Matcont Tutorial: ODE GUI version

Hil Meijer
Nottingham, Oct-Nov, 2011

“If you want to get credit for solving a complicated mathematical problem, you will have to provide a full proof. But if you’re trying to make something as easy as possible, you want to make it foolproof—so simple even a fool couldn’t screw it up.”

1 Introduction

This tutorial tries to explain the basics of how to use the numerical bifurcation package MATCONT by going through an example. We aim at explaining how you can use the software so we assume a basic knowledge of bifurcation theory. As with many things one learns, you will only master it by performing the procedures yourself. So hands on!

We try to cover the following procedures in this tutorial

• Installation, including steps on installing a compiler
• System definition
• Simulation
• Continuation of equilibria in one parameter
• Continuation of codim 1 bifurcations of equilibria in two parameters
• Starting Limit Cycle from a Hopf point
• Starting Limit Cycle from a time simulation
• Computing PRCs and their derivatives
• Computing homoclinic orbits

This is a condensed overview, see the lab session 1-5 and the manual at sourceforge or 2 for more.

2 Getting started/Installation

Go to sourceforge.net/projects/matcont/ (go to matlab, 3rd entry) and download the latest version of matcont, at the time of writing matcont4p2. That is: go to Browse all files|matcont|matcont4p2 and select the zip file. Extract it and in MATLAB change your working directory to the newly created matcont4p2 directory. From the command line you now type ‘matcont’ to start it. If you receive complaints about a compiler, type ‘mex -setup’ on the command line and choose the lcc compiler, the Matlab compiler, start matcont again.

1 Every one computer is different from the other. So, although these instructions have been checked step by step, do not be surprised if your computations take a different direction. Just try again but now in the other direction.

For periodic orbits we use compiled C-code to speedup the computation of the Jacobians. Over the years, the support of MATLAB for compilers has reduced now relying on manual installations. We will try to implement vectorization in the near future. For now the situation is as follows

1. Windows XP (or Vista/7 with 32-bits processor), works fine, run mex -setup and choose the lcc compiler.
2. Windows 7: You will need to install Visual Studio 2010 Express and SDK 7, as indicated in http://www.mathworks.com/support/compilers/R2010b/win64.html
3. MAC: If at mex -setup you can select gcc without problems, proceed. Otherwise you have to install the XCode package.
4. Linux: this has its gcc compiler properly installed on most distributions.

![Figure 1: A typical matcont startup screen](image)

Now several windows will open with a standard system called adapt2. A typical screenshot is shown in Figure 1. One window is titled matcont and has several menu options. For instance, to end your matcont session, choose ‘Select’ in the matcont window and then ‘Exit’. Hereafter we will indicate this with matcont:Select|Exit. Alternatively one clicks on the close button of the matcont window.

3 Equilibria in Lorenz84

In this tutorial we will investigate some simple equilibrium bifurcations in the Lorenz 84 model, which can be seen as a metaphor of the large-scale atmospheric circulation, given by

\[
\begin{align*}
    x' &= -y^2 - z^2 - ax + aF, \\
    y' &= xy - bzx - y + G, \\
    z' &= bxy + xz - z.
\end{align*}
\]  

\( (1) \)
3.1 Specifying a new model

To specify the differential equations of the system click ‘\((\text{matcont}):\text{Select|Systems|New}\)’ and a window titled ‘System’ appears with fields ‘Name system’, ‘Coordinates’, ‘Parameters’, ‘Time’, options for derivatives and a large box. The exact appearance of the System window depends on the operating system and the availability of the Matlab Symbolic Toolbox. If the latter is present, the lowest line will read ‘symbolically’. If possible, one should always choose to compute first, second and third order derivatives symbolically as this is the most reliable and fastest option. If you want to change the system or correct the input choose ‘\((\text{matcont}):\text{Select|Systems|Edit/Load}\)’, select the system you want to edit and press ‘Edit’.

Common mistakes (apart from simple typos):

- Make sure not to put spaces between variables or parameters in their input fields
- Make sure that multiplication is written explicitly with * or you get errors like ‘Unbalanced or misused parentheses or brackets’ or ‘error using == eval, Index exceeds matrix dimensions’.
- If your model has some strange behavior like ever increasing variables, check whether you have specified the differential equations in the same order as the order of the coordinates.

