Determining the Drag Coefficient Of A Model Rocket Using Acclerometer Based Payloads

By Tim Van Milligan
C-Division
NAR 35872

R&D Event
NARAM-54
July, 2012

Summary

In order to accurately predict the performance of a model rocket, whether by doing simple hand calculations or using sophisticated computer software, the modeler must have a reliable number for the rocket's Drag Coefficient.

The most common method of deriving the Drag Coefficient is to use back-tracking, after measuring the peak altitude of the rocket. Back-tracking is where you guess at a Cd value, enter it into the computer program, and find out if the resulting predicted altitude was too high or too low compared to the actual flight data. If it was too low, you'd lower the Drag Coefficient guess, and then rerun the simulations. Through an iterative process of making good guesses, you would zero in on the actual Drag Coefficient that was necessary to obtain the actual altitude.

The problem with this approach is that in real life, there are a lot of other variables that would alter the altitude. The way to solve this is to perform multiple flights and average out the results. Unfortunately, this can get expensive as you burn up a lot of expensive rocket engines.

In this report, I will show several other ways to solve for the Drag Coefficient without having to burn up extra motors. In the end, if the modeler has a recording altimeter that utilizes an accelerometer to make the altitude measurements, it is relatively easy to get an actual and accurate Drag Coefficient value with just a single flight. While these altimeters cost approximately $100, the savings in rocket motors could easily justify the expense. Additionally, the cost of altimeters is coming down, while the expense of motors is going up. So I predict that in the future, this will be a very economical way to get an accurate value of the Drag Coefficient.

The Objectives of the Work:

The purpose of this report is to discuss the ways a modeler can accurately determine the Drag Coefficient (C_d) of their model rockets.

The reason this is important, as it allows you to more accurately predict the trajectory, altitude, and the general performance of your rocket. This increas in accuracy means that you have greater confidence that your rocket will fly as intended. For example, it allows you to stay below the maximum altitude waiver when flying high power rockets, or for saving money when trying to dial in the performance of the model competitions, such as Team Aerica Rocket Competition (TARC), where consistency is of critical importance.

Once the C_d is know, what do you do with it?

What most people do is plug this number back into computer programs that predict performance. RockSim is currently the program that many modelers use, so I will reference it in this report.

Because it accounts for so many of the variables associated with flight, people really have a pretty good understanding of just how accurate RockSim can be. The one Achilles heel is determining the Drag Coefficient of the rocket.

This is a problem that even plagues NASA. While software has gotten better, and it will continue to get better, there is still an uncertainty of just what is the actual Drag Coefficient of the flying object.

But still, people want to know how RockSim can be made more accurate at predicting the Drag Coefficient. That will be the topic of this report.

Note to Contest Judges: The majority of this work was published in Apogee Component's *Peak-of-Flight Newsletter,* Issue #303 (January 3, 2012). I have added additional material toward the end that further refines the process and makes it easier to find the Cd of the rocket.

Approaches Taken For Determing The C_d of a Rocket.

There are several approaches to finding the C_d value of the rocket. Let's start with the most common approach.

Altitude Back-Tracking

The method that most people use to get a better value on the Drag Coefficient is called back-tracking. It is pretty easy to do, and here is the procedure:

1. Put an altimeter in the rocket. Measure the actual height of the rocket. For example, in your test flight, the altimeter says the rocket went to a height of 1000 feet.

2. Input the design into RockSim. Run a simple simulation, using a fixed C_d value. A good starting point is to set the C_d at 0.75.

3. Compare the altitude that RockSim predicts against the altitude measured by the altimeter.

4. Adjust the C_d value in the software (make it higher if the previous simulation went higher than the measured altitude from the altimeter, and lower if it didn't simulate high enough). Run a new simulation.

5. Go back to step 3. Repeat this process until the altitudes match.

The final C_d entered into RockSim that was used to match the actual altitude from the Altimeter is now your "back-tracked" C_d value.

This process does involve a little bit of effort. But you can usually zero-in the C_d value within 5 or 6 iterations of the process.

Apogee Components used to sell a program called SMARTSim that sped up the process. But unfortunately, the programmer got tired of repairing it every time Microsoft created a Windows OS update. Apogee's long-term goal is to get that program back, and maybe even incorporated directly into RockSim for this very purpose.

How Good Is Altitude Back-Tracking?

That is the million-dollar question, isn't it?

It should be pretty good. And many successful TARC teams have proved that it does work well.

But, the big assumption is that the rocket flew the same trajectory as predicted by RockSim. In RockSim, you probably set the conditions to try to match the actual flight conditions, but it may not be "perfect." And, as a lot of TARC teams know, there is some variability between rocket motors. So while RockSim assumes the motor may have put out exactly 80 N-s of total power, in real life, you might have gotten to the altitude with just 75 N-s of power.

So on your next real-life flight, if you have a motor with 79 N-s of power, it is going to go a bit higher, and now you're thinking that your back-tracked C_d is too low.

The solution to this, of course, is to fly a high number of test flights and back-track the C_d value for each of them so that you average out the variability of the motors. Then you'd average them so that you'd get a better approximation of the actual C_d value for the rocket.

This is where it starts getting a little expensive, because you're burning a lot of rocket motors. But it is actually the right thing to do.

We Need More Data

As you can tell, we need to gather a lot of flight data, so that we can dial-in on that elusive C_d value. The more data you can get, the quicker you can find that C_d value.

One additional piece of data that we can get on the flight is the velocity of the rocket, either through the recording altimeter, or through the AltimeterTwo altimeter (www.ApogeeRockets.com/AltimeterTwo.asp). It is better to use a device that has an accelerometer on board, as you'll get more accurate readings.

A pressure-sensing altimeter can give you rougher speed readings on velocity, but it isn't quite as accurate.

For one thing, it has a little lag in sensing the air pressure inside the rocket. To understand this, think of the electronics bay as a balloon filled with air. There are holes in the balloon, so air is escaping. Air continues to flow out until the pressure inside the balloon equals the outside air pressure. It doesn't change instantly, as you know. It takes a few moments for the air pressure to equalize.

The same thing happens in a rocket. On the ground, the pressure inside the e-bay is the same as the outside. As the rocket ascends, the outside air pressure decreases. So the air inside the e-bay has to flow out until they equalize. When it is equalized, that is the point where you want to take the pressure reading. The big problem comes when the rocket is moving really fast, since the outside air pressure is rapidly changing because the altitude changes quickly.

The most accurate readings for a pressure sensor occur when the rocket is moving slow – such as when it nears the apogee point in the flight (see Figure 1). At this point, there is no upward velocity, so it is a great time to make a pressure reading. There is plenty of time for the air pressure to equalize when the outside pressure is staying constant. And that is exactly why pressure sensors are used to measure height.

