Iterative Logic for Nested Compression Spring Design

By Randy DeFord
Mid-West Spring & Stamping

For years, spring companies have had the task of designing springs that fit inside or over each other. Nested springs are nothing new, but designing them can be a bit time consuming unless you have special tools (i.e. software). Fortunately, these are easy to come by nowadays.

Programming languages have become more and more user-friendly, not to mention flexible. Microsoft's Visual Basic, which made its debut around 1992, has evolved into an intense graphic powerhouse, allowing all kinds of entry formats, buttons, boxes, sliders and gages, as well as audiovisual capability. For non-programmers, there's Excel. A lot of the bells and whistles aren't there, but it provides a powerful way to create and adapt math models, which are the very structure of spring design.

Spring calculations, being linear for the most part, make it easy to set a condition and then compare them one to another. What do I mean by that, you ask?

Let's take the case of a valve spring – an outer/inner situation. Most assumption or guesswork comes from not knowing what wire size to use for the first spring – the outer. The very first step is to define what we need, beginning with what we know. The following list should be typical.

1. Material type (defines modulus).
2. First load at the first height.
3. Second load at the second height (defines total rate).
4. Percent of the load for the outer spring to carry (defines outer spring rate).
5. Stress limit.
6. The outside diameter (O.D.).
7. The maximum solid height (defines total coils limit, based on whether the ends are ground or unground).

It should be obvious at this point that we have a lot of initial information. How do we use it? Let's start by using an example and defining each variable with a typical value.

1. Material: Chrome Silicon Valve per ASTM A877.
2. First load at first height: 450 lb. at 3.500 in.
3. Second load at second height: 700 lb. at 2.900 in.
4. Percent of load for the outer spring to carry: 65%.
5. 140,000 psi stress limit.
6. O.D.: 2.200 in.
7. Maximum solid height: 2.400 in.

At this point it's important to mention that we would need to have a table of the materials for the torsional modulus of that material. That, however, is a simple list with very little programming time invested.

Now, we need one more parameter before we start calculating – the wire size. I am aware that we don't know what it is. So, we work backwards. Not knowing what the wire size should be, we will put in a ridiculously large size, just to have a place to start, and let the computer do a ton of work in a second. If manual calculation were the method of choice, this idea would not be so user-friendly. But with a computer, it's a cinch. We're going to do a calculation, then have the program remove a wire size increment until we meet certain conditions. We would also need to view all this calculated output (text boxes, cells or whatever format you would use to see your resulting parameters) Ready? Here we go:

Step 1. Enter a **d**/wire size of **0.600 in**.

Step 2. Calculate **D**/mean diameter: O.D. - d = 2.200 in. - 0.600 in. = **1.600 in.**

Step 3. Calculate **I.D.**/ inside diameter: D - d = 1.600 in. - **0.600 in.**

Step 4. Calculate **C**/Index: $\frac{D}{d} = \frac{1.600\text{ in.}}{0.600\text{ in.}} = \mathbf{2.667}$

Step 5. Calculate **R**/Rate:

$$R = \frac{P2 - P1}{L1 - P2} \times 0.65 = \frac{700 - 450}{3.5 - 2.9} \times 0.65 = \mathbf{270.833\ lb./in.}$$

Step 6. Calculate **Na**/active coils:

$$Na = \frac{G \times d^4}{8 \times R \times D^3} = \frac{11{,}500{,}000 \times 0.6^4}{8 \times 270.833 \times 1.600^3}$$

$$= \mathbf{167.9}$$

Step 7. Calculate **Nt**/total coils Na +Ni = 167.9 +2 = **169.9**. (With valve springs, Ni may be 1.75.)

Step 8. Calculate **SH**/solid height: Nt × d = 169.9 × 0.6 = **101.94 in.**

Step 9. Calculate **FL**/free length:

$$\frac{P1 \times 0.65}{R} + L1 = \frac{450 \times 0.65}{270.833} + 3.5 = \mathbf{4.580\ in.}$$

Step 10. Calculate **Psh**/load at solid height:

$$R(FL - SH) = 270.833(4.580 - 101.94) = \mathbf{-26{,}368.3lb.}$$

Step 11. Calculate **K**/Wahl's correction factor:

$$\frac{(4 \times c) - 1}{4 \times c} + \frac{0.615}{C} = \frac{(4 \times 2.667) - 1}{(4 \times 2.667) - 4} + \frac{0.615}{2.667} = \mathbf{1.681}$$