3.2 Time Simulations

Click ‘\((\text{matcont}):\text{Type|Initial point|Point}\)’ to initialize the computation of an orbit starting from a point. In the matcont window the curve type is now P_O, every curve type has a similar meaning. Two extra windows ‘Starter’ and ‘Integrator’ appear, the first is to specify initial conditions/points, the second to select the numerical integrator and change its settings. In the starter window set $a = .25, b = 4, F = .5, G = .5$. We also want to see how the orbits look like and for this we

---

3 The first and second are mostly used in the continuation, the third order is useful for computing normal forms accurately.
open a window ‘(matcont):Window|Plot|2D-plot’. In ‘Layout|Variables on axes’ select $x$ as abscissa(standard) and $y$ as ordinate, press ‘OK’ and let the axes range from 0 to .5 for the abscissa and -.5 to .5 for the ordinate using ‘(2Dplot:1):Layout|Plotting region’. In ‘Integrator’ change ‘Interval’ to 50 and ‘MaxStepsize’ to 0.1.

![Figure 3: A spirally converging orbit](image)

After these preparations we are ready to compute some orbits: Select ‘(matcont): Compute|Forward’. Starting from $(x,y,z) = (0,0,0)$ converges to $(-.1344588, .359872257, -.170597270)$. You can monitor the orbit and its final values by opening via numeric window. Select ‘(matcont): Window|Numeric’ and recompute the orbit. You can try also other initial points and check that these also converge to this point. So we have an apparent global attractor.

### 3.3 Continuation of Equilibria and their bifurcations

We are now going to study the effect on the stability of the equilibrium when varying either $F$ or $G$. Select the last point via ‘(matcont):Initial point’ and then choose the last point on the previous curve. Change the curve type to EP_EP by selecting ‘(matcont):Type|Initial point|Equilibrium’. Activate the parameter $F$ by clicking on the button next to it. In the 2Dplot:1 window change the abscissa to the parameter $F$ via ‘(2Dplot:1):Layout|Variables on axes’. Also change the plotting region for $F$ to 0 to 2 and for $y$ to -1 to 1.

The stability changes upon the variation of $F$ thus we would also like to see how the eigenvalues of the Jacobian at the equilibrium develop. For this we select ‘(Numeric):Window|Layout’ and highlight the option ‘eigenvalues’ (it changes to capitals) and press ‘OK’. Select ‘(matcont): Compute|Forward’.

**NOTE:** In the lower left corner a small window appears with buttons ‘Pause’, ‘Resume’ and ‘Stop’. Whenever a bifurcation is detected, MATCONT pauses and shows some relevant information. In the sequel a few such points will be detected, but we will not always indicate to resume the computation.

At $F = 1.1345161$ the message ‘Limit point’ appears in the status field of the matcont window. Here one eigenvalue has zero real part (Check that in the numeric window!) and the curve(branch) of equilibria has a turning point, on one side the equilibria are stable on the other unstable. Press the ‘Resume’ button in the lower left of your screen or press space to continue the computation of the now unstable branch. At $F = 0.73193833$ another Limit point is found and the unstable branch turns into a stable one. When $F = 1.0757564$ is reached two complex conjugate eigenvalues have zero real part and a Hopf bifurcation occurs. The 2Dplot should now look like Figure 4.

If the computation goes to fast you can select to pause the computation of the EP_EP curve.
after each computed point. For this choose ‘(matcont):Options|Pause|At each point’. (Do not 

forget to change it to ‘At special points’ after this computation.

Now we turn to the command line, i.e. the Matlab window, where you most probably now 

have the following information:

first point found
tangent vector to first point found
label = LP, x = ( 0.085554 0.479610 0.179485 1.134516 )
a=-9.177168e-001
label = LP, x = ( 0.461768 0.072707 0.249512 0.731938 )
a=-4.844850e-001
label = H , x = ( 1.015103 -0.000458 0.123138 1.075756 )
First Lyapunov coefficient = -9.496611e+000
elapsed time = xxx secs
points curve = 300

The first two lines indicate that the continuation has started, and the last two that it has ended. 

Any other messages are displayed here, for instance the three bifurcations together with their 

normal form coefficients. This information is useful as it indicates the non-degeneracy of the 

bifurcations. First, the two Limit Point bifurcations are shown, x indicates the three variabels 

x, y, z and the value of the active parameter F of the bifurcation point. Then the normal form 

coefficients are given, note that for a Limit Point the sign of this coefficient is not unique (Why?). 

Finally, we know that the limit cycle born from the Hopf bifurcation is stable as the first Lyapunov 

coefficient is negative.

Figure 4 showed a phenomenon called hysteresis or bistability. This is an important feature of 

nonlinear (differential) equations. In parameter space regions of bistability are usually delimited 

by a wedge of Limit point bifurcations which we will compute here.

