



Mathematical Modelling and Scientific Computing in the Biosciences

| 26 June 2007

Lecture 10: Overview

- **MATCONT software**
 - capabilities, examples
 - numerical method

Softwares for Analysis of Dynamical Systems

- **AUTO: Software for Continuation and Bifurcation Problems in ODEs**
 - developed from 1980–present, by E. Doedel *et al*
 - widely used in dynamical systems community
 - written in Fortran
 - website: <http://indy.cs.concordia.ca/auto>
- **CONTENT: Interactive Environment for Studying Dynamical Systems**
 - developed from 1995–present, by Y. A. Kuznetsov *et al*
 - written in C
 - website: <http://www.enm.bris.ac.uk/staff/hinke/dss/continuation/content.html>
- **MATCONT: Continuation Software**
 - developed from 2003–present, by W. Govaerts, Y. A. Kuznetsov *et al*
 - encompasses most features of existing softwares such as AUTO, CONTENT
 - new capabilities being developed
 - easily used from MATLAB environment
 - website: <http://www.matcont.ugent.be/>

Softwares for Analysis of Dynamical Systems

Supported functionalities for ODEs in AUTO (A), CONTENT (C) and MATCONT (M)

	A	C	M
time-integration		+	+
Poincaré maps			+
monitoring user functions along curves computed by continuation	+	+	+
continuation of equilibria	+	+	+
detection of branch points and codim 1 bifurcations (limit and Hopf points) of equilibria	+	+	+
computation of normal forms for codim 1 bifurcations of equilibria		+	+
continuation of codim 1 bifurcations of equilibria	+	+	+
detection of codim 2 equilibrium bifurcations (cusp, Bogdanov-Takens, fold-Hopf, generalized and double Hopf)		+	+
computation of normal forms for codim 2 bifurcations of equilibria			+

Softwares for Analysis of Dynamical Systems

continuation of codim 2 equilibrium bifurcations in three parameters		+	
continuation of limit cycles	+	+	+
computation of phase response curves and their derivatives			+
detection of branch points and codim 1 bifurcations (limit points, flip and Neimark-Sacker (torus)) of cycles	+	+	+
continuation of codim 1 bifurcations of cycles	+		+
branch switching at equilibrium and cycle bifurcations	+	+	+
continuation of branch points of equilibria and cycles			+
computation of normal forms for codim 1 bifurcations of cycles			+
detection of codim 2 bifurcations of cycles			+
continuation of orbits homoclinic to equilibria	+		+

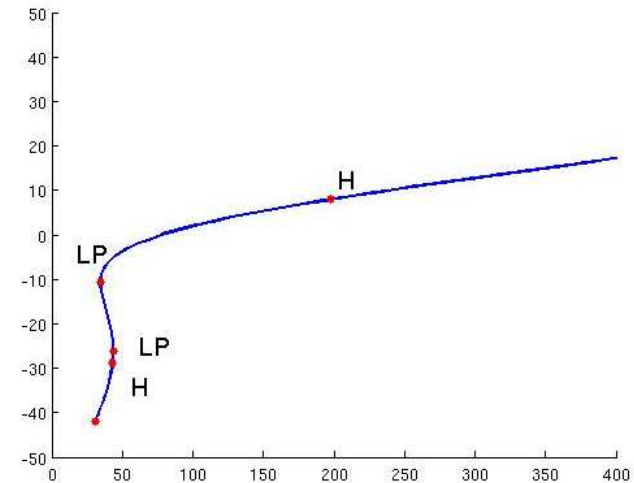
Simple Analysis with MATCONT: 2D Morris–Lecar System

```

>> opt=contset;opt=contset(opt,'Singularities',1);
opt=contset(opt,'MaxStepsize',10);
opt=contset(opt,'MaxNumpoints',200);
>> [x1,v1]=init_EP_EP(@MyML,x0,[30;6],[1]);
>> [x,v,s,h,f]=cont(@equilibrium,x1,[],opt);
first point found
tangent vector to first point found
label = H , x = ( -28.700772 0.018189 43.312018 )
First Lyapunov coefficient = 6.462166e-03
label = LP, x = ( -26.127769 0.024296 43.740592 )
a=9.306353e-03
label = LP, x = ( -10.804120 0.126584 34.546930 )
a=2.105192e-03
label = H , x = ( 7.947868 0.555741 197.796288 )
First Lyapunov coefficient = 8.904936e-04

elapsed time = 1.2 secs
npoints curve = 200
>>

