Diamanlab documentation Version 1.0

Bruno COCHELIN [1] **Isabelle CHARPENTIER** [2]

[1] LMA, UPR 7051 CNRS, Centrale Marseille
F-13402 MARSEILLE Cedex 20, France

[2] Icube - UMR 7357 CNRS
Laboratoire des Sciences de l'Ingénieur, de l'Infiormatique et de l'Imagerie
F-67412 Illkirch Cedex, France

August 2014

1 Getting started

Diamanlab allows you to interactively compute and visualize the solution branches of algebraic system of equations $R(U) = 0$ with n equations and $n+1$ unknowns.

Example ECrea [1] : The (column) vector of unknowns is $U = [u_1, u_2, \lambda]^T$ and the (column) vector of equations

$$\begin{array}{rll} r_1(u_1,u_2,\lambda) = & 2u_1 - u_2 + 100\frac{u_1}{1+u_1+u_1^2} - \lambda & = 0 \\ r_2(u_1,u_2,\lambda) = & 2u_2 - u_1 + 100\frac{u_2}{1+u_2+u_2^2} - (\lambda+\mu) & = 0 \end{array} \tag{1}$$

μ is a fixed parameter.

To solve this system with diamanlab,

1. launch MATLAB software
2. add the source directory `SRC` of the Diamanlab distribution in the MATLAB path with the command `addpath('/... ... /Diamanlab-1.0/SRC')`
3. `set ECrea as the Current Directory of MATLAB`
4. `type ecrea in the Command Window.`

A graphical interface (GUI) and the projected bifurcation diagram (Figure 2) shows up. Click on the `Forward >>` button of the graphical interface to compute and visualize a part of the solution branch. Click again to enlarge the solution branch.

1. Taken from DOEDEL et Al

On each curves of the projected bifurcation diagram (Figure 2) a point with a square black marks and an arrows appears. This set of points is called the `current point` of continuation. It is from that point, and toward the arrow directions, that new sections of branch are computed.

Everything related to this `ECrea` example is in the `@ECrea` directory. Important information for a beginner is

- the `R.m` matlab file of the @ECrea directory which implement the equation to be solved (written by the user).
- the argument `U0value` in the `diamanlab` command (see at the end of `ecrea.m` file). It gives [0 0 0] as an initial point to start the continuation. If the initial point is not an exact solution (it is the case here) Diamanlab perform Newton iteration.
- the argument `displavariables` containing several row of index pair. Here, [3 1 ; 3 2] means that 2 curves will be drawn in the projected bifurcation diagram, λ versus u_1 (first row 3 1) and λ versus u_2 (second row 3 2).

The best way to solve your own system is to make a copy of the `@ECrea` directory and to modify the names and the data.

2 A quick overview of Diamanlab for users

At launch, Diamanlab opens at least two windows

— the Diamanlab graphical user interface, or GUI (button box).

— the "figure 2" windows for displaying 2D curves (projection of the solution branches), for managing the `current point` and for selecting point with the mouses .

The GUI interface is itself divided into fives frames

— `Continuation` : for advancing the continuation of the branch. The number of steps and the required precision may be changed.

— `Correction` : to enable Newton correction when the tolerance error exceeds the threshold.

— `current point` : for managing the position of the current point

— `2D curves display` : for managing the projected bifurcation diagram (markers, displayed variables) and the `user display` facilities

— `diagram` : for saving and loading diagram, erasing selected section, erasing the full diagram

Computing a bifurcation diagram is an interplay between : (i) advancing on the branch of solution with the `Forward >>` button, (ii) changing the position of the current point with `set` and `jump` button in order to explore others branches, (iii) analysing the solution with the user display facilities.

3 A quick overview of Diamanlab for programmers

Diamanlab-1.0 is programmed in the MATLAB Language using Object-Oriented techniques. It is composed of one main `diamanlab.m` file, four generic classes named @Syst, @Taylor, @ContDriver, @Checkpoint, and a least one user-defined class containing the problem to be solved (@ECrea for instance).

Permanent classes quick overview

— Syst class :
  - properties : ninc (unknown number), neq (equation number) and various parameters for numerical method
  - methods : numerical methods related to the system $R(U) = 0$, for instance, compute the Jacobian, Perform Newton Correction, compute ANMseries, ...)

— TAYLOR class : library devoted to high order differentiation
  - properties : order, Values, Coef -¿ for storing a Taylor series
  - methods : overloaded intrinsic operators and functions for the high order AD. Evaluation of Taylor series.

— ContDriver class : manage the full continuation process
  - properties : the current point `U0now`, `Utnow` and a list of `Section` ((a section is a finite part of a solution branch)
  - methods : functions for all the tasks associated with the Graphical User Interface

— Checkpoint class : to store all the information associated to a series computation
  - properties : start point `U0`, `Ut`, end point `Uend` , `Utend` , point by point form `Upp`...
  - methods : to construct and display branch section

User-defined class overview

— ECrea (as example) : for implementing the system equation $R(U) = 0$
  - properties : parameters for $R(U)$
  - methods : R.m (input -¿ U, ouput -¿ R(U)), disp.m for user display facilities

Notice that each class has a `get` and a `set` method to access and modify the properties.

quick overview of Ecrea example

The `ecrea.m` script run diamanlab with a list of name-value couple as input argument. At first, `diamanlab.m` create the object `sys` instance of the `Ecrea` class, which itself derives from `Syst` class. Diamanlab then create the object `diagram` instance of the `ContDriver` class. Diamanlab launch the Figure 2 and the GUI, display the `current point` and wait for user actions.

