PARALLEL BIFURCATION ANALYSIS OF NONLINEAR CONTROL SYSTEMS

M.N. Demenkov
Nonlinear Dynamics & Control Lab (№ 16)
Institute of Control Sciences RAS
65 Profsoyuznaya, Moscow, 117997, Russia
E-mail: demenkov@ipu.ru

Numerical bifurcation analysis has emerged in 1970s as a valuable component in the analysis of mathematical models described by nonlinear differential equations. Up to date it is usually performed using the so called continuation technique, based on curve following using Newton's method. The presence of parallel computer architecture, like today's multi-core processors or GPU's, may allow us to use different kind of bifurcation analysis techniques that were not popular in the past due to their great computational demands. In this paper, we discuss how bifurcation analysis can be effectively parallelized using piecewise-affine approximations of nonlinear dynamical systems.

1. Introduction

The application of bifurcation analysis [1] to engineering control problems began in the early 1980s. So far, numerical bifurcation analysis has been employed many times for the analysis of particular engineering systems, although its industrial applications remain quite limited. In several publications bifurcation analysis has been posed as a method for post-design verification of modern control laws as well as a supplementary tool for their design [2].

We consider a nonlinear system in the following form:

$$\frac{dx}{dt} = f(x, p, u), \tag{1}$$

where x – the vector of state variables, p – the vector of system parameters and u – the control vector.

The valuable information about the behaviour of the system (1) can be obtained from the investigation of *invariant manifolds* of system trajectories. The simplest manifold is well known *equilibrium points* for which $\frac{\partial x}{\partial t}$=0. Another example is a *limit cycle*, which can be defined as a closed trajectory of the system. The existence of more complex manifolds (like torus) is also possible. These manifolds can be stable or unstable, that is, a manifold can attract all system trajectories in the neighbourhood of the manifold or repel it. As a result, these manifolds are called *attractors* or *repellers*. If the number, type or stability properties of these manifolds have been changed by varying parameter vector p, we say that a *bifurcation* occurs at some value of p.

There are exist software packages, like AUTO [3] or MATCONT [4], which can compute dependencies of the invariant manifolds characteristics on p (usually p is one- or two-dimensional) for a system in the form of (1). These dependencies (*bifurcation diagrams*) are usually computed using the so called continuation technique based on curve following using Newton method. Unfortunately, quite often in realistic industrial applications these software packages behave not quite well.

To allow wider application of bifurcation analysis in control engineering, several problems have to be solved, including taking into account realistic control constraints on amplitude and rate of control system actuators and other non-smooth characteristics in the system. Initially, differential equations representing engineering mathematical models were considered without non-smooth effects to allow application of classical numerical continuation methods. Also, continuation

methods that have been used for bifurcation analysis cannot find all invariant manifolds that exist in the system because, as it follows from their name, they can only trace those solutions that have been already found and the solutions that evolve from them.

2. Main results

Mathematically, we want to approximate the system (1) (having parameter vector p fixed) by the following piecewise-affine one:

$$\frac{dx}{dt} = A_i x + B_i u + f_i, \text{ if } G_i \begin{bmatrix} x \\ u \end{bmatrix} \le h_i, \; i = \overline{1, N}. \tag{2}$$

As one can see, we have divided our state-space (combined with control space) into polyhedral cells, which are bounded by hyperplanes. For each cell, we collect all such hyperplanes in matrix G_i and vector h_i, where each row represents one hyperplane. As a result, we can decide on cell to which the system state and a control belong, using a system of linear inequalities.

