A generic and efficient Taylor series based continuation method using a quadratic recast of smooth nonlinear systems

Louis Guillot, Bruno Cochelin, Christophe Vergez

A generic and efficient Taylor series based continuation method using a quadratic recast of smooth nonlinear systems

Louis Guillot, Bruno Cochelin, Christophe Vergez

Aix Marseille Univ, CNRS, Centrale Marseille, LMA, UMR 7031, Marseille, France.

July 2, 2018

Abstract

This paper is concerned with a Taylor series based continuation algorithm, ie, the so-called Asymptotic Numerical Method (ANM). It describes a generic continuation procedure that apply the ANM principle at best, that is to say, that presents a high level of genericity without paying the price of this genericity by low computing performances. The way to quadratically recast a system of equation is now part of the method itself, and the way to handle elementary transcendental function is detailed with great attention. A sparse tensorial formalism is introduced for the internal representation of the system, which, when combines with a block condensation technique, provides a good computational efficiency of the ANM. Three examples are developed to show the performance and the versatility of the implementation of the continuation tool. Its robustness and its accuracy are explored. Finally, the potentiality of this method for complex non linear finite element analysis is enlightened by treating 2D elasticity problem with geometrical nonlinearities.

Keywords : continuation, Taylor series, quadratic recast, asymptotic numerical method, nonlinear systems, finite element method.

1 Introduction

The so-called Asymptotic Numerical Method (ANM), first described in [12] and [11], is a continuation technique based on high order Taylor series expansions of the unknowns with respect to a path parameter. The solution branches are computed step by step as in a classical predictor-corrector algorithm (see [25], [17] and [31]), but as opposed to first order prediction, the accuracy of a high order Taylor series prediction is so high that no correction is needed in general. Each step provides a local continuous representation of the branch whose length is computed afterward directly from the convergence properties of the series. Having a continuous description of the branch makes the continuation very robust and provides some opportunities in the detection of bifurcations [15].

ANM has first been applied to compute equilibrium branches of geometrically nonlinear finite elements models in structural mechanics [13] [33] [1]. It has been later

[1] The term "numerical" stands in the name of the method because of this finite element discretization.

extended to more complex and less regular non linearities such as the one associated to material non linearities [34], [2], contact conditions and friction [26], [3], vibrations of viscoelastic shells [19] and material instabilities [30]. Stationary solution of Navier-Stokes flow [4], [28] has also been adressed, showing that very large scale problems with more than a million degree of freedom can be treated efficiently with this method. The continuation of periodic solutions of dynamical systems is presented in [16], [24] where the ANM is combined with a Fourier series expansion known as the harmonic balance method. Extension to the continuation of quasi-periodic solutions is adressed in [22]. The continuation solver MANLAB-1.0 [1] has been the first attempt to design a general purpose continuation software working on the ANM principle. It allows a user to enter its own algebraic system and to interactively draw the bifurcation diagram. It works well for systems with a few hundreds of equations but the quadratic recast of elementary transcendental functions was a little abstruse and finite element mechanical models could not be implemented because of the lack of performances.

A common denominator of the works mentioned above is the requirement to recast the governing equations in a quadratic format. The motivation for this recast is that the Taylor series computation becomes easy and efficient when a system is quadratic. However, this requirement has turn out to be the most difficult point of the method. Its misunderstanding has prevented many potential users and developpers from succeeding in the use of the method, especially when the system of equation contains elementary transcendental functions (exp, log, sin, cos, power, ...). Introducing Automatic Differentiation (AD) principle inside the ANM process [9], [10] has been the first elegant way of removing this difficulty. The DiaManlab software [7] allows the users to write the governing equations in a natural way, without any need of a quadratic recast. However, once again, the price to pay for this genericity is a reduction of the computing performance that may be again a drawback for large systems resulting for instance from the finite element (FE) discretization of a continuous problem.

In this paper, combinig the experiences of twenty years of work on the ANM and this experience on Automatic Differentiation, we present a generic way to solve an algebraic system with the ANM principle applied at best, *ie*, providing both genericity and efficiency. The quadratic recast of the original system is now a part of the method itself instead of being a pre-processing task left to the user. The declaration of a list of auxiliary variables that follow the linear declaration rule provides a clear and useful distinction between the main and the auxiliary variables, and between the main and the auxiliary equations. Attention is paid to the treatment of elementary transcendental functions [35] and the way their (quadratic) differential form should appear in the process. Additionally, this results in a clear separation between quadratic and functional equations. Another important improvement is the use of a sparse tensor formalism for the internal representation of a quadratic system on a computer. This allows an efficient construction of the linear systems that have to be solved for the series computation, including a block inversion for the condensation of the auxiliary variables. Another key point is that the sparse tensors are automatically constructed from the equations by using polarization formula. This allows to write the quadratic equations in a quite natural way.

The generic method described here has been implemented in Matlab as the fourth version of the Manlab suite. It permits to deal with a very large class of algebraic systems containing many elementary transcendental functions as it will be shown on three examples in section 5. Another very important potentiality of this method is the capacity to implement a FE mechanical model in a quite simple way and, that time, without suffering from poor computing performances. Only a quadratic writing of the governing equation, *ie*, of the residual vector equation, is required.

2 Continuation using the Asymptotic Numerical Method (ANM)

2.1 Definitions, notations and theoretical background

Consider the algebraic system

$$R(u,\lambda) = 0 \tag{1}$$

where $R : \mathbb{R}^{N_{\text{eq}}+1} \mapsto \mathbb{R}^{N_{\text{eq}}}$ is a function called the residue, $u \in \mathbb{R}^{N_{\text{eq}}}$ is the vector of unknowns and $\lambda \in \mathbb{R}$ is a parameter of interest. The notation $U = (u,\lambda) \in \mathbb{R}^{N_{\text{eq}}+1}$ is sometimes used in the following parts. R is a real analytic function and can be written as a sum, product, composition of rational and elementary transcendental functions such as polynomials, $\exp, \cos, \log$, *etc...* The aim of continuation is to find the solution set of (1). This is a union of various solution branches that possibly cross at bifurcation points, which is called the bifurcation diagram. The procedure to compute a continuation step is now explained.

Let $U_0 = (u_0, \lambda_0)$ be a regular solution of equation (1). This means that the rank of the Jacobian matrix at point U_0,

$$\frac{\partial R}{\partial U} = \left[\frac{\partial R}{\partial u}, \frac{\partial R}{\partial \lambda}\right] \tag{2}$$

is N_{eq}. Let U_0 be the starting point. Let $U_1 = (u_1, \lambda_1)$ be a unitary tangent vector at point U_0 and $a = (u - u_0) \cdot u_1 + (\lambda - \lambda_0)\lambda_1 = (U - U_0) \cdot U_1$ be the pseudo arc-length parameter introduced in [6], [18]. Note that the definition of a is used as a closing equation for the system (1). The analytic implicit function theorem (which complex form can be found in [21]) allows to search the solution branch of equation (1) around U_0 as an analytic functions of a, *i.e* as Taylor series expansion with respect to a:

$$U(a) = U_0 + aU_1 + a^2U_2 + a^3U_3 + \dots \tag{3}$$

Note that in (3) λ is also developed as a series of the path parameter a. In practice, the series are truncated at an order p. They are then introduced in equation (1) which gives the Taylor series development of $R(U(a))$:

$$R(U(a)) = R(U(0)) + aR_1 + a^2R_2 + a^3R_3 + \dots \tag{4}$$

where

$$\begin{array}{rcccl}
R_1 & = & \frac{\mathrm{d}R}{\mathrm{d}a}\Big|_{a=0} & = & \frac{\partial R}{\partial U}U_1 \\
R_2 & = & \frac{1}{2}\frac{\mathrm{d}^2R}{\mathrm{d}a^2}\Big|_{a=0} & = & \frac{\partial R}{\partial U}U_2 - F_2(U_1) \\
R_3 & = & \frac{1}{3!}\frac{\mathrm{d}^3R}{\mathrm{d}a^3}\Big|_{a=0} & = & \frac{\partial R}{\partial U}U_3 - F_3(U_1,U_2) \\
\vdots & & \vdots & & \vdots \\
R_p & = & \frac{1}{p!}\frac{\mathrm{d}^pR}{\mathrm{d}a^p}\Big|_{a=0} & = & \frac{\partial R}{\partial U}U_p - F_p(U_1,\dots,U_{p-1})
\end{array} \tag{5}$$

with F_k being functions that depend only on already computed terms of the series. The system (1) then looks like : $\forall k \leq p, R_k = 0$. It has to be noticed that all these equations share the same matrix to inverse that is the Jacobian matrix (2) at point U_0.

When the series (3) is computed up to order p, the domain of utility $[0, a_{max}]$ is determined. The maximum increase of the residue (1) is set to a small value ε_1,

such that $|R(U(a)) - R(U(0))| < \varepsilon_1$ for all $a \in [0, a_{max}]$. With the approximation $R(U(a)) - R(U(0)) = a^{p+1} R_{p+1}$, it gives

$$a_{max} = \left(\frac{\varepsilon_1}{||R_{p+1}||}\right)^{\frac{1}{p+1}} \tag{6}$$

Finally, the ending point of the continuation step $U(a_{max})$ is computed and it is used as the starting point of the next continuation step. This approach leads to a continuous representation of the solution branch. There are almost no corrections needed to stay on the branch in practice because the Taylor series development (3) is a very accurate approximation of the true solution branch on its domain of utility. This domain of utility can be computed directly from the knowledge of the series with the formula (6).