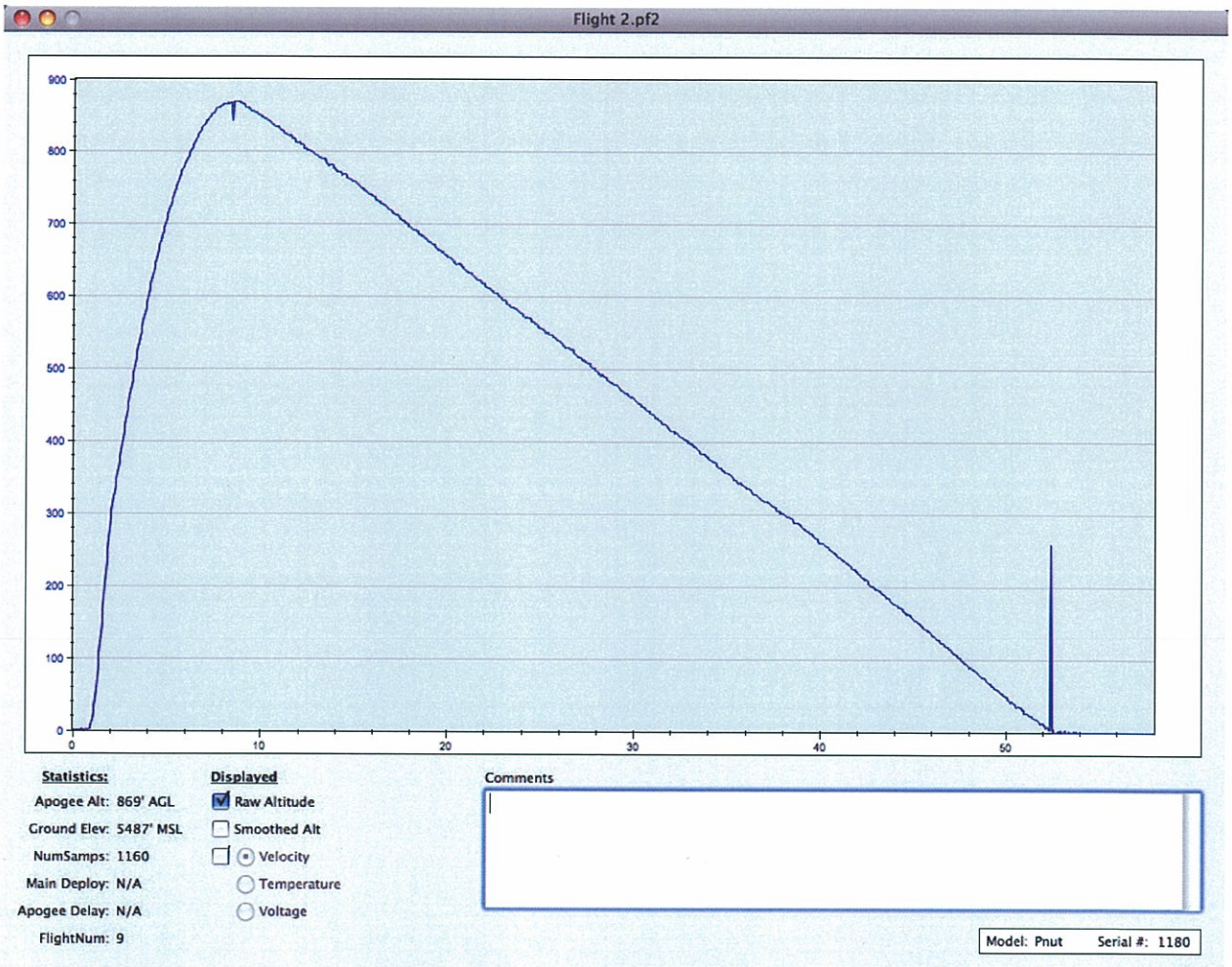

Figure 1: Altimeter data from a recording altimeter. The curve is nice and smooth indicating good data.

But unfortunately, when it is moving slow, even subtle changes in pressure readings can indicate fast *speed* changes. If you look at the speed chart from a recording Barometric altitude sensor, they fluctuate wildly when the rocket is moving at a slow speed. For example, in Figure 2, notice how jagged the graph is when the rocket is descending under the parachute. Also note, that even before the rocket is launched, the speed is rapidly changing. And you know that can't be right, since the rocket is actually stationary prior to launch.

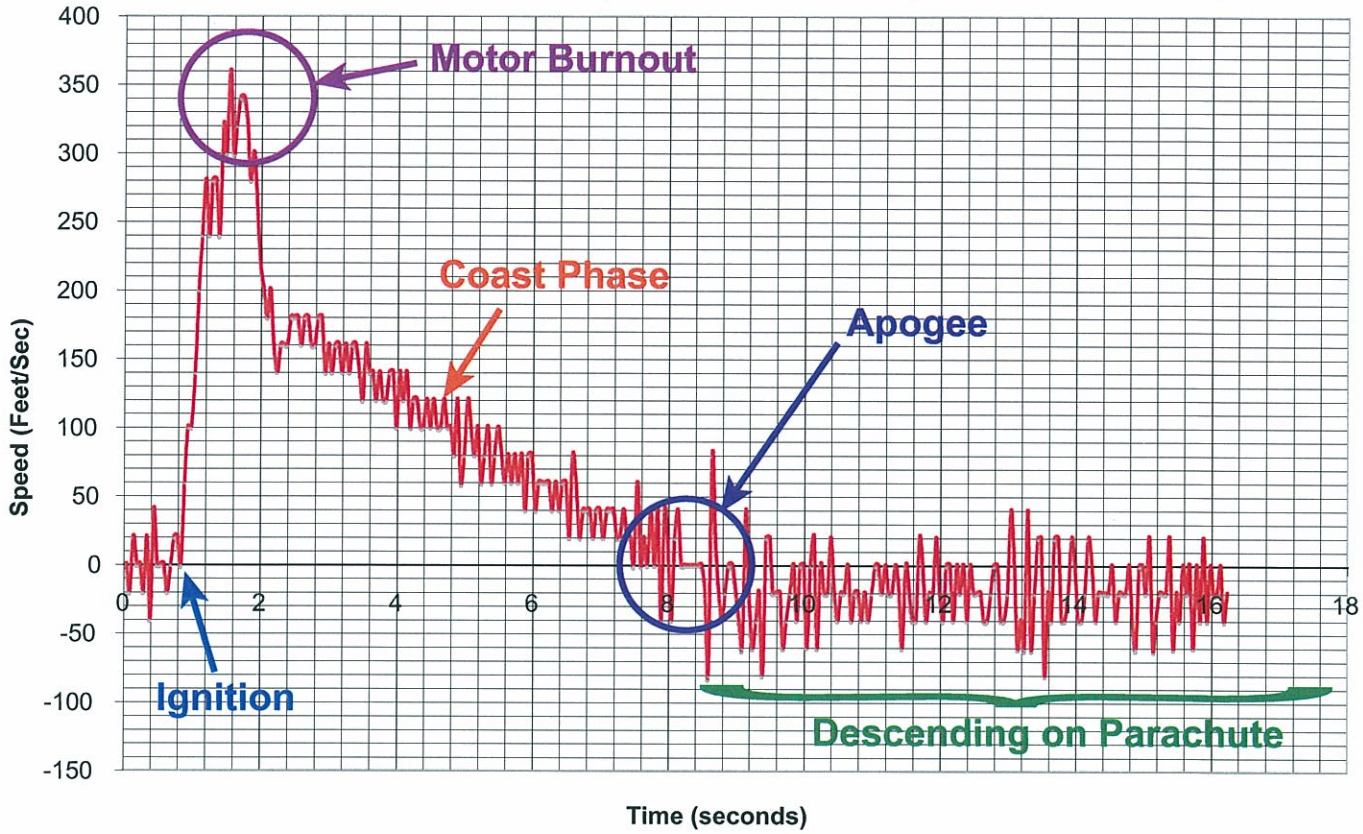

Figure 2: Speed graph created from barometric altitude data. Notice the data is more jumpy at slow speeds.

