分岐集合の数値計算

吉田勝俊

2007 年 12 月 11 日

1 力学系

一定の規則に従って時間変動するシステムを，力学系という．力学系の配位 (物体なら位置と向き) を表すのに必要な変数の組 $(x_1, x_2, \cdots, x_n) \in \mathbb{R}^n$ を，一般化座標という．一般化座標が張る数ベクトル空間を，配位空間という．一般化座標の時間微分を加味した，

$$(x_1, \dot{x}_1, x_2, \dot{x}_2 \cdots, x_n, \dot{x}_n) \in \mathbb{R}^{2n}$$

が張る数ベクトル空間を，相空間という．

時間変動が，n 連立の微分方程式，

$$\begin{cases} \dot{x}_1 = f_1(x_1, x_2, \cdots, x_n) \\ \dot{x}_2 = f_2(x_1, x_2, \cdots, x_n) \\ \qquad\vdots \\ \dot{x}_n = f_n(x_1, x_2, \cdots, x_n) \end{cases} \tag{1}$$

に支配される力学系を考える．解は相空間上の曲線となるが，これを相軌道という．ある瞬間の力学系の状態は相軌道上の 1 点に対応する．このような，右辺が時間 t を陽には含まない力学系を，**自律系**という．

これに対して，右辺が時刻 t を陽に含む系：

$$\begin{cases} \dot{x}_1 = f_1(x_1, x_2, \cdots, x_n, t) \\ \dot{x}_2 = f_2(x_1, x_2, \cdots, x_n, t) \\ \qquad\vdots \\ \dot{x}_n = f_n(x_1, x_2, \cdots, x_n, t) \end{cases} \tag{2}$$

を，**非自律系**という．具体例を挙げると，

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = -cx_2 - kx_1 - x_1^3$$

は自律系である．その一方で，

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = -cx_2 - kx_1 - \epsilon x_1^3 + A\cos\omega t$$

とか，

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = -cx_2 - kx_1 \cos\omega t$$

などは陽に t を含むので非自律系である．ようするに，非自律系には，x_i や $\dot{x}_i$ と全く無関係な時間変動が，外部から注入される．

2 自律系の平衡点

自律系 (1) を考える．そこに至ると時間変動が止まる相空間の点を，平衡点または不動点という．この定義より，平衡点では一般化座標の時間微分が全て 0 になる．

$$\dot{x}_1 = \dot{x}_2 = \cdots = \dot{x}_n = 0$$

これを自律系 (1) に代入すれば，平衡点が満すべき条件として，

$$\begin{cases} f_1(x_1, x_2, \cdots, x_n) = 0 \\ f_2(x_1, x_2, \cdots, x_n) = 0 \\ \qquad\vdots \\ f_n(x_1, x_2, \cdots, x_n) = 0 \end{cases}$$

が得られる．

次の 2 次元力学系を例題に，数値計算法を示そう．

$$\begin{cases} \dot{x}_1 = x_2, \\ \dot{x}_2 = -x_2 - px_1 - x_1^3 + F. \end{cases} \tag{3}$$

この系を，3 次の Duffing 系という．一般に，復元力が多項式で表わされる振動系を Duffing 系という．

2.1 平衡点の探索

自律系 (3) の平衡点は，停止条件 $\dot{x}_1 = \dot{x}_2 = 0$ より，

$$x_2 = 0, \quad -px_1 - x_1^3 + F = 0$$

を満足する．簡単のため $p = -1$, $F = 0$ としよう．

このとき，解くべき非線形方程式は，

$$x_1 - x_1^3 = 0$$

となるから，この場合の平衡点は手計算できる．

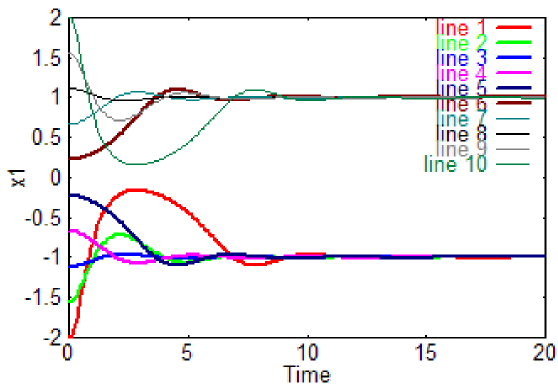
$$x_1(1 - x_1^2) = 0 \qquad \therefore x_1 = 0, \pm 1$$

