Quadratization of Differential Equations

Bychkov Andrey
Moscow Institute of Electronics and Mathematics
Higher School of Economics
Moscow, Russia
abychkov@edu.hse.ru

Abstract— **Quadratic-linearized (QL) dynamic systems have been successfully used in areas such as the Model Order Reduction (MOR) problem [1]-[2]. A byproduct of quadratic-linearization is the introduction of several auxiliary equations into the system. Since one can cast the system to QL form using different paths, the number of introduced equations for each path will be different. For the MOR problem, it is highly advisable to use a path introducing the least number of equations into the system. Nevertheless, the question of finding such a path remains open. In this article, we propose several methods and algorithms that can find a sub-optimal path to cast the system to QL form.**

Keywords—quadratic-linearization; ODE's; mathematical software; lifting transformation; MOR;

INTRODUCTION

Model Order Reduction (MOR) has been one of the rapidly developing areas of scientific interest in recent decades. The statement of the MOR problem is to bring a large dynamic system to a system with a lower dimension while capturing the main dynamics of the original system. Despite the number of successful solutions in this area, there remains a significant gap in the successes obtained for two types of systems: linear and non-linear. For linear systems, the MOR problem has developed significantly, which includes the study of such areas of knowledge as stability analysis, error estimation and structure preservation. Moreover, many solutions are computationally efficient. [3]-[5] As for non-linear systems, the success rate is not so great, which is significant, since most applications fundamentally have a non-linear nature. One of the possible solutions to the MOR problem for non-linear systems could be the linearization of this system. However, during linearization essential dynamic components of the system may be lost. For example, the bistability of the Lorenz attractor cannot be modeled by a linear system [6].

Nevertheless, in recent years, there are promising approaches for solving the MOR problem for non-linear systems. Such approaches require transforming a system to a special quadratic-linear (QL) form. This transformation is known as a lifting transformation [2]. However, in its current state, there is a significant drawback in this method. It lies in the fact that the lifting transformation algorithm is defined loosely and is suitable only for human calculation. This leads to problems such as:

1. There is no guarantee that lifting transformation is performed optimally.
2. Performing the lifting transformation is almost impossible for systems with an enormous number of equations.

In our research, we shall consider formalization of the lifting transformation algorithm and propose approaches that will help to obtain a sub-optimal lifting transformation.

The paper is organized as follows. In Literature Review QL systems and lifting transformation are considered, Methods includes our formalization of lifting transformation and its approximate algorithm, we anticipated contribution in this area in Results, Conclusion makes a summary of the paper.

LITERATURE REVIEW

Quadratic-linear systems are defined as follows:

$$\begin{gathered} \dot{x_1} = g_1(x) \\ \dots \\ \dot{x}_k = g_k(x) \end{gathered} \qquad (1)$$

where $x = (x_1, \dots, x_k)$ – state variables, $g = (g_1, \dots, g_k)$ – polynomials such that the total degree of each monomial is ≤ 2.

Define lifting transformation as the transformation of a general ODE system to a QL form. Gu [1] divides the lifting transformation algorithm into 2 parts: polynomialization and quadratic-linearization.

I. Polynomialization

Define polynomialization as the process of converting ODE system into a polynomial system.

$$\begin{gathered} \dot{x_1} = f_1(x) \\ \dots \\ \dot{x}_k = f_k(x) \end{gathered} \qquad (2)$$

An ODE system is called polynomial if each function f defines a polynomial of state variables. Polynomialization can be performed in the following ways:

A. Polynomialization by Adding Polynomial Algebraic Equations.

1. Find a non-polynomial element in the system $a_i(x)$.
2. Declare a new variable $y = a_i(x)$.
3. Replace $a_i(x)$ with y in the system.
4. Add the equation $y - a_i(x) = 0$ to the system.
5. Repeat 1-4 until the system becomes polynomial.

B. Polynomialization by Taking Lie Derivatives

1. Find a non-polynomial element in the system $a_i(x)$.
2. Declare a new variable $y = a_i(x)$.
3. Replace $a_i(x)$ with y in the system.