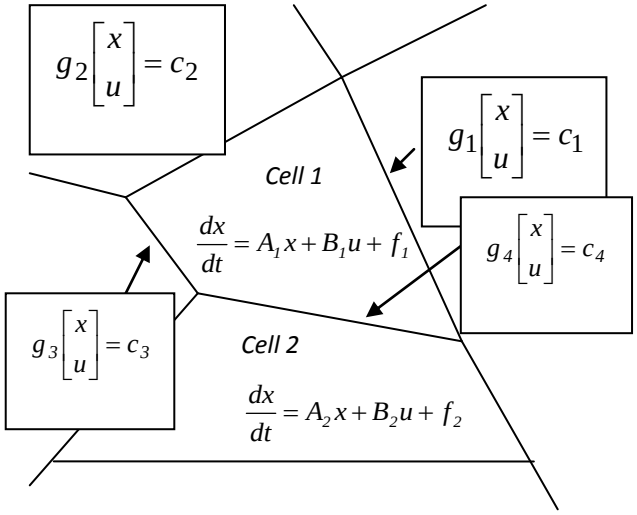

Figure 1. An example of piecewise-affine system

Take a look at Fig.1, where hyperplanes that constitute *Cell 1* are described by row vectors $g_{1\text{-}4}$ and scalars $c_{1\text{-}4}$. If the *Cell 1* contains the origin $\begin{bmatrix} x \\ u \end{bmatrix} = 0$ and all $c_{1\text{-}4} > 0$, then the following system of linear inequalities is satisfied for each point $\begin{bmatrix} x \\ u \end{bmatrix}$ in the *Cell* 1:

$$G_1 \begin{bmatrix} x \\ u \end{bmatrix} \le h_1, \text{ where } G_1 = \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \end{bmatrix} \text{ and } h_1 = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix}.$$

Outside the *Cell* 1, these inequalities do not hold.

We use a simple method of determining all equilibrium points of the system (1) (for a given p) that is based on its piecewise-affine approximation (2):

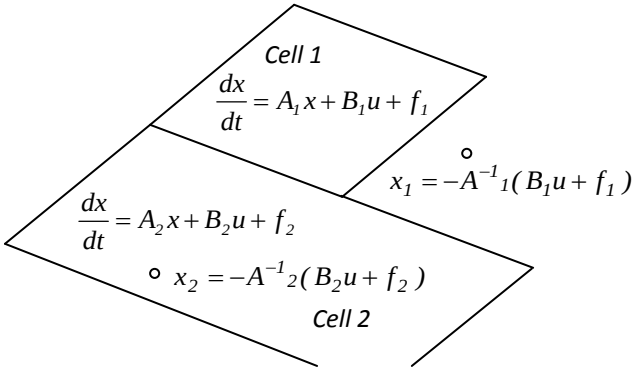

Figure 2. Determining equilibrium states: x_2 is the true equilibrium state, while x_1 is not.

This method can be applied both to search for equilibrium (if we consider original equations (2)) or to search of limit cycles, if we consider (2) as piecewise-linear approximation of a so called Poincare section [1]. Note that in Fig.1, one can assume $u=f(x)$ or 0 to incorporate control feedback.

Given u_0, for each cell determine the solution x_i of the following linear equation:

$$A_i x_i = -B_i u_0 - f_i .$$

If the solution satisfies the following inequalities:

$$G_i \begin{bmatrix} x_i \\ u_0 \end{bmatrix} \le h_i ,$$

then x_i is an equilibrium point for the piecewise-linear system; otherwise it is not.

The stability properties of an equilibrium located in the interior of a cell can be obtained from investigation of the eigenvalues of matrix A_i. If all eigenvalues of the matrix have negative real parts, the equilibrium is stable; if at least one eigenvalue has positive real part, it is unstable. However, if the equilibrium state belongs to a boundary between two cells, the question about stability becomes more complex. In this case, we might deduce stability using the original system (1), i.e. linearizing it in the neighbourhood of the equilibrium state.

For example, imagine that matrices A_1 and A_2 in Fig. 2 are invertible. Then x_2 is the true equilibrium state, while x_1 is not because it lies outside the *Cell 1*.

One can vary parameters in (1), build a piecewise-affine approximation for each value of p and in this way draw some kind of bifurcation diagram. This method has some advantages: it guarantees that all solutions will be determined, even if some branches of solutions are not connected with each other. The disadvantage is that we need to approximate the original system by piecewise-affine one for every change of parameters, but here is the part where today's parallel techniques may come to rescue. The effective parallelization is dependent on the achievable number of parallel threads in multi-processor or GPU system and can be done because of independence between different cells for the search procedure.

