Numerical continuation in ODE and maps

Modelling challenges in a changing climate and environment

Wei Hao Tey

VAST, Hanoi

18 March 2025

Bifurcation diagrams

Given a family of Ordinary Differential Equation (ODE)

$$\dot{u} = f(u, \alpha),$$

or a family of differential equation

$$u_{i+1} = f(u_i, \alpha),$$

for $u, u_i \in \mathbb{R}^n$, $\alpha \in \mathbb{R}$ and smooth $f : \mathbb{R}^{n+1} \to \mathbb{R}^d$.

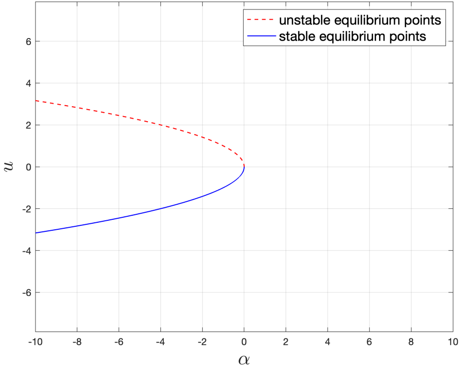
We aim to construct a bifurcation diagram of equilibria (or limit cycles) numerically.

Analytic construction

Example:

$$\dot{u} = f(u, \alpha) = \alpha + u^2.$$

Then, equilibrium points are $u = \pm\sqrt{-\alpha}$ when $\alpha \leq 0$, and no equilibrium points when $\alpha > 0$. Saddle node bifurcation occurs at $\alpha = 0$, where $f_u(0,0) = 0$ and $f_{uu}(0,0), f_\alpha(0,0) \neq 0$.



This is not always possible especially in high dimensional complex systems.

Naive solution: change parameter $\alpha_{i+1} = \alpha_i + h$ and perform forward (backward) integration on the previous equilibrium point. This does not work in complicated systems.

Numerical continuation toolbox

There are some toolboxes available which specialise in continuation of solutions and bifurcation analysis

- MatCont in MATLAB
- PyDSTool in Python
- Oscill8
- BifurcationKit.jl in Julia

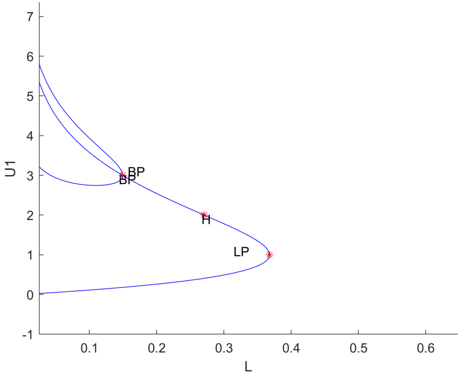
Some other software specialised on solutions of dynamical system

- AUTO
- GAIO (Global Analysis of Invariant Objects)

MatCont: Continuation toolbox for ODE

Spotlight of the numerical toolbox **MatCont** in MATLAB for continuation and bifurcation analysis for ODE (MatContm for discrete-time) by Yuri A. Kuznetsov, Hil G.E. Meijer, Willy Govaerts, and others.

Give insights on the framework and numerical methods used behind the toolbox and go through some practical examples tomorrow.



Numerical construction

- Time integration
- Numerical continuations of equilibria (limit cycles)
- Detection of bifurcation via singularity of test functions
- Numerical continuations of co-dim 1 bifurcations to detect co-dim 2 bifurcations

Simulate trajectories by integration

$$\dot{u} = f(u, \alpha)$$

- First step is to fix a parameter α and find an equilibrium point (or limit cycle).
- Simulate solution flow by numerical integration available in built-in ODE solver in MATLAB, e.g. `ode45, ode23 etc`.

Continuation of equilibrium points

$$\dot{u} = f(u, \alpha) = F(x), \; x = (u, \alpha)$$

Given an initial equilibrium point $x_0 = (u_0, \alpha_0)$, we find the rest of the equilibrium curve $M \subset \mathbb{R}^{n+1}$ satisfying $F(x) = 0$ for all $x \in M$, where $F : \mathbb{R}^{n+1} \to \mathbb{R}^n$. This is an example of a general **Algebraic Continuation Problem (ALCP)**.

Numerically, the solution means: given x_0 close to $x \in M$, we compute sequence of points $x_1, x_2, x_3, \ldots$, where the union of line segments connecting consequent points approximates the equilibrium curve M.

Continuation of equilibrium points

A naive way is to change $\alpha_{i+1} = \alpha_i + h$ by some step size h and perform time integration (forward or backward) of $\dot{u} = f(u, \alpha)$ with initial condition u_i, α_i. This method is dependent on the dynamics of the system which can be unreliable in complicated system, e.g. chaotic system.