```



- Vector field associated with the ODE system:
 - passed function handle, `@MyML`
- `init_EP_EP`: initialize structure for equilibrium continuation
 - from starting equilibrium solution `x0`
 - starting parameter values `[30; 6]`
 - choosing the first, `[1]`, as bifurcation parameter

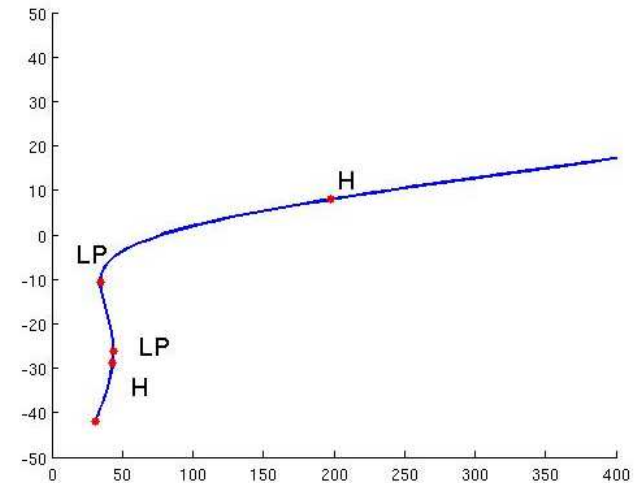
Simple Analysis with MATCONT: 2D Morris–Lecar System

```

>> opt=contset;opt=contset(opt,'Singularities',1);
opt=contset(opt,'MaxStepsize',10);
opt=contset(opt,'MaxNumpoints',200);
>> [x1,v1]=init_EP_EP(@MyML,x0,[30;6],[1]);
>> [x,v,s,h,f]=cont(@equilibrium,x1,[],opt);
first point found
tangent vector to first point found
label = H , x = ( -28.700772 0.018189 43.312018 )
First Lyapunov coefficient = 6.462166e-03
label = LP, x = ( -26.127769 0.024296 43.740592 )
a=9.306353e-03
label = LP, x = ( -10.804120 0.126584 34.546930 )
a=2.105192e-03
label = H , x = ( 7.947868 0.555741 197.796288 )
First Lyapunov coefficient = 8.904936e-04

elapsed time = 1.2 secs
npoints curve = 200
>>

```



- **cont**: performs continuation
 - using method specified by the **@equilibrium** function
 - specifying the num. of solution points, step size, detect singularities
- computed solution contained in **[x,v,s,h,f]**
 - **x**: all the computed points
 - **s**: list of singularity points (**H** = Hopf, **LP** = Limit Point)



Simple Analysis with MATCONT: 2D Morris–Lecar System

- Specifying vector field of ODE:
 - `@fun_eval`
 - optional analytical derivatives: `@jacobian`, `@jacobianp`, ...

```
function out = MyML
out{1} = @init;
out{2} = @fun_eval;
out{3} = @jacobian;
out{4} = @jacobianp;
out{5} = @hessians;
out{6} = @hessiansp;
out{7} = [];
out{8} = [];
out{9} = [];

% -----
function dydt = fun_eval(t,kmrgd,Inp,V3)
Minf=(1+tanh((kmrgd(1)+1.2)/18))/2;
Ninf=(1+tanh((kmrgd(1)-V3)/17.4))/2;
tau=1/15*cosh((kmrgd(1)-V3)/34.8);
dydt=[1/5*(Inp-2*(kmrgd(1)+60)-4*Minf*(kmrgd(1)-120)-8*kmrgd(2)*(kmrgd(1)+80);
tau*(Ninf-kmrgd(2))];
```

Simple Analysis with MATCONT: 2D Morris–Lecar System

- computed solution $[x,v,s,h,f]$
 - x : equilibrium solutions and the associated bifurcation parameter

```
>> x
x =
  1.0e+03 *
Columns 1 through 6
-0.0419  -0.0419  -0.0419  -0.0419  -0.0419  -0.0419
 0.0000   0.0000   0.0000   0.0000   0.0000   0.0000
 0.0305   0.0305   0.0305   0.0305   0.0305   0.0305
Columns 7 through 12
-0.0418  -0.0418  -0.0418  -0.0417  -0.0417  -0.0416
 0.0000   0.0000   0.0000   0.0000   0.0000   0.0000
 0.0306   0.0306   0.0306   0.0307   0.0308   0.0309
Columns 13 through 18
```

