Numerical Algorithm for First Order Non-Homogeneous Time Varying Coefficient Coupled set of Differential Equations

Sai Nikhil.T
Department of Civil Engineering
Indian Institute of Technology
Kharagpur, WestBengal 721302, India

December 3, 2012

Abstract

A standard analytic technique/algorithm for finding a solution to the set of coupled differential equations with variable coefficients is not known till date. A Numerical Algorithm is presented. As special cases, the solutions of nonhomogeneous and homogeneous linear differenctial equations of order N with variable coefficients are obtained.

Introduction

For the Single equation of the form,

$$y'(x) = A(x).y(x) + B(x)$$

the general solution is of the form,

$$y(x) = c.e^{\int A(x).dx} + \int B(x).e^{-\int A(x).dx}.dx$$

A standard set of coupled differential equations can be represented as :

$$y_1'(x) = f_1(x, y_1, y_2, ...y_n)$$

$$y'_2(x) = f_2(x, y_1, y_2, ...y_n)$$
$$...$$
$$...$$
$$y'_n(x) = f_n(x, y_1, y_2, ...y_n)$$

Which can further be represented in matrix form as :

$$\begin{pmatrix} y'_1 \\ y'_2 \\ . \\ . \\ y'_n \end{pmatrix} = \begin{pmatrix} A_{11}(x) & A_{12}(x) & . & . & A_{1n}(x) \\ A_{21}(x) & A_{22}(x) & . & . & A_{2n}(x) \\ . & . & . & . & . \\ . & . & . & . & . \\ A_{n1}(x) & A_{n2}(x) & . & . & A_{nn}(x) \end{pmatrix} \times \begin{pmatrix} y_1 \\ y_2 \\ . \\ . \\ y_n \end{pmatrix} + \begin{pmatrix} B_1(x) \\ B_2(x) \\ . \\ . \\ B_n(x) \end{pmatrix}$$

i.e.,

$$Y' = A(x).Y + B(x)$$

.

Solution

For a set of n coupled differential equations, with constant coefficients, the matrix A contains constant co-efficients. The solution is of the form, similar to one dimensional problem, which is presented below :

$$Y(x) = e^{Ax}.C + \int e^{-Ax}.B(x).dx$$

where, e^A is the Matrix Exponential of Matrix A.

Matrix Exponential of a Matrix A is calculated by Diagonalizing Matrix A and by using the special property of Matrix Exponential for a Diagonal Matrix, as shown below:

$$A = P^{-1}DP$$
$$e^A = P^{-1}e^D P$$

However, if the coefficients are varying functions of X, then a standard analytic solution hasn't been known till date, where as a heuristic case solution exists when the matrix A satisfies the following property :

$$A(s).A(t) = A(t).A(s)$$

i.e., if matrix is self-symmetric, the solution is of the form,

$$Y(x) = e^{\int A(x).dx}.C + \int e^{-\int A(x)dx}.B(x).dx$$

The Numerical Algorithms available for implementation in solving One Dimensional problem are,

$$Euler's Method (Error : O(h))$$

$$Taylor's Method (Error : O(h^n))$$

$$RungeKutta's 4^{th} Order Method (Error : O(h^4))$$

$$Finite - Difference Method (Error : O(h^2))$$

where, $h = \frac{b-a}{n}$, is the length of the partition interval.
Of all the above methods, Runge Kutta's 4^{th} order method is more practical and has been implemented in this article.

Implementation

For, 1-Dimensional case, the implementation goes like this:

$$Runge[a0, b0, \alpha, m0] :=$$
$$Module[a = a0, b = b0, j, m = m0,$$
$$h = (b - a)/m;$$
$$Y = T = Table[0, m + 1];$$
$$T[[1]] = a;$$
$$T[[1]] = \alpha;$$
$$For[j = 1, j <= m, j + +,$$
$$k_1 = hf[T[[j]], Y[[j]]];$$
$$k_2 = hf[T[[j]] + h/2, Y[[j]] + k_1/2];$$
$$k_3 = hf[T[[j]] + h/2, Y[[j]] + k_2/2];$$
$$k_4 = hf[T[[j]] + h, Y[[j]]] + k_3];$$
$$Y[[j + 1]] = Y, [[j]] + 1/6(k_1 + 2k_2 + 2k_3 + k_4);$$
$$T[[j + 1]] = a + hj;];$$
$$Return[Transpose[T, Y]]]$$

where as, for an N-Dimensional case, all the equations are coupled and so, the values of k in above implementation are coupled. Hence, a slight modification of the above implementation gives us the following implementation:

$Runge[a0_, b0_, a_, m0_] :=$

$Module[a = a0, b = b0, i, j, m = m0,$

$h = (b - a)/m;$

$l = Length[a];$

$T = Table[0, \{m + 1\}];$

$T[[1]] = a;$

$Y = Table[0, m + 1, l];$

$k = Table[0, 4, l];$

$For[j = 1, j \leq l, j + +, Y[[1]][[j]] = a[[j]];];$

$For[i = 1, i \leq m, i + +,$

$yList4k1 = \{\};$

$For[j = 1, j \leq l, j + +, yList4k1 = Append[yList4k1, Y[[i]][[j]]];];$

$For[j = 1, j \leq l, j + +, k[[1]][[j]] = h * f[T[[i]], yList4k1][j];];$

$yList4k2 = \{\};$

$For[j = 1, j \leq l, j + +, yList4k2 = Append[yList4k2, Y[[i]][[j]] + k[[1]][[j]]/2];];$

$For[j = 1, j \leq l, j + +, k[[2]][[j]] = h * f[T[[i]], yList4k2][j];];$

$yList4k3 = \{\};$

$For[j = 1, j \leq l, j + +, yList4k3 = Append[yList4k3, Y[[i]][[j]] + k[[2]][[j]]/2];];$

$For[j = 1, j \leq l, j + +, k[[3]][[j]] = h * f[T[[i]], yList4k3][j];];$

$yList4k4 = \{\};$

$For[j = 1, j \leq l, j + +, yList4k4 = Append[yList4k4, Y[[i]][[j]] + k[[3]][[j]]];];$

$For[j = 1, j \leq l, j + +, k[[4]][[j]] = h * f[T[[i]], yList4k4][j];];$

$For[j = 1, j \leq l, j + +, Y[[i + 1]][[j]] = Y[[i]][[j]] + 1/6 * (k[[1]][[j]]$

$+ 2 * k[[2]][[j]] + 2 * k[[3]][[j]] + k[[4]][[j]]);];$

$T[[i + 1]] = a + h * i;];$

$Y = Prepend[Transpose[Y], T];$

$Return[Y];];$

Run Time Analysis

For a 1-Dimensional case,

$$
\begin{aligned}
Running\ \ Time &= Sum\ \ of\ \ running\ \ times\ \ of\ \ each\ \ individual\ \ step \\
&= c_1 + c_2(m+1) + c_3 + c_4 + c_5(m) + c_6 \\
&= O(1) + O(m+1) + O(1) + O(1) + O(m) + O(1) \\
&= O(m)
\end{aligned}
$$

For an N-Dimensional case,

$$
\begin{aligned}
&Running\ \ Time = Sum\ \ of\ \ running\ \ times\ \ of\ \ each\ \ individual\ \ step \\
&\qquad = c_1 + c_2 + c_3(m+1) + c_4 + c_5 \times (m+1) \times l + c_6 \times 4 \times (l) + c_7 \times l + c_8 \times l \times m \\
&\qquad = O(1) + O(1) + O(m+1) + O(1) + O((m+1) \times l) + O(4 \times l) + O(1) + O(l \times m) \\
&\qquad = O(m \times l)
\end{aligned}
$$

Results

Following is a test case verified over the set of Coupled Differential Equations :

$$
\begin{aligned}
&Map[f[t,y_]], 1, 2];\\
&f[t,y][1] := y[[1]] + y[[2]];\\
&f[t,y][2] := \frac{Cos[t] + Sin[t]}{2 - Cos[t] + Sin[t]} \times y[[2]];\\
&n = 100;\\
&ptsList = Runge[0, 1.0, 1, 10, n];\\
&pts1 = Transpose[List[ptsList[[1]], ptsList[[2]]]];\\
&npts1 = N[pts1];\\
&pts2 = Transpose[List[ptsList[[1]], ptsList[[3]]]];\\
&npts2 = N[pts2];\\
&Needs[\text{"}Graphics\text{`"}];\\
&graph1 = ListPlot[npts1, PlotJoined- > True];\\
&graph2 = ListPlot[npts2, PlotJoined- > True];\\
&Show[graph1]\\
&Show[graph2]
\end{aligned}
$$

The graphs obtained are as shown below :

Conclusions

In many real-world situations such as, Earthquake Engg. (with variable mass and spring constants, movement of ground as a function of time), Environmental Engg.(variable rate constants in chemical reactions), Electrical Engg. (in many current and voltage regulation models involving RLC circuits), etc. there are a lot of applications of this Numerical Method of solving Differential Equations.

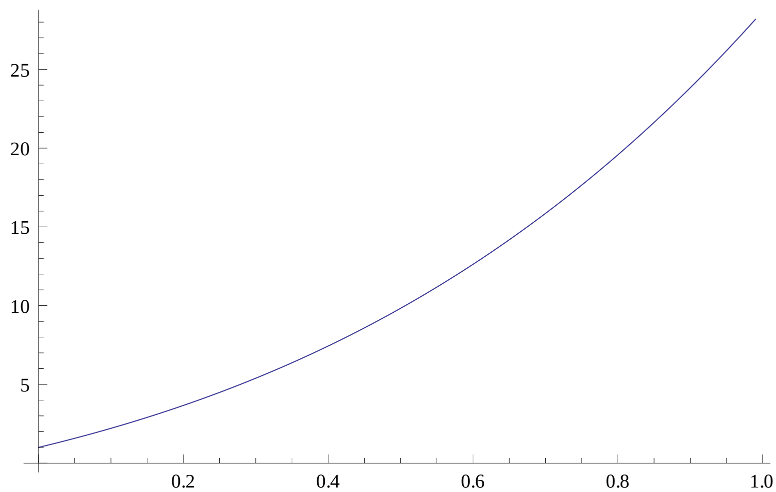

Figure 1: Plot of y_1 vs x.

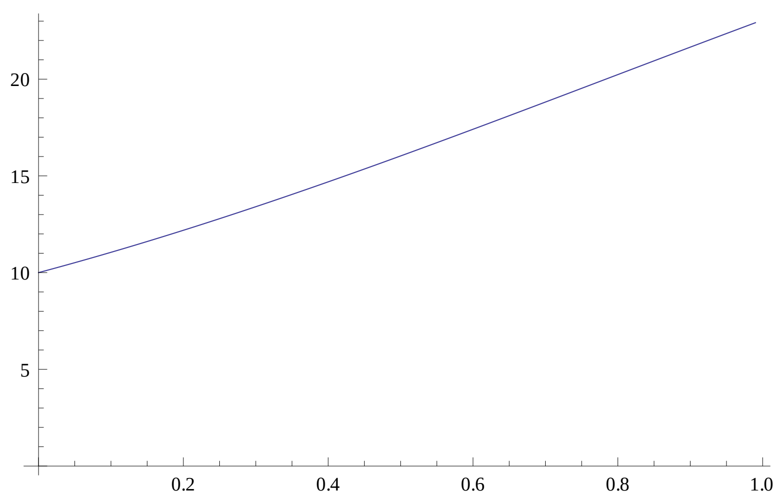

Figure 2: Plot of y_2 vs x.

References

[1] Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later, Cleve Moler, Charles Van Loan.

[2] Modules for Numerical Methods using Mathematica, Mathematics Department Website, California State University - Fullerton.