Drawing Solution Curve of Differential Equation
Farida Benmakrouha, Christiane Hespel, Edouard Monnier

To cite this version:

HAL Id: hal-00567060
https://hal.science/hal-00567060
Submitted on 21 Feb 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers. L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
DRAWSNIG SOLLUTIO CURVE OF DIFFERENTIAL EQUATION

Farida Benmakrouha, Christiane Hespel, and Edouard Monnier

Abstract. We develop a method for drawing solution curves of differential equations. This method is based on the juxtaposition of local approximating curves on successive intervals $[t_i, t_{i+1}]_{0 \leq i \leq n-1}$. The differential equation, considered as a dynamical system, is described by its state equations and its initial value at time $t_0$.

A generic expression of its generating series $G_t$ truncated at any order $k$, of the output and its derivatives $y^{(j)}(t)$ expanded at any order $k$, can be calculated. The output and its derivatives $y^{(j)}(t)$ are expressed in terms of the coefficients of the series $G_t$ and of the Chen series.

At the initial point $t_i$ of each interval, we specify the expressions of $G_t$ and $y^{(j)}(t)$. Then we obtain an approximated output $y(t)$ at order $k$ in every interval $[t_i, t_{i+1}]_{0 \leq i \leq n-1}$.

We have developed a Maple package corresponding to the creation of the generic expression of $G_t$ and $y^{(j)}(t)$ at order $k$ and to the drawing of the local curves on every interval $[t_i, t_{i+1}]_{0 \leq i \leq n-1}$. For stable systems with oscillating output, or for unstable systems near the instability points, our package provides a suitable result when a Runge-Kutta method is wrong.

Keywords. Curve drawing, differential equation, symbolic algorithm, generating series, dynamical system, oscillating output

1 Introduction

The usual methods for drawing curves of differential equations consist in an iterative construction of isolated points (Runge-Kutta). Rather than calculate numerous successive approximative points $y(t_i)_{i \in I}$, it can be interesting to provide some few successive local curves $\{y(t)\}_{t \in [t_i, t_{i+1}]_{0 \leq i \leq n-1}}$.

Moreover, the computing of these local curves can be kept partly generic since a generic expression of the generating series $G_t$ of the system can be provided in terms of $t_i$.

The expression of the local curves $\{y(t)\}_{t \in [t_i, t_{i+1}]}$ is only a specification for $t = t_i$ at order $k$ of the formula given in the proposition of section 3.

We consider a differential equation

$$y^{(N)}(t) = \phi(t, y(t), \ldots, y^{(N-1)}(t))$$

with initial conditions

$$y(0) = y_0, 0, \ldots, y^{(N-1)}(0) = y_{0, N-1}$$

We assume that $\phi(t, y(t), \ldots, y^{(N-1)}(t))$ is polynomial in $y, \ldots, y^{(N-1)}$.

Then this differential equation can be viewed as an affine single input dynamical system.

2 Preliminaries

2.1 Affine system, Generating series

We consider the nonlinear analytical system affine in the input:

$$y^{(j)}(t) = g_0(q) + \sum_{j=1}^{m} f_j(q)u_j(t) \quad (\Sigma)$$

with $y(0) = y_0, 0, \ldots, y^{(N-1)}(0) = y_{0, N-1}$.

We assume that $\phi(t, y(t), \ldots, y^{(N-1)}(t))$ is polynomial in $y, \ldots, y^{(N-1)}$.

2.2 Fliess’s formula and iterated integrals

The output $y(t)$ is given by the Fliess’s equation (2)

$$y(t) = \sum_{w \in Z^*} \langle G_0 | w \rangle \int_{0}^{t} \delta(w)$$

where $G_0$ is the generating series of (Sigma) at $t = 0$.

$$G_0 = \sum_{w \in Z^*} \langle G_0 | w \rangle = \sum_{l \geq 0} \sum_{j_1, \ldots, j_l} f_{j_0} \cdots f_{j_l} g(q) |_{q = 0} z_{j_0} \cdots z_{j_l}$$

and $\int_0^t \delta(w)$ is the integral associated with the word $w \in Z^*$.