3. Numerical example

Consider a simple aeromechanical system [5] in the form of free-to-roll delta wing mounted on a fixed sting in a wind tunnel. The equations are as follows:

$$\frac{d\beta}{dt} = p\sin\theta_0$$
$$\frac{dp}{dt} = k(C_{l\beta_0}\beta + C_{lp_0}p + C_{lv})$$
$$\tau_1\frac{dC_{lv}}{dt} = -C_{lv} + k_2 C_{l\beta_0} f(\beta),\quad f(\beta) = \frac{\beta}{1+\beta^2}$$

These equations contain only one nonlinear function of one variable *f(β)*, so their piecewise-affine approximation is trivial. There are no control inputs, and hence we consider only state-space, which we cut into slices along β-axis. Let us denote as β_i, $i=\overline{1,N}$ gradual values of β so that

$$\beta_{i+1} = \beta_i + \Delta\beta.$$

We can define *Cell i* as the cell for which $\beta_i \le \beta \le \beta_{i+1}$. For each *i*-th cell the matrices in (2) are as follows (note that the matrix B_i and parts of the matrices G_i and h_i are omitted, since we have no control inputs):

$$A_i = \begin{bmatrix} 0 & \sin\theta_0 & 0 \\ kC_{l\beta_0} & kC_{lp_0} & k \\ \dfrac{k_2 C_{l\beta_0}}{\tau_1}\Delta f_i & 0 & -\dfrac{1}{\tau_1} \end{bmatrix},\ f_i = \begin{bmatrix} 0 \\ 0 \\ \dfrac{f(\beta_i) - k_2 C_{l\beta_0}\Delta f_i \beta_i}{\tau_1} \end{bmatrix},\ G_i = \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix},\ h_i = \begin{bmatrix} -\beta_i \\ \beta_{i+1} \end{bmatrix},$$

where $\Delta f_i = \dfrac{f(\beta_{i+1}) - f(\beta_i)}{\Delta\beta}$.

Fig. 3 shows the values of β at equilibrium points as a function of parameter k_2. Other parameters are k=0.008, θ_0=0.5, $C_{l\beta_0}$=-0.4 and C_{lp_0}=-0.1. One can see the so-called pitch-fork bifurcation at k_2=-1. The original system was split into piecewise-affine pieces with $\Delta\beta$=0.2, $-4 \le \beta_i \le 4$.

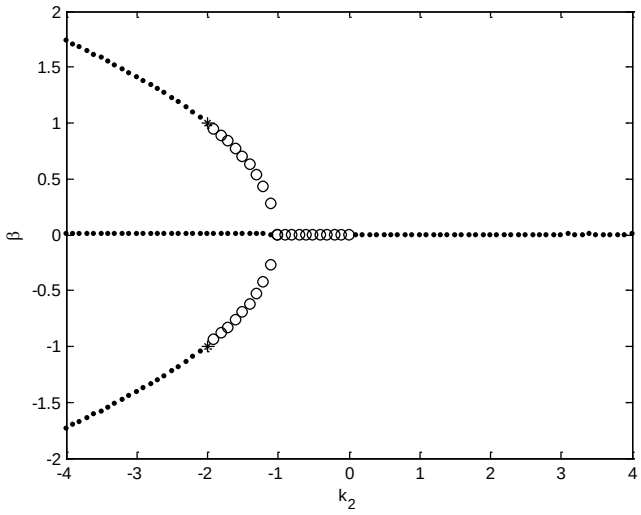

Figure 3. Bifurcation diagram of equilibrium values for β vs. parameter k_2. Circles mark stable equilibriums, while dots mark unstable ones.

References

[1] Kuznetsov, Y. (2004). Elements of Applied Bifurcation Theory. Springer, New York.
[2] Chen, G., Hill, D., and Yu, X. (eds.) (2003). Bifurcation Control: Theory and Application. Springer, Berlin.
[3] Doedel, E. (1981). AUTO: a program for the automatic bifurcation analysis of autonomous systems. Congressus Numerantium, 30, 265–284.
[4] Govaerts, W., Dhooge, A., and Kuznetsov, Y. (2003). MATCONT: a Matlab package for numerical bifurcation of ODE's. ACM Transactions on Mathematical Software, 29, 141–164.
[5] Goman M.G., Krabrov A.N. and Khramtsovsky A.V.(2002) Chaotic dynamics in a simple aeromechanical system. In: Blackledge J.M., Evans A.K. and Turner M.J. (eds.) *Fractal geometry: mathematical methods, algorithms, applications*. Horwood Publishing, Chichester.