ISSUE 60

APOGEE PEAK OF FLIGHT NEWSLETTER

Smarter Guessing and Simulating with RockSim 5

Written by Ken Karbon

Introduction

I don't like haphazard guesses and trial and error. I like efficient and systematic approaches to problem solving. RockSim is a very elaborate design and simulation tool, but the software doesn't do everything for you automatically. For more advanced operations, such as finding maximum liftoff mass, optimum launch rod angle, or backcalculating a drag coefficient, you still need to plug and chug the parameters and rerun the simulations until you are satisfied with the outcome. This is the brute force method. In this article, I will describe a simple routine that can be used to supplement RockSim and minimize the guesswork and calculations that you and your computer need to do.

Method

The three examples above, like many problems in engineering, require the solution of a nonlinear equation. Given a continuous nonlinear function, you want to know the value of x that produces a desired value of y. When the desired value is y = 0, this is commonly called "finding the root" of the equation. In our cases, we don't even know the function, since RockSim is a numerical solver. However, we can consider the software itself to be a single variable "function", given that you plug in an x value (such as drag coefficient) and get a y value (such as maximum altitude) in return.

Several numerical schemes are used to solve nonlinear problems. One of the most powerful and efficient is the secant method. Two initial guesses are used to create a straight line (a secant) through two points on the nonlinear curve. The solution of the straight line is then taken as the next guess to the nonlinear function. This procedure is applied repetitively until the routine converges on the desired solution within some specified criteria. If more than one root or local minima exist in the function, the secant method can oscillate or converge slowly. Therefore, it is good practice to pick the two initial guesses close to a probable solution. Consult a college-level text on numerical methods for more details. For this article, I created a simple spreadsheet to perform the secant method. Let's apply it to a couple problems.

Click Here to launch the Excel spreadsheet template.

Example 1. Maximum Lift Off Mass

In E-Zine #34, Tim discusses an iterative approach to finding maximum lift-off mass:

"It is the brute force computer simulation method. Basically, you perform the computer simulation and see if the rocket is still going up when the ejection charge fires. If it the rocket is still traveling upward you add some weight and repeat the simulation. This continues until the rocket deploys at or within one second of apogee. It can take numerous simulations if you didn't pick a good starting point. But eventually, by running enough simulations, you'll get an answer."

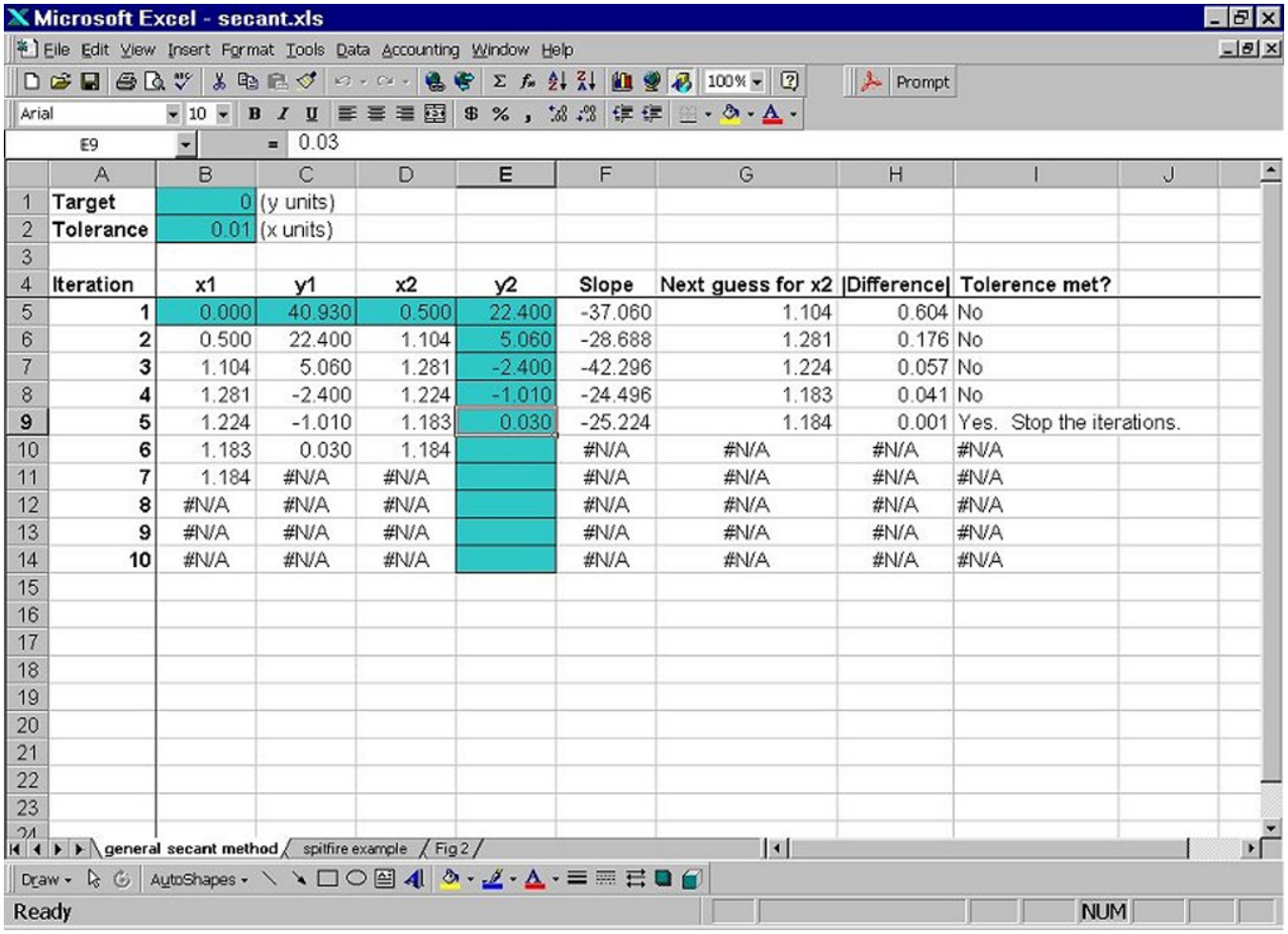
The secant method is a good choice for Tim's procedure because it quickly zeros in on the correct mass and minimizes the number of guesses and simulations.

