Generation of DIRK Methods

Math 6321, Fall 2012

Introduction

Within the realm of diagonally implicit Runge Kutta (DIRK) methods for solution of the ODE system,

\[ y' = f(t, y), \quad y \in \mathbb{R}^n, \quad (1) \]

almost all methods are chosen so that the diagonal entries of the Butcher table are all identical. The reasoning for this is rather simple: any \( s \)-stage DIRK method for solution of the time step \( t_n \to t_{n+1} \) may be written as

\[
z_i - h \gamma f(t_n + c_i h, z_i) = y_n + \sum_{j=1}^{i-1} A_{i,j} f(t_n + c_j h, z_j), \quad i = 1, \ldots, s,
\]

\[
y_{n+1} = y_n + \sum_{i=1}^{s} b_i f(t_n + c_i h, z_i),
\]

where \( h = t_{n+1} - t_n \) and \( \gamma \) corresponds to the value of the diagonal entries in the Butcher table. For linear ODE systems, where the linearity does not depend on \( t \),

\[ f(t_n + c_i h, z_i) \equiv J z_i, \quad J \in \mathbb{R}^{n \times n}, \]

so the above DIRK method becomes

\[
z_i - h \gamma J z_i = y_n + \sum_{j=1}^{i-1} A_{i,j} f(t_n + c_j h, z_j), \quad i = 1, \ldots, s,
\]

\[
y_{n+1} = y_n + \sum_{i=1}^{s} b_i f(t_n + c_i h, z_i).
\]

Hence in this scenario, each stage solution \( z_i \) may be found as the solution to a linear system with the same matrix, \( M \equiv I - h \gamma J \). When \( n \) is not too large, the method of choice for solving these linear systems is to perform an \( LU \) factorization of \( M \) once, and to reuse this factorization for each stage solve. Typical ODE solvers also leverage this \( LU \) factorization cost by striving to keep the time step size \( h \) fixed for as long as possible within the simulation.

However, most ODE systems are not in fact linear, and they often have time dependent right-hand-side functions \( f(t, y) \). Similarly, for very large ODE systems (\( n \gg 1 \)), dense linear algebra is intractable and must be replaced with iterative linear solvers such as multigrid methods or Krylov subspace methods. For these types of problems, having a fixed diagonal \( \gamma \) within the Butcher matrix provides little benefit. Moreover, choosing \textit{a priori} that this value remain fixed when deriving the method, one loses \( s - 1 \) degrees of freedom within the derivation of DIRK method coefficients, that could instead be used to create a more powerful method.

Project
The goal of this project is to construct a code in either Maple or Mathematica to derive new s-stage DIRK methods that do not have a fixed diagonal $\gamma$. With these additional degrees of freedom, you are free to pursue other goals for the DIRK method, including higher-order accuracy or increased stability. While methods that include embeddings are preferrable, they are not required.

Once you have constructed this code, you will use it to generate coefficients for DIRK methods having 2, 3 and 4 stages (5 would be fantastic). These should be inserted into the Matlab Runge-Kutta solver framework \texttt{RKLab} [1], and tested on the three built-in example problems therein. All development of both your Matlab and Maple/Mathematica code should be done within your own personal “fork” of the \texttt{RKLab} repository, following the instructions to do so on the course \texttt{projects} web page.

\textbf{Related Ideas}

A thorough discussion on the use of \textit{Butcher 1-trees} for the derivation of ERK methods is given in chapter II.2 of [2], which is extended to IRK and DIRK methods in chapters IV.5 and IV.6 of [3].

\textbf{References}