Pressure sensors are great for measuring peak altitude, but not so great when the rocket is moving rapidly upward.

That is why you want to use an accelerometer-based sensor to measure speed. It reacts much quicker.

For your reference, the payloads Apogee Components sells that have accelerometers on them are: Altimeter-Two, G-Wiz Flight Computer (www.ApogeeRockets.com/G-Wiz_flight_computers.asp), and the TeleMetrum (www.ApogeeRockets.com/Altus_Metrum_GPS.asp).

What Do You Do With Velocity?

Now you have this extra velocity data, what do you do with it?

Basically, you treat it the same way as you'd treat peak altitude. You back-track out a C_d based on that velocity.

1. Put an accelerometer in the rocket. Measure the maximum speed of the rocket. For example, in your test flight, the accelerometer says the rocket went to a speed of 250 miles per hour.

2. Input the design into RockSim. Run a simple simulation, using a fixed C_d value. A good starting point is to set the C_d at 0.75.

3. Compare the maximum speed that RockSim predicts against the speed measured by the accelerometer.

4. Adjust the C_d value in the software (make it higher if the previous simulation went faster than the measured speed from the accelerometer, and lower if it didn't simulate fast enough). Run a new simulation.

5. Go back to step 3. Repeat this process until the maximum speeds match.

The final C_d entered into RockSim that was used to match the actual maximum speed from the accelerometer

is now your *"back-tracked"* C_d value.

If you use any of the accelerometer-based electronic payloads that also have a barometric sensor included (like the AltimeterTwo), the advantage is that you have two independant sources of data. So you can back-track out a C_d value based on the maximum speed, and a C_d based on the peak altitude. You've now doubled the amount of data for the same cost of the rocket motor!

With this extra data, you can average out the C_d values and get to a final value in a faster amount of time.

Eliminating Motor Variation

As mentioned previously, the one big unknown we still have is the rocket motor variability.

First of all, why is there some variability in the thrust produced by the rocket motors? There are two factors.

First is the chemical formulation of the propellant. As you know, there are different chemicals that make up the final propellant. These are weighed out and mixed together. Unfortunately, errors creep into the manufacturing process. Are the weights exactly equal down to the micro-gram? Probably not. Are they mixed perfectly together so that the chemicals are evenly distributed throughout the entire volume? Probably not. Are the purity of the chemicals used the same from batch to batch? Again, probably not.

The second way errors creep into the motors is through the geometric properties of the motors. For example, are the propellant grains exactly the same length from motor to motor? Are the slots or the holes in the propellant exact from motor to motor? Are there voids (air bubbles) in the propellant? And are those voids in the same place from motor to motor? Nope.

The motor manufacturers are really good at trying to limit the variability by extensive quality assurance procedures in place during production. But the errors still creep in, and that means the thrust of each motor is going to be slightly different.

That means that your back-tracked C_d is going to be off. The reason is that RockSim is assuming the motors are perfect from one flight to the next. But they are actually off a little bit.

What would be ideal is if we could eliminate the motor from the determination of the Drag Coefficient. And there is a couple of ways to do this.

Wind Tunnels

The obvious method is to use a wind tunnel to measure the drag forces on the rocket. And this is a great way to do it. In fact, this is the way that NASA does it. If you have a good wind tunnel with perfectly smooth airflow, and good measuring equipment, you can get highly accurate values for the Drag Coefficient.

Unfortunately, most TARC teams nor average hobbyists have access to such test equipment.

But there is a second way to get C_d values that eliminate the effects of the rocket motors.

Terminal Velocity Measurements

In the February 1970 issue of *Model Rocketry Magazine,* Tom Milkie wrote an article on how to determine the drag coefficient of your rocket by dropping them out of a window. It was actually that article that sparked the idea for this one.

The reason for dropping the rocket out the window is to get the rocket to reach its terminal velocity. This is good, because it eliminates most all the variables except the one important one. That is the *Drag Coefficient.*

When the rocket falls, the two forces acting on the rocket pull in opposite directions. Gravity pulls harder at first and the falling rocket gains speed. As the speed increases, the Drag force increases. When it gains enough speed, the two forces eventually equal out. This is the terminal velocity speed.

Terminal velocity occurs when the downward force of gravity (F_g) equals the upward force of drag (F_d). This causes the net force on the object to be zero, resulting in an acceleration of zero.

$$\text{Force due to gravity} = F_g = m\,g$$

$$\text{Force due to Drag} = F_d = \frac{1}{2}\rho\, v^2\, A\, C_d$$

Where:

v = velocity (m/s)

m = mass of the falling object – (kg)

g = acceleration due to gravity = 9.80665 m/s^2

C_d = drag coefficient

ρ = density of the air = 1.225 kg/m^3 at 15°C

A = projected area of the object - Typically this is the cross-sectional area at the base of the nose cone, or the largest diameter of the rocket. You have a choice in RockSim, but make sure you use the one that RockSim is using when you work through the equations.

Now let's set the two to be equal:

Gravity Force = Drag Force

$$m\,g = \frac{1}{2}\rho\, v^2\, A\, C_d$$

Solving for this equation for the Drag Coefficient (C_d) yields.

$$C_d = \frac{2\,m\,g}{\rho\, v^2\, A}$$

Note that this is the same equation we use when finding the drag coefficient of a descending parachute. The only thing that changes is the *Area.*

At this point, when you find the terminal velocity, you can simply plug it into this equation.

Watch your UNITS!

When you solve this equation, you have to be sure to watch your units. It is a lot easier to solve this using metric units, so make sure all your numbers are in metric before solving it.

How do you get the rocket to reach terminal velocity?

Simple. Drop it from a very great height. In our case, we'll launch the rocket to a very great height, let it arc over and come screaming down. Obviously, that isn't too safe if the rocket impact the grounds at a high velocity. So you'll want to do two things.

First, select a motor with a very long delay, so that it ejects before it hits the ground.

Second, make the rocket as lightweight as possible. The lower the weight, the slower the terminal velocity. Slower is safer.

Here are some other things to consider:

First, you have to use a recording accelerometer, because you'll be getting the downward portion of the trajectory. This currently eliminates all but the G-Wiz Flight Computer and the TeleMetrum. But at the present time, this means you're flying something larger than a typical model rocket.

