Reduced System Computing

for

Singularly Perturbed Differential Equations

Warren Weckesser

Department of Mathematics
Colgate University

Acknowledgments

- NSF Grant DMS-0514468: *RUI: Reduced System Computing for Singularly Perturbed Differential Equations*

Acknowledgments

- NSF Grant DMS-0514468: *RUI: Reduced System Computing for Singularly Perturbed Differential Equations*
- Undergraduate students:
  - *Brian Kinney*
  - *Tomas Gruszka*
  - *Dimitar Simeonov*

Slow and Fast Subsystems, $x \in \mathbb{R}^m$, $y \in \mathbb{R}^n$

$$\boxed{\begin{aligned} \varepsilon\dot{x} &= f(x,y) \\ \dot{y} &= g(x,y) \end{aligned}}$$

Slow and Fast Subsystems, $x \in \mathbb{R}^m$, $y \in \mathbb{R}^n$

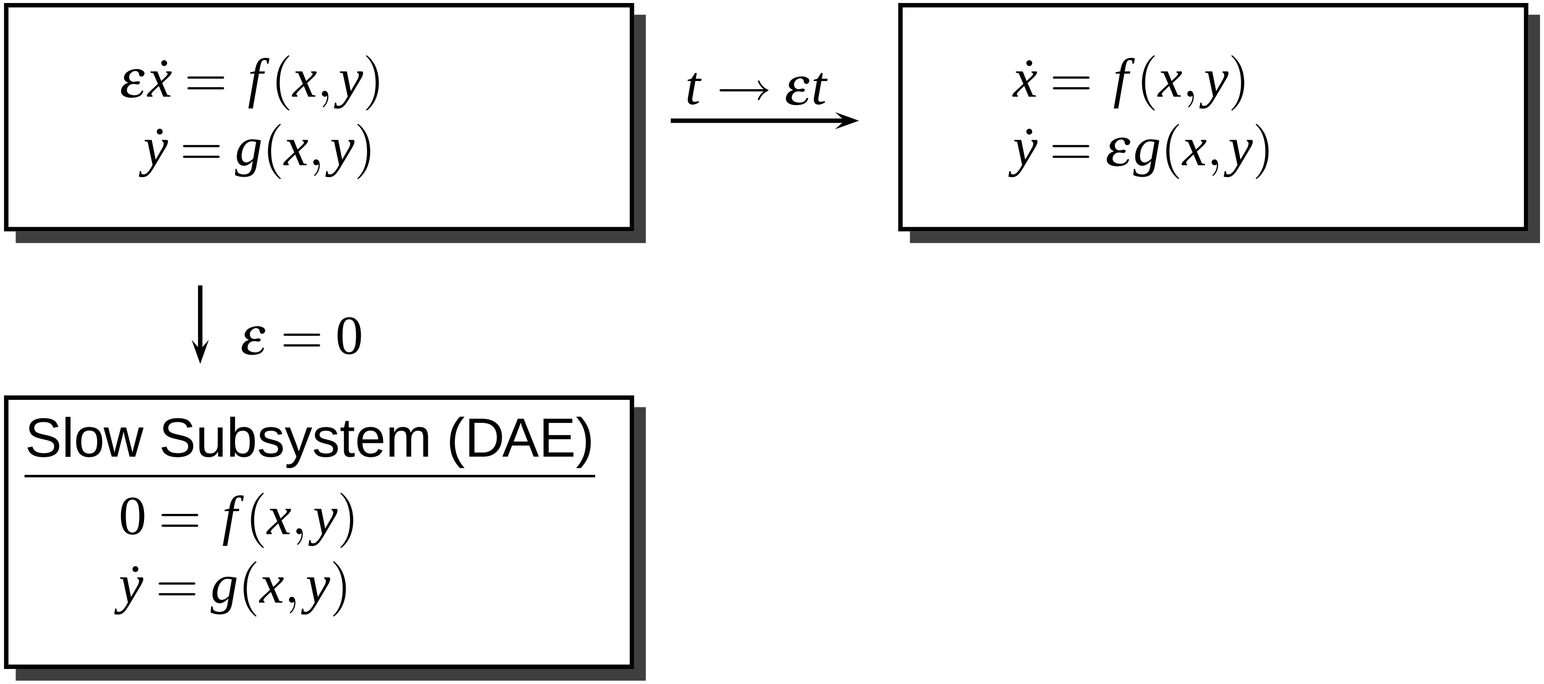
$$\begin{aligned} \varepsilon \dot{x} &= f(x,y) \\ \dot{y} &= g(x,y) \end{aligned}$$

$$\downarrow \; \varepsilon = 0$$

Slow Subsystem (DAE)

$$\begin{aligned} 0 &= f(x,y) \\ \dot{y} &= g(x,y) \end{aligned}$$

Slow and Fast Subsystems, $x \in \mathbb{R}^m$, $y \in \mathbb{R}^n$



$$\begin{aligned} \varepsilon\dot{x} &= f(x,y) \\ \dot{y} &= g(x,y) \end{aligned}$$

$\xrightarrow{t \longrightarrow \varepsilon t}$

$$\begin{aligned} \dot{x} &= f(x,y) \\ \dot{y} &= \varepsilon g(x,y) \end{aligned}$$

$\downarrow \; \varepsilon = 0$

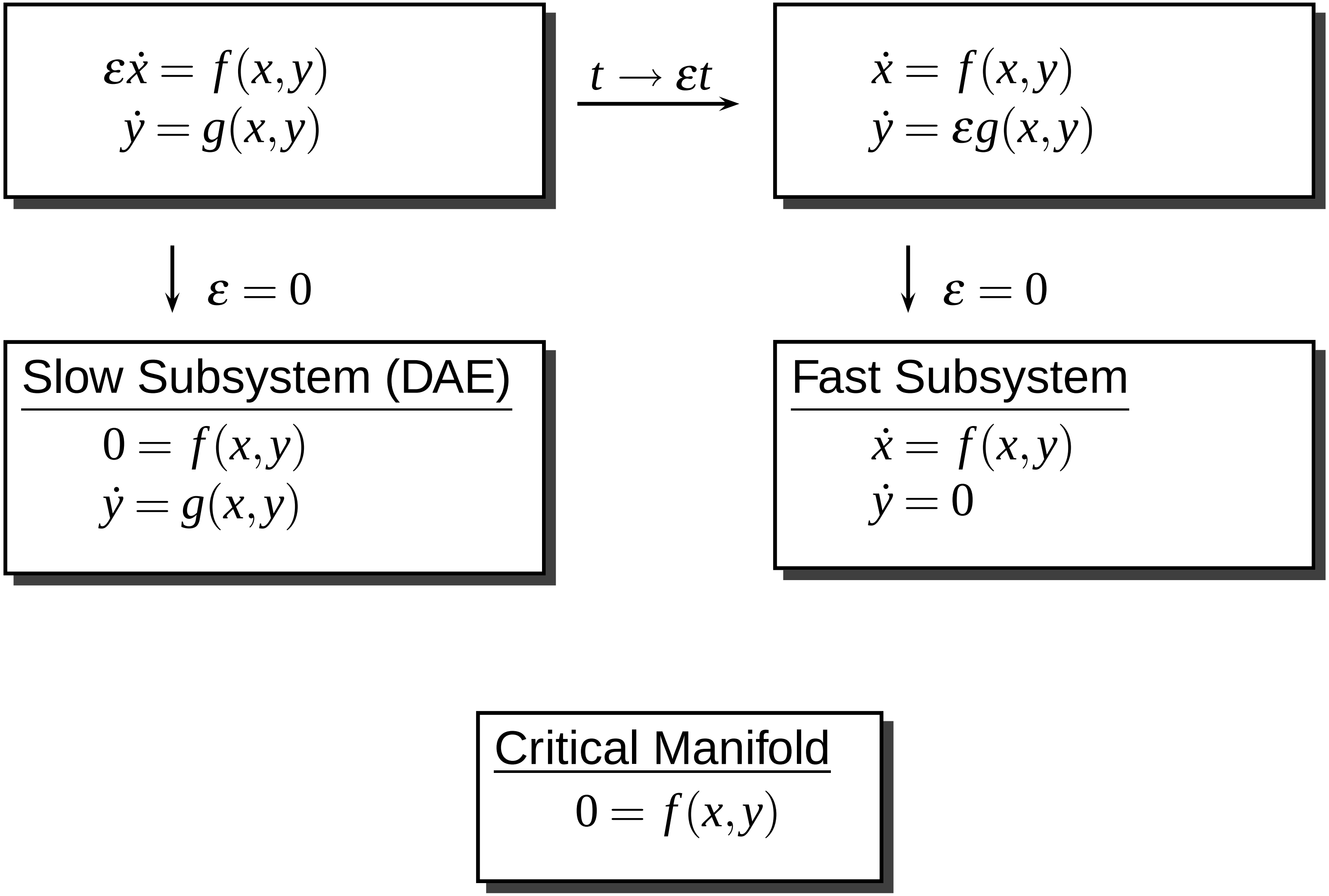
Slow Subsystem (DAE)

$$\begin{aligned} 0 &= f(x,y) \\ \dot{y} &= g(x,y) \end{aligned}$$

Slow and Fast Subsystems, $x \in \mathbb{R}^m$, $y \in \mathbb{R}^n$

$$\begin{tikzcd}
\boxed{\begin{array}{l} \varepsilon\dot{x} = f(x,y) \\ \dot{y} = g(x,y) \end{array}} \arrow[r, "t \longrightarrow \varepsilon t"] \arrow[d, "\varepsilon = 0"] & \boxed{\begin{array}{l} \dot{x} = f(x,y) \\ \dot{y} = \varepsilon g(x,y) \end{array}} \arrow[d, "\varepsilon = 0"] \\
\boxed{\begin{array}{l} \text{Slow Subsystem (DAE)} \\ \hline 0 = f(x,y) \\ \dot{y} = g(x,y) \end{array}} & \boxed{\begin{array}{l} \text{Fast Subsystem} \\ \hline \dot{x} = f(x,y) \\ \dot{y} = 0 \end{array}}
\end{tikzcd}$$

Slow and Fast Subsystems, $x \in \mathbb{R}^m$, $y \in \mathbb{R}^n$



$$\begin{aligned} \varepsilon \dot{x} &= f(x,y) \\ \dot{y} &= g(x,y) \end{aligned}$$

$t \to \varepsilon t$

$$\begin{aligned} \dot{x} &= f(x,y) \\ \dot{y} &= \varepsilon g(x,y) \end{aligned}$$

$\varepsilon = 0$

$\varepsilon = 0$

Slow Subsystem (DAE)

$$\begin{aligned} 0 &= f(x,y) \\ \dot{y} &= g(x,y) \end{aligned}$$

Fast Subsystem

$$\begin{aligned} \dot{x} &= f(x,y) \\ \dot{y} &= 0 \end{aligned}$$

Critical Manifold

$$0 = f(x,y)$$

Example: The Periodically Forced van der Pol System

$$x'' + d(x^2 - 1)x' + x = a\sin(\nu\tau)$$

Example: The Periodically Forced van der Pol System

$$x'' + d(x^2 - 1)x' + x = a\sin(\nu\tau)$$

New parameters: $\varepsilon = 1/d^2$, $\omega = \dfrac{\nu d}{2\pi}$

New variables: $t = \sqrt{\varepsilon}\tau$, $\theta = \omega t$, $y = \varepsilon\dot{x} + x^3/3 - x$

Then

$$\begin{aligned} \varepsilon\dot{x} &= x - \tfrac{1}{3}x^3 + y \\ \dot{y} &= -x + a\sin(2\pi\theta) \\ \dot{\theta} &= \omega \end{aligned}$$

Example: The Periodically Forced van der Pol System

$$x'' + d(x^2 - 1)x' + x = a\sin(\nu\tau)$$

New parameters: $\varepsilon = 1/d^2$, $\omega = \dfrac{\nu d}{2\pi}$

New variables: $t = \sqrt{\varepsilon}\tau$, $\theta = \omega t$, $y = \varepsilon\dot{x} + x^3/3 - x$

Then

$$\varepsilon\dot{x} = x - \frac{1}{3}x^3 + y$$

$$\dot{y} = -x + a\sin(2\pi\theta)$$

$$\dot{\theta} = \omega$$

Symmetry: $x \longrightarrow -x$, $y \longrightarrow -y$, $\theta \longrightarrow \theta + 1/2$

Forced van der Pol Slow and Fast Subsystems

$$\boxed{\begin{aligned} &\varepsilon\dot{x} = x - \frac{1}{3}x^3 + y \\ &\dot{y} = -x + a\sin(2\pi\theta) \\ &\dot{\theta} = \omega \end{aligned}}$$

Forced van der Pol Slow and Fast Subsystems

$$\boxed{\begin{aligned}\varepsilon\dot{x} &= x - \frac{1}{3}x^3 + y\\ \dot{y} &= -x + a\sin(2\pi\theta)\\ \dot{\theta} &= \omega\end{aligned}}$$

$$\Big\downarrow\ \varepsilon = 0$$

Slow Subsystem (DAE)

$$\begin{aligned}y &= \frac{1}{3}x^3 - x\\ \dot{y} &= -x + a\sin(2\pi\theta)\\ \dot{\theta} &= \omega\end{aligned}$$

Forced van der Pol Slow and Fast Subsystems

$$\begin{aligned}
\varepsilon\dot{x} &= x - \frac{1}{3}x^3 + y\\
\dot{y} &= -x + a\sin(2\pi\theta)\\
\dot{\theta} &= \omega
\end{aligned}$$

$$\Big\downarrow \; \varepsilon = 0$$

Slow Subsystem (DAE)

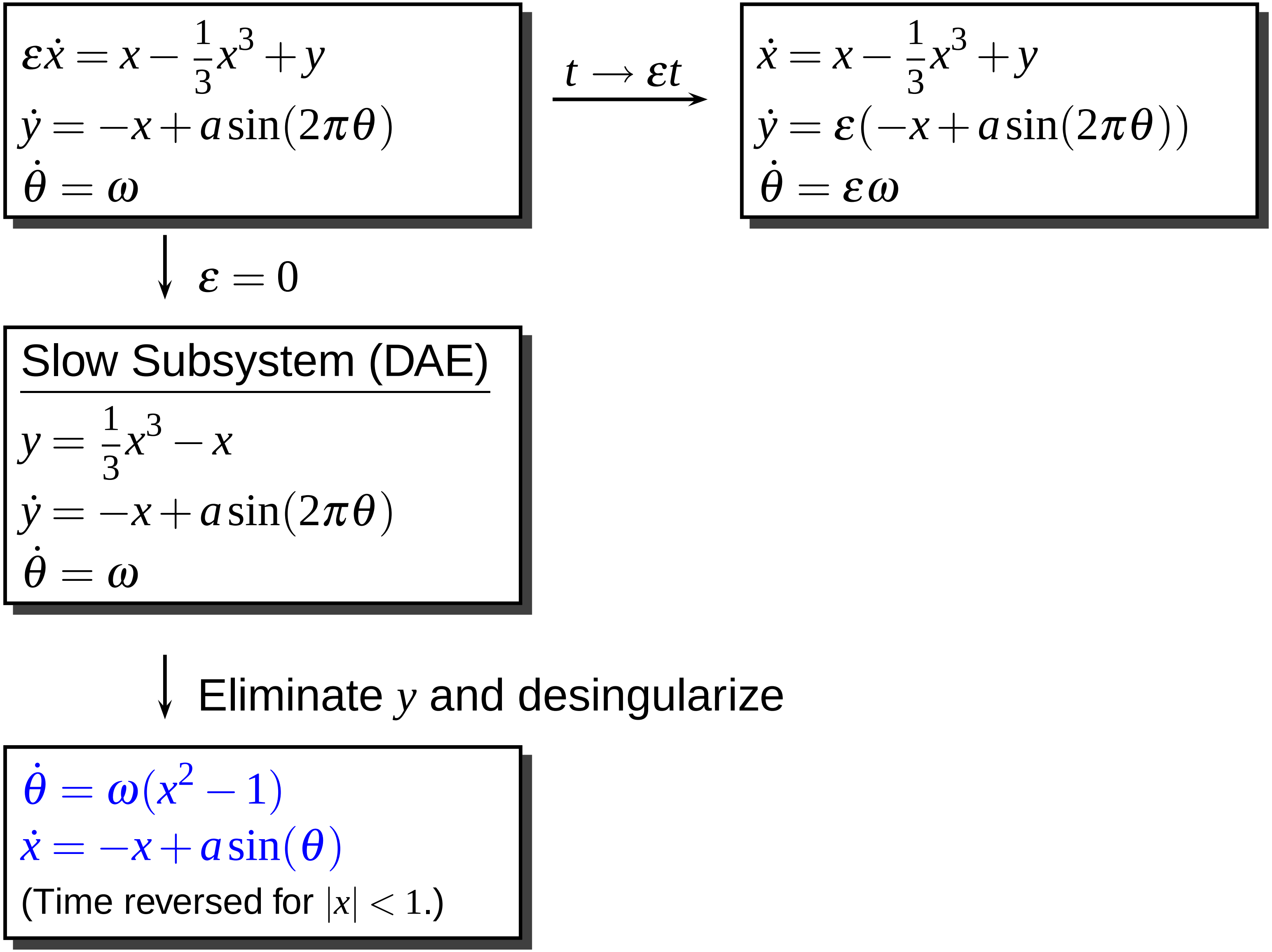
$$\begin{aligned}
y &= \frac{1}{3}x^3 - x\\
\dot{y} &= -x + a\sin(2\pi\theta)\\
\dot{\theta} &= \omega
\end{aligned}$$

$\Big\downarrow$ Eliminate y and desingularize

$$\begin{aligned}
\dot{\theta} &= \omega(x^2 - 1)\\
\dot{x} &= -x + a\sin(\theta)
\end{aligned}$$

(Time reversed for $|x| < 1$.)

Forced van der Pol Slow and Fast Subsystems



$$\begin{aligned} \varepsilon\dot{x} &= x - \frac{1}{3}x^3 + y \\ \dot{y} &= -x + a\sin(2\pi\theta) \\ \dot{\theta} &= \omega \end{aligned}$$

$$\xrightarrow{t \to \varepsilon t}$$

$$\begin{aligned} \dot{x} &= x - \frac{1}{3}x^3 + y \\ \dot{y} &= \varepsilon(-x + a\sin(2\pi\theta)) \\ \dot{\theta} &= \varepsilon\omega \end{aligned}$$

$\downarrow$ $\varepsilon = 0$

Slow Subsystem (DAE)

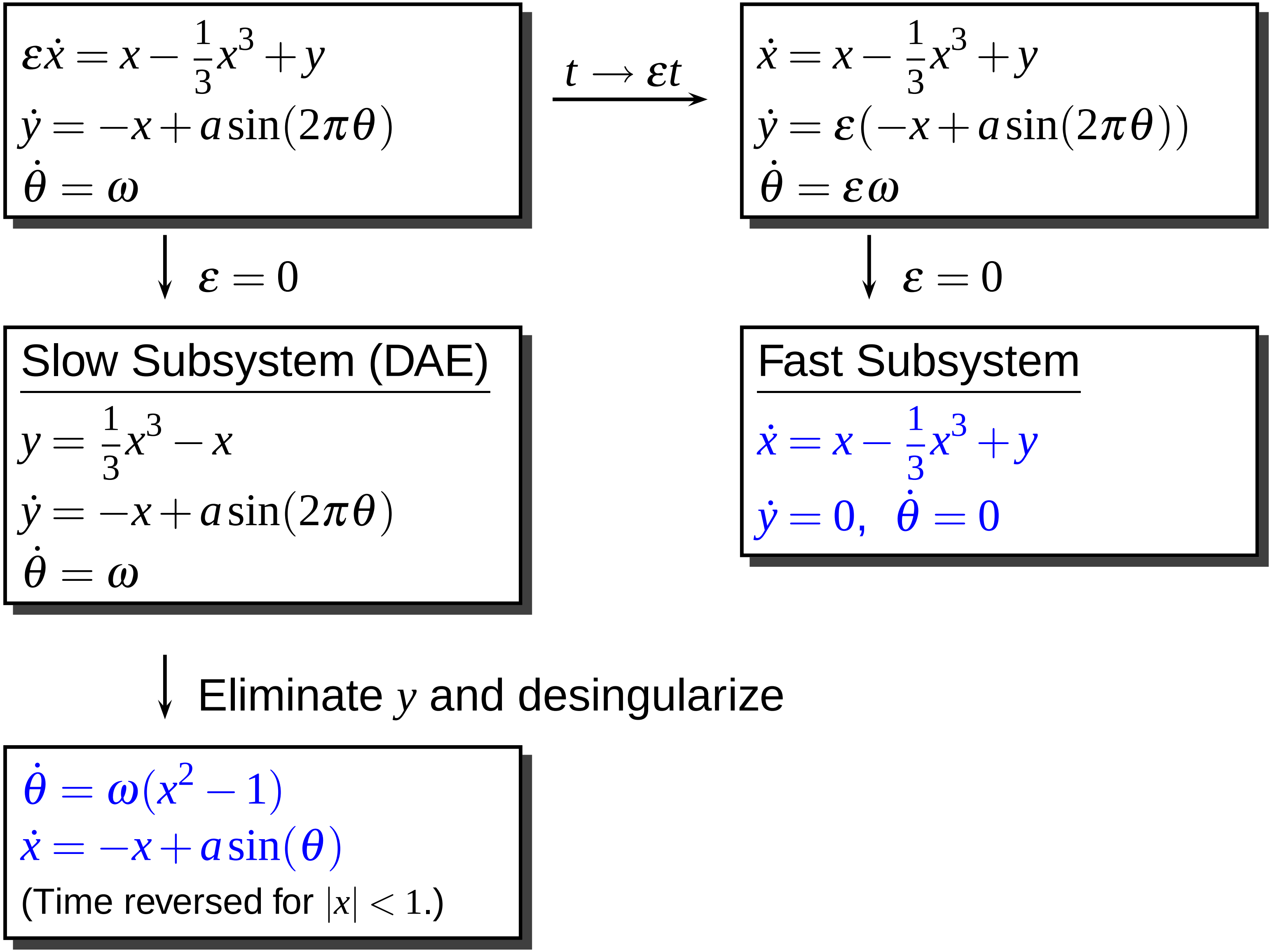
$$\begin{aligned} y &= \frac{1}{3}x^3 - x \\ \dot{y} &= -x + a\sin(2\pi\theta) \\ \dot{\theta} &= \omega \end{aligned}$$

$\downarrow$ Eliminate y and desingularize

$$\begin{aligned} \dot{\theta} &= \omega(x^2 - 1) \\ \dot{x} &= -x + a\sin(\theta) \end{aligned}$$

(Time reversed for $|x| < 1$.)

Forced van der Pol Slow and Fast Subsystems



$$\varepsilon\dot{x} = x - \frac{1}{3}x^3 + y$$
$$\dot{y} = -x + a\sin(2\pi\theta)$$
$$\dot{\theta} = \omega$$

$$\xrightarrow{t \to \varepsilon t}$$

$$\dot{x} = x - \frac{1}{3}x^3 + y$$
$$\dot{y} = \varepsilon(-x + a\sin(2\pi\theta))$$
$$\dot{\theta} = \varepsilon\omega$$

$\downarrow$ $\varepsilon = 0$

Slow Subsystem (DAE)

$$y = \frac{1}{3}x^3 - x$$
$$\dot{y} = -x + a\sin(2\pi\theta)$$
$$\dot{\theta} = \omega$$

$\downarrow$ $\varepsilon = 0$

Fast Subsystem

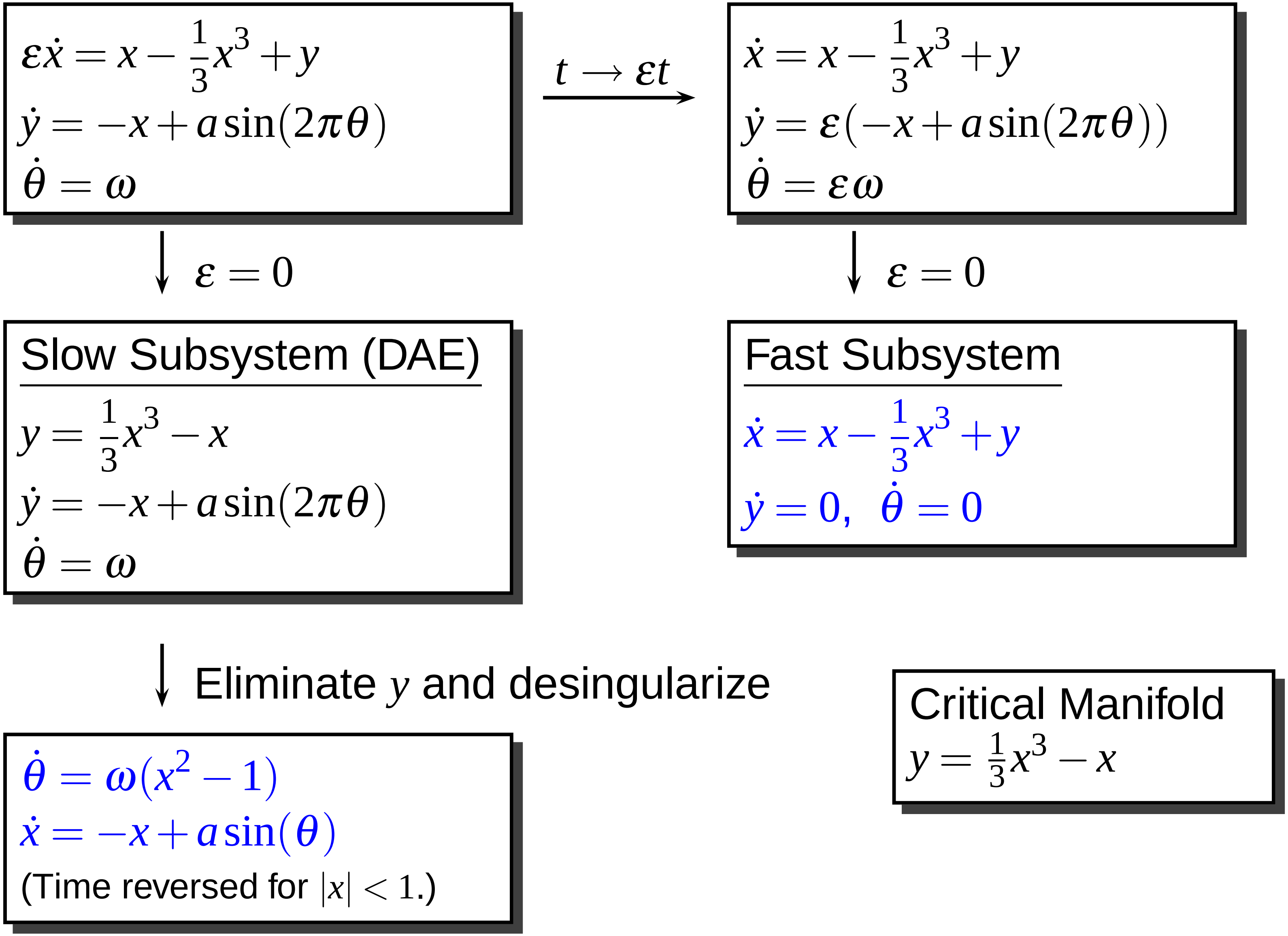
$$\dot{x} = x - \frac{1}{3}x^3 + y$$
$$\dot{y} = 0, \quad \dot{\theta} = 0$$

$\downarrow$ Eliminate y and desingularize

$$\dot{\theta} = \omega(x^2 - 1)$$
$$\dot{x} = -x + a\sin(\theta)$$

(Time reversed for $|x| < 1$.)