As an example, I chose the spitfire.rkt model that came with RockSim. The engine is a G125-5. I added a mass object to use as the variable. I want to calculate the mass of the object (x value) that will produce zero velocity at deployment (desired y value). This means the rocket safely ejects at apogee and the resulting mass object, when added to the sustainer mass, gives an indication of the maximum allowable lift-off weight of the rocket.

To use the spreadsheet, fill in only the shaded areas as needed (Figure 1).

Enter "0.0" in the "Target" field as our goal of zero velocity at deployment. Let's use a tolerance of 0.01 kg as a convergence criterion to tell us when the new answer is not signifi-

APOGEE COMPONENTS
1130 Elkton Drive, Suite A
Colorado Springs, CO 80907 USA
www.ApogeeRockets.com
orders @ApogeeRockets.com
phone 719-535-9335 fax 719-534-9050



	A	B	C	D	E	F	G	H	I	J
1	Target	0	(y units)							
2	Tolerance	0.01	(x units)							
3										
4	Iteration	x1	y1	x2	y2	Slope	Next guess for x2	|Difference|	Tolerence met?	
5	1	0.000	40.930	0.500	22.400	-37.060	1.104	0.604	No	
6	2	0.500	22.400	1.104	5.060	-28.688	1.281	0.176	No	
7	3	1.104	5.060	1.281	-2.400	-42.296	1.224	0.057	No	
8	4	1.281	-2.400	1.224	-1.010	-24.496	1.183	0.041	No	
9	5	1.224	-1.010	1.183	0.030	-25.224	1.184	0.001	Yes. Stop the iterations.	
10	6	1.183	0.030	1.184		#N/A	#N/A	#N/A	#N/A	
11	7	1.184	#N/A	#N/A		#N/A	#N/A	#N/A	#N/A	
12	8	#N/A	#N/A	#N/A		#N/A	#N/A	#N/A	#N/A	
13	9	#N/A	#N/A	#N/A		#N/A	#N/A	#N/A	#N/A	
14	10	#N/A	#N/A	#N/A		#N/A	#N/A	#N/A	#N/A	

Figure 1

cantly changing, and thus the iterations are complete. Enter "0.01" in the "Tolerance" field. Now, for iteration number 1, enter two initial guesses and results. I choose "x1" to be 0.0 kg of mass object. After performing the RockSim simulation, I get a "y1" value of +40.93 m/s in the "velocity at deployment" summary column. Similarly, I'll choose 0.5 kg for "x2" and get a simulated value of +22.4 m/s for "y2". The velocity is still greater than zero, but it's getting smaller. The spreadsheet calculates what the next guess should be, in this case 1.104 kg. Continuing in this fashion gives a final value of 1.183 kg of mass object that meets my criteria in just 5 iterations (6 simulations), and the deployment velocity is near zero.