2.2 Implementation methods

To summarize, the Asymptotic Numerical Method (ANM) is a method to compute a solution branch from the knowledge of a starting point U_0, of a way to compute the Jacobian matrix (2) of the system and of a procedure to compute the Taylor series (3) *i.e.* to compute U_k from $U_j, j < k$. As explained in the Introduction, there are several ways to deal with these requirements. One of the solution is the use of automatic differentiation. In this case, the knowledge of equation (1) only is enough to compute the bifurcation diagram. It has been done in [8] in an elegant way. The set up is surprisingly simple for the user, however it becomes computationally expensive when the size of the system increases above several hundreds of unknowns.

Here we follow the traditional idea of the ANM that is described in the book [14] and the papers [11], [12] and [13]. The system is written in a quadratic format that allows direct computation of both the jacobian matrix (2) and of the coefficients of the Taylor series (3).

3 Recast of the system of equations

In this section the equations (1) are rewritten in a quadratic format in order to apply the ANM. The general set up follows an explanatory example.

3.1 Explanatory example

Let us consider an academic example containing two elementary transcendental functions, exp and tanh for which the residue function (1) is given by $R = (r_1, r_2)$:

$$\begin{cases} r_1(u_1, u_2, \lambda) = u_1 + \lambda \frac{\exp(u_2)}{1+u_1} \\ r_2(u_1, u_2, \lambda) = u_2 + u_1 \tanh(\frac{-5u_1}{1+u_1 u_2}) \end{cases} \tag{7}$$

3.2 Definition of the auxiliary variables

The equations (7) are rewritten hereafter in a quadratic format. To achieve this, auxiliary variables are introduced. These additional variables are defined from the *main variables* u_1, u_2, λ.

To compute r_1 from given input values (u_1, u_2, λ), one first decomposes the process into elementary operations and stores intermediate results. For instance, an evaluation of $\exp(u_2)$ is stored. Then, an evaluation of $1 + u_1$ is stored as well. After,

the first result is divided by the second one. This number is multiplied by λ, *etc...* Following this, let v_1, v_2, v_3, v_4, v_5 be:

$$\begin{array}{rl} v_1 = & \exp(u_2) \\ v_2 = & \frac{v_1}{1+u_1} \\ v_3 = & 1 + u_1 u_2 \\ v_4 = & -\frac{5u_1}{v_3} \\ v_5 = & \tanh(v_4) \end{array} \tag{8}$$

From these definitions we can now write the system (7) with only quadratic non-linearities :

$$\begin{array}{rl} r_1 = & u_1 + \lambda v_2 \\ r_2 = & u_2 + u_1 v_5 \end{array} \tag{9}$$

In the definition of auxiliary variables (8) there are different types of equations. Some are already quadratic, here $v_3 = 1+u_1u_2$. Some are easily made quadratic, here $v_2 = \frac{v_1}{1+u_1}$ and $v_4 = -\frac{5u_1}{v_3}$ are respectively rewritten $v_2(1+u_1) = v_1$ and $v_4v_3 = -5u_1$ which are equivalent to the first definitions if $u_1 \neq -1$ and $v_3 \neq 0$. The last type of auxiliary variables, here $v_1 = \exp(u_2)$ and $v_5 = \tanh(v_4)$ will be treated in the next section.

The auxiliary variables are defined by following the *Linear declaration rule i.e.* only from the main variables and from auxiliary variables with a smaller index. To write it in a compact way : $\forall i, v_i = f(u, \lambda, v_1, \ldots, v_{i-1})$. It ensures that r_1, r_2 can be computed from the input u_1, u_2, λ. The choice of the auxiliary variables is not unique. For example, the definitions of v_3 and v_4 could be replace by $v_3 = u_1u_2$ and $v_4 = -\frac{5u_1}{1+v_3}$ respectively. Of course, this does not change the final result.

The linear declaration rule ensures that the system of equations (7) is equivalent to the two systems (8) and (9).

3.3 Quadratic recast of the elementary transcendental functions

The case of $v_1 = \exp(u_2)$ and $v_5 = \tanh(v_4)$ is treated here. To achieve their quadratic recast, these equations are first differentiated:

$$\begin{array}{lll} \mathrm{d}v_1 & = \exp(u_2)\mathrm{d}u_2 & = v_1\mathrm{d}u_2 \\ \mathrm{d}v_5 & = (1 + \tanh^2(v_4))\mathrm{d}v_4 & = (1 + v_5^2)\mathrm{d}v_4 \end{array} \tag{10}$$

Then the auxiliary variable $v_6 = (1+v_5^2)$ is defined to obtain the final quadratic recast of the differentiated equations

$$\begin{array}{ll} \mathrm{d}v_1 & = v_1\mathrm{d}u_2 \\ \mathrm{d}v_5 & = v_6\mathrm{d}v_4 \\ v_6 & = 1 + v_5^2 \end{array} \tag{11}$$

This formulation is said to be quadratic because it is quadratic with respect to $(u, v, \lambda, \mathrm{d}u, \mathrm{d}v)$. A table of recast of the most common transcendental functions is available in Appendix A.

3.4 Final recast of the example

The unknowns are now split in two families : $U = (u_1, u_2, \lambda)$ for the original main unknowns and $U_{\text{aux}} = (v_1, v_2, v_3, v_4, v_5, v_6)$ for the auxiliary unknowns. The full vector

of unknowns is called $U_{\text{tot}} = (U, U_{\text{aux}})$. From the previous section, the system (7) has been written

$$R_{\text{tot}}(U_{\text{tot}}) = \left[\begin{array}{c} u_1 + \lambda v_2 \\ u_2 + u_1 v_5 \\ \hline v_1 - \exp(u_2) \\ v_1 - v_2 - v_2 u_1 \\ 1 + u_1 u_2 - v_3 \\ -5u_1 - v_4 v_3 \\ v_5 - \tanh(v_4) \\ 1 + v_5^2 - v_6 \end{array}\right] = \left[\begin{array}{c} R \\ \hline \\ R_{\text{aux}} \\ \\ \\ \\ \\ \end{array}\right] \tag{12}$$

where the two first equations correspond to the initial system of equations (7) and the other ones to the auxiliary variables. The horizontal line materializes this separation : $R_{\text{tot}} = (R, R_{\text{aux}})$. There is another useful separation of the residue between already quadratic equations, R_{quad}, and the other ones, R_{fun} where an elementary function appears. R_{tot} is split up in these two parts :

$$R_{\text{tot}}(U_{\text{tot}}) = \left[\begin{array}{c} u_1 + \lambda v_2 \\ u_2 + u_1 v_5 \\ \hline 0 \\ v_1 - v_2 - v_2 u_1 \\ 1 + u_1 u_2 - v_3 \\ -5u_1 - v_4 v_3 \\ 0 \\ 1 + v_5^2 - v_6 \end{array}\right] + \left[\begin{array}{c} 0 \\ 0 \\ \hline v_1 - \exp(u_2) \\ 0 \\ 0 \\ 0 \\ v_5 - \tanh(v_4) \\ 0 \end{array}\right] = R_{\text{quad}} + R_{\text{fun}} \tag{13}$$

To this "total" residue is added the differentiated form of its third and seventh components, that is of R_{fun}, in $\mathrm{d}R_{\text{fun}}$ vector,

$$\mathrm{d}R_{\text{fun}} = \left[\begin{array}{c} 0 \\ 0 \\ \mathrm{d}v_1 - v_1 \mathrm{d}u_2 \\ 0 \\ 0 \\ 0 \\ \mathrm{d}v_5 - v_6 \mathrm{d}v_4 \\ 0 \end{array}\right] \tag{14}$$

The differentiated form of the already quadratic equations does not need to be specified.

3.5 General formula

From a general point of view, the residue (1) R can be written in the following way:

$$\begin{array}{rcccc} R_{\text{tot}}(U_{\text{tot}}) & = & R_{\text{quad}}(U_{\text{tot}}) & + & R_{\text{fun}}(U_{\text{tot}}) \\ & = & [C + L(U_{\text{tot}}) + Q(U_{\text{tot}}, U_{\text{tot}})] & + & [L_d(U_{\text{tot}}) - f(U_{\text{tot}})] \end{array} \tag{15}$$

where $U_{\text{tot}} = (u, \lambda, v)$ is the vector of all the unknowns, v are the auxiliary variables. R_{quad} is the already quadratic part of the residue and R_{fun} is the part which is not quadratic (but its differential form is). C is a constant operator, L and L_d are linear operators, Q is a quadratic operator and each component of $f(U_{\text{tot}})$ is an

elementary transcendental function of U_{tot} or zero. f is a non–quadratic function whose differential form can be written quadratically. Formally it means that the application $(U_{\text{tot}}, dU_{\text{tot}}) \mapsto \mathrm{d}_{U_{\text{tot}}} f(dU_{\text{tot}})$, where $\mathrm{d}_{U_{\text{tot}}} f$ is the differential application of f at point U_{tot}, is bi–linear. This bi–linear application is called Q_d. One has $Q_d(U_{\text{tot}}, dU_{\text{tot}}) = \mathrm{d}_{U_{\text{tot}}} f(dU_{\text{tot}})$. To the system (15) is added the differentiated form of $R_{\text{fun}}(U_{\text{tot}}) = L_d(U_{\text{tot}}) - f(U_{\text{tot}})$:

$$\mathrm{d}_{U_{\text{tot}}} R_{\text{fun}}(dU_{\text{tot}}) = L_d(dU_{\text{tot}}) - Q_d(U_{\text{tot}}, dU_{\text{tot}}) \tag{16}$$

This is used in the next section to compute the Jacobian matrix (2) of the system and the Taylor series (3).

4 Details on the computation of the series

The formulations (15) and (16) and only these, are used for the following computations.