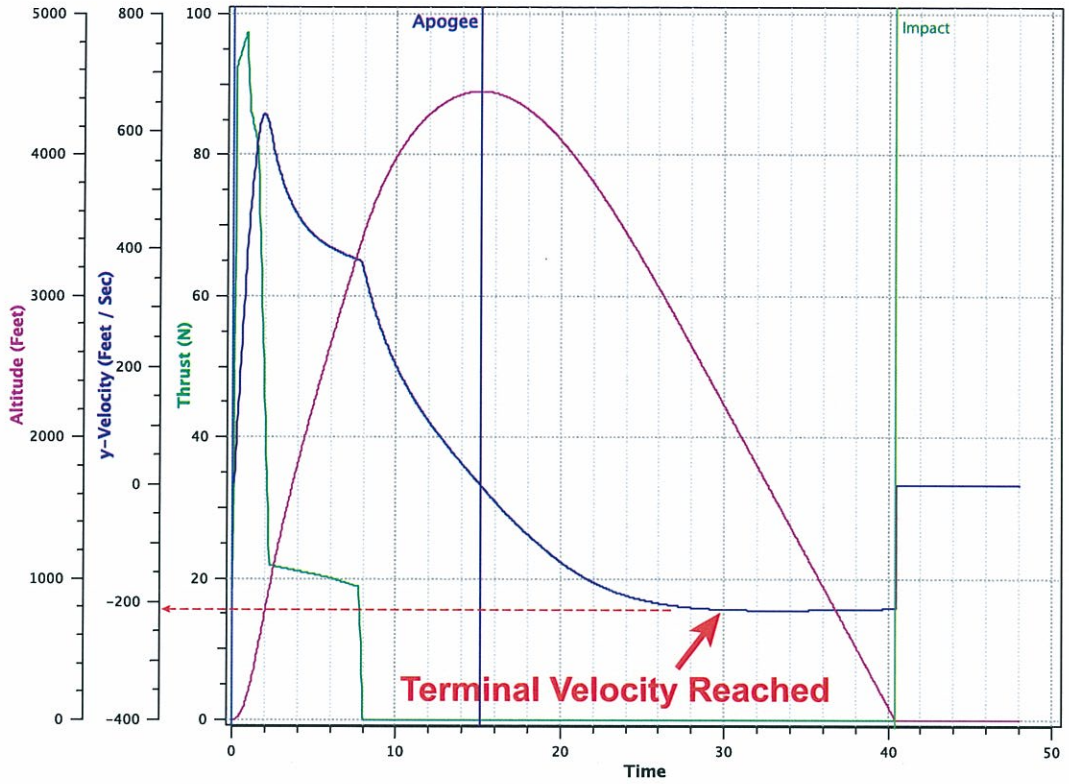

Figure 3: Terminal velocity is reached when the speed curve flattens out.

The Drag Coefficient is dependant on the shape of the rocket, so instead of using a full-size rocket, use a smaller one (but with the same shape). Not only is it going to be lighter weight, but you can use a smaller motor to launch it into the air. In other words, smaller equals cheaper! You want to save money doing this, right? Make it just large enough to use the recording accelerometer.

Also make sure that your recovery device can withstand a high-speed deployment. You'll probably want to switch over to a streamer instead of a parachute, at least for the expensive part of the altimeter.

If I were to do this myself, here is the procedure that I'd use:

1. Start with a few RockSim simulations to dial in on the delay to use in the rocket motor. You want to initially set the Drag Coefficient to a low number (such as 0.5), so that the rocket will come down faster. Why? Because I'd rather deploy a little too early than have the rocket impact into the ground. In RockSim, pick a really long delay (at least 15 seconds on a typical size rocket). You're going to *TYPE* this number into RockSim's delay field. Hit the **ENTER** key (not the **RETURN**) to force RockSim to accept the new value for the delay.

2. Remember, you also want to select a motor that is powerful enough to put the rocket to a sufficient height, that when it arcs over and comes down, it can reach that terminal velocity before it impacts the ground.

3. Launch the real rocket straight up, with as little wind as possible. You want the downward trajectory to be perfectly straight. Collect the terminal velocity speed from the recording accelerometer.

4. Compare the terminal velocity speed that RockSim predicts against the speed measured by the accelerometer.

5. Adjust the C_d value in the software (make it higher if the previous simulation went faster than the measured speed from the accelerometer, and lower if it didn't simulate fast enough). Run a new simulation.

6. Go back to step 4. Repeat this process until the terminal velocity speeds match.

The last C_d value is now the "back-tracked" C_d value of the rocket.

The advantage of this method, as we said, is that it completely eliminates the random effects of the motor. That's cool, huh? That means it will only take a few flights to dial in on the actual C_d of the design.

What if The Rocket Doesn't Reach Terminal Velocity?

You launch the rocket into the air, and it doesn't quite reach terminal velocity as it descends. What do you do then?

You do what all good engineers do. You *cheat*.

What you do is look at the curve, and extrapolate out where it levels out to zero. You can see how this is done in Figure 4, as compared to Figure 3. You may be off a little bit, but it won't be much. The longer the curve that you captured in the data, the more accurate your results will be.