Forced van der Pol Slow and Fast Subsystems



$$\varepsilon\dot{x} = x - \frac{1}{3}x^3 + y$$
$$\dot{y} = -x + a\sin(2\pi\theta)$$
$$\dot{\theta} = \omega$$

$$\xrightarrow{t \to \varepsilon t}$$

$$\dot{x} = x - \frac{1}{3}x^3 + y$$
$$\dot{y} = \varepsilon(-x + a\sin(2\pi\theta))$$
$$\dot{\theta} = \varepsilon\omega$$

$\downarrow$ $\varepsilon = 0$

$\downarrow$ $\varepsilon = 0$

Slow Subsystem (DAE)

$$y = \frac{1}{3}x^3 - x$$
$$\dot{y} = -x + a\sin(2\pi\theta)$$
$$\dot{\theta} = \omega$$

Fast Subsystem

$$\dot{x} = x - \frac{1}{3}x^3 + y$$
$$\dot{y} = 0, \quad \dot{\theta} = 0$$

$\downarrow$ Eliminate y and desingularize

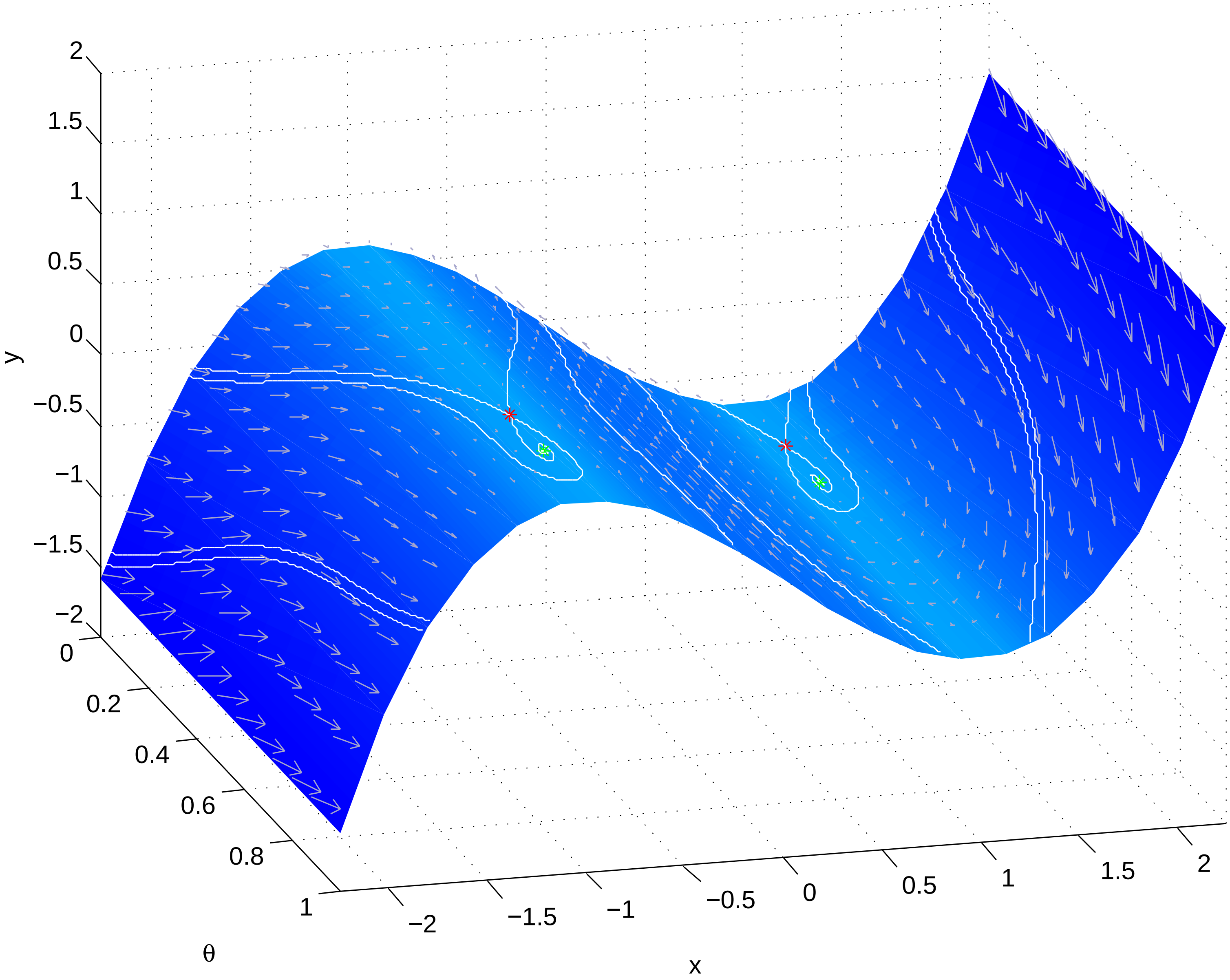
$$\dot{\theta} = \omega(x^2 - 1)$$
$$\dot{x} = -x + a\sin(\theta)$$

(Time reversed for $|x| < 1$.)

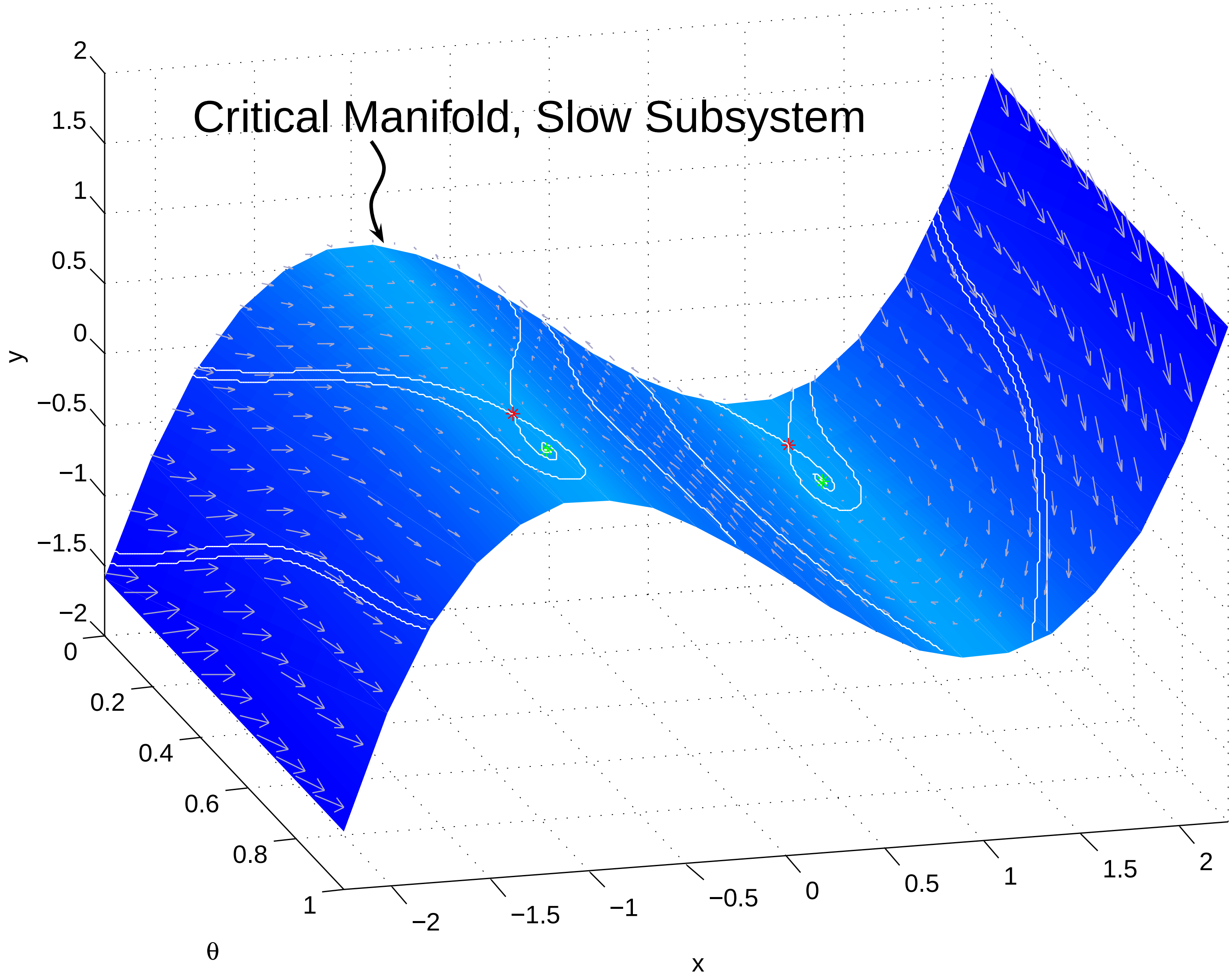
Critical Manifold

$$y = \frac{1}{3}x^3 - x$$

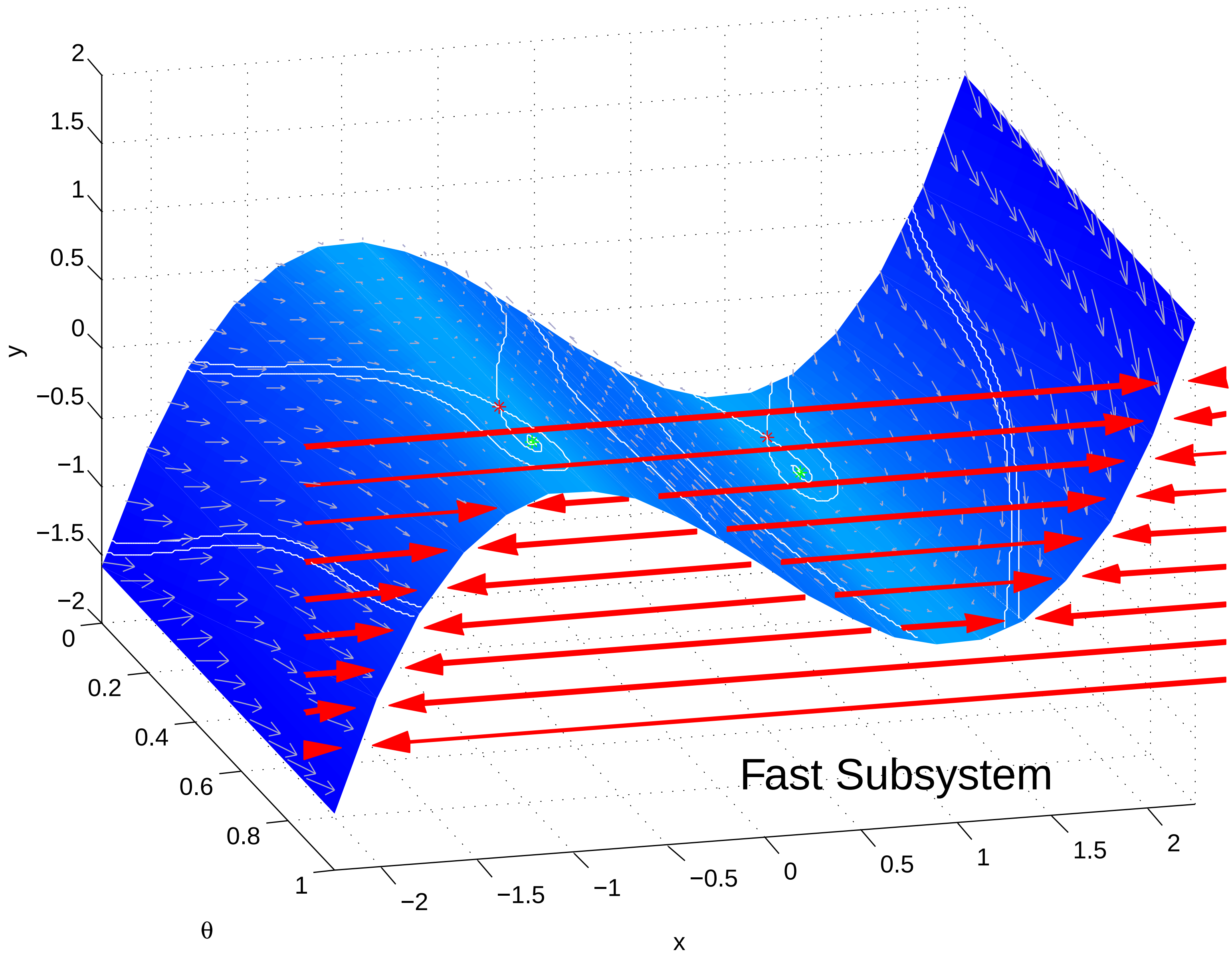
Forced van der Pol: Fast and Slow Subsystems



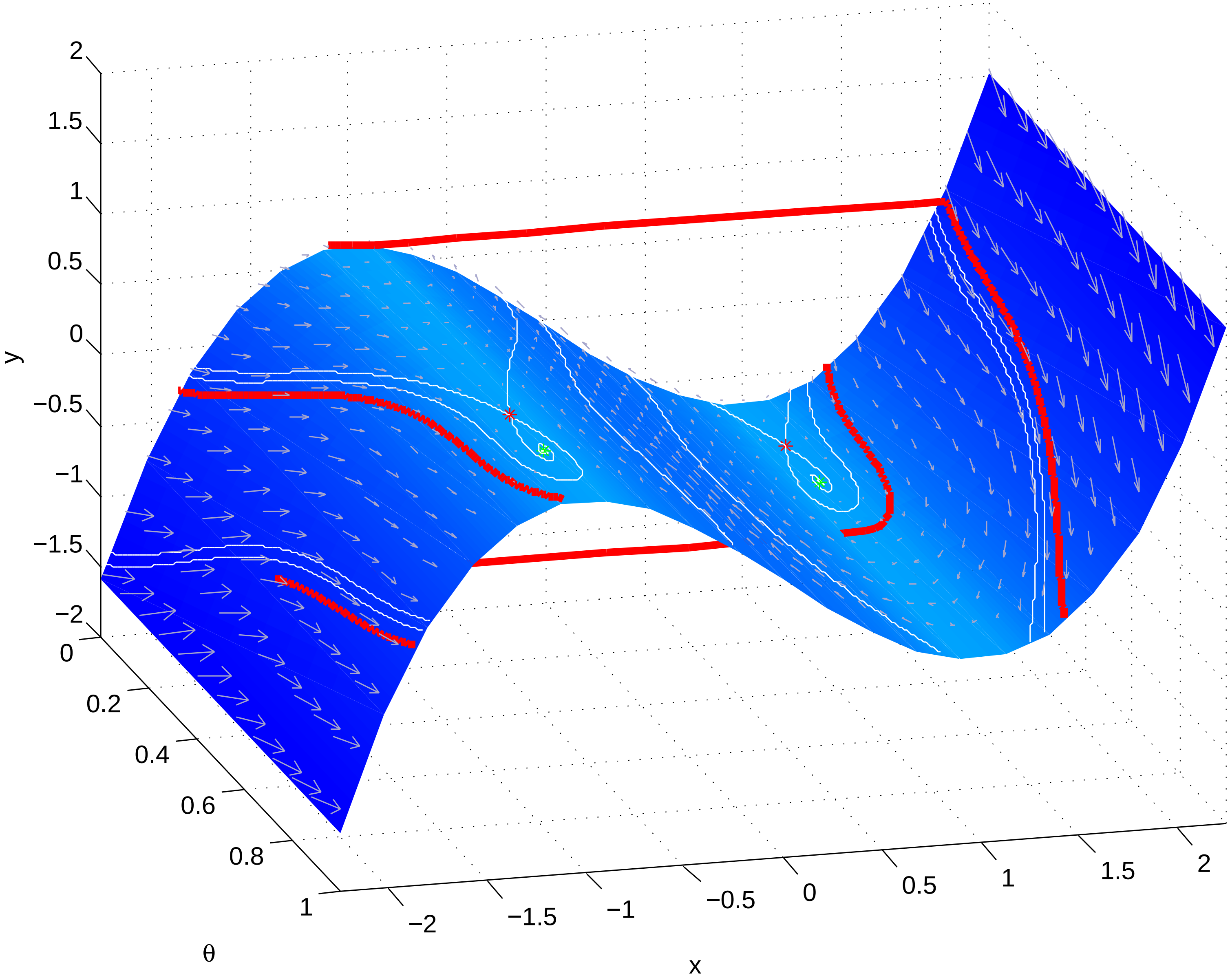
Forced van der Pol: Fast and Slow Subsystems


Critical Manifold, Slow Subsystem

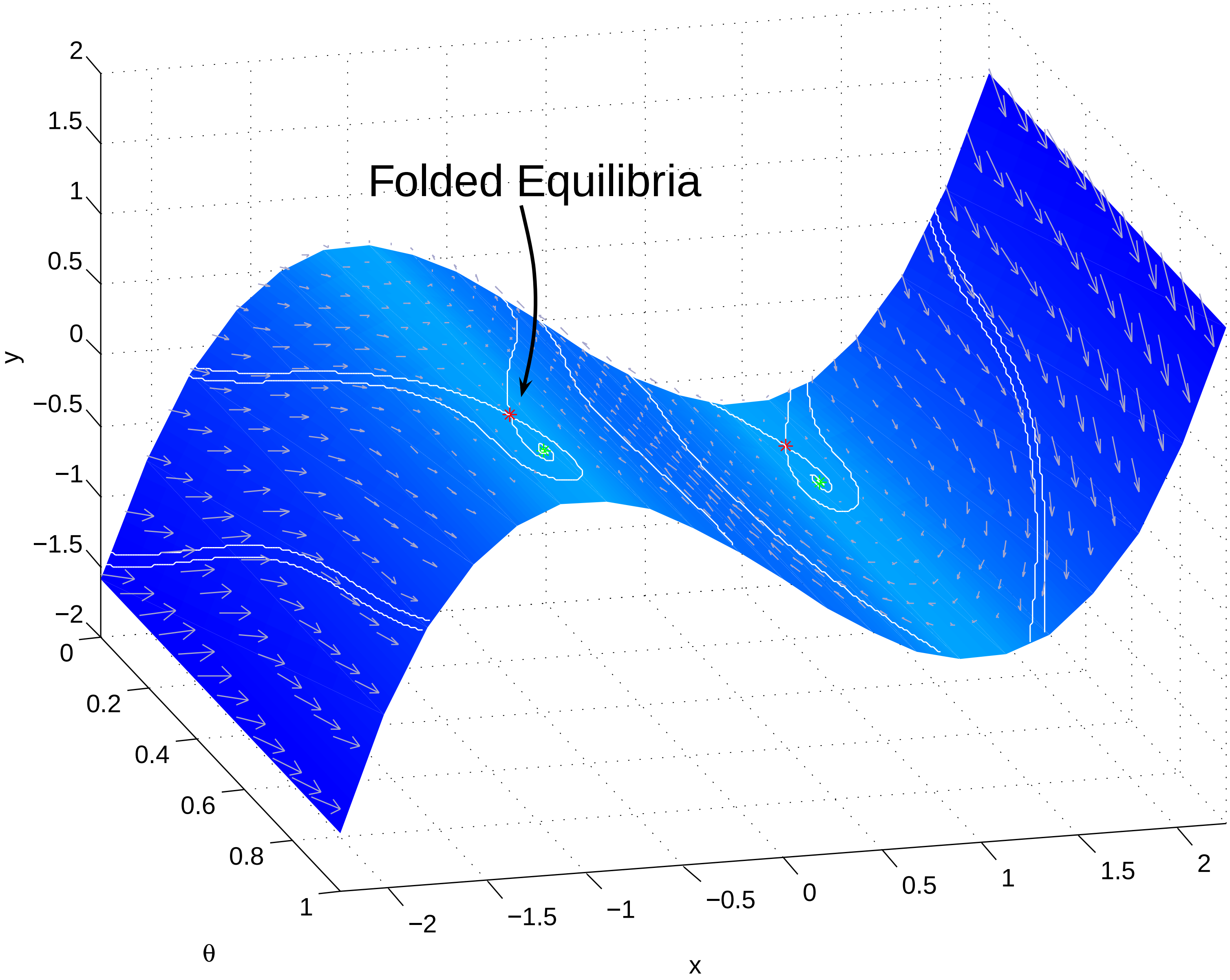
Forced van der Pol: Fast and Slow Subsystems



Forced van der Pol: Fast and Slow Subsystems



Forced van der Pol: Fast and Slow Subsystems



Canards at a Folded Saddle

Representative System (one fast, two slow variables):

$$\begin{aligned} \varepsilon\dot{x} &= y + x^2 \\ \dot{y} &= az + bx \\ \dot{z} &= 1 \end{aligned}$$

Critical manifold is $y = -x^2$. The origin is a folded equilibrium.

Canards at a Folded Saddle

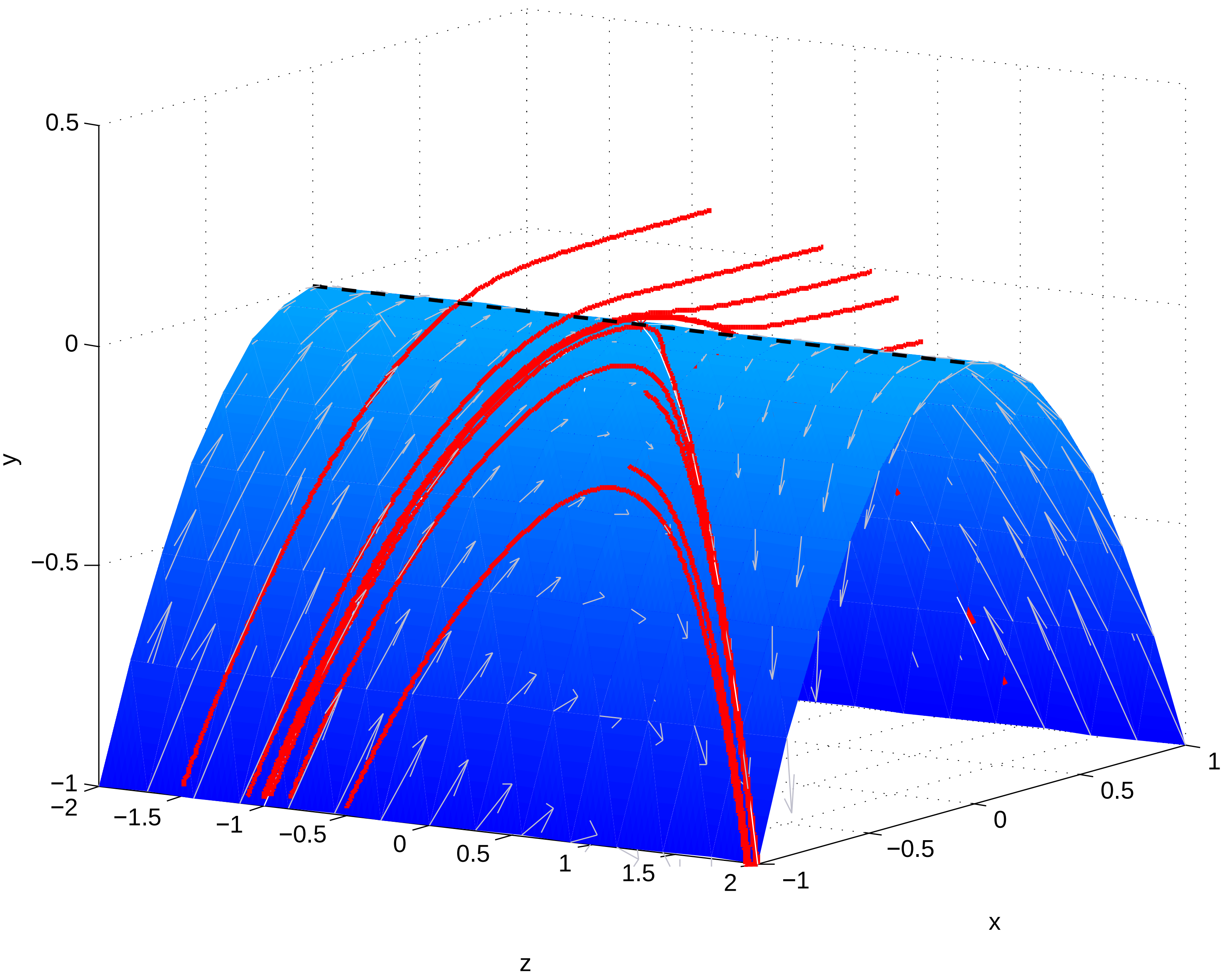
Representative System (one fast, two slow variables):

$$\begin{aligned}\varepsilon\dot{x} &= y + x^2\\ \dot{y} &= az + bx\\ \dot{z} &= 1\end{aligned}$$

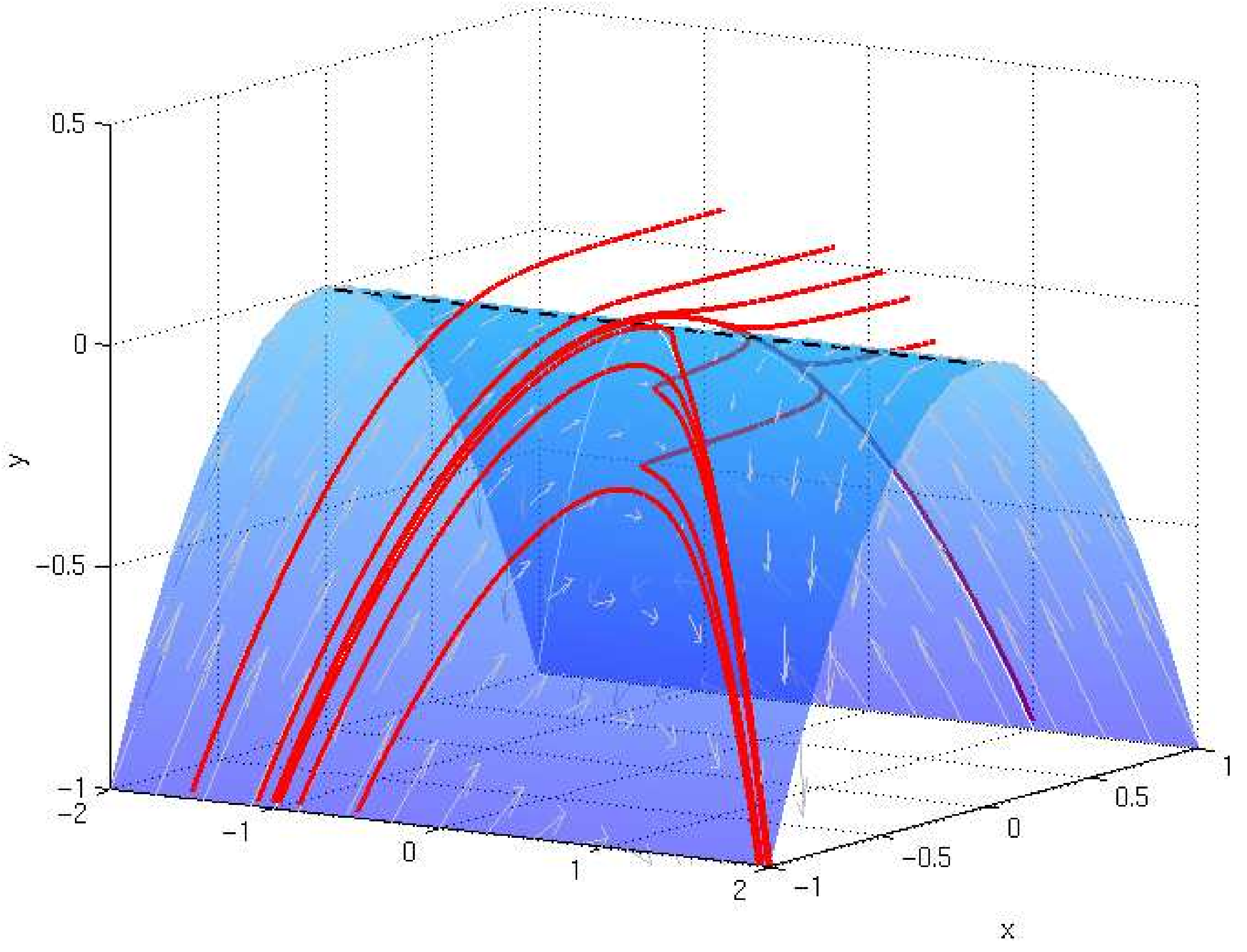
Critical manifold is $y = -x^2$. The origin is a folded equilibrium.