以上，この系には平衡点が 3 個ある．

この例題は幸運にも手で解けたが，一般の非線形方程式は手では解けないので，数値計算の実例を示していく．

■数値積分の欠点　平衡点とは，現象的には定常状態であるから，元の微分方程式 (3) の数値解を求めて観察すれば，いずれ平衡点に収束しそうである．

```
function dx = eqn(x, t)
  global P;
  dx(1) = x(2);
  dx(2) = -x(2) + x(1) - x(1)**3;
endfunction
t=linspace( 0.0, 20.0, 200 );
initp=linspace( -2.0, 2.0, 10 );
all=[];
for i=1:10
  sol = lsode( "eqn", [initp(i); 0.0], t );
  all=[all, sol(:,1)];
endfor
xlabel("Time");
ylabel("x1");
plot(t,all);
```



このように，この方法の欠点として，3 つある平衡点のうち $x = \pm 1$ は検出できるが，$x = 0$ が検出できない．理由は後述するように，$x = \pm 1$ が**安定平衡点**，$x = 0$ が**不安定平衡点**だからである．不安定な平衡点 $x = 0$ は，自然状態では発現しない[*1]．自由状態の振り子を例にとると，下死点が安定平衡点，上死点が不安定平衡点である．

数値積分法は，このような物理現象を忠実にシュミレートしてしまうので，不安定平衡点の数値解もまた，通常の観測には掛からない[*2]

■作図による方法　そこで少々泥臭いが，平衡点の方程式のグラフを書く方法を紹介しておく．ようするに，(速度の項を零においた) 静的な復元力,

$$F(x_1) = -(x_1 - x_1^3)$$

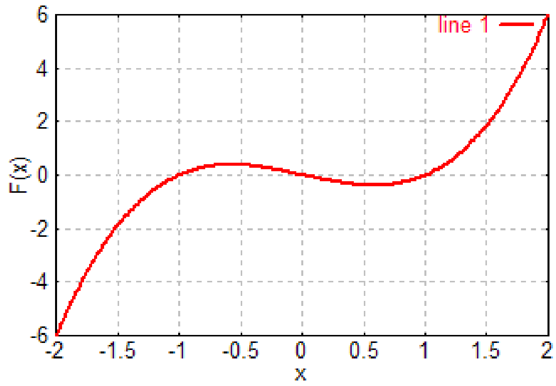
のグラフを書いて[*3]，x_1 軸との交点を求めれば，それが平衡点である．

```
function y = F(x)
  y = -(x - x.**3);
endfunction
x=linspace( -2.0, 2.0, 200 );
xlabel("x"); ylabel("F(x)"); grid;
plot(x,F(x));
```

[*1] ただし，クーロン摩擦などがあるときは，不安定平衡点に静止する状態が安定に観察できる．

[*2] 不安定平衡点が求まる数値計算法も存在するが (Shooting 法など)，プログラムの微調整に熟練が必要である．

[*3] これを復元力とみたときに辻褄が合うように，マイナスを付けたが，x_1 軸との交点に影響はない．



x_1 軸との交点は，やはり 3 点ある．数値で求めるには $F(x)$ が十分小さくなる x を拾ってやればよかろう．

```
function y = F(x)
  y = -(x - x.**3);
endfunction
n=5000; eps=1e-3;
x=linspace( -2.0, 2.0, n );
fp0=[];
for i=1:n
  if( abs(F(x(i))) < eps )
     printf( "%e %e\n", x(i), F(x(i)) );
     fp0 = [fp0, x(i)];
  endif
endfor
save fp0.dat fp0
```

これを実行すると，(必要なら n や eps を調整しながら)

```
-9.998000e-01 3.999600e-04
-4.000800e-04 4.000800e-04
4.000800e-04 -4.000800e-04
9.998000e-01 -3.999600e-04
```