4.1 Computation of the Jacobian matrix

4.1.1 General computation

From equation (15), the Jacobian matrix can be computed automatically :

$$\begin{array}{rcl} \frac{\partial R_{\text{tot}}}{\partial U_{\text{tot}}} & = & L + Q(U_{\text{tot}}, \cdot) + Q(\cdot, U_{\text{tot}}) \quad + \quad L_d - \frac{\partial f}{\partial U_{\text{tot}}}(\cdot) \\ & = & L + Q(U_{\text{tot}}, \cdot) + Q(\cdot, U_{\text{tot}}) \quad + \quad L_d - Q_d(U_{\text{tot}}, \cdot) \end{array} \tag{17}$$

In our example the only non-zero elements of Q_d are $\exp(u_1)\mathrm{d}u_1$ and $(1-\tanh(v_4)^2)\mathrm{d}v_4$. They can be written $v_1\mathrm{d}u_1$ and $v_6\mathrm{d}v_4$ respectively, as in (14).

The *Linear declaration rule* ensures that the right-bottomed sub-matrix $\frac{\partial R_{\text{aux}}}{\partial U_{\text{aux}}}$ that contains the partial derivatives of the equations defining the auxiliary variables with respect to the auxiliary variables is triangular (see example below). This information is used for efficient block–solving in the computation of the series.

4.1.2 Explanatory example

The jacobian matrix is the matrix of partial derivatives of the full residue with respect to all the variables. The jacobian matrix of the example is computed with the equations (13) and (14).

$$J_{U_{\text{tot}}} = \begin{bmatrix} \frac{\partial R}{\partial U} & \frac{\partial R}{\partial U_{\text{aux}}} \\ \frac{\partial R_{\text{aux}}}{\partial U} & \frac{\partial R_{\text{aux}}}{\partial U_{\text{aux}}} \end{bmatrix} = \left[\begin{array}{ccc|ccccc} 1 & 0 & v_2 & 0 & \lambda & 0 & 0 & 0 & 0 \\ v_5 & 1 & 0 & 0 & 0 & 0 & 0 & u_1 & 0 \\ \hline 0 & -v_1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ -v_2 & 0 & 0 & 1 & -1-u_1 & 0 & 0 & 0 & 0 \\ u_2 & u_1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ -5 & 0 & 0 & 0 & 0 & -v_4 & -v_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -v_6 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2v_5 & -1 \end{array}\right] \tag{18}$$

4.2 Computation of the series

In this section, the subscripts "tot" are omitted : U_{tot} is simply written U.

The focus is on the general case. The equations (15) are written with an already quadratic part R_{quad} and a functional part R_{fun}. For the quadratic part of equation (15), U is replaced by its series and the expansions are truncated at order N :

$$\begin{aligned} 0 &= C + L\sum_{k=0}^{N} a^k U_k + Q\left(\sum_{k=0}^{N} a^k U_k, \sum_{k=0}^{N} a^k U_k\right) \\ &= C + \sum_{k=0}^{N} a^k \left(LU_k + \sum_{i+j=k} Q(U_i, U_j)\right) + \mathcal{O}(a^{N+1}) \end{aligned} \tag{19}$$

Equating the coefficients of the series for each power of a to 0,

$$\begin{aligned} C + LU_0 + Q(U_0, U_0) &= 0 \\ LU_1 + Q(U_0, U_1) + Q(U_1, U_0) &= 0 \\ LU_2 + Q(U_0, U_2) + Q(U_2, U_0) &= -Q(U_1, U_1) \\ &\vdots \\ LU_k + Q(U_0, U_k) + Q(U_k, U_0) &= -\sum_{i=1}^{k-1} Q(U_i, U_{k-i}) \end{aligned} \tag{20}$$

The first equation is just the definition of a solution point U_0 and the second one the definition of the tangent vector U_1 at U_0. $LU_k + Q(U_0, U_k) + Q(U_k, U_0)$ is the first part of the jacobian matrix $\frac{\partial R_{\text{quad}}}{\partial U}$ at point U_0 applied at point U_k as shown in equation (5).

For the non-quadratic part of equation (15), the differentiated form is used :

$$L_d(\mathrm{d}U) - Q_d(U, \mathrm{d}U) = 0 \tag{21}$$

Replacing U and $\mathrm{d}U$ by their series expansion one gets

$$\begin{aligned} 0 = &\ L_d(\tfrac{\partial U}{\partial a}) - Q_d\left(U, \tfrac{\partial U}{\partial a}\right) \\ = &\ \sum_{k=0}^{N-1}(k+1)a^k L_d U_{k+1} - Q_d\left(\sum_{k=0}^{N} a^k U_k, \sum_{k=0}^{N-1}(k+1)a^k U_{k+1}\right) \\ = &\ \sum_{k=0}^{N-1} a^k \left((k+1)L_d U_{k+1} - \sum_{i+j=k} Q_d\left(U_i, (j+1)U_{j+1}\right)\right) + \mathcal{O}(a^N) \end{aligned} \tag{22}$$

Equating the coefficients of the series to 0,

$$\begin{aligned} 1L_d U_1 - Q_d(U_0, 1U_1) &= 0 \\ 2L_d U_2 - Q_d(U_0, 2U_2) &= Q_d(U_1, U_1) \\ &\vdots \\ kL_d U_k - Q_d(U_0, kU_k) &= \sum_{i=1}^{k-1} Q_d(U_i, (k-i)U_{k-i}) \end{aligned} \tag{23}$$

Or dividing by k :

$$\begin{aligned} L_d U_1 - Q_d(U_0, U_1) &= 0 \\ L_d U_2 - Q_d(U_0, U_2) &= \tfrac{1}{2} Q_d(U_1, U_1) \\ &\vdots \\ L_d U_k - Q_d(U_0, U_k) &= \tfrac{1}{k}\sum_{i=1}^{k-1}(k-i) Q_d(U_i, U_{k-i}) \end{aligned} \tag{24}$$

$L_d U_k - Q_d(U_0, U_k)$ is the second part of the jacobian matrix $\frac{\partial R_{\text{fun}}}{\partial U}$ at point U_0 applied at point U_k as shown in equation (5).

Calling J_{U_0} the jacobian matrix of R at point U_0 and putting together the preceding computations, the system to solve at order $p \geq 2$ is deduced :

$$J_{U_0}(U_p) = \sum_{i=1}^{p-1} \left(-Q(U_i, U_{p-i}) + \frac{p-i}{p} Q_d(U_i, U_{p-i})\right) := \mathrm{F}_p^{\text{nl}} \tag{25}$$

These systems are solved efficiently using a sparse tensorial formalism and a fast block-solving described hereafter.

4.3 Block solving of the linear systems

In the two previous subsections, the computation of the Jacobian matrix (2) of a system of type (15) was explained. The computation of the Taylor series (3) of a point solution is also explained. Once $U_k, k \leq p-1$ are known, the formula (25) is used to compute U_p :

$$U_p = J_{U_0}^{-1} \mathrm{F}_p^{\mathrm{nl}} \tag{26}$$

The focus is on the inversion of the jacobian matrix as it is the most expensive operation in terms of time computation. The system has been written thanks to auxiliary variables and is written separating the auxiliary part from the main part:

$$R_{\mathrm{tot}}(U_{\mathrm{tot}}) := \begin{bmatrix} R(U_{\mathrm{tot}}) \\ R_{\mathrm{aux}}(U_{\mathrm{tot}}) \end{bmatrix} \tag{27}$$

with $U_{\mathrm{tot}} = (U, U_{\mathrm{aux}})^t$, $U = (u, \lambda) \in \mathbb{R}^{N_{\mathrm{eq}}+1}$ and $U_{\mathrm{aux}} \in \mathbb{R}^{N_{\mathrm{eqaux}}}$. The Jacobian matrix of R_{tot} is computed by block :

$$J_{U_0} = \begin{bmatrix} \frac{\partial R}{\partial U} & \frac{\partial R}{\partial U_{\mathrm{aux}}} \\ \frac{\partial R_{\mathrm{aux}}}{\partial U} & \frac{\partial R_{\mathrm{aux}}}{\partial U_{\mathrm{aux}}} \end{bmatrix} := \begin{bmatrix} K & B \\ C & K_{\mathrm{aux}} \end{bmatrix} \tag{28}$$

The *Linear declaration rule* of the auxiliary variables ensures that K_{aux} is triangular, hence it is very easy to inverse. This information is used to solve the linear systems (26) efficiently:

$$\begin{aligned} & J_{U_{\mathrm{tot}}^0}(U_{p,\mathrm{tot}}) && = \mathrm{F}_{p,\mathrm{tot}}^{\mathrm{nl}} \\ \Leftrightarrow \quad & \begin{bmatrix} K & B \\ C & K_{\mathrm{aux}} \end{bmatrix} \begin{bmatrix} U_p \\ U_{p,\mathrm{aux}} \end{bmatrix} && = \begin{bmatrix} \mathrm{F}_p^{\mathrm{nl}} \\ \mathrm{F}_{p,\mathrm{aux}}^{\mathrm{nl}} \end{bmatrix} \end{aligned} \tag{29}$$

which is equivalent to the system:

$$\begin{cases} KU_p + BU_{p,\mathrm{aux}} & = \mathrm{F}_p^{\mathrm{nl}} \\ CU_p + K_{\mathrm{aux}} U_{p,\mathrm{aux}} & = \mathrm{F}_{p,\mathrm{aux}}^{\mathrm{nl}} \end{cases} \tag{30}$$

Now the second line is multiplied by the inverse of K_{aux} and $U_{p,\mathrm{aux}}$ is replaced in the first line accordingly:

$$\begin{cases} \left(K - BK_{\mathrm{aux}}^{-1}C\right) U_p & = \mathrm{F}_p^{\mathrm{nl}} - BK_{\mathrm{aux}}^{-1}\mathrm{F}_{p,\mathrm{aux}}^{\mathrm{nl}} \\ U_{p,\mathrm{aux}} & = K_{\mathrm{aux}}^{-1}\left(\mathrm{F}_{p,\mathrm{aux}}^{\mathrm{nl}} - CU_p\right) \end{cases} \tag{31}$$

Finally, there are two systems to solve : one of size $N_{\mathrm{eq}}+1$ and a triangular one of size N_{eqaux}, instead of one system of size $N_{\mathrm{eq}} + N_{\mathrm{eqaux}} + 1$. It goes without saying that this manipulation improves the time computation a lot.