Step 12. Calculate **Ssh cor**/corrected stress at solid height:

$$\frac{8 \times Psh \times D}{Pi \times d^3} \times K = \frac{8 \times -26{,}368.3 \times 1.6}{Pi \times 0.600^3} \times 1.681 = \mathbf{-836{,}101psi.}$$

At this point, if you're not wondering what gibberish this is, you're not with the program. A spring engineer will look at the resulting data and know immediately that something is dead wrong. Actually, we're right on track. What we've done is give the program a wire size. The program then calculates all the needed parameters, based on that wire size. If I had to use an analogy, I'd ask that you think of a tree trunk (all possible wire sizes) being whittled into the ideal toothpick (the desired wire size) one stroke at a time using a chainsaw (program) that can do the job in a few thousand nanoseconds. We are now ready to whittle. First we have to state the condition. That condition is the *calculated corrected stress.* We know what stress level we want to reach – 140,000 psi.

Computers possess an enormous efficiency to compare conditions. A programmer will easily know how to use a DO/LOOP WHILE loop to make this happen. However, for the rest of us, here's what happens. DO these instructions:

1. Remove 0.001 in. (or any increment you choose) from the current wire size.

2. Do all those calculations again!

3. Is the calculated corrected stress LESS than the stress level I want?

a) IF calculated stress is LESS, then go back to Step 1, and DO again

b) IF calculated stress is MORE, then go to Step 4

4) Show all data, including the resulting wire size, because we're done with this design

If you were to actually do the example, the resulting wire size would be 0.296 in. because, at that size, the stress will be *just* over 140,000 psi. True or not? Let's test it.

d = 0.296, Therefore:

D = 2.200 in. - 0.296 in. = **1.904 in.**

I.D. = 1.904 in. - 0.296 in. = **1.608 in.**

C = 1.904/0.296 = **6.43**

R = the same, **270.833 lb./in**

$$\mathrm{Na} = \frac{11{,}500{,}000 \times 0.296^4}{8 \times 270.833} \times 1.904^3 = \mathbf{5.90}\ \mathrm{Nt} = 5.90 + 2$$

$$= \mathbf{7.90}\ \mathrm{SH} = 7.90 \times 0.296 = 2.339\ \mathrm{in.}\ \text{(within the needed max.)}$$

FL = the same, at 4.580 in. since no loads or deflections changed

Psh = 270.833 × (4.580 - 2.339) = **606.86 lb.**

$$\mathrm{K} = \frac{(4 \times 6.43) - 1}{(4 \times 6.43) - 4} + \frac{0.615}{6.43} = \mathbf{1.234}$$

$$\mathrm{Ssh\ cor} = \frac{8 \times 606.86 \times 1.904}{\mathrm{Pi} \times 0.296^3} \times 1.234 = 140{,}003\ \mathrm{psi.}$$

This yields a spring with very good parameters, as far as a spring designs go. You could experiment by raising the stress level and do "what ifs." Most importantly, you get a design instantly. Now that you have the wire size, you can check to see if you have a stock size close to the calculated size and do a detailed design with all the data you need.

Here's the absolute best part: Once you get the outside spring nailed, you can have another design using the same calculations for the second spring. . . or a third. . . or fourth. With nested springs, you can define an interference limit. For instance, if you need to be sure on this design that there is a .050 in. space between the two springs, then subtract that amount from the outside spring I.D., and let the program instantly calculate your next spring. Or, if you design valve springs, which often must have interference so the inner spring is actually a bit bigger, you can define a negative interference, and the program will *add* that amount to the outer spring I.D., and the rest is history.

"Iterative" simply means the program goes back and recalculates as many times as needed to get a result. The "logic" part is merely the path that it takes, one parameter depending on the next for a result. If the calculations aren't in the proper logical order, you're pretty much stopped cold. The many variations of this type of program are beyond this article. However, the concept is completely open to creative changes. For example, it would be an easy task for a programmer to include a check that stops the program when the maximum solid height is reached. . . or feedback a myriad of information from the already calculated parameters. For valve spring designers, natural frequency would be nice to have. On the other hand, what about having the computer search out only stock wire sizes and skip the "close to" result?

The bottom line is that computers eat math for lunch, and this is a good example of how they can be put to work when hundreds or thousands of comparative iterations are required. You can get a result in seconds and remove the guesswork!

Randy DeFord is engineering manager at Mid-West Spring and Stamping in Mentone, IN. Readers may contact him by phone at (574) 353-7611, ext. 231. ❖