How External Ballistics Programs Work

By Bryan Litz

There are several possible methods for calculating the flight path and other properties of a ballistic trajectory like: velocity, time of flight and wind drift. Some programs calculate more things than others. Some have the potential to be inherently more accurate than others. In this article, I'll try to explain the features of each method. In general, there are two classes of solution methods:

1. Closed form, analytic solutions, and
2. Time stepping, or numeric solutions.

The Siacci and Pejsa methods are both analytic solutions. Point mass and 6-DOF are both numerical solutions. I'll go into a little detail on the facts, as well as some of my opinions regarding these methods:

Closed Form Methods:

The Siacci method is a fine analytic solution for flat fire ballistics. It's strengths are run time and ease of use. Its major drawback is accuracy potential, especially because it usually uses BC's referenced to the G1 standard. When referencing BC's to a standard, you will always compromise some accuracy, depending on how well the drag profile of your projectile matches the standard. By using the G1 standard for long range bullets, you compromise quite a bit. Why use G1 then? Well, some bullet makers will tell you that it's because G1 is actually the best fit. I find it more than somewhat suspicious that using the G1 standard also allows them to advertise the highest BC's, but I digress. Piecewise defining BC as a function of velocity can help, but your still making a potential compromise.

Bottom line for Siacci: If you use a standard drag function that closely matches your projectile (like the G7 standard for long range boat tailed bullets) you have a decent shot at getting good numbers. Unfortunately, the G1 function is usually used, which is not a good standard for typical long range bullets.

Pejsa's method is very similar to Siacci, except that the program doesn't use tables generated by standard shaped projectiles. You're effectively using an analytic function (of Mach number) that you tailor to fit your specific projectile. In this way, you can have an exact solution, provided you can come up with an analytic function of Mach number that fits your projectiles drag profile (this isn't easy, and requires careful testing by the shooter for each bullet he wants to model). Chapter 5 of Modern Exterior Ballistics gives solutions for cd as a function of Mach (cd = const, cd = 1/M, cd = 1/M^(1/2)). Pejsa's method is very similar, except it allows cd = 1/M^(x) where x is some variable that depends on your specific projectile.

Bottom line on Pejsa's method: Very similar to Siacci, only it allows a more custom solution for a specific projectile. Using the Pejsa method, the burden of modeling velocity dependant drag is on the shooter instead of in the program where it belongs.

Numeric Solutions:

The point mass solution is the simplest numerical solution. It accounts for the x y and z travel of the projectile, but doesn't solve for any rotations. Whereas closed form

solution methods are restrictive on how the drag profile is characterized, a point mass model lets you use a table of cd vs Mach that can be any shape. The numerical solution is a time stepping method. This means that at each time step (about every 0.001 seconds), the computer program indexes into the cd table to find exactly what value to use for that specific instant, and solves all of the equations of motion for the period of time leading to the next time step. Now if you had good representations of your cd function in the closed for method, you won't gain much (if any) accuracy with a point mass model, but you will experience significant increases in computer run time. One thing that a point mass model does that an analytic solution can't is predict trajectories that are not flat fire. You see, assumptions are made in order to make the closed form solutions solvable.

These include simplifications to the geometry (assume most of the velocity is in the forward direction, and not up & down.) Also, a constant value is assumed for air density (constant for each trajectory, but can be different for different trajectories). The constant air density assumption is ok unless you traverse large amounts of altitude as in artillery trajectories. With such high angles of fire, you need some way to account for the fact that the air density changes throughout the trajectory. Point mass can do this well because it looks up air density in a table as it 'time steps' thru the trajectory, just like it looks up cd as it time steps. Equations of motion are solved with just the right value of air density for each 0.001 second time increment.

Bottom line for point mass: Marginal improvement over analytic solution for flat fire, small arms ballistics, and is required for high angle of fire (> 10 degrees).

