \lambda(t)$  (adjoint system) independent of input,  $u^*(r) \in \arg \max \{\langle \lambda(r), \mathbf{u} \rangle \mid \mathbf{u} \in U\}$ .



# Linear systems with uncertain input (cont'd)

Computing reachable set  $\delta_t(Y)$  at time  $r$  using the Maximal Principle

- The initial polyhedron can be written as intersection of half-spaces. Each half-space  $H = \{\mathbf{x} \mid \langle \lambda, \mathbf{x} \rangle \leq \langle \lambda, \mathbf{y} \rangle\}$ ;  $\lambda$ : normal vector,  $\mathbf{y}$ : supporting point
- For every half-space  $H$ , there exists an input  $u^*$  s.t. calculating its successors under  $u^*$  is sufficient to derive a *tight polyhedral approximation* of  $\delta_t(Y)$ .
- Evolution of normal vector  $\dot{\lambda}(t) = -A^T \lambda(t)$  (adjoint system) independent of input,  $u^*(r) \in \arg \max \{\langle \lambda(r), \mathbf{u} \rangle \mid \mathbf{u} \in U\}$ .



# Plan

1. Reachability analysis using set integration
2. Reachability analysis of linear systems
3. Reachability analysis of polynomial systems

# Definitions

A **multi-index**  $\mathbf{i} = (\mathbf{i}[1], \dots, \mathbf{i}[n+1])$  is a vector of non-negative integers. The norm  $\|\mathbf{i}\| = \sum_{j=1}^{n+1} \mathbf{i}[j]$

$\Delta$  simplex with vertices  $\{\mathbf{v}_1, \dots, \mathbf{v}_{n+1}\}$ .

$\mathbf{x} \in \Delta$ ,  $\lambda(\mathbf{x})$ : barycentric coordinates of  $\mathbf{x}$  w.r.t  $\Delta$ , that is,  $\mathbf{x} = \sum_k \lambda_k(\mathbf{x}) \mathbf{v}_k$  and  $\sum_k \lambda_k(\mathbf{x}) = 1$ .

For  $\mathbf{x} \in \Delta$ , polynomial  $\pi(\mathbf{x})$  (of degree  $d$ ) can be expressed as a **Bézier simplex**:

$$\pi(\mathbf{x}) = \sum_{\|\mathbf{i}\|=d} \mathbf{p}_{\mathbf{i}} B_{\mathbf{i},d}(\lambda_1(\mathbf{x}), \dots, \lambda_{n+1}(\mathbf{x}))$$

where  $B_{\mathbf{i},d}(y_1, \dots, y_{n+1}) = \binom{d}{\mathbf{i}} y_1^{\mathbf{i}[1]} y_2^{\mathbf{i}[2]} \dots y_{n+1}^{\mathbf{i}[n+1]}$  (**Bernstein polynomials**), with the multimomial coefficient  $\binom{d}{\mathbf{i}} = \frac{d!}{\mathbf{i}[1]! \mathbf{i}[2]! \dots \mathbf{i}[n+1]!}$ .

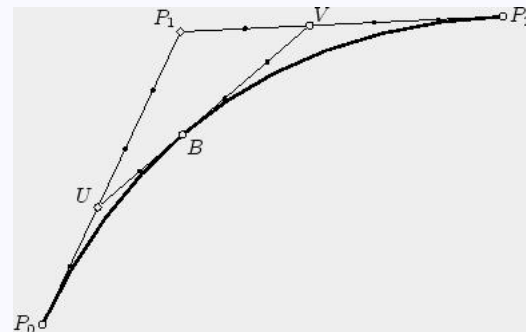
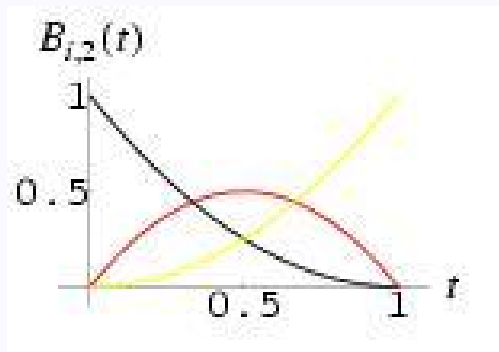
Each vector  $\mathbf{p}_{\mathbf{i}} \in \mathbb{R}^n$ : **Bézier control point**. All such  $\mathbf{p}_{\mathbf{i}}$  form the **Bézier control net**.