The Jacobian matrix J of the main system, *ie*, without auxiliary variables, is computed with the condensation formula : $J = \left(K - BK_{\mathrm{aux}}^{-1}C\right)$. In our example, the jacobian matrix (18) is divided in four blocks as in equation (28)

$$
\begin{aligned}
K &= \begin{bmatrix} 1 & 0 & v_2 \\ v_5 & 1 & 0 \end{bmatrix} \\
B &= \begin{bmatrix} 0 & \lambda & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & u_1 & 0 \end{bmatrix} \\
C &= \begin{bmatrix} 0 & -v_1 & 0 \\ -v_2 & 0 & 0 \\ u_2 & u_1 & 0 \\ -5 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \\
K_{\text{aux}} &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1-u_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -v_4 & -v_3 & 0 & 0 \\ 0 & 0 & 0 & -v_6 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2v_5 & -1 \end{bmatrix}
\end{aligned}
\tag{32}
$$

As expected, K_{aux} is triangular. The condensation formula gives :

$$
J = \left(K - BK_{\text{aux}}^{-1}C\right) = \begin{bmatrix} 1 - \frac{\lambda v_2}{u_1+1} & \frac{\lambda v_1}{u_1+1} & v_2 \\ v_5 + \frac{10u_1v_5v_6}{v_3} + \frac{2u_1u_2v_4v_5v_6}{v_3} & \frac{2v_4v_5v_6u_1^2}{v_3} + 1 & 0 \end{bmatrix} \tag{33}
$$

Replacing the auxiliary variables by their values (8), one could check that J is the jacobian matrix of the original system (7). For clarity, this computation is not shown here.

4.4 Sparse tensorial formalism

4.4.1 Sparse tensors

The constant, linear and quadratic operators are written using a sparse tensorial formalism. For $R(X) = C + L(X) + Q(X, X) \in \mathbb{R}^N$ with $X \in \mathbb{R}^M$, we write :

$$
R_i = C_i + \sum_{j=1}^{M} L_{ij}X_j + \sum_{j,k=1}^{M} Q_{ijk}X_jX_k, 1 \leq i \leq N. \tag{34}
$$

Constant sparse tensor For $C \in \mathbb{R}^N$, a constant operator, the tensorial representation is simply : $C_i, 1 \leq i \leq N$. Its sparsity leads to a representation with two lists : the list of indexes $iC = \{i | C_i \neq 0\}$ of the positions of non–zero coefficients and the list of associated values $vC = \{C_i | i \in iC\}$.

Linear sparse tensor For $L : \mathbb{R}^M \mapsto \mathbb{R}^N$, a linear operator, the tensorial representation is $L_{ij}, 1 \leq i \leq N, 1 \leq j \leq M$. Its sparsity leads to a representation with three lists : the list of indexes iL and the list of associated variables jL that are defined by $iL \times jL = \{(i, j) | L_{ij} \neq 0\}$ with associated values $vL = \{L_{ij} | (i, j) \in iL \times jL\}$.

Quadratic sparse tensor For $Q : \mathbb{R}^M \times \mathbb{R}^M \mapsto \mathbb{R}^N$, a quadratic operator, the tensorial representation is $Q_{ijk}, 1 \leq i \leq N, 1 \leq j, k \leq M$. Its sparsity leads to a representation with four lists : the list of indexes iQ and the two list of associated variables jQ and kQ that are defined by $iQ \times jQ \times kQ = \{(i, j, k) | Q_{ijk} \neq 0\}$ with associated values $vQ = \{Q_{ijk} | (i, j, k) \in iQ \times jQ \times kQ\}$.

4.4.2 Explanatory example

The sparse tensors of the example are now given with their lists of indexes. The vector of unknowns is here $U_{\text{tot}} = (u_1, u_2, \lambda, v_1, v_2, v_3, v_4, v_5, v_6)$ of size $M = 9$. The $N = 8$ equations are written in (12). The lists are

$$\begin{array}{rl} iC & = [5 \quad 8] \\ vC & = [1 \quad 1] \end{array} \tag{35}$$

$$\begin{array}{rl} iL & = [1 \quad 2 \quad 4 \quad 4 \quad 5 \quad 6 \quad 8] \\ jL & = [1 \quad 2 \quad 4 \quad 5 \quad 6 \quad 7 \quad 9] \\ vL & = [1 \quad 1 \quad 1 \quad -1 \quad -1 \quad -5 \quad -1] \end{array} \tag{36}$$

$$\begin{array}{rl} iLd & = [3 \quad 7] \\ jLd & = [4 \quad 8] \\ vLd & = [1 \quad 1] \end{array} \tag{37}$$

$$\begin{array}{rl} iQ & = [\mathbf{1} \quad 2 \quad 4 \quad 5 \quad 6 \quad 8] \\ jQ & = [\mathbf{3} \quad 1 \quad 1 \quad 1 \quad 6 \quad 8] \\ kQ & = [\mathbf{5} \quad 8 \quad 5 \quad 2 \quad 7 \quad 8] \\ vQ & = [\mathbf{1} \quad 1 \quad -1 \quad 1 \quad -1 \quad -1] \end{array} \tag{38}$$

The index iQ represents the line of the equation, jQ the number of a variable, kQ the number of a variable also and vQ the associated value. For example here one can read that on the **first** line appears the product between the **third** variable ($U(3) = \lambda$) and the **fifth** variable ($U(5) = v_2$) with associated value **1**. Indeed, one can read the term $1 \times \lambda \times v_2$ on the first line of the total residue vector (12).

$$\begin{array}{rl} iQd & = [\ \mathbf{3} \quad 7] \\ jQd & = [\ \mathbf{4} \quad 9] \\ kQd & = [\ \mathbf{2} \quad 7] \\ vQd & = [\mathbf{\text{-}1} \quad -1] \end{array} \tag{39}$$

The index jQd corresponds to the variable which is not differentiated and kQd to the variable which is differentiated. Here one can read : On the **third** equation, there is the product of the **fourth** variable ($U(4) = v_1$) with the differential of the **second** variable ($\mathrm{d}U(2) = \mathrm{d}u_2$) with associated value **-1**. Indeed, the third equation of (14) is the differentiated form of the definition of the first auxiliary variable that looks $0 = \mathrm{d}v_1 - v_1 \mathrm{d}u_2$.

4.4.3 Automatic generation of the tensor lists using polarization formula

The construction of the tensor lists from the equations can be done by hand for this simple example, but it is evident that an automated process has to be used

to deal with large systems. The key point in this list generation is the separation between the constant, the linear and the quadratic terms. This can be achieved by using the polarization formula as follows : C is computed by the evaluation of $R(0) = C + L(0) + Q(0,0) = C$. The vector $L(X)$ is computed by the evaluation of

$$R(X) - R(-X) = C + L(X) + Q(X,X) - C - L(-X) - Q(-X,-X) = 2L(X). \tag{40}$$

The vector $Q(X,Y)$ is computed by the evaluation of

$$\begin{array}{rcl} R(X+Y) - R(X-Y) & = & C + L(X+Y) + Q(X+Y, X+Y) \\ & & -(C + L(X-Y) + Q(X-Y, X-Y)) \\ & = & 2L(Y) + 2Q(X,Y) + 2Q(Y,X) \\ & = & 2L(Y) + 4Q(X,Y), \quad \text{if } Q \text{ is symmetric.} \end{array} \tag{41}$$

With the formula $R(Y) - R(-Y) = 2L(Y)$, one gets :

$$\begin{array}{rcl} C & = & R(0) \\ L(X) & = & \frac{1}{2}(R(X) - R(-X)) \\ Q(X,Y) & = & \frac{1}{4}\left(R(X+Y) - R(X-Y) - (R(Y) - R(-Y))\right) \end{array} \tag{42}$$

The lists are then constructed by applying the polarization formula (42) to all the vector of the canonical base. This process is automatic from the knowledge of the residue function (15).

5 Examples

The objective of this section is twofold. Firstly, to illustrate the quadratic recast on three more examples, some with elementary transcendental functions. Secondly, to illustrate the robustness of the continuation and the simple bifurcation detection. The input files that a user has to provide for running these examples in Manlab-4.0 are not given for brevity. The input files of the explanatory example, corresponding to (13) and (14), can be found on the web site Manlab at http://manlab.lma.cnrs-mrs.fr/.

5.1 Logistic map

The first example is the well-known logistic map. The sequence $(x_n)_{n\in\mathbb{N}}$ is defined by the following recurrence relation :

$$x_{n+1} = \mu x_n(1 - x_n) := f_\mu(x_n) \tag{43}$$

$0 \leq \mu \leq 4$ is the bifurcation parameter. Depending on its value, the sequence $(x_n)_{n\in\mathbb{N}}$ converges towards a fixed point, a periodic orbit or its behaviour is chaotic. The logistic map undergoes an infinite number of period doubling bifurcation when μ is increased up to a critical value $\mu_{\text{crit}} \simeq 3.56994567$ above which its behaviour is chaotic [27]. These bifurcations are detected using the method described in [15].