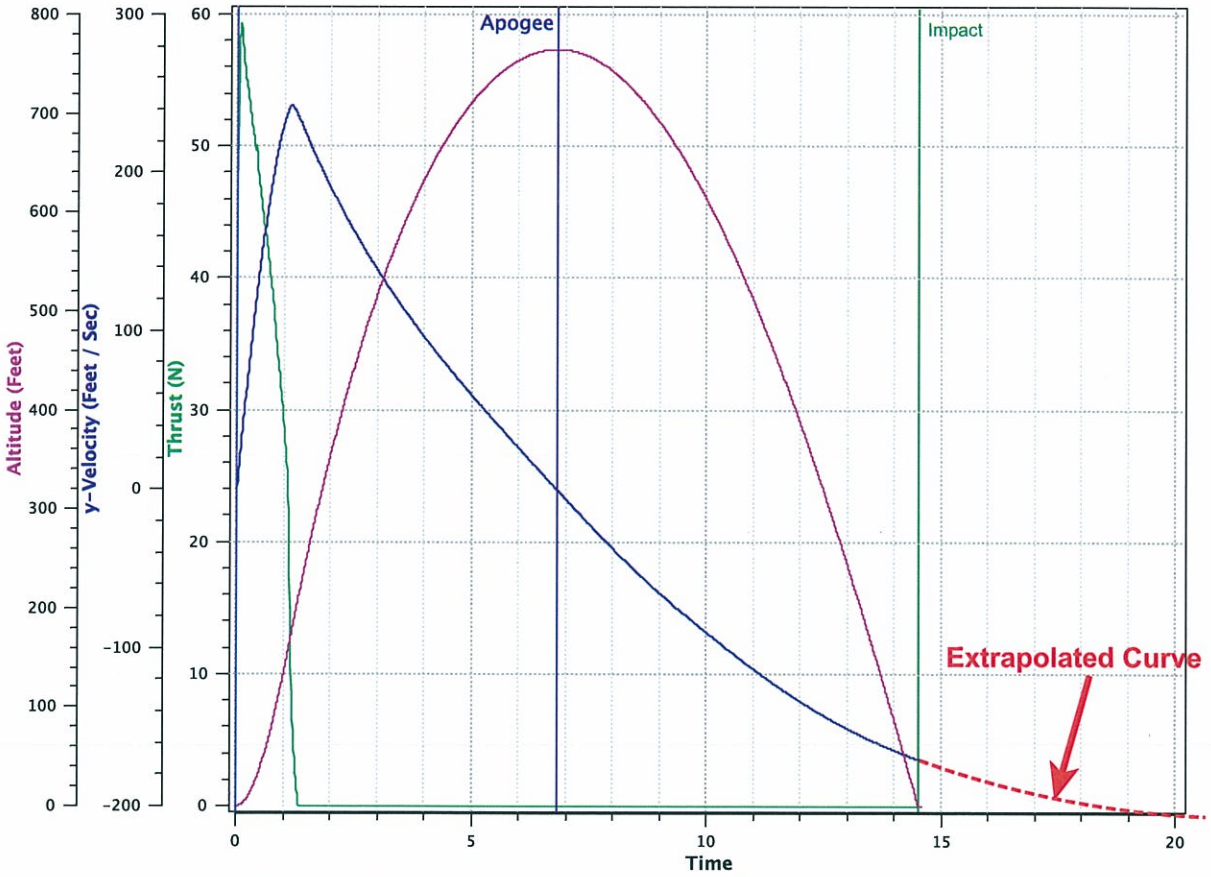

Figure 4: If you can't get the curve to flatten out, just extrapolate the end portion. Your eyeball is good at smoothing out the curve! Compare this to Figure 3.

In the simulations I did here, when this particular rocket did reach Terminal Velocity, the speed according to RockSim's detailed flight report was 210.32 ft/sec. See Figure 5. From the graph in Figure 4, I would estimate that the extrapolation would show a terminal velocity around 208 ft/sec. That's an error of just 1%!

Where The Error Creeps In

Now we didn't eliminate all the variables, but we got pretty close. What variables might we reduce if you want even greater accuracy?

First, the rocket has to come down perfectly straight. If it wobbles at all, the Drag Coefficient is going to fluctuate wildly. That is because the rocket is presenting the side of the tube to the airstream, instead of just the top of the nose cone. In other words, the surface area term has changed. But we actually see this in the C_d changing, since we assumed the reference area was a constant value.

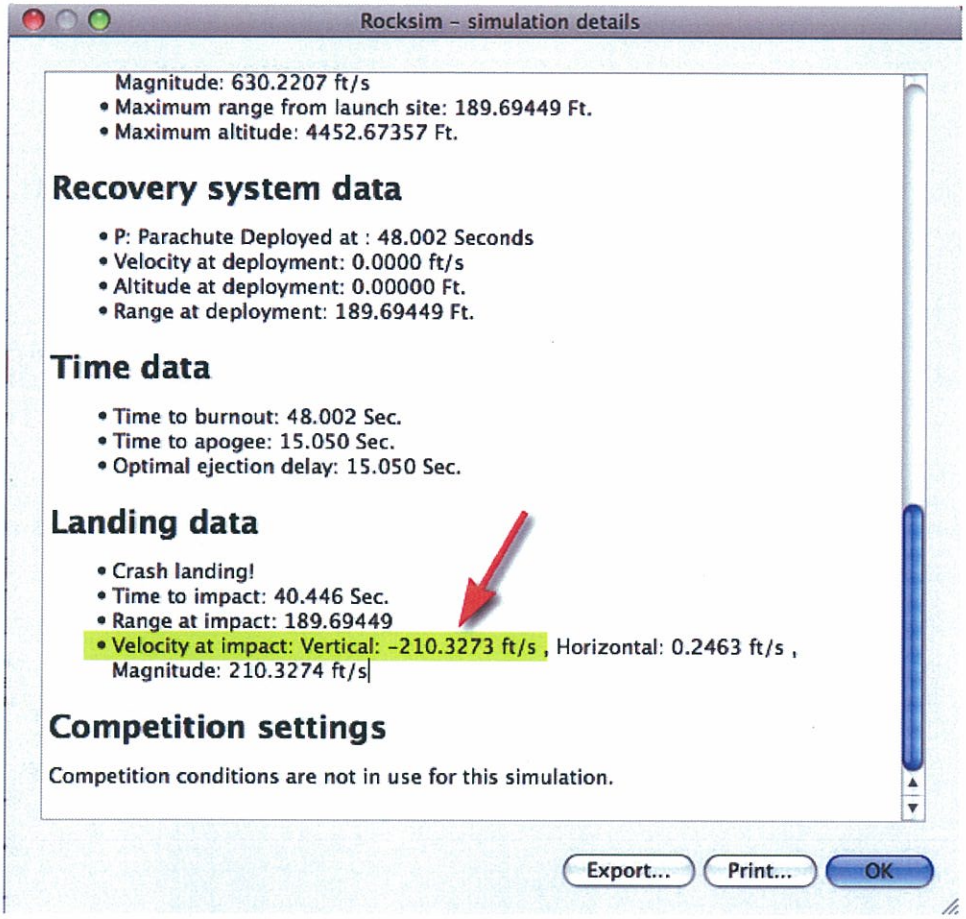

Figure 5: The terminal velocity can be read directly from the Simulation Details Report in RockSim.

Secondly, the density of the air is changing on us too. The rocket starts at a high altitude where the air is thin, and descends into thicker air as it comes down. This means that as it descends, the Terminal Velocity is going to be a little lower. You see this subtle change in Figure 3 where the velocity begins to fall again as the rocket gets closer to the ground.

The good news is that if you launch on a calm wind day, you can probably eliminate most of the wobble. And that is about as good as you'll get.

Is There a Safer Way?

Having your rockets come streamlining down to the ground at a high speed is not something that I want to do either. It carries a high risk. If the ejection charge doesn't deploy at the right time – SMACK! Or if the chute strips off because of the high-speed deployment – CRACK! It is a brutal deployment situation, and I would worry about the electronics surviving this kind of trajectory.

I would only do it if I was in a remote area further away from spectators and property. You must be extremely careful.

Is there a safer way?

Not quite yet. But we're not too far from the day. In fact, if you're a computer programmer, you might speed that day along by writing a program to do what I'm about to describe.

Consider this situation…

Why did we want to get the rocket to come in ballistically to reach terminal velocity? Right. To eliminate the variability of the motor's performance. We had to wait until the motor burned out to start making measurements.

Furthermore, we had to wait until the rocket's speed evened out so that there was no net forces acting on the rocket – where the drag force equaled the gravity force.

The question I have for you is *"why even wait for the drag force to equal the gravity force?" Can we deduce the drag force directly in another manner?*

The answer is yes.

We already are calculating the drag force in RockSim (assuming we know the Drag Coefficient). **The result is that you can predict the speed and acceleration of the rocket as it ascends.**

You may be confused by this, so let me try to explain. What I'm saying is, we should be able to predict how fast the rocket starts to decelerate after the motor burns out. Why? Because, the only thing that controls this is the force of gravity (which is known), and the force of drag – the deceleration is the sum of drag plus the gravity force.