のような出力を得る．計算結果を収めたベクトル変数 fp0 の内容は fp0.dat というファイルに保存される．$x = -1, 0, 1$ 付近の点が出力されており，粗いながらも，平衡点の推定値が得られている．

■ニュートン法　作図による推定値の精度は $|F(x)| < 10^{-4}$ 程度だが，これらを「磨く」ために，**ニュートン法**を使おう．Octave にはそれ用に fsolve という関数が用意されている．

```
load fp0.dat
dn=length(fp0);
function y = F(x)
  y = -(x - x.**3);
endfunction
fp=[];
for i=1:dn
  fp(i)=fsolve("F", fp0(i));
  printf( "%e %e\n", fp(i), F(fp(i)) );
endfor
save fp.dat fp
```

これを実行すると，

```
-1.000000e+00 -0.000000e+00
0.000000e+00 -0.000000e+00
0.000000e+00 -0.000000e+00
1.000000e+00 -0.000000e+00
```

となり，計算機精度の範囲で正確な平衡点が求まる．ここでは幸運にも，厳密解と同じ数値が得られたが，通常は 10^{-9} 程度の精度なら満足することになる．

■まとめ 最初からニュートン法を使えばよさそうなものだが，そうできない理由がある．ニュートン法は，平衡点の精度を「磨く」用途には適しているが，平衡点を見付ける用途には適していないのである．これはよく知られたニュートン法の欠点で，ニュートン法の初期値，

```
fp(i)=fsolve("F", fp0(i));
                  ^^^^^^ 初期値
```

を，厳密解から離れた値にセットすると，ニュートン法は厳密解に到達できない．

全ての平衡点を漏れなく探索できる数値解法があれば理想だが，現状では，数値積分や作図，またはその他のアイデアで，臨機応変に平衡点を見付けていくしかない．特に数値積分が不安定平衡点を見逃すことには，十分留意すべきである．いったん見付かりさえすれば少々精度が悪くても，ニュートン法で磨ける．

2.2 平衡点の安定性

■第 1 変分 $y = f(x_1, \cdots, x_n)$ という関数関係において，x_i を独立変数，y を従属変数と呼んだ．ここで，独立変数 x_i を一斉に，ちょっとだけずらそう．

$$x_i \mapsto x_i + \xi_i \quad (i = 1, 2, \cdots, n) \tag{4}$$

$\xi_i \ll 1$ は微小な変数である．このとき，特別な場合を除けば，従属変数 y もズレるはずだ．

従属変数 y に起こるズレ η を，ごく自然に，

$$\eta := f(x_1 + \xi_1, \cdots, x_n + \xi_n) - f(x_1, \cdots, x_n) \tag{5}$$

と定義する．ここで，独立変数のズレ ξ_i が十分に微小で，関数 $f(x_1, \cdots, x_n)$ が全微分可能ならば，従属変数のズレ η は，独立変数のズレ ξ_i の一次式によって，次のように近似計算できる．

$$\eta \approx \frac{\partial f}{\partial x_1}\xi_1 + \cdots + \frac{\partial f}{\partial x_n}\xi_n \tag{6a}$$

1 次近似式 (6a) を，$y = f(x_1, \cdots, x_n)$ の**第 1 変分**という．定義式 (5) を用いれば，近似式 (6a) を，

$$\begin{aligned} &f(x_1 + \xi_1, \cdots, x_n + \xi_n) \\ &\qquad \approx f(x_1, \cdots, x_n) + \frac{\partial f}{\partial x_1}\xi_1 + \cdots + \frac{\partial f}{\partial x_n}\xi_n \end{aligned} \tag{6b}$$

と表記しておくこともできる．

■ヤコビ行列 自律系 (1) にズレ (4) を代入すると，

$$\begin{aligned} &(x_i + \xi_i)^{\cdot} = f_i(x_1 + \xi_1, \cdots, x_n + \xi_n) \\ &\implies \underline{\dot{x}_i} + \dot{\xi}_i = p\underline{f(x_1, \cdots, x_n)} \\ &\qquad\qquad + \frac{\partial f_i}{\partial x_1}\xi_1 + \cdots + \frac{\partial f_i}{\partial x_n}\xi_n \quad \because (6\mathrm{b}) \end{aligned}$$