The iterative integral $\int_0^t \delta(w)$ of the word $w$ for the input $u$ is defined by

$$\int_0^t \delta(w) = \frac{1}{\tau^2} \int_{u \leq \tau} \delta(u)$$

where $\epsilon$ is the empty word, $u_0 \equiv 1$ is the drift and $u_0 [1, m]$ is the ith input.

We define the Chen’s series as follows (1)

$$C_u(t) = \sum_{w \in Z^*} \langle G_0 | w \rangle C_u(t)$$

We set

$$\xi_{i, 1}(t) = \int_{0}^{t} u_i(\tau) d\tau$$

From the previous definitions, we obtain the following expression:

$$y(t) = \sum_{w \in Z^*} \langle G_0 | w \rangle \langle C_u(t) | w \rangle$$

2.3 Iterated derivatives $y^{(N)}(0)$ of the output

$G_0$ being the generating series of the system, the $i$th derivative of $y(t)$ is

$$y^{(i)}(t) = \langle G_0 | C^{(i)}_u(t) \rangle$$

We prove the following lemma (5) based on the Picart-Vessiot theory (3).

Lemma:

Let be $\sum_{0 \leq j \leq m} u_j z_j = A$. Then the derivative of the Chen’s series is $\frac{d}{dt} C_u = C_u A$. 

1 F. Benmakrouha, C. Hespel, E. Monnier are with the Department of Computer Science, INSA-IRISA, 20 avenue des Buttes de Coesmes, 35043 Rennes cedex, France, E-mail: Benma@insa-rennes.fr, Hespel@insa-rennes.fr, Monnier@insa-rennes.fr

2 Manuscript received October 5, 2009.
From it, results the following recurrence relation:

\[ C_u^{(i)} = C_u A_i, \quad A_1 = A, \quad A_{i+1} = AA_i + D_t A_i \]  
(10)

\[ D_t \] being the operator of time derivation.

Since \( C_u(0) = 1 \) and \( C_u^{(i)}(0) = A_i(0) \) then

\[ y^{(i)}(0) = \sum_{w \in Z^*} (G_0|w)(C_u^{(i)}(0)|w) = \langle G_0|A_i(0) \rangle \]  
(11)

Let us remark that the successive derivatives \( y_0(t), y^{(1)}(t), \ldots, y^{(k)}(t) \) are obtained from the coefficients \( \langle G_0|w \rangle \) associated with the words whose length is \( \leq k \).

It results that the Taylor expansion of \( y(t) \) up to order \( k \) only depends on the coefficients \( G_0(w) \) truncated at order \( k \).

For instance, for a single input \( u(t) \) with drift \( u_0(t) \equiv 1 \), the derivatives are the following

\[ y(0) = \langle G_0|\epsilon \rangle \]
\[ y^{(1)}(0) = \langle G_0|z_0 \rangle + \langle G_0|z_1 \rangle u(0) \]
\[ y^{(2)}(0) = \langle G_0|z_0^2 \rangle + \langle G_0|z_0 z_1 \rangle u(0) + \langle G_0|z_1^2 \rangle u(0)^2 + \langle G_0|z_1 u^{(1)}(0) \rangle \]

\[ \ldots = \ldots \]

This method allows us to compute recursively the successive derivatives of \( y(t) \) at \( t = 0 \).

### 3 Main results

#### 3.1 Approximate value of \( y^{(n)}(t) \)

The Fließ’s formula can be written

\[ y(t) = \langle G_0|\epsilon \rangle + \sum_{w \in Z^* - \{\epsilon\}} (G_0|w)(C_u(t)|w) \]  
(13)

An approximate function \( y_k(t) \) to \( y(t) \) up to order \( k \) in a neighborhood of \( t = 0 \) is obtained by expanding this expression up to the same order \( k \). Then we have

\[ |y(t) - y_k(t)| = O(t^{k+1}) \]  
(14)

For instance, at order \( k = 1 \), \( y(t) \) has the following approximate expression for a single input with drift

\[ y_1(t) = \langle G_0|\epsilon \rangle + \langle G_0|z_0 \rangle t + \langle G_0|z_1 \rangle \xi_1(t) \]  
(15)

where \( \xi_k(t) \) denotes the \( k \)th primitive of \( u(t) \).

This computing can be generalized to the successive derivatives of \( y(t) \).

