Computer-aided proofs in dynamics IV

Heterodimensional cycles* / invariant manifolds / CAPD

Maciej Capiński

AGH University of Kraków

***joint work with**

Bernd Krauskopf, Hinke Osinga, Piotr Zgliczyński

Plan of the presentation

- A correction
- Heterofimensional cycles
- Stable/unstable manifolds
- Some hints concerning CAPD

Hyperbolic set - correction!

$$f : \mathbb{R}^n \to \mathbb{R}^n$$

$N_i = (|N_i|, \eta_i)$ are h-sets

$$f_{ji} = \eta_j^{-1} \circ f \circ \eta_i$$

Theorem

Let $\Lambda = \mathrm{Inv}(f, \bigcup N_i)$ *and let*

$$\mathcal{K} = \mathrm{diag}(Id_u, -Id_s).$$

If for $z \in \eta_i^{-1}(f^{-1}(N_j) \cap N_i)$

$$Df_{ji}(z)^T \mathcal{K} Df_{ji}(z) - \mathcal{K}$$

is positive definite, then Λ *is a hyperbolic set.*

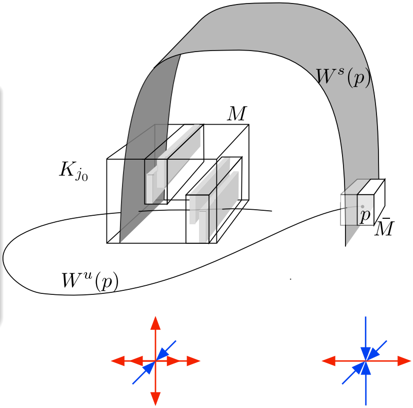
[W] Wilczak, Uniformly hyperbolic attractor of the Smale-Williams type for a Poincaré map in the Kuznetsov system, SIADS (2010)

Heterodimensional cycles

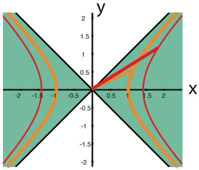
Definition (Heterodim. cycle)

$\Lambda, \bar{\Lambda}$ hyperbolic sets and

$$W^s(\Lambda) \cap W^u(\bar{\Lambda}) \neq \emptyset$$
$$W^u(\Lambda) \cap W^s(\bar{\Lambda}) \neq \emptyset$$
$$\dim W^s(\Lambda) \neq \dim W^s(\bar{\Lambda})$$



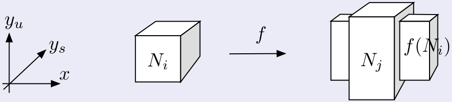
Our tools



$$C(z) = \{q : \|\pi_x(q-z)\| \geqslant \|\pi_y(q-z)\|\}$$

Definition

We say that $N_i \implies N_j$ iff



Lemma

If h is a disc in N and $N \overset{cc}{\implies} M$ then $\exists$ hor. disc $h^* : B_u \to M$

$$F(h(B_u)) \cap M = h^*(B_u)$$

Definition

f satisfies cone conditions iff

$$f(C(z)) \subset C(f(z))$$

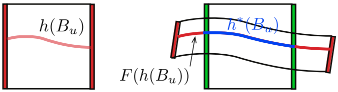
Remark: Cone conditions checked in local coordinates

Notation:

$$N_i \overset{cc}{\implies} N_j$$

covering and cone conditions

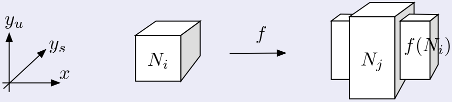
Our tools



$$C(z) = \{q : \|\pi_x(q-z)\| \geqslant \|\pi_y(q-z)\|\}$$

Definition

We say that $N_i \implies N_j$ iff



Lemma

If h is a disc in N and $N \overset{cc}{\implies} M$ then $\exists$ hor. disc $h^* : B_u \to M$

$$F(h(B_u)) \cap M = h^*(B_u)$$

Definition

f satisfies cone conditions iff

$$f(C(z)) \subset C(f(z))$$

Remark: Cone conditions checked in local coordinates

Notation:

$$N_i \overset{cc}{\implies} N_j$$

covering and cone conditions

Heterodimensional cycles

Consider two hyperbolic sets:

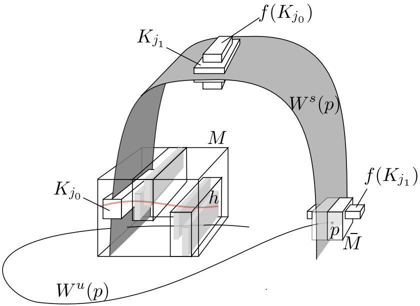
$$\Lambda \subset \bigcup N_i \qquad \bar{\Lambda} \subset \bigcup \bar{N}_i$$

Let $M \in \{N_i\}, \bar{M} \in \{\bar{N}_i\}$ and $\{K_j\}$ h-sets.