Select one of the Limit points found in the equilibrium continuation as initial point ‘(matcont):Select|Initial 

point’. Activate F and G as active parameters. Close the 3D-plot and re-open the 2D-plot.
Change the ordinate of the 2D-plot to the parameter $G$ and its range to 0 to 2. Now select ‘(matcont):Compute|Forward’ and when finished also ‘Backward’. At $(F, G) = (0.4663649, 0.2919545)$ a Cusp bifurcation (CP) and at $(F, G) = (1.6840517, 1.6829686)$ a Zero-Hopf bifurcation is detected. You have now computed the wedge within which you have bistability.

Finally we also compute a curve of Hopf bifurcations: select a Hopf point analogously as for the Limit point as initial point. Change the curve type to $H_H$ via ‘(matcont):Type|Curve|Hopf’. Take here $F$ and $G$ as active parameters. Press ‘Compute|Forward’ and ‘Compute|Backward’ to obtain the diagram as in Figure 5.

![Figure 5: Bifurcation curves in the $(F,G)$-plane with codimension 2 points.](image)

### 4 Continuation of Limit Cycles from a Hopf point

We are now going to compute a branch of limit cycles starting from the Hopf point. Select the Hopf point using ‘(matcont):Select|Initial point’ as new initial point.

Activate the parameter $F$ and the period by clicking on the buttons in the Starter window. In the starter window set Monitor Singularities for Period-doubling to Yes. 4 Close the 2DPlot(for speed reasons:plotting the cycles takes too much time now) and ‘(matcont):Compute|Forward’. Notice that at $F = 7.129405$ and $F = 12.8991$ a Period-Doubling bifurcation is detected. This is part of another tutorial and we will not look at it here. You can however look at the command line to see the normal form coefficient of the Period-Doubling bifurcation and interpret it, or select one of the PD points change curve type to Period-doubling and continue the PD bifurcation with $F, G$ as free parameters.

Resume the computation and wait until it has finished. Open a 3D plot with ‘(matcont):Window|Plot|3D-Plot’. Choose $OX=z$, $OY=F$, $OZ=y$ in the draw3_attributes window. Change the plotting the region to $z, y \in [-2, 2], F \in [0, 15]$. Your 3D-plot will resemble the one in Figure 6.

---

4By default, and for good reasons (many spurious otherwise), detection of bifurcations is switched off.
5 Continuation of Bifurcations in 2 Parameters

5.1 Starting from a codim 2 point

Now we have also the option to start a special bifurcation curve from the Zero-Hopf point straight-away. The ZH-point has codimension two and a codimension 1 Neimark-Sacker emanates here. Select the ZH point as initial point and Curve|Neimark-Sacker. Now check that $F$ and $G$ are the two free parameters, select to detect R4,R3,R2 along the curve, set MaxStepsize to 0.3 and Compute|Forward. This saves a lot of time! It is known not to work if the bifurcating periodic orbit changes direction quickly or when the normal form coefficients are very small or very large. Then one has to compute the bifurcation point with higher accuracy, use symbolic derivatives and play with the amplitude see 5.

6 Visualizing your output

We have already plotted some of the computations. It is also possible to load the output and plot and manipulate it yourself. We will give a nontrivial example of plotting the profile.

6.1 Plotting the profile

To plot the profile, i.e. the evolution of a variable during the whole period, is nontrivial as it is stored in a condensed way. Suppose we know how many mesh-points were used. This can be determined from the f variable, simply look up the first entry equal to 1. Now we need

- the index of the periodic orbit we want to plot: say ii.
- the profile of one variable, say the jj-th: $x(jj:\text{dim:end-2,ii})$
- the time-mesh $\tau=\tau_0<\tau_1<...<\tau_{\text{ntst}}=1$.
- the period $T=x(\text{end-1,ii})$ to scale the mesh back from $[0,1]$ to $[0,T]$

---

So we could plot:

```cpp
ntst=20;ii=4;jj=1;ndim=3;
plot(f(1:ntst+1,ii)*x(end-1,ii),x(jj:ndim:end-2,ii))
```

You will get an error message because the computed curve \( x \) contains also data about the orbit between mesh-points. While you are trying to plot the profile only coarsely at the mesh-points. A first correction would be

```cpp
ntst=20;ii=250;jj=1;ndim=3;ncol=4;
plot(f(1:ntst+1,ii)*x(end-1,ii),x(jj:ndim*ncol:end-2,ii))
```