2^N–periodic orbits are sought for with $N \in \mathbb{N}$; in other words, fixed points of $f_\mu^{2^N}$:

$$x_1 = f_\mu^{2^N}(x_1) \tag{44}$$

As this equation (44) is not quadratic, $2^N - 1$ auxiliary variables that are the iterates of x_1 by f_μ along the orbit are introduced :

$$\forall n \in \{1, \ldots, 2^N - 1\}, x_{n+1} = f_\mu(x_n), \tag{45}$$

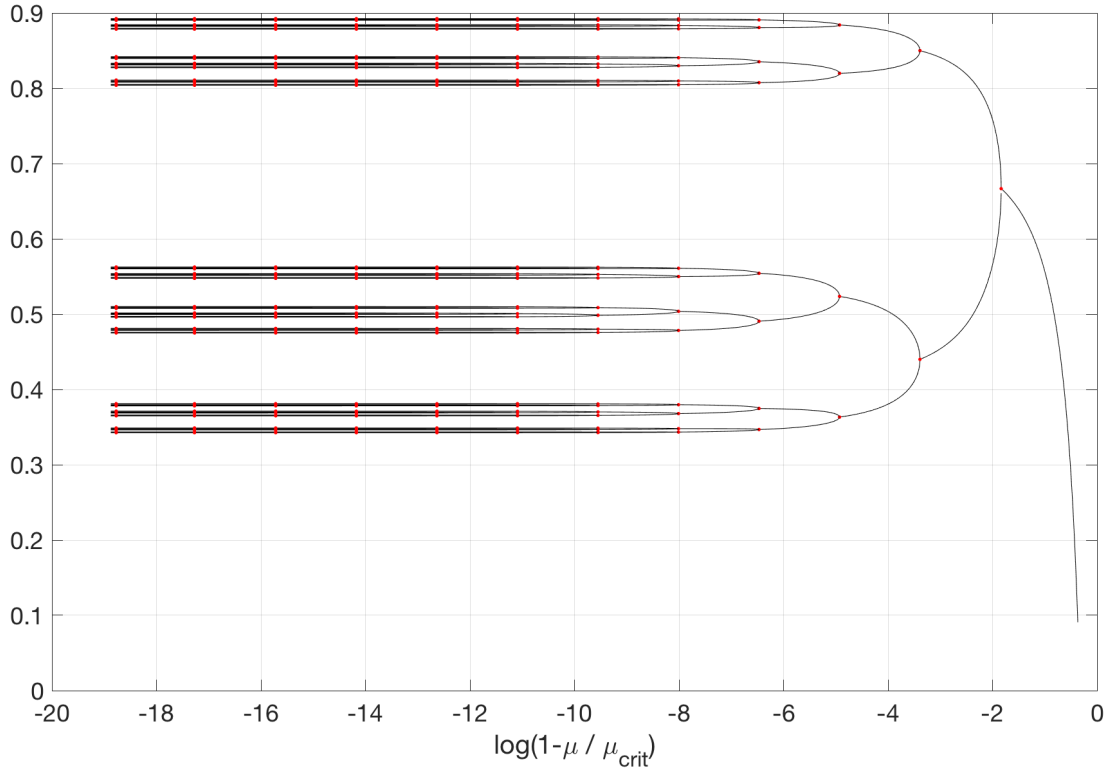

Figure 1: Bifurcation diagram of the logistic map obtained with Manlab 4.0. The red dots are period doubling bifurcations. The x–axis is $\log(1-\frac{\mu}{\mu_{\text{crit}}})$ where μ_{crit} is the critical value above which the system becomes chaotic. The 12 first bifurcations were computed.

and $\lambda = \frac{1}{\mu}$ is defined so that the equations are recast quadratically as follows :

$$\begin{array}{rcc} \lambda x_1 & = & x_{2^N}(1-x_{2^N}) \\ \hline \lambda\mu & = & 1 \\ \lambda x_2 & = & x_1(1-x_1) \\ \lambda x_3 & = & x_2(1-x_2) \\ \vdots & & \vdots \\ \lambda x_{2^N} & = & x_{2^N-1}(1-x_{2^N-1}) \end{array} \tag{46}$$

The horizontal lines materializes the separation between the main equation and the auxiliary equations. Feigenbaum [20] showed that the ratio of consecutive intervals between two period doubling converges toward a constant δ now called Feigenbaum's constant. This convergence is shown on figure 2. The last two bifurcations are closer than the machine precision as can be seen on figure 1. It is probable that the last bifurcation position was not computed with enough accuracy. This explains the position of the last point on figure 2.

This example is interesting since when N is set to 12 as it is done here, there is only one main equation (44) and $2^N = 4096$ auxiliary variables (46). The inversion of a sparse triangular matrix of size 4096 is immediate and then the system to solve becomes of size 1.

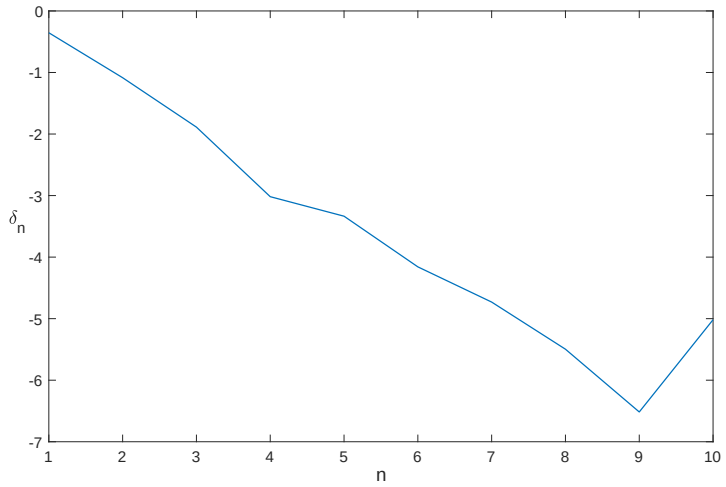

Figure 2: Convergence of the ratio of the length of two consecutive intervals between two period doubling bifurcations to Feigenbaum constant δ. $\delta_n = \log_{10}\left(\left|\frac{\mu_{n+2}-\mu_{n+1}}{\mu_{n+1}-\mu_n} - \delta\right|\right)$ is plotted to show the exponential rate of convergence unambiguously.

5.2 Layne–Watson

The Layne–Watson system of equations has been first introduced in [32]. The problem is to find the fixed point of a function g with a homotopy technique. This example has been chosen as it is very tough for continuation algorithms. There are no branch crossing but the bifurcation diagram is very intricate as can be seen on the last plot of figure 3. For $N \in \mathbb{N}$ and $x = (x_1, \ldots, x_N) \in \mathbb{R}^N$, g is defined by

$$\forall i \in \{1, \ldots, N\}, g_i(x) := \exp\left(\cos\left(i \sum_{k=1}^{N} x_k\right)\right) \tag{47}$$

To find the fixed point, the system

$$R(x, \lambda) = x - \lambda g(x) \tag{48}$$

is introduced. Obviously $R(\mathbf{0}, 0) = 0$. This point is used as a starting point of the continuation. If $\lambda = 1$ is reached throughout the continuation then a fixed point of g is known. The recast of the equations is, for $i \in \{1, \ldots, N\}$,

$$\begin{aligned} x_i - \lambda g_i &= 0 \\ c_i - \cos\left(i \textstyle\sum_{k=1}^{N} x_k\right) &= 0 \\ s_i - \sin\left(i \textstyle\sum_{k=1}^{N} x_k\right) &= 0 \\ g_i - \exp(c_i) &= 0 \end{aligned} \tag{49}$$

And the differentiated form of the last three vectorial equations, for $i \in \{1, \ldots, N\}$,

$$\begin{aligned} \mathrm{d}c_i + s_i \mathrm{d}\left(i \textstyle\sum_{k=1}^{N} x_k\right) &= 0 \\ \mathrm{d}s_i - c_i \mathrm{d}\left(i \textstyle\sum_{k=1}^{N} x_k\right) &= 0 \\ \mathrm{d}g_i - g_i \mathrm{d}c_i &= 0 \end{aligned} \tag{50}$$

For $N = 10$, 11 fixed points were find. For $N = 50$, 4000 continuation steps were needed to find 4 fixed points. The computation time was about 45 seconds on a laptop computer (two processors 2,5 GHz Intel Core i7 with 16 Go RAM).