Definition

Λ is connected with $\bar{\Lambda}$ iff
$\forall h \in M$ hor. disc $\exists j_0, ..., j_k$ s.t.

- $h \cap K_{j_0}$ is a horizontal disc
- $K_{j_0} \overset{cc}{\Longrightarrow} \cdots \overset{cc}{\Longrightarrow} K_{j_k} \overset{cc}{\Longrightarrow} \bar{M}$



Heterodimensional cycles

Consider two hyperbolic sets:

$$\Lambda \subset \bigcup N_i \qquad \bar{\Lambda} \subset \bigcup \bar{N}_i$$

Let $M \in \{N_i\}, \bar{M} \in \{\bar{N}_i\}$ and $\{K_j\}$ h-sets.

Definition

Λ is connected with $\bar{\Lambda}$ iff
$\forall h \in M$ hor. disc $\exists j_0, \ldots, j_k$ s.t.

- $h \cap K_{j_0}$ is a horizontal disc
- $K_{j_0} \overset{cc}{\Longrightarrow} \cdots \overset{cc}{\Longrightarrow} K_{j_k} \overset{cc}{\Longrightarrow} \bar{M}$

Theorem

Consider two hyperbolic sets Λ_1, Λ_2 s.t.

$$\dim W^s(\Lambda_1) \neq \dim W^s(\Lambda_2)$$

If for every $a, b \in \{1, 2\}$

$$\Lambda_a \text{ is connected with } \Lambda_b$$

then there exists a (robust) heterodimensional cycle.

[BD] Bonatti, Diaz, Persistent nonhyperbolic transitive diffeomorphisms, Ann. of Math. 1996

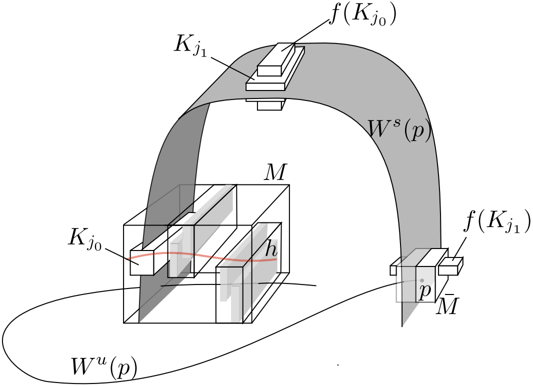
Heterodimensional cycles

Comments:

- Assumptions verifiable with CAP

$$K_0 \overset{cc}{\Longrightarrow} \cdots \overset{cc}{\Longrightarrow} K_n$$

- General framework
- In search of examples



Stable manifold theorem

$$F : \mathbb{R}^n \to \mathbb{R}^n$$

$$N = B_u \times B_s$$

Cone condition:
$Q(F(p) - F(q)) > Q(p - q)$

Assume: $N \overset{cc}{\Rightarrow} N$

Lemma

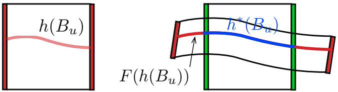
There exists $h^* : B_u \to N$

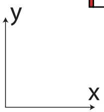
$$F(h(B_u)) \cap N = h^*(B_u)$$

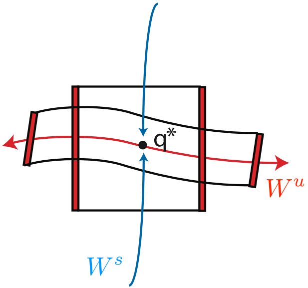
Theorem

There exists W^s *such that for any* $q \in W^s$

$$F^n(q) \in N \quad n \geqslant 0$$







P. Zgliczyński "Covering relations, cone conditions and the stable manifold theorem" JDE 2009

Stable manifold theorem

$$F : \mathbb{R}^n \to \mathbb{R}^n$$

$$N = B_u \times B_s$$

Cone condition:
$Q(F(p) - F(q)) > Q(p - q)$

Assume: $N \overset{cc}{\Rightarrow} N$

Lemma

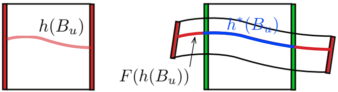
There exists $h^* : B_u \to N$

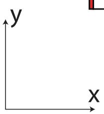
$$F(h(B_u)) \cap N = h^*(B_u)$$

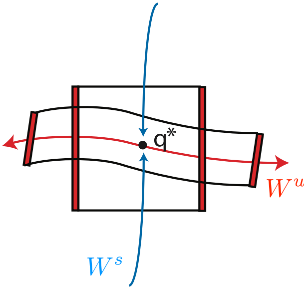
Theorem

There exists W^u *such that for any* $q \in W^u$

$$F^{-n}(q) \in N \quad n \geqslant 0$$





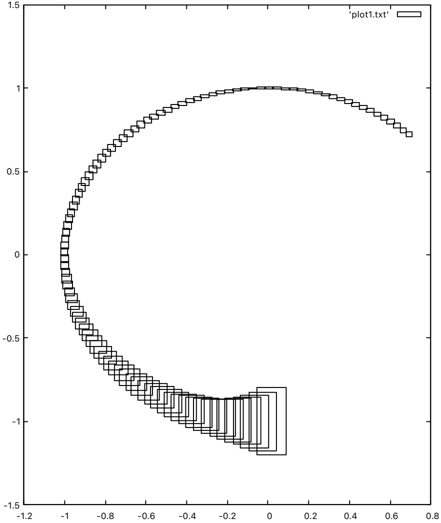


P. Zgliczyński "Covering relations, cone conditions and the stable manifold theorem" JDE 2009

CAPD - some tools

Flow along a solution of an ODE

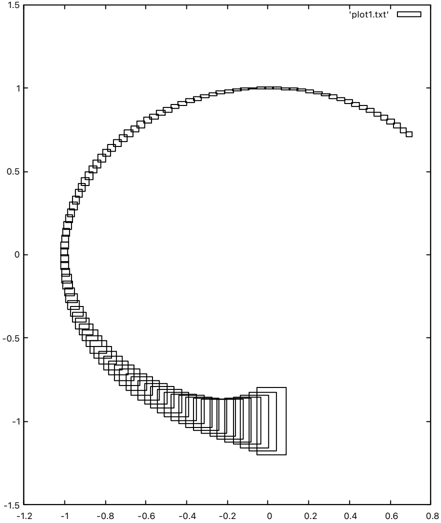
```
int order = 20;
IMap f = "var:x,y;"
  "fun:-y+x*(1-(x^2+y^2)^(-0.5)),"
  "x+y*(1-(x^2+y^2)^(-0.5));";

IOdeSolver solver(f,order);
ITimeMap Phi(solver);
interval T(4);

IVector x(2);
x[0] = interval(1)/sqrt(interval(2));
x[1] = interval(1)/sqrt(interval(2));

interval I=0.001*interval(-1,1);
IVector Y({I,I});
C0Rect2Set X(x+Y);
x = Phi(T,X);
```



CAPD - some tools

Flow along a solution of an ODE

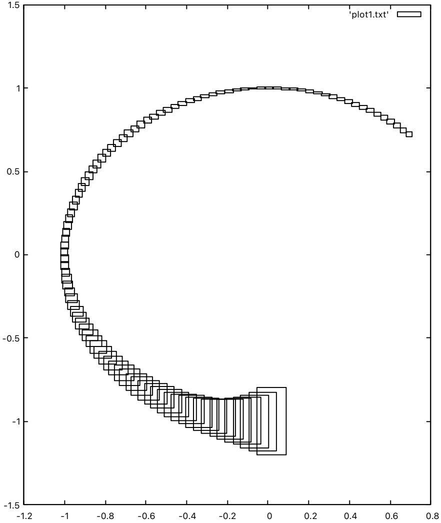
```
int order = 20;
IMap f = "var:x,y;"
  "fun:-y+x*(1-(x^2+y^2)^(-0.5)),"
  "x+y*(1-(x^2+y^2)^(-0.5));";

IOdeSolver solver(f,order);
ITimeMap Phi(solver);
interval T(4);

IVector x(2);
x[0] = interval(1)/sqrt(interval(2));
x[1] = interval(1)/sqrt(interval(2));

interval I=0.001*interval(-1,1);
IVector Y({I,I});
C0Rect2Set X(x+Y);
x = Phi(T,X);
```