# Example - Bézier curves

Dimension  $n = 1$ . With  $d = 2$ , multi-indices with  $||\mathbf{i}|| = 2$ :  $\{(0, 2), (1, 1), (2, 0)\}$ .

A quadratic Bézier curve is:  $\pi(\mathbf{x}) = \sum_{\mathbf{i}} \mathbf{p}_{\mathbf{i}} B_{\mathbf{i},2}(\lambda_1(\mathbf{x}), \lambda_2(\mathbf{x}))$ , for  $\mathbf{x} \in [\mathbf{v}_1, \mathbf{v}_2]$ .

There are 3 control points  $\mathbf{p}_{\mathbf{i}}$ .



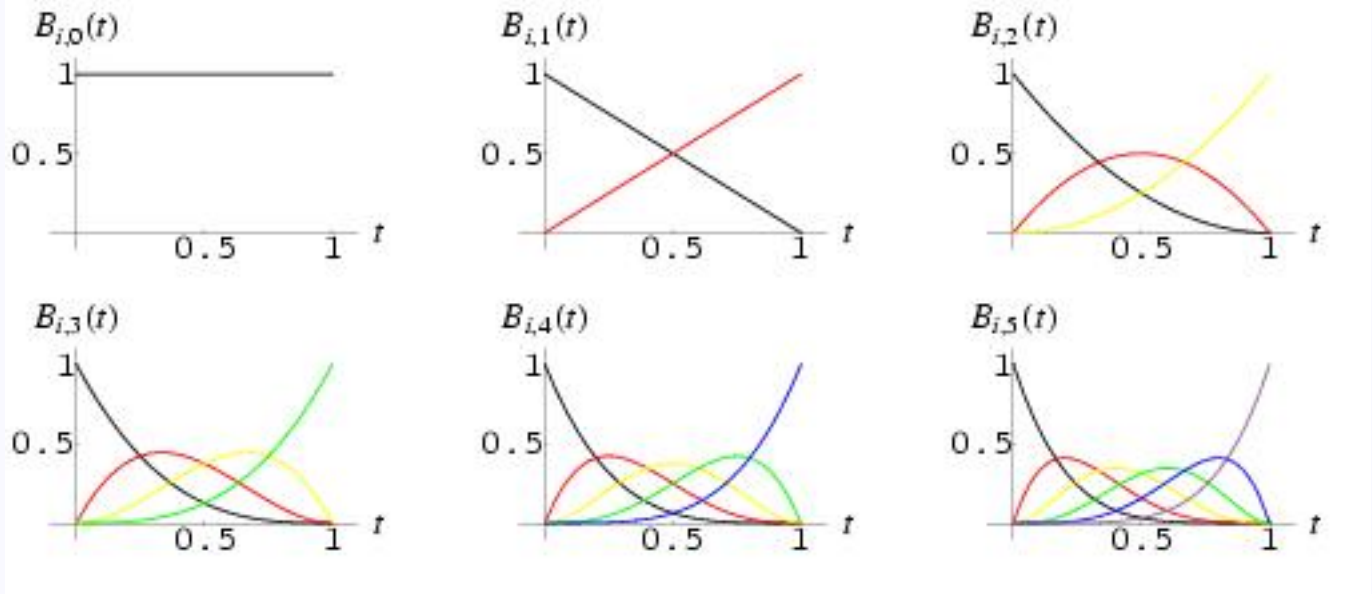
Bernstein polynomials

$$B_{\mathbf{i},2}(t) = B_{\mathbf{i},2}(y_1, y_2) = \binom{2}{\mathbf{i}} y_1^{\mathbf{i}[1]} y_2^{\mathbf{i}[2]}, y_1 = t \in [0, 1], y_2 = 1 - t$$



# Example - Bézier curves (cont'd)

Dimension  $n = 1$ .