This is usually not good enough, so we actually want the detailed grid. For this we need to know that between to mesh-points the orbit is stored at equi-distant time-points \( t_{i,j} = t_i + \frac{1}{m}(\tau_{i+1} - \tau) \), \( j = 0, 1, ..., m \). So the correct commands are:

```cpp
ii=240;jj=1;ndim=3;ntst=20;ncol=4;
mesh=sort([reshape(repmat(f(1:ntst,ii),1,ncol)+...
    repmat((0:ncol-1)/ncol,ntst,1).*repmat(diff(f(1:ntst+1,ii)),1,ncol),ntst*ncol,1); 1]);
plot(mesh*x(end-1,ii),x(jj:ndim:end-2,ii))
```

## 7 Starting Limit Cycle from a time simulation

To make this really motivating we look at a periodically forced system made autonomous. In the main matcont window: Select | System | New and enter all fields for the SEIR model. Note that you will be working with an equivalent system \( s = \log S \) for numerical stability. Select symbolic derivatives of order 1 and 2. Press ok.

We will start by integrating to observe periodic dynamics. Press matcont: Type | Point and a starter and integrator window will appear. We specify initial values in the starter window: \( ss=-3, ee=-3, ii=-3, x=1, y=0 \) \( \alpha=35.482, \beta=5000, \gamma=100, \delta=.2, \mu=.02, \omega=6.28 \). The system is still sensitive so you may have to decrease MaxStepSize to .01 in the integrator window. Also increase the Interval (integration time) to 20. Open a graphic window with matcont: Window | Graphic | 2Dplot. Select the variables ss and ee on the axes. Adjust the Layout | Plotting region to \(-10\) for both axes. Press matcont: Compute | Forward. You will see that after some initial excursions it will settle to a periodic orbit.

You could inspect the \( x, y \)-dynamics by changing abscissa and ordinate via Layout | variables on axes to see a circle.

At the bottom of the starter window we see a tempting button Select Cycle. We cannot yet push it. What the algorithm does is to find the point of the trajectory nearest to the initial value. If it finds one, it takes that the time between these points as the period and fits a mesh to the computed solution. We proceed as follows: matcont: Select | Initial point and select the last point on the curve we have just computed. In integrator window change the interval to 1 and press matcont: Compute | Forward. Select Plot | Clear and Plot | Redraw curve to see that it is a periodic orbit.

Now press the Select Cycle button in the starter window. A popup will appear with two fields: "Tolerance" equal to 1e-2 and "test intervals" equal to 20. Change test intervals to 40 and press OK. Tick the boxes for beta0 and omega as free parameters in the new starter window and select "yes" for Period-doubling under the heading monitor singularities. Change MaxStepsize to 5 in the Continuer window. Open a Window | Numeric window and close the figure. Press matcont: Compute | Forward.
If everything went well, you found two period-doubling bifurcations. Note as well in the Numeric window that omega is now really equal to $2\pi$. Select one of them as initial point. Change \texttt{Type|Curve} to Perioddoubling and select beta0 and delta as free parameters.

8 Phase Response Curves

We assume some knowledge about PRCs and synchronization. This part only shows how to compute them. We will take the classical Hodgkin-Huxley equations. Make a new system, copy+paste the equations from the template and select 1st and 2nd order derivatives. Press \texttt{ok}. Select \texttt{Type|Initial Point|Point} and open a graphic 2D plot. Select time for the horizontal axis. Adjust the plotting region to $[0, 100] \times [-90, 50]$ and the interval in the integrator to 100 as well. Set $I_{stim}$ equal to 20 and press \texttt{Compute|Forward} and see some nice spikes, i.e. a periodic orbit here.

In this example we could start LC continuation from a Hopf bifurcation, but that takes way too much time as it is far away. So, as before let us take the last point via \texttt{Select|Initial Point} of the simulated curve and an estimate by the eye of the period says that we should change the Interval to 12. Press \texttt{Select Cycle} in the Starter window and accord with standard options.

In the Starter window you can now see at the bottom (scroll if necessary) a heading Calculate Phase Response with three fields. We will set the input to 1.0. The amplitude of the input does not really matter as this is the linearized response. Set the PRC and dPRC fields to \texttt{Yes}. Also set the Period and $I_{stim}$ as active parameters. Two windows will appear and press \texttt{Compute|Forward} to see a family of PRCs being computed as a byproduct of the continuation.

- If you are interested in just one PRC, just set MaxNumPoints in the Continuer window to 1.
- The derivative is also computed as it is related to phase-locking and synchronization, see \textsuperscript{6}.

9 Connecting Orbits

In the following instructions we will see how to start the computation of homoclinic orbits. The main difficulty is to get good starting data for the continuation. We will show two methods that may be useful to accomplish this.