CAPD - some tools

Flow along a solution of an ODE

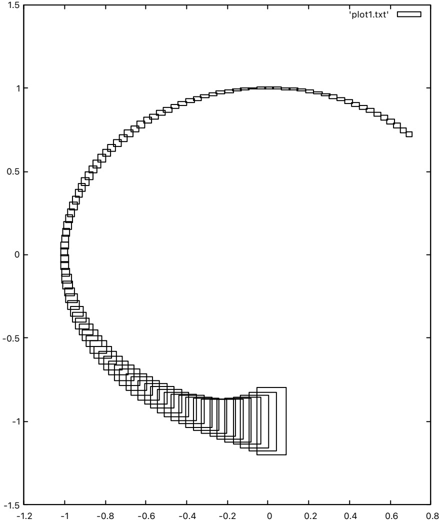
```
int order = 20;
IMap f = "var:x,y;"
  "fun:-y+x*(1-(x^2+y^2)^(-0.5)),"
  "x+y*(1-(x^2+y^2)^(-0.5));";

IOdeSolver solver(f,order);
ITimeMap Phi(solver);
interval T(4);

IVector x(2);
x[0] = interval(1)/sqrt(interval(2));
x[1] = interval(1)/sqrt(interval(2));

interval I=0.001*interval(-1,1);
IVector Y({I,I});
C0Rect2Set X(x+Y);
x = Phi(T,X);
```



CAPD - some tools

Flow along a solution of an ODE

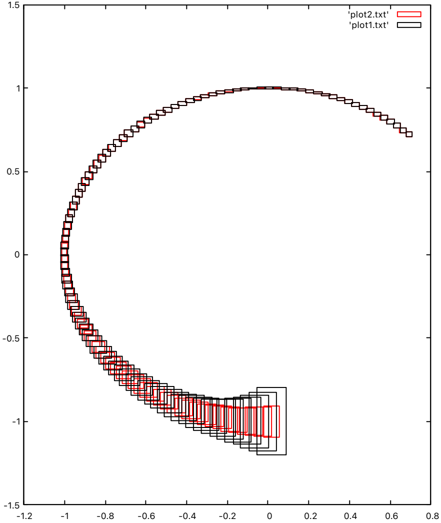
```
int order = 20;
IMap f = "var:x,y;"
  "fun:-y+x*(1-(x^2+y^2)^(-0.5)),"
  "x+y*(1-(x^2+y^2)^(-0.5));";

IOdeSolver solver(f,order);
ITimeMap Phi(solver);
interval T(4);

IVector x(2);
x[0] = interval(1)/sqrt(interval(2));
x[1] = interval(1)/sqrt(interval(2));

interval I=0.001*interval(-1,1);
IVector Y({I,I});
C0Rect2Set X(x+Y);
x = Phi(T,X);
```



CAPD - some tools

Flow along a solution of an ODE

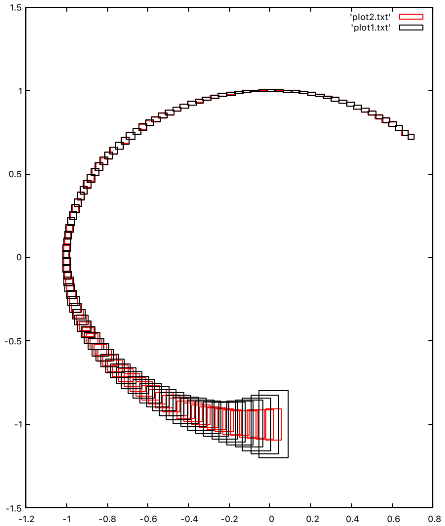
```
int order = 20;
IMap f = "var:x,y;"
  "fun:-y+x*(1-(x^2+y^2)^(-0.5)),"
  "x+y*(1-(x^2+y^2)^(-0.5));";

IOdeSolver solver(f,order);
ITimeMap Phi(solver);
interval T(4);

IVector x(2);
x[0] = interval(1)/sqrt(interval(2));
x[1] = interval(1)/sqrt(interval(2));

interval I=0.001*interval(-1,1);
IVector Y({I,I});
IMatrix A(2,2);
A = ...
C0Rect2Set X(x,A,Y);
x = Phi(T,X);
```