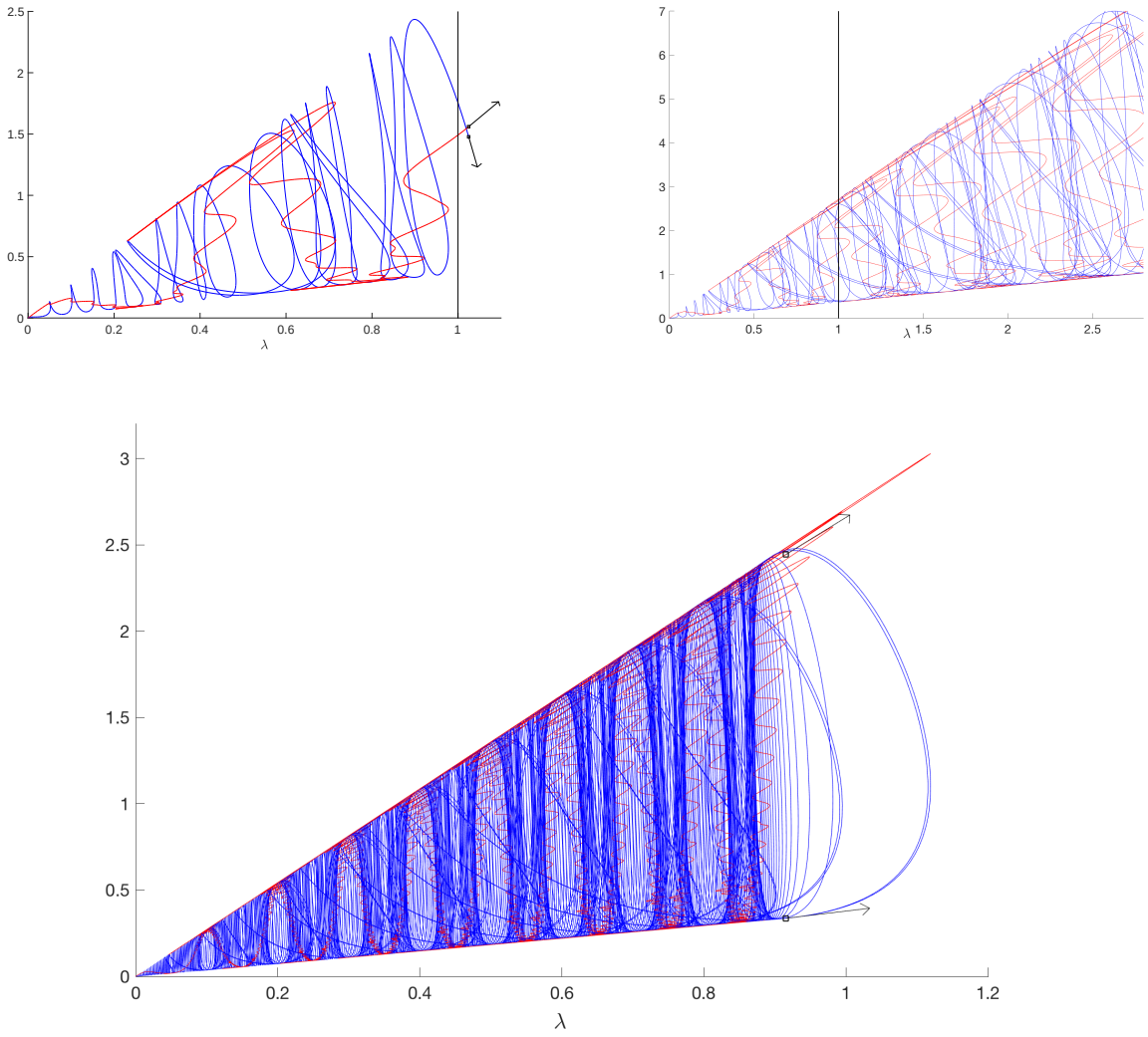

Figure 3: Bifurcation diagram of the Layne–Watson homotopy system (48). On the top with $N = 10$ and on the bottom with $N = 50$. On the plots, x_1 is in red and x_N is in blue.

In [29], the last value tried was $N = 130$ and some issues appeared while dropping the tolerance under 10^{-11}. This issue seems to persist here but above $N = 150$ approximately. For $N = 200$, a fixed point of g was found at the tolerance 10^{-11}, after more than 75000 continuation steps *i.e.* 35 minutes of computation, but it was not possible to find a fixed point at a smaller tolerance.

5.3 Stick-slip motion of a one d.o.f. oscillator

Let consider a one d.o.f. damped spring-mass oscillator which is driven by a belt with a constant velocity v_b. Its position $u(t)$ is governed by the (dimensionless) Newton law

$$\ddot{u} + 2\eta\,\dot{u} + u = \mu(v) \tag{51}$$

where η is the damping ratio, $\mu(v)$ the friction force and $v = \dot{u} - v_b$ the sliding velocity. The friction force is assumed to follow a Coulomb law with constant static and dynamic friction ratio denoted by μ_S and μ_d. Precisely, $-\mu_s < \mu(v) < \mu_s$ when $v = 0$ (stick), $\mu(v) = -\mu_d$ when $v > 0$ and $\mu(v) = \mu_d$ when $v < 0$ (slip). In the following, we look for a periodic solution made of a single stick phase and a single

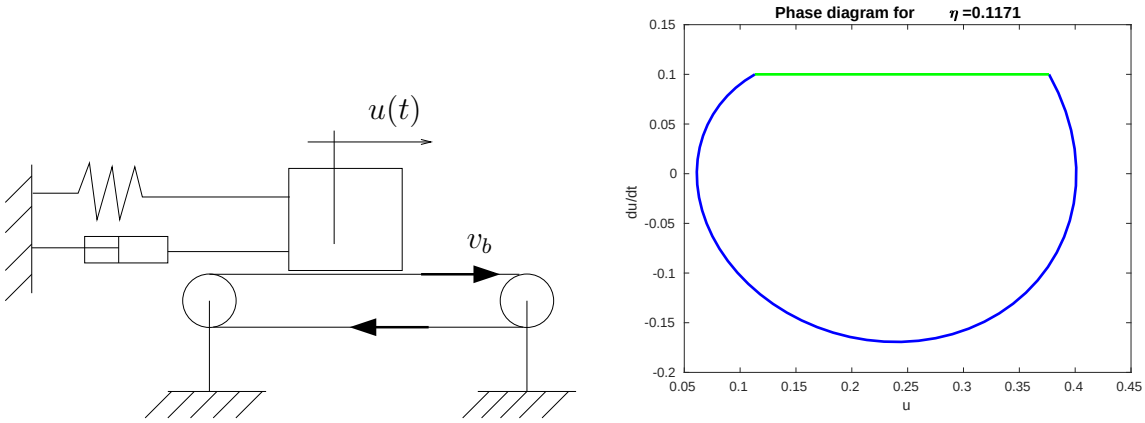

Figure 4: Scheme of the system and a phase portrait for $\eta = 0.1171$ and $v_b = 0.1$ taken from the full bifurcation diagram (not shown here).

slip phase.

Without loss of generality, the origin of time is taken at the transition between stick and slip, where $u(0) = \mu_s - 2\eta v_b$ and $\dot{u}(0) = v_b$. The slip phase occurs for $t \in [0, t_1]$ with $\mu(v) = \mu_d$. The solution to (51) with above mentioned initial conditions is

$$u_g(t) = \mu_d + e^{-\eta t}\left\{A\cos(\sqrt{1-\eta^2}\ t) + B\sin(\sqrt{1-\eta^2}\ t)\right\} \tag{52}$$

with $A = \mu_s - \mu_d - 2\ \eta\ v_b$ and $B = \frac{v_b+\eta A}{\sqrt{1-\eta^2}}$. The stick phase occurs for $t \in [-t_2, 0]$ with $u''(t) = 0$, and the solution is

$$u_s(t) = (\mu_s - 2\eta v_b) + v_b\ t \tag{53}$$

The main equation are the following periodicity conditions involving the two unknowns t_1, t_2 and the two free parameters η and v_b (μ_s and μ_d are constant).

$$\begin{aligned} u_g(t_1) &= u_s(t_2) \\ \dot{u}_g(t_1) &= \dot{u}_s(t_2) \end{aligned} \tag{54}$$

We introduce the following auxiliary variables

$$\begin{aligned} \omega_D &= \sqrt{1-\eta^2} \\ \theta &= \omega_D\ t_1 \\ C &= \cos\theta \\ S &= \sin\theta \\ A &= \mu_s - \mu_d - 2\ \eta\ v - b \\ B &= \tfrac{v_b+\eta\ A}{\omega_D} \\ D &= A * \omega_D + \eta\ B \\ X &= A\ C + B\ S \\ Y &= v_b\ C - D\ S \\ \alpha &= -\eta\ t_1 \\ E &= \exp\alpha \end{aligned} \tag{55}$$

and recast the main equations as

$$\begin{aligned} r_1(t_1, t_2, \eta, vb, \omega_D, \ldots, E) &= E\ X + 2\eta\ v_b + v_b\ t_2 + \mu_d - \mu_s \\ r_2(t_1, t_2, \eta, vb, \omega_D, \ldots, E) &= E\ Y - v_b \end{aligned} \tag{56}$$

Continuation with respect to η or v_b is performed by adding a third equation to stick one of these two parameters to a constant value. These diagrams are not shown here.

6 Application to finite element analysis in mechanics

As mentioned in the introduction, the Asymptotic Numerical Method has first been design for nonlinear finite element analysis in solid and structural mechanics. Several specialized finite element programs have been developed by the researcher working on the ANM each time a new formulation or a new class of nonlinearity was under study.

The interest of the generic Taylor series based continuation method presented here is to provide a unified framework for implementing a large class a finite element analysis whatever the complexity of the formulation or of the constitutive law. Firstly, recalling that the input of this generic method is only the quadratic recast of the equation (15) ((13) and (14) for the explanatory example), implementing a finite element analysis only requires a quadratic writing of the residual equation vector with suitable auxiliary variables. The consistent tangent stiffness matrix which is always a key issue in a finite element implementation is here automatically constructed from the sparse tensor lists and the block solving procedure. Secondly, the internal representation of the system with the sparse tensor lists allows to keep a very good performance of the computations even for complex mechanical model.

As an illustration of the performance of this method for finite element nonlinear analysis, the problem of geometrically nonlinear 2D elasticity in the framework of large displacements, small strains and a total Lagrangian formulation is addressed. A linear elastic constitute law is assumed for the simplicity of the presentation.

Consider an elastic body Ω submitted to a load growing proportionally to a load parameter λ. Let u be the displacement field, $X = \nabla u$ the displacement gradient tensor, $F = I_d + X$ the transformation gradient tensor, $E = \frac{1}{2}(X + X^t + X^t.X)$ the Green-Lagrange strain tensor, D the elastic constant tensor, S and P the second and first Piola-Kirchhoff stress tensor.

Taking the displacement field u as the main variable and the tensor field X, S and P as the auxiliary variables, a possible formulation of this continuous problem is : Find u and X, S, P satisfying

$$\text{Auxiliary equations} \quad \left\{ \begin{array}{l} X = \nabla u \\ S = D : (\frac{1}{2}(X + X^t + X^t.X)) \\ P = (I_d + X).S \end{array} \right. \tag{57}$$

$$\text{main equation} \quad \int_\Omega \delta F : P \, \mathrm{d}v - \lambda < F_{ext}, \delta u >= 0 \quad \forall \delta u \tag{58}$$

The auxiliary equations are quadratic and follows the linear declaration rule. They allows to compute P from a given u. The main equation corresponds to the virtual work theorem and expresses the equilibrium inside Ω. Notice that this equation is linear with respect to P.