9.1 Homoclinic orbits starting from a limit cycle

Sometimes when following a limit cycle you will notice that the parameters do not change anymore. Meanwhile the period goes to infinity. Typically, in a graphic window plotting the continuation parameter versus the period you see something as in Figure 7right. In such cases you may suspect that you are close to a connecting orbit. You will have to look at the orbit to be sure. But suppose you notice that the orbit is close to a steady state at some point, wouldn’t it be nice to take this as starting data for the computation of connecting orbit? This method is implemented in Matcont for homoclinic orbits only. Along a limit cycle we can make a guess for an equilibrium by checking at which point of the cycle the vectorfield has smallest norm. This goes well for a single equilibrium, but not for two.

We will do this for the Lorenz84 system, which you already specified in Matcont. If it is not the present selected system (check the main matcont window), you can select it via \textsuperscript{6}W. Govaerts and B. Sautois, Phase Response Curves, Delays and Synchronization in Matlab, Lecture Notes in Computer Science 3992 (2006) 391-398.
Figure 7: Left: a suitable initial orbit for finding nearby homoclinic orbits. Right: The period blows up while the parameter $G$ does not change anymore...

**System|Edit/Load.** Alternatively you specify the system again:

\[
\begin{align*}
x' &= -y^2 - z^2 - ax + aF, \\
y' &= xy - bxz - y + G, \\
z' &= bxy + xz - z. \\
\end{align*}
\]  

(2)

We will fix $a = .25$ and $b = 4$ as before. To have a good starting limit cycle we first do a simulation.²

Specify **Type|Initial Point|Point** and in the Starter window set $x = .1, y = 1, z = .3$ and $F = 3, G = .8$. In the Integrator window set Interval to 50. Open **Window|2D graphic** with $x$ between 0 and 2 and $y$ between -1.5 and +1.5. If you **Compute|Forward** you will get Figure 7left.

Now via the main window **Select|Initial Point** the last point of the orbit you just computed. Change the interval in the Integrator window to 10 and **Compute|Forward**. If you **redraw the curve in the 2Dplot (In the 2Dplot window there is an option Plot|Clear/Redraw curve) , it looks like we have an interesting limit cycle. So in the Starter window press **Select Cycle** and change the number of test intervals to 40. Select the Period and $G$ as free parameters (tick the boxes) and in the Continuer window set MaxStepsize to 1.0. Open a numeric window and check that the parameters and Period are displayed. Press **Compute|Backward** and during the continuation observe that $G$ first goes up and settles to $G_{\text{final}}=1.018893$. If not, select **Compute|Forward**. You can zoom in (where?!) to see that we the limit cycle might be close to a saddle-focus. At the end of the continuation, change the variables in the 2Dplot via **Layout|Variables on axes** to $G$ and Period for the abscissa and ordinate, respectively, and change **Layout|Plotting region** the range to $[0.5, 1.5] \times [0, 100]$ to obtain Figure 7.

Select the last point of the LC, LC curve just computed as initial point. Next select **Curve|Homoclinic to saddle** and select $F, G$ and eps1 as free parameters. ² In the 2Dplot change the axes to $F$ and $G$ with ranges $[0,5]$ and $[0,2.5]$ respectively. Set Maxstepsie to 3 in the continuer window and **Compute|Forward**. You will obtain a small piece of a homoclinic bifurcation curve shown in Figure 8. You can extend the computation at least once.

³The limit cycle we get from starting at the Hopf bifurcation does not get close to a homoclinic bifurcation. So we have to find another.

⁴It turns out that eps1 is the correct auxiliary parameter to free as the stable eigenvalues are complex. This is the hardest part for the continuation. In general, you may need to try all three of T,eps0,eps1 sequentially.
We see that the homoclinic bifurcation wiggles towards the ZH-point. Here two homoclinic curves come together. As a final remark we mention that you could start the continuation of the other homoclinic bifurcation by the same procedure but that at the beginning step in the limit cycle continuation you follow the branch for decreasing $G$.

### 9.1.1 A dirty trick

Sometimes you do not get this method to work, but still you have this limit cycle with very large period at parameters close enough to the homoclinic bifurcation. So you could also do a continuation of the limit cycle keeping the period fixed and continue in two system parameters. You will get a good indication where the homoclinic bifurcation is in parameter space. However, there can be a large discrepancy between this continuation result and the actual homoclinic bifurcation.

### 9.2 Connecting orbits by homotopy

For connecting orbits, homoclinics and heteroclinics, we will follow lab session 5 written by Yuri Kuznetsov, at sourceforge see lab5.pdf in files–matcont–matcont3p3.