CAPD - some tools

Flow along a solution of an ODE

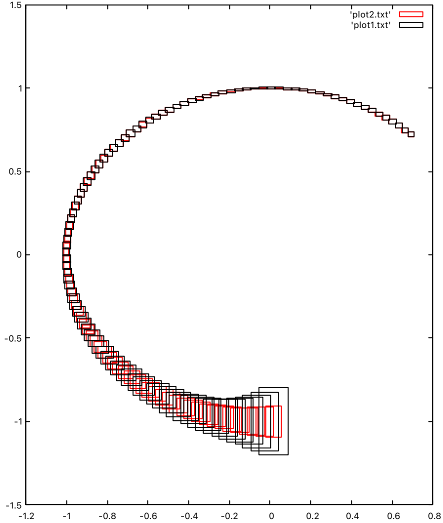
```
int order = 20;
IMap f = "var:x,y;"
  "fun:-y+x*(1-(x^2+y^2)^(-0.5)),"
  "x+y*(1-(x^2+y^2)^(-0.5));";

IOdeSolver solver(f,order);
ITimeMap Phi(solver);
interval T(4);

IVector x(2);
x[0] = interval(1)/sqrt(interval(2));
x[1] = interval(1)/sqrt(interval(2));

interval I=0.001*interval(-1,1);
IVector Y({I,I});
IMatrix A(2,2);
A = ...
C0Rect2Set X(x,A,Y);
SolutionCurve curve(0.0);
Phi(T,X,curve);
interval t=interval(1.0,1.1);
x = curve(t);
```



CAPD - some tools

Flow along a solution of an ODE

```
int order = 20;
IMap f = "var:x,y;"
  "fun:-y+x*(1-(x^2+y^2)^(-0.5)),"
  "x+y*(1-(x^2+y^2)^(-0.5));";

IOdeSolver solver(f,order);
ITimeMap Phi(solver);
interval T(4);

IVector x(2);
x[0] = interval(1)/sqrt(interval(2));
x[1] = interval(1)/sqrt(interval(2));

interval I=0.001*interval(-1,1);
IVector Y({I,I});
IMatrix A(2,2);
A = ...
C1Rect2Set X(x,A,Y);
IMatrix D(2,2)

Phi(T,X,D);
```

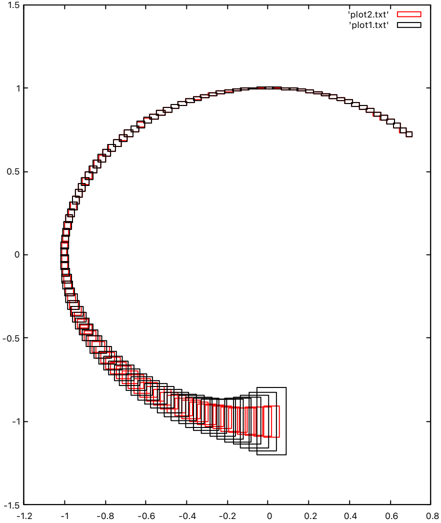


CAPD - some tools

Flow along a solution of an ODE

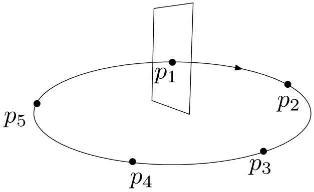
```
int order = 20;
IMap f = "var:x,y;"
  "fun:-y+x*(1-(x^2+y^2)^(-0.5)),"
  "x+y*(1-(x^2+y^2)^(-0.5));";

IOdeSolver solver(f,order);
ITimeMap Phi(solver);
interval T(4);

IVector x(2);
x[0] = interval(1)/sqrt(interval(2));
x[1] = interval(1)/sqrt(interval(2));

interval I=0.001*interval(-1,1);
IVector Y({I,I});
IMatrix A(2,2);
A = ...
C1Rect2Set X(x,A,Y);
IMatrix D(2,2);

Phi(T,X,D);
```

Coordinates are important.



How to set up Poincaré sections

Example



$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.

$$F : \mathbb{R}^{3n} \to \mathbb{R}^{3n}$$

$$p = (x, y, z)$$

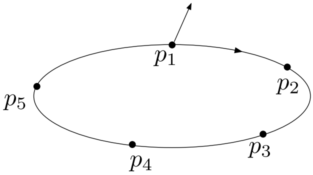
$$F(y_1, z_1, p_2, \ldots, p_{n-1}, T) = \begin{pmatrix} \phi_T(0, y_1, z_1) - p_2 \\ \phi_T(p_2) - p_3 \\ \ldots \\ \phi_T(p_{n-2}) - p_{n-1} \\ \phi_T(p_{n-1}) - (0, y_1, z_1) \end{pmatrix}$$

Use standard (not interval) Newton method

One can use Krawczyk method to validate $F(q^*) = 0$ for $q^* \in X$.