$$a < 0 \implies \text{folded saddle}$$

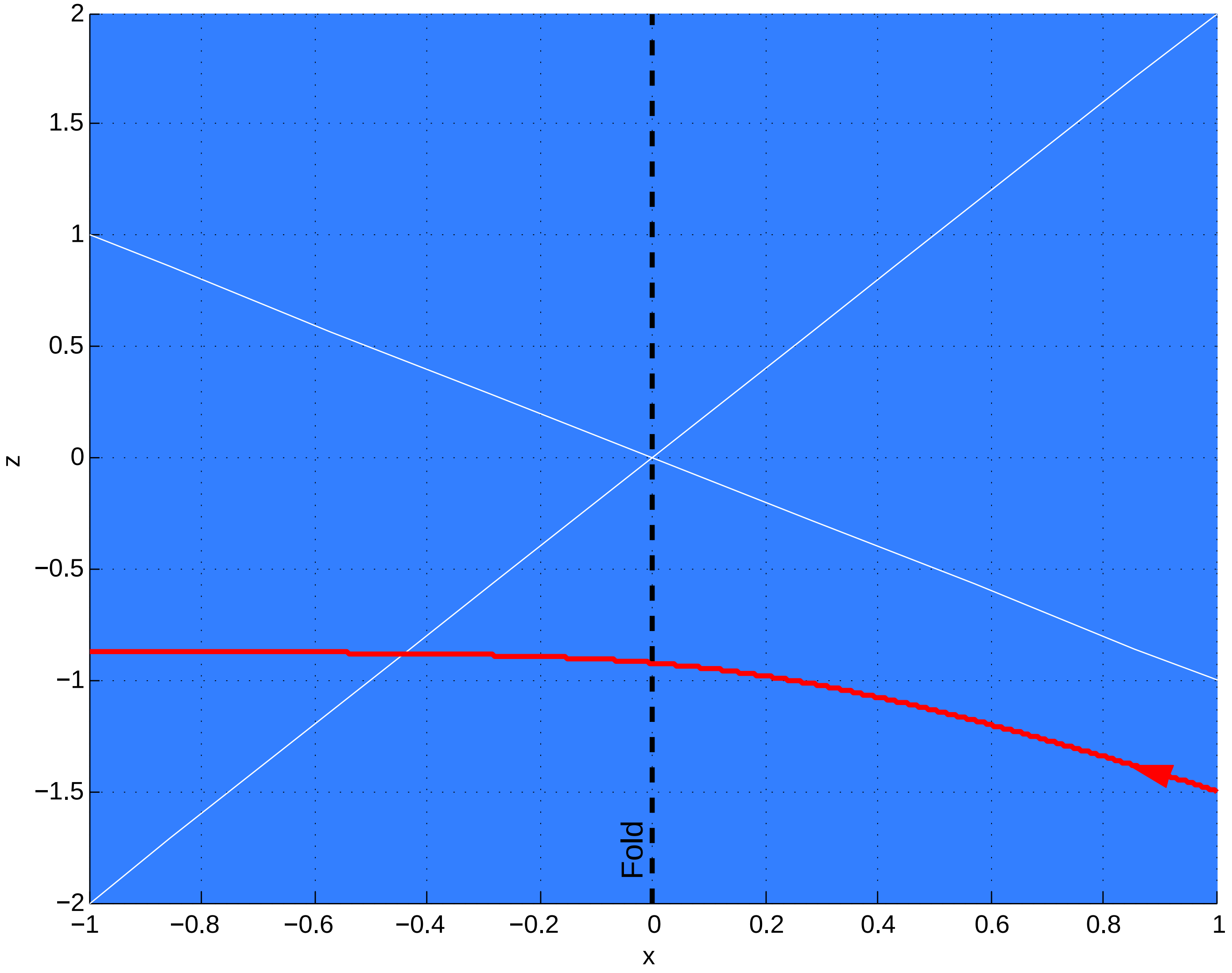
Canards at a Folded Saddle



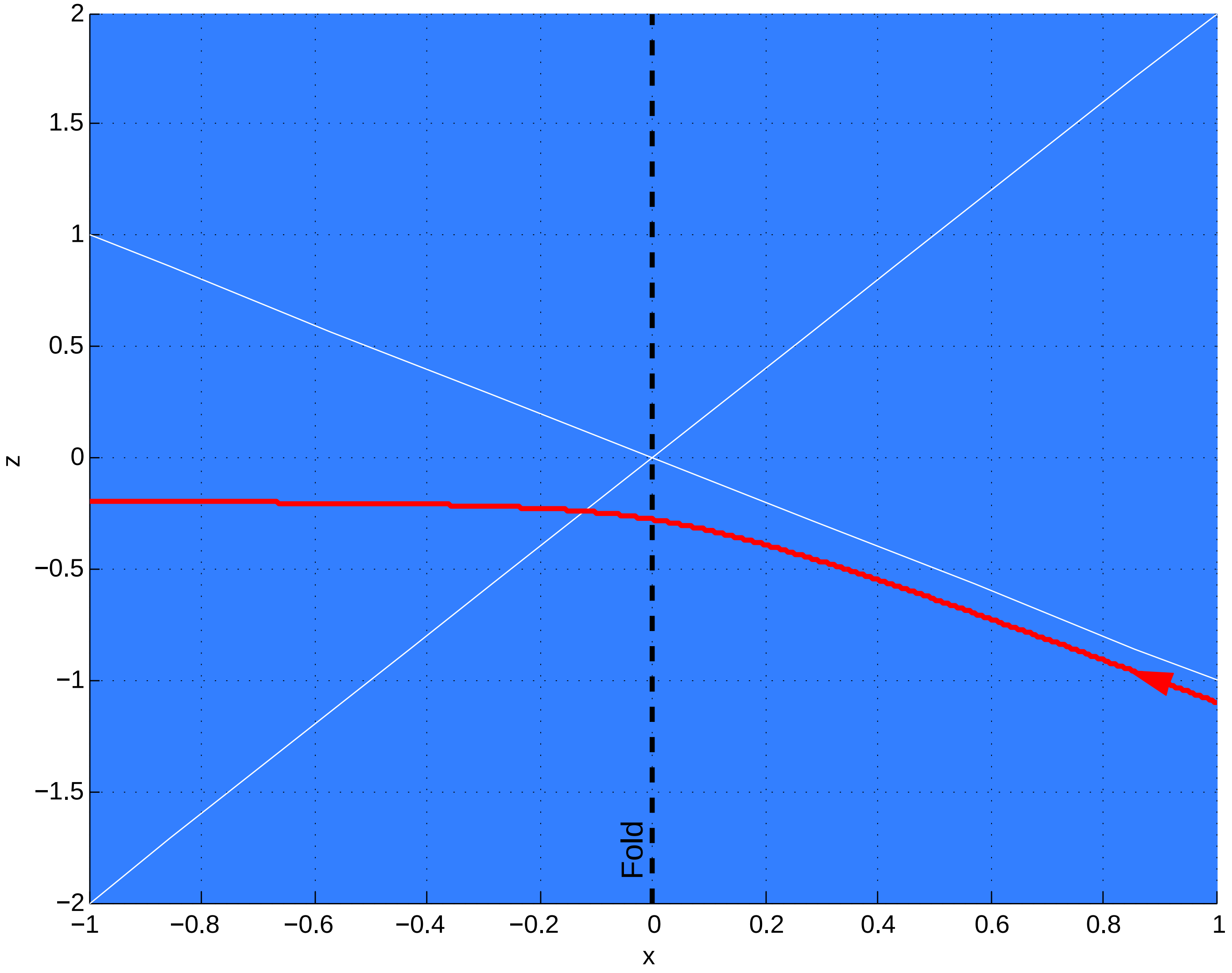
Canards at a Folded Saddle



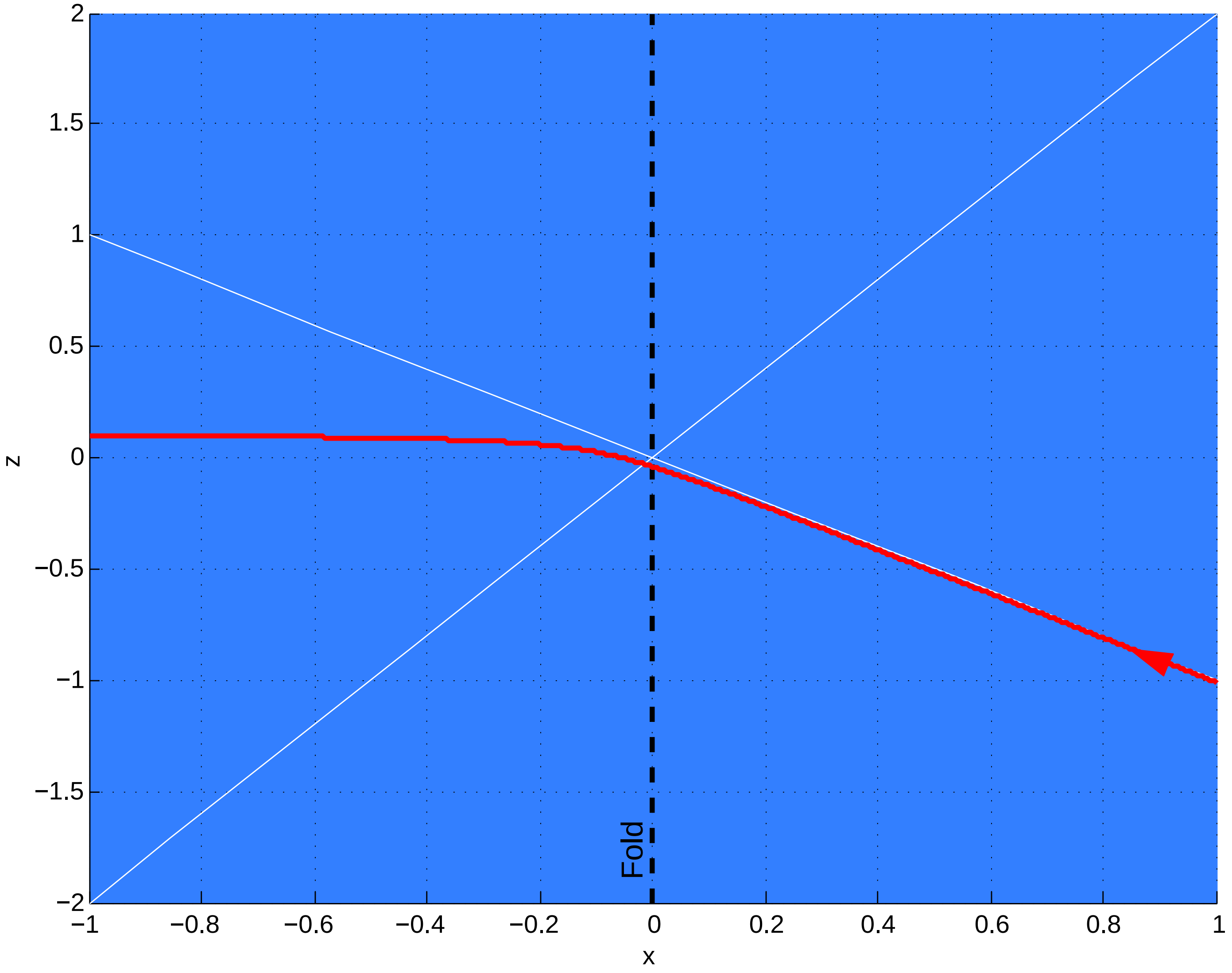
Canards at a Folded Saddle



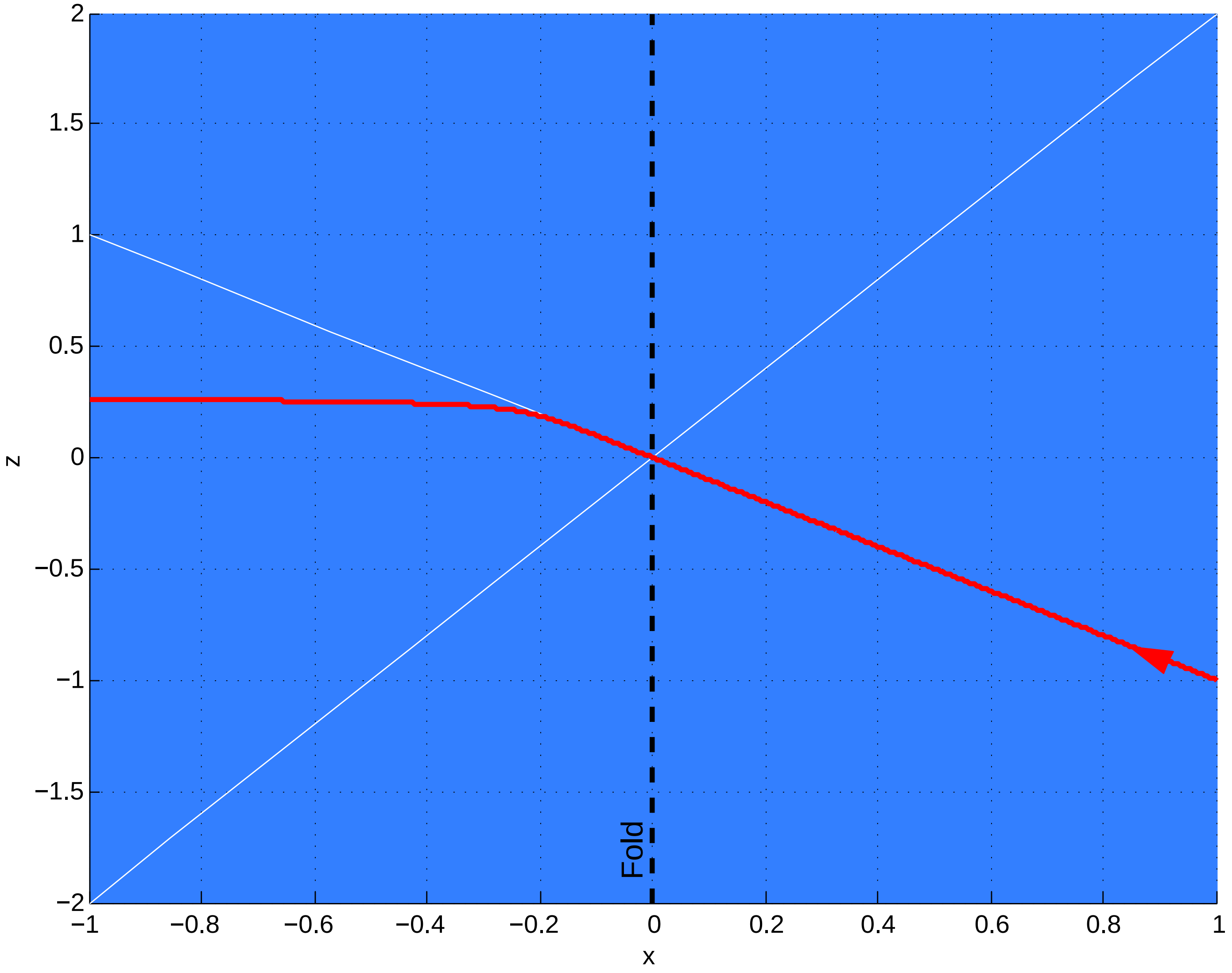
Canards at a Folded Saddle



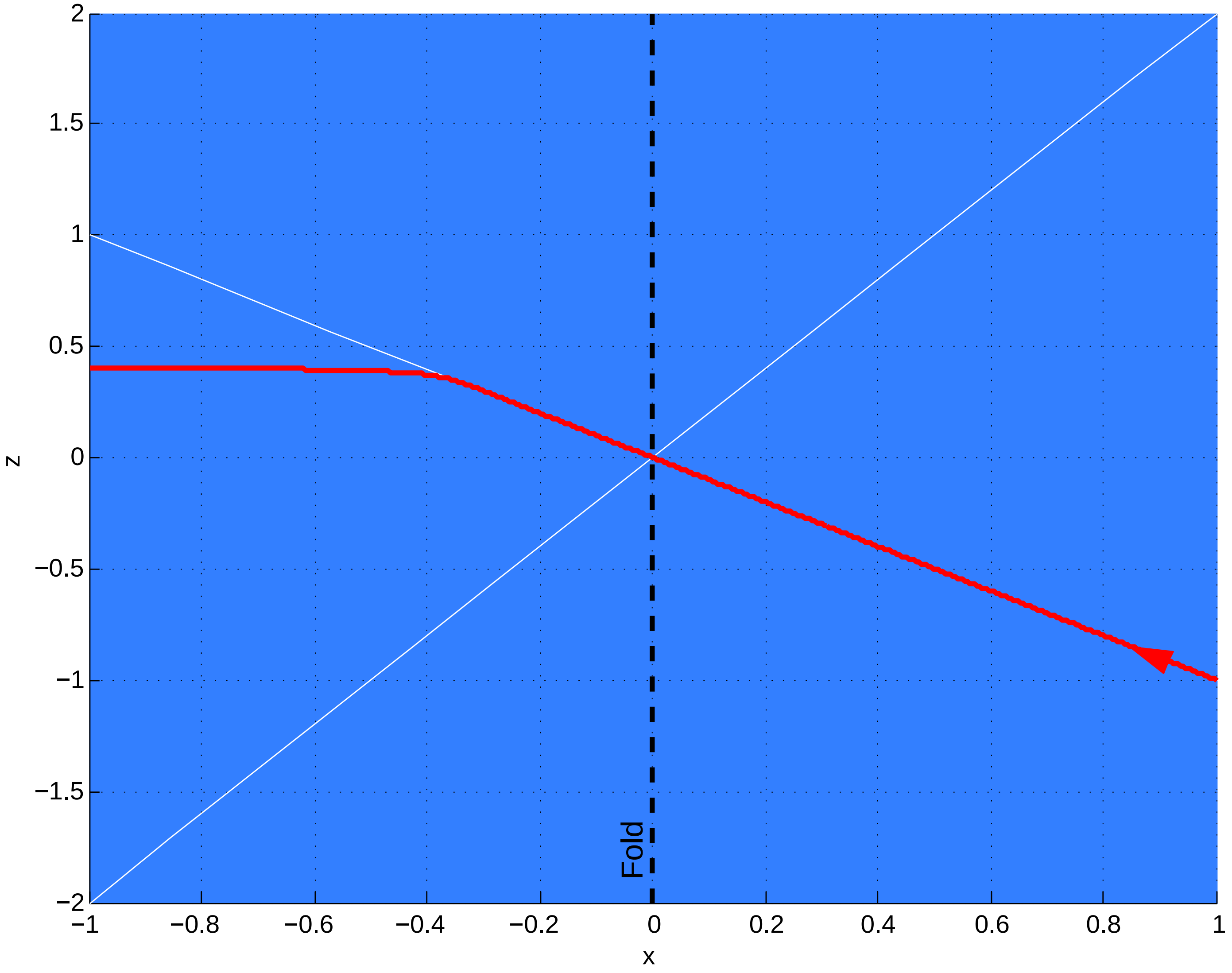
Canards at a Folded Saddle



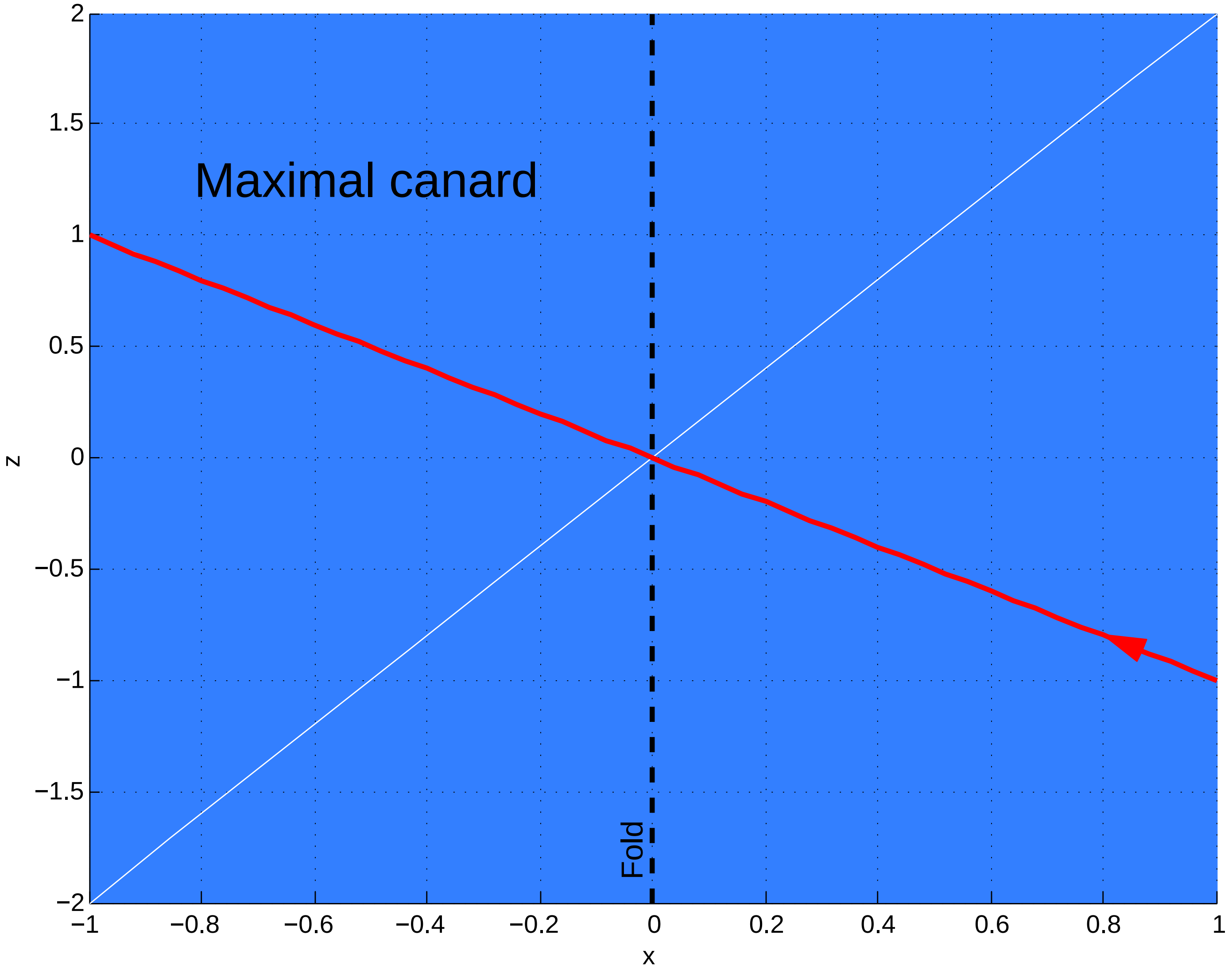
Canards at a Folded Saddle



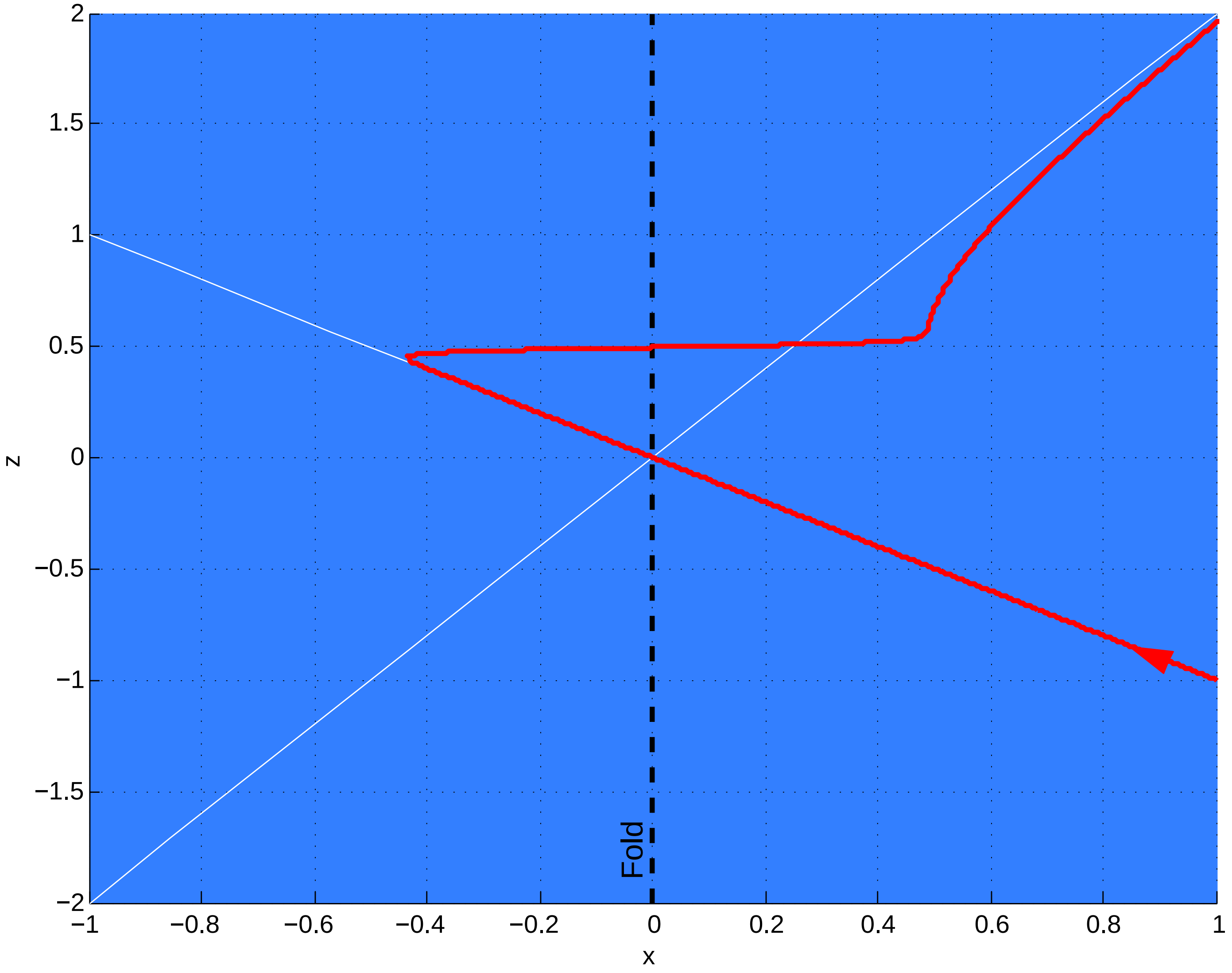
Canards at a Folded Saddle



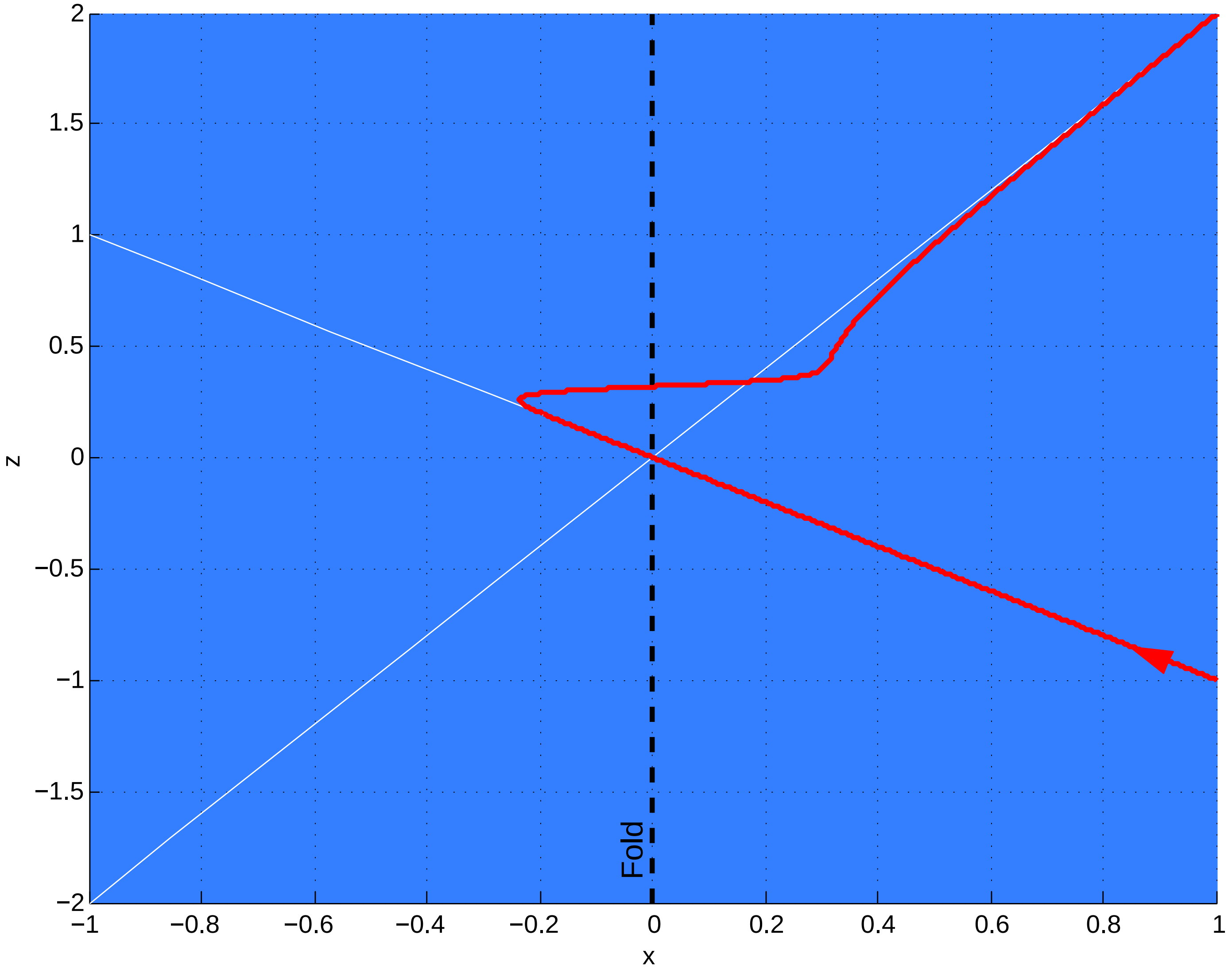
Canards at a Folded Saddle



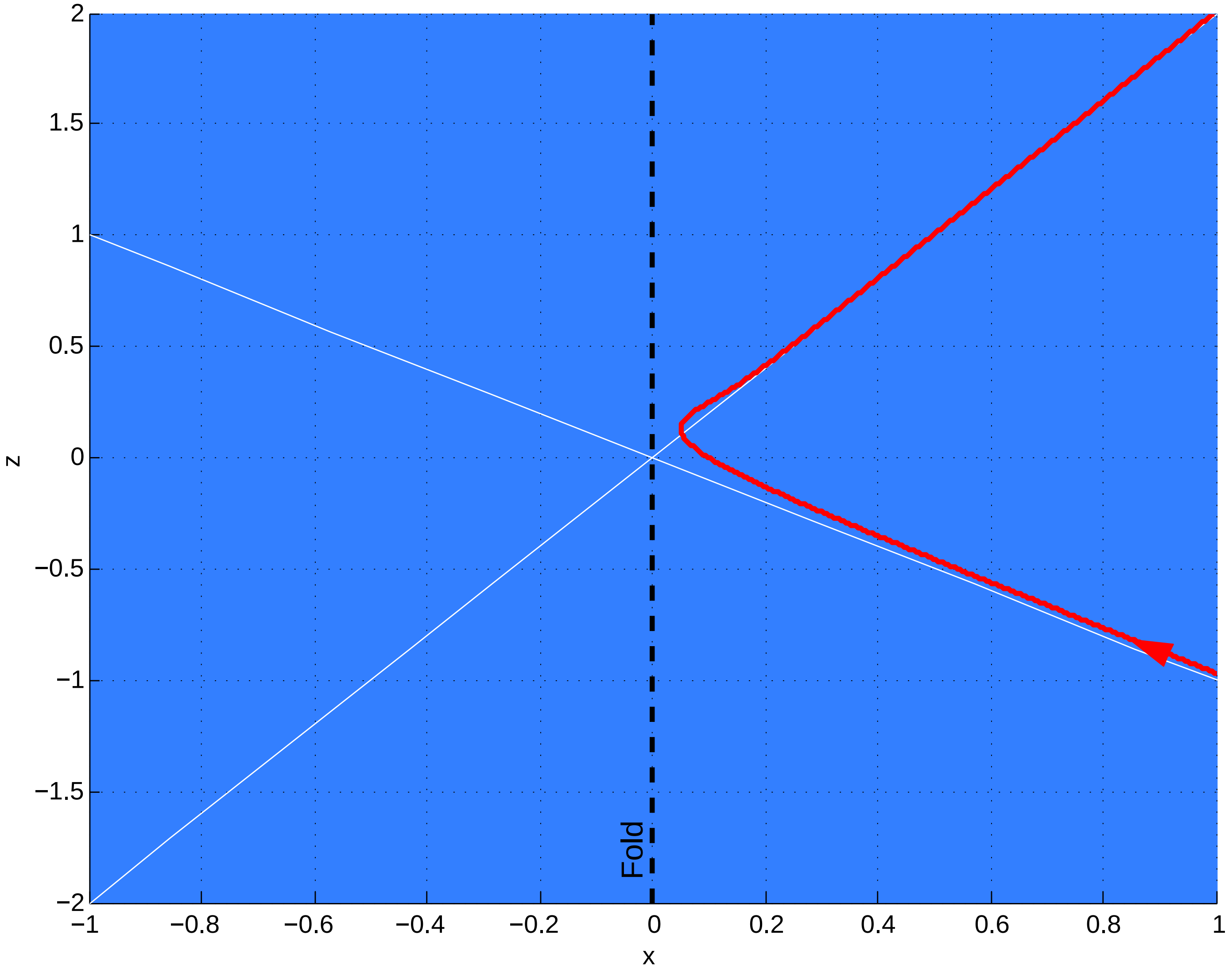
Canards at a Folded Saddle