となるが，下線部は元の式 (1) だからキャンセルして，ズレだけの微分方程式，

$$\dot{\xi}_i = \frac{\partial f_i}{\partial x_1}\xi_1 + \cdots + \frac{\partial f_i}{\partial x_n}\xi_n$$

が得られる．$i = 1, \cdots, n$ について全部並べて，行列表記すると，

$$\begin{bmatrix} \dot{\xi}_1 \\ \dot{\xi}_2 \\ \vdots \\ \dot{\xi}_n \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}}_{A} \begin{bmatrix} \xi_1 \\ \xi_2 \\ \vdots \\ \xi_n \end{bmatrix} \tag{7}$$

となるが，これを，自律系 (1) の**第 1 変分方程式**という．行列部分 A を，自律系 (1) の**ヤコビ行列**という．

自律系 (1) を，太字のベクトル表記，

$$\boldsymbol{x} := \begin{bmatrix} x_1 \\ \vdots \\ x_2 \end{bmatrix}, \quad \boldsymbol{f}(\boldsymbol{x}) := \begin{bmatrix} f_1(x_1, \cdots, x_n) \\ \vdots \\ f_n(x_1, \cdots, x_n) \end{bmatrix}$$

によって，$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$ と書くとき，ヤコビ行列 A は $\dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$ とか $D_{\boldsymbol{x}}\boldsymbol{f}$ と表記されることが多い[*4]．

■線形化系の安定性 試しに Duffing 系，

$$\begin{cases} \dot{x}_1 = x_2, \\ \dot{x}_2 = -x_2 + x_1 - x_1^3. \end{cases}$$

の第 1 変分方程式を求めてみよう．P は定数とする．式 (7) に当てはめると，ヤコビ行列は，

$$\begin{bmatrix} \frac{\partial(\text{第 1 式の右辺})}{\partial x_1} & \frac{\partial(\text{第 1 式の右辺})}{\partial x_2} \\ \frac{\partial(\text{第 2 式の右辺})}{\partial x_1} & \frac{\partial(\text{第 2 式の右辺})}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 - 3x^2 & -1 \end{bmatrix} \tag{8}$$

となり，第 1 変分方程式は，

$$\begin{bmatrix} \dot{\xi}_1 \\ \dot{\xi}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 - 3x^2 & -1 \end{bmatrix} \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} \tag{9}$$

となる．ようするに，Duffing 系に与えるズレ ξ が十分に小さければ，ズレ ξ の時間発展は線形力学系 (9) で近似される．このことから，第 1 変分方程式を**線形化方程式**もしくは**線形化系**と呼ぶことがある．

線形力学系の安定性は，システム行列[*5]の固有値で判別できた．Duffing 系の線形化系 (9) のシステム行列はヤコビ行列 (8) だが，その成分には，元の Duffing 系の変位 x が含まれている．ズレ ξ の安定性は，x に何を代入するかで変化する．

この問題について，次の定理が知られている[*6]．

[*4] $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$ は式 (7) に出てきた行列の略記である．ベクトルでベクトルを偏微分する方法を独自にあみだす必要はない．

[*5] $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ の A のこと．

[*6] 線形化系の固有値実部が零のときには，**分岐**という非線形現象が起こり，定理が成立しなくなるのだが，詳しくは後述する．

- 非線形力学系と線形化系の解軌道が似るのは，平衡点 $x = \bar{x}$ の近くである．

この定理を前提に，平衡点まわりの線形化系の固有値を求める．

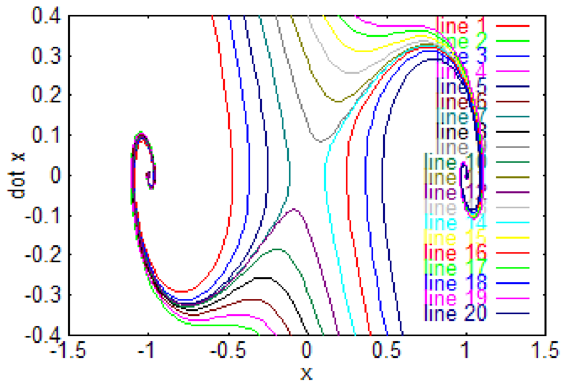
```
load fp.dat #fp
fixp=fp; fp=[]; fp=fixp([1,2,4]);
function A = jac( x )
  A = [0, 1; 1-3*x**2, -1];
endfunction
ev(1,:)=eig(jac(fp(1)))';
ev(2,:)=eig(jac(fp(2)))';
ev(3,:)=eig(jac(fp(3)))';
fp'
ev
save fp_ev.dat fp ev
```