How to set up Poincaré sections

Example



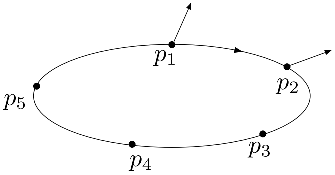
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$. Compute

$$v_{k+1} = \lambda_k D\mathcal{P}_k v_k$$

How to set up Poincaré sections

Example



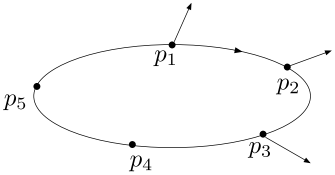
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$. Compute

$$v_{k+1} = \lambda_k D\mathcal{P}_k v_k$$

How to set up Poincaré sections

Example



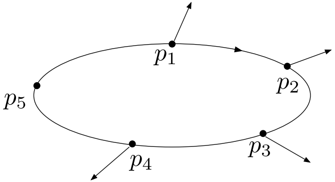
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$. Compute

$$v_{k+1} = \lambda_k D\mathcal{P}_k v_k$$

How to set up Poincaré sections

Example



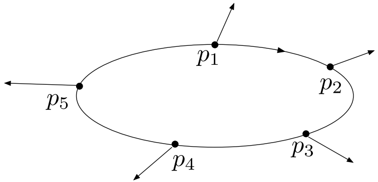
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$. Compute

$$v_{k+1} = \lambda_k D\mathcal{P}_k v_k$$

How to set up Poincaré sections

Example



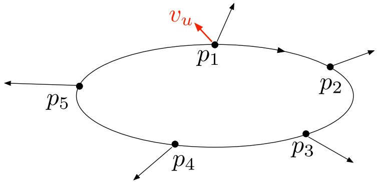
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$. Compute

$$v_{k+1} = \lambda_k D\mathcal{P}_k v_k$$

How to set up Poincaré sections

Example



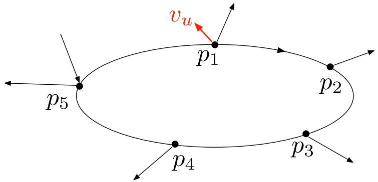
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$. Compute

$$v_{k+1} = \lambda_k D\mathcal{P}_k v_k$$

How to set up Poincaré sections

Example



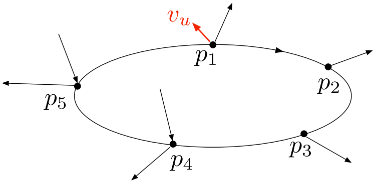
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$.

$$v_k = \lambda_k D\mathcal{P}_k^{-1} v_{k+1}$$

How to set up Poincaré sections

Example



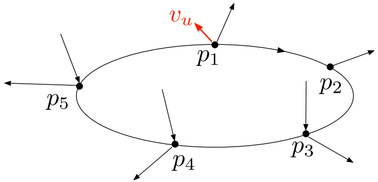
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$.

$$v_k = \lambda_k D\mathcal{P}_k^{-1} v_{k+1}$$

How to set up Poincaré sections

Example



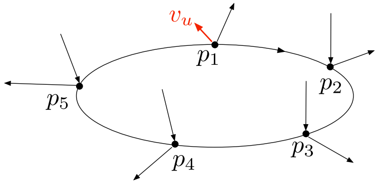
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$.

$$v_k = \lambda_k D\mathcal{P}_k^{-1} v_{k+1}$$

How to set up Poincaré sections

Example



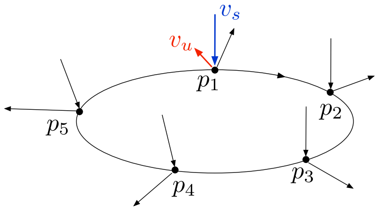
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$.

$$v_k = \lambda_k D\mathcal{P}_k^{-1} v_{k+1}$$

How to set up Poincaré sections

Example



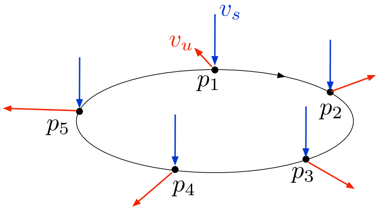
$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$.

$$v_k = \lambda_k D\mathcal{P}_k^{-1} v_{k+1}$$

How to set up Poincaré sections

Example



$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$

1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$.

$$v_k = \lambda_k D\mathcal{P}_k^{-1} v_{k+1}$$

3. Propagate v_u, v_s to $p_2, p_2, \ldots$

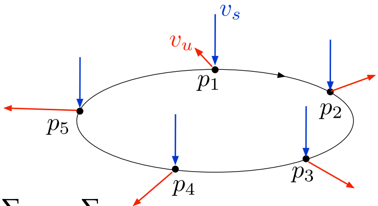
---

4. In interval arithmetic, make sure that v_u and v_s are on sections!

How to set up Poincaré sections

Example

$f : \mathbb{R}^3 \to \mathbb{R}^3 \qquad p' = f(p)$



1. Find periodic orbit.
2. Generate unstable/stable vector.
   Choose $\Sigma_k = \{v : v \perp f(p_k)\}$ and $\mathcal{P}_k : \Sigma_k \to \Sigma_{k+1}$.