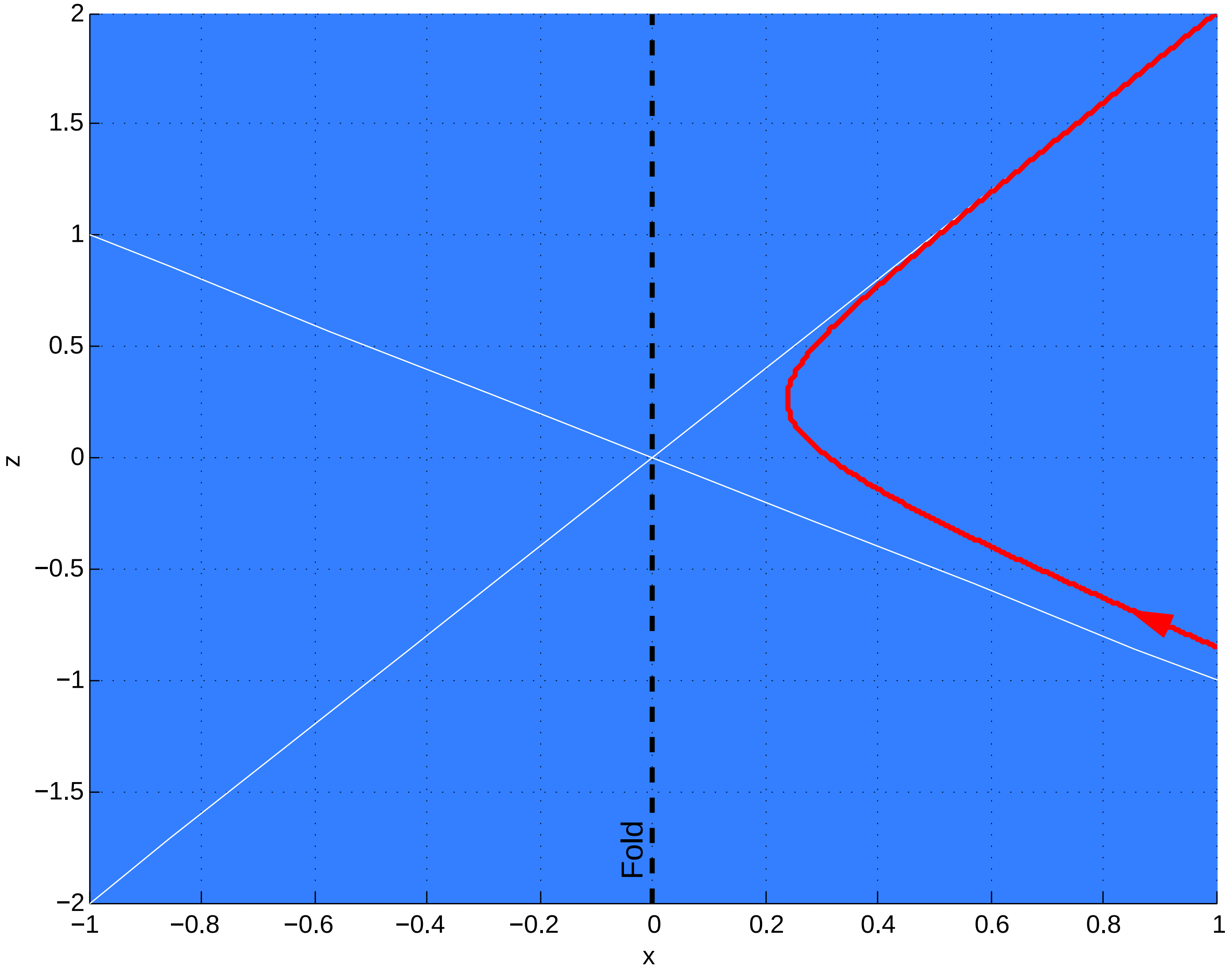
Canards at a Folded Saddle



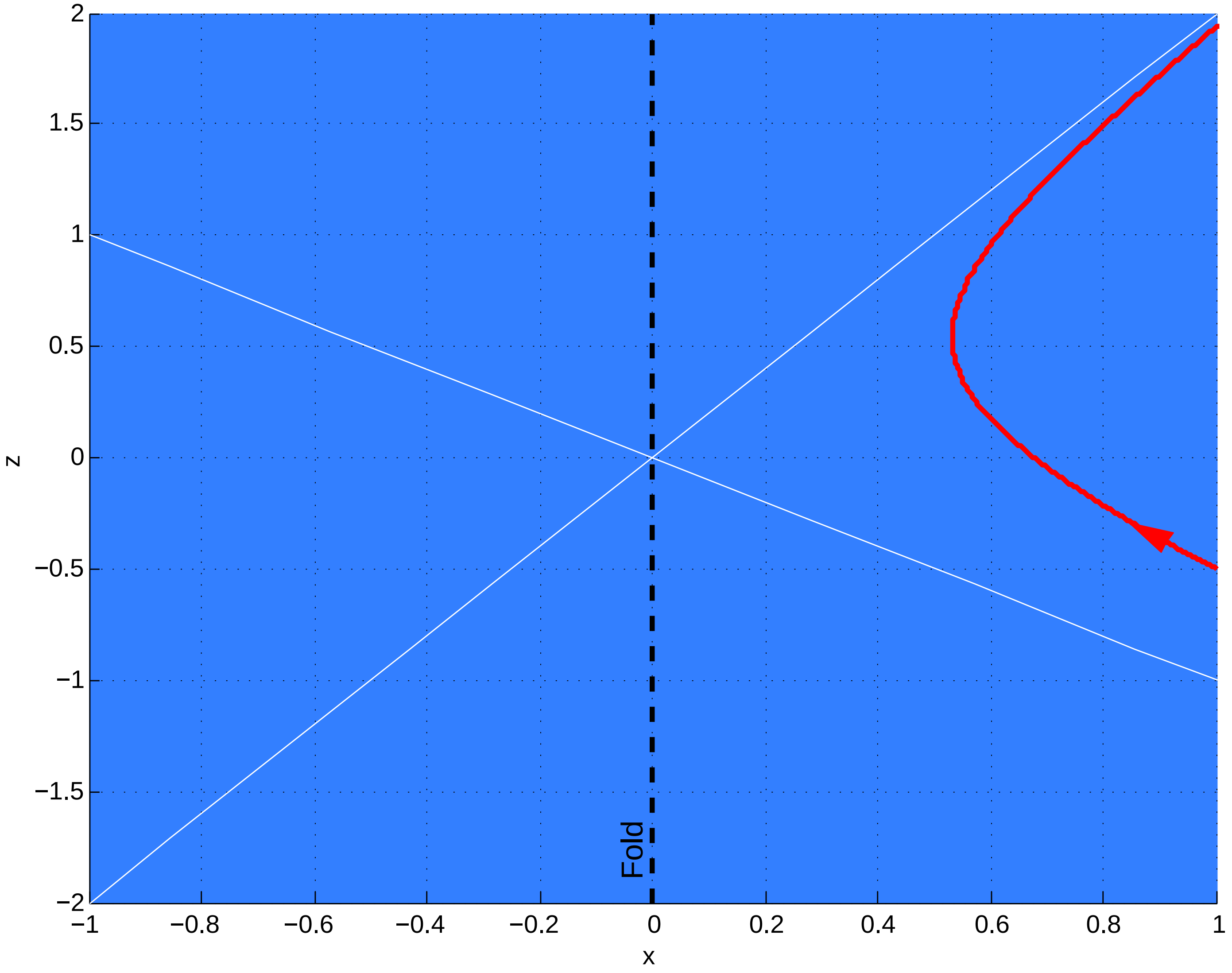
Canards at a Folded Saddle



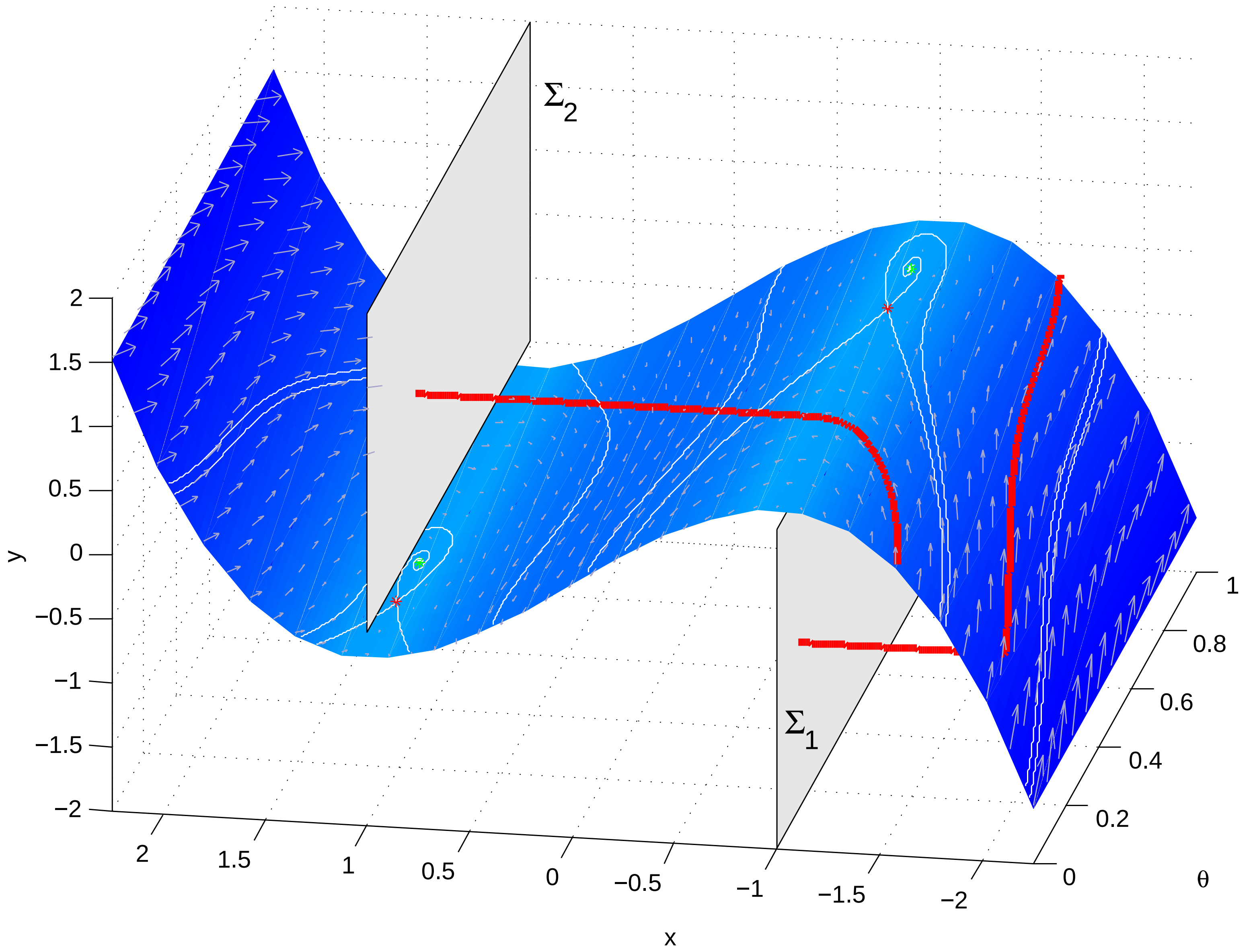
Canards at a Folded Saddle



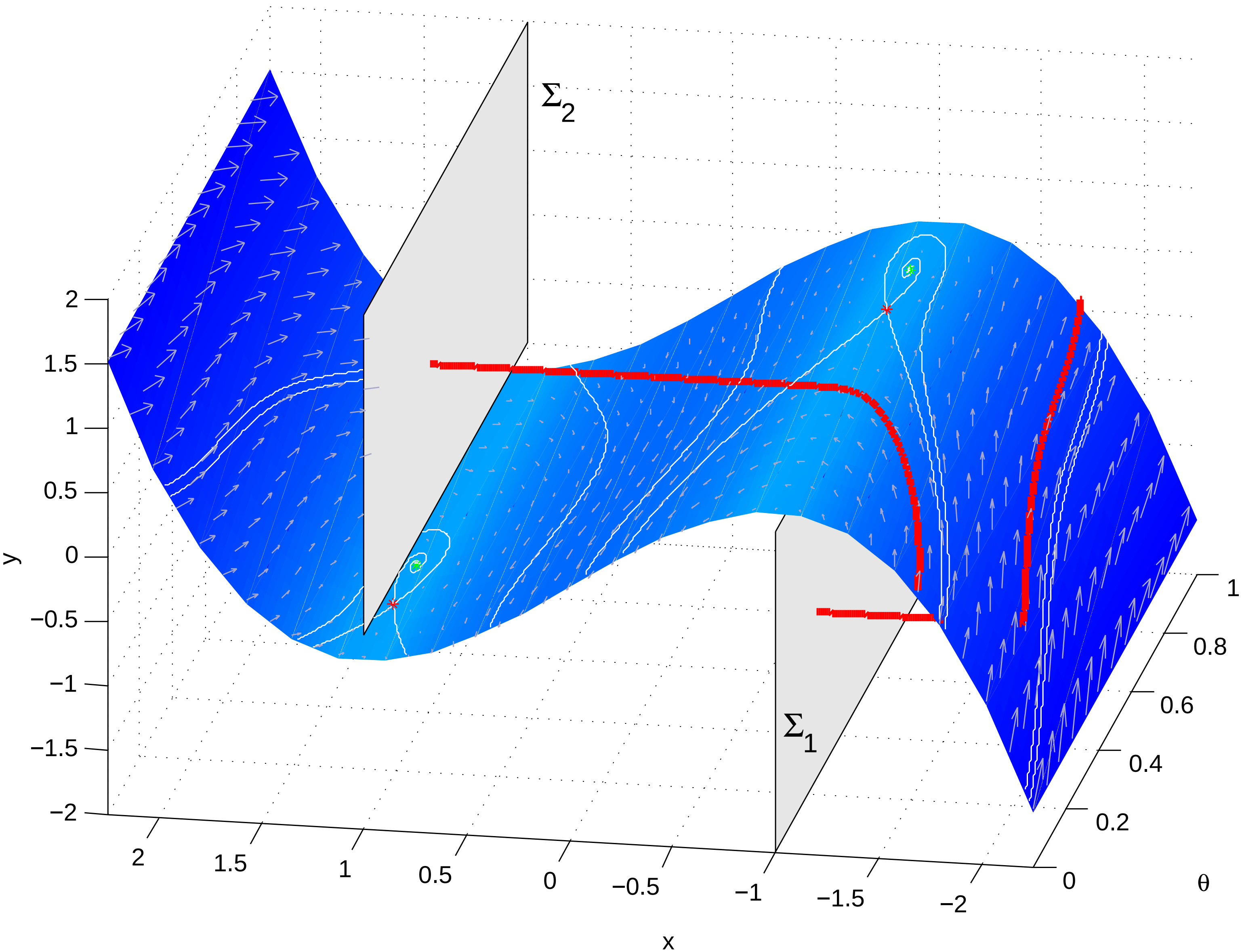
Canards at a Folded Saddle



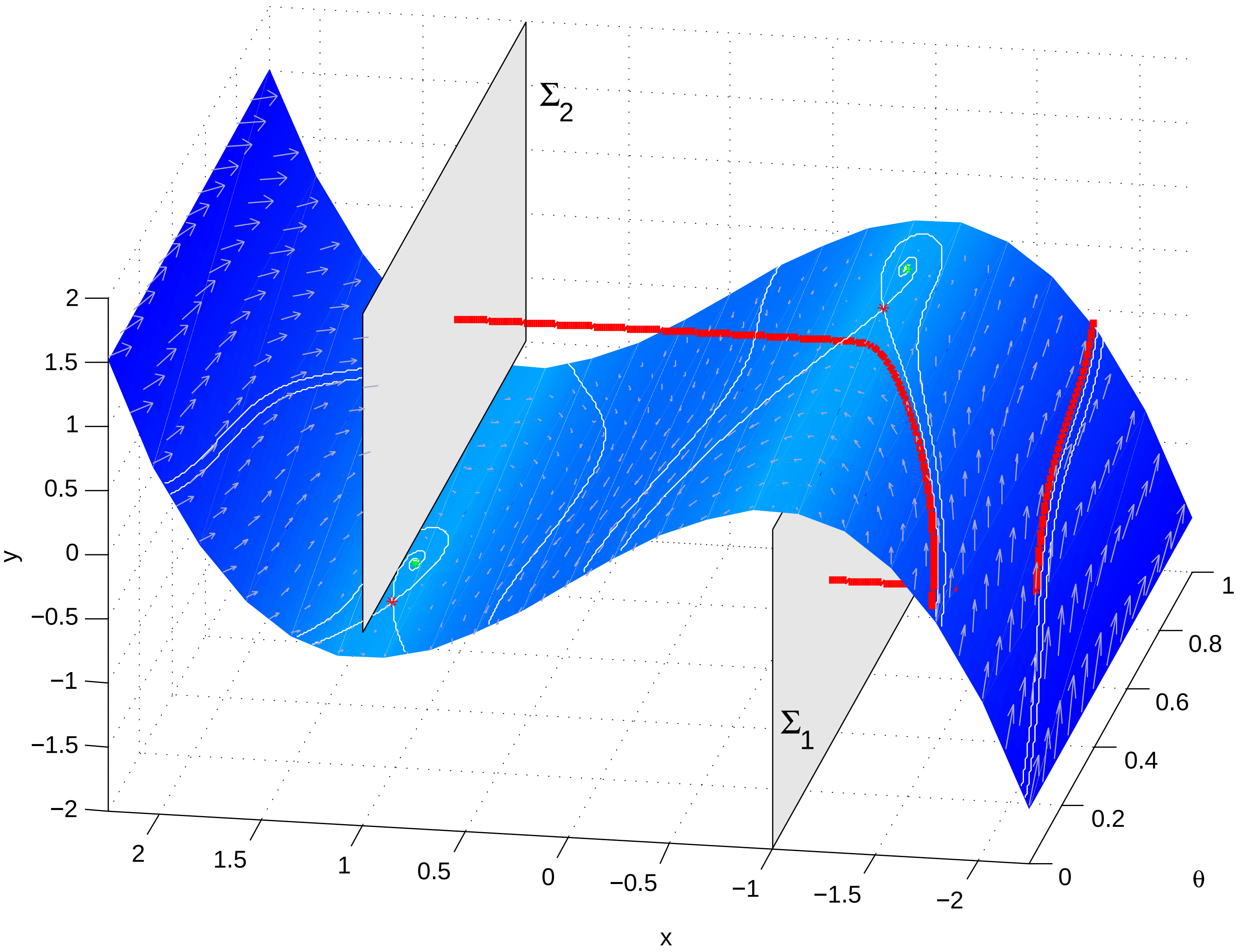
Forced van der Pol Poincaré Map



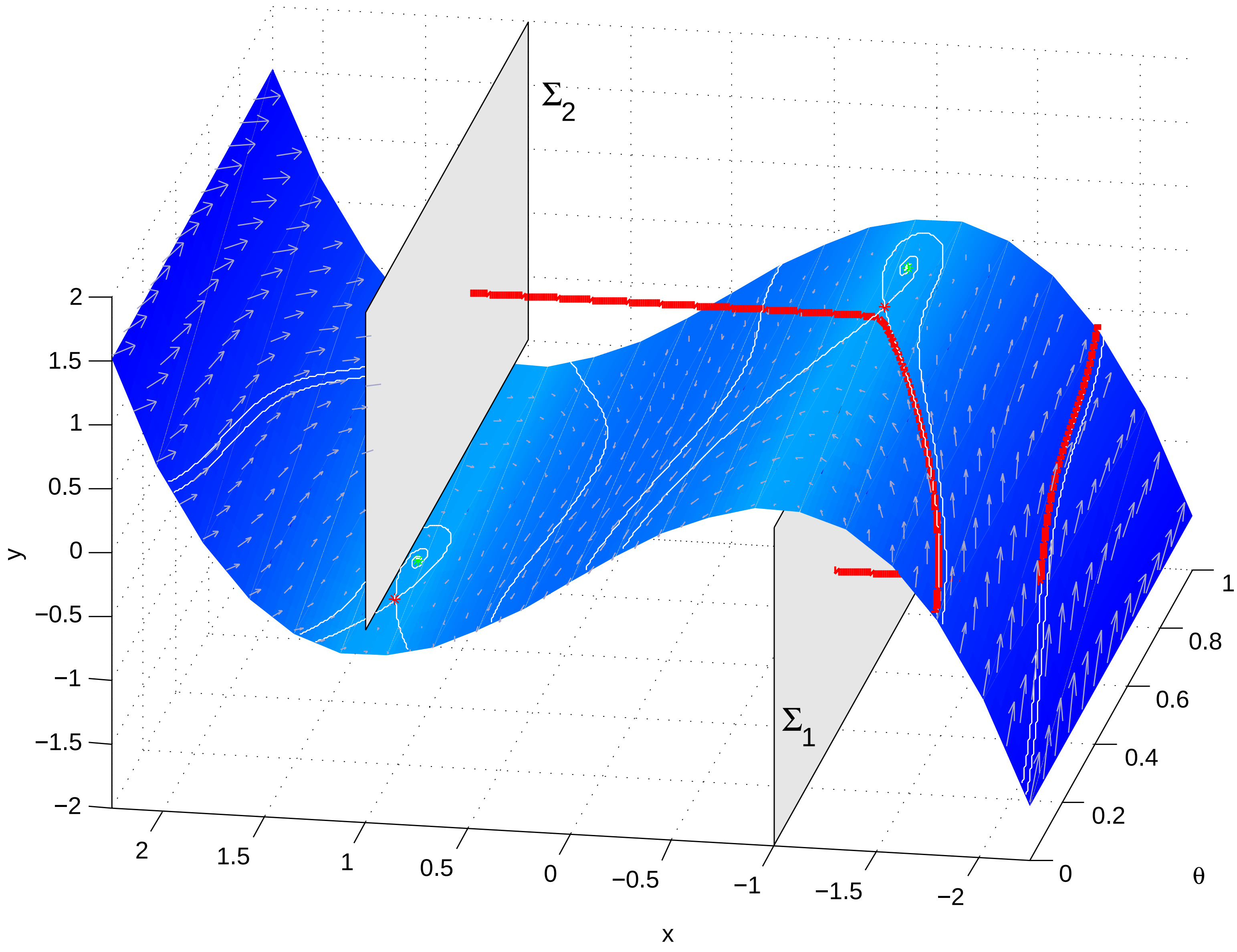
Forced van der Pol Poincaré Map



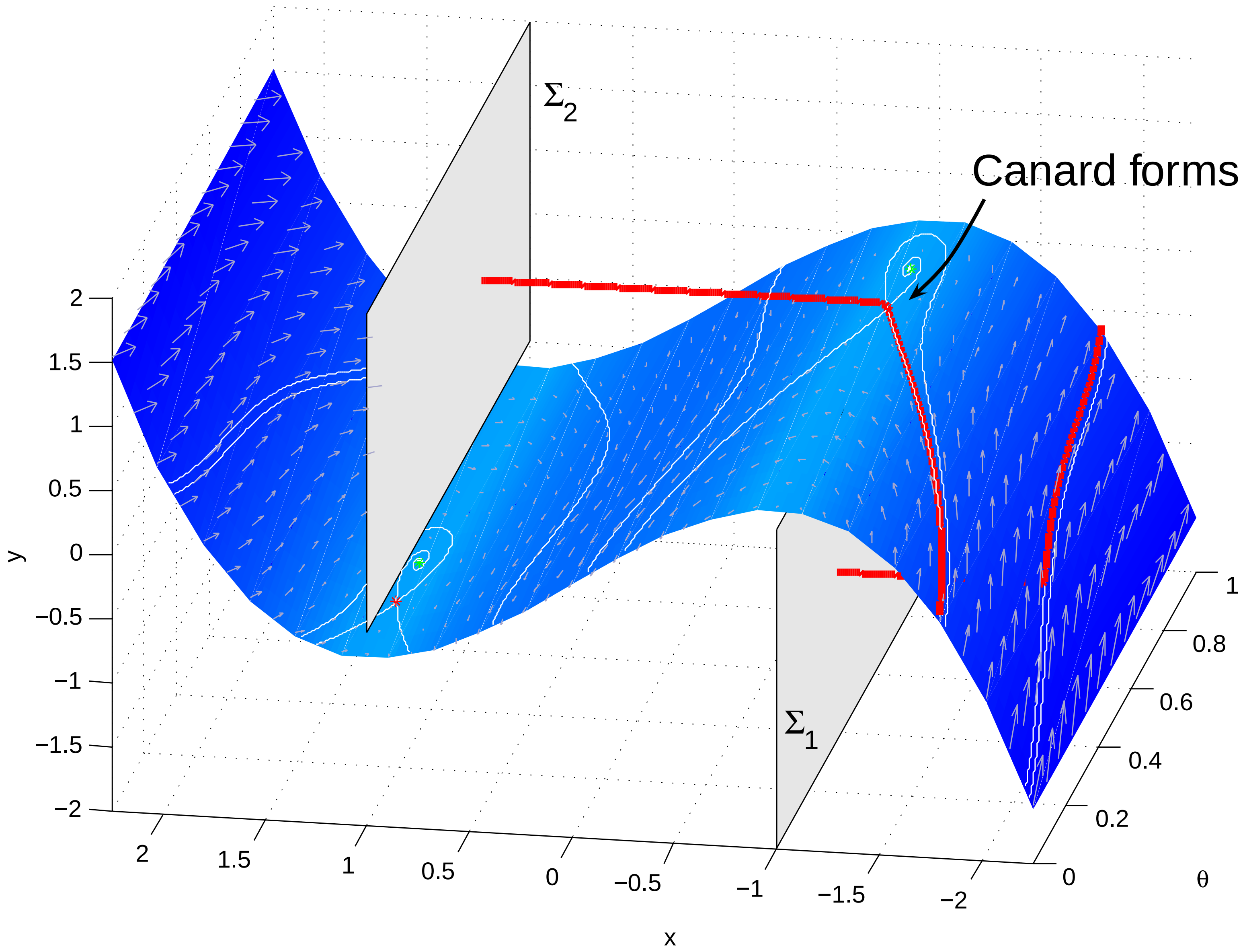
Forced van der Pol Poincaré Map



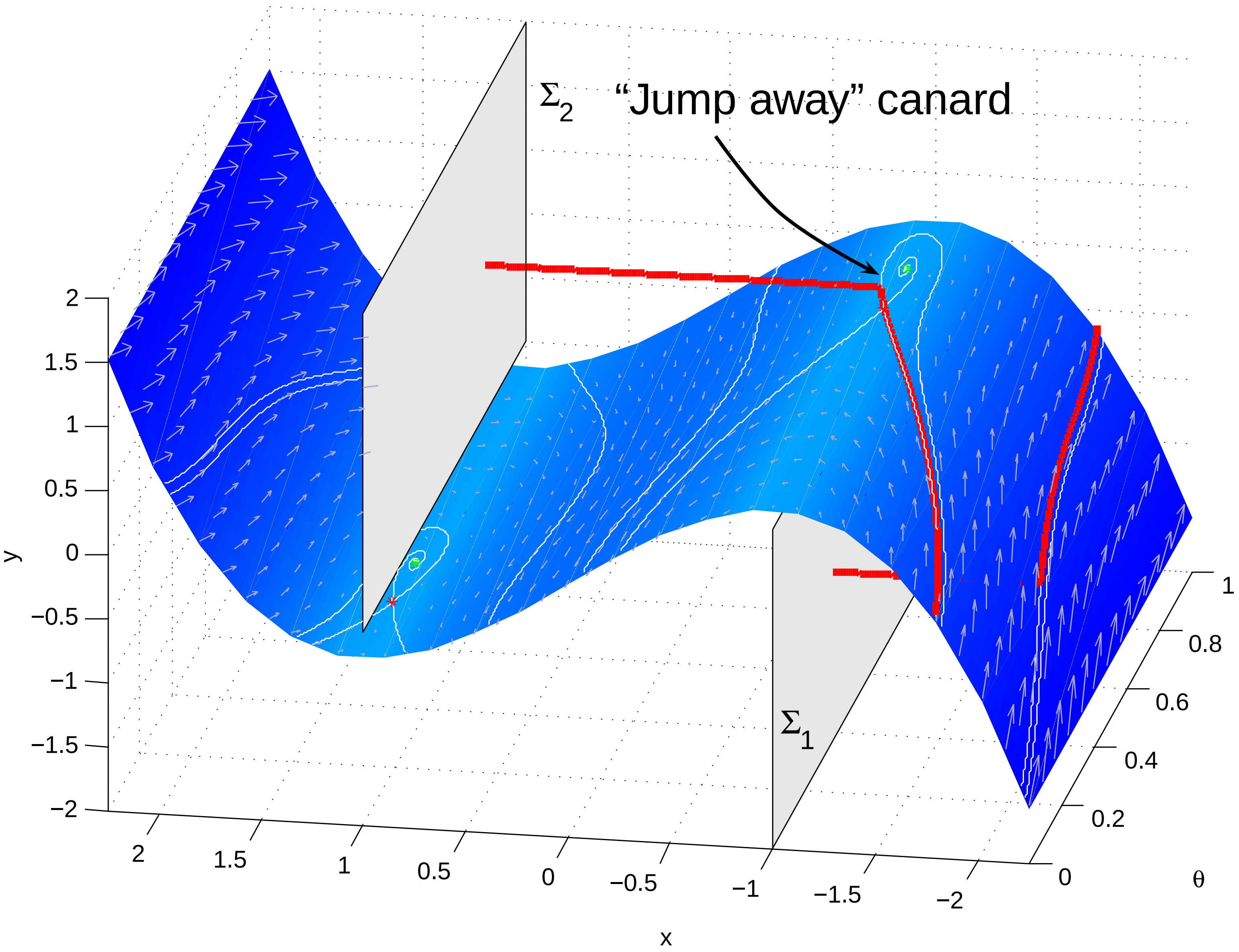
Forced van der Pol Poincaré Map



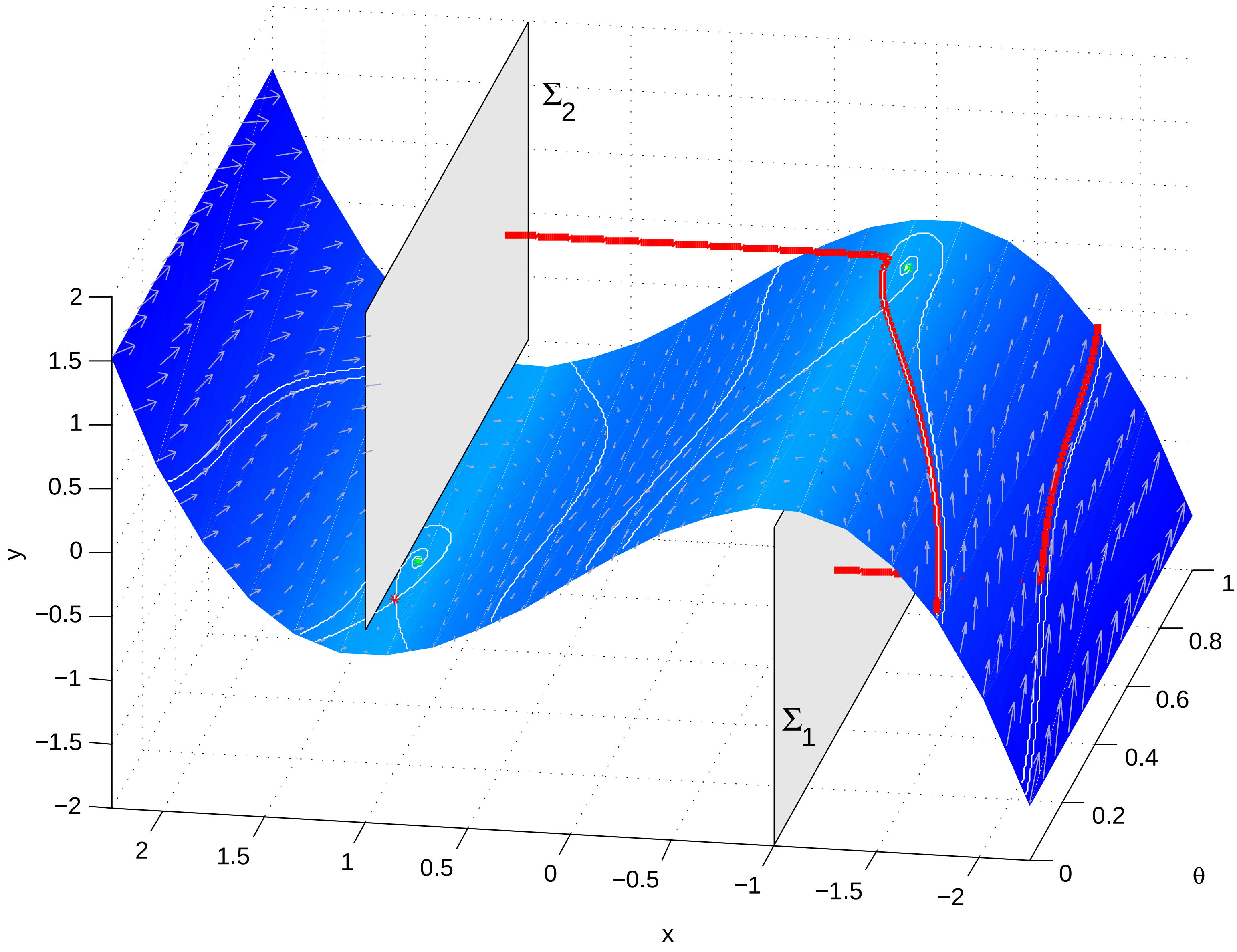
Forced van der Pol Poincaré Map