**Proposition**

Given the expression of \( y^{(n)}(t) \) in terms of the coefficients of \( G_0 \) and of the derivatives of order \( \leq n-1 \) of the input \( u(t) \) obtained recursively according to the previous section, we can deduce the expression of \( y^{(n)}(t) \) by executing in \( y^{(n)}(t) \) the following transformations

1. We substitute \( u^{(i)}(t) \) to \( u_0^{(i)}(t) \) for \( 0 \leq i \leq n-1 \)
2. For every occurrence of a coefficient \( \langle G_0|w \rangle \) where \( v \in Z^* \), we add the following corrective term

\[ \sum_{w \neq v} (G_0|w)(C_u(t)|w) \]

The proof is based on the following properties

\[ \sum_{v \neq u} \langle G_u|v \rangle u(t) = \langle G_u|u \rangle u(t) \]
(16)

For instance, for a single input with drift, we compute from

\[ y^{(1)}(0) = \langle G_0|z_0 \rangle + \langle G_0|z_1 \rangle u(0) \]

the expression of \( y^{(1)}(t) \):

\[ y^{(1)}(t) = \langle G_0|z_0(t) \rangle + \sum_{w \neq \epsilon} (G_0|w)(C_u(t)|w) + \langle G_0|z_1(t) \rangle u(t) \]  
(17)

By restricting the sums to the words \( w \) whose length \( |w| \) satisfies \( 1 \leq |w| \leq k \), we obtain a function \( y_k^{(n)}(t) \) approximating \( y^{(n)}(t) \) up to order \( k \). And then

\[ |y_k^{(n)}(t) - y^{(n)}(t)| = O(t^{k+1}) \]  
(18)

#### 3.2 Generalization at time \( t = t_i \)

For a single input with drift, the system \( (\Sigma) \) can be written at \( t = t_i \):

\[ \begin{aligned}
q(t_i + h) &= f_0(q(t_i) + h) + f_1(q(t_i) + h)u(t_i + h) \\
y(t_i + h) &= q_0(q(t_i) + h) 
\end{aligned} \]

By setting

\[ \begin{aligned}
U_i(h) &= u(t_i + h) \\
Y_i(h) &= y(t_i + h) \\
Q_i(h) &= q(t_i + h) 
\end{aligned} \]

we obtain the following system

\[ (\Sigma_i) \]

\[ \begin{aligned}
Q_i(h) &= f_0(Q_i(h)) + f_1(Q_i(h))U_i(h) \\
Y_i(h) &= q_0(Q_i(h)) 
\end{aligned} \]

And \( G_i \) is the generating series of \( (\Sigma_i) \).

By setting \( \psi_{i,k}(h) = \xi_k(t_i + h) \), then \( \psi_{i,k}(h) \) is the \( k \)th primitive of \( u(t_i + h) \) or the \( k \)th primitive of \( U_i(h) \).

We have the equalities

\[ \xi_1(t_i + h) = \int_{t_i}^{t_i + h} u(r)dr = \int_0^h U_i(t)dt = \psi_{1,i}(h) \]

And then, we can prove recursively that the Chen’s integral \( \int_{t_i}^{t_i + h} \delta(W) \) can be computed as an integral \( \int_0^h \delta(W) \) by considering \( U_i(t) \) instead of \( u(t_i + t) \).

### 4 Application to curves drawing

We present an application to the curve drawing of the solution of differential equations. We consider a differential equation

\[ y^{(N)}(t) = \phi(t, y(t), \ldots, y^{(N-1)}(t)) \]  
(23)

with initial conditions

\[ y(0) = y_{0,0}, \ldots, y^{(N)}(0) = y_{0,N} \]

It can be written for \( y = q_1 \):

\[ \begin{aligned}
q_1^{(1)} &= q_2 \\
& \ldots \ldots \\
q_1^{(N)} &= \phi(t, q_1, \ldots, q_N) 
\end{aligned} \]

We assume that

\[ \phi(t, q_1, \ldots, q_N) = P_0(q_1, \ldots, q_N) + \sum_{j=1}^m P_j(q_1, \ldots, q_N)u_j(t) \]

for \( P_0, P_1, \ldots, P_N \) polynomials in commutative variables \( q_1, \ldots, q_N \).