$$v_k = \lambda_k D\mathcal{P}_k^{-1} v_{k+1}$$

3. Propagate v_u, v_s to $p_2, p_2, \ldots$

---

4. In interval arithmetic, make sure that v_u and v_s are on sections!
5. Choose

$$A_k = [v_u^k \; v_s^k \; f(p_k)]$$

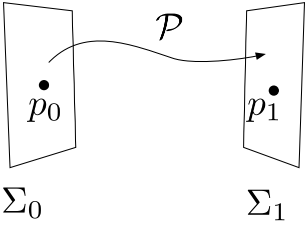
$$p = p_k + A_k w \qquad \Sigma_k = \{p_k + A_k(x, y, 0) : x, y \in \mathbb{R}\}$$

local map

$$(x, y) \mapsto \pi_{x,y}[A_{k+1}^{-1}](\mathcal{P}_k(p_k + A_k(x, y, 0)) - p_{k+1})$$

CAPD - some tools

Poincaré map code

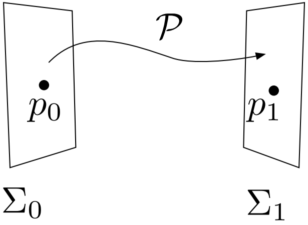


$$\Sigma_k = \{p_k + A_k(x, y, 0) : x, y \in \mathbb{R}\}$$

```
int order = 20;
IMap f = ...
IOdeSolver solver(f,order);

IVector p0=..., p1=...;
IAffineSection section(p1,f(p1));

IPoincareMap P(solver,section);

IMatrix A1=..., A0=...;
IVector Y=...;

C0Rect2Set X(p0,A0,Y);

cout << "f10(Y) = " << gauss(A1,P(X)-p1) << endl;
```

CAPD - some tools

Poincaré map code

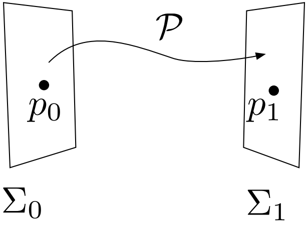


$$\Sigma_k = \{p_k + A_k(x, y, 0) : x, y \in \mathbb{R}\}$$

```
int order = 20;
IMap f = ...
IOdeSolver solver(f,order);

IVector p0=..., p1=...;
IAffineSection section(p1,f(p1));

IPoincareMap P(solver,section);

IMatrix A1=..., A0=...;
IVector Y=...;

C0Rect2Set X(p0,A0,Y);

cout << "f10(Y) = " << gauss(A1,P(X)-p1) << endl;
```

CAPD - some tools

Poincaré map code

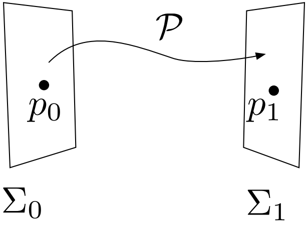


$$\Sigma_k = \{p_k + A_k(x, y, 0) : x, y \in \mathbb{R}\}$$

```
int order = 20;
IMap f = ...
IOdeSolver solver(f,order);

IVector p0=..., p1=...;
IAffineSection section(p1,f(p1));

IPoincareMap P(solver,section);

IMatrix A1=..., A0=...;
IVector Y=...;

C0Rect2Set X(p0,A0,Y);

cout << "f10(Y) = " << gauss(A1,P(X)-p1) << endl;
```

CAPD - some tools

Poincaré map code

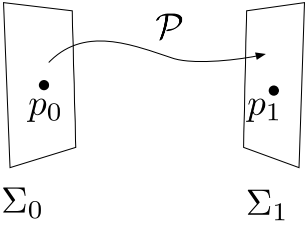


$$\Sigma_k = \{p_k + A_k(x, y, 0) : x, y \in \mathbb{R}\}$$

```
int order = 20;
IMap f = ...
IOdeSolver solver(f,order);

IVector p0=..., p1=...;
IAffineSection section(p1,f(p1));

IPoincareMap P(solver,section);

IMatrix A1=..., A0=...;
IVector Y=...;

C0Rect2Set X(p0,A0,Y);

cout << "f10(Y) = " << gauss(A1,P(X)-p1) << endl;
```

