

MATCONT: short reference

Frank Schilder

Principle use of MATCONT

1. Setting up the equations file.
2. Initialising a continuation.
3. Computing a solution branch.

A continuation can be initialised with a known initial solution or at a bifurcation point (branch-switching). All initialisation functions have names of the form

`init_<point-type>-<solution-type>`

where *point-type* is the type of the starting point and *solution-type* is the type of solutions of the branch to be computed. The allowed combinations are listed in the sections “Initialising ...”

Point types / solution types	
EP	equilibrium point
LC	limit cycle
Bifurcations of equilibria	
LP	limit point (saddle-node)
BP	branch point (pitchfork / transcritical)
H	Hopf-bifurcation point
Bifurcations of limit cycles	
LPC	limit point of cycles
BPC	branch point of cycles
NS	Neimark-Sacker bifurcation point
PD	period-doubling bifurcation point

The equations file (model file)

Example 1: Pitchfork normal form

$$\dot{x} = \mu x - x^3$$

The right-hand side of an ODE is defined in a so-called *model file*. A template for such a model file is:

```
pitch.m
function out = pitch
out{1} = @init;
out{2} = @fun_eval;
out{3} = []; % @jacobian;
out{4} = []; % @jacobianp;
out{5} = []; % @hessians;
out{6} = []; % @hessiansp;
out{7} = []; % @der3;
out{8} = []; % @der4;
out{9} = []; % @der5;
% out10= []; % define user functions here
```

```
% -----
function dydt = fun_eval(t,x,mu)
dydt(1,1) = mu*x - x^3;

% -----
function [tspan,y0,options] = init
y0=[0,0];
options = odeset;
handles = feval(pitch);
tspan = [0 10];
```

The model file contains additional information used by MATCONT. We have to edit only the body of the function `fun_eval`, which evaluates our right-hand side.

Example 2: Hopf normal form

$$\begin{aligned}\dot{x} &= \mu x - y - x(x^2 + y^2) \\ \dot{y} &= x + \mu y - y(x^2 + y^2)\end{aligned}$$

We change only `fun_eval` and one line in `init`:

```
hopf.m (changed lines only!)
function out = hopf
[...]
function dydt = fun_eval(t,x,mu)
dydt(1,1) = mu*x(1) - x(2) - x(1)*(x(1)^2+x(2)^2);
dydt(2,1) = x(1) + mu*x(2) - x(2)*(x(1)^2+x(2)^2);
[...]
handles = feval(hopf);
```

Initialising equilibrium point continuation

`[x0,v0] = init_EP_EP(@mf, x0, p0, #ap)`

Initialise continuation at known equilibrium point.

Arguments: mf name of model file, x0 initial equilibrium point, p0 initial parameter value, #ap index of continuation parameter (active parameter)

Return values: [x0,v0] initial point and tangent

`[x0,v0] = init_BP_EP(@mf, x0, p0, s, h)`

Initialise continuation at branch point (BP).

Arguments: mf name of model file, x0 branch point, p0 parameter value of branch point, s BP structure (returned by `cont`), h initial step-size

Computing branches of equilibria

`[x,v,s,h,f]=cont(@equilibrium,x0,v0,opt)`

Compute branch of equilibrium points.

Arguments: x0,v0 return values of `init` function (see above), opt options structure created with `contset`

Return values: x list of points along branch, format: $x = \{x_i, \mu_i\}$, $i = 1, \dots, \text{end}$, v tangents at the points along branch, s list of special points (initial point, bifurcation points, end point), f values of user functions at points along branch

Example: computation of branch of equilibria of Example 1, starting at $x = 0$ and $\mu = -1$. At $\mu = 0$ a branch-point is located. At this point we switch branches in a subsequent continuation.

```
pfdemo.m
% setting some options (see manual p. 14f)
opt=contset;
opt=contset(opt,'MaxNumPoints',30);
opt=contset(opt,'Singularities',1);

% run first continuation, start at x=0, mu=-1
[x0 v0]=init_EP_EP(@pitch, [0], [-1], [1]);
[x v s h f]=cont(@equilibrium,x0,v0,opt);
cpl(x,v,s,[2 1]);

% run second continuation, start at branch-point
% x = x1, mu = m1 (index 2 in s)
x1 = x(1:end-1,s(2).index);
m1 = x(end,s(2).index);
[x0 v0]=init_BP_EP(@pitch, x1, m1, s(2), 0.01);
[x v s h f]=cont(@equilibrium,x0,v0,opt);
cpl(x,v,s,[2 1]);
```

The branch-switching to a limit-cycle continuation is shown in “Computing branches of limit cycles” below.

Plotting results (equilibrium points)

`cpl(x, v, s, e)`

Plot bifurcation diagram in two or three dimensions.

Arguments: x,v,s return values of `cont` (see above), e indices of components of x to plot (two indices for 2D-plot and three for 3D-plot)

The function `cpl` always adds to a plot. Use `clf` to clear a figure.

Initialising limit cycle continuation

`[x0,v0] = init_H_LC(@mf, x0, p0, #ap, h, ntst, ncol)`

Initialise continuation at Hopf bifurcation point.

Arguments: mf name of model file, x0 Hopf point, p0 parameter value of Hopf point, #ap index of continuation parameter (active parameter), h initial step-size,

ntst number of mesh-intervals, **ncol** number of collocation points per mesh-interval

Return values: **[x0,v0]** initial limit cycle and tangent

The choices of **ntst** and **ncol** depend on the problem. A good choice for **ncol** is 4. The number of mesh-intervals **ntst** should be as small as the problem permits. Start with **ntst** = 5 and increase slowly when necessary.

```
[x0,v0] = init_PD_LC(@mf, x, s, ntst, ncol, h)
```

Initialise continuation at period-doubling bifurcation point.