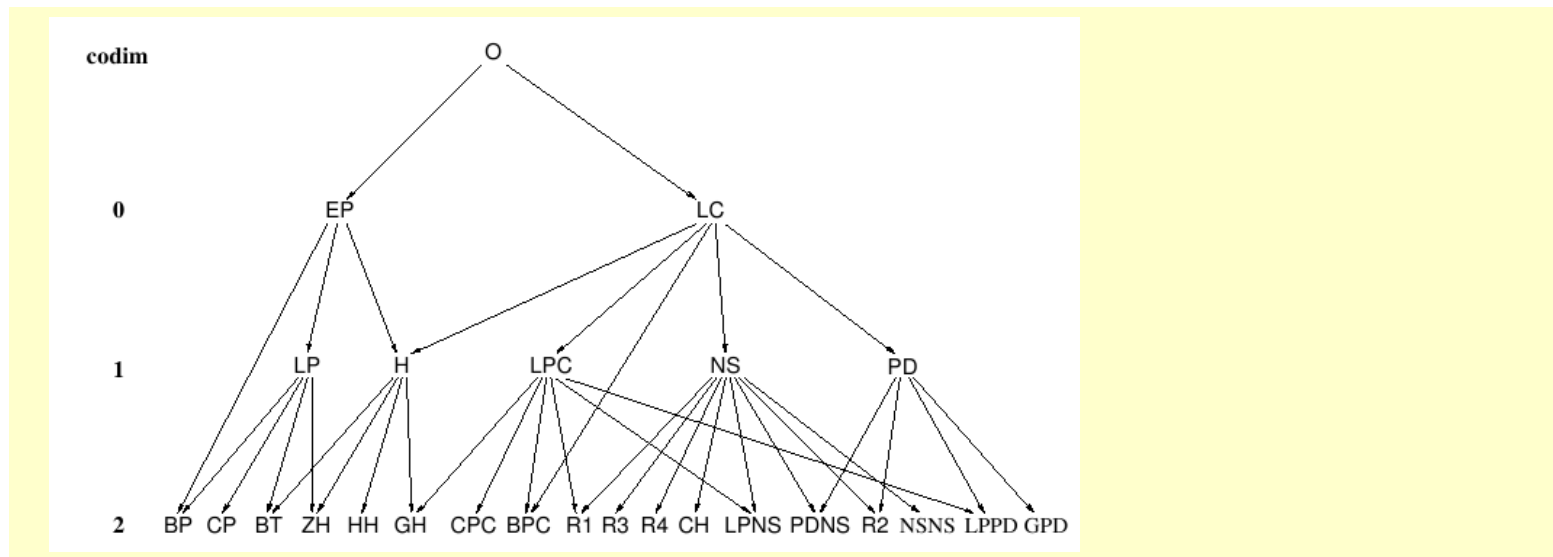
Simple Analysis with MATCONT: 2D Morris–Lecar System

- computed solution $[x,v,s,h,f]$
 - s : list of singularity points (H = Hopf, LP = Limit Point)

```
>> s
s =
6x1 struct array with fields:
    index
    label
    data
    msg
>> s(2)
ans =
    index: 26
    label: 'H '
    data: [1x1 struct]
    msg: 'Hopf'
>>
```