We now introduce a classical displacement based Finite Element method (see for instance [5],[23]) with classical isoparametric elements. After discretization, the main variable u becomes the vector of active degree of freedom $\left[u\right]$ that collects displacements at the free nodes, and the auxiliary variable becomes the value of the tensor X, S and P at the Gauss point of the elements. Let introduce the notation $\left[u_e\right]$ for the vector of d.o.f of an element, $\left[P\right] = \left[P_{11}\ P_{12}\ P_{21}\ P_{22}\right]^T$ for the vector of components of the tensor P, (same notation for X, and $S = [S_{11}, S_{22}, S_{12}]^T$), $\left[G\right]$ for the gradient-displacement matrix on an element.

The discrete form of the virtual work equation becomes the residual equilibrium at the free node

$$\cup \int_{\Omega_e} \left[G\right]^T \left[P\right] \ \mathrm{dv} - \lambda \left[F_{ext}\right] \tag{59}$$

where $\cup$ stands for the classical assembly of the elementary internal forces vector. A classical Gauss integration is performed to evaluate these integrals at element level.

Finally, the discrete problem is : Find $U := \left[u\right], \lambda$ and the auxiliary variables $U_{aux} = \left[\left[X\right], \left[S\right], \left[P\right], \dots\right]$ at each Gauss point verifying

$$\text{Auxiliary equations at Gauss point} \begin{cases} \left[X\right] = \left[G\right]\left[u_e\right] \\ \left[S\right] = \left[D\right] \begin{bmatrix} X_{11} + (X_{11}^2 + X_{21}^2)/2 \\ X_{22} + (X_{12}^2 + X_{22}^2)/2 \\ (X_{12} + X_{21} + X_{11}X_{12} + X_{21}X_{22})/2 \end{bmatrix} \\ \left[P\right] = \begin{bmatrix} (1+X_{11}) * S_{11} + X_{12} * S_{12} \\ (1+X_{11}) * S_{12} + X_{12} * S_{22} \\ X_{21} * S_{11} + (1+X_{22}) * S_{12} \\ X_{21} * S_{12} + (1+X_{22}) * S_{22} \end{bmatrix} \end{cases} \tag{60}$$

$$\text{main equation } \cup \sum_{Gauss} \left[G\right]^T \left[P\right] det(J_e) w_e - \lambda \left[F_{ext}\right] \tag{61}$$

From these quadratic equations, the sparse tensor list are generated thanks to the use of the polarization formulas (42). Since the main variable is the vector of active d.o.f of the model, the jacobian matrix $J = \left(K - BK_{\text{aux}}^{-1}C\right)$ which is introduced for the block solving process is exactly the consistent tangent stiffness matrix of the finite element model.

In figures 5, the buckling behaviour of a sandwich-like beam with X stiffeners is considered as an illustrative example. The beam is clamped on the left and submitted to a uniform compressive force on the right. The bifurcation diagram is composed of a symmetric fundamental branch from which secondary branches bifurcate. The fundamental branch has been computed with 23 steps. Six bifurcation points have been automatically detected and located with the method explained in [15]. The first three bifurcated branches have been computed with 20 steps of continuation each. The deformed structure consists of a global buckling mode with local wrinkling of skins in the compressed parts. The three first deformed modes are shown in figure 5.

For this example, the mesh is composed of 282 quadrilateral 8 nodes elements with a 2×2 reduced integration scheme, the number of active d.o.f is 2616 and the number of auxiliary variables is 12408. The number of nonzero coefficients of the quadratic sparse tensor Q is here 31584. The computing time for these 83 steps of continuation is 25 seconds on a laptop computer (two processors 2,5 GHz Intel Core i7 with 16 Go RAM) with the Matlab software Manlab-4.0.

As discussed so far, the formulation of a mechanical model is not unique and it is also the case for its quadratic recast. A natural question is then "what is the best recast" and also "is it worth to have a minimal number of auxiliary variables". As an answer to these questions, the problem considered above has also been written by taking only the stress tensor S for the auxiliary variables. The consequence of not storing the gradient X and the first Piola-Kirchhoff P is a tremendous increase of the size of quadratic sparse tensor Q which pass from 32 terms per Gauss point for the previous formulation to more than a thousand. This results in a severe increase of the computing time of the right hand side $\mathrm{F}_p^{\mathrm{nl}}$ of the linear systems to be solved (25). The corresponding gain of the computing time in the block solving procedure that results from a smaller system is not significant. Hence, it follows that the best formulation

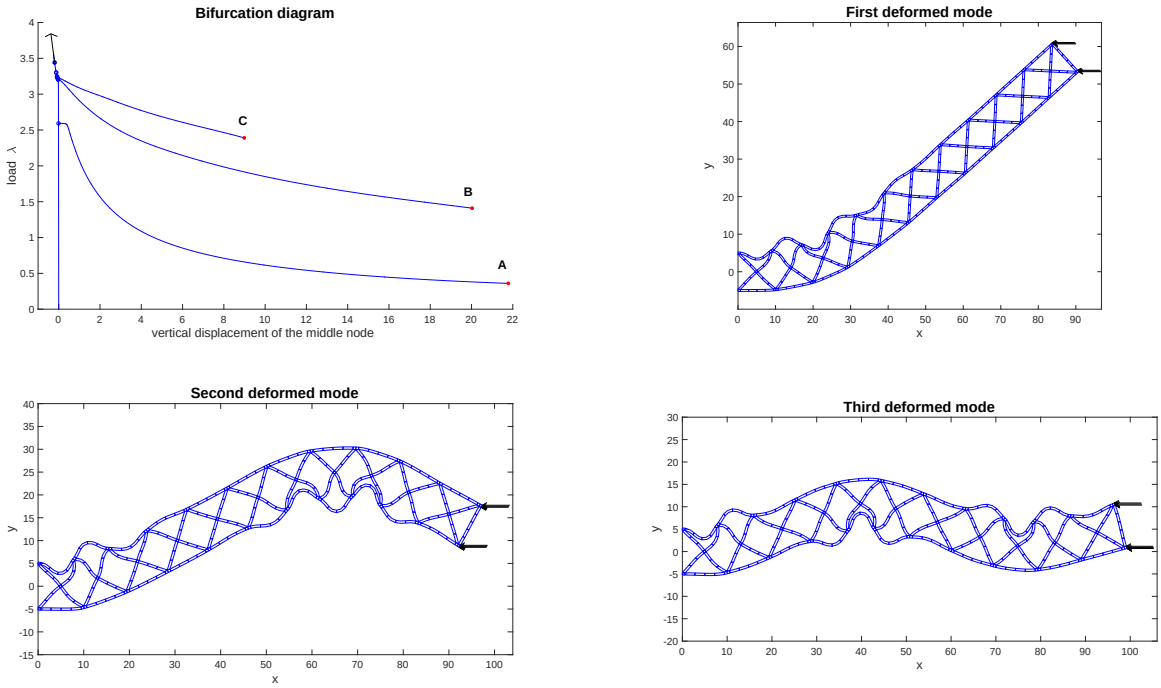

Figure 5: Bifurcation diagram of a sandwich-like beam with X stiffeners showing the three first deformed modes. Only an excerpt of the bifurcation diagram is shown here. The first, the second and the third mode are shown here and corresponds respectively to the red dot A, B and C.

is not the one with a minimal number of auxiliary variables but the one with the minimal size for the quadratic sparse tensor here.

To finish these comments on the use of the proposed method for finite element analysis, we come back to the elementary transcendental function. For the sake of simplicity, the mechanical model used here for the illustration does not contain any elementary transcendental function. This is however not the case for most non linear finite element analysis. for instance, the functions ln, exp and power are frequently used for the modelization of the constitutive law of nonlinear material. For the structural models such as beams, plates, and shells, rotational degrees of freedom are frequently introduced to described the rotation of the section, and as a consequence the functions sines and cosines also frequently appear in the formulation. From the preceding presentation on how to manage these functions by using the adequate auxiliary variables and adequate companion functions (see the table in appendix A), it is clear that these more complex mechanical models can also be treated with this generic continuation method. It should be noticed that, whatever the complexity of the model and whatever the use of transcendental functions, the consistent tangent stiffness matrix is automatically constructed as explained above.

7 Conclusion

An efficient implementation of a Taylor series based continuation algorithm is presented. Its need of a quadratic formulation of the equations is explained and extended to the case of elementary transcendental functions. The decades of experience using the ANM continuation resulted in fourth version of the continuation software Manlab

that is used here to treat successfully different systems. The accuracy of the computations is shown following the branches of the classical Logistic map and the twelve first flip bifurcations. The robustness of the method is demonstrated when applied to the Layne–Watson problem. The algorithm is then applied on a stick–slip problem, showing the possibility to make the continuation of a smooth by part system with this method. The last section presents a generic application to mechanical problems issued from FEM.

The quadratic recast of the system of equations is the key point of this continuation tool that allows to compute the Jacobian matrix and the Taylor series of the ANM automatically. The separation of the variables into main or auxiliary variables, the condensation of the auxiliary variables using a block solving technique and the sparse tensorial formalism ensure a very significant improvement of the computation time as compared to the previous versions of generic implementation of the ANM in the Manlab suite[2].

Acknowledgment

The authors want to thank Stéphane Bourgeois for fruitful discussions on the finite element analysis part.
This work has been carried out in the framework of the Labex MEC (ANR-10-LABX-0092) and of the A*MIDEX project (ANR-11-IDEX-0001-02), funded by the Investissements d'Avenir French Government program managed by the French National Research Agency (ANR).

Bibliography

[1] Manlab - an interactive path-following and bifurcation alnalysis software. available at https:\\manlab.lma.cnrs-mrs.fr.