CAPD - some tools

Poincaré map code



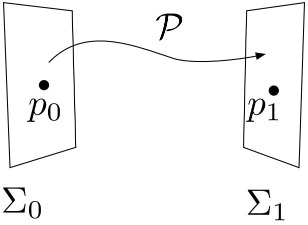
$$\Sigma_k = \{p_k + A_k(x, y, 0) : x, y \in \mathbb{R}\}$$

```
int order = 20;
IMap f = ...
IOdeSolver solver(f,order);

IVector p0=..., p1=...;
IAffineSection section(p1,f(p1));

IPoincareMap P(solver,section);

IMatrix A1=..., A0=...;
IVector Y=...;

C0Rect2Set X(p0,A0,Y);

cout << "f10(Y) = " << gauss(A1,P(X)-p1) << endl;
```

CAPD - some tools

Poincaré map code

```
int order = 20;
IMap f = ...

IVector p0=..., p1=...;
IAffineSection section(p1,f(p1));

IOdeSolver solver(f,order);
IPoincareMap P(solver,section);

IMatrix A1=...,A0=...;
IVector Y=...;

C1Rect2Set X(p0,A0,Y);
IMatrix DPhi;
IVector y = P(X,DPhi);
IMatrix DP = P.computeDP(y,DPhi);

cout << "f10(Y) = " << gauss(A1,y-p1) << endl;

cout << "Df10(Y) = " << gaussInverseMatrix(A1)*DP*A0 << endl;
```

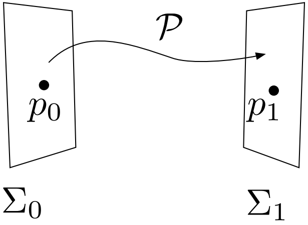


$$\Sigma_k = \{p_k + A_k(x, y, 0) : x, y \in \mathbb{R}\}$$

- **We should use:** $f_{10}(Y) \subset f_{10}(y_0) + [Df_{10}(Y)](Y - y_0)$

CAPD - some tools

Poincaré map code



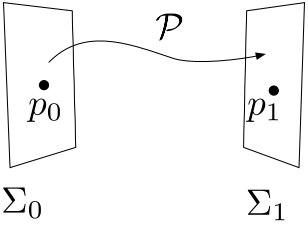
$$\Sigma_k = \{p_k + A_k(x, y, 0) : x, y \in \mathbb{R}\}$$

```
int order = 20;
IMap f = ...

IVector p0=..., p1=...;
IAffineSection section(p1,f(p1));

IOdeSolver solver(f,order);
IPoincareMap P(solver,section);

IMatrix A1=...,A0=...;
IVector Y=...;

C1Rect2Set X(p0,A0,Y);
IMatrix DPhi;
IVector y = P(X,DPhi);
IMatrix DP = P.computeDP(y,DPhi);

cout << "f10(Y) = " << gauss(A1,y-p1) << endl;

cout << "Df10(Y) = " << gaussInverseMatrix(A1)*DP*A0 << endl;
```

- **We should use:** $f_{10}(Y) \subset f_{10}(y_0) + [Df_{10}(Y)](Y - y_0)$

CAPD - some tools

Poincaré map code

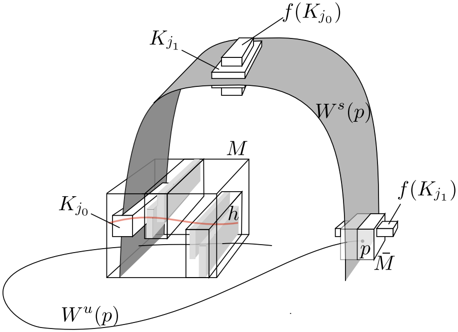


$$\Sigma_k = \{p_k + A_k(x, y, 0) : x, y \in \mathbb{R}\}$$

```
int order = 20;
IMap f = ...

IVector p0=..., p1=...;
IAffineSection section(p1,f(p1));

IOdeSolver solver(f,order);
IPoincareMap P(solver,section);

IMatrix A1=...,A0=...;
IVector Y=...;

C1Rect2Set X(p0,A0,Y);
IMatrix DPhi;
IVector y = P(X,DPhi);
IMatrix DP = P.computeDP(y,DPhi);

cout << "f10(Y) = " << gauss(A1,y-p1) << endl;

cout << "Df10(Y) = " << gaussInverseMatrix(A1)*DP*A0 << endl;
```

- **We should use:** $f_{10}(Y) \subset f_{10}(y_0) + [Df_{10}(Y)](Y - y_0)$

Final comments



- CAP provide a versatile tool for establishing properties of dynamical systems
- Blenders in concrete examples / 'real life' problems?
- Heterodimensional cycles?

Thank you for your attention.