Continuation of Bifurcation Point

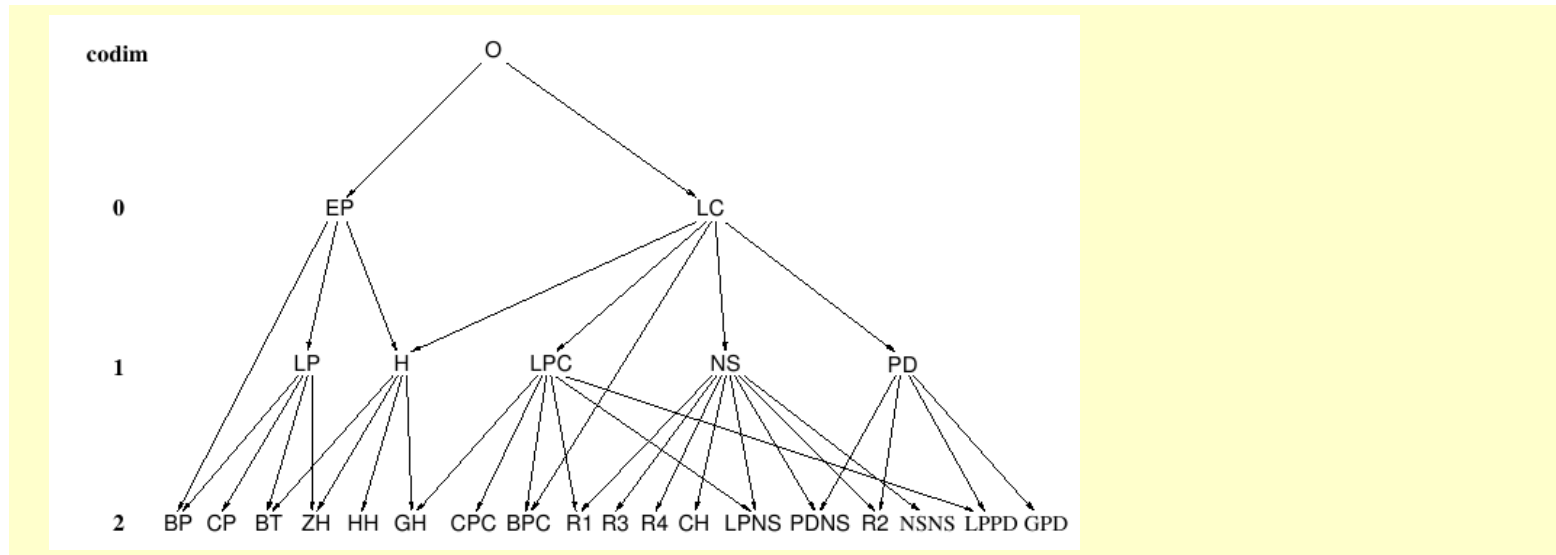
- Having detected bifurcation points (**H**, **LP**), what else can we do?
 - continuation of bifurcation points
 - detection of (higher **co-dim**) singularity points on bifurcation manifold



- Singularities on continuation of Limit Point:
 - Branching Point (**BP**), Cusp Point (**CP**), Bogdanov–Takens (**BT**), ...
- Singularities on continuation of Hopf:
 - Zero Hopf (**ZH**), Double Hopf (**HH**), Generalized Hopf (**GH**), ...

Continuation of Bifurcation Point

- Detection of (higher **co-dim**) singularity points on bifurcation manifold
 - continuation of bifurcation points
 - detection of (higher **co-dim**) singularity points on bifurcation manifold



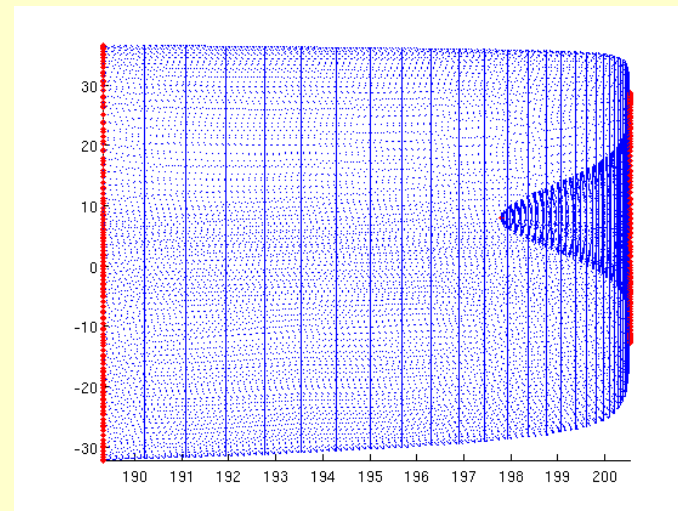
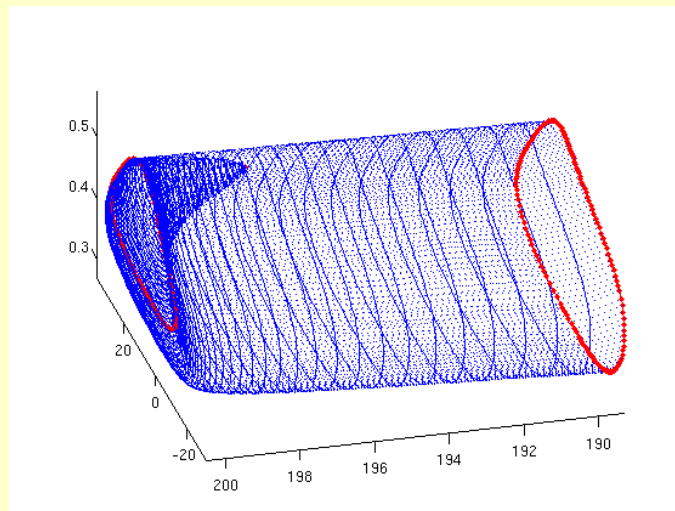
Limit Cycles from Hopf Bifurcation