4. Add the equation $\dot{y} = \frac{da_i(x)}{dt}$ to the system.
5. Repeat 1-4 until the system becomes polynomial.

I. Quadratic-Linearization

Define quadratic-linearization as a process of converting a regular polynomial system into a QL polynomial system.

$$\begin{gathered}\dot{x}_1 = p_1(x) \\ \dots \\ \dot{x}_k = p_k(x)\end{gathered} \qquad (3)$$

Quadratic-Linearization can be performed in the following ways:

A. Quadratic-Linearization by Adding Polynomial Algebraic Equations.

1. Find a monomial $m_i(x)$. with total degree $\leq$ 2 in the system
2. Declare a new variable $y = m_i(x)$.
3. Replace $m_i(x)$ with y in the system.
4. Add the equation $y - m_i(x) = 0$ to the system.
5. Repeat 1-4 until the system becomes quadratic-linear.

B. Quadratic-Linearization by Taking Lie Derivatives

1. Find a monomial $m_i(x)$. with total degree $\leq$ 2 in the system
2. Declare a new variable $y = m_i(x)$.
3. Replace $m_i(x)$ with y in the system.
4. Add the equation $\dot{y} = \frac{dm_i(x)}{dt}$ to the system.
5. Repeat 1-4 until the system becomes quadratic-linear.

Consider the differences between methods A and B:

1. Method A converts the system of ordinary differential equations into a system of differential-algebraic equations (DAE) as the output. This difference is significant since not every DAE system manages to implement MOR at the current moment [2].

2. The number of introduced auxiliary equations for method A is less than for B [1].

METHODS

We propose using abstract syntax trees (AST) as a representation of mathematical expressions. This will allow us to use efficient graph algorithms for our tasks. Such a representation is used in mathematical software libraries such as SymPy and Wolfram Mathematica.

A. Polynomialization

To solve the polynomialization problem, we need to provide an algorithm for finding non-polynomial elements in an expression. We suggest using a Breadth-First Search (BFS) algorithm for AST of the equation system, starting with leaves. In this way, we can at first find atomic non-linear elements, not composite ones. This will simplify the calculation of derivatives for auxiliary equations. It can also help in the analysis of the resulting polynomial system.

B. Quadratic-Linearization

Since quadratic-linearization by taking the Lie derivative is a more general case, we will consider it a default case.

Let $G = (V; E)$ — oriented simple graph, where V is the set of vertices, E is the set of directed edges. Each vertex $v \in V$ defines a polynomial ODE system and all systems at the vertices V are equivalent to each other. Each edge $ed \in E$ represents a system transformation using a differential variable replacement. Under this notation, the problem of quadratic-linearization for the polynomial system is formulated as follows:

For a given initial vertex on the graph, find the shortest path to any vertex containing the QL system.

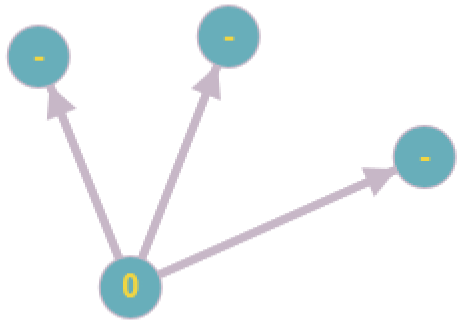

Fig 1. The initial state of G.

One of the canonical methods for solving this kind of problem for unweighted graphs is a breadth-first search (BFS) algorithm. However, we obtained an estimate of the running time of the BFS algorithm $T\ (n)\ =\ O\ (n^{n^2})$, where n — number of equations in the initial system, which makes this approach hardly applicable in practice. Such problems are called NP-hard and can be solved using approximation algorithms — efficient algorithms that find sub-optimal solutions.

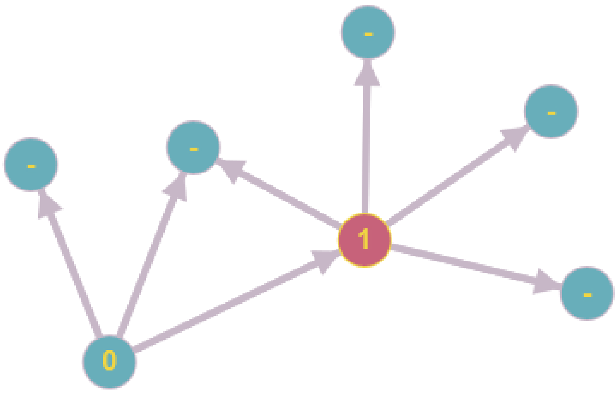

Fig 2. State of G after the first step of the algorithm.

Consider one of the approaches to the construction of approximation algorithms: the use of heuristics. In our problem, we will try to predict what properties an optimal path in a graph often has. An example of such a property is the prioritization of edges containing the square of one variable over edges representing the product of two different variables.

It is also planned to study and try to apply the following path search algorithms:

1. Iterative deepening depth-first search (ID DFS) — basic algorithm of iterative deepening family. It's possible to use it as a fast graph analysis due to the algorithm's high performance.
2. Iterative deepening A* — heuristics of this algorithm are applicable enhancing (ID DFS) performance over known vertices.
3. Lifelong Planning A* — pathfinding algorithm that can adapt to change in a graph that is suitable for our quadratic-linearization problem.

The most important difference between our problem and above pathfinding algorithms is that the location of the output node is not known in advance. Thus, if we can roughly determine the location of the output vertex, we can take advantage of these algorithms over classical pathfinding algorithms.

RESULTS

During the study, the quadratic-linearization algorithm was formulated in graph language. This allowed estimating the running time of the basic algorithm and represent the structure of our algorithm. Also, the polynomialization procedure and the quadratic-linearization procedure with several heuristics were implemented. Besides, a software application was developed to generate large ODE systems for checking algorithms of lifting transformation.

Henceforward, we plan to obtain several additional heuristics that will accelerate our algorithm. Based on the heuristics obtained, several algorithms will be proposed and estimated on benchmarks. For the best ones we will try to achieve the following properties:

1. The possibility of parallel execution on multiple CPUs. If possible, a linear speedup will be achieved.
2. Optimal solution searching speedup, using information about the found sub-optimal solutions.

CONCLUSION

In this article, we examined the problem of performing the lifting transformation of an ODE system. During the study, we proposed the formalization of the quadratic-linearization procedure using graphs. Thus, we showed that the problem is NP-hard, and it is necessary to use approximation algorithms to solve it in practice.

Also, an approach was considered that will help speed up the process of finding both sub-optimal solutions and, subsequently, optimal ones.

We hope that our work will help to improve MOR for a non-linear control system with numerous of state variables. Also, a contribution to pathfinding tasks is considered.

Further work includes the improvement of the lifting transformation algorithm. It is also possible to consider a generalization of the methods used for a wider class of models and tasks. Especially the achievements in this task can be useful in the field of robotics, which is worth considering.

REFERENCES

[1] C. Gu, "QLMOR: A Projection-Based Nonlinear Model Order Reduction Approach Using Quadratic-Linear Representation of Nonlinear Systems," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 30, no. 9, pp. 1307-1320, Sept. 2011.

[2] B. Kramer, K. E. Willcox. “Nonlinear Model Order Reduction via Lifting Transformations and Proper Orthogonal Decomposition.” AIAA Journal pp. 2297–2307, 2019.

[3] E. Grimme, “Krylov projection methods for model reduction,” Ph.D. dissertation, Dept. Electric. Eng., Univ. Illinois Urbana-Champaign, Urbana, 1997.

[4] A. Odabasioglu, M. Celik, and L. Pileggi, “PRIMA: Passive reduced order interconnect macromodelling algorithm,” in Proc. IEEE Int. Conf. Comput.-Aided Des., Nov. 1997, pp. 58–65.

[5] Z. Bai and D. Skoogh, “Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems,” Appl. Numeric. Math., vol. 43, nos. 1–2, pp. 9–44, 2002.

[6] S. H. Strogatz, Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry and Engineering. New York: Perseus Books Group, Jan. 2001.

[7] Grit Denker, Carolyn Talcott (Eds.), “Canonical abstract syntax trees“ Proceedings of WRLA 2006, Electronic Notes in Theoretical Computer Science, pp. 165-179, 2006