

Webnucleo Technical Report: Newton Tool

Allen Parker

February 22, 2006

The following is a brief explanation of the IDL code at the base of the Newton Tool. This report describes the algorithm used and the calculations performed in IDL and a bit about its applications and limits. The tool itself and tutorials explaining its use can be found at:

http://nucleo.ces.clemson.edu/home/online_tools/newton/0.2

1 Forth-Order Runge-Kutta

At the root of the IDL procedure, `newton_nd.pro`, is the fourth order Runge-Kutta function, `rk4`, which is native to IDL. This function accepts an array, `y`, of `n` variables and `n` first derivatives of those variables in the form:

$$\left[q_1 \quad q_2 \quad \cdots \quad q_n \quad \dot{q}_1 \quad \dot{q}_2 \quad \cdots \quad \dot{q}_n \right] \quad (1)$$

The `rk4` function then propagates this array forward one small step in the dependent coordinate using a set of `n` differential equations supplied by the user. Since the differential equations supplied are second order, the Newton Tool breaks each equation into two first order differential equations. Time is the dependent variable for this tool. So, the first order differential equations are:

$$\frac{dq_i}{dt} = \dot{q}_i \quad (2)$$

$$\frac{d\dot{q}_i}{dt} = \ddot{q}_i \quad (3)$$

For the Newton Tool, the coordinates are the variables `x`, `y`, `z`, and the first derivatives are with respect to time. These variables are named after the classic Cartesian variables but can represent any generalized coordinates.

2 Adaptive Time Step

In the Newton Tools's main loop, another IDL function is called that can change the step size. This is done using the equation for the fifth order truncation error:

$$(\Delta t)_0 = (\Delta t)_1 \left| \frac{(\Delta x)_0}{(\Delta x)_1} \right|^{\frac{1}{2}} \quad (4)$$

The IDL function, `stepfunc.pro`, uses this equation to scale the step size for the next iteration. In equation (4), $(\Delta x)_1$ represents the difference between the change in the variable made with one time step of size $(\Delta t)_1$ and two iterations with a step size of $(\Delta t)_1/2$. The variable $(\Delta x)_0$ is the desired accuracy of the iterations. The ratio of these two differences scales the next time step. This scaling prevents the code from going through too many iterations for a simple problem, or not accurately propagating the motion of a system that requires high accuracy.

3 Online Tool (Front End)

The online Newton Tool is a front end written around the IDL procedure. The tool obtains the necessary initial parameters from the user to be entered into the IDL function. The parameters that are needed are the differential equations (accelerations), initial conditions, and the parameter for the adaptive step.

In order to be user friendly, the strings are taken in terms of Cartesian variables, their associated velocities, and time. These, more explicitly, are; x , y , z , v_x , v_y , v_z , and t . These equations are then checked with a function called ‘`check_string`’ that replaces these variables with the variable names as used in the `newton.nd` procedure shown in equation (1).

If the necessary information has been entered, the ‘Calculate’ button located on the ‘Initial Conditions’ Panel calls the IDL procedure to propagate the system. The resulting data is written to a temporary file which can then be read by the ‘Plot Tool’ and ‘Table Tool’. The ‘Plot Tool’ allows the user to make a two dimensional plot of any of the parameters propagated through the code. The ‘Table Tool’ lets the user make a table of the parameters in either an ASCII or html format.

4 Limitations

The current limitations of the Newton Module mostly involve scaling problems. The success of a given simulation is highly dependent upon how accurate each time step is. Equations or initial conditions that produce numbers very large or very small may cause problems in the `rk4` function called. To address this issue, the Tool’s Tutorials contain tips on assuring appropriately scaled numbers and equations are entered.

Another limitation is that, because the tool is written for a general set of equations, it does not check to make sure the equations are not evaluated with a zero in the denominator. This may lead to the Newton Tool crashing and failing to return the motion because of undefined point it tried to evaluate. The Tool’s Tutorials contains information regarding this limitation and suggestions to user with how to deal with it.

Lastly, the user may find computation time to be a limitation as well. Because the calculations are performed on the webnucleo server, an individual calculation is limited to run within a time interval of 3 minutes.

If more lengthy computations are required, the IDL source has been made available online under the Tool's Technical Resources. The interested user may download this source code to run longer calculations and to modify the source for particular applications.