Bernstein polynomials

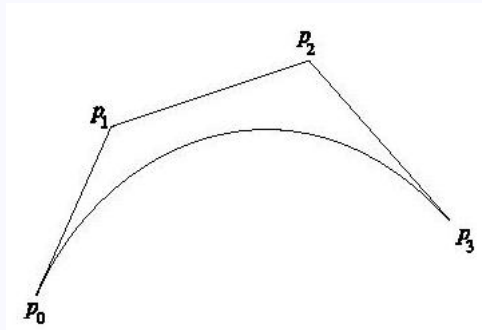
$$B_{i,d}(t) = B_{i,d}(y_1, y_2) = \binom{d}{i} y_1^{i^{[1]}} y_2^{i^{[2]}}, y_1 = t \in [0, 1], y_2 = 1 - t$$

# Bézier simplices - Shape Properties

Any polynomial can be written in form of a Bézier simplex. Given an arbitrary point  $\mathbf{x} \in \Delta$ ,

1. **Convex hull property:** the point  $\pi(\mathbf{x})$  lies inside the convex hull of the control net
2. **End-point interpolation property:** the polynomial  $\pi$  interpolates the control net at the corner control points.

Number of control points  $\mathbf{p}_i$  is  $\binom{d+n}{n} = \frac{(d+n)!}{d! n!}$ .



$\Rightarrow$  Using the convex hull of the Bézier control net as a **tight over-approximation** of  $\pi(\Delta)$ .

# Computing the Bézier control points

**Problem:** For a polynomial  $\pi$  given in monomial form and a simplex  $\Delta = \text{conv}\{\mathbf{v}_1, \dots, \mathbf{v}_{n+1}\}$ , we want to compute the control net of  $\pi$  with respect to  $\Delta$ .

**Blossoming principle** For any polynomial  $\pi : \mathbb{R}^n \rightarrow \mathbb{R}^n$  of degree  $d$ , there is a unique symmetric  $d$ -affine map  $\beta : \mathbb{R}^{nd} \rightarrow \mathbb{R}^n$  such that for all  $\mathbf{x} \in \mathbb{R}^n$ :  $\beta(\mathbf{x}, \dots, \mathbf{x}) = \pi(\mathbf{x})$ .

**Recall:**  $q(\mathbf{x}_1, \dots, \mathbf{x}_d)$  is *d-affine* if it is affine when all but one of its arguments are kept fixed; symmetric if its value does not depend on the ordering of the arguments.

Connection with the Bézier control net:

$$\mathbf{p}_i = \beta(\underbrace{\mathbf{v}_1, \dots, \mathbf{v}_1}_{i[1]}, \underbrace{\mathbf{v}_2, \dots, \mathbf{v}_2}_{i[2]}, \dots, \underbrace{\mathbf{v}_{n+1}, \dots, \mathbf{v}_{n+1}}_{i[n+1]})$$

$\Rightarrow$  To compute  $\mathbf{p}_i$  we can evaluate the blossom with some particular arguments

# Evaluating blossom values

Illustrate the computation of blossom values of polynomial  $(\mathbf{x}[i])^h(\mathbf{x}[j])^k$ .

$$\beta_{h,k}^d(\mathbf{u}_1, \dots, \mathbf{u}_d) = \frac{1}{\binom{d}{h} \binom{d-h}{k}} \sum_{\substack{I \cup J \subseteq \{1, \dots, d\}, \\ |I| = h, |J| = k, I \cap J = \emptyset}} \prod_{r \in I} \mathbf{u}_r[i] \prod_{s \in J} \mathbf{u}_s[j]$$

Denote  $\sigma_{h,k}^d = \frac{1}{\binom{d}{h} \binom{d-h}{k}} \beta_{h,k}^d(\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_d)$ .