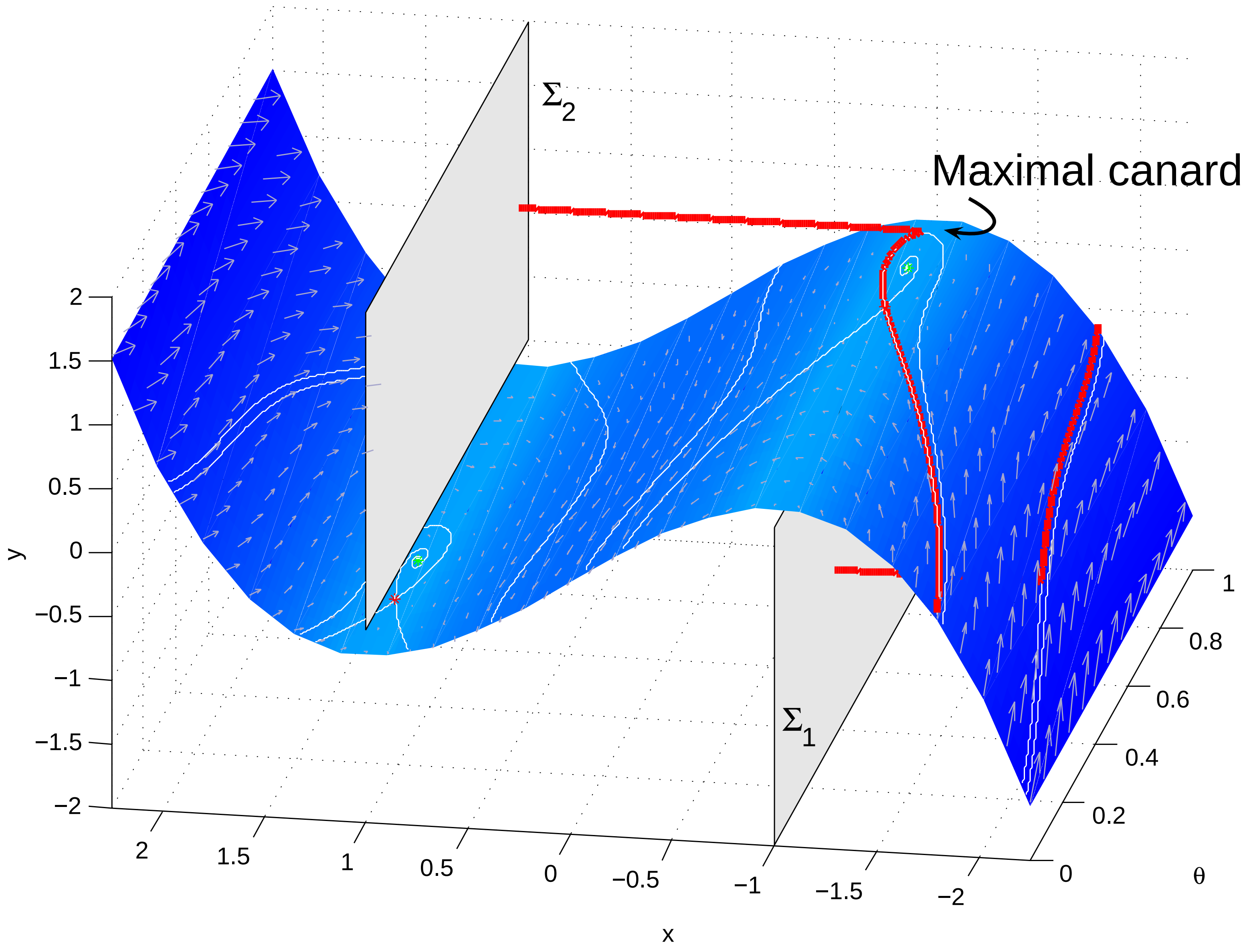
Forced van der Pol Poincaré Map



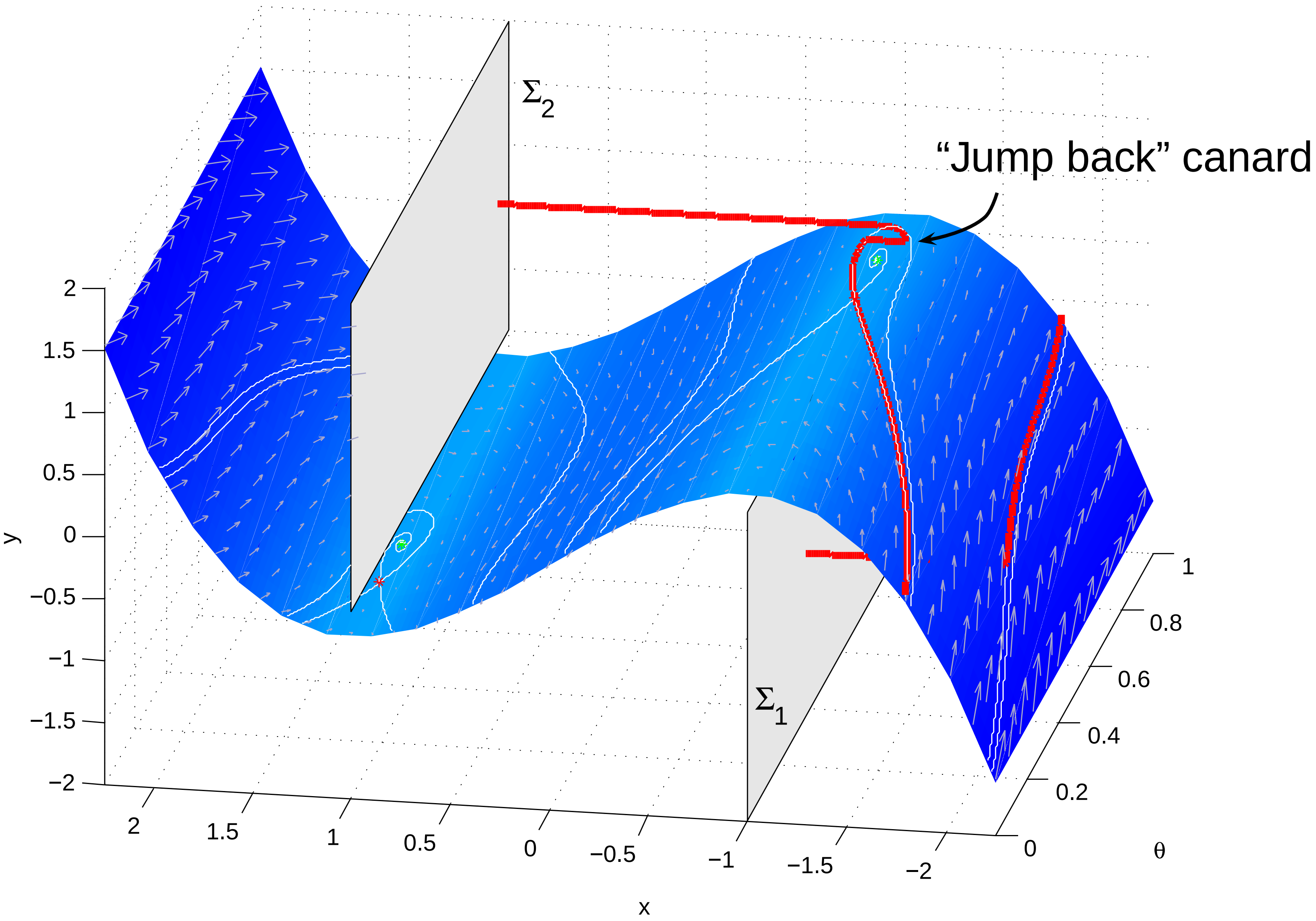
Forced van der Pol Poincaré Map



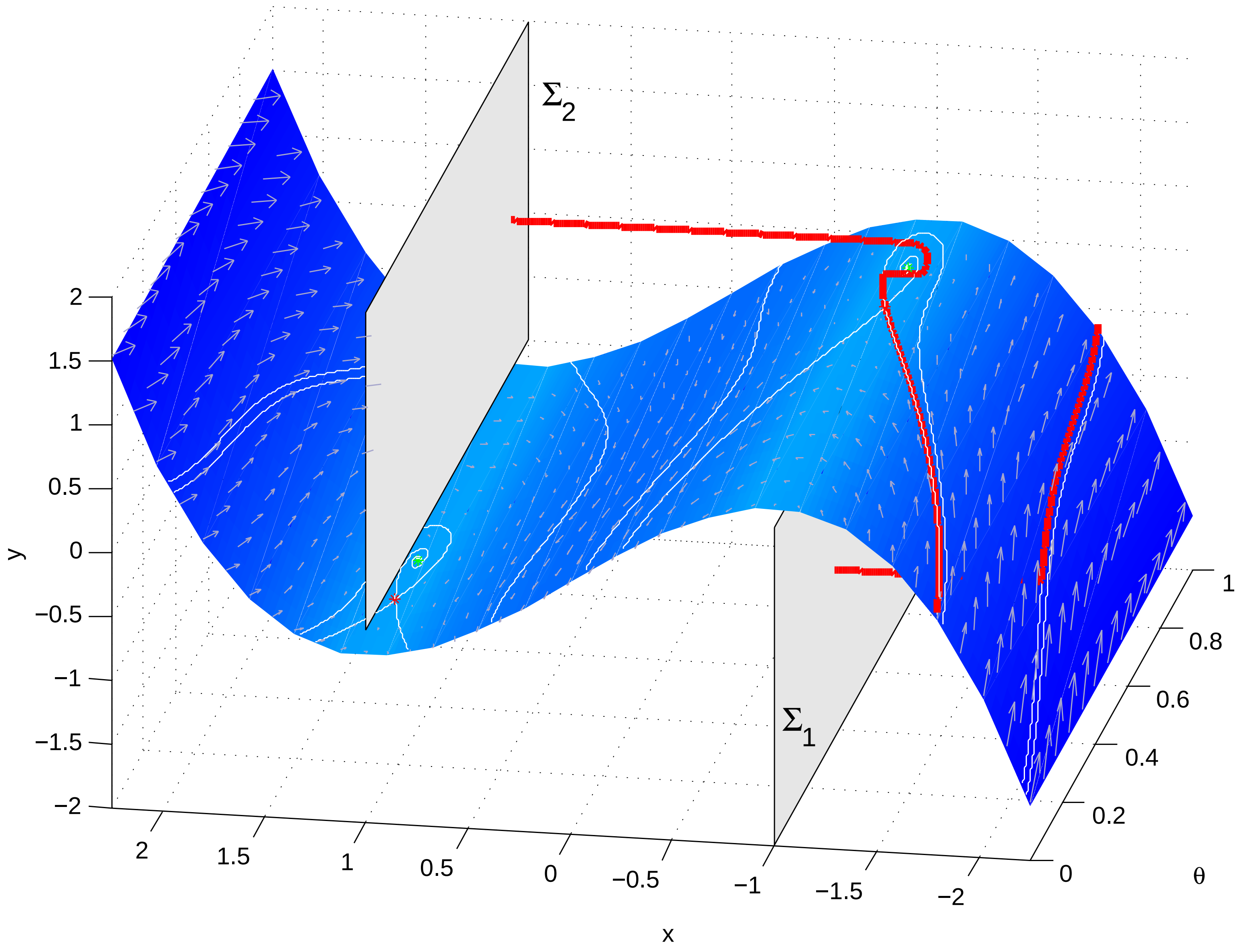
Forced van der Pol Poincaré Map



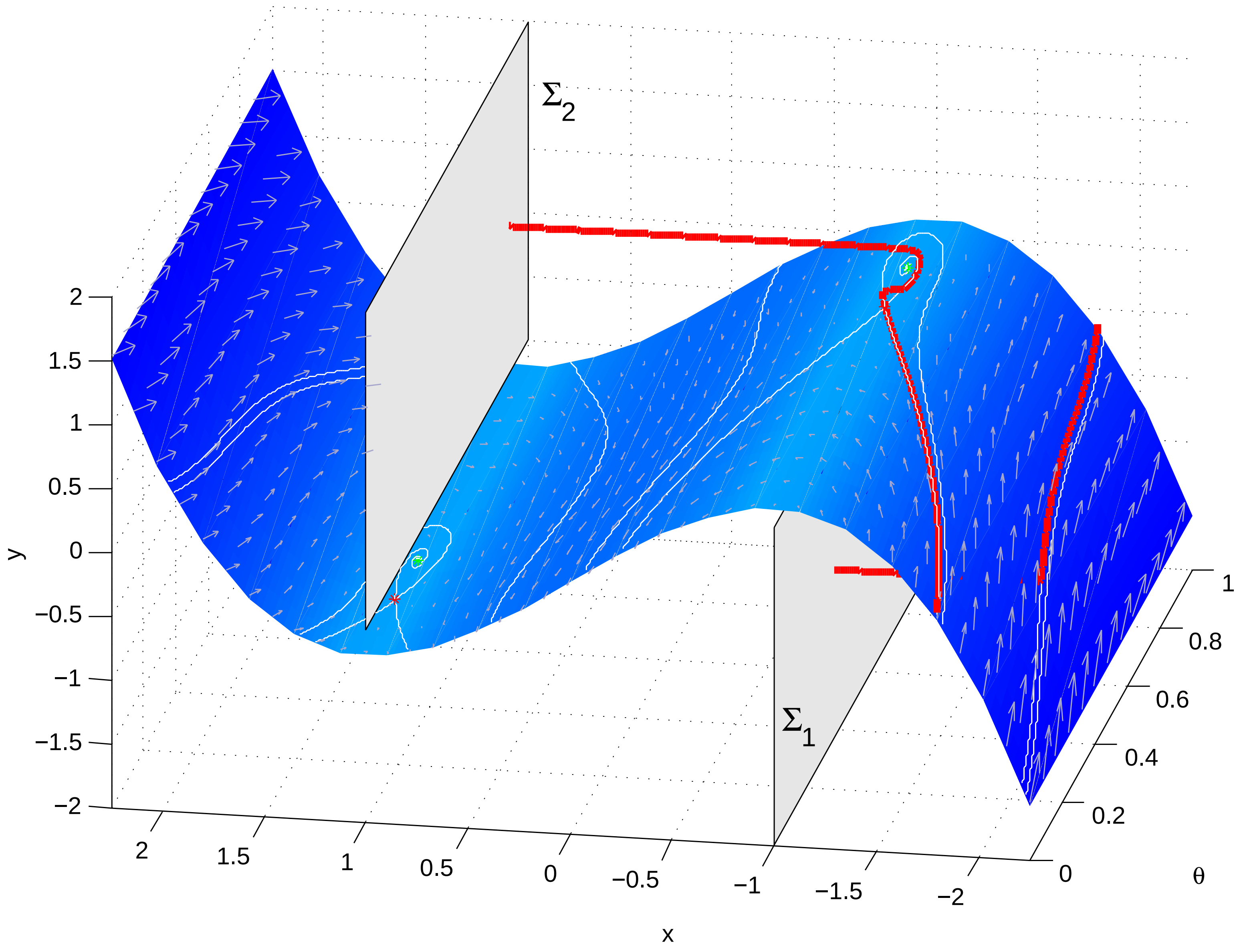
Forced van der Pol Poincaré Map



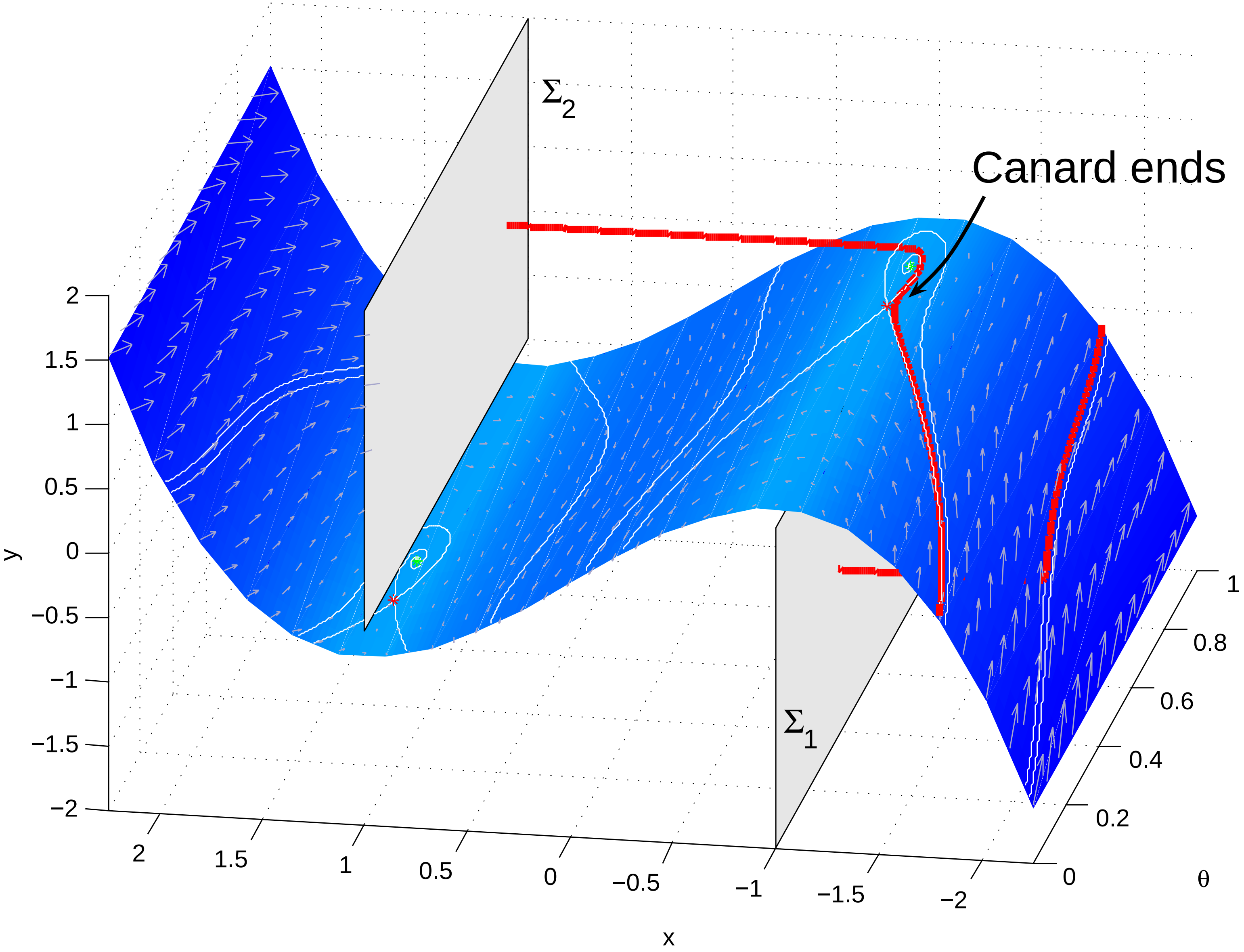
Forced van der Pol Poincaré Map



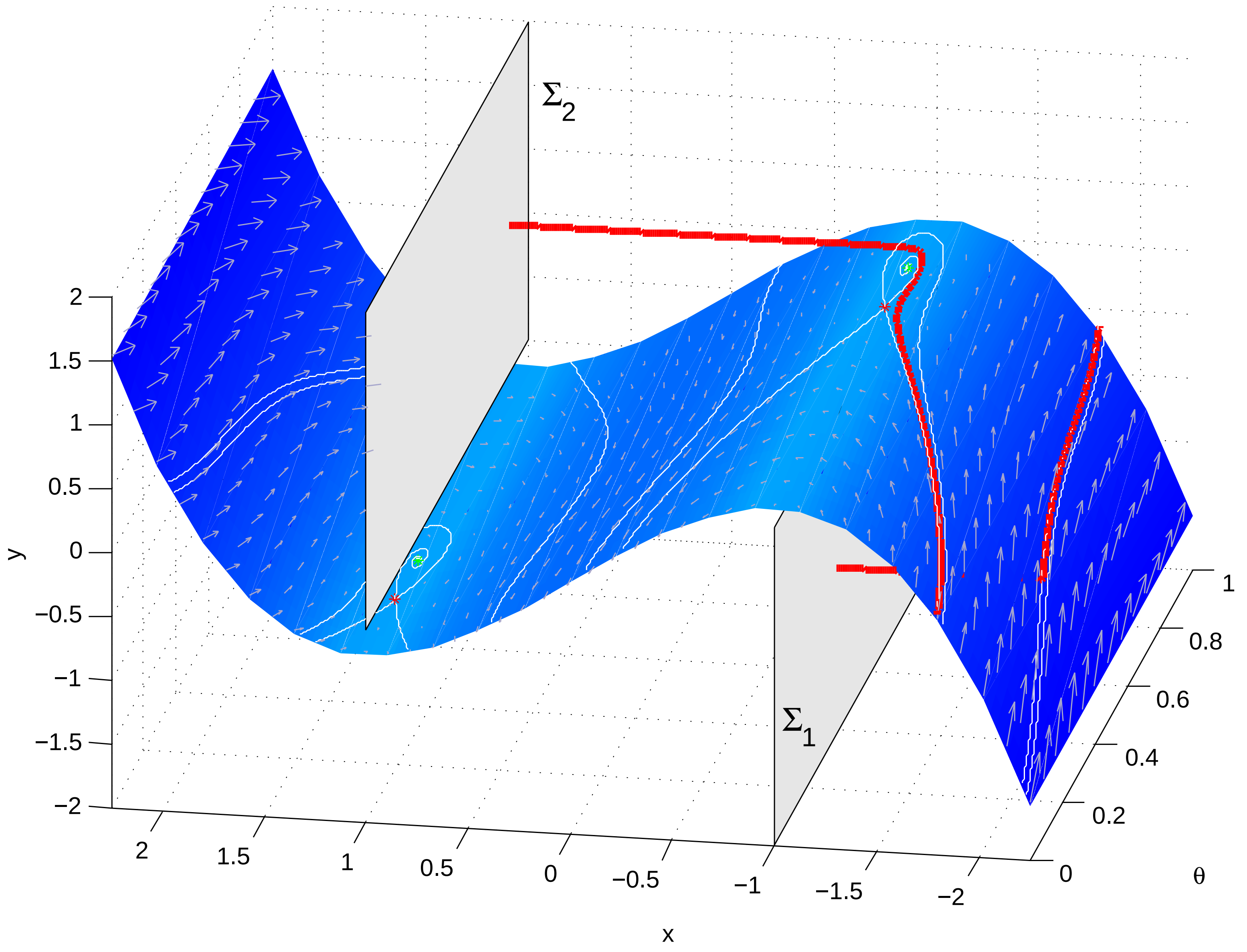
Forced van der Pol Poincaré Map



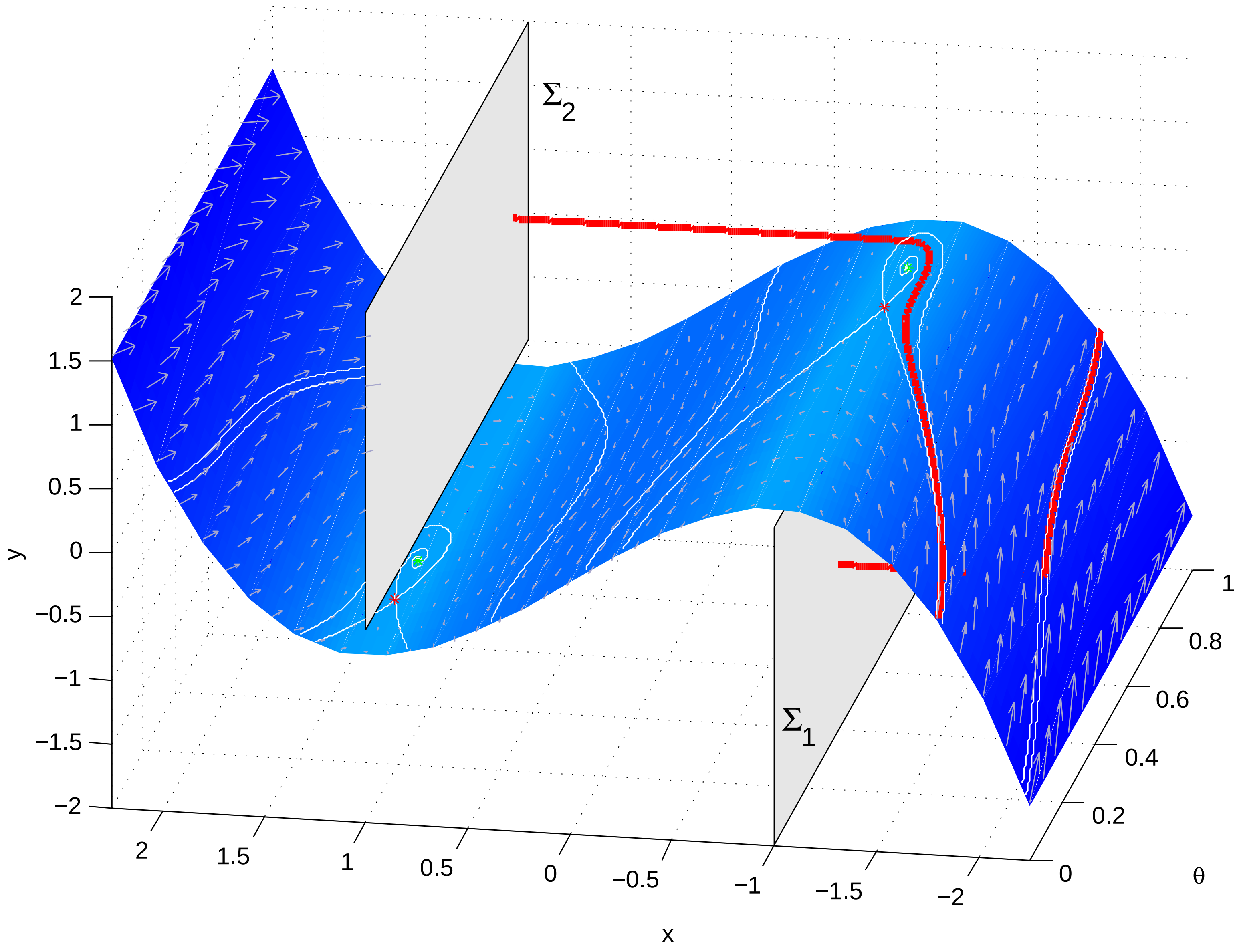
Forced van der Pol Poincaré Map



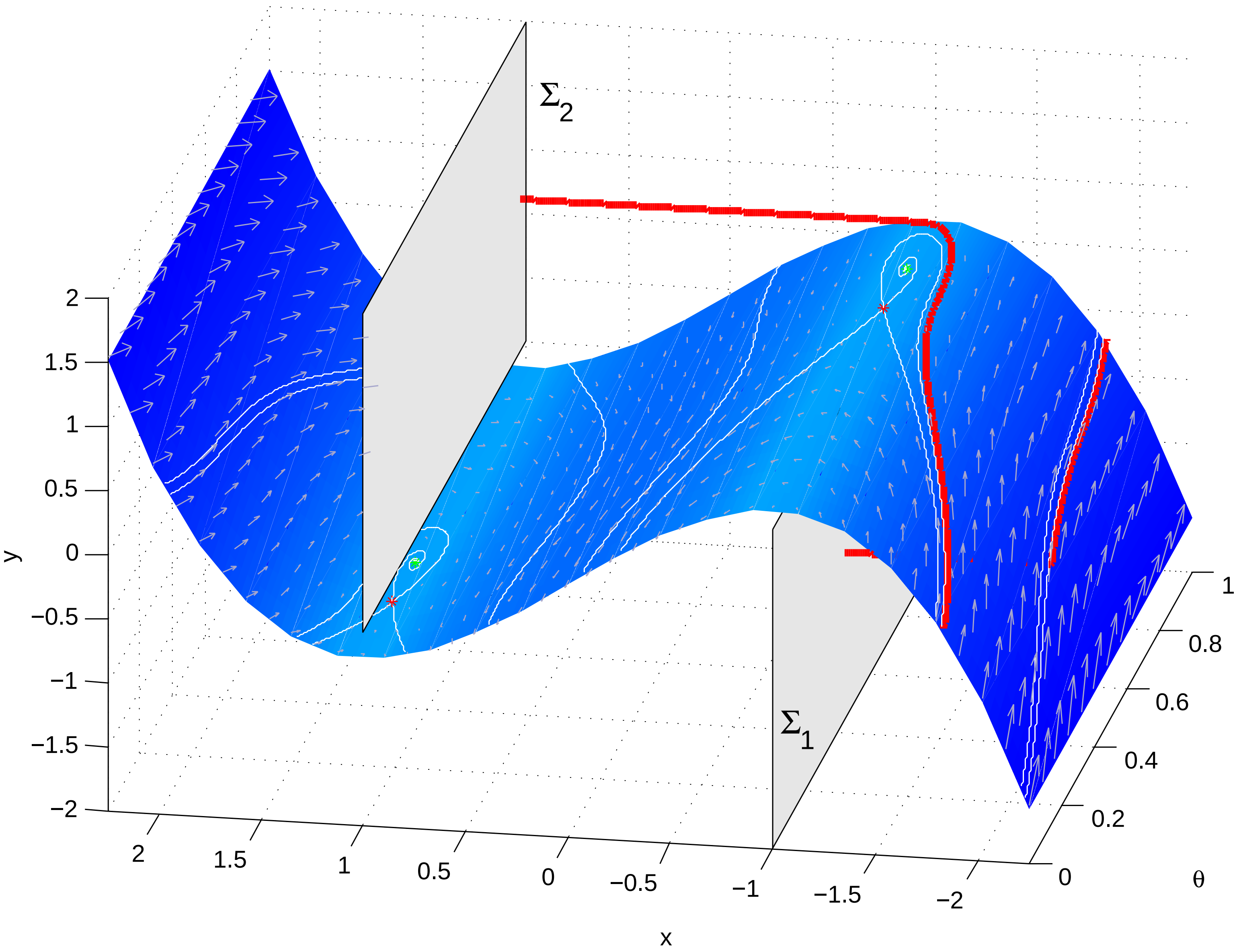
Forced van der Pol Poincaré Map



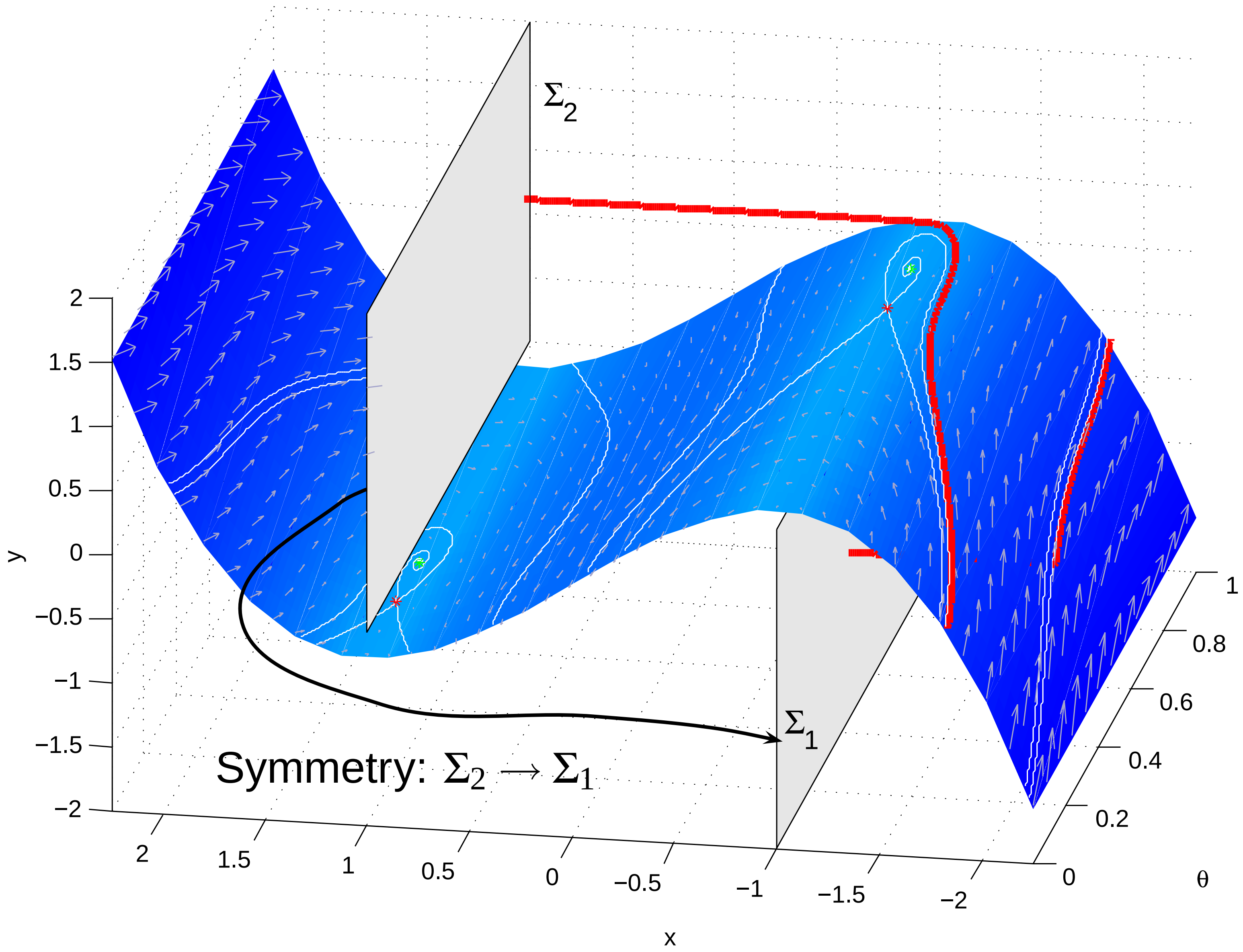
Forced van der Pol Poincaré Map