For example, Figure 6 shows the deceleration curves of a rocket after motor burnout, and before it reaches Apgoee for three different C_d values. In other words, this is just a snipet of the flight during coast, but before the rocket reaches apogee.

Do you see in Figure 6 that each Cd gives us a different deceleration curve? This shape is important!

Before I go on, I want to point out that this is something that can actually be measured using the accelerometer data!

Doing it this way, we can back-track out the C_d of the rocket, not by comparing a single point on the curve (like just peak altitude, burnout speed, or Terminal Velocity), but by comparing the "SHAPE" of the curve itself.

There is only one C_d value that can create a specific-shape deceleration curve.

What you see in Figure 6 are three different deceleration-curves based on different Drag Coefficients. You can definitely see that they are different. But unfortunately, they are a little hard to make comparisons in this format. They need to be separated out so that it is easier to distinguish the different shapes for the different drag coefficient values.

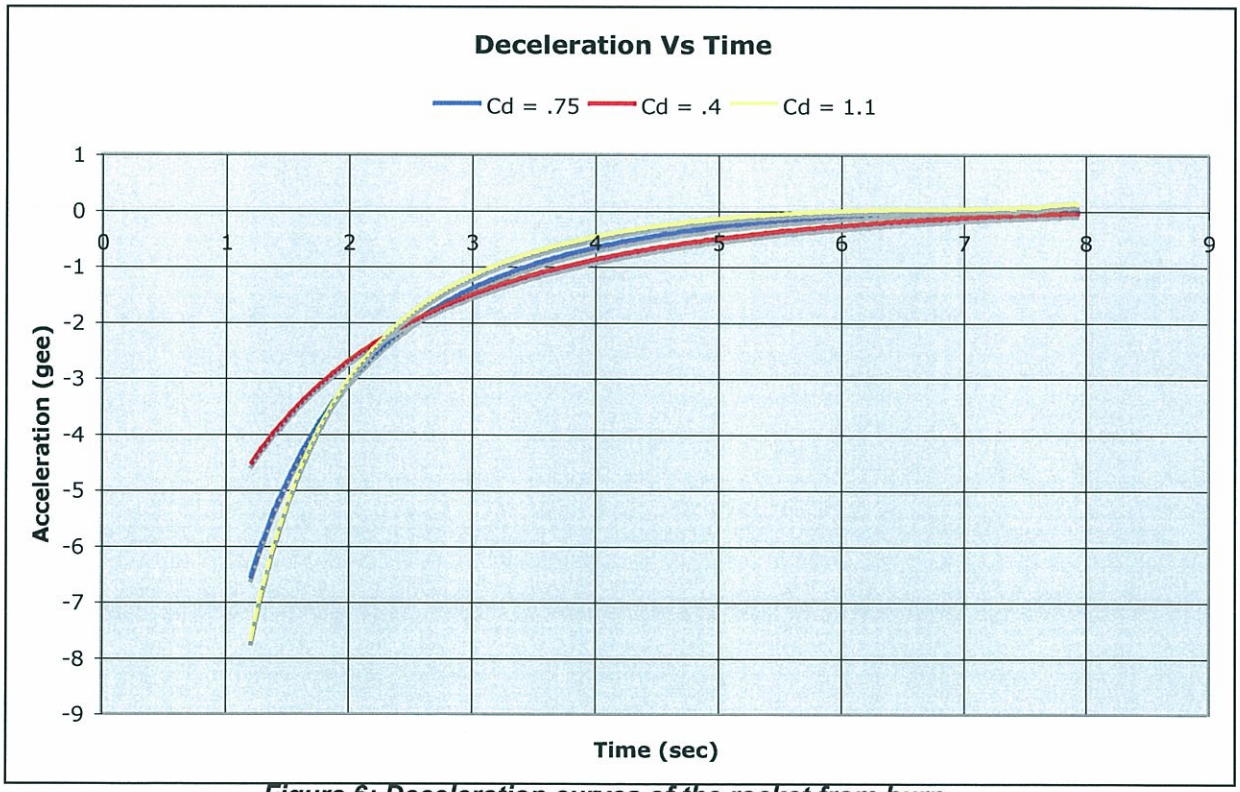

Figure 6: Deceleration curves of the rocket from burnout to apogee for three different C_d values.