The framework which works for general system is a predictor-corrector method:

- initial tangent prediction: $X^0 = x_i + hv_i$, where v_i is tangent to M at x_i ($F_x(x_i)v_i = 0$) and h is the step size.
- Newton-like correction.
- step size control.

Pseudo-arclength continuation

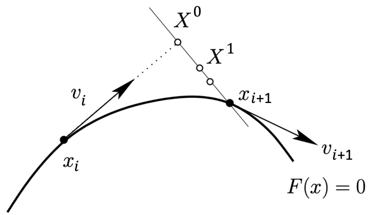
Recall $F : \mathbb{R}^{n+1} \to \mathbb{R}^n$, we need a scalar condition $g(x) = 0$ to use Newton's method. One way is to restrict x to a hyperplane passing through X^0 orthogonal to v_i, i.e. $g(x) = \langle x - X^0, v_i \rangle$. We then apply Newton's method to $G(x) = \begin{pmatrix} F(x) \\ g(x) \end{pmatrix} = 0$, giving

$$X^{k+1} = X^k - G_x(X^k)^{-1} G(X^k),$$

where $G_x(X^k) = \begin{pmatrix} F_x(X^k) \\ v_i^T \end{pmatrix}$ and $G(X^k) = \begin{pmatrix} F(X^k) \\ 0 \end{pmatrix}$.

Lemma (Keller-Lemma)

Consider a regular point $p \in \mathbb{R}^{n+1}$, i.e. $\mathrm{rank}(F_x(p)) = n$ and given a tangent vector v at p, i.e. $F_x(p)v$. The $(n+1) \times (n+1)$ Jacobian matrix $G_x(p) = \begin{pmatrix} F_x(p) \\ v^T \end{pmatrix}$ is full rank.

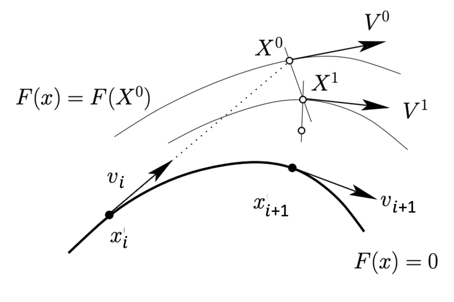


Moore-Penrose continuation

An improved continuation method is the (approximated) Moore-Penrose method, where we change V^k (and thus the hyperplane) in each steps instead of $V^k = v_i$ for all $k \geq 0$.

Find V^k such that $F_x(X^k)V^k = 0$ and

$$X^{k+1} = X^k - \begin{pmatrix} F_x(X^k) \\ (V^k)^T \end{pmatrix}^{-1} \begin{pmatrix} F_x(X^k) \\ 0 \end{pmatrix}$$



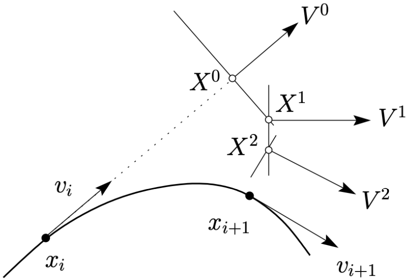
Approximated Moore-Penrose continuation

The disadvantage of the method is the need to calculate V^k every step. Instead, we approximate V^k with initial $V^0 = v_i$ by applying Newton's method w.r.t. W for $\begin{pmatrix} F_x(X^k)W \\ \langle V^k, W\rangle - 1 \end{pmatrix} = 0$.

The overall algorithm gives

$$X^{k+1} = X^k - \begin{pmatrix} F_x(X^k) \\ (V^k)^T \end{pmatrix}^{-1} \begin{pmatrix} F_x(X^k) \\ 0 \end{pmatrix}$$

$$V^{k+1} = V^k = \begin{pmatrix} F_x(X^k) \\ (V^k)^T) \end{pmatrix}^{-1} \begin{pmatrix} F_x(X^k)V^k \\ 0 \end{pmatrix}$$



Step size control

- Stepsize control is important in the algorithm: computational expensive if we use unnecessarily small step size and large step size loses details of the equilibrium curve.
- Convergent dependant control: decrease stepsize if not converge (convergent defined by some tolerance and maximum Newton's iterations), increase stepsize if converge too quickly (small Newton's iterations).

Detection of singularity on test functions

- The idea is to construct test functions which have simple zeros at points of singularity.
- These singularities represents bifurcations along the equilibrium curve $F(u, \alpha) = 0$.
- For example at saddle node bifurcation point, $f_u(u, \alpha) = 0, f_{uu}(u, \alpha) \neq 0, f_\alpha(u, \alpha) \neq 0$.
- Since numerically the test function $\phi(x)$ never reaches 0, we detect change of sign, i.e. $\phi(x_i)\phi(x_{i+1}) < 1$.