Forced van der Pol Poincaré Map

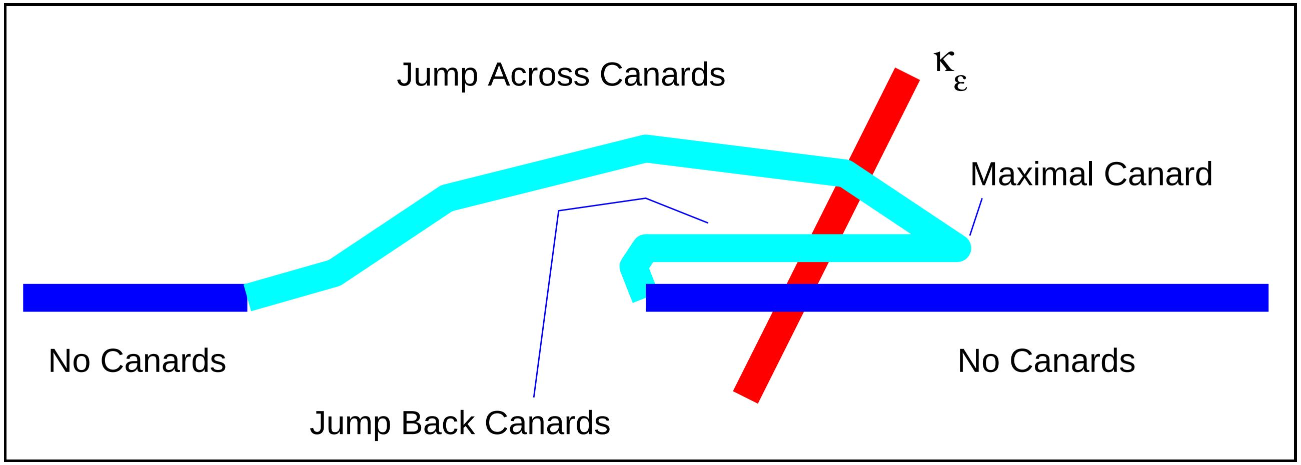


Forced van der Pol Poincaré Map

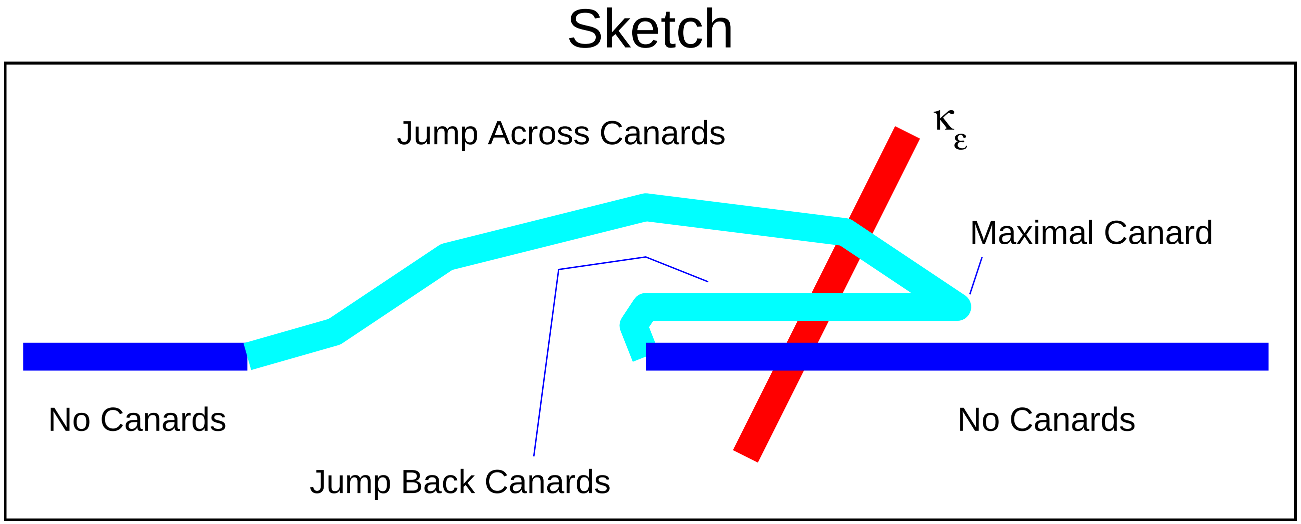


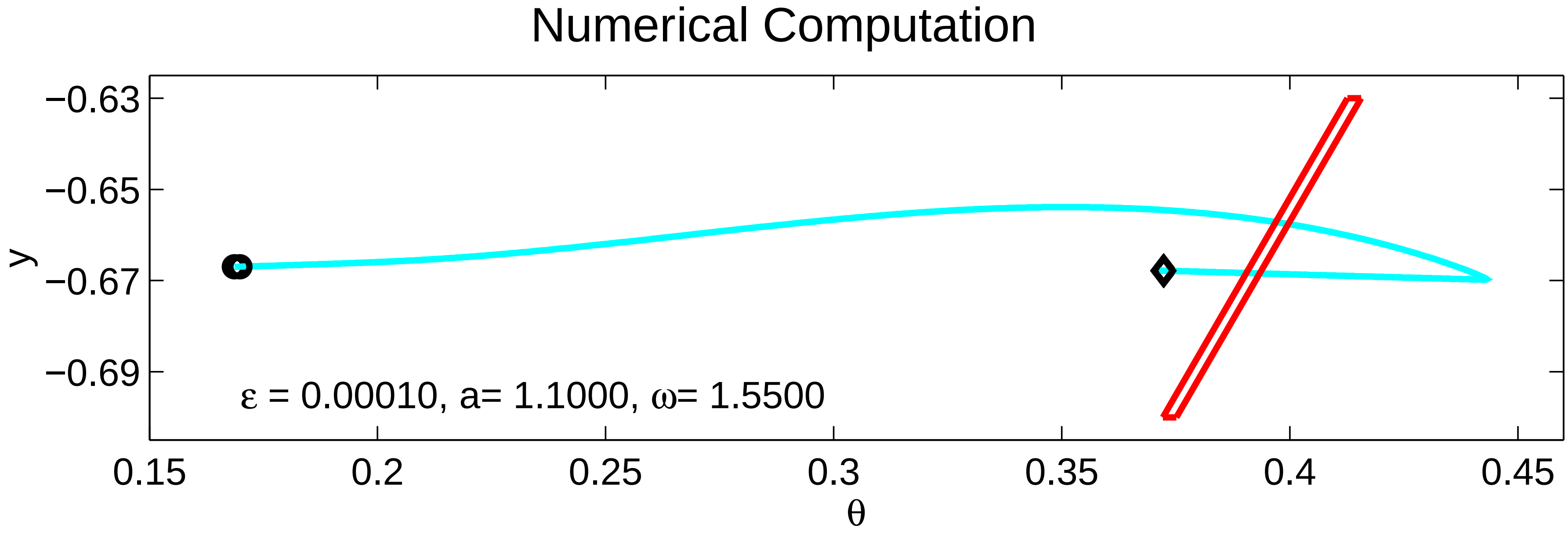
Horseshoe in the Forced van der Pol System

Sketch

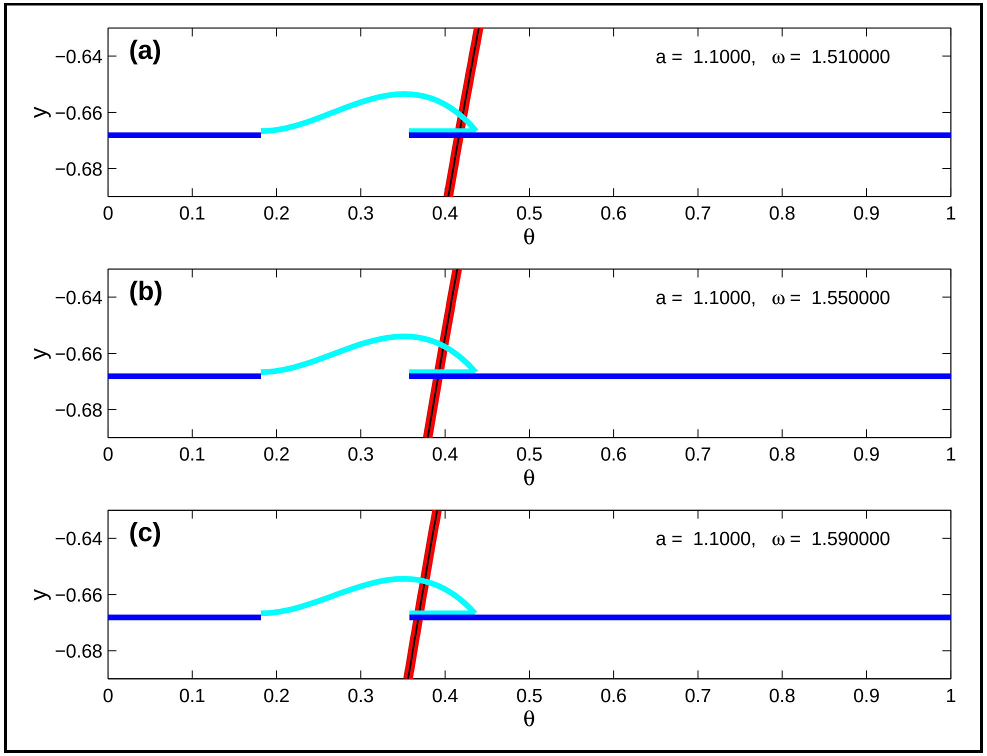


Horseshoe in the Forced van der Pol System



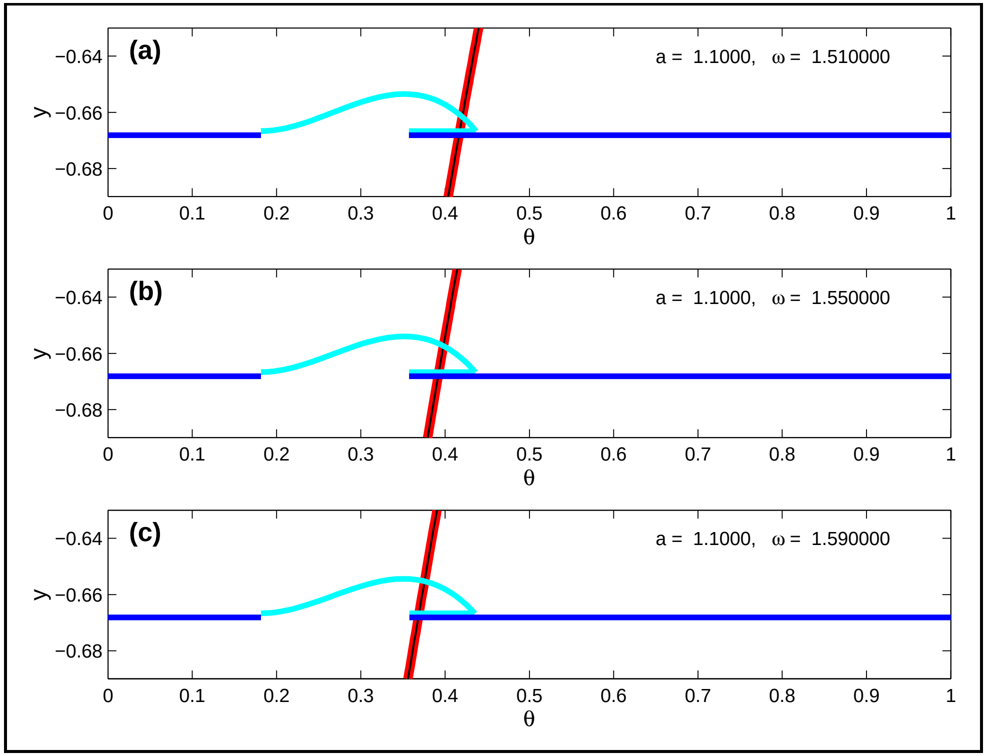


Singular Poincaré Maps



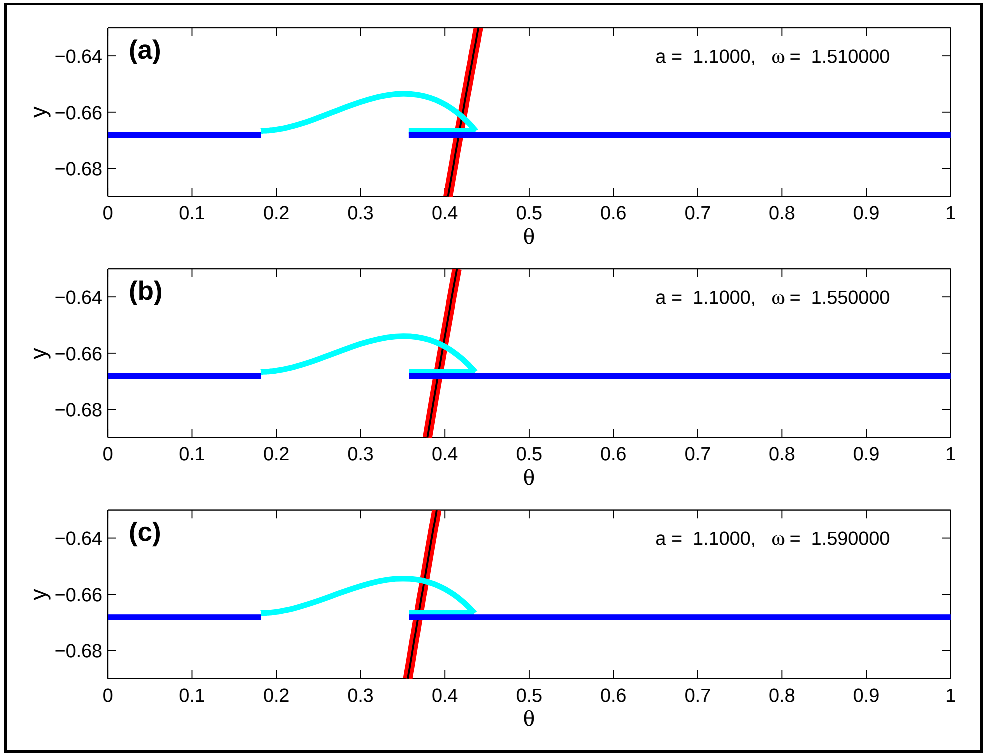
- Based on the fast and slow subsystems only.

Singular Poincaré Maps



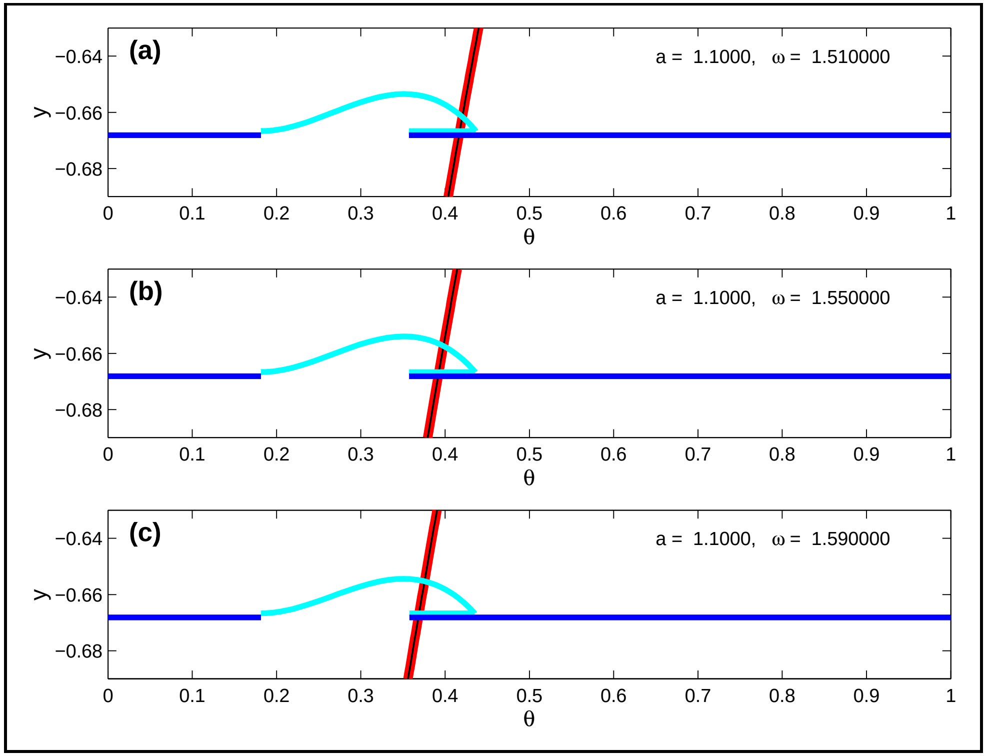
- ❑ Based on the fast and slow subsystems only.
- ❑ Singular limit of canard trajectories results in nonuniqueness.

Singular Poincaré Maps



- Based on the fast and slow subsystems only.
- Singular limit of canard trajectories results in nonuniqueness.
- The “reduced” system provides useful information.

Singular Poincaré Maps



- Based on the fast and slow subsystems only.
- Singular limit of canard trajectories results in nonuniqueness.
- The “reduced” system provides useful information.
- Many programs were written... can we automate this?

Reduced System Computing

Goal:

- Create computational tools for the study of the reduced system of a singularly perturbed differential equation.

Reduced System Computing

Goal:

- ❑ Create computational tools for the study of the reduced system of a singularly perturbed differential equation.

"Simple" Idea:

- ❑ Concatenate solutions to fast and slow subsystems.
- ❑ Create "zeroth order" approximations.

Reduced System Computing

Goal:

- Create computational tools for the study of the reduced system of a singularly perturbed differential equation.

“Simple” Idea:

- Concatenate solutions to fast and slow subsystems.
- Create “zero*th* order” approximations.

Complications:

Reduced System Computing

Goal:

- Create computational tools for the study of the reduced system of a singularly perturbed differential equation.

“Simple” Idea:

- Concatenate solutions to fast and slow subsystems.
- Create “zeroth order” approximations.

Complications:

- Canards

Reduced System Computing

Goal:

- ❑ Create computational tools for the study of the reduced system of a singularly perturbed differential equation.

“Simple” Idea:

- ❑ Concatenate solutions to fast and slow subsystems.
- ❑ Create “zeroth order” approximations.

Complications:

- ❑ Canards
- ❑ Fast periodic orbits (and more general ω limit sets of the fast subsystem)

Slow Subsystem - A Few More Details

$$\begin{aligned} 0 &= f(x,y) \\ \dot{y} &= g(x,y) \end{aligned}$$

Slow Subsystem - A Few More Details

$$\begin{aligned} 0 &= f(x,y) \\ \dot{y} &= g(x,y) \end{aligned}$$

Differentiate $0 = f(x,y)$, solve for $\dot{x}$ to obtain $\dot{x} = -(D_x f)^{-1}(D_y f)g(x,y)$. Multiply by $\det(D_x f)$.

Desingularized slow equations:

$$\begin{aligned} \dot{x} &= -(\operatorname{adj} D_x f)(D_y f)g(x,y) \\ \dot{y} &= \det(D_x f)g(x,y) \end{aligned}$$

Slow Subsystem - A Few More Details

$$\begin{aligned} 0 &= f(x,y) \\ \dot{y} &= g(x,y) \end{aligned}$$

Differentiate $0 = f(x,y)$, solve for $\dot{x}$ to obtain $\dot{x} = -(D_x f)^{-1}(D_y f)g(x,y)$. Multiply by $\det(D_x f)$.

Desingularized slow equations:

$$\begin{aligned} \dot{x} &= -(\operatorname{adj} D_x f)(D_y f)g(x,y) \\ \dot{y} &= \det(D_x f)g(x,y) \end{aligned}$$

Fold points:

$$f(x,y) = 0, \quad \det(D_x f) = 0$$

Fold points are saddle-node equilibria of the fast subsystem.

Example: Two Coupled Neurons

$$\dot{v_1} = -v_1 + \tanh(\sigma_1 v_1) - q_1 - \omega f(v_2)(v_1 + 4)$$

$$\dot{v_2} = -v_2 + \tanh(\sigma_2 v_2) - q_2 - \omega f(v_1)(v_2 + 4)$$

$$\dot{q_1} = \varepsilon(-q_1 + v_1)$$

$$\dot{q_2} = \varepsilon(-q_2 + v_2)$$

$$f(v) = \frac{1}{1 + e^{-40(v - 1/75)}}$$

- Each (v_i, q_i) is a relaxation oscillator.
- When one is firing, the other's v nullcline is depressed ("reciprocal inhibition").
- This system has two fast variables and a two dimensional critical manifold.

Fast/Slow System Definition File: `cpldosc.fs`

```
# Definitions for the coupled oscillator fast/slow system.
cpldosc
# Fast variables:  v1, v2
2
v1
v2
# Slow variables:  q1, q2
2
q1
q2
# Parameters:
3
omega
sigma1
sigma2
# Vector field for the fast variables
-v1+tanh(sigma1*v1) - q1 - omega*(v1+4)/(1+exp(-40*(v2-1/75)))
-v2+tanh(sigma2*v2) - q2 - omega*(v2+4)/(1+exp(-40*(v1-1/75)))
# Vector field for the slow variables
-q1 + v1
-q2 + v2
```

Fast/Slow System Definition File: `cpldosc.fs`

```
# Definitions for the coupled oscillator fast/slow system.
cpldosc
# Fast variables:  v1, v2
2
v1
v2
# Slow variables:  q1, q2
2
q1
q2
# Parameters:
3
omega
sigma1
sigma2
# Vector field for the fast variables
-v1+tanh(sigma1*v1) - q1 - omega*(v1+4)/(1+exp(-40*(v2-1/75)))
-v2+tanh(sigma2*v2) - q2 - omega*(v2+4)/(1+exp(-40*(v1-1/75)))
# Vector field for the slow variables
-q1 + v1
-q2 + v2
```

Fast/Slow System Definition File: `cpldosc.fs`

```
# Definitions for the coupled oscillator fast/slow system.
cpldosc
# Fast variables:  v1, v2
2
v1
v2
# Slow variables:  q1, q2
2
q1
q2
# Parameters:
3
omega
sigma1
sigma2
# Vector field for the fast variables
-v1+tanh(sigma1*v1) - q1 - omega*(v1+4)/(1+exp(-40*(v2-1/75)))
-v2+tanh(sigma2*v2) - q2 - omega*(v2+4)/(1+exp(-40*(v1-1/75)))
# Vector field for the slow variables
-q1 + v1
-q2 + v2
```

Fast/Slow System Definition File: `cpldosc.fs`

```
# Definitions for the coupled oscillator fast/slow system.
cpldosc
# Fast variables:  v1, v2
2
v1
v2
# Slow variables:  q1, q2
2
q1
q2
# Parameters:
3
omega
sigma1
sigma2
# Vector field for the fast variables
-v1+tanh(sigma1*v1) - q1 - omega*(v1+4)/(1+exp(-40*(v2-1/75)))
-v2+tanh(sigma2*v2) - q2 - omega*(v2+4)/(1+exp(-40*(v1-1/75)))
# Vector field for the slow variables
-q1 + v1
-q2 + v2
```

Fast/Slow System Definition File: cpldosc.fs

```
# Definitions for the coupled oscillator fast/slow system.
cpldosc
# Fast variables:  v1, v2
2
v1
v2
# Slow variables:  q1, q2
2
q1
q2
# Parameters:
3
omega
sigma1
sigma2
# Vector field for the fast variables
-v1+tanh(sigma1*v1) - q1 - omega*(v1+4)/(1+exp(-40*(v2-1/75)))
-v2+tanh(sigma2*v2) - q2 - omega*(v2+4)/(1+exp(-40*(v1-1/75)))
# Vector field for the slow variables
-q1 + v1
-q2 + v2
```

Fast/Slow System Definition File: `cpldosc.fs`

```
# Definitions for the coupled oscillator fast/slow system.
cpldosc
# Fast variables:  v1, v2
2
v1
v2
# Slow variables:  q1, q2
2
q1
q2
# Parameters:
3
omega
sigma1
sigma2
# Vector field for the fast variables
-v1+tanh(sigma1*v1) - q1 - omega*(v1+4)/(1+exp(-40*(v2-1/75)))
-v2+tanh(sigma2*v2) - q2 - omega*(v2+4)/(1+exp(-40*(v1-1/75)))
# Vector field for the slow variables
-q1 + v1
-q2 + v2
```

Fast/Slow System Definition File: `cpldosc.fs`

```
# Definitions for the coupled oscillator fast/slow system.
cpldosc
# Fast variables:  v1, v2
2
v1
v2
# Slow variables:  q1, q2
2
q1
q2
# Parameters:
3
omega
sigma1
sigma2
# Vector field for the fast variables
-v1+tanh(sigma1*v1) - q1 - omega*(v1+4)/(1+exp(-40*(v2-1/75)))
-v2+tanh(sigma2*v2) - q2 - omega*(v2+4)/(1+exp(-40*(v1-1/75)))
# Vector field for the slow variables
-q1 + v1
-q2 + v2
```

Computer Code Generation

- C code is generated, to be used with the SUNDIALS suite [http://www.llnl.gov/CASC/sundials] SUNDIALS includes CVODE for ODEs and IDA for DAEs.

Computer Code Generation

- C code is generated, to be used with the SUNDIALS suite [http://www.llnl.gov/CASC/sundials]
  SUNDIALS includes CVODE for ODEs and IDA for DAEs.
- Jacobians are generated automatically (via GiNaC)

Computer Code Generation

- C code is generated, to be used with the SUNDIALS suite [http://www.llnl.gov/CASC/sundials]
  SUNDIALS includes CVODE for ODEs and IDA for DAEs.
- Jacobians are generated automatically (via GiNaC)
- Some MATLAB code is also generated.
  Other target languages or libraries could be implemented.

Computer Code Generation

- C code is generated, to be used with the SUNDIALS suite
  [http://www.llnl.gov/CASC/sundials]
  SUNDIALS includes CVODE for ODEs and IDA for DAEs.
- Jacobians are generated automatically (via GiNaC)
- Some MATLAB code is also generated.
  Other target languages or libraries could be implemented.

Input file: `cpldosc.fs`

Output files:

`fs_cpldosc.c`	General C functions
`fs_cpldosc_cvode.c`	C functions for CVODE
`fs_cpldosc_ida.c`	C functions for IDA
`fs_cpldosc.m`	MATLAB functions

Computer Code Generated

The generated code includes:

Computer Code Generated

The generated code includes:

- *Fast vector field and IVP solver*

Fast vector field, to be used with CVODE

```
int cpldoscF_cv(realtype t, N_Vector Xvec, N_Vector Xdotvec,
                                                      void *params)
```

Fast vector field Jacobian

```
int cpldoscFx_cv(long int N, DenseMat Fx, realtype t,
                        N_Vector Xvec, N_Vector Fvec, void *params,
                       N_Vector tmp1, N_Vector tmp2, N_Vector tmp3)
```

Solve the fast subsystem IVP

```
void cpldosc_fast(FILE *fastfile, double X[], double Y[],
                      double params[], SolverParams *solver_params)
```

Computer Code Generated

The generated code includes:

- *Slow subsystem DAE, configured for IDA*

Compute residuals for the IDA DAE solver

```
int cpldosc_idares(realtype t, N_Vector Zvec, N_Vector Zdotvec,
                                       N_Vector rvec, void *params)
```

Jacobian for IDA DAE solver

```
int cpldosc_idajac(long int Neq, realtype t, N_Vector Zvec,
                                   N_Vector Zdotvec, N_Vector rvec,
                       realtype c_j, void *params, DenseMat jacmat,
                       N_Vector tmp1, N_Vector tmp2, N_Vector tmp3)
```

Solve the slow subsystem DAE IVP

```
void cpldosc_slow_dae(FILE *slowfile, double X[], double Y[],
                      double params[], SolverParams *solver_params)
```

Computer Code Generated

The generated code includes:

- *Desingularized slow subsystem (for CVODE)*

  Desingularized slow subsystem vector field

  ```
  int cpldosc_SlowDE(realtype t, N_Vector Zvec, N_Vector Zdotvec,
                                                        void *params)
  ```

  Desingularized slow subsystem IVP solver

  ```
  void cpldosc_slow_des(FILE *slowfile, double X[], double Y[],
                        double params[], SolverParams *solver_params)
  ```

Computer Code Generated

The generated code includes:

- *Fold function and Jacobian*

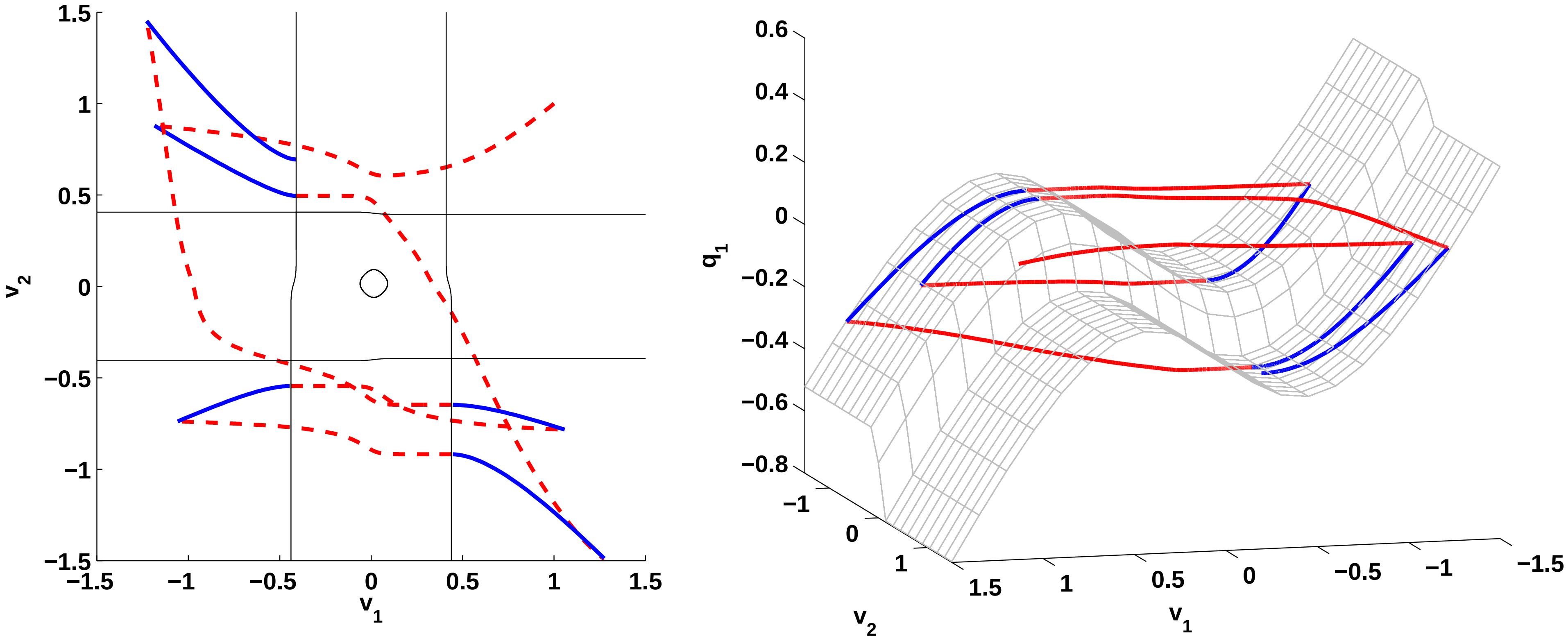
  Fold function

  ```
  realtype cpldosc_foldfunc_nv(N_Vector Zvec, void *params)
  ```

  Fold function gradient (with respect to all variables)

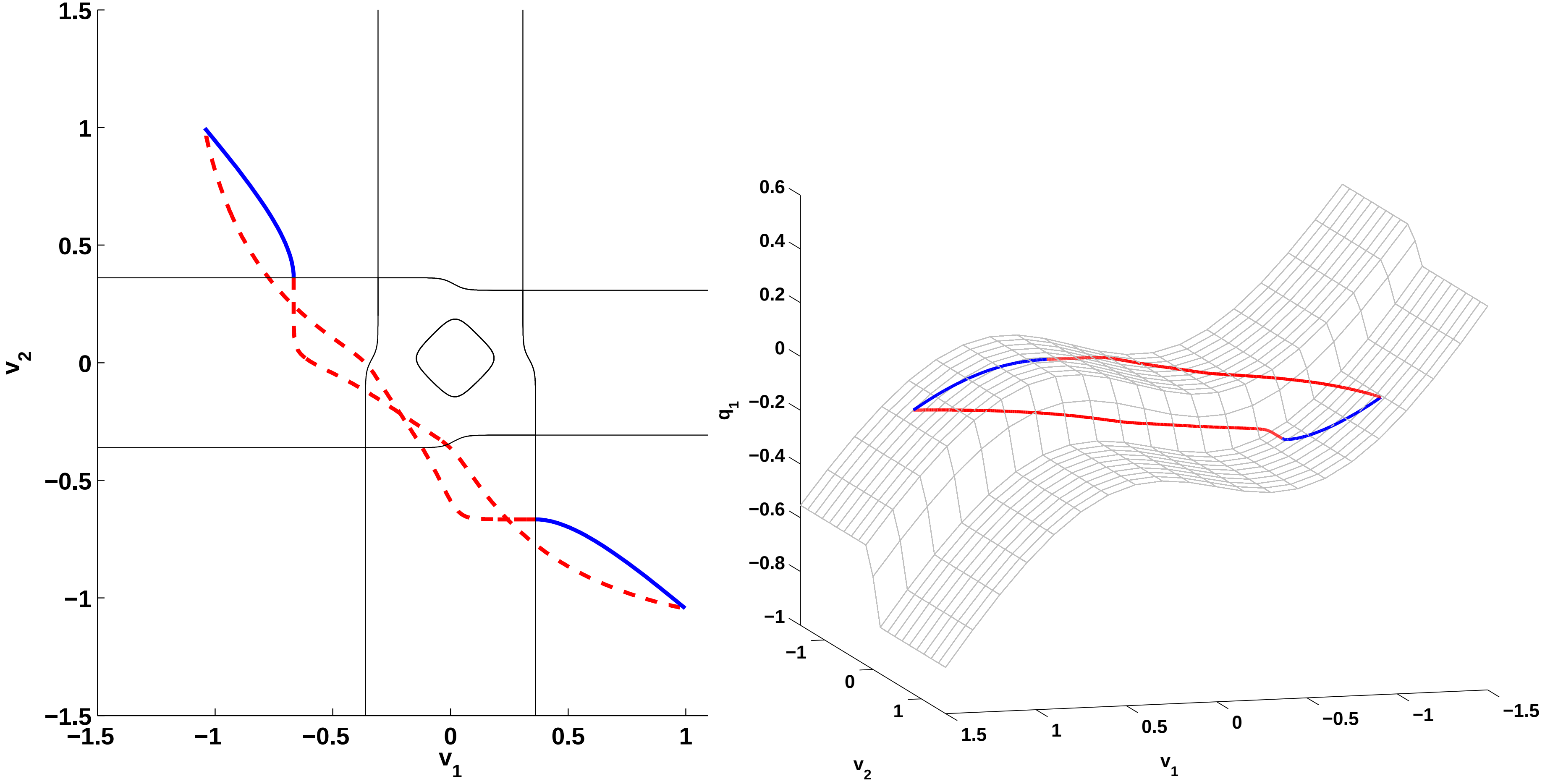
  ```
  void cpldosc_foldfunc_grad_nv(double *grad, N_Vector Zvec,
                                                        void *params)
  ```

Example: Initial Value Problem for the Coupled Oscillator System



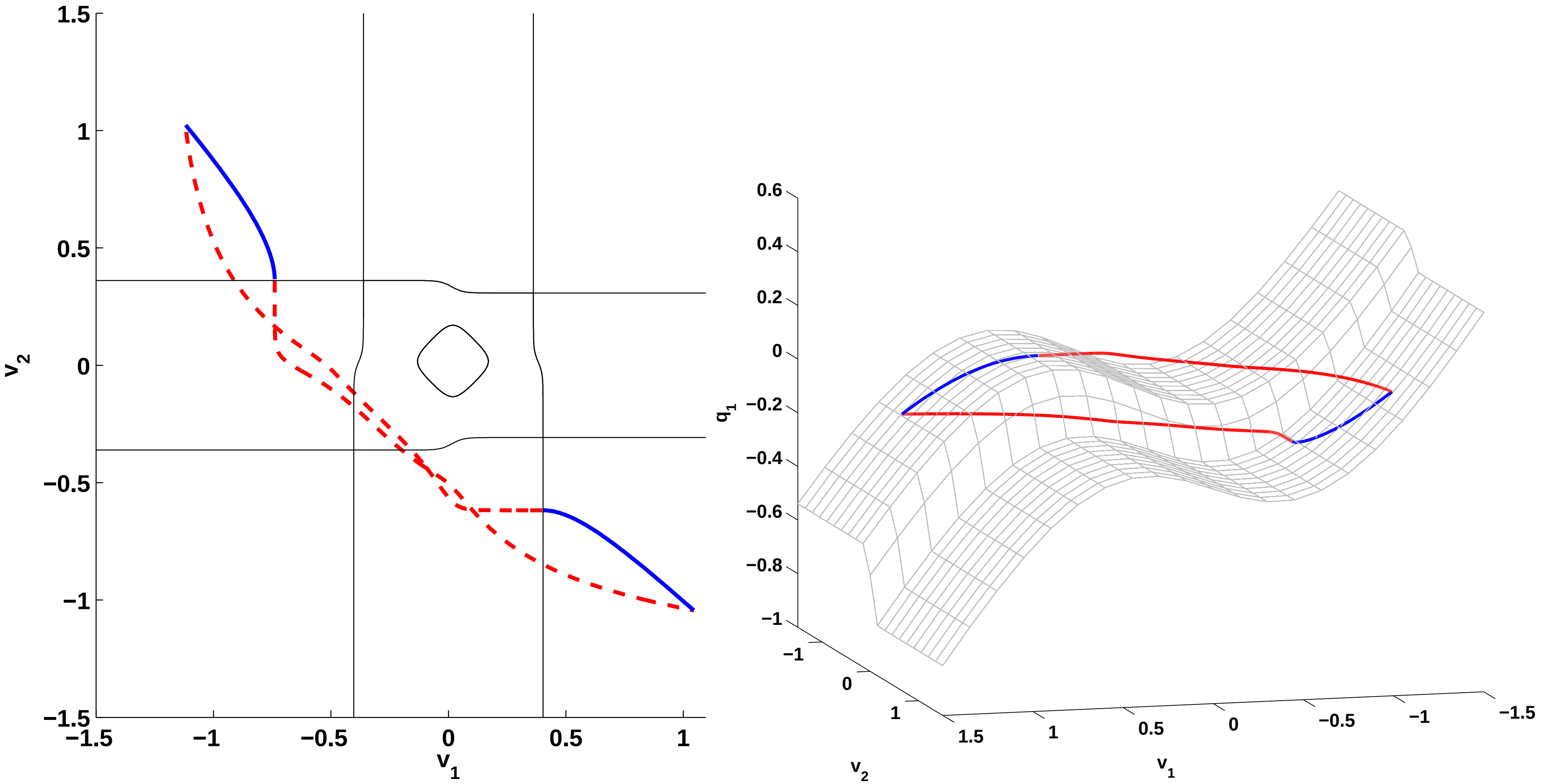
$(\omega = 0.05,\ \sigma_1 = 1.5,\ \sigma_2 = 2.6)$

Periodic Orbits of the Coupled Oscillator System ($\omega = 0.05$, $\sigma_2 = 1.2$)



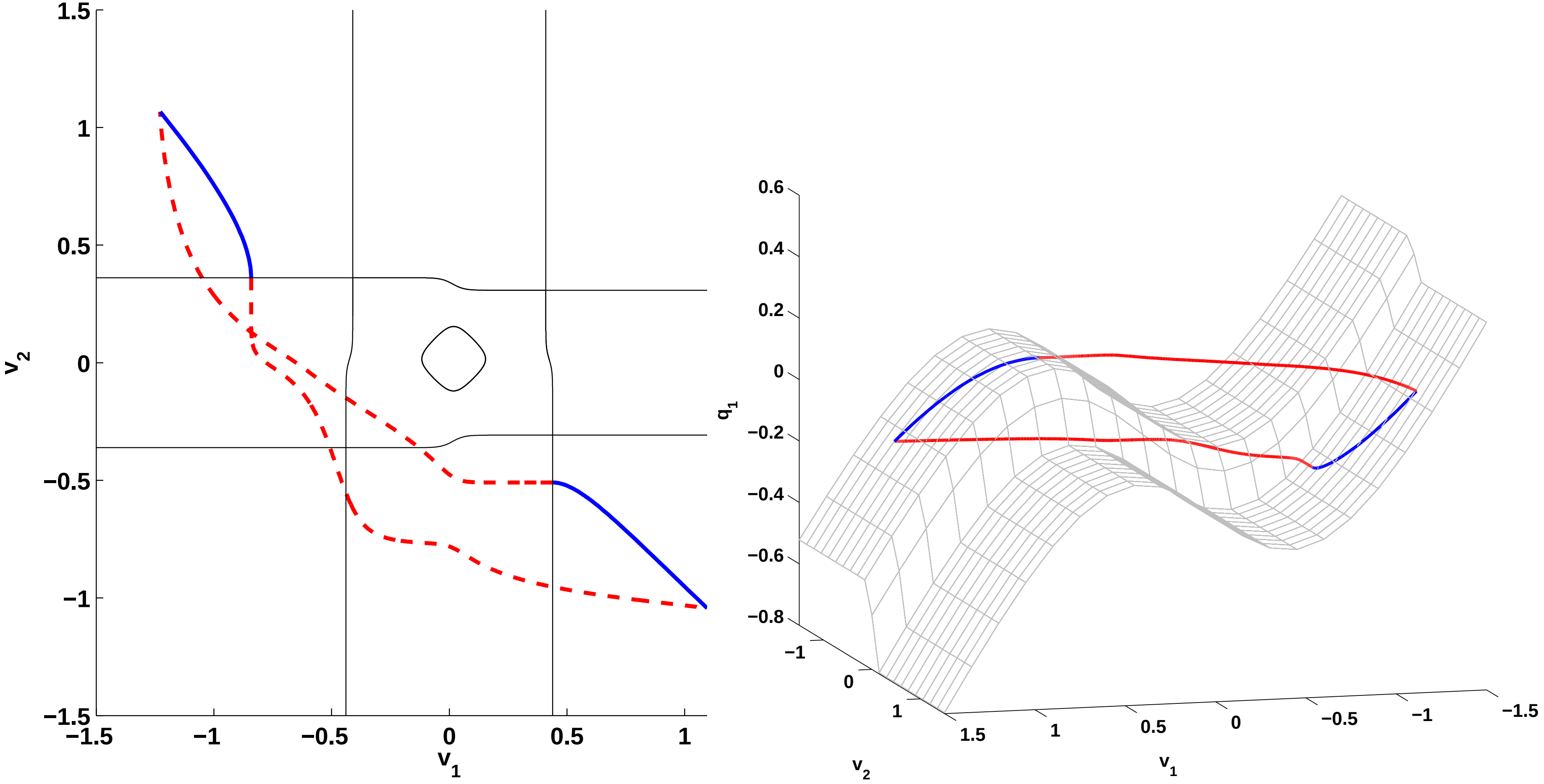
$\sigma_1 = 1.2$

Periodic Orbits of the Coupled Oscillator System $(\omega = 0.05,\ \sigma_2 = 1.2)$



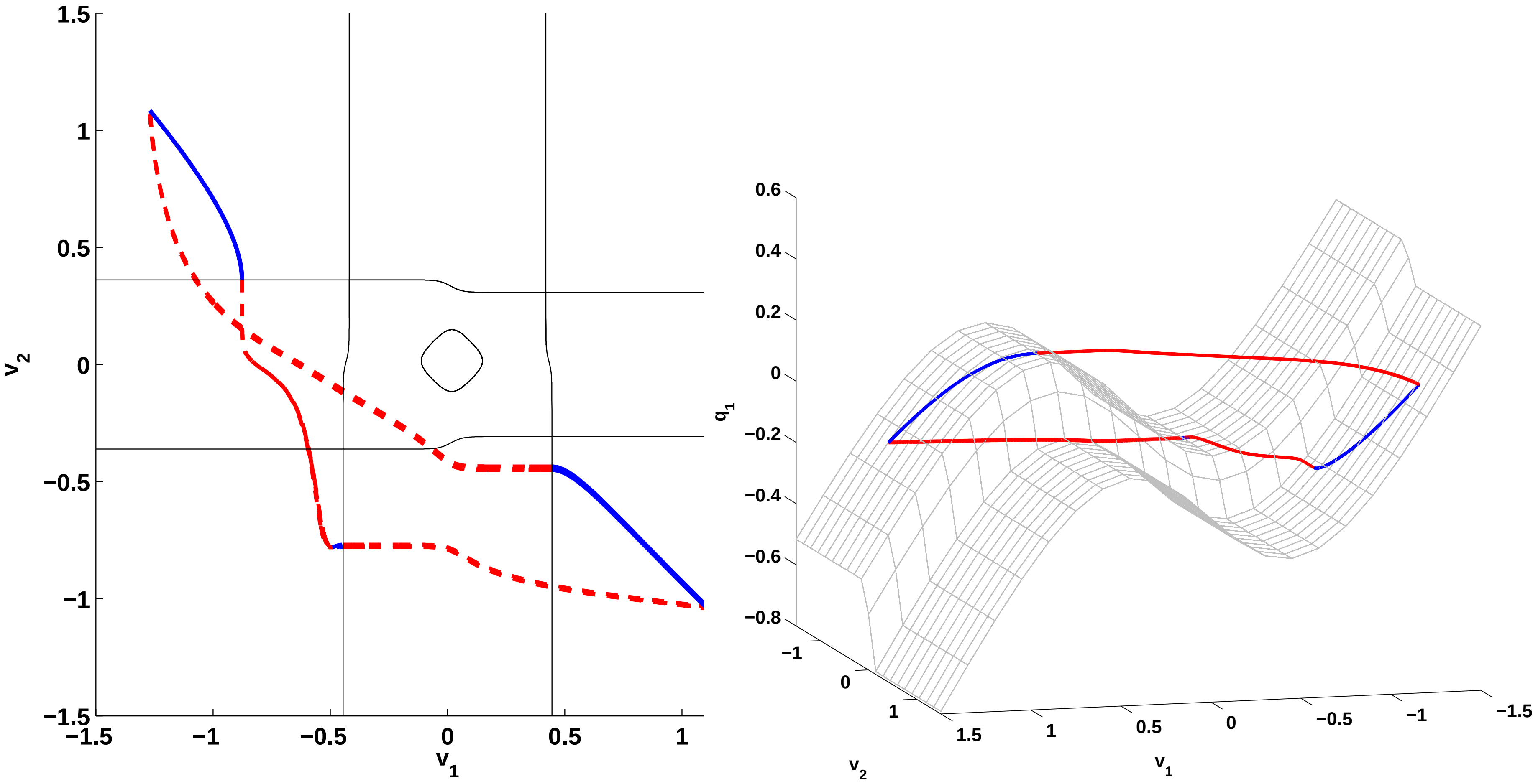
$\sigma_1 = 1.3$

Periodic Orbits of the Coupled Oscillator System $(\omega = 0.05,\ \sigma_2 = 1.2)$



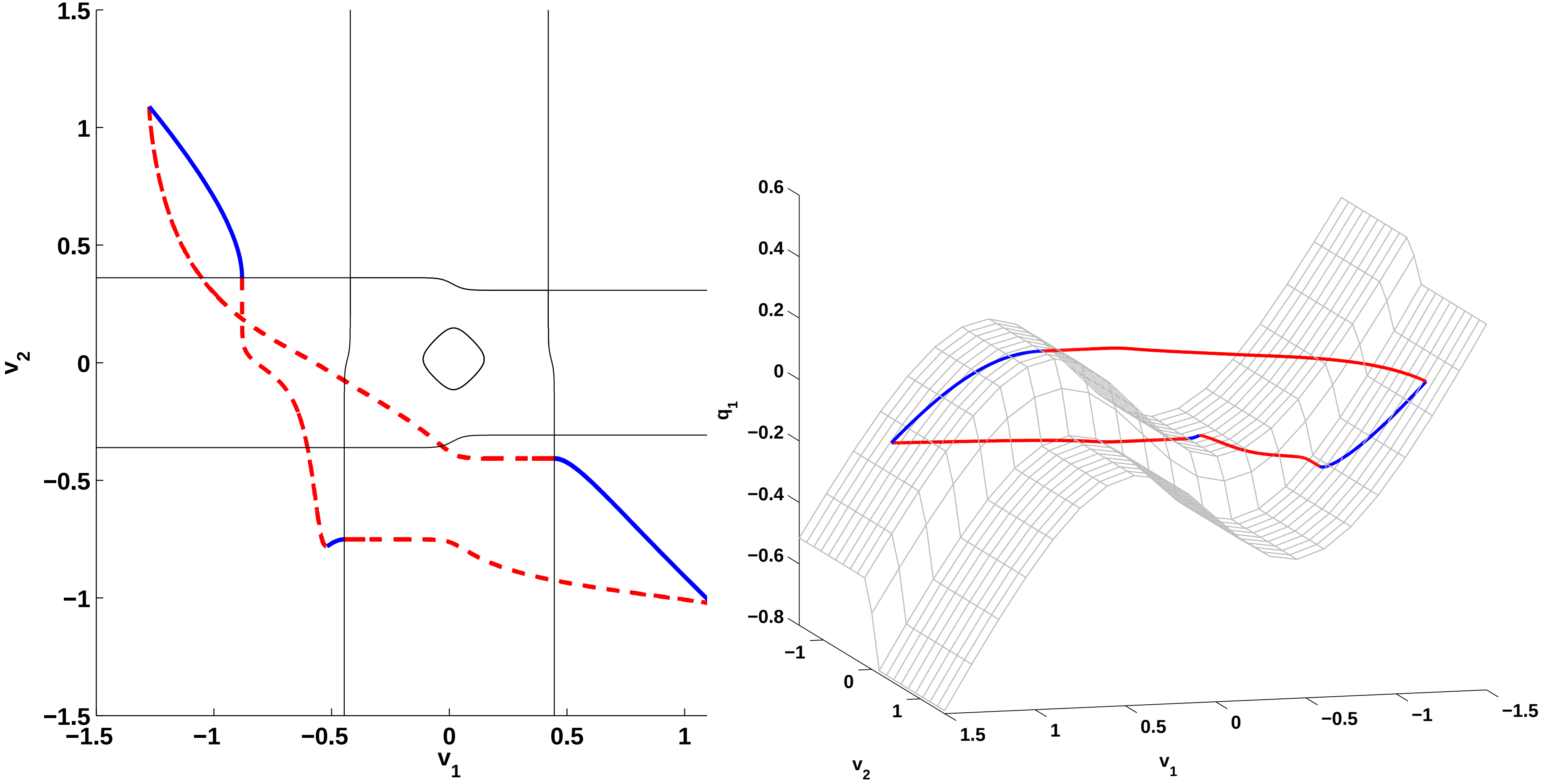
$$\sigma_1 = 1.4$$

Periodic Orbits of the Coupled Oscillator System $(\omega = 0.05,\ \sigma_2 = 1.2)$



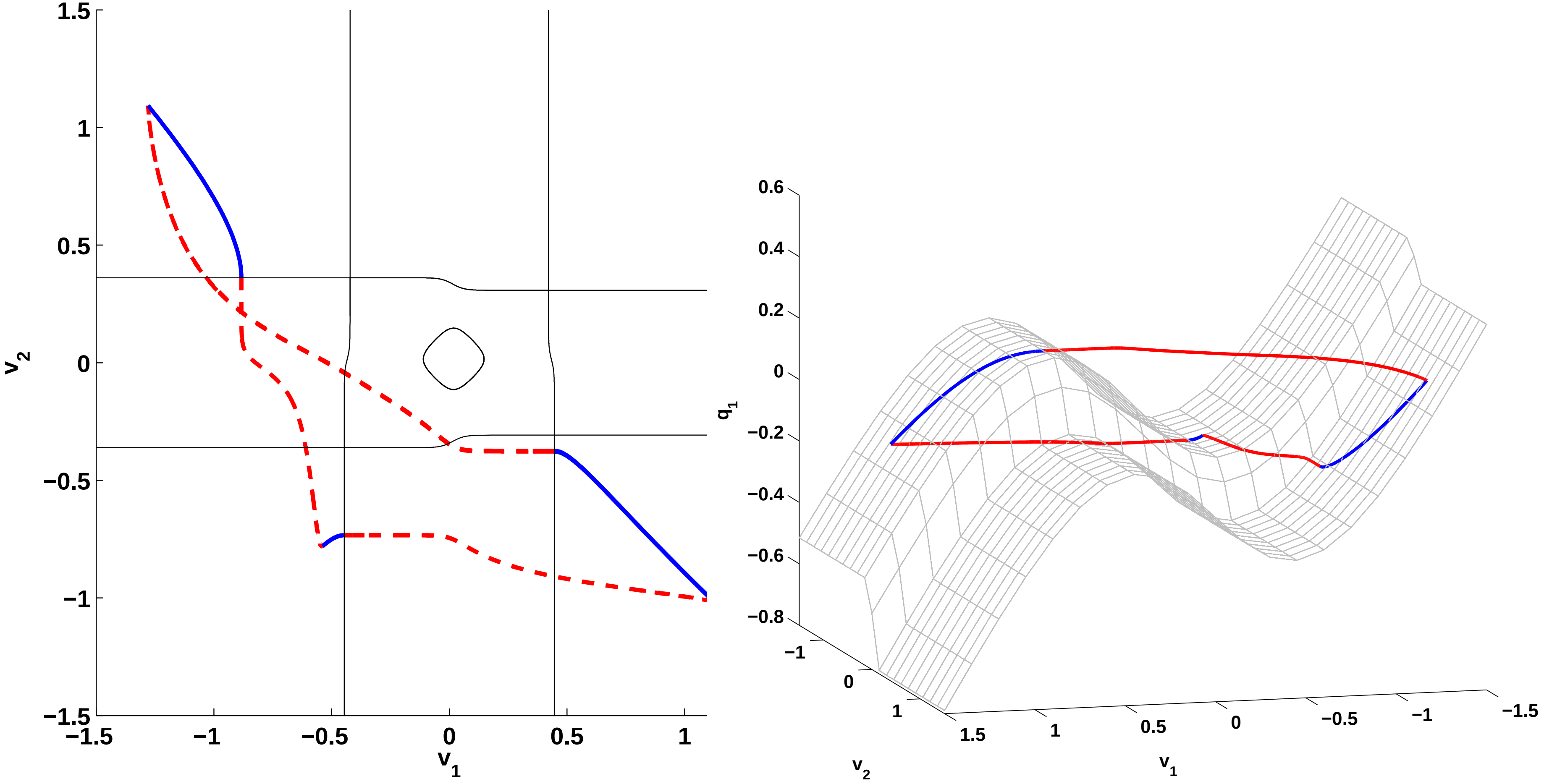
$$\sigma_1 = 1.5925$$

Periodic Orbits of the Coupled Oscillator System ($\omega = 0.05$, $\sigma_2 = 1.2$)



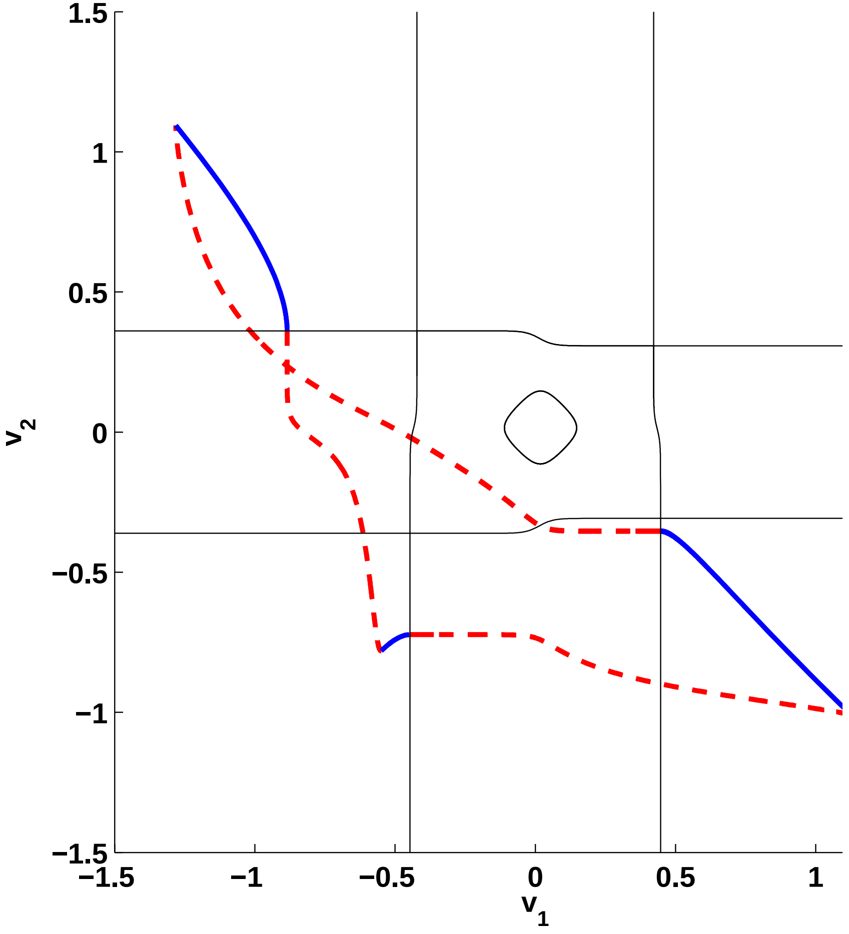
$$\sigma_1 = 1.6125$$

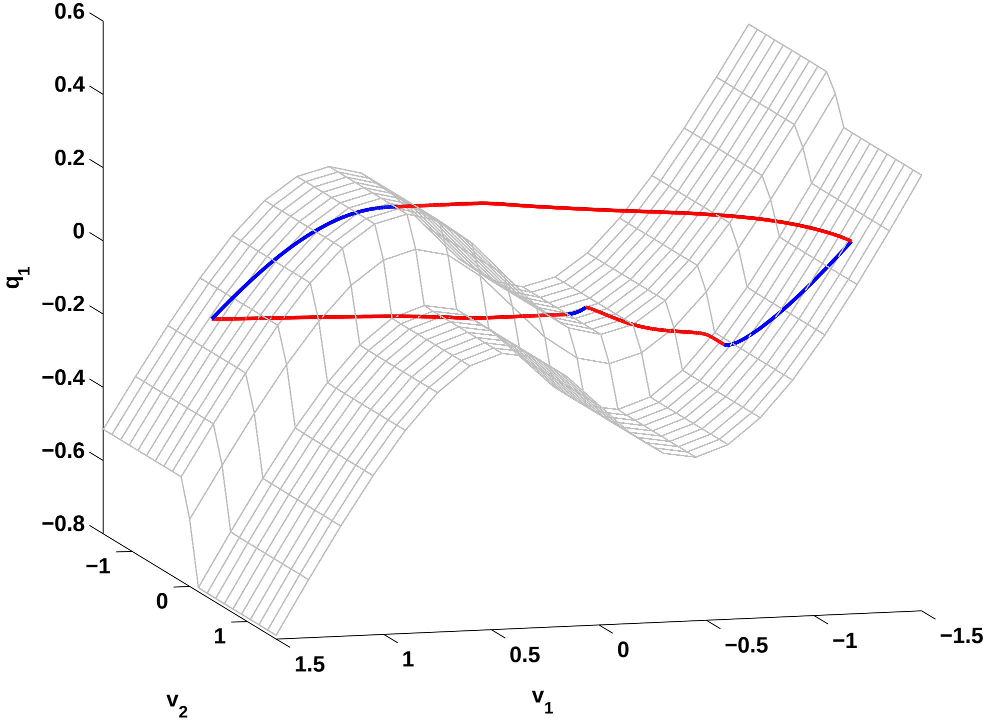
Periodic Orbits of the Coupled Oscillator System ($\omega = 0.05$, $\sigma_2 = 1.2$)



$$\sigma_1 = 1.6250$$

Periodic Orbits of the Coupled Oscillator System ($\omega = 0.05$, $\sigma_2 = 1.2$)

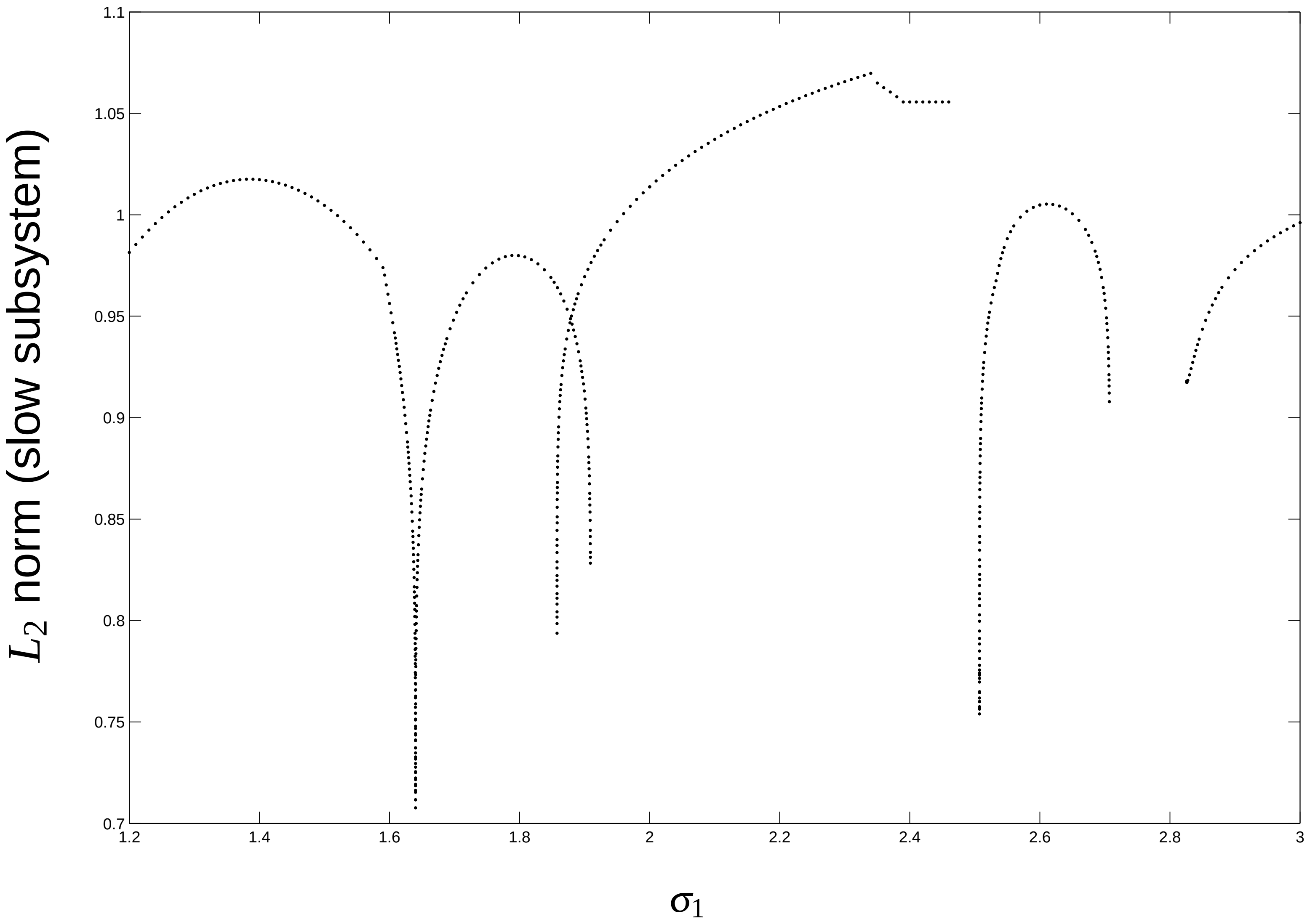




$$\sigma_1 = 1.6325$$

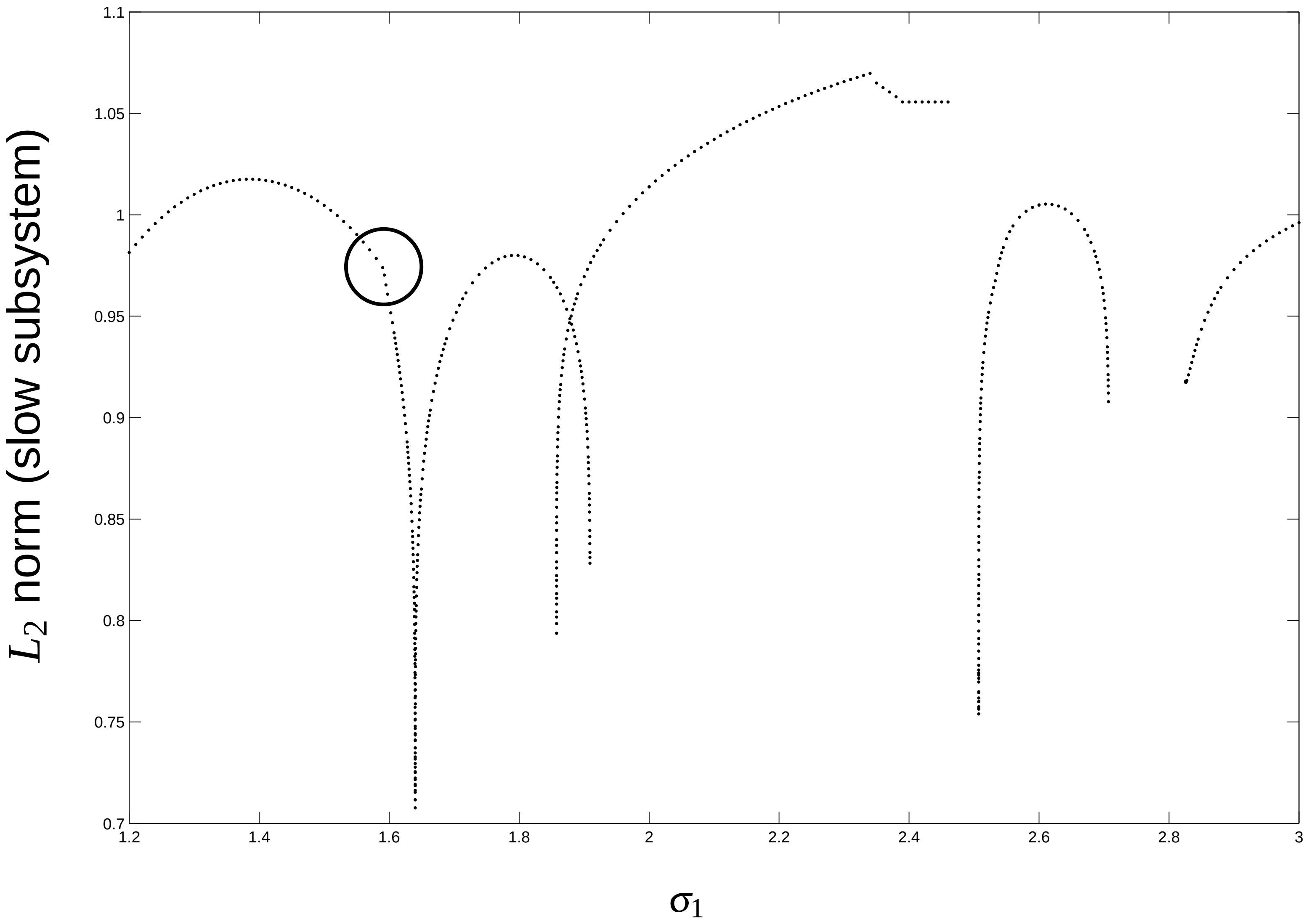
Bifurcation Diagram: Stable Periodic Orbits

(“Poor man’s” continuation; not a complete diagram)



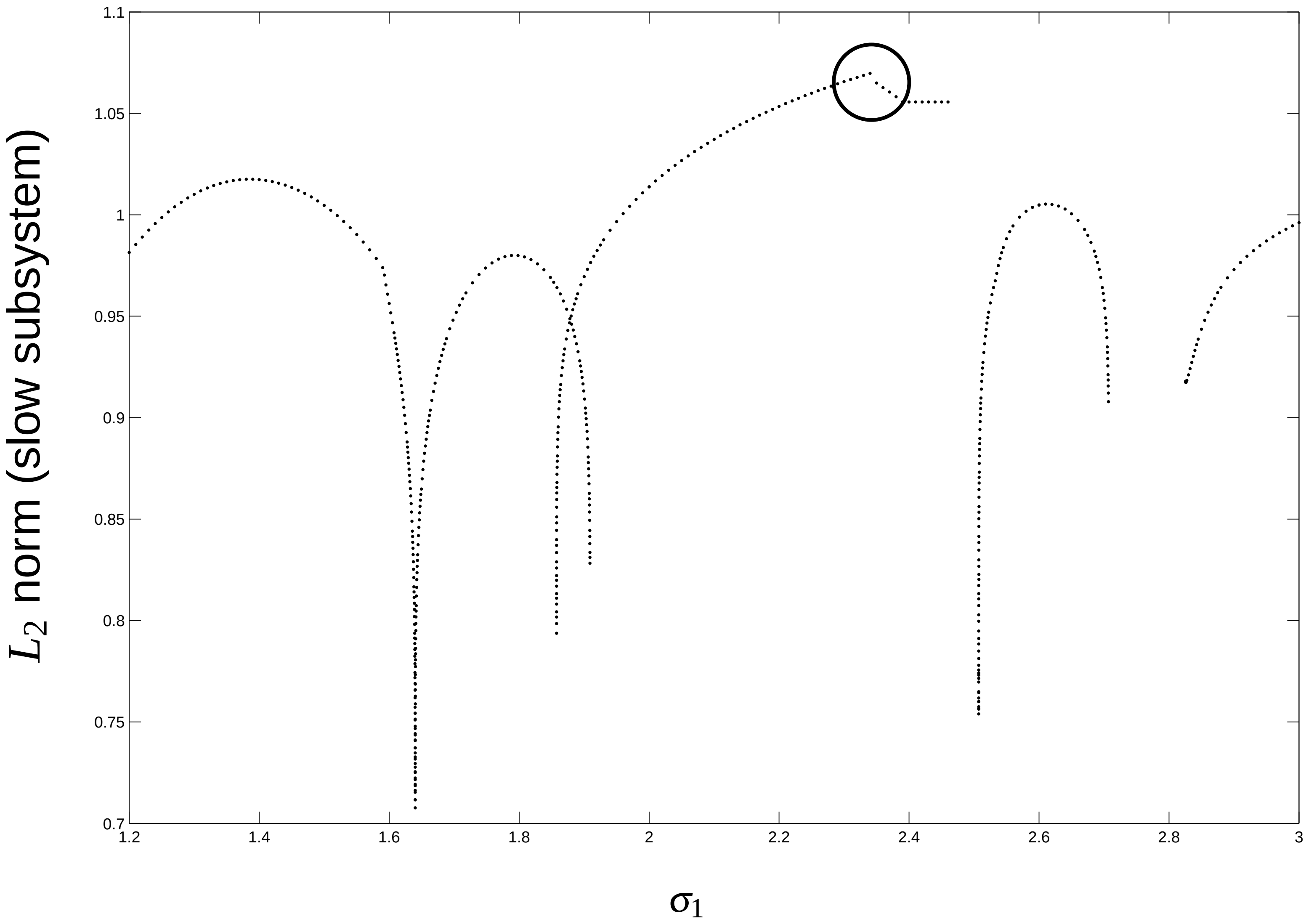
Bifurcation Diagram: Stable Periodic Orbits

(“Poor man’s” continuation; not a complete diagram)



Bifurcation Diagram: Stable Periodic Orbits

(“Poor man’s” continuation; not a complete diagram)



Current and Future Development

Current and Future Development

- Boundary value problem solver that handles canards (like the forced van der Pol example).

Current and Future Development

- Boundary value problem solver that handles canards (like the forced van der Pol example).
- Computing and continuing periodic orbits.

Current and Future Development

- Boundary value problem solver that handles canards (like the forced van der Pol example).
- Computing and continuing periodic orbits.
- Find fast/slow orbit homoclinic to folded singular points.
  These play an import role in the bifurcations of periodic orbits:
  - J. Guckenheimer, K. Hoffman, W. Weckesser,
    Bifurcations of relaxation oscillations near folded saddles,
    Int. J. Bif. Chaos 15(11), (2005)
  - M. Brøns, M. Krupa, M. Wechselberger,
    Mixed Mode Oscillations Due to the Generalized Canard Phenomenon,
    Fields Institute Communications Vol. 49, 39-63 (2006)

Current and Future Development

- Boundary value problem solver that handles canards (like the forced van der Pol example).
- Computing and continuing periodic orbits.
- Find fast/slow orbit homoclinic to folded singular points.
  These play an import role in the bifurcations of periodic orbits:
  - J. Guckenheimer, K. Hoffman, W. Weckesser,
    Bifurcations of relaxation oscillations near folded saddles,
    Int. J. Bif. Chaos 15(11), (2005)
  - M. Brøns, M. Krupa, M. Wechselberger,
    Mixed Mode Oscillations Due to the Generalized Canard Phenomenon,
    Fields Institute Communications Vol. 49, 39-63 (2006)
- Singular variational equations.

Current and Future Development

- Boundary value problem solver that handles canards (like the forced van der Pol example).
- Computing and continuing periodic orbits.
- Find fast/slow orbit homoclinic to folded singular points.
  These play an import role in the bifurcations of periodic orbits:
  - J. Guckenheimer, K. Hoffman, W. Weckesser,
    Bifurcations of relaxation oscillations near folded saddles,
    Int. J. Bif. Chaos 15(11), (2005)
  - M. Brøns, M. Krupa, M. Wechselberger,
    Mixed Mode Oscillations Due to the Generalized Canard Phenomenon,
    Fields Institute Communications Vol. 49, 39-63 (2006)
- Singular variational equations.
- Fast periodic orbits and the averaged fast subsystem.