実行すると，

```
> octave -q DufFixpEig.m
ans =
  -1
   0
   1
ev =
  -0.50000 - 1.32288i  -0.50000 + 1.32288i
  -1.61803 + 0.00000i   0.61803 + 0.00000i
  -0.50000 - 1.32288i  -0.50000 + 1.32288i
```

となる．平衡点 $x = -1, 1$ は固有値 $-0.50000 \pm 1.32288i$ より安定フォーカス，平衡点 $x = 0$ は固有値 -1.61803, 0.61803 よりサドルであることが分かる．以下，元の Duffing 系の解軌道を示す．

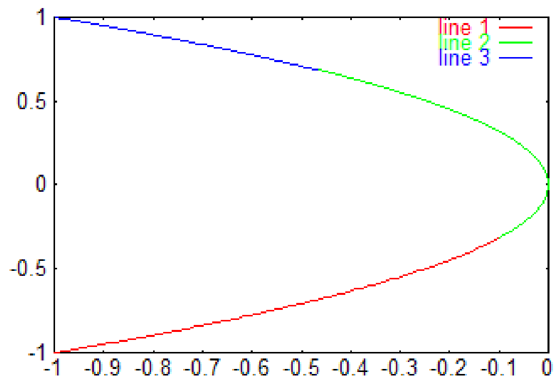
```
function dx = eqn(x, t)
  dx(1) = x(2);
  dx(2) = -x(2) + x(1) - x(1)**3;
endfunction
t=linspace( 0.0, 20.0, 200 );
ip1=linspace( -0.6, 0.3, 10 );
ip2=linspace( -0.3, 0.6, 10 );
x=[]; dx=[];
for i=1:10
  sol = lsode( "eqn", [ip1(i); 0.4], t );
  x=[x, sol(:,1)]; dx=[dx, sol(:,2)];
  sol = lsode( "eqn", [ip2(i); -0.4], t );
  x=[x, sol(:,1)]; dx=[dx, sol(:,2)];
endfor
xlabel("x");
ylabel("dot x");
plot(x,dx); pause;
```



2.3　平衡点の接続

■平衡点の追跡　パラメータ p を少しずつ変えながらニュートン法を繰り返すと，平衡点 x をパラメータの関数 $x(p)$ として描ける．得られたグラフを**分岐図**という．

```
load fp.dat
global x p; p=-1;
function y = F(x,p)
  y = p*x + x.**3;
endfunction
function y = F_x(x)
  global p;
  y = F(x,p);
endfunction
parn = 101;
bifpar(1,:)=linspace(-1,-0.1,parn); # p
x0=fp(1); bif(1,:)=[]; # result
for i=1:parn
  p = bifpar(1,i); x0=fsolve("F_x", x0);
  bif(1,i) = x0;
endfor
plot( bifpar(1,:), bif(1,:) );
function y = F_p(p)
  global x;
  y = F(x,p);
endfunction
p0 = bifpar(1,parn); # last param
x0 = bif(1,parn);    # last fixp
bifpar(2,:)=linspace( x0, x0+1, parn ); # x
bif(2,:)=[]; # result
for i=1:parn
  x = bifpar(2,i); p0=fsolve("F_p", p0);
  bif(2,i) = p0;
endfor
plot( bifpar(1,:), bif(1,:), bif(2,:), bifpar(2,:) );
p0 = bif(2,parn);    # last param
x0 = bifpar(2,parn); # last fixp
bifpar(3,:)=linspace(p0,-1,parn); # p
bif(3,:)=[]; # result
for i=1:parn
  p = bifpar(3,i); x0=fsolve("F_x", x0);
  bif(3,i) = x0;
endfor
plot( bifpar(1,:), bif(1,:), bif(2,:), bifpar(2,:),\
      bifpar(3,:), bif(3,:) );
input("hit any key");
fpset1(:,1) = [bifpar(1,:),bif(2,:),bifpar(3,:)]';
fpset1(:,2) = [bif(1,:),bifpar(2,:),bif(3,:)]';
save fpset1.dat fpset1
plot( fpset1(:,1), fpset1(:,2) ); pause;
```