Arguments: **mf** name of model file, **x** branch of periodic solutions returned by **cont**, **s** “special-point structure” of period-doubling point, **ntst** number of mesh-intervals, **ncol** number of collocation points per mesh-interval, **h** initial step-size

Return values: **[x0,v0]** initial limit cycle and tangent

The number **ntst** may need to be doubled at a period-doubling bifurcation point.

```
[x0,v0] = init_BPC_LC(@mf, x, v, s, #ap, ntst, ncol, h)
```

Initialise continuation at branch point.

Arguments: **mf** name of model file, **x,v** branch of periodic solutions as returned by **cont**, **s** “special-point structure” of branch point, **#ap** index of continuation parameter (active parameter), **ntst** number of mesh-intervals, **ncol** number of collocation points per mesh-interval, **h** initial step-size

Return values: **[x0,v0]** initial limit cycle and tangent

One can usually keep the number **ntst** constant at branch points.

```
[x0,v0] = init_LC_LC(@mf, x, v, s, #ap, ntst, ncol)
```

Initialise continuation with previously computed limit cycle.

Arguments: **mf** name of model file, **x,v** branch of periodic solutions as returned by **cont**, **s** “special-point structure” as returned by **cont**, **#ap** index of continuation parameter (active parameter), **ntst** number of mesh-intervals, **ncol** number of collocation points per mesh-interval

Return values: **[x0,v0]** initial limit cycle and tangent

This function can be used to adapt the number **ntst** when necessary.

Limit cycles of forced ODEs

The current command-line version of **MATCONT** does not provide a mechanism for starting directly at a limit cycle that is known *a-priori* or by simulation.

Computing branches of limit cycles

```
[x,v,s,h,f]=cont(@limitcycle,x0,v0,opt)
```

Compute branch of limit cycles.

Arguments: **x0,v0** return values of **init** function (see above), **opt** options structure created with **contset**

Return values: **x** list of periodic solutions together with parameter and period along branch, format: **x(:,i) = [x_{i,1}, x_{i,2}, ..., x_{i,ntst-ncol+1}, T_i, μ_i]**, $i = 1, \dots, \text{end}$, T_i is the period of solution i (that is, of $x_i = x(1:\text{end}-2, i)$), **v** tangents at the points along branch, **s** list of special points (initial point, bifurcation points, end point), **h** internal data of the continuation algorithm, **f** values of user functions at points along branch

Example: computation of branch of equilibria of Example 2, starting at $(x, y) = (0, 0)$ and $\mu = -1$. At $\mu = 0$ a Hopf-bifurcation point is located. At this point we switch branches in a subsequent continuation and compute a branch of limit-cycles.

```
hdemo.m
% setting some options (see manual p. 14f)
opt=contset;
opt=contset(opt,'MaxNumPoints',30);
opt=contset(opt,'Singularities',1);

% run first continuation, start at
% (x,y) = (0,0), μ = -1
[x0 v0]=init_EP_EP(@hopf, [0;0], [-1], [1]);
[x v s h f]=cont(@equilibrium,x0,v0,opt);
cpl(x,v,s,[3 1 2]);

% changing some options (see manual p. 14f)
opt=contset(opt,'InitStepsize',0.1);
opt=contset(opt,'MaxStepsize',0.2);

% run second continuation, start at Hopf-point
% (x,y) = x1, μ = m1 (index 2 in s)
x1 = x(1:end-1,s(2).index);
m1 = x(end,s(2).index);
[x0 v0]=init_H_LC(@hopf, x1, m1, [1], 0.01, 5, 4);
[x v s h f]=cont(@limitcycle,x0,v0,opt);
```

```
hold on;
plotcycle(x([1:end-2 end],:),v,s, ...
    [size(x,1)-1 1 2]);
hold off;
```

The branch-switching is similar to the previous example, but there is no need to separate the solution components manually. Suppose a continuation returned **x**, **v**, **s**, **h**, and **f** and a period-doubling bifurcation is detected as the third special point. Then branch-switching is done like in

```
code fragment for branch-switching at period-
doubling
[...]
[x v s h f]=cont(@limitcycle,x0,v0,opt);

[x0 v0]=init_PD_LC(@hopf, x, s(3), 10, 4, 0.01);
[x v s h f]=cont(@limitcycle,x0,v0,opt);
[...]
```

Plotting results (limit cycles)

```
plotcycle(x([1:end-2 end],:),v,s, e)
```

Plot periodic orbits in three dimensions (parameter+two space coordinates).

Arguments: **x,v,s** return values of **cont** (see above), **e** indices of components of **x** to plot, the first one is always the parameter, that is, **e(1)=size(x,1)-1**, the second and third one are phase-space coordinates

The function **plotcycle** creates a new plot. Use **hold** to add to an existing figure.

Computing solution measures is more complicated. The computation of $\max_t \{x(t)\}$ and $\min_t \{x(t)\}$ can be done as in

```
hdemo.m (modification: solution measures)
[m points] = size(x); % # of continuation steps
dim = 2; % dimension of ODE

% max and min of x along periodic solutions
xx = x(1:end-2,:);
xx = reshape(xx, [dim (m-2)/dim points]);
xa = max(squeeze(xx(1,:,:), []), 1);
xi = min(squeeze(xx(1,:,:), []), 1);

mu = x(end,:);

cpl([mu; xa; zeros(size(xa))],v,s,[1 2 3]);
cpl([mu; xi; zeros(size(xa))],v,s,[1 2 3]);
view(2);
```