SENIOR ENGINEERING HONORS

Fliess Series Method for Ricatti Differential Systems

Brady Simon

Faculty Advisor: Dr. Thitsa

Abstract

We propose an algebraic combinatoric approach to solve matrix Riccati differential equations. Functional series solutions for matrix Riccati differential equation are computed by formal power series method using Fliess operators. Finite escape time of the Riccati differential system is determined from such series solutions. Sufficient conditions for the nonexistence of finite escape time are also presented. Results are compared with solutions obtained by numerical integration and closed form solutions when they exist.

Algorithm Results

This algorithm was created to replace an older version that run significantly slower. This algorithm is faster than the former version while also being more dynamic and allowing for a larger range of values. The range of the algorithm is theoretically infinite, but there are realistic restraints on computing power, time, and computer memory. A user interface has been generated for the algorithm, but needs improvement.

Project Overview

Many feedback control design problems involve finding stabilizing feedback control laws by numerically solving unstable matrix differential equations with time varying coefficients; prominent examples include linear-quadratic regular optimal control problems and optimal filtering. This task is nontrivial and the numerical approach can fail especially when one deals with stiff differential equations. In the case of periodic matrix Riccati equations, Semidefinite Programming (SDP)-based approach was proposed recently. Another issue often encountered in dealing with unstable differential equations is the finite escape time. Determining the finite escape time of a given nonlinear system and a given input is an important and difficult problem. In many man-made and naturally occurring unstable systems, it is important to know if and when the solution escapes to infinity. An algorithm was developed to determine these escape time coefficients.

Algorithm Overview

The algorithm is broken into different sections to solve smaller parts of the equation to make the problem easier to solve. Particularly, the program relies on three classes: NSolver, KSolver, and CSolver. Each of these algorithms solves a smaller part of the problem and builds upon each other, eventually reaching the full word representation (contained in CSolver). Since the full description of a word can never be fully determined, the coefficients are efficiently-calculated using linear approximation instead.

NSolver
Solves for the set of n values.

→

KSolver
Solves for the K values using the set of n values.

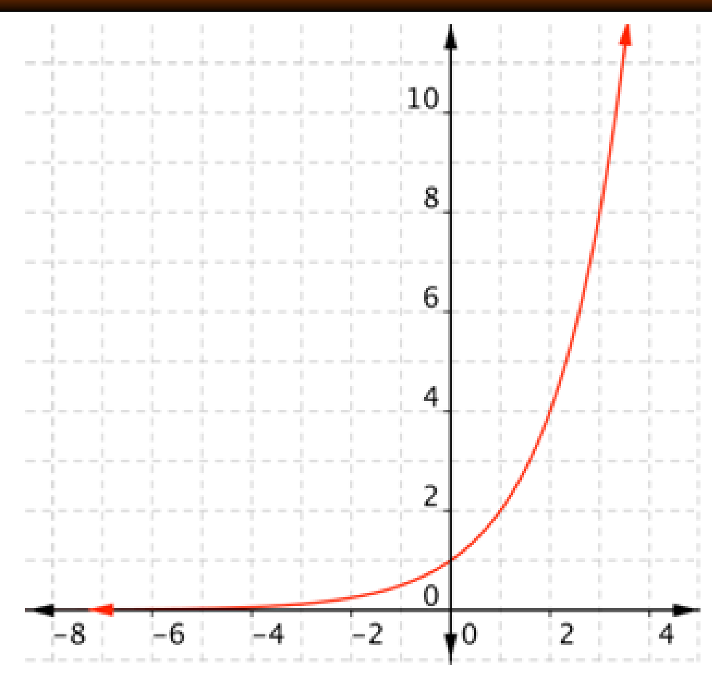
→

CSolver
Solves for the coefficients using both n and K values

Efficiency

The algorithm tends to get exponentially more expensive to use as the word size and accuracy increases. Therefore, it is important that the user consider the accuracy needed so that the system is not overloaded with calculations. While some efficiencies have been applied, the algorithm has the potential to take significant time.

Exponential Algorithm



Future Work

While this algorithm calculates coefficient values efficiently, there is still more to be done. In the future, creating a programming language to represent this mathematical process would be an asset to those who need coefficient values. This would allow for faster iterations of research, which would allow the user to save time and learn more. This future language could include other formal language concepts, such as shuffling, to create a more in-depth program.

This program is also written in Java, but the User Interface (UI) is lacking. A more robust UI would make it more useful for a wider range of researchers.

Efficiency gains could also be made by rewriting a portion of the algorithm that is recusive, which puts a bottleneck on the application based on the computer's memory. An iterative process would be more efficient for these calculations.

References

M. Fliess, M. Lamnabhi, F. Lamnabhi-Lagarrigue, An algebraic approach to nonlinear functional expansions IEEE Trans. Circuits Systems, vol. 30, pp. 554-570, 1983.

Thitsa, Makhin & Simon, Brady & Martin, Clyde. (2018). Fliess Series Method for Riccati Differential Systems. 2944-2949. 10.1109/CDC.2018.8619595.

Acknowledgements

Thanks to Dr. Thitsa for introducing me to research and giving me guidance.

Lockheed Martin is a generous financial supporter of the Mercer University Engineering Scholars Program.