For an analytical affine single input system \( (\Sigma) \) then \( m = 1 \) and the vector fields are \( f_0, f_1 \), corresponding to \( P_0, P_1 \).

We propose a curve drawing of the output \( y(t) \) of this system in \([0, T] = \bigcup_{0 \leq s \leq N-1} [t_i, t_{i+1}] \) according to the following algorithm:

Firstly, we compute a generic expression of the generating series \( G_i \).
The vector fields are

\[ \langle G_i | z_j \rangle = f_{j_0} \cdots f_{j_l} g(q) | q(t_i) \]

is generic.

The computing of the expressions of

\[ Y_i(h) = y(t_i + h) = y(t_i) + \sum_{|w| \leq k} \langle G_i | w \rangle (C_{U_i}(h) | w) \]

and of

\[ Y_i^{(1)}(h) = \langle G_i | z_0 \rangle + \sum_{1 \leq |w| \leq k} \langle G_i | w_0 z_0 \rangle (C_{U_i}(h) | w) + \langle (G_i | z_1) + \sum_{1 \leq |w| \leq k} \langle G_i | w_1 z_1 \rangle (C_{U_i}(h) | w) U_i(h) \]  

(25)

are generic too.

We use the previous algorithm by specifying \( t_i \) at every step in the previous expressions.

### 4.2 Example 1: Duffing equation

Its equation is the following:

\[
\begin{align*}
y''(t) + ay'(t) + by(t) + cy^3(t) &= u(t) \\
y(0) &= y_0, \\
y'(0) &= y_1 \end{align*}
\]

(26)

It can be written as a first order differential system

\[
\begin{align*}
q_1^{(1)}(t) &= q_2(t) \\
q_2^{(1)}(t) &= -aq_2(t) - bq_1(t) - cy_1^3(t) + u(t) \\
y(t) &= q_1(t) = g(q) \\
y_1(0) &= y_0, \\
y_2(0) &= y_1 \end{align*}
\]

(27)

The vector fields are

\[
\begin{align*}
f_0(q_1, q_2) &= q_2^2 - (aq_2 + bq_1 + cy_1^3) \frac{\partial}{\partial q_2} \\
f_1(q_1, q_2) &= q_2 \frac{\partial}{\partial q_2} \\
f_2(q_1, q_2) &= 0
\end{align*}
\]

1. We write generic equations describing the generating series \( G_i \) at \( t = t_i \):

\[
\forall t_i \quad \langle G_i | z_{j_1} \cdots z_{j_{l_0}} \rangle = (f_{j_1} \cdots f_{j_{l_0}} g(q)) | q(t_i)
\]

Let us remark that

\[
\langle G_i | w_{z_1} \rangle = 0 \quad \forall w \in Z^+, \quad \langle G_i | w_0 z_0 \rangle = 0 \quad \forall w \in Z^+
\]

For instance, for order \( k = 3 \), we have only to compute 6 coefficients of \( G_i \) instead of 15 coefficients.

\[
\begin{align*}
\langle G_i | e \rangle &= q_1(t_i) \\
\langle G_i | z_0 \rangle &= q_2(t_i) \\
\langle G_i | z_2 \rangle &= F(q(t_i)) \\
\langle G_i | z_{12} \rangle &= 1 \\
\langle G_i | z_0^2 \rangle &= \langle q_2 \frac{\partial}{\partial q_2} F(q) + F(q) \frac{\partial}{\partial q_2} F(q) \rangle | q(t_i) \\
\langle G_i | z_0 z_2 \rangle &= -a
\end{align*}
\]

(28)

2. We write generic approximate expression of the output \( y(t_{i+1}) \) and its derivative \( y^{(1)}(t_{i+1}) \) for every \( t = t_{i+1} = t_i + h \) at order \( k \):

\[
\begin{align*}
y(t_{i+1}) &= \langle G_i | e \rangle + \sum_{1 \leq |w| \leq k} \langle G_i | w \rangle (C_{U_i}(h) | w) \\
y^{(1)}(t_{i+1}) &= \langle G_i | z_0 \rangle + \sum_{1 \leq |w| \leq k} \langle G_i | w_0 z_0 \rangle (C_{U_i}(h) | w) + \langle (G_i | z_1) + \sum_{1 \leq |w| \leq k} \langle G_i | w_1 z_1 \rangle (C_{U_i}(h) | w) U_i(h) \]
\]