The symmetric function  $\sigma$ : choose  $h$   $i^{th}$  coordinates of  $d$  argument points and  $k$   $j^{th}$  coordinates and form their product, then sum these products over all possible choices.

$$\begin{cases} \sigma_{h,k}^d = \sigma_{h,k}^{d-1} + \mathbf{u}_d[i] \sigma_{h-1,k}^{d-1} + \mathbf{u}_d[j] \sigma_{h,k-1}^{d-1} & \text{if } h, k \geq 0 \text{ and } h+k \geq 1, \\ \sigma_{0,0}^d = 0 \end{cases}$$

Complexity  $O(d^3)$

# Error bound and subdivision

Error in the approximation of the polynomial map  $\pi$  by its Bézier control points is  $\mathcal{O}(\rho^2)$  ( $\rho$  = max side length).

When  $\Delta$  is large  $\Rightarrow$  subdivide it into smaller simplices, which creates new Bézier bases and therefore new control points.

**Computation of the new control nets** that reuses the results obtained for the original simplex:

1. Partition the simplex  $\Delta$  by adding a point  $\mathbf{x} \in \Delta$  and forming new smaller simplices.
2. We can use *de Casteljau* algorithm to evaluate  $\pi(\mathbf{x})$  and this computation also produces the control net for the new simplices.

# Reachability algorithm

$$R_0 = X_0, k = 0$$

**REPEAT**

$$S_\Delta = \text{triangulation}(\text{vertices}(R_k))$$

$$C = \emptyset$$

**FORALL**( $\Delta \in S_\Delta$ )

$$C = C \cup \textcolor{red}{Bez}(\Delta)$$

**ENDFOR**

$$R_{k+1} = \text{conv}(C)$$

$$k = k + 1$$

**UNTIL**( $R_{k+1} = R_k$ )

$\textcolor{red}{Bez}$  over-approximates  $P(\Delta)$ ,  $P$  is the integration map

# Bacteria Vibrio Fisheri

$$\begin{cases} \dot{x}_1 = k_2x_2 - k_1x_1x_3 + u_1 \\ \dot{x}_2 = k_1x_1x_3 - k_2x_2 \\ \dot{x}_3 = k_2x_2 - k_1x_1x_3 - nx_3 + nu_2 \end{cases}$$

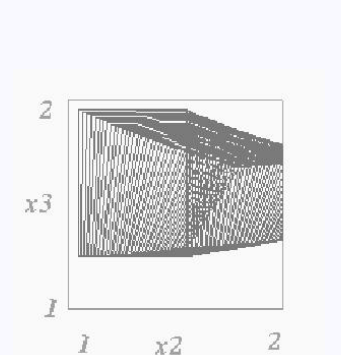
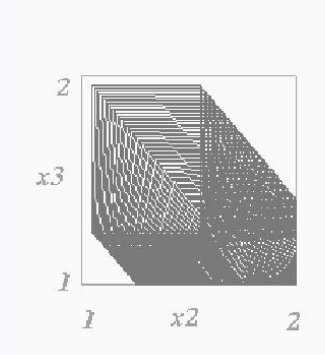
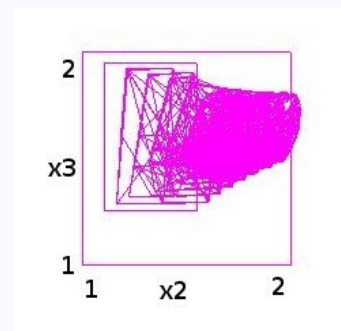
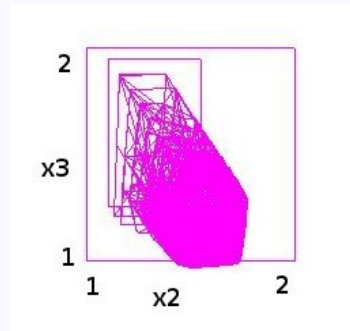
**State variables**  $(x_1, x_2, x_3)$ : cellular concentrations of different species.