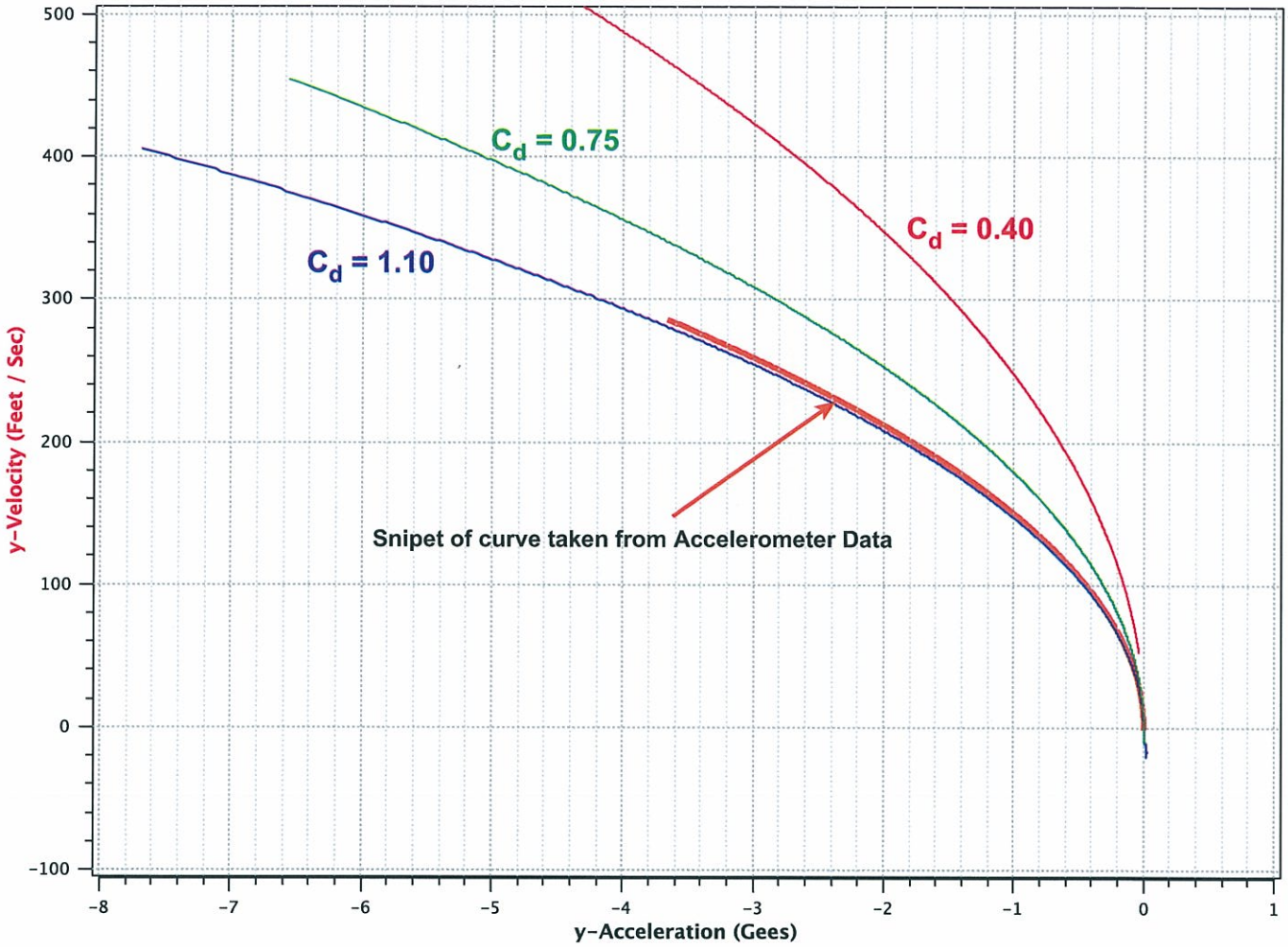

Figure 7: Just by plotting out the data a little differently, (Deceleration vs Speed), you get much greater separation of the curves for the three different C_d values.

I did this in Figure 7 by plotting out the deceleration values versus Speed. Now you can definitely distinguish the different shapes of the curves.

How would you use this procedure for finding the Cd value of the rocket?

1. Here is the part where we need some new software. The first thing we need to do is to create a chart like Figure 7, with different deceleration curves for the various values of Drag Coefficients. It has to use the rocket design we intend to fly, just like we do with the other back-tracking methods. We still need an accurate shape and mass, plus all the launch conditions accounted for.

2. After running these simulations, now go out and launch the real rocket straight up on a windless day. We always need windless, because we don't want the rocket to wobble or arc over as it takes off. Record the acceleration data from lift-off to apogee.

3. From the acceleration data, create a new plot showing the acceleration versus speed. You can toss out that part of the plot during the motor's thrust phase. It will be skewed by the variations in thrust anyway.

4. Overlay the actual plot on top of the deceleration curves. Make sure the scales on the X-Y axes are the same. In Figure 7, I placed a snipet of acceleration data on the graph so you can get an idea of how you're going to overlay the curves.

5. The C_d value is one that best matches the curve from the launch.

The Advantages of This Method

There are several major advantages to this method.

First, it is safe. We can prep the rocket like any normal flight. We're not going to have it come streamlining toward the ground at a high rate of speed.

Second, it eliminates the variation in motor thrust between launches. So it will take fewer test flights to dial in on the C_d value.

Third, we can use the actual rocket to make the measurements. We don't have to make a scale model just to keep the weight low - like you have to do when making Terminal Velocity measurements.

Fourth, it should be highly accurate because it is easy to distinguish how the curves overlay each other.

Fifth, if you look at the chart in Figure 7, one thing should pop right out at you: the higher the speed of the rocket, the further the lines are apart for each C_d value. What this means is that you actually want to use a high-thrust motor when you measure the C_d value. You want your rocket to go REAL fast, so it has a longer amount of time to decelerate. That way, you'll get a longer curve, which makes matching up on the overlay a lot easier.

Also, the accelerometers are more accurate when there are larger forces to measure, which means that going faster makes the accuracy higher.

So what you'd do is make your Drag Coefficients with high thrust motors, and then fly your actual launches (like TARC qualification flights), with any motor you'd like.

One last thing...

Notice that the snipet of deceleration data in Figure 7 does not match perfectly up with the C_d curves. The reason is that we're seeing some of the affects of density variation in the flight.

The curves were generated with a high thrust motor (I used an E30 in the simulations). The snipet of deceleration data was generated using a lower thrust motor (an E15). The E15, because it has lower thrust, is more efficient in fighting drag because it keeps the speed of the rocket lower. Remember, drag is proportional to the square of velocity (double the velocity, the drag goes up four times).

So the E15 was higher in the sky when the motor burned out. At higher altitudes, the air is less dense, so it acts like the rocket has a lower Drag Coefficient. That is why the curve bends slightly upward and isn't a perfect match for the Drag Coefficient of 1.1. But the shape is so close, that you can easily tell what the C_d value was.

C_d Calculation Without Curves

A couple of months after I had written this article in the *Peak-of-Flight Newsletter,* I got an email from Dan Moses, who was a mentor for a NASA SLI team from Falls Church, Virginia. They took my original article, and went one step futher to calculate the C_d without having to make comparison charts. I don't know why I didn't think of this, because it is so logical.

Here is what that SLI team did:

They went back to the original equations for drag. When the rocket motor burns out, the forces acting on the model are the Drag Force, and the Force of Gravity.

Total Forces acting on model = Drag Force + Force of Gravity

F= m a = Drag Force + Force of Gravity

That gives us this formula:

$$m\,a = \frac{1}{2}\rho\,V^2\,A\,C_d + m\,g$$

where g is the force due to gravity (9.81 m/s^2)

From there, we just solve for the Drag Coefficient. This gives us this equation:

$$C_d = \frac{2\,m\,(a - g)}{\rho\,V^2\,A}$$

At this point, we only need to find the acceleration force as measured by the accelerometer on board the rocket.

But we're going to find that the accleration as measured, is going to be a little scattered which means selecting the "a" to use is going to be difficult. The reason is due to a couple effects, such as density variation with altitude which was mentioned previously. Also, if there is any wobble in the flight or cross wind, the model will experience slightly more drag (in effect, the reference area will change). These slight variations need to be smoothed out.

What the NASA SLI team did, was to plot Acceleration versus Velocity squared from burnout to apogee, as shown in Figure 8. The nice thing about this choice of variables is that the data creates a rather straight line, which makes creating a best-fit line rather easy.

You'll notice on this chart, that most of the data points are at the lower speeds, which makes sense. That is