グラフが垂直に立ってくると追跡困難になるので，途中で掃引方向を変更し，平衡点 x を少しずつ変えながらパラメータ p を求めて，関数 $p(x)$ を描いている．

Exercise 1 平衡点 $x = 0$ について，$p = -1$ から 1 まで接続せよ．

プログラム例を示す．

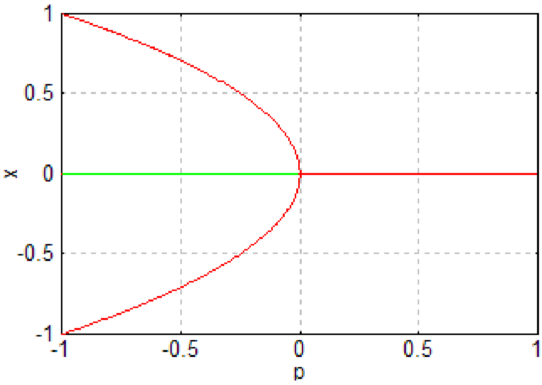
```
load fp.dat
global x p; p=-1;
function y = F(x,p)
  y = p*x + x.**3;
endfunction
function y = F_x(x)
  global p;
  y = F(x,p);
endfunction
parn = 301;
bifpar(1,:)=linspace(-1,1,parn); # p
x0=fp(2); bif(1,:)=[]; # result
for i=1:parn
  p = bifpar(1,i); x0=fsolve("F_x", x0);
  bif(1,i) = x0;
endfor
plot( bifpar(1,:), bif(1,:) );
input("hit any key");
fpset2(:,1) = bifpar(1,:)';
fpset2(:,2) = bif(1,:)';
load fpset1.dat
plot( fpset1(:,1), fpset1(:,2), \
      fpset2(:,1), fpset2(:,2) );
save fpset2.dat fpset2
pause;
```

■平衡点の安定判別 固有値実部が負となる安定平衡点と，それ以外の不安定平衡点を分離して，色分けしてプロットする方法を示す．

```
load fpset1.dat
load fpset2.dat
function A = jac( x, p )
  A = [0, 1; -p-3*x**2, -1];
endfunction
function [s, u] = stab(fpset)
  x_s = x_u = p_s = p_u =[];
  for i=1:rows(fpset)
    p = fpset(i,1);
    x = fpset(i,2);
    ev_real = real(eig(jac(x,p)));
    if ( ev_real(1) < 0 )
      p_s=[p_s, p]; x_s=[x_s, x];
    else
      p_u=[p_u, p]; x_u=[x_u, x];
    endif
  endfor
  s=[]; u=[];
  if ( length(p_u) > 0 )
    u=[u, [p_u;x_u]];
  endif
  if ( length(p_s) > 0 )
    s=[s, [p_s;x_s]];
  endif
endfunction

[s1,u1]=stab(fpset1);
[s2,u2]=stab(fpset2);
fpset_s=[s2,s1]; fpset_u=[u1,u2];
grid on;
xlabel("p"); ylabel("x");
plot( s1(1,:), s1(2,:), "1",\
      s2(1,:), s2(2,:), "1",\
      u2(1,:), u2(2,:), "2" ); pause;
```

※ u1 は空なので，プロットしようとするとエラーになる．



■ピッチフォーク分岐 以上の結果を，$p = 1$ から $p = -1$ の方向にたどると，次のことがいえる．

- $p > 0$ では安定平衡点 $x = 0$ のみが存在する．
- $p = 0$ において，$x = 0$ が不安定化し，$x = 0$ の上下に新たな安定平衡点が生じる．