**Parameters**  $k_1, k_2, n$ : binding, dissociation and diffusion constants.

**Control objective**: steering the system to the face  $x_2 = 2$  (activation of some genes).

**Control laws**:  $u_1(\mathbf{x}) = -10(x_2 + x_1(-1 + 3) - 4x_3)$  and  $u_2(\mathbf{x}) = x_1(3 + x_2(-1 + x_3)) - (-2 + x_2)x_3$ .

# Bacteria Vibrio Fisheri



The results are more precise than those obtained using abstraction by projection [Asarin Dang 03].



# Bacteria Vibrio Fisheri (cont'd)

Hybrid model with two modes and one additional continuous variable  $x_4$ . The continuous dynamics is  $\dot{\mathbf{x}} = A\mathbf{x} + g(\mathbf{x}) + b_{ij}$  where  $b_{01}$  and  $b_{10}$  correspond respectively to the non-luminescent and luminescent modes

$$A = \begin{pmatrix} \frac{-1}{H_{sp}} & 0 & 0 & r_{Co} \\ 0 & 0 & 0 & \frac{-1}{H_{sp}} - r_{Co} \\ 0 & x_0 r_{AII} & \frac{-1}{H_{AI}} & x_0 r_{Co} \\ 0 & \frac{-1}{H_{sp}} & 0 & 0 \end{pmatrix}; \quad g(\mathbf{x}) = \begin{pmatrix} -1 \\ 1 \\ -x_0 \\ 0 \end{pmatrix} r_{AIR} x_1 x_3$$

**Question:** determine the sets of states from which the system can reach the luminescent equilibrium. The condition for switching between the two modes is  $x_2 = x_{2sw}$ .

**Qualitative results compatible** with the previous study using  $d/dt$  (on a linearized model by fixing  $x_1$  constant),

**Larger set of states** that can reach the equilibrium.

# Concluding remarks

Combining the idea of set integration with techniques from CAGD

We achieved a second order approximation method

Higher order methods can be derived (but the resulting polynomials are more complex, i.e. more monomial terms)

Future directions: using splines??

Questions?? and thank you.

# Subdivision

$$\mathbf{p}_i^l = \beta(\underbrace{\mathbf{v}_1, \dots, \mathbf{v}_1}_{i[1]}, \dots, \underbrace{\mathbf{v}_{n+1}, \dots, \mathbf{v}_{n+1}}_{i[n+1]}, \underbrace{\mathbf{x}_1, \dots, \mathbf{x}_l}_l)$$

$$\mathbf{p}_i^l = \lambda_1(\mathbf{x}_l)\mathbf{p}_{i+\mathbf{e}_1}^{l-1} + \dots + \lambda_n(\mathbf{x}_l)\mathbf{p}_{i+\mathbf{e}_n}^{l-1}, \quad i[1] + \dots + i[n+1] + l = d$$

where  $\mathbf{e}_1 = (1, 0, \dots, 0)$ ,  $\mathbf{o} = (0, 0, \dots, 0)$ . Note that  $\mathbf{p}_i^0$  are the Bézier control points, and  $\mathbf{p}_o^d = \beta(\mathbf{x}_1, \dots, \mathbf{x}_d)$ .

We run the algorithm from  $l = 0$  to  $l = d$  with all  $\mathbf{x}_l = \mathbf{x}$  to obtain  $\mathbf{p}_o^d = \beta(\mathbf{x}, \dots, \mathbf{x})$ . In the following example,  $\mathbf{p}_{(2,0,0)}^1 = \lambda_1(\mathbf{x})\mathbf{p}_{(3,0,0)}^0 + \lambda_2(\mathbf{x})\mathbf{p}_{(2,1,0)}^0 + \lambda_3(\mathbf{x})\mathbf{p}_{(2,0,1)}^0$