- From the 5th singularity point (H), compute the emerging limit cycle by continuing the 1st parameter

```
x1=x(1:2,s(5).index);p=[x(end,s(5).index);6];
[x0,v0]=init_H_LC(@MyML,x1,p,[1],1e-6,40,4);
opt = contset(opt,'MaxNumPoints',100);
opt = contset(opt,'Multipliers',0);
opt = contset(opt,'Adapt',1);
opt = contset(opt,'MaxStepsize',5);
opt = contset(opt,'FunTolerance',1e-6);
opt = contset(opt,'VarTolerance',1e-6);
[x]c,v]c,s]c,h]c,f]c]=cont(@limitcycle,x0,v0,opt);
```

```
first point found
tangent vector to first point found
Limit point cycle (period = 1.227099e+01, pa
Normal form coefficient = 1.321124e-01
Limit point cycle (period = 2.254127e+01, pa
Normal form coefficient = -3.155002e-01
```

```
elapsed time = 353.0 secs
npoints curve = 100
```





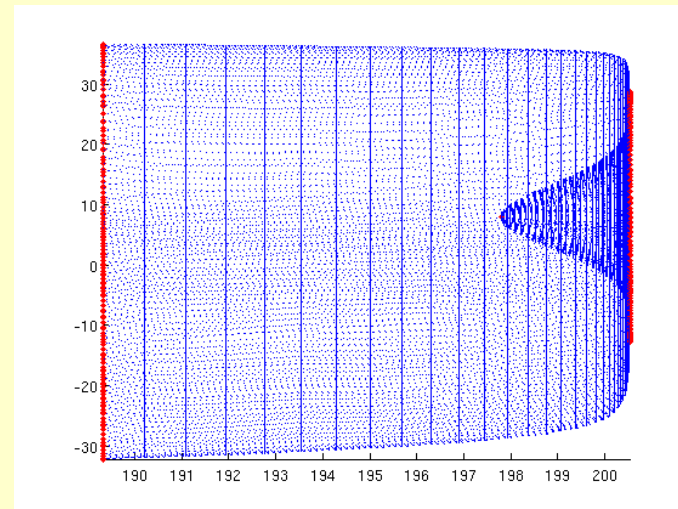
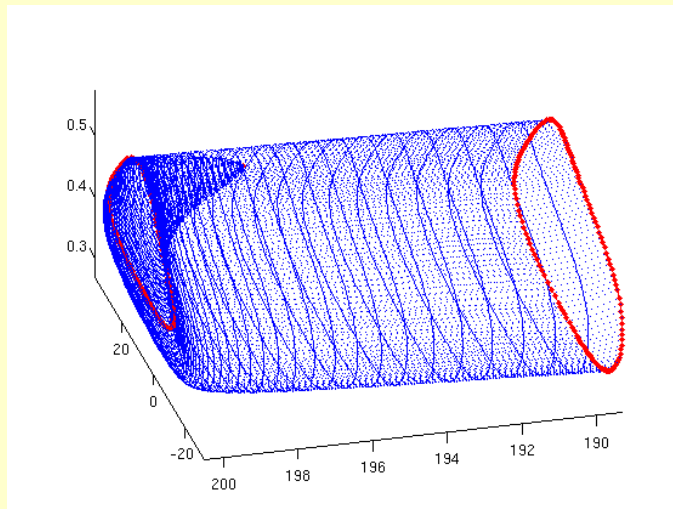
Limit Cycles from Hopf Bifurcation