The 6 degree of freedom (6-DOF) is the most inclusive of all ballistics programs. This type of simulation accounts for rotation on all 3 axis, as well as translation in all 3 directions. There are significant downsides to using a 6 DOF simulation. For one, the run time is significant. On modern computers, they're usually about 100:1 slow (meaning 100 seconds of run time for 1 second of bullet flight). This can be sped up by compiling the code, and by other various computer science efficiency tricks. But it will never run long trajectories instantaneously like closed form methods can.

Secondly, you need large amounts of aerodynamic data that is hard to get. In addition to cd, you need similar tables for: lift coefficient, moment coefficient, magnus force, magnus moment, roll damping, pitch damping, etc. There are literally dozens of various aerodynamic coefficients that can be used. One of the challenges of using a 6 DOF simulation is knowing which aero coefficients are significant enough to include, and which ones aren't. The decision to keep or discard various aero coefficients depends on what type of projectile you're modeling, and what flight regime your modeling it in.

When used properly, 6 DOF modeling is the most inclusive representation of a projectiles motion thru space. If you have good aerodynamics, you can model pitching and yawing motion, be able to predict transonic dynamic stability, and many other investigative endeavors. Of particular interest to the long range hunter is the ability to predict gyroscopic drift, Coreolis effects, and aerodynamic jump due to crosswind. You can also do sensitivity analysis on dispersion due to various factors like in-bore yaw, muzzle blast effects, etc.

Bottom line for 6-DOF simulations: Definitely overkill for the average shooter. Most people don't have access to aerodynamic prediction methods required to

generate the necessary data tables. Even when predictions are available, it's hard to get some of the coefficients accurate.

1. Required for those interested in every possible detail of a trajectory.
2. 6-DOF trends can be modeled and appended to analytic solutions to sort of correct for spin drift, ect.
3. Most effective application is in design, and academic investigations

A major point concerning all types on ballistic prediction methods:

More important than the type of method used is the accuracy of the inputs.

Major inputs being: characterization of drag, atmospheric properties, and muzzle velocity.

That being said, one cannot argue that one method is more accurate than another, and then use inaccurate BC. There is usually more error in the calculated drag of a bullet (correlating to BC) than in any of the solution methods. For example, the potential benefits of Pejsa's method can only be realized if you actually test fire the projectiles, and customize the drag function according to test results.

Likewise, unless you're measuring temperature, pressure, and humidity and calculating density for each trajectory, it doesn't matter what method you use.

If I were to write a ballistics program for small arms that's intended to surpass the existing available packages, I would use a point mass solver (3-DOF numeric solver) for the following reasons:

1. Modern computers, even field deployable devices like palm pilots and cell phones have fast enough processors these days to solve a numerical solution in a reasonable amount of run time.
2. The program does not require you to store large tables (S, T and V functions) like the Siacci method.
3. You can make use of multiple standards (G1, G7, etc) depending on whichever one is best suited to the bullet you're modeling. (Siacci also has this feature).
4. If you have access to a 6-DOF simulation, you can investigate trends like gyroscopic drift as a function of flight time for certain classes of projectiles, and then apply the trends as corrections to the point mass solution (Ref article: Extending Max Effective Range of Small Arms on this website). Applying the 6-DOF corrections won't significantly affect computer run time.

I would avoid the Pejsa solution because of the difficulty of modeling bullet drag. Since the Pejsa method does not make use of any standard projectile drag curves, it's up to the user to describe the drag of his own bullets. This requires establishing obscure coefficients and exponents for each bullet for several velocity bands. Large compromises are made when the projectile slows to transonic speeds and the drag curve is approximated with linear segments. The complexity of Mach dependant projectile drag belongs in the solution method, it should not be up to the shooter to figure out.

Well, there you have it. More than any sane person cares to know about how ballistics programs work. In the end, ballistic programs are just another tool. Like all tools, some have more inherent potential than others.

Ultimately, the actual effectiveness of the tool depends mostly on how it's used (like feeding it accurate inputs).

There's a phrase we use in the modeling and simulation business: 'Garbage in, Garbage out'. No surprise there.