4 More presentation

This section is devoted to users and developpers that wants to learn more with the main concepts and idea behind the Diamanlab software.

4.1 Definition, notation, theroretical background

Consider the algebraic system

$$R(U) = 0 \tag{2}$$

with $R \in \mathbb{R}^n$ and $U \in \mathbb{R}^{n+1}$. When R is smooth, the solution set of (2) is a union of various solution branches that possibly cross at bifurcation points. The solution set of (2) is called the "bifurcation diagram"

Remarks : it is sometime convenient to distinguish the state variable $u \in \mathbb{R}^n$ and the control parameter $\lambda \in \mathbb{R}$ (also called bifurcation parameter), and to write the system : $R(u, \lambda) = 0$.

4.1.1 Derivative of R

— $R_{,u}$ is a square $n \times n$ matrix that may be banded
— $R_{,\lambda}$ is a colum vector
— the jacobian $R_{,U} = \left[R_{,u} R_{,\lambda}\right]$ is a matrix with n rows and $n+1$ colums. We use an upperscript to indicate the point where the matrix is evaluated, for instance, $R_{,U}^{U_0}$.

4.1.2 Regular solution

A solution U_0 of (2) is said to be regular if the jacobian matrix $R_{,U}^{U_0}$ is full rank, ie, Rank $(R_{,U}^{U_0}) = n$.

At a regular solution point U_0, the system constituted by the linearisation of (2) around U_0, ie, ($R_{,U}^{U_0} \Delta U = 0$ and the normalisation condition $\Delta U^T \Delta U = 1$) has two solutions (with opposite sign) denoted by U_1 and $-U_1$. The `tangentvector` method of the `Syst` class compute one of these two solution.

Near a regular solution U_0, there exists a unique one dimensional continum of solutions called a solution branch. For smooth R, the solution branch can be represented by a Taylor series with respect to the pseudo-arclength path parameter $a =< U - U_0, U_1 >$. Here, $< U, V >= U^T.A.V$ where A is a diagonal matrix for balancing the component of the state vector. For example, when the last diagonal term of A is zero, the λ parameter is not involved in the pseudo-arclength definition. The diagonal of A is set in the property `arclengthdef` of the `Syst` class, by default A is set to identity.

4.1.3 Singular solution

When the jacobian matrix $R^{U_0}_{,U}$ is not full rank, the solution U_0 is said to be singular. The particular case of simple bifurcation point (when the Rank is $n - 1$) corresponds to the crossing of two solution branch at U_0. Simple bifurcation points are detected along the branch. Putting the current point on a bifurcation point permit to swith to the bifurcated branch.

4.2 The current point

The current point (shown in figure 2 as a set of points with a black square marks and an arrows) is stored in the `Unow` and `Utnow` properties of the `diagram` objet (instance of the `ContDriver` class). The user can change the position of this current point with the button of the "current point" frame of the GUI.

4.3 Continuation principle

In Diamanlab, the solution branch are determined by computing new sections from the "current point". Computing a new section of a solution branch means

— compute the tangent U_1 at the start point U_0 and choose its sign according to the traveling direction U_t.

— compute a high order Taylor series of the solution branch at the start point U_0

— analyse the range of utility of the series, ie, the interval of the path parameter $[0, a_{max}]$ for which the Taylor series satisfy the required accuracy.

— detect if there is simple bifurcation inside the range of utility and store them.

— provide a discrete (point by point) representation of the branch inside the range of utility. The property `nbpts` of the `CONTDRIVER` class give the number of such points.

— provide an end point `Uend` and an end traveling direction `Utend`, to be used as start point and traveling direction for the next section.

4.4 Visualisation

Because the number of state variable n is generally large, one can only draw projections of the bifurcation diagram. In Diamanlab, such projection is a collection of `ncurve` 2D curves where the axis are user-chosen variables of the state vecteur : the `dispvars(ncurves,2)` property of the `params` structure. This table must be given by the user before launch. It may be modified afterward using the `variable` button of the `2D curves display` GUI frame. An interface then appear to modify the `display-variable`. The graphical windows (figure 2) that display these 2D curves is called the "projected bifurcation diagram".

4.4.1 User display facilities

Reading the component values of a vector solution U is genrally not the best way to analyse a solution point. It is often much better to draw a graphical representation of U to have a good insight of the computed solution. This facilities is provided as follow :

- the user must program its own `disp.m` method and put it in the same directory than the `R.m`. The `disp.m` method should have U as a single entry argument and produce any kind of action (graphics, video, sound, ...) from that U data.
- the `disp.m` fonction is called each time a new branch section is computed (the 'end' point of the section is used as the argument for `disp.m`) or when the user press the "select point" of the user display frame and select a point.

The Bratu example provides a basic example for this facility.

Références

[1] I. Charpentier, B. Cochelin, K. Lampoh, Diamanlab - An interactive Taylor-based continuation tool in MATLAB, 11 pages (Mar. 2013).
URL http ://hal.archives-ouvertes.fr/hal-00853599