(29)

For instance, for order \( k = 3 \)

\[
\begin{align*}
y(t_{i+1}) &= y(t_i) + \langle G_i | z_0 \rangle h + \langle G_i | z_0^2 \rangle h^2/2 + \langle G_i | z_1 z_0 \rangle \psi_{i,2}(h) + \langle G_i | z_0^3 \rangle h^3/3! + \langle G_i | z_0^2 z_2 \rangle \psi_{i,3}(h) \\
y^{(1)}(t_{i+1}) &= y^{(1)}(t_i) + h \quad \forall a(q(t)) + u(t) \\
y(t) &= q(t), q(0) = y_0
\end{align*}
\]

(30)

and

\[
\begin{align*}
y(t_{i+1}) &= y(t_i) + \langle G_i | z_0 \rangle h + \langle G_i | z_0^2 \rangle h^2 + \langle G_i | z_1 z_0 \rangle \psi_{i,2}(h) + \langle G_i | z_0^3 \rangle h^3/2 + \langle G_i | z_0^2 z_2 \rangle \psi_{i,3}(h) \\
y^{(1)}(t_{i+1}) &= y^{(1)}(t_i) + h \quad \forall a(q(t)) + u(t) \\
y(t) &= q(t), q(0) = y_0
\end{align*}
\]

(31)

3. And we use the algorithm of section 4 by specifying \( t_i \) at every step. So we obtain the drawing of \( y(t) \).
4.4 Maple package: some demonstrations

In this section, we produce a demonstration in the following cases

- For stable system (electric equation with positive parameters) for oscillating input \( u(t) = \sin(100t) \), step=0.01 (Runge-Kutta vs our method)

![Fig.1: Stable system, oscillating input, small step, by Runge-Kutta or our method](image1)

The drawings are similar by both methods (Runge-Kutta or our method).

- For stable system (electric equation, linear equation) with oscillating output (Runge-Kutta vs our method vs Exact Solution)

1. Electric equation with positive parameters for oscillating input \( u(t) = \sin(100t) \), step= 0.5 (Runge-Kutta vs our method)

![Fig.2: Stable system, oscillating input, large step, by Runge-Kutta (without oscillation) and our method](image2)

The oscillations of the output are not described by Runge-Kutta method when our method displays a lot of oscillations.

2. Linear equation for oscillating input \( u(t) = t^2 \sin(100t) \), step= 0.05 (Runge-Kutta vs our method vs Exact Solution)

![Fig.3: Linear equation, oscillating input, small step by Runge-Kutta](image3)

![Fig.4: Linear equation, oscillating input, small step by our method](image4)

![Fig.5: Linear equation, oscillating input, small step by exact method](image5)

The drawing of the exact solution is similar to the drawing of our method.

- For unstable system (electric equation with negative parameters), \( u(t) = \sin(100t) \), step = 0.01., Runge-Kutta method notifies an error when our method displays a suitable curve.
The Runge-Kutta method does not apply to this case when the drawing of our method displays an infinite branch.

5 Conclusion

We develop a method for drawing a solution curve of a differential equation, based on the symbolic computing.

The symbolic computing allows us to profit from the genericity: We propose that one uses the formal expression of the generating series \( G_i \) and of the output \( y(t_i) \) and its derivative \( y^{(1)}(t_i) \). Then we replace these expressions by their values at every step.

The symbolic computing allows us to profit from the precision: We can choose any order \( k \) for approximating the output and its derivative. The error is on the order of \( k + 1 \).

And then an interest of this method consists in choosing the precision, not only by the size of the time interval \( h \) but by the order of the approximation.

The quality of any approximation depends on the order, the size of the interval but also depends on the roughness of the curve and the stability of the system. From a lot of examples, we express the following conclusions:

For stable systems with smooth outputs, our method and a Runge-Kutta method provide similar results.

For unstable systems, our methods allows us to obtain a suitable result near the instability points, when the Runge-Kutta methods give an error message.

For stable systems with rough or oscillating outputs, our method provides a suitable result when a Runge-Kutta method is wrong.

References