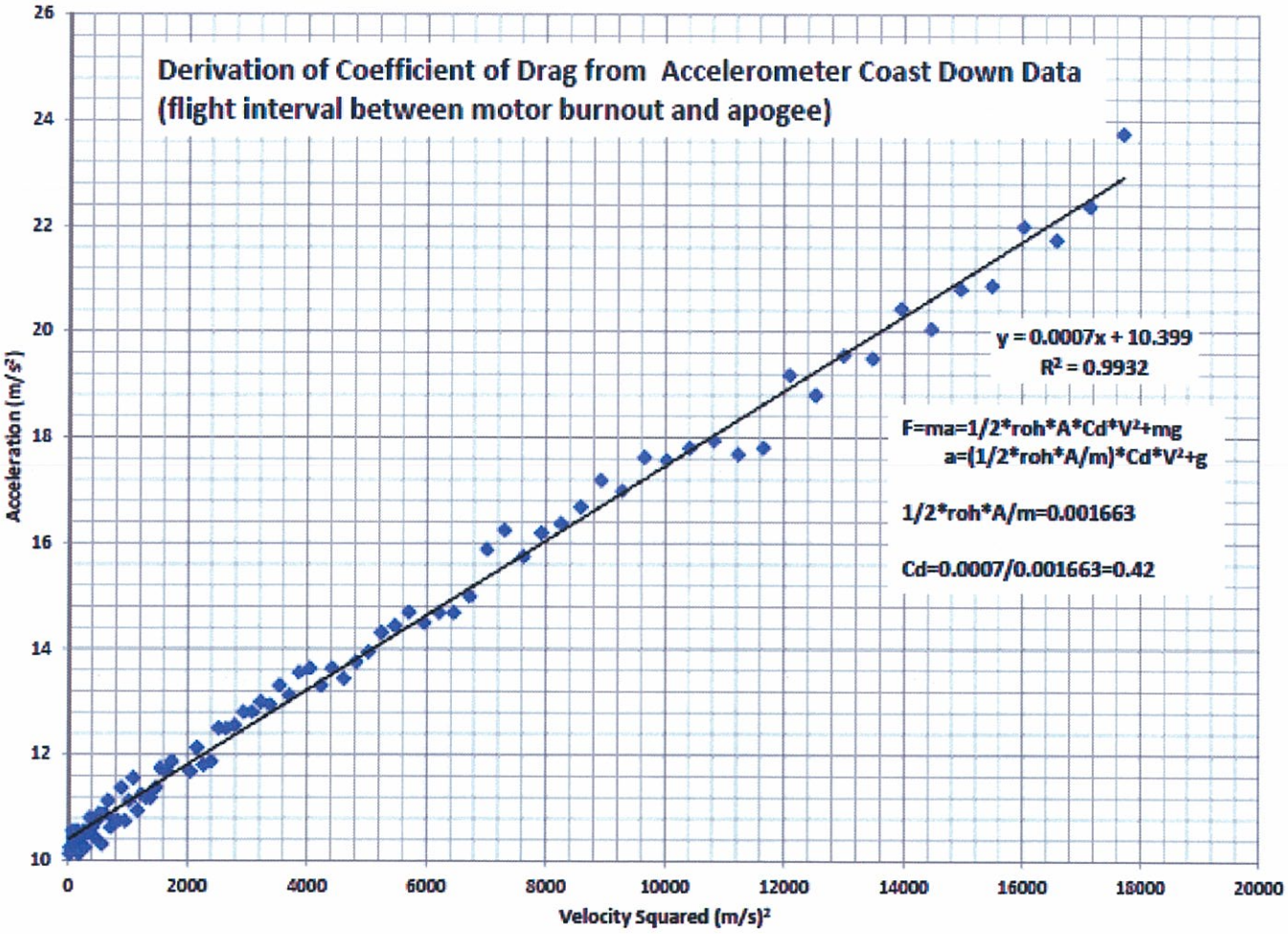

Figure 8: Formatting the accelerometer data to plot out the acceleration versus velocity squared gives a nice straight line. (Source: Dan Moses)

the point near apogee where the intervals in sampling are bunched together.

At this point, you can easily pick a point on the line to select the acceleration, and then running it through the Cd formula.

This method is the very direct, as you only have to make a single plot using the accelerometer data, and thus is the preferred way of finding an accurate estimate of the rocket's Drag Coefficient.

Additional Information

Maximum Simulation Accuracy from RockSim, Peak-of-Flight Newsletter 45 (www.ApogeeRockets.com/Education/Downloads/Newsletter45.pdf).

Smarter Guessing and Simulating With RockSim 5, Peak-of-Flight Newsletter 60 (www.ApogeeRockets.com/Education/Downloads/Newsletter60.pdf).

Optimizing Your Design Using SMARTSim, Peak-of-Flight Newsletter 130 (www.ApogeeRockets.com/Education/Downloads/Newsletter130.pdf).

SMARTSim Tips for design Optimization, Peak-of-Flight Newsletter 134 (www.ApogeeRockets.com/Education/Downloads/Newsletter134.pdf).

Sleuthing Altimeter Data for Critical Answers, Peak-of-Flight Newsletter 208 (www.ApogeeRockets.com/Education/Downloads/Newsletter208.pdf).

How To Zero In on The TARC Altitude and Duration Objectives, Peak-of-Flight Newsletter 249 (www.ApogeeRockets.com/Education/Downloads/Newsletter249.pdf).

References:

Via email, Dan Moses wrote: *"In response to your interest in deriving a rocket C_d from altimeter data* (Peak-of-Flight Newsletter 303 at www.ApogeeRockets.com/Education/Downloads/Newsletter303.pdf), *I would like to direct your attention to the results obtained by the NASA SLI group I mentor at Falls Church High School. They used one of the AltusMetrum altimeters (*www.ApogeeRockets.com/Electronics_Payloads/Altimeters/TeleMetrum_Starter_Set*) purchased from Apogee in their full scale test flight in March. From a plot of acceleration vs velocity squared during the coast phase, they came up with a very good estimate of the rocket C_d. Have a look at slides 11 and 12 of their FRR presentation posted on their web site: http://www.fallschurchsli.org/documents* (See Figure 8).

Previous R&D Reports On This Subject by Author

N/A

Equipement Used

Data collected in this report came from the Telemetrum GPS altimeter.

Facilities Used

N/A

Money Spent

No money was spent. Data was gathered from flights perfomed for other projects.

Data Collected

N/A

Results Obtained

N/A

Conclusions Drawn

Getting an accurate C_d value is no longer a trial and error process involving a number of flights. With mod-

ern electronic payloads that include an accelerometer, it is possible to dial in the the elusive Cd value with just a single flight. This number can then be input into computer software to make predictions of future flights even more accurate.

Further Work

John Beans, the creator of the AltimeterOne and AltimeterTwo, also saw the original article in the *Peak-of-Flight Newsletter*, and it inspired him also. He mentioned to me that he might make a special version of the AltimeterTwo that would directly report the deceleration of the rocket during the period between engine burnout and apogee. This is really cool as it will make finding the C_d even easier, as the computer on board the device can do most of the calculation for you. All you would need is the burnout mass and the cross sectional area of the rocket.

John wrote via email: "It wouldn't be too hard to add another data readout to the AltimeterTwo to help with C_d calculations using the deceleration method. After all, all of the data is available internally to the AltimeterTwo firmware. But it would necessarily end up being a "partial" answer, a figure that the user would have to multiply by stuff the AtlimeterTwo can't sense like rocket weight and frontal area to get to a pure C_d. That might be awkward to use in practice. What would you call AC_d/m? The instructions would be: *"To get the C_d, take the value of AC_d/m calculated by the AltimeterTwo, multiply it by the weight of your rocket AFTER launch (empty motor still in, add back wadding), then divide by the frontal area (in appropriate units)."* Assumptions would have to be made about air density, which would vary with weather. Without measuring outside air temp, humidity, and actual altitude that's tough."

As further work, I will to continue to push electronics manufacturers to create devices like this. It would mean more accurate results with less work on the part of the modeler.

This R&D Report provided as a membership bonus for joining the National Association of Rocketry at http://nar.org

Check out the other membership bonuses at http://nar.org/members/

Thank you for joining the National Association of Rocketry!