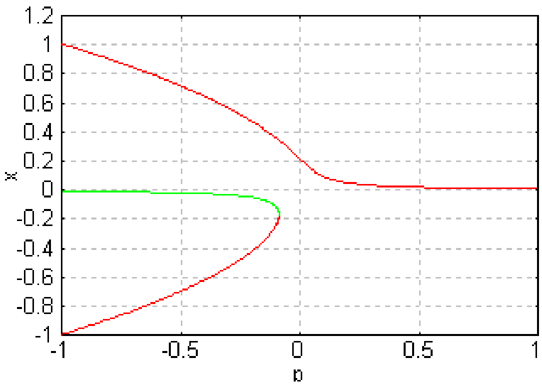
このような変化の仕方を，超臨界ピッチフォーク分岐 (super critical pitchfork bifurcation) という．

一般に，平衡点の性質や個数が変化する現象を**分岐現象**といい，分岐が起こる境界点を**分岐点**という．以上の例では，$p = 0$ が分岐点である．

■サドル・ノード分岐 これまでの例では，以下の Duffing 系において $F = 0$ としてきた．

$$\begin{cases} \dot{x}_1 = x_2, \\ \dot{x}_2 = -x_2 - px_1 - x_1^3 + F. \end{cases}$$

Exercise 2 $F \neq 0$ のときに分岐図がどう変化するか，確認せよ．



このような $F \neq 0$ のときに生じる分岐現象を，サドル・ノード分岐という．サドルノード分岐点では，サドルと安定ノードの組が消失する．

2.4 分岐点の接続

■分岐点の判定 各平衡点 $\bar{\boldsymbol{x}}$ のヤコビ行列 $Df(\bar{\boldsymbol{x}})$ の固有値 λ:

$$P(\lambda) := \det(Df(\bar{\boldsymbol{x}}) - \lambda I) = 0$$

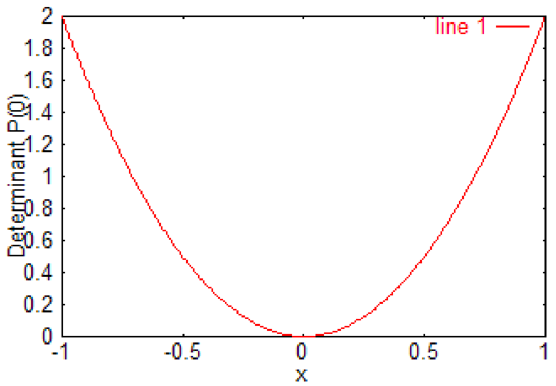
を計算すると，分岐図が垂直に切り立つ点で実部が零 0 になる．これを**分岐点**という．

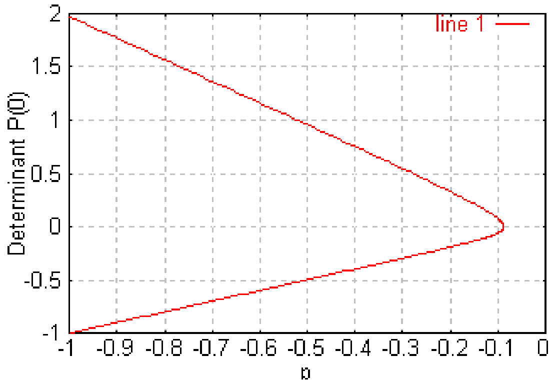
分岐点の特殊な場合として，虚部も同時に消えて固有値が 0 になるときは，行列式の性質によって，

$$P(0) = \det(Df(\bar{\boldsymbol{x}})) = 0$$

が成立する．この条件に該当する分岐点として，**ピッチフォーク分岐点**や**サドル・ノード分岐点**がある．

```
load fpset1.dat;  #DufCont1.m
load fpset2.dat;
fps = fpset1;
function A=jaco( x, p )
 A=[0, 1; -p-3*x**2, -1];
endfunction
p=fps(:,1); x=fps(:,2);
det_jac=[];
for  i=1:rows(x)
  det_jac(i) = det(jaco(x(i),p(i)));
endfor
ylabel("Determinant P(0)");
xlabel("x"); plot( x, det_jac ); pause;
xlabel("p"); plot( p, det_jac ); pause;
```





■分岐点の接続 平衡点の方程式と $P(0) = 0$ を連立して解く．