[2] H. Abichou, H. Zahrouni, and M. Potier-Ferry. Asymptotic numerical method for problems coupling several nonlinearities. *Computer methods in applied mechanics and engineering*, 191:5795–5810, 2002.

[3] W. Aggoune, H. Zahrouni, and M. Potier-Ferry. Asymptotic numerical methods for unilateral contact. *International journal for numerical methods in engineering*, 68(6):605–631, 2006.

[4] C. Allery, J.-M. Cadou, A. Hamdouni, and D. Razafindralandy. Application of the asymptotic numerical method to the coanda effect study. *Revue Européenne des Eléments*, 13(1-2):57–77, 2004.

[5] K.-J. Bathe. *Finite element procedures*. Prentice-Hall Inc, 1996.

[6] T. F. Chan and H. Keller. Arc-length continuation and multigrid techniques for nonlinear elliptic eigenvalue problems. *SIAM Journal on Scientific and Statistical Computing*, 3(2):173–194, 1982.

[7] I. Charpentier and B. Cochelin. Towards a full higher order ad-based continuation and bifurcation framework. *Optimization Methods & Software*, 2018.

[2]Manlab 4.0 is available online on the dedicated website https://manlab.lma.cnrs-mrs.fr/ .

[8] I. Charpentier, B. Cochelin, and K. Lampoh. Diamanlab-an interactive taylor-based continuation tool in matlab. 2013.

[9] I. Charpentier, A. Lejeune, and M. Potier-Ferry. The diamant approach for an efficient automatic differentiation of the asymptotic numerical method. *Advances in Automatic Differentiation*, pages 139–149, 2008.

[10] I. Charpentier and M. Potier-Ferry. Différentiation automatique de la méthode asymptotique numérique typée: l'approche diamant. *Comptes Rendus Mécanique*, 336(3):336–340, 2008.

[11] B. Cochelin. A path-following technique via an asymptotic-numerical method. *Computers & structures*, 53(5):1181–1192, 1994.

[12] B. Cochelin, N. Damil, and M. Potier-Ferry. Asymptotic–numerical methods and pade approximants for non-linear elastic structures. *International journal for numerical methods in engineering*, 37(7):1187–1213, 1994.

[13] B. Cochelin, N. Damil, and M. Potier-Ferry. The asymptotic-numerical method: an efficient perturbation technique for nonlinear structural mechanics. *Revue européenne des éléments finis*, 3(2):281–297, 1994.

[14] B. Cochelin, N. Damil, and M. Potier-Ferry. Méthode asymptotique numérique. 2008.

[15] B. Cochelin and M. Medale. Power series analysis as a major breakthrough to improve the efficiency of asymptotic numerical method in the vicinity of bifurcations. *Journal of Computational Physics*, 236:594–607, 2013.

[16] B. Cochelin and C. Vergez. A high order purely frequency-based harmonic balance formulation for continuation of periodic solutions. *Journal of sound and vibration*, 324(1):243–262, 2009.

[17] E. Doedel, H. B. Keller, and J. P. Kernevez. Numerical analysis and control of bifurcation problems (i): bifurcation in finite dimensions. *International journal of bifurcation and chaos*, 1(03):493–520, 1991.

[18] E. J. Doedel and R. F. Heinemann. Numerical computation of periodic solution branches and oscillatory dynamics of the stirred tank reactor with a yields b yields c reactions. Technical report, Wisconsin univ-Madison mathematics research center, 1982.

[19] L. Duigou, E. Daya, and M. Potier-Ferry. Iterative algorithms for nonlinear eigenvalue problems. application to vibrations of viscoelastic shells. *Computer methods in applied mechanics and engineering*, 192:1323–1335, 2003.

[20] M. J. Feigenbaum. The universal metric properties of nonlinear transformations. *Journal of Statistical Physics*, 21(6):669–706, 1979.

[21] K. Fritzsche and H. Grauert. *From holomorphic functions to complex manifolds*, volume 213. Springer Science & Business Media, 2012.

[22] L. Guillot, P. Vigué, C. Vergez, and B. Cochelin. Continuation of quasi-periodic solutions with two-frequency harmonic balance method. *Journal of Sound and Vibration*, 394:434–450, 2017.

[23] T. J. Hughes. *The finite element method: linear static and dynamic finite element analysis*. Prentice-Hall Inc, 1987.

[24] S. Karkar, B. Cochelin, and C. Vergez. A high-order, purely frequency based harmonic balance formulation for continuation of periodic solutions: The case of non-polynomial nonlinearities. *Journal of Sound and Vibration*, 332(4):968–977, 2013.

[25] H. Keller. Lectures on numerical methods in bifurcation problems. *Applied Mathematics*, 217:50, 1987.

[26] N. Kessab, B. Braikat, H. Lahmam, E. Mallil, N. Damil, and M. Potier-Ferry. High order predictor-corrector algorithms for strongly nonlinear problems. *Revue de*, 1(8):587–613, 2006.

[27] T.-Y. Li and J. A. Yorke. Period three implies chaos. *The American Mathematical Monthly*, 82(10):985–992, 1975.

[28] M. Medale and B. Cochelin. High performance computations of steady-state bifurcations in 3d incompressible fluid flows by asymptotic numerical method. *Journal of Computational Physics*, 299:581–596, 2015.

[29] N. S. Nedialkov and J. D. Pryce. Solving differential-algebraic equations by taylor series (iii): the daets code. *Journal of Numerical Analysis, Industrial and Applied Mathematics*, 1(1):1–30, 2007.

[30] S. Nezamabadi, J. Yvonnet, H. Zahrouni, and M. Potier-Ferry. A multilevel computational strategy for handling microscopic and macroscopic instabilities. *Computer methods in applied mechanics and engineering*, 198:2099–2110, 2009.

[31] R. Seydel. *From equilibrium to chaos: practical bifurcation and stability analysis*. North-Holland, 1988.

[32] L. T. Watson. A globally convergent algorithm for computing fixed points of c2 maps. *Applied mathematics and computation*, 5(4):297–311, 1979.

[33] H. Zahrouni, B. Cochelin, and M. Potier-Ferry. Computing finite rotations of shells by an asymptotic-numerical method. *Computer methods in applied mechanics and engineering*, 175(1-2):71–85, 1999.

[34] H. Zahrouni, M. Potier-Ferry, H. Elasmar, and N. Damil. Asymptotic numerical method for nonlinear constitutive laws. *Revue européenne des éléments finis*, 7(7):841–869, 1998.

[35] R. Zucker. Elementary transcendental functions: Logarithmic, exponential, circular and hyperbolic functions. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables. New York: Dover*, pages 65–94, 1965.

A Table of quadratic recasts of the most common elementary transcendental functions

auxiliary variable	companion variable(s)	full quadratic recast	differentiated form (if needed)
$v = \exp(u)$		$v - \exp(u) = 0$	$\mathrm{d}v - v\mathrm{d}u = 0$
$v = \log(u)$	$w = \frac{1}{u}$	$v - \log(u) = 0$ $u \times w - 1 = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$
$v = u^{\alpha}$	$w = \frac{v}{u}$	$v - u^{\alpha} = 0$ $w \times u - v = 0$	$\mathrm{d}v - \alpha w\mathrm{d}u = 0$
$v = \sin(u)$	$w = \cos(u)$	$v - \sin(u) = 0$ $w - \cos(u) = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$ $\mathrm{d}w + v\mathrm{d}u = 0$
$v = \cos(u)$	$w = \sin(u)$	$v - \cos(u) = 0$ $w - \sin(u) = 0$	$\mathrm{d}v + w\mathrm{d}u = 0$ $\mathrm{d}w - v\mathrm{d}u = 0$
$v = \tan(u)$	$w = 1 - v^2$	$v - \tan(u) = 0$ $w - 1 + v^2 = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$
$v = \sinh(u)$	$w = \cosh(u)$	$v - \sinh(u) = 0$ $w - \cosh(u) = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$ $\mathrm{d}w - v\mathrm{d}u = 0$
$v = \cosh(u)$	$w = \sinh(u)$	$v - \cosh(u) = 0$ $w - \sinh(u) = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$ $\mathrm{d}w - v\mathrm{d}u = 0$
$v = \tanh(u)$	$w = 1 + v^2$	$v - \tanh(u) = 0$ $w - 1 - v^2 = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$
$v = \arcsin(u)$	$z = \sqrt{1 - u^2}$ $w = \frac{1}{z}$	$v - \arcsin(u) = 0$ $z^2 - 1 + u^2 = 0$ $w \times z - 1 = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$
$v = \arccos(u)$	$z = \sqrt{1 - u^2}$ $w = \frac{-1}{z}$	$v - \arccos(u) = 0$ $z^2 - 1 + u^2 = 0$ $w \times z + 1 = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$
$v = \arctan(u)$	$z = 1 + u^2$ $w = \frac{1}{z}$	$v - \arctan(u) = 0$ $z - 1 - u^2 = 0$ $w \times z - 1 = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$
$v = \mathrm{argsh}(u)$	$z = \sqrt{u^2 + 1}$ $w = \frac{1}{z}$	$v - \mathrm{argsh}(u) = 0$ $z^2 - 1 - u^2 = 0$ $w \times z - 1 = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$
$v = \mathrm{argch}(u)$	$z = \sqrt{u^2 - 1}$ $w = \frac{1}{z}$	$v - \mathrm{argch}(u) = 0$ $z^2 + 1 - u^2 = 0$ $w \times z - 1 = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$
$v = \mathrm{argth}(u)$	$z = 1 - u^2$ $w = \frac{1}{z}$	$v - \mathrm{argth}(u) = 0$ $z - 1 + u^2 = 0$ $w \times z - 1 = 0$	$\mathrm{d}v - w\mathrm{d}u = 0$