- `init_H_LC`
 - initialize the computation of limit cycles emerging from Hopf point

```
x1=x(1:2,s(5).index);p=[x(end,s(5).index);6];
[x0,v0]=init_H_LC(@MyML,x1,p,[1],1e-6,40,4);
opt = contset(opt,'MaxNumPoints',100);
opt = contset(opt,'Multipliers',0);
opt = contset(opt,'Adapt',1);
opt = contset(opt,'MaxStepsize',5);
opt = contset(opt,'FunTolerance',1e-6);
opt = contset(opt,'VarTolerance',1e-6);
[xlc,vlc,slc,hlc,flc]=cont(@limitcycle,x0,v0,opt);
```

```
first point found
tangent vector to first point found
Limit point cycle (period = 1.227099e+01, pa
Normal form coefficient = 1.321124e-01
Limit point cycle (period = 2.254127e+01, pa
Normal form coefficient = -3.155002e-01
```

```
elapsed time = 353.0 secs
npoints curve = 100
```



Limit Cycles from Hopf Bifurcation

- Result of continuation: `[xlc, vlc, slc, ...]`
 - list of singular points: `slc`

```
slc =  
  
4x1 struct array with fields:  
    index  
    label  
    data  
    msg  
  
>> slc(3)  
  
ans =  
  
    index: 62  
    label: 'LPC'  
    data: [1x1 struct]  
    msg: 'Limit point cycle'
```



Numerical Algorithm for Solution Continuation and Bifurcation

- Pseudo–arclength continuation: consider smooth function

$$F : \mathbb{R}^{n+1} \rightarrow \mathbb{R}^n$$

we would like to trace the curve given by

$$F(\mathbf{x}, \alpha) = 0$$

where the scalar α is a free parameter

- Main idea: generate a sequence of points

$$X_i \equiv (\mathbf{x}_i, \alpha_i), \quad i = 1, 2, \dots$$

such that

$$\|F(X_i)\| \leq \epsilon, \quad \|\delta X_i\| \leq \epsilon'$$

- Suppose at point $X_i = (\mathbf{x}_i, \alpha_i)$ we have normalized tangent vector $V_i = (v_i, \eta_i)$:

$$F_X(X_i) V_i = 0, \quad \|V_i\| = 1$$

- Calculation of next point $X_{i+1} = (\mathbf{x}_{i+1}, \alpha_{i+1})$ consists of 2 steps:
 - prediction of new point
 - correction from the predicted point

Numerical Algorithm for Solution Continuation and Bifurcation

- Predictor step:

$$\tilde{X}_{i+1} = X_i + h V_i$$

for some sufficiently small step-size h

This linear approximation takes us off the curve; hence:

- Corrector step: a Newton-like procedure is used;

However, Newton's method can only be applied to systems with as many equations as unknowns

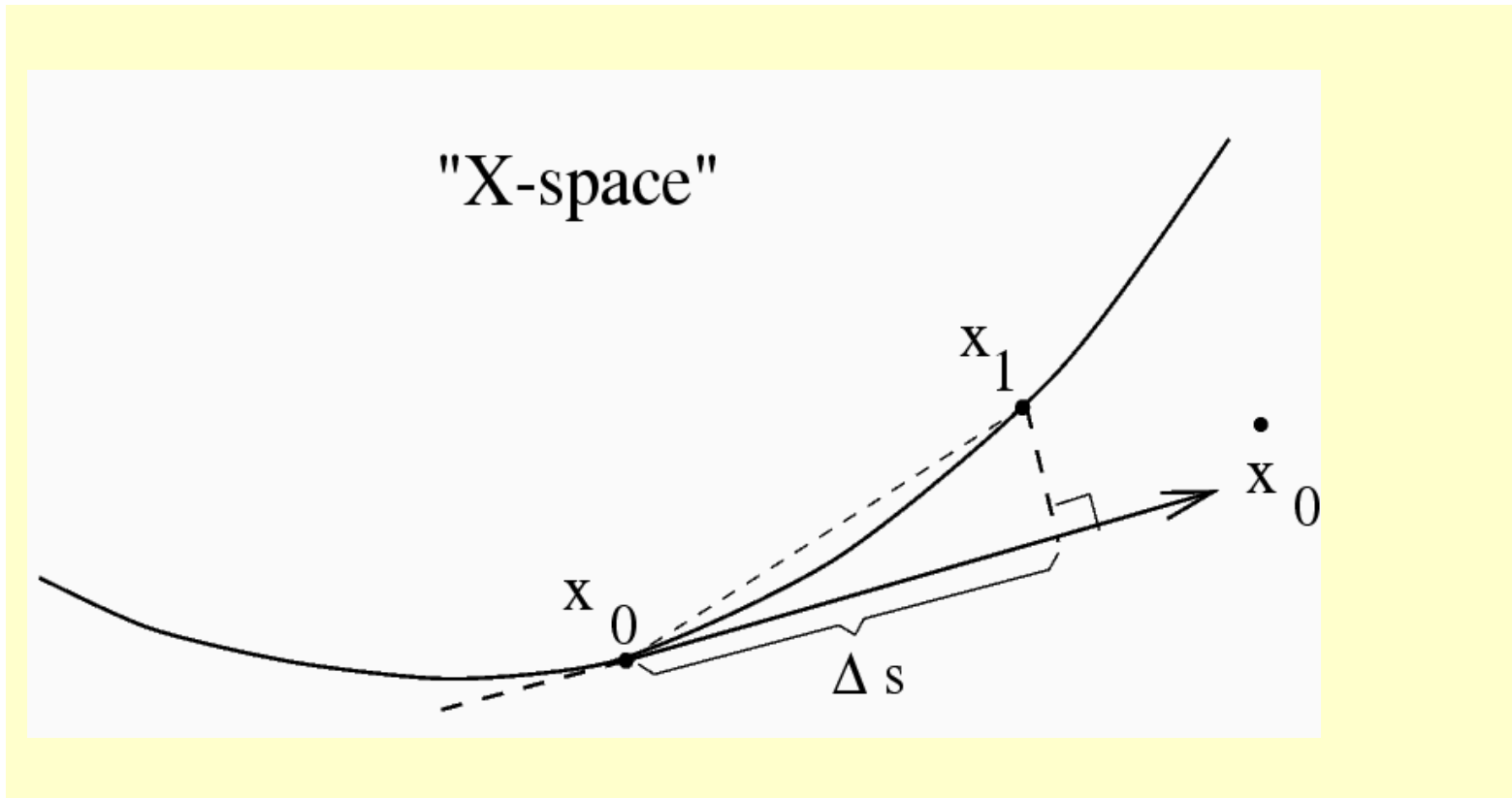
Need to append a scalar equation, giving rise to system:

$$\begin{cases} F(X) = 0 \\ g(X) = 0 \end{cases}$$



Numerical Algorithm for Solution Continuation and Bifurcation

- How to choose $g(\mathbf{X})$?
 - make the direction of the corrector step orthogonal to predictor step
 - $g(\mathbf{X}) = \langle \mathbf{X} - \tilde{\mathbf{X}}_{i+1}, \mathbf{V}_i \rangle$



Numerical Algorithm for Solution Continuation and Bifurcation

- Hence, the corrector steps $X_{C,k}$ consist of Newton iterations of the form

$$X_{C,0} = \tilde{X}_{i+1}$$

$$X_{C,k+1} = X_{C,k} - H_X^{-1}(X_{C,k}) H(X_{C,k}),$$

$$H(X) = \begin{pmatrix} F(X) \\ 0 \end{pmatrix}, \quad H_X(X) = \begin{pmatrix} F_X(X) \\ V_i^T \end{pmatrix}$$

- Having found new point X_{i+1} on the curve $F(X) = 0$, compute new tangent vector V_{i+1} that preserve curve

$$\begin{cases} F_X(X_{i+1}) V_{i+1} = 0 \\ \langle V_{i+1}, V_i \rangle = 1 \end{cases}$$

Numerical Algorithm for Solution Continuation and Bifurcation

- Having traced a curve solving $F(\mathbf{x}, \alpha) = 0$, we need to detect singularities on the curve (e.g., F
- Method: define smooth scalar **test functions** $\phi(\mathbf{x})$ which have simple zeros at singularity points o:
 - e.g., for Limit Point: $\phi(\mathbf{x}) = \det F_{\mathbf{x}}(\mathbf{x})$
- Locate singularities by finding the zeros of $\phi(\mathbf{x})$
 - by e.g., 1–dimensional secant method
- Singularity is detected if: $\phi(\mathbf{x}_i) \phi(\mathbf{x}_{i+1}) < 0$



Conclusions

- MATCONT is an ongoing effort, encompasses many capabilities of existing dynamical analysis software
- Build up bifurcation diagram by
 - performing solution continuation, and
 - bifurcation detection
 - continuation of bifurcation, and
 - singularity detection
- MATCONT has many other capabilities, including:
 - handling higher co-dimension bifurcations
 - computing normal form coefficients