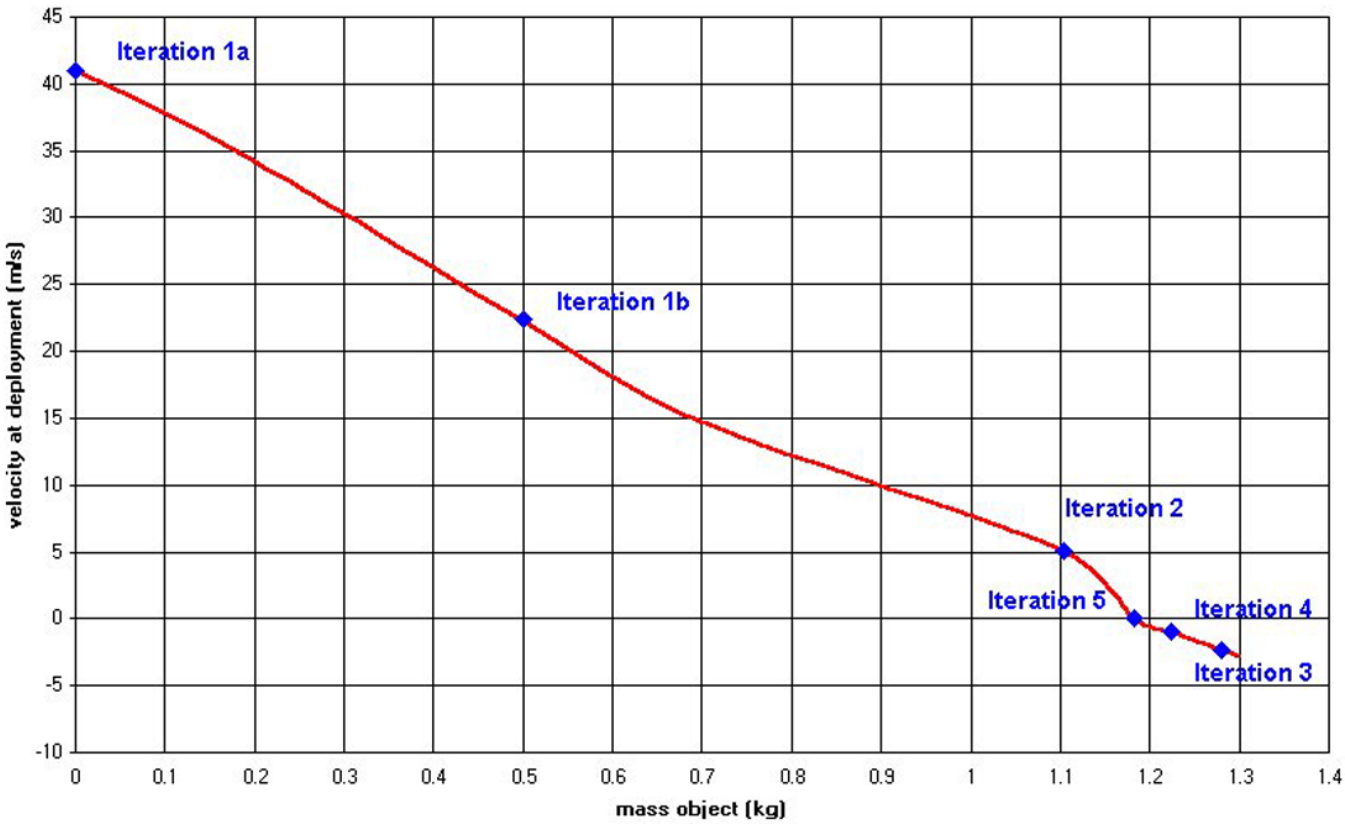
Using the correct sign of velocity is critical! To find a zero, the secant method must distinguish between positive velocity (going up) and negative velocity (going down). As far as I can tell, RockSim only reports the velocity magnitude in the simulation summary. You must plot the results to decipher if the velocity at deployment is positive or negative.

The secant method is so efficient, many problems can be solved in 10 iterations or less given reasonable starting points. Before I used the spreadsheet, it took me 8 brute force guesses just to get the velocity at deployment down to 3 m/s. The secant method got within a fraction of that in just 6 simulations. Figure 2 shows the nonlinear relationship of our problem and the convergence of the solution.

Example 2. Backcalculating Drag Coefficient

Determining the drag coefficient on a rocket is still one of the major assumptions when performing flight simulations, especially for rockets with non-typical geometry, pods, or add-ons. One way to get an average Cd for a given flight is to measure the apogee (via timing, optical tracking, or altimeter) and work backwards in the simulation to iterate on a fixed

Fig 2. Spitfire Example



drag coefficient that produces the observed altitude or time to apogee. The secant method can again be used to minimize the iterations.

For example, my scratch-built rocket recorded a maximum altitude of 720 m with an altimeter. This is my target. I want to determine a fixed Cd to within 0.001 to achieve this flight. In RockSim, I uncheck "Calculate Cd at simulation time" and use a fixed Cd of 0.6 in the rocket design information panel. Running the simulation gives me an altitude of 795.10 m. My second guess is Cd=0.7 for an altitude of 749.40 m. Using the spreadsheet returns a final Cd of 0.774 that meets the tolerance in three iterations, and the altitude is within a meter of the target value. This is the average Cd needed to produce this particular flight. Instead of using drag coefficient as the convergence criterion (the x value), change in altitude (the y value) from iteration to iteration may also be used. For added accuracy, you can specify tolerances on both x and y.

Summary

This article briefly describes a way to efficiently iterate rocket simulations using the secant method for solving non-linear functions. This can save you the frustration of brute force guessing. Here I used a spreadsheet to complement RockSim, but in my own simulation programs, I wrote